VERITAS Storage Foundation™ 4.1 *for DB2*

Administrator's Guide

Solaris

N14625F

March 2005

Contents

Preface

VERITAS Storage Foundation *for DB2*, formerly known as VERITAS Database Edition *for DB2*, is an integrated set of system software enhancements and configuration guidelines that combine to help DB2 database administrators configure a database system with high performance, availability, manageability, and reliability.

What's in This Guide?

This guide is organized as follows:

Chapters in This Guide

Chapter	Description
Chapter 1. "VERITAS Storage Foundation for DB2" on page 1	Introduces the features and characteristics of VERITAS Storage Foundation *for DB2*.
Chapter 2. "Setting Up Databases" on page 29	Discusses how to select volume layouts and create optimal file system and database configurations.
Chapter 3. "Using VERITAS Quick I/O" on page 63	Describes how to set up and use VERITAS Quick I/O.
Chapter 4. "Using VERITAS Cached Quick I/O" on page 89	Describes how to set up and use VERITAS Cached Quick I/O.
Chapter 5. "Using Storage Mapping" on page 103	Discusses how to use topology mapping with VERITAS Storage Foundation *for DB2* to map containers to logical volumes and physical devices.
Chapter 6. "Converting Existing Database Configurations to VxFS" on page 115	Discusses how to convert existing databases to VxFS file systems.

Chapters in This Guide

Chapter	Description
Chapter 7. "Using Storage Checkpoints and Storage Rollback" on page 121	Discusses how to use Storage Checkpoint and Storage Rollback.
Chapter 8. "Using Database FlashSnap for Backup and Off-Host Processing" on page 151	Describes the online snapshot facilities provided with VxVM.
Chapter 9. "Using VERITAS NetBackup for Database Backup" on page 215	Describes the online backup and recovery facilities that VERITAS NetBackup provides.
Chapter 10. "Using the VERITAS Storage Foundation for DB2 Graphical User Interface" on page 221	Describes various DB2 database administrative operations that can be easily performed using the graphical user interface (GUI).
Chapter 11. "Using the VxDBA Utility" on page 291	Describes various DB2 database administrative operations that can be easily performed using the VxDBA utility.
Chapter 12. "Tuning for Performance" on page 329	Provides tuning tips and describes commands you can use to monitor and tune database performance.
Appendix A. "VERITAS Storage Foundation for DB2 Command Line Interface" on page 353	Describes the command line interface to key operations also provided in the VxDBA utility.
Appendix B. "Using Third-Party Software to Back Up Files" on page 417	Describes special consideration and tips for using popular third-party software to back up VERITAS Quick I/O files.
Appendix C. "VERITAS Database FlashSnap Status Information" on page 419	Describes the various snapshot and snapshot database status messages you will see in the VERITAS Database FlashSnap CLI and GUI.
Appendix D. "Accessibility and VERITAS Storage Foundation for DB2" on page 429	Provides information relating to Storage Foundation *for DB2*'s conformance with Section 508 of the Rehabilitation Act.
"Glossary" on page 441	Provides terminology and general background information for DB2 database administration.

How to Use This Guide

This guide is intended for database and system administrators responsible for configuring and maintaining DB2 databases with VERITAS Storage Foundation *for DB2*, which includes:

- VERITAS Volume Manager (VxVM)
- VERITAS File System (VxFS) with Quick I/O
- VERITAS Storage Foundation Graphical User Interface
- VERITAS VxDBA Menu Utility
- VERITAS Enterprise Administrator (VEA)
- VERITAS Database FlashSnap (included with the Enterprise Edition)
- VERITAS Storage Mapping (included with the Enterprise Edition)

This guide assumes that the administrator has a:

- Basic understanding of system and database administration
- Working knowledge of the operating system
- General understanding of file systems

Getting Help

VERITAS offers you a variety of support options.

Accessing the VERITAS Support Web Site

The VERITAS Support Web site allows you to:

- contact the VERITAS Support staff and post questions to them
- get the latest patches, upgrades, and utilities
- view the VERITAS Storage Foundation *for DB2* Frequently Asked Questions (FAQ) pages
- search the knowledge base for answers to technical support questions
- receive automatic notice of product updates
- find out about VERITAS Storage Foundation *for DB2* training
- read current white papers related to VERITAS Storage Foundation *for DB2*

The address for the VERITAS Support Web site is:

- http://support.veritas.com

Subscribing to VERITAS Email Notification Service

Subscribe to the VERITAS Email notification service to be informed of software alerts, newly published documentation, Beta programs, and other services.

Go to http://support.veritas.com. Select a product and click "E-mail Notifications" on the right side of the page. Your customer profile ensures you receive the latest VERITAS technical information pertaining to your specific interests.

Accessing VERITAS Telephone and Fax Support

Telephone support for VERITAS Storage Foundation *for DB2* is only available with a valid support contract. To contact VERITAS for technical support, dial the appropriate phone number listed on the Support Guide included in the product box and have your product license information ready for quick navigation to the proper support group.

The address for the VERITAS telephone support directory is:

- http://support.veritas.com

Contacting VERITAS Licensing

For license information call 1-800-634-4747 option 3, fax 1-650-527-0952, or e-mail amercustomercare@veritas.com.

VERITAS Storage Foundation for DB2 Guides

The following guides, along with the online help, comprise the VERITAS Storage Foundation *for DB2* documentation set:

Guides in VERITAS Storage Foundation for DB2 Documentation Set

Guide Title	**Filename**
VERITAS Storage Foundation Release Notes	`sf_notes.pdf`
VERITAS Storage Foundation Installation Guide	`sf_install.pdf`
VERITAS Storage Foundation for DB2 Administrator's Guide	`sf_db2_admin.pdf`

Guides in VERITAS Storage Foundation for DB2 Documentation Set (continued)

Guide Title	**Filename**
VERITAS Storage Foundation Intelligent Storage Provisioning Administrator's Guide	`sf_isp_admin.pdf`
VERITAS Storage Foundation Cross-Platform Data Sharing Administrator's Guide	`sf_cds_admin.pdf`
VERITAS Array Integration Layer Configuration Guide	`vail_config.pdf`
VERITAS File System Administrator's Guide	`vxfs_admin.pdf`
VERITAS File System Programmer's Reference Guide	`vxfs_ref.pdf`
VERITAS Volume Manager Administrator's Guide	`vxvm_admin.pdf`
VERITAS Enterprise Administrator (VEA 500 Series) Getting Started	`vea5x_getting_started.pdf`
VERITAS FlashSnap Point-In-Time Copy Solutions Administrator's Guide	`flashsnap_admin.pdf`
VERITAS Volume Manager Hardware Notes	`vxvm_hwnotes.pdf`
VERITAS Volume Manager Troubleshooting Guide	`vxvm_tshoot.pdf`

Related Resources

While not shipped with VERITAS Storage Foundation, the following documents provide related information if you plan to use VERITAS NetBackup™ to back up your databases:

- The *VERITAS NetBackup Release Notes* provide important, up-to-date, and release-specific information for VERITAS NetBackup. Reading all of the *Release Notes* before installing or using any VERITAS products is recommended.
- The *VERITAS NetBackup BusinessServer Getting Started Guide* explains how to install and configure VERITAS NetBackup.
- The *VERITAS NetBackup User's Guide* explains how to use VERITAS NetBackup to back up, archive, and restore files and directories.

- The *VERITAS NetBackup System Administrator's Guide for UNIX, Volume I* and *VERITAS NetBackup System Administrator's Guide for UNIX, Volume II* describe how to configure and manage the operation of VERITAS NetBackup.
- The *VERITAS NetBackup BusinesServer Media Manager System Administrator's Guide* and *VERITAS NetBackup DataCenter Media Manager System Administrator's Guide* describe how to use the extensive media management capabilities of VERITAS NetBackup.

Conventions

Typographical and Symbolic

The following tables explain the typographical and symbolic conventions used throughout the guides:

Typeface Conventions

Typeface	Usage	Examples
`monospace`	Computer output, files, directories, software elements such as command options, function names, and parameters	Read tunables from the `/etc/vx/tunefstab` file. See the `ls`(1) manual page for more information.
`monospace` (**`bold`**)	User input	# **`mount -F vxfs /h/filesys`**
italic	New terms, book titles, emphasis, variables replaced with a name or value	See the *User's Guide* for details. The variable *ncsize* determines the value of...

Symbolic Conventions

Symbol	Usage	Examples
`%`	C shell prompt	
`$`	Bourne/Korn shell prompt	

Symbolic Conventions

Symbol	Usage	Examples
#	Superuser prompt (all shells)	
`db2 =>`	DB2 command processor prompt	`db2 =>` **`list database directory`**
\	Continued input on the following line; you do not type this character	`#` **`mkfs -F vxfs -o largefiles`** `\` **`/dev/vx/rdsk/PRODdg/db01`**
[ ]	In a command synopsis, brackets indicates an optional argument	`ls [ -a ]`
|	In a command synopsis, a vertical bar separates mutually exclusive arguments	`mount [ suid | nosuid ]`
blue text	In PDF and HTML files, click on these active hyperlinks to move to the specified location	See "Using Snapshots for Database Backup" on page 97 for more information.

Notes and Cautions

Note A Note provides information that makes it easier to use the product or helps you avoid problems.

Caution A Caution warns you about situations that can cause data loss.

Comment on the Documentation

Let us know what you like and dislike about the documentation. Were you able to find the information you needed quickly? Was the information clearly presented? Report errors and omissions, or tell us what you would find useful in future versions of our manuals and online help.

Please include the following information with your comment:

- The title and product version of the manual you are commenting on
- The topic (if relevant) you are commenting on
- Your comment
- Your name

Email your comment to the following address:

- `sfuadocs@veritas.com`

Please only use this address to comment on product documentation. See "Getting Help" on page xvii for how to contact Technical Support about our software.

We appreciate your feedback.

VERITAS Storage Foundation for DB2 1

This chapter provides an overview of the features, components, and database-related functionality of VERITAS Storage Foundation *for DB2*.

Topics include:

VERITAS Storage Foundation for DB2

Standard Edition/Enterprise Edition

There are two versions of this product:

- VERITAS Storage Foundation *for DB2* Standard Edition
- VERITAS Storage Foundation *for DB2* Enterprise Edition

  The Enterprise Edition contains everything in the Standard Edition plus Storage Mapping, Database FlashSnap, Storage Checkpoints, and Storage Rollback.

Note VERITAS Storage Foundation/High Availability (HA) *for DB2* is available only with the Enterprise Edition.

Unless otherwise noted, features pertain to both the Standard and Enterprise Edition products.

Features

VERITAS Storage Foundation *for DB2* combines the strengths of the core VERITAS technology products with database-specific enhancements to offer performance, availability, and manageability for database servers.

VERITAS Storage Foundation *for DB2* includes the following products:

- VERITAS Volume Manager (VxVM)

  A disk management subsystem that supports disk striping, disk mirroring, and simplified disk management for improved data availability and performance.

- VERITAS Database FlashSnap

  Database FlashSnap, a feature of the Enterprise Edition, lets you create, resynchronize, and reverse resynchronize volume snapshots for databases. The snapshots can be used on a second host. Also, database administrators can perform these tasks without `root` privileges. Database FlashSnap tasks may be performed through the VERITAS Storage Foundation *for DB2* GUI or the command line interface.

- VERITAS Storage Mapping

  Storage Mapping, a feature of the Enterprise Edition, lets you take full advantage of DB2 storage mapping to map containers to physical devices and display storage object I/O statistics. Both storage object I/O statistics and the storage structure can be displayed using either the `vxstorage_stats` command or the VERITAS Storage Foundation *for DB2* GUI.

- VERITAS File System (VxFS)

  A high-performance, fast-recovery file system that is optimized for business-critical database applications and data-intensive workloads. VxFS offers online administration, letting you perform most frequently scheduled maintenance tasks (including online backup, resizing, and file system changes) without interrupting data or system availability. VxFS also provides support for large file systems (up to 8 exabytes.)

  VERITAS File System offers performance-enhancing features that are of particular interest in a database environment:

  - VERITAS Quick I/O is a VxFS feature that improves the throughput for DB2 databases built on VERITAS File Systems. Quick I/O delivers raw device performance to databases run on VxFS, providing the administrative advantages of using file systems without the performance penalties.
  - VERITAS Cached Quick I/O further enhances database performance by leveraging large system memory to selectively buffer the frequently accessed data.
  - A feature of the Enterprise Edition, VxFS Storage Checkpoint technology lets you create a point-in-time image of a file system. Storage Checkpoints are treated like any other VxFS file system and can be created, mounted, unmounted, and removed with VxFS and VERITAS Storage Foundation *for DB2* administrative utilities.

- VERITAS VxDBA Menu Utility

  The VxDBA menu-driven utility allows you to perform various administrative tasks, including database monitoring.

  You can also use the graphical user interface or command line interface to perform these tasks.

- VERITAS Enterprise Administrator

  VERITAS Enterprise Administrator (VEA) is the infrastructure that allows you to access VERITAS Storage Foundation *for DB2*, VERITAS Volume Manager, and VERITAS File System information and features through the GUI.

An optional High Availability (HA) version of VERITAS Storage Foundation *for DB2* Enterprise Edition, which includes VERITAS Cluster Server, is available for customers who have high system-availability requirements.

The following is an example of the VERITAS Storage Foundation *for DB2* storage stack:

DB2	
VxFS	Quick I/O
VxVM	
Disks	RAID

VERITAS Volume Manager

Databases require their storage media to be robust and resilient to failure. It is vital to protect against hardware and disk failures and to maximize performance using all the available hardware resources. Using a volume manager provides this necessary resilience and eases the task of management. A volume manager can help you manage hundreds of disk devices and makes spanning, striping, and mirroring easy.

VERITAS Volume Manager (VxVM) builds virtual devices called *volumes* on top of physical disks. Volumes are accessed by a file system, a database, or other applications in the same way physical disk partitions would be accessed. Using volumes, VxVM provides the following administrative benefits for databases:

- Spanning of multiple disks—eliminates media size limitations.
- Striping—increases throughput and bandwidth.
- Mirroring or RAID-5—increases data availability.
- Online relayout—allows online volume layout changes to improve database performance.
- Volume resynchronization—ensures that all mirrors contain exactly the same data and that the data and parity in RAID-5 volumes agree.
- Dirty Region Logging (DRL)—speeds the recovery of mirrored volumes after a system crash.
- Volume snapshots—allows backup of volumes based on disk mirroring. With this release, VxVM introduces full-sized and space-optimized instant snapshots, the preferred way to implement online and off-host point-in-time copy solutions.
- FastResync—separately licensed, optional feature that performs quick and efficient resynchronization of stale mirrors. FastResync is included with the Enterprise Edition and is also included as part of the VERITAS FlashSnap option with the Standard Edition.
- Disk group split and join—separately licensed, optional feature that supports general disk group reorganization and allows you to move volume snapshots to another host for off-host backup. Disk group split and join is included with the Enterprise Edition and is also included as part of the VERITAS FlashSnap option with the Standard Edition.
- Hot-relocation—automatically restores data redundancy in mirrored and RAID-5 volumes when a disk fails.
- Dynamic multipathing (DMP)—allows for transparent failover, load sharing, and hot plugging of SCSI devices.
- Volume sets—allows several volumes to be represented by a single logical mount device.

- Dynamic LUN Expansion—allows you to resize a disk after it has been initialized while preserving the existing data on the disk.
- Storage Expert—helps diagnose configuration problems with VxVM.
- Cluster Volume Manager (CVM)—separately licensed, optional feature that allows you to use VxVM in a cluster environment.
- VERITAS Volume Replicator (VVR)—separately licensed, optional feature that provides data replication for disaster recovery planning.
- Free space pool management—simplifies administration and provides flexible use of available hardware.
- Online administration—allows configuration changes without system or database down time.

The following sections provide brief overviews of VxVM concepts and features that are relevant to database administration. The information and examples presented in the remainder of this guide assume that you are using VERITAS Volume Manager. For a more detailed description of VxVM and its features, refer to the *VERITAS Volume Manager Administrator's Guide.*

Volumes

A *volume* is a virtual disk device that appears to applications, databases, and file systems like a physical disk partition without the physical limitations of a disk partition. A volume consists of one or more plexes, each holding a copy of the selected data in the volume. Due to its virtual nature, a volume is not restricted to a particular disk or a specific area of a disk. For example, a volume can span multiple disks and can be used to create a large file system.

Volumes consist of other virtual objects that can be manipulated to change the volume's configuration. Volumes and their virtual components are referred to as *Volume Manager objects*. You can manipulate VERITAS Volume Manager objects in a variety of ways to optimize performance, provide redundancy of data, and perform backups or other administrative tasks on one or more physical disks without interrupting applications. As a result, data availability and disk subsystem throughput are improved.

You can change the configuration of a volume without causing disruption to databases or file systems that are using the volume. For example, you can mirror a volume on separate disks or move the volume to use different disk storage.

Disk Groups

A *disk group* is a collection of disks that share a common configuration (for example, configuration objects that belong to a single database). We recommend creating one disk group for each database.

You can move a disk group and its components as a unit from one host to another host. For example, you can move volumes and file systems that belong to the same database and are created within one disk group as a unit. You must configure a given volume from disks belonging to one disk group.

In releases before VERITAS Storage Foundation 4.0 *for DB2*, the default disk group was `rootdg`. For VxVM to function, the rootdg disk group had to exist and it had to contain at least one disk. This requirement no longer exists, and VxVM can work without any disk groups configured (although you must set up at least one disk group before you can create any volumes of other VxVM objects).

Volume Layouts

A *Redundant Array of Independent Disks* (RAID) is a disk array in which a group of disks appears to the system as a single virtual disk or a single volume. VxVM supports several RAID implementations, as well as spanning. The following volume layouts are available to satisfy different database configuration requirements:

- Spanning and concatenation
- Striping (RAID-0)
- Mirroring (RAID-1)
- Mirrored-Stripe Volumes (RAID-0+1)
- Striped-Mirror Volumes (RAID-1+0)
- RAID-5

Caution Spanning or striping a volume across multiple disks increases the chance that a disk failure will result in failure of that volume. Use mirroring or RAID-5 to substantially reduce the chance of a single volume failure caused by a single disk failure.

Spanning and Concatenation

Concatenation maps data in a linear manner onto one or more subdisks in a plex. To access all of the data in a concatenated plex sequentially, data is first accessed in the first subdisk from beginning to end. Data is then accessed in the remaining subdisks sequentially from beginning to end, until the end of the last subdisk.

The subdisks in a concatenated plex do not have to be physically contiguous and can belong to more than one VM disk. Concatenation using subdisks that reside on more than one VM disk is called *spanning*.

Spanning is useful when you need to read or write data sequentially (for example, reading from or writing to database logs) and there is not sufficient contiguous space.

Striping (RAID-0)

Striping is a technique of mapping data so that the data is interleaved among multiple physical disks. Data is allocated in equal-sized units (called *stripe units*) that are interleaved between the disks. Each stripe unit is a set of contiguous blocks on a disk. A *stripe* consists of the set of stripe units at the same position across all columns. A column is a set of one or more subdisks within a striped plex.

Striping is useful if you need large amounts of data written to or read from physical disks, and performance is important. Striping is also helpful in balancing the I/O load from multi-user applications across multiple disks. By using parallel data transfer to and from multiple disks, striping significantly improves data-access performance.

When striping across multiple disks, failure of any one disk will make the entire volume unusable.

Mirroring (RAID-1)

Mirroring is a technique of using multiple copies of the data, or mirrors, to duplicate the information contained in a volume. In the event of a physical disk failure, the mirror on the failed disk becomes unavailable, but the system continues to operate using the unaffected mirrors. For this reason, mirroring increases system reliability and availability. A volume requires at least two mirrors to provide redundancy of data. A volume can consist of up to 32 mirrors. Each of these mirrors must contain disk space from different disks for the redundancy to be effective.

Striping Plus Mirroring (Mirrored-Stripe or RAID-0+1)

VxVM supports the combination of mirroring with striping. When used together on the same volume, mirroring plus striping offers the benefits of spreading data across multiple disks while providing redundancy of data.

Mirrored-stripe volumes have multiple plexes as mirrors, each constructed as a striped plex. Allocate subdisks used in the same striped plex from separate disks, and use a disk in only one mirror of a volume.

Mirroring Plus Striping (Striped-Mirror Volumes, RAID-1+0 or RAID-10)

VxVM supports the combination of striping with mirroring. When used together on the same volume, striping plus mirroring offers the benefits of spreading data across multiple disks while providing redundancy of data.

Striped-mirror volumes combine striping and mirroring, but the mirroring is done at stripe column level. In case of failure, this type of volume recovers faster than RAID-0+1 volumes and the tolerance for disk failure is greater.

For databases that support online transaction processing (OLTP) workloads, we recommend either mirrored-stripe or striped-mirror volumes to improve database performance and reliability. For highest availability, we recommend striped-mirror volumes (RAID 1+0).

RAID-5 (Striping with Parity)

RAID-5 provides data redundancy through the use of *parity* (a calculated value that the system uses to reconstruct data after a failure). While data is written to a RAID-5 volume, parity is also calculated by performing an *exclusive OR* (XOR) procedure on data. The resulting parity is then written to another part of the volume. If a portion of a RAID-5 volume fails, the data that was on that portion of the failed volume can be recreated from the remaining data and the parity.

RAID-5 offers data redundancy similar to mirroring, while requiring less disk space. RAID-5 read performance is similar to that of striping but with relatively slow write performance. RAID-5 is useful if the database workload is read-intensive (as in many data warehousing applications). You can snapshot a RAID-5 volume and move a RAID-5 subdisk without losing redundancy.

Online Relayout

As databases grow and usage patterns change, online relayout lets you change volumes to a different layout, with uninterrupted data access. Relayout is accomplished online and in place. Use online relayout to change the redundancy or performance characteristics of the storage, such as data organization (RAID levels), the number of columns for RAID-5 and striped volumes, and stripe unit size.

Volume Resynchronization

When storing data redundantly, using mirrored or RAID-5 volumes, VERITAS Volume Manager ensures that all copies of the data match exactly. However, if the system crashes, small amounts of the redundant data on a volume can become inconsistent or *unsynchronized*. For mirrored volumes, unsynchronized data can cause two reads from the same region of the volume to return different results if different mirrors are used to satisfy the read request. In the case of RAID-5 volumes, unsynchronized data can lead to parity corruption and incorrect data reconstruction.

In the event of a system crash, VERITAS Volume Manager ensures that all mirrors contain exactly the same data and that the data and parity in RAID-5 volumes agree. This process is called *volume resynchronization*. Not all volumes require resynchronization after a system failure. VxVM notices when a volume is first written and marks it as *dirty*. Only volumes that are marked dirty when the system reboots require resynchronization.

The process of resynchronization can impact system and database performance. However, it does not affect the availability of the database after system reboot. You can immediately access the database after database recovery although the performance may suffer due to resynchronization. For very large volumes or for a very large number of volumes, the resynchronization process can take a long time. You can significantly reduce resynchronization time by using Dirty Region Logging (DRL) for mirrored volumes or by making sure that RAID-5 volumes have valid RAID-5 logs. However, using logs can slightly reduce the database write performance.

For most database configurations, we recommend using dirty region logs or the RAID-5 logs when mirrored or RAID-5 volumes are used. It is also advisable to evaluate the database performance requirements to determine the optimal volume configurations for the databases.

Dirty Region Logging

Dirty Region Logging (DRL), if enabled, speeds the recovery of mirrored volumes after a system crash. DRL keeps track of the regions that have changed due to I/O writes to a mirrored volume. DRL uses this information to recover only those portions of the volume that need to be recovered.

Note In VxVM 4.1, if a version 20 data change object (DCO) volume is associated with a volume, a portion of the DCO volume can be used to store the DRL log. There is no need to create a separate DRL log for a volume that has a version 20 DCO volume.

Volume Sets

Volume sets are an enhancement to VxVM that allow several volumes to be represented by a single logical mount device. 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-device 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.

Volume Snapshots

A volume snapshot is a point-in-time image of a volume. VERITAS Volume Manager provides three volume snapshot features based on disk mirroring:

- Full-sized instant snapshots
- Space-optimized instant snapshots
- Emulation of third-mirror snapshots

For detailed information on performing database backups using volume snapshots, see "Using Database FlashSnap for Backup and Off-Host Processing" on page 151

VERITAS FastResync (Optional)

VERITAS FastResync (previously called Fast Mirror Resynchronization or FMR) is included with the Enterprise Edition. It is also included as part of the VERITAS FlashSnap option with the Standard Edition.

VERITAS FastResync performs quick and efficient resynchronization of stale mirrors (mirrors that are not synchronized). This increases the efficiency of the VxVM snapshot mechanism, and improves the performance of operations such as backup and decision support. Typically, these operations require that the volume is quiescent, and that they are not impeded by updates to the volume by other activities on the system. To achieve these goals, the snapshot mechanism in VxVM creates an exact copy of a primary volume at an instant in time. After a snapshot is taken, it can be accessed independently of the volume from which it was taken.

VERITAS Storage Foundation *for DB2* Enterprise Edition includes a feature called Database FlashSnap, which takes advantage of the FastResync and disk group split and join features. Database FlashSnap provides a quicker and easier way for database administrators to use volume snapshots. For more information, see "VERITAS Database FlashSnap" on page 23.

Non-Persistent FastResync

Non-persistent FastResync allocates its change maps in memory. If non-persistent FastResync is enabled, a separate FastResync map is kept for the original volume and for each snapshot volume. Unlike a dirty region log (DRL), these maps do not reside on disk nor in persistent store. The advantage is that updates to the FastResync map have little impact on I/O performance, as no disk updates need to be performed. However, if a system is rebooted, the information in the map is lost, so a full resynchronization is required when performing a `snapback` operation. This limitation can be overcome for volumes in cluster-shareable disk groups, provided that at least one of the nodes in the cluster remains running to preserve the FastResync map in its memory.

Persistent FastResync

Non-persistent FastResync has been augmented by the introduction of persistent FastResync. Unlike non-persistent FastResync, Persistent FastResync keeps the FastResync maps on disk so that they can survive system reboots and system crashes. When the disk groups are rejoined, this allows the snapshot plexes to be quickly resynchronized. This ability is not supported by non-persistent FastResync.

If persistent FastResync is enabled on a volume or on a snapshot volume, a DCO and a *DCO log volume* are associated with the volume.

In VxVM 3.2 and 3.5, the DCO object only managed information about the FastResync maps. These maps track writes to the original volume (and to each of up to 32 snapshot volumes) since the last `snapshot` operation. The DCO log volume on disk holds the 33 maps, each of which is 4 blocks in size by default.

In VxVM 4.1, the DCO object is used not only to manage FastResync maps, but also to manage DRL recovery maps and special maps called copy maps that allow instant snapshot operations to be resume following a system crash.

Persistent FastResync can also track the association between volumes and their snapshot volumes after they are moved into different disk groups. When the disk groups are rejoined, this allows the snapshot plexes to be quickly resynchronized. This ability is not supported by non-persistent FastResync.

Disk Group Split and Join (Optional)

Disk group split and join is included with the Enterprise Edition. It is also included as part of the VERITAS FlashSnap option with the Standard Edition.

VxVM provides a disk group content reorganization feature that supports general disk group reorganization and allows you to move volume snapshots to another host for off-host backup. Additional options to the `vxdg` command enable you to take advantage of the ability to remove all VxVM objects from an imported disk group and move them to a newly created target disk group (`split`), and to remove all VxVM objects from an imported disk group and move them to an imported target disk group (`join`). The move operation enables you to move a self-contained set of VxVM objects between the imported disk groups.

Hot-Relocation

In addition to providing volume layouts that help improve database performance and availability, VxVM offers features that you can use to further improve system availability in the event of a disk failure. *Hot-relocation* is a feature that allows a system to react automatically to I/O failures on mirrored or RAID-5 volumes and restore redundancy and access to those volumes.

VxVM detects I/O failures on volumes and relocates the affected portions to disks designated as *spare disks* or free space within the disk group. VxVM then reconstructs the volumes that existed before the failure and makes them redundant and accessible again.

The hot-relocation feature is enabled by default and is recommended for most database configurations. After hot-relocation occurs, we recommend verifying the volume configuration for any possible performance impact. It is also a good idea to designate additional disks as spares to augment the spare pool.

While a disk is designated as a spare, you cannot use the space on that disk for the creation of VxVM objects within its disk group. VxVM also lets you free a spare disk for general use by removing it from the pool of hot-relocation disks.

DMP-Supported Disk Arrays

VxVM provides administrative utilities and driver support for disk arrays that can take advantage of its Dynamic Multipathing (DMP) feature. Some disk arrays provide multiple ports to access their disk devices. These ports, coupled with the host bus adaptor (HBA) controller and any data bus or I/O processor local to the array, make up multiple hardware paths to access the disk devices. Such disk arrays are called multipathed disk arrays. This type of disk array can be connected to host systems in many different configurations, (such as multiple ports connected to different controllers on a single host, chaining of the ports through a single controller on a host, or ports connected to different hosts simultaneously). DMP is available for multiported disk arrays from various vendors and provides improved reliability and performance by using path failover and load balancing.

See the *VERITAS Volume Manager Administrator's Guide* for detailed information and the *VERITAS Volume Manager Hardware Notes* for information about supported disk arrays.

Dynamic LUN Expansion

Dynamic LUN expansion allows you to resize a disk after it has been initialized while preserving the existing data on the disk. See the *VERITAS Volume Manager Administrator's Guide* for more information.

Storage Expert

Storage Expert consists of a set of simple commands that collect VxVM configuration data and compare it with "best practice." Storage Expert then produces a summary report that shows which objects do not meet these criteria and makes recommendations for VxVM configuration improvements.

These user-configurable tools help you as an administrator to verify and validate systems and non-optimal configurations in both small and large VxVM installations.

Storage Expert components include a set of rule scripts and a rules engine. The rules engine runs the scripts and produces ASCII output, which is organized and archived by Storage Expert's report generator. This output contains information about areas of VxVM configuration that do not meet the set criteria. By default, output is sent to the screen, but you can redirect it to a file using standard UNIX redirection. See the *VERITAS Volume Manager Administrator's Guide* for more information.

Cluster Functionality (Optional)

VxVM includes an optional, separately licensable clustering feature, known as Cluster Volume Manager, that enables VxVM to be used in a cluster environment. With the clustering option, VxVM supports up to 16 nodes per cluster. See the *VERITAS Volume Manager Administrator's Guide* for more information.

VERITAS Volume Replicator (Optional)

VERITAS Volume Replicator (VVR) is an optional, separately licensable feature of VxVM. VVR is a data replication tool designed to maintain a consistent copy of application data at a remote site. It is built to contribute to an effective disaster recovery plan. If the data center is destroyed, the application data is immediately available at the remote site, and the application can be restarted at the remote site.

VVR 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 can use existing VxVM configurations with some restrictions. Any application, even with existing data, can be configured to use VVR transparently.

See the VERITAS Volume Replicator documentation for more information.

VERITAS File System

VERITAS File System (referred to as VxFS) is an extent-based, intent logging file system intended for use in UNIX environments that deal with large volumes of data and that require high file system performance, availability, and manageability. VxFS also provides enhancements that make file systems more viable in database environments.

The following sections provide a brief overview of VxFS concepts and features that are relevant to database administration. For a more detailed description of VxFS and its complete feature set, see the *VERITAS File System Administrator's Guide*.

VERITAS Quick I/O

Databases can run on either file systems or raw devices. Database administrators often create their databases on file systems because it makes common administrative tasks (such as moving, copying, and backing up) easier. However, running databases on most file systems significantly reduces database performance.

When performance is an issue, database administrators create their databases on raw devices. VxFS with Quick I/O presents regular, preallocated files as raw character devices to the application. Using Quick I/O, you can enjoy the management advantages of databases created on file systems and achieve the same performance as databases created on raw devices. See "Using VERITAS Quick I/O" on page 63 for more information.

VERITAS Cached Quick I/O

Cached Quick I/O allows databases to make more efficient use of large system memory while still maintaining the performance benefits of Quick I/O. Cached Quick I/O provides an efficient, selective buffering mechanism to back asynchronous I/O. Using Cached Quick I/O, you can enjoy all the benefits of Quick I/O and achieve even better performance.

Cached Quick I/O is first enabled for the file system and then enabled on a per file basis. See "Using VERITAS Cached Quick I/O" on page 89 for more information.

Extent-Based Allocation

The UFS file system supplied with Solaris uses block-based allocation schemes that provide good random access to files and acceptable latency on small files. For larger files, such as database files, this block-based architecture limits throughput. This limitation makes the UFS file system a less than optimal choice for database environments.

The VxFS file system addresses this performance issue by allocating storage in groups of extents rather than a block at a time. An *extent* is one or more adjacent blocks of data within the file system. An extent is presented as an *address-length* pair that identifies the starting block address and the length of the extent (in file system or logical blocks). When storage is allocated to a file on a VxFS file system, it is grouped in extents, as opposed to being allocated a block at a time as with the UFS file system.

By allocating disk space to files in extents, disk I/O to and from a file can be done in units of multiple blocks. This type of I/O can occur if storage is allocated in units of consecutive 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 of multiple blocks.

The VxFS file system allocates disk space to files in groups of one or more extents. VxFS also allows applications to control some aspects of the extent allocation for a given file. *Extent attributes* are the extent allocation policies associated with a file.

For information on how to create preallocated database files using extent attributes, see "Preallocating Space for Quick I/O Files Using the setext Command" on page 70.

Fast File System and Database Recovery

VERITAS File System begins recovery procedures within seconds after a system failure by using a tracking feature called *intent logging*. 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. 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 completing a full structural check of the entire file system. Replaying the intent log may not completely recover the damaged file system structure if there was a disk hardware failure; hardware problems may require a complete system check using the `fsck` utility provided with VERITAS File System.

Online System Administration

The VxFS file system provides online system administration utilities to help resolve certain problems that impact database performance. You can defragment and resize a VxFS file system while it remains online and accessible to users.

Defragmentation Utility

Free resources are originally aligned in the most efficient order possible and are allocated to files in a way that is considered by the system to provide optimal performance. When a file system is active for extended periods of time, new files are created, old files are removed, and existing files grow and shrink. Over time, the original ordering of free resources is lost and the file system tends to spread along the disk, leaving unused gaps or *fragments* between areas that are in use. This process, known as *fragmentation*, leads to degraded performance because the file system has fewer choices when selecting an extent (a group of contiguous data blocks) to assign to a file. You should analyze the degree of fragmentation before creating new database files.

VxFS provides the online administration utility `fsadm` to resolve fragmentation problems. The utility can be run on demand and should be scheduled regularly as a `cron` job.

Resizing Utility

Changes in database size can result in file systems that are too large or too small for the current database. Without special utilities, expanding or shrinking a file system becomes a a matter of stopping applications, offloading the contents of the file system, rebuilding the file system to a new size, and then restoring the data. Data is unavailable to users while these administrative tasks are performed.

The VxFS file system utility `fsadm` provides a mechanism to resize file systems without unmounting them or interrupting users' productivity. Because the VxFS file system can only be mounted on one device, expanding a file system means that the underlying device must also be expandable while the file system is mounted. Working with VxVM, VxFS provides online expansion capability.

Cross-Platform Data Sharing

VERITAS Cross-Platform Data Sharing allows data to be *serially* shared among heterogeneous systems where each system has direct access to the physical devices that hold the data. This feature can be used only in conjunction with VERITAS Volume Manager. See the *VERITAS Storage Foundation Cross-Platform Data Sharing Administrator's Guide* for more information. Shared or parallel access is possible for read-only data.

Quality of Storage Service (Optional)

The Quality of Storage Service (QoSS) feature is included with the Enterprise Edition.

The QoSS option is built on the multi-volume support technology introduced in this release. Using QoSS, you can map more than one device to a single file system. You can then configure policies that automatically relocate files from one device to another, or relocate files by running file relocation commands. Having multiple devices lets you determine where files are located, which can improve performance for applications that access specific types of files and reduce storage-related costs.

Support for Large File Systems and Large Files (Optional)

Support for large file systems is included with the Enterprise Edition.

In conjunction with VxVM, VxFS can support file systems up to 8 exabytes in size. You have the option of creating a file system using:

- Version 4 disk layout, which supports file systems up to one terabyte. The Version 4 disk layout encompasses all file system structural information in files, rather than at fixed locations on disk, allowing for greater scalability.
- Version 5, which supports file systems up to 32 terabytes. Files can be a maximum of two terabytes. File systems larger than one terabyte must be created on a VERITAS Volume Manager volume.
- Version 6, which supports file systems up to 8 exabytes. The Version 6 disk layout enables features such as multi-device support, Cross-Platform Data Sharing, named data streams, file change log. File systems created on VxFS 4.1 will by default use the Version 6 disk layout. An online conversion utility, `vxupgrade`, is provided to upgrade existing disk layouts to Version 6 on mounted file systems.

For large database configurations, this eliminates the need to use multiple file systems because of the size limitations of the underlying physical devices.

Changes implemented with the VxFS Version 4 disk layout have greatly expanded file system scalability, including support for large files. You can create or mount file systems with or without large files by specifying either the `largefiles` or `nolargefiles` option in `mkfs` or `mount` commands. See "Creating a VxFS File System" on page 47 for more information.

Multi-Volume File System Support

The multi-volume file system (MVS) feature allows several volumes to be represented by a single logical object. All I/O to and from an underlying logical volume is directed by way of *volume sets.* A volume set is a container for multiple different volumes. This feature can be used only in conjunction with VERITAS Volume Manager.

Storage Checkpoint and Storage Rollback

The Storage Checkpoint and Storage Rollback features are included with the Enterprise Edition. With the Standard Edition, they can be purchased as part of the VERITAS FlashSnap option.

VERITAS File System provides a separately licensed *Storage Checkpoint* facility that allows you to create a persistent, point-in-time image of all user files in a file system—the Storage Checkpoint remains even after the file system is unmounted or the system is rebooted. Storage Checkpoints present a view of a file system at a point in time, and subsequently identifies and maintains copies of the original file system blocks. Instead of using a disk-based mirroring method, Storage Checkpoints save disk space and significantly reduce I/O overhead by using the free space pool available to a file system.

The time required to create a Storage Checkpoint is typically only a couple of seconds. After a Storage Checkpoint is created, a consistent database backup image is made and the database can then resume its normal operation. The Storage Rollback facility can then be used for rolling back the file system image to the point in time when the Storage Checkpoints were taken. In addition, Storage Checkpoints also keep track of the block change information that enables incremental database backup at the block level.

Storage Checkpoints are writable, and can be created, mounted, and removed. Performance enhancements in maintaining *data Storage Checkpoints* (Storage Checkpoints that are complete images of the file system) makes using the *Storage Rollback* feature easier and more efficient, therefore more viable for backing up large databases.

Multi-file system Storage Checkpoint creation allows database backups without having to shut down the database.

MVSs provide the ability to create and administer Storage Checkpoint allocation policies. Storage Checkpoint allocation policies specify a list of volumes and the order in which to allocate Storage Checkpoint data to them. These allocation policies can be used to control where a Storage Checkpoint is created, allowing for separating Storage Checkpoint metadata and data onto different volumes. They can also be used to isolate data allocated to a Storage Checkpoint from the primary file system, which can help prevent the Storage Checkpoint from fragmenting space in the primary file system.

For more information on understanding and using Storage Checkpoints, see “Using Storage Checkpoints and Storage Rollback” on page 121.

For more information on using the GUI to manage Storage Checkpoints and Storage Rollback, see “Using the VERITAS Storage Foundation for DB2 Graphical User Interface” on page 221.

Storage Checkpoint File System Restores

Storage Checkpoints can be used by backup and restore applications to restore either individual files or an entire file system. Restoring from Storage Checkpoints can recover data from incorrectly modified files, but typically cannot be used to recover from hardware damage or other file system integrity problems. File restoration can be done using the `fsckpt_restore`(1M) command. See the *VERITAS File System Administrator's Guide* for more information.

Quotas

VxFS supports quotas, which allocate per-user and per-group quotas and limit the use of two principal resources: files and data blocks. You can assign quotas for each of these resources. Each quota consists of two limits for each resource:

- The *hard limit* represents an absolute limit on data blocks or files. A user can never exceed the hard limit under any circumstances.
- The *soft limit* is lower than the hard limit and can be exceeded for a limited amount of time. This allows users to temporarily exceed limits as long as they fall under those limits before the allotted time expires.

You can use quotas to limit the amount of file system space used by Storage Checkpoints. With VERITAS Storage Foundation *for DB2*, you can enable, disable, set, and display quota values for a single file system, for multiple file systems, or for all file systems in a database using the `db2ed_ckptquota` command.

For details on using VxFS quotas, see the *VERITAS File System Administrator's Guide*. For information about `db2ed_ckptquota`, see "VERITAS Storage Foundation for DB2 Command Line Interface" on page 353.

Cluster Functionality (Optional)

File system clustering is an optional, separately licensed feature of VxFS, where one system is configured as a primary server for the file system, and the other members of a cluster are configured as secondaries. All servers access shared disks for file data operations. If the primary server fails, one of the secondary servers takes over the file system operations. See the *VERITAS File System Administrator's Guide* for more information.

VERITAS Storage Mapping

VERITAS Storage Mapping is a feature included with VERITAS Storage Foundation *for DB2* Enterprise Edition.

VERITAS has defined and implemented a library called VERITAS Federated Mapping Service (VxMS) that provides a mapping interface to VxFS file systems, VxVM volumes, and physical disks. With VERITAS Storage Foundation *for DB2*, you can take full advantage of this feature to map containers to physical devices and display storage object I/O statistics. With the `vxstorage_stats` command, you can view the complete I/O topology mapping of containers through intermediate layers like logical volumes down to actual physical devices. You can also use `vxstorage_stats` to view statistics for VxFS file systems, VxVM volumes, and physical devices. This information can be used to determine the exact location of a data block on a disk and to help identify hot spots.

This command can help you avoid I/O contention. For example, you can use the information to avoid backing up two tablespaces that share the same physical disk.

Both storage object statistics and the storage structure are displayed in the VERITAS Storage Foundation *for DB2* GUI.

For more information, see “Using Storage Mapping” on page 103.

VERITAS Database FlashSnap

VERITAS Database FlashSnap is a feature included with VERITAS Storage Foundation *for DB2* Enterprise Edition. It is also a separately licensed option available with VERITAS Storage Foundation *for DB2* Standard Edition.

VERITAS Database FlashSnap offers a flexible and efficient means of managing business-critical data. Database FlashSnap lets you capture an online image of an actively changing database at a given instant, called a point-in-time copy. You can perform system backup, upgrade, or perform 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.

Database FlashSnap takes advantage of the Persistent FastResync and Disk Group Content Reorganization features of VxVM. Database FlashSnap also streamlines database operations. Once configured, the database administrator can create snapshots, resynchronize data, and reverse resynchronize data without involving the system administrator.

VERITAS Storage Foundation *for DB2* provides three commands that can be executed by the database administrator and do not require `root` privileges:

- `db2ed_vmchecksnap`
- `db2ed_vmsnap`
- `db2ed_vmclonedb`

These commands let database administrators take advantage of the VxVM snapshot functionality without having to deal with storage operations in day-to-day database uses. To use Database FlashSnap, you must configure the volumes used by the database according to the guidelines in "Preparing Hosts and Storage for Database FlashSnap" on page 159.

VERITAS VxDBA Menu Utility

The VxDBA menu utility helps you manage the storage used by databases. You can use VxDBA to:

- Display database, tablespace, container, and file system information and manage the database state
- Collect and display statistics on file system and DB2 space usage
- Monitor file system and DB2 tablespace and container space usage and automatically grow the file system as needed
- Examine volumes used by the file systems and overall system configuration
- Start and stop database instances

For more detailed information about the VxDBA menu-driven utility, see "Using the VxDBA Utility" on page 291.

VERITAS Storage Foundation for DB2 Graphical User Interface

An alternative to the command line interface, the VERITAS Storage Foundation *for DB2* GUI allows you to manage the storage used by databases. You can use the GUI to:

- Display database, tablespace, container, and file system information and manage the database state.
- Create, display, mount, unmount, and remove Storage Checkpoints.
- Roll back databases, tablespaces, or containers to Storage Checkpoints.
- Collect and display statistics on file system and DB2 space usage.
- Collect and display storage object I/O statistics and the storage structure.
- Monitor file system and DB2 tablespace and container space usage and automatically grow the file system as needed.
- Examine volumes used by the file systems and overall system configuration.
- Start, stop, or duplicate database instances. You can duplicate a database using Storage Checkpoints or Database FlashSnap.
- Create or shut down a clone database.
- Create, modify, validate, or remove snapplans.
- Create, resynchronize, or reverse resynchronize a snapshot database.

Click the DB2 icon on the object tree to expand the tree view. Every DB2 instance running on the server can be displayed on the screen. An instance can contain any number of databases. Each database is displayed under the instance and is represented by a unique internal DB2 object.

For more detailed information about the GUI, see "Using the VERITAS Storage Foundation for DB2 Graphical User Interface" on page 221.

VERITAS NetBackup (Optional)

VERITAS 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.

VERITAS NetBackup can be configured for DB2 in an Extended Edition (EE) or Extended-Enterprise Edition (EEE) environment. For detailed information and instructions on configuring DB2 for EEE, see "Configuring for a DB2 EEE (DPF) Environment" in the *VERITAS NetBackup for DB2 System Administrator's Guide for UNIX*.

VERITAS NetBackup, while not a shipped component of VERITAS Storage Foundation *for DB2*, can be purchased separately.

VERITAS Storage Foundation/High Availability for DB2 (Optional)

VERITAS Storage Foundation/High Availability (HA) (VCS) *for DB2* lets database administrators integrate multiple servers into high availability database configurations that can significantly reduce the down time of DB2 databases caused by a system hardware or software failure.

In addition to the VERITAS products included in the base VERITAS Storage Foundation *for DB2,* VERITAS Storage Foundation/HA *for DB2* incorporates the following products:

- VERITAS Cluster Server™ (VCS) *for DB2*
- VERITAS Cluster Server (VCS) Enterprise Agent *for DB2*

Note VERITAS Storage Foundation/HA (VCS) *for DB2* is available only for the Enterprise Edition.

Setting Up Databases 2

This chapter describes how to use VERITAS Volume Manager and VERITAS File System to set up optimal system configurations for DB2 Databases.

Topics covered in this chapter include:

Setting Up a New Database

VERITAS Storage Foundation *for DB2* contains a combination of performance, management, and high availability features. This section provides an overview of the steps to follow if you are using VERITAS Storage Foundation *for DB2* to set up a new database.

▼ **To set up a new database using VERITAS Storage Foundation for DB2**

1. Determine the number and sizes of file systems you need for the database you want to create. For detailed information, see the *VERITAS File System Administrator's Guide*.

2. Create volumes to meet your file system needs. You can use disk mirroring as a safeguard against disk failures and striping for better performance. For more information see the "Selecting a Volume Layout" on page 38 and "Creating a Volume" on page 40. For information about performance tuning to improve database performance, see "Tuning for Performance" on page 329.

3. Create the VxFS file systems you need on the volumes. See "File System Creation Guidelines" on page 46 and "Creating a VxFS File System" on page 47.

4. Install and configure your database. For best OLTP performance, use Quick I/O. For information on creating and using Quick I/O files, and converting existing files to Quick I/O, see "Using VERITAS Quick I/O" on page 63.

   If you would like the ability to view detailed storage stack topology information to ensure your storage stack configuration is optimized for the database, configure and use Storage Mapping. For more information, see "Using Storage Mapping" on page 103.

   If you are not currently running on VxVM and VxFS, see the *VERITAS Storage Foundation for DB2 Installation Guide* and "Converting Existing Database Configurations to VxFS" on page 115 for information about how to convert your existing database storage configuration to VERITAS Storage Foundation *for DB2*.

5. For backup and recovery on the same host, you can use the Storage Checkpoint facility to create file system snapshots of the database. A Storage Checkpoint creates an exact image of a database instantly and provides a consistent image of the database from the point in time the Storage Checkpoint was created. For more information about creating and managing Storage Checkpoints, see "Using Storage Checkpoints and Storage Rollback" on page 121

Setting Up a Disk Group

Before creating file systems for a database, set up a disk group for each database. A disk group lets you group disks, volumes, file systems, and files that are relevant to a single database into a logical collection for easy administration. Because you can move a disk group and its components as a unit from one machine to another, you can move an entire database when all the configuration objects of the database are in one disk group. This capability is useful in a failover situation.

Disk Group Configuration Guidelines

Follow these guidelines when setting up disk groups for use with databases:

- 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 DB2 database name specified by the environment variables `$DB2DATABASE` and `$DB2INSTANCE`, and a `dg` suffix. The `dg` suffix helps identify the object as a disk group. Also, each disk name must be unique within the disk group.

Note Users should not share a disk group between different DB2 instances. Although it is not recommended, sharing a disk group among all databases in the same instance may make sense if the instance contains several small databases. In this case, name the disk group using the DB2 instance name specified by the environment variable `$DB2INSTANCE` and a `dg` suffix.

- Never create container files using file systems or volumes that are not in the same disk group.

See "Tuning for Performance" on page 329 for more information.

Note In earlier releases of VERITAS Volume Manager, a system installed with VxVM was configured with a default disk group, `rootdg`, that had to contain at least one disk. VxVM can now function without any disk group having been configured. Only when the first disk is placed under VxVM control must a disk group be configured.

Note Most VxVM commands require superuser or equivalent privileges.

Creating a Disk Group for a Database

You can use the `vxdg` command or the graphical user interface (GUI) to create a new disk group. A disk group must contain at least one disk at the time it is created. You also have the option to create a shared disk group for use in a cluster environment.

Disks must be placed in disk groups before they can be used by VxVM. You can create disk groups to organize your disks into logical sets of disks.

When you place a disk under VxVM control, the disk is initialized. Initialization destroys any existing data on the disk.

Prerequisites

- Only disks that are online and do not belong to a disk group can be used to create a disk group.
- The disk group name must be unique in the host or cluster.
- Creating a disk group requires at least one disk.

Usage Notes

- For information on the `vxdg` command, see the `vxdg`(1M) manual page.
- In the GUI, if multiple disks are specified in the **Disk Device(s)** field and only one disk name is specified in the **Disk Name(s)** field, VxVM appends numbers to the disk name so that each disk name is unique within its disk group.
- New disks must be placed under VxVM control and then added to a dynamic disk group before they can be used for volumes. The **Actions > Rescan** command performs these tasks to prepare new disks for VxVM use.
- When you place a disk under VxVM control, the disk is either encapsulated or initialized. Encapsulation preserves any existing data on the disk in volumes. Initialization destroys any existing data on the disk.
- If you place the root disk under VxVM control, you must encapsulate the disk. If you want to create an alternate boot disk, you can mirror the encapsulated root disk.
- Disks are automatically assigned a default name. Once a disk is under VxVM control, you can select **Actions > Rename Disk** in the GUI to change the disk name.

Usage Notes

- The disk group name must be unique.
- The new disk group must contain at least one disk.
- Only disks that are online and do not belong to another disk group can be used to create a disk group.

▼ To create a new disk group using the command line

Use the `vxdg` command as follows:

```
# /opt/VRTS/bin/vxdg init PRODddg PRODdg01=c1t1d1s2
```

Example

To create a disk group named `PRODdg` on a raw disk partition `c1t1d0s2`, where the disk name `PRODdg01` references the disk within the disk group:

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

▼ To create a new disk group using the GUI

1. Select the **Disk Groups** folder.

2. Select **Actions > New Disk Group**.

   The **New Disk Group** wizard welcome appears.

3. You can suppress the **Welcome to Create New Disk Group** page by selecting the **Do not show this page next time** checkbox.

4. Click **Next** to continue. The **New Disk Group** wizard appears. Enter the following information:

Group name:	Enter a new disk group name in the **Group Name** field.
Create cluster group	If the cluster feature is available, you can select the **Create cluster group** checkbox if the new disk group is to be used with clusters.
Available disks: Selected disks:	Select which disks (from **Available disks:**) you want to include in the group. Make sure the disks you want to include are in the right pane of the window (**Selected disks:**). (Normally, you would add all the disks you want in the group at this point. However, you can always add more disks later with the **Add Disk to Disk Group** procedure.)

Disk names:	Type in the names of any disks to be added that do not appear under **Selected disks:**.

5. When you have provided all the necessary information in the dialog box, click **Next**.

6. Click **Next** to add the disks.

7. The **Organization principle** screen now appears.

8. Click None, if you do not want to organize the disk group. Otherwise, click any of the other organization categories under **Organize Disk Group By** to create an ISP disk group (refer to Organizing Disk Groups).

9. If you choose an organization principle other than None, the **Specify Pool Names** screen appears. Specify the names for the Data pool and the Clone pool and click **Next**. You must specify the pool names in order to create the first storage pool for an ISP disk group.

10. The next screen confirms the disks you have selected. Click **Finish** to continue if you are satisfied with the disk selection. If you are not satisfied, you can click the **<Back** button to go back to the previous screen in order to modify your selections.

11. After clicking **Finish**, the new disk group will then appear under the Disks Groups node.

12. If the disks are not empty, indicate whether the disk should be initialized or encapsulated. If you initialize the disk, any existing data on the disk will be destroyed. If you encapsulate the disk, any existing data will be preserved in volumes.

    If you choose to encapsulate the root disk, the disk needs to be rebooted before encapsulation can take effect. You have the option of rebooting the system now or later.

13. Click **Next** to add the disks to the new disk group.

    The **Organization Principle** window appears.

14. Click **None**, if you do not want to organize the disk group. Otherwise, click **Organize Disk Group By** to create an ISP disk group.

15. If you choose an organization principle other than **None**, the **Specify Pool Names** window appears. Specify the names for the **Data pool** and the **Clone pool** and click **Next**. You must specify the pool names in order to create the first storage pool for an ISP disk group.

16. If you chose to encapsulate the root disk, reboot the system.

Adding Disks to a Disk Group

When a disk group is first created, it contains only a single disk. You may need to add more disks to the disk group. This section describes how to add disks to a disk group using the `vxdg` command or the GUI. If you have many disks to add to the disk group, it is easier to use the `vxdg` command.

Usage Notes

- By default, the GUI assigns default disk names by appending numbers to the disk group name so that each disk name is unique within its disk group. After the disk is under VxVM control, you can rename it.
- When you place a disk under VxVM control, the disk is either encapsulated or initialized. Encapsulation preserves any existing data on the disk in volumes. Initialization destroys any existing data on the disk.
- If you place the boot disk under VxVM control, you must encapsulate it. If you want to create an alternate boot disk, you can mirror the encapsulated boot disk.
- Disk encapsulation requires a system reboot.
- Disks cannot be added to deported disk groups.
- Disks must be under VxVM control and in a disk group before they can be used to create volumes.
- Disks must be online before they can be added to a disk group.
- Disks that already belong to a disk group cannot be added to another disk group.

▼ To add disks to a disk group using the command line

Use the `vxdg` command as follows:

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

Example

To add disks named `PRODdg02`, `PRODdg03`, and `PRODdg04` to the disk group `PRODdg`:

```
# /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
```

▼ To add a disk to a disk group using the GUI

1. Select the uninitialized disk to be placed under VxVM control.
2. Click the Disk group to which you wish to add a disk.
3. Choose **Actions > Add Disk to Disk Group**. The **Add Disk to Disk Group** wizard appears. Click **Next** to continue.
4. Complete the **Add Disk to Dynamic Group** wizard as follows:

Disk group name:	From the pull-down menu, select the group you want to add the disk to. To add the disk to a new disk group, click the **New disk group** button and enter the name of the new disk group in the dialog box.
Available disks: Selected disks:	Move the disk to be added from **Available disks** to **Selected disks**.

5. When you have provided all the necessary information in the dialog box, click **Next**.
6. When the confirmation window appears, click **Yes** to confirm your selection.
7. Click **Finish** to add the disk to the selected disk group.

Selecting a Volume Layout

VERITAS Volume Manager offers a variety of layouts that allow 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 your database's 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 and tablespaces are distributed across the available disks may be sufficient.

Striping also helps sequential table scan performance. When a table is striped across multiple physical devices, a high transfer bandwidth can be achieved by closely matching several DB2 parameters to ensure that database extents match the *stripe size* for the device. See "Tuning for Performance" on page 329 for more information. Another very important consideration when using the DB2 database, which by default performs striping at the tablespace container level, is setting the DB2_STRIPED_CONTAINERS variable. See "DB2_STRIPED_CONTAINERS (DB2 v7.x only)" on page 341 for more information on this issue.

Choosing Appropriate Stripe Unit Sizes

When creating a striped volume, you need to decide the number of columns to form a striped volume and the stripe unit size. You also need to decide how to stripe the volume. You may stripe a volume across multiple disk drives on the same controller or across multiple disks on multiple controllers. By striping across multiple controllers, disk I/O can be balanced across multiple I/O channels. The decision is based on the disk and controller bandwidth and the database workload. In general, for most OLTP databases, use the default stripe unit size of 64 K or smaller for striped volumes and 16 K for RAID-5 volumes.

Choosing Between Mirroring and RAID-5

VxVM provides two volume configuration strategies for data redundancy: mirroring and RAID-5. Both strategies allow continuous access to data in the event of disk failure. For most database configurations, we recommend using mirrored, striped volumes. If hardware cost is a significant concern, but having higher data availability is still important, use RAID-5 volumes.

RAID-5 configurations have certain performance implications you must consider. Writes to RAID-5 volumes require parity-bit recalculation, which adds significant I/O and CPU overhead. This overhead can cause considerable performance penalties in online transaction processing (OLTP) workloads. If the database has a high read ratio, however, RAID-5 performance is similar to that of a striped volume.

Volume Configuration Guidelines

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.
- When normal system availability is acceptable, put the tablespaces on file systems 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, ensure that the `NUM_IOSERVERS` and `DB2_PARALLEL_IO` settings are tuned to match the number of disk devices used in the stripe. See “Tuning for Performance” on page 329 for further information.
- For most workloads, use the default 64 K stripe-unit size for striped volumes and 16 K for RAID-5 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.
- When system availability is critical, use RAID-5 for read-intensive OLTP workloads to improve database performance and availability. Use RAID-5 logs 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 a Volume

VERITAS Volume Manager 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.

For databases where the data storage needs to be resilient and the data layout needs to be optimized for maximum performance, we recommend using VxVM. The striping and mirroring capabilities offered by a volume manager will help you achieve your manageability, availability, and performance goals.

After you decide on a volume layout, you can use the `vxassist` command or the GUI to create the volume.

If you choose to use the GUI, the *VERITAS Volume Manager Administrator's Guide* provides a detailed comparison of the layout choices and more detailed procedures on creating each volume layout type. The GUI gives you the option of placing a file system on the new volume or mirroring during volume creation.

Usage Notes

- Creating a volume requires a disk group name, volume name, volume size, and volume layout. You will also need to know subdisk names if you are creating a striped volume.
- Striped or mirrored volumes require at least two disks.
- Striped pro and concatenated pro volumes are mirrored by default, so a striped pro volume requires more disks than an unmirrored striped volume and a concatenated pro volume requires more disks than an unmirrored concatenated volume.
- You cannot use a striped pro or concatenated pro volume for a `root` or `swap` volume.
- A RAID-5 volume requires at least three disks. If RAID-5 logging is enabled, a RAID-5 volume requires at least four disks.
- RAID-5 mirroring is not supported.

▼ To create a volume using the command line

Use the `vxassist` command as follows:

```
# /opt/VRTS/bin/vxassist -g disk_group make volume_name size \
layout=layout_type
```

Example

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

```
# /opt/VRTS/bin/vxassist -g PRODdg make db01 1g layout=mirror
```

▼ To create a volume using the GUI

1. Under **Disk Groups** under the selected host in the left pane, click on the disk group to be used to create the volume.

2. Choose **Actions** > **New Volume**. You are asked if you want Volume Manager to select the disks to use or if you want to select them manually.

3. Select **Manually select disks to use with this volume**, then click **Next**.

4. The **Select Disks to use for Volume** screen appears. You can now:

   - Select disks to be used by VxVM when creating the volume. (Move to the **Included:** area.)
   - Select disks not to be used by VxVM when creating the volume. (Move to **Excluded:** area.)
   - Specify that the volume is to be mirrored across (**Mirror Across:**) or striped across (**Stripe Across:**) controllers, trays, targets, or enclosures.

Note For ISP volumes, the layouts **Mirror Across:** and **Stripe Across:** cannot be specified.

   - Specify ordered allocation (**Ordered**). Ordered allocation uses the specified storage to first concatenate disks, then to form columns, and finally to form mirrors.

5. Click **Next**. The **Select the attributes for this volume** screen appears.

6. Type in the Volume Name. This is a Volume Manager-specific name that is used in some Volume Manager commands. It is different from the volume label for the file system.

7. Specify the volume size, or select **Maxsize**.

   As you have already selected a disk, a size is shown in the volume size box that represents the maximum concatenated (simple or spanned) volume size on the disk. If you then click the **Maxsize** button, a new number appears in the volume size box that represents the maximum size for a spanned volume that spans all the disks in the group. You can also click on another volume layout and then click the **Maxsize** button to get the maximum size for that layout that involves unallocated space on all disks in the group.

If you choose Striped or RAID-5, Number of Columns and Stripe Unit Size need to have an entry. Defaults are provided.

8. Select the required layout:
   - Concatenated (Concatenated and Concatenated Mirrored)
   - Striped (Striped and Striped Mirrored)
   - RAID-5
   - Mirrored

9. Select the required options

Option	Notes
Mirror Info:	◆ To mirror the volume, select **Mirrored**. In the **Total Number of Mirrors** field, enter the total number of mirrors for the volume. **Note** Concatenated mirrored volumes are mirrored by default.
Initialize zero:	◆ To clear the volume before enabling it for general use, select **Initialize Zero**.
No layered volumes:	◆ To prevent the creation of a layered volume, select **No Layered Volumes**. In cases where a layered volume layout is appropriate, VxVM can create a layered volume when a non-layered layout is specified. This option ensures that the volume has a non-layered layout. If a layered (Pro) layout is selected, this option is ignored.
Enable FastResync Enable logging	◆ If **Enable FastResync** is checked, then **Enable Logging** is an option only if the volume is mirrored. (The reason it is only enabled when **Enable FastResync** is checked is to ensure that a new style DRL is created.)

10. After you provide all the necessary information in the dialog box, click **Next**. You are now asked if you want to create a file system.

11. If you want to create a file system, select **Create a File System** and follow the instructions. If you do not want to create a file system, select **No File System**, then click **Next**.

12. A summary of your selections appears. Click **Back** to make changes; otherwise, click **Finish**.

Creating a Volume Set

Volume Sets enable the use of the Multi-Volume Support feature with VERITAS File System (VxFS). It is also possible to use the VERITAS Enterprise Administrator (VEA) to create and administer volumes sets. For more information, see the VEA online help.

Note For details regarding usage of the `vxvset` command, see the `vxvset(1M)` manual page.

Note Most VxVM commands require superuser or equivalent privileges.

Usage Notes

- A maximum of 256 volumes may be configured in a volume set.
- Only VERITAS File System is supported on a volume set.
- The first volume in a volume set must be larger than 20MB.
- Raw I/O from and to a volume set is not supported.
- Volume sets can be used instead of volumes with the following `vxsnap` operations on instant snapshots: `addmir`, `dis`, `make`, `prepare`, `reattach`, `refresh`, `restore`, `rmmir`, `split`, `syncpause`, `syncresume`, `syncstart`, `syncstop`, `syncwait`, and `unprepare`. The third-mirror break-off usage model for full-sized instant snapshots is supported for volume sets provided that sufficient plexes exist for each volume in the volume set. Refer to the *VERITAS Volume Manager 4.1 Administrator's Guide* for more information on instant snapshots.
- A full-sized snapshot of a volume set must itself be a volume set with the same number of volumes and the same volume index numbers as the parent. The corresponding volumes in the parent and snapshot volume sets are also subject to the same restrictions as apply between standalone volumes and their snapshots.

▼ To create a volume set for use by VERITAS File System (VxFS), using the CLI

Use the following command:

```
# /opt/VRTS/bin/vxvset [-g diskgroup] -t vxfs make volset volume
```

Here `volset` is the name of the volume set, and `volume` is the name of the first volume in the volume set. The `-t` option defines the content handler subdirectory for the application that is to be used with the volume. This subdirectory contains utilities that an application uses to operate on the volume set. The operation of these utilities is determined by the requirements of the application and not by VxVM.

For example, to create a volume set named `myvset` that contains the volume `vol1`, in the disk group `mydg`, you would use the following command:

```
# /opt/VRTS/bin/vxvset -g mydg -t vxfs make myvset vol1
```

Note For further details on Creating a Volume Set, please refer to the *VERITAS Volume Manager Administrator's Guide*.

Adding a Volume to a Volume Set

Having created a volume set containing a single volume, you can use the following command to add further volumes to the volume set:

```
# /opt/VRTS/bin/vxvset [-g diskgroup] [-f] addvol volset volume
```

For example, to add the volume `vol2`, to the volume set `myvset`, use the following command:

```
# /opt/VRTS/bin/vxvset -g mydg addvol myvset vol2
```

Caution The `-f` (force) option must be specified if the volume being added, or any volume in the volume set, is either a snapshot or the parent of a snapshot. Using this option can potentially cause inconsistencies in a snapshot hierarchy if any of the volumes involved in the operation is already in a snapshot chain.

Note For further details on volumes and volume sets, refer to the *VERITAS Volume Manager Administrator's Guide*.

File System Creation Guidelines

Follow these guidelines when creating VxFS file systems:

- ◆ To take advantage of Quick I/O, online administration, fast recovery of the VxFS file system, and superior reliability features, select `vxfs` as the file system type.
- ◆ Specify the maximum block size and log size when creating file systems for databases.

Note Choose a file system block size that matches or is a multiple of the `PAGESIZE` parameter in the `create database` or `create tablespace` statement for your DB2 database or tablespace. The PAGESIZE parameter is defined in the `create database` or `create tablespace` statement.

It is possible to have a file system block size that is smaller than the database page size because the database page-size limit can be bigger than the file system block size. It is fine if the file system block size is smaller than the database page size because VxFS will not perform multiple I/O operations for each database I/O operation. VxFS is capable of performing I/Os with multiple blocks. For example, if your database page size is 8k and your file system block size is 4K, VxFS can put two 4k blocks together to perform one 8k database I/O operation. The DB2 instance will also need data in `EXTENTSIZE`, which is a multiple of `PAGESIZE`. These page size rules also apply for extent size

When creating the file system, set the number of file system blocks in the intent log so that the log size is 16MB. For example, if the file system block size is 8K (that is, 8192), it will take 2000 blocks to make a 16MB log (2000 x 8192 = ~16MB). If the file system block size is 4K (that is, 4096), then twice as many blocks as in the 8K case would need to be allocated (4000 in this example).

- ◆ Never disable the intent logging feature of the file system.
- ◆ For database logs, create a single file system using a simple (and mirrored, if necessary) volume. Put the other tablespaces and database files on separate file systems created on striped, striped and mirrored, or RAID-5 volumes.
- ◆ 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.

Creating a VxFS File System

Always specify `vxfs` as the file system type to take advantage of Quick I/O, Storage Rollback, online administration, fast recovery of the VxFS file system, and superior reliability features.

The GUI lets you add a file system on a new volume during the volume creation process. See the *VERITAS Volume Manager User's Guide - VERITAS Enterprise Administrator* for a detailed comparison of the layout choices and more detailed procedures on creating each volume layout type.

Usage Notes

- See the `mkfs`(1M) and `mkfs_vxfs`(1M) manual pages for more information about the options and variables available for use with the `mkfs` command.
- See the `mount`(1M) and `mount_vxfs`(1M) manual pages for more information about mount settings.
- In the GUI, you must specify a file system mount point if the file system is to be mounted at startup.
- If you select the **Add to file system table** checkbox in the GUI, the file system table file will be automatically updated when the file system is mounted.
- When specifying a mount point in the GUI, you must use an absolute path name (that is, it must begin with `/`).

▼ To create a VxFS file system on an existing volume using the command line

Use the `mkfs` command to create a VxFS file system on an existing volume as follows:

```
# /usr/sbin/mkfs -F 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 VxVM volume on which to create the file system (for example, `/dev/vx/rdsk/PRODdg/db01`)
- *`size`* is the size of the new file system (optional)

If you do not specify *`size`*, the file system will be as large as the underlying volume or device partition.

Example

To create a VxFS file system that supports files larger than 2GB on the newly created `db01` volume:

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

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

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.

The `mkfs` command displays output similar to the following:

```
version 6 layout
20480 sectors, 10240 blocks of size 1024, log size 1024 blocks
```

You can now mount the newly created file system. See "Mounting a File System" on page 51.

▼ To add a new file system to an existing volume using the GUI

1. Select the volume to contain the file system.
2. Choose **Actions** > **File System** > **New File System**.
3. Complete the **New File System** dialog box as follows:

File System Type:	Select the file system type from the pull-down menu.
Create Options:	◆ Specify the allocation size and block size if you do not want to use the default. ◆ To specify whether large files (files greater than or equal to 2 GB) will be supported, click **New File System Details**. Select either **largefiles** or **nolargefiles**.
Compress (checkbox):	◆ If your platform supports file compression, then you can select **Compress** to compress the files on your file system

Mount Options:	◆ Enter the mount point for the file system, if you want the file system mounted at system startup. ◆ Select the **Create mount point** checkbox if you want the system to create the mount point if it does not already exist. ◆ Select the **Read only** and **Honor setuid** checkboxes, as required. ◆ Select the **Add to file system table** and **Mount at boot** checkboxes to update the system table file and mount the file system at system startup. ◆ To update the system table file and *not* mount the file system at system startup, select **Add to file sytem table** checkbox and leave the **Mount at Boot** checkbox unselected. ◆ To specify `mount` options, click **Mount File System Details** and specify the appropriate options in the Mount Details dialog box.

4. After you provide all necessary information in the dialog box, click **OK**.

Support for Large File Systems and Large Files

In conjunction with VxVM, VxFS can support file systems up to 8 exabytes in size. For large database configurations, this eliminates the need to use multiple file systems because of the size limitations of the underlying physical devices.

Changes implemented with the VxFS Version 6 disk layout have greatly expanded file system scalability, including support for large files. You can create or mount file systems with or without large files by specifying either the `largefiles` or `nolargefiles` option in `mkfs` or `mount` commands. If you specify the `nolargefiles` option, a file system cannot contain files 2 GB or larger.

Usage Notes

- See the `mount_vxfs`(1M) and `mkfs_vxfs`(1M) manual pages for detailed information on mounting and creating file systems.
- See the `fsadm_vxfs`(1M) manual pages for detailed information about large files.

▼ **To enable large files on a file system that was created without the largefiles option**

Use the `fsadm` command as follows:

```
# /opt/VRTS/bin/fsadm -F vxfs -o largefiles /mount_point
```

Caution Make sure the applications and tools you use can handle large files before enabling the large file capability. Applications and system administration utilities can experience problems if they are not large file aware.

Multi-volume Support

The multi-volume support feature enabled by Version 6 disk layout allows several volumes to be represented by a single logical object, known as a *volume set*. The `vxvset` command can be used to create and administer volume sets in VERITAS Volume Manager.

VxFS's multi-volume support feature can be used with volume sets. There are two VxFS commands associated with multi-volume support:

- `fsapadm` - VxFS allocation policy administration utility
- `fsvoladm` - VxFS device administration utility

For more information about volume sets and multi-volume support, see the *VERITAS File System Administrator's Guide*.

Mounting a File System

After creating a VxFS file system, mount the file system using the `mount` command. By default, the `mount` command tries to enable Quick I/O. If Quick I/O is not installed or licensed, no error messages are displayed unless you explicitly specify the `-o qio` mount option. If necessary, you can turn the Quick I/O option off at mount time or you can remount the file system with the `-o noqio` option.

Prerequisites

- You must be logged in as `root` to mount a file system.
- DBAs should log in as the DB2 instance owner.

Usage Notes

- See the `mount_vxfs`(1M) manual page for more information about mount settings.
- See the `mount`(1M) manual page for more information about generic mount options.
- If you use the GUI, the file system table file is automatically updated.
- The mount point must be an absolute path name (that is, it must begin with `/`).
- If you use the GUI, the path specified for the mount point will be created if it does not already exist.

▼ To mount a file system using the command line

Use the `mount` command as follows:

```
# /usr/sbin/mount -F vxfs [generic_options] [-r] \
[-o specific_options] special /mount_point
```

where:

- *`generic_options`* are the options common to most file systems
- `-r` mounts the file system as read only
- *`specific_options`* are options specific to the VxFS file system
- *`special`* is a block special device
- */`mount_point`* is the directory where the file system will be mounted

Example

To mount a file system named `/db01` that supports large files on volume `/dev/vx/dsk/PRODdg/db01`:

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

If you would like `/db01` to be mounted automatically after rebooting, add an entry for it in `/etc/vfstab` as follows:

```
/dev/vx/dsk/PRODdg/db01 /dev/vx/rdsk/PRODdg/do1 /db01 vxfs \
largefiles,qio  0    2
```

If you do not need to use Quick I/O files, set `noqio` instead of `qio` as one of the options.

▼ To mount a file system on an existing volume using the GUI

1. Select the volume that contains the file system to be mounted.
2. Choose **Actions** > **File System** > **Mount File System**.
3. Complete the **Mount File System** dialog box (see step 3 on page 52 in the procedure below).
4. After you have provided all the necessary information in the dialog box, click **OK**.

▼ To mount any file system using the GUI

1. Select the file system to be mounted.
2. Choose **Actions** > **Mount File System**.
3. Complete the **Mount File System** dialog box as follows:

FS Type:	Select the file system type.

Mount Options:	◆ If you want the system to use the mount options defined in the system table, check **Mount using options in the file system table**. ◆ Enter the mount point for the file system, if you want the file system mounted at system startup. ◆ Select the **Create Mount Point** checkbox if you want the system to create the mount point if it does not already exist. ◆ Select the **Read Only** and **Honor setuid** checkboxes, as required. ◆ To specify mount options, click **Mount File System Details** and specify the appropriate options in the **Mount Details** dialog box.

4. After you have provided all the necessary information in the dialog box, click **OK**.

Unmounting a File System

If you no longer need to access the data in a file system, you can unmount the file system using the `umount` command.

Prerequisites

- A file system must exist and be mounted in order to be unmounted.

Usage Notes

- See the `umount`(1M) manual page for more information on mounting file systems.
- You cannot unmount a file system that is in use.

▼ To unmount a file system using the command line

1. Use the `fuser` command to make sure that the file system is not being used:

   ```
   # fuser -c /mount_point
   ```

   where the `-c` option provides information on file system mount points and any files within mounted file systems.

Note If the file system is being used and you need to unmount it, use the `fuser -ck` command. See the `fuser`(1M) man page for more information.

2. Unmount the file system using the `umount` command:

   ```
   # umount special
   ```

   or

   ```
   # umount /mount_point
   ```

   or

   ```
   # umount -f /mount_point
   ```

   where:

   - *`special`* is a block special device
   - *`/mount_point`* is the location where the file system is mounted
   - `-f` forcibly unmounts the mount point

Example

To verify that the file system /db01 is not in use and then unmount the file system:

```
# fuser -c /db01
/db01:
# umount /db01
```

▼ To unmount a file system on a volume using the GUI

1. Select the volume containing the file system to be unmounted.
2. Choose **Actions > File System > Unmount File System**.
3. Click **Yes** in the **Unmount File System** dialog box to confirm that you want to unmount the file system.
4. If an entry exists for a file system in the file system table, a **Remove File System** dialog appears. Click **Yes** in the **Remove File System** dialog if you want the file system table entry removed.

Understanding Fragmentation

When free resources are initially allocated to files in a VERITAS file system, they are aligned in the most efficient order possible to provide optimal performance. On an active file system, the original order is lost over time as files are created, removed, or resized. As space is allocated and deallocated from files, the available free space becomes broken into fragments. This means that space must be assigned to files in smaller and smaller extents. This process is known as fragmentation. Fragmentation leads to degraded performance and availability. The degree of fragmentation depends on file system usage and activity.

Controlling Fragmentation

Allocation units in VxFS are designed to help minimize and control fragmentation. Over time, however, file systems eventually become fragmented.

VxFS provides online reporting and optimization utilities to enable you to monitor and defragment a mounted file system. These utilities are accessible through the file system administration command, `fsadm`. Using the `fsadm` command, you can track and eliminate fragmentation without interrupting user access to the file system.

Types of Fragmentation

VxFS addresses two types of fragmentation:

- Directory Fragmentation

  As files are created and removed, gaps are left in directory inodes. This is known as directory fragmentation. Directory fragmentation causes directory lookups to become slower.

- Extent Fragmentation

  As files are created and removed, the free extent map for an allocation unit changes from having one large free area to having many smaller free areas. Extent fragmentation occurs when files cannot be allocated in contiguous chunks and more extents must be referenced to access a file. In a case of extreme fragmentation, a file system may have free space that cannot be allocated.

Monitoring Fragmentation

You can monitor fragmentation in VxFS by running reports that describe fragmentation levels. Use the `fsadm` command to run reports on directory fragmentation and extent fragmentation. The `df` command, which reports on file system free space, also provides information useful in monitoring fragmentation.

Use the following commands to report fragmentation information:

- `fsadm -D`, which reports on directory fragmentation.
- `fsadm -E`, which reports on extent fragmentation.
- `/opt/VRTS/bin/df -F vxfs -o s`, which prints the number of free extents of each size.

Defragmenting a File System

You can use the online administration utility `fsadm` to defragment or reorganize file system directories and extents. The `fsadm` utility defragments a file system mounted for read/write access by:

- Removing unused space from directories.
- Making all small files contiguous.
- Consolidating free blocks for file system.

The following options are for use with the `fsadm` utility:

Options

`-d`	Reorganizes directories. Directory entries are reordered to place subdirectory entries first, then all other entries in decreasing order of time of last access. The directory is also compacted to remove free space.
`-a`	Use in conjunction with the `-d` option to consider files not accessed within the specified number of days as “aged” files. Aged files are moved to the end of the directory. The default is 14 days.
`-e`	Reorganizes extents. Files are reorganized to have the minimum number of extents.
`-D -E`	Produces reports on directory and extent fragmentation, respectively.
`-v`	Specifies verbose mode and reports reorganization activity.
-l	Specifies the size of a file that is considered large. The default is 64 blocks.
-t	Specifies a maximum length of time to run, in seconds.

-p	Specifies a maximum number of passes to run. The default is five.
-s	Prints a summary of activity at the end of each pass.
-r	Specifies the pathname of the raw device to read to determine file layout and fragmentation. This option is used when *fsadm* cannot determine the raw device.

Usage Notes

- If you specify `-d` and `-e`, directory reorganization is always completed first.
- If you use both `-D` and `-E` with the `-d` and `-e` options, the fragmentation reports are produced both before and after reorganization.
- The `-t` and `-p` options control the amount of work performed by `fsadm`, either in a specified time or by a number of passes. By default, `fsadm` runs five passes. If both `-t` and `-p` are specified, `fsadm` exits if either of the terminating conditions are reached.

Note You must have superuser (`root`) privileges to reorganize a file system using the `fsadm` command.

▼ To defragment a file system using the command line

Run the `fsadm` command followed by the options specifying the type and amount of defragmentation. Complete the command by specifying the mount point or raw device to identify the file system.

```
# /opt/VRTS/bin/fsadm [-d] [-D] [-e] [-E] [-s] [-v] \
  [-l largesize] [-a days] [-t time] [-p pass_number] \
  [-r rawdev_path] mount_point
```

Refer to the *File System Administrator's Guide* for instructions and information on scheduling defragmentation.

Example

To defragment a file system:

```
# /opt/VRTS/bin/fsadm -d -D /db2data_qiovm
Directory Fragmentation Report
          Dirs           Total    Immed      Immeds   Dirs to    Blocks to
          Searched       Blocks   Dirs       to Add   Reduce     Reduce
 total       5               1     4              0       00


 Direct ory Fragmentation Report
          Dirs           Total    Immed      Immeds   Dirs to    Blocks to
          Searched       Blocks   Dirs       to Add   Reduce     Reduce
 total       5               1     4              0       00
```

▼ To defragment a file system on a volume using the GUI

1. Select the volume containing the file system to be defragmented.
2. Choose **Actions > File System > Defrag File System**.
3. Select **Yes** in the displayed dialog box.

Resizing a File System

If you need to extend or shrink a VxFS file system, you can use the `fsadm` command.

If a VxFS file system requires more space, you can use this procedure to extend the size of the file system. If a VxFS File System is too large and you need the space elsewhere, you can use this procedure to shrink the file system.

Note If you are using the command line, remember to increase the size of the underlying device or volume before increasing the size of the file system. See the *VERITAS Volume Manager Administrator's Guide* for more information.

Prerequisites

- ◆ This task requires a mounted file system.
- ◆ You must know either the desired size or the amount of space to add to or subtract from the file system size.

Usage Notes

- ◆ See the `format`(1M) and `fsadm_vxfs`(1M) manual pages for more details.

▼ To resize a file system using the command line

Use `fsadm` command as follows:

```
# /opt/VRTS/bin/fsadm -F vxfs [-b newsize] \
  [-r rawdev] /mount_point
```

where:

- ◆ *`newsize`* is the size (in sectors) to which the file system will increase or shrink
- ◆ *`rawdev`* specifies the name of the raw device if there is no entry in `/etc/vfstab` and `fsadm` cannot determine the raw device
- ◆ */`mount_point`* is the location where the file system is mounted

Example

To extend the file system `/db01` to 2 GB:

```
# /opt/VRTS/bin/fsadm -F vxfs -b 2g /db01
```

Note See the *VERITAS File System Administrator's Guide* and `fsadm_vxfs`(1M) manual page for information on how to perform common file system tasks using `fsadm`.

Resizing a File System and the Underlying Volume

The `fsadm` command resizes the file system only. If you attempt to use `fsadm` to make the file system the same size or larger than the underlying volume, the `fsadm` command will fail. To resize the file system *and* its underlying volume, use the `vxresize` command instead.

Prerequisites

- ◆ You must know the new desired size of the file system.

Usage Notes

- ◆ If you use the GUI, the underlying volume is resized when the file system is resized.
- ◆ `vxresize` works with VxFS file systems only.
- ◆ When resizing large volumes, `vxresize` may take a long time to complete.
- ◆ Resizing a volume with a usage type other than FSGEN or RAID5 can result in data loss. If such an operation is required, use the `-f` option to forcibly resize such a volume.
- ◆ You cannot resize a volume that contains plexes with different layout types.
- ◆ See the `vxresize`(1M) manual page for more details.

Example

To extend a 1-gigabyte volume, `homevol`, that contains a VxFS file system, to 10 gigabytes using the spare disks `disk10` and `disk11, enter`:

```
# /etc/vx/bin/vxresize -b -F vxfs -t homevolresize homevol 10g \
disk10 disk11
```

The `-b` option specifies that this operation runs in the background. Its progress can be monitored by specifying the task tag `homevolresize` to the `vxtask` command.

Growing a File System Automatically Using VxDBA Monitoring Agent

You can use the VxDBA Monitoring Agent to monitor file system space, and when the space usage reaches a configured threshold value, a predefined action script automatically grows the file system. See “Managing File System Space” on page 306 for more information.

Using VERITAS Quick I/O 3

VERITAS Quick I/O is a VxFS feature included in VERITAS Storage Foundation *for DB2* that lets applications access preallocated VxFS files as raw character devices. Quick I/O provides the administrative benefits of running databases on file systems without the performance degradation typically associated with running databases on file systems or raw devices. This chapter describes how to set up and use Quick I/O.

Topics covered in this chapter include:

Understanding Quick I/O

How Quick I/O Works

VERITAS Quick I/O supports direct I/O and kernel asynchronous I/O and allows databases to access regular files on a VxFS file system as raw character devices.

The benefits of using Quick I/O for DB2 databases 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 containers

How Quick I/O Improves Database Performance

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

- Supporting kernel asynchronous I/O
- Supporting direct I/O
- Avoiding kernel write locks on database files
- Avoiding double buffering

Supporting Kernel Asynchronous I/O

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 lets the database server take advantage of kernel-supported asynchronous I/O on file system files accessed using the Quick I/O interface.

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.

Avoiding Kernel Write Locks

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.

Quick I/O Requirements

To use Quick I/O, you must:

- Preallocate files on a VxFS file system
- Use a special file naming convention to access the files

Preallocation

Preallocating database files for Quick I/O allocates contiguous space for the files. The file system space reservation algorithms attempt to allocate space for an entire file as a single contiguous extent. When this is not possible due to lack of contiguous space on the file system, the file is created as a series of direct extents. Accessing a file using direct extents is inherently faster than accessing the same data using indirect extents. Internal tests have shown performance degradation in OLTP throughput when using indirect extent access. In addition, this type of preallocation causes no fragmentation of the file system.

You must preallocate Quick I/O files because they cannot be extended through writes using their Quick I/O interfaces. They are initially limited to the maximum size you specify at the time of creation. To extend Quick I/O files, see "Extending a Quick I/O File" on page 81.

Naming Convention

VxFS uses a special naming convention to recognize and access Quick I/O files as raw character devices. VxFS recognizes the file when you add the following extension to a file name:

```
::cdev:vxfs:
```

Whenever an application opens an existing VxFS file with the extension `::cdev:vxfs:` (`cdev` being an acronym for *character device*), the file is treated as if it were a raw device. For example, if the file `temp01` is a regular VxFS file, then an application can access `temp01` as a raw character device by opening it with the name:

```
.temp01::cdev:vxfs:
```

Note We recommend reserving the `::cdev:vxfs:` extension *only* for Quick I/O files. If you are not using Quick I/O, you could technically create a regular file with this extension; however, doing so can cause problems if you later enable Quick I/O.

How to Set Up Quick I/O

Quick I/O is part of the VxFS binaries shipped with VERITAS Storage Foundation *for DB2*. By default, Quick I/O is enabled when you mount a VxFS file system.

If Quick I/O is not available in the kernel, or the VERITAS Storage Foundation *for DB2* license is not installed, a file system mounts by default without Quick I/O, the Quick I/O file name is treated as a regular file, and no error message is displayed. If, however, you specify the `-o qio` option, the `mount` command prints the following error message and terminates without mounting the file system.

```
VxFDD: You don't have a license to run this program
vxfs mount: Quick I/O not available
```

Depending on whether you are creating a new database or are converting an existing database to use Quick I/O, you have the following options:

- If you are creating a new database:
  - You can use the `qiomkfile` command to preallocate space for database files and make them accessible to the Quick I/O interface. See “Creating Database Containers as Quick I/O Files Using qiomkfile” on page 68 for more information.
  - You can use the `setext` command to preallocate space for database files and create the Quick I/O files. See “Preallocating Space for Quick I/O Files Using the setext Command” on page 70.
- If you are converting an existing database:
  - You can create symbolic links for existing VxFS files, and use these symbolic links to access the files as Quick I/O files. See “Accessing Regular VxFS Files as Quick I/O Files” on page 72 for more information.
  - You can convert your existing DB2 database files to use the Quick I/O interface using the `qio_getdbfiles` and `qio_convertdbfiles` commands. See “Converting DB2 Containers to Quick I/O Files” on page 74 for more information.

Creating Database Containers as Quick I/O Files Using qiomkfile

You can create Database Managed Space (DMS) containers with the type 'DEVICE' using Quick I/O. The best way to preallocate space for DB2 tablespace containers and to make them accessible using the Quick I/O interface is to use the `qiomkfile` command. You can use the `qiomkfile` command to create Quick I/O files for either temporary or permanent tablespaces.

Prerequisites

- You can create Quick I/O files only on VxFS file systems.
- If you are creating containers 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 DB2 Quick I/O files.

qiomkfile Options

`-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 DB2 tablespace resizing. See "Extending a Quick I/O File" on page 81 for more information.
`-r`	Increases the file *to* a specified size to allow DB2 tablespace resizing. See "Extending a Quick I/O File" on page 81 for more information.
`-s`	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.

Caution 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.

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.

▼ To create a container as a Quick I/O file using qiomkfile

1. Create a Quick I/O-capable file using the `qiomkfile` command:

```
$ /opt/VRTS/bin/qiomkfile  -s file_size \
  /mount_point/filename
```

2. Create tablespace containers using this file with the following SQL statements:

```
$ db2 connect to database
$ db2 "create tablespace tbsname managed by database using \
( DEVICE '/mount_point/filename' size )"
$ db2 terminate
```

Example

- To create a 100MB Quick I/O-capable file named `dbfile` on the VxFS file system `/db01` using a relative path name:

```
  $ /opt/VRTS/bin/qiomkfile  -s 100m /db01/dbfile
  $ ls -al
-rw-r--r--   1 db2inst1 db2iadm1 104857600 Oct 2 13:42  .dbfile
lrwxrwxrwx   1 db2inst1 db2iadm1      19    Oct 2 13:42  dbfile -> \
                                             .dbfile::cdev:vxfs:
```

In this 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 (soft) 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.

We can then add the file to the DB2 database `PROD`:

```
$ db2 connect to PROD
$ db2 create tablespace NEWTBS managed by database using \
( DEVICE '/db01/dbfile' 100m )
$ db2 terminate
```

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.

Prerequisites

- The `setext` command requires superuser (`root`) privileges.

Usage Notes

- You need to use the `chown` and `chgrp` commands 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:

```
# chown db2inst1 .filename
# chgrp db2iadm1 .filename
# chmod 660 .filename
```

Example

To access the mount point `/db01`, create a container, 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 db2inst1 .dbfile
  # chgrp db2iadm1 .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
   ```

Example

To access the VxFS file `dbfile` as a Quick I/O file:

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

To show the symbolic link created:

```
$ ls -lo .dbfile dbfile
-rw-r--r--   1 db2inst1       104890368     Oct 2 13:42  .dbfile
lrwxrwxrwx   1 db2inst1              19     Oct 2 13:42  dbfile -> \
                                                       .dbfile::cdev:vxfs:
```

Converting DB2 Containers to Quick I/O Files

Special commands, available in the `/opt/VRTSdb2ed/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. For information on creating Storage Checkpoints, see "Using Storage Checkpoints and Storage Rollback" on page 121.

Prerequisites

- Log in as the DB2 instance owner (typically, the user ID `db2inst1`) to run the `qio_getdbfiles` and `qio_convertdbfiles` commands.
- You must predefine the DB2 environment variable *$DB2DATABASE*.
- The repository must exist before you can convert to Quick I/O files. Run the `db2ed_update` command to update or create the repository.
- Files you want to convert must be regular files on VxFS file systems or links that point to regular VxFS files.

Options

For the `qio_getdbfiles` command:

`-T`	Lets you specify the type of database as `db2`. Specify this option only in environments where the type of database is ambiguous (for example, when multiple types of database environment variables, such as *`$ORACLE_SID`*, *`SYBASE`*, *`DSQUERY`*, and *`$DB2INSTANCE`*, are present on a server).

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).
`-f`	Reports on the current fragmentation levels for database files listed in the mkqio.dat file. Fragmentation is reported as not fragmented, slightly fragmented, fragmented, highly fragmented.
`-h`	Displays a help message.

`-i`	Creates the extra links for all containers and log files in the `/dev` directory to support SAP's `brbackup`.
`-T`	Lets you specify the type of database as db2. Specify this option only in environments where the type of database is ambiguous (for example, when multiple types of database environment variables, such as *`$ORACLE_SID`*, *`SYBASE`*, *`DSQUERY`*, and *`$DB2INSTANCE`* are present on a server).
`-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.

Usage Notes

- Converting existing database files to be Quick I/O files may not be the best choice if the files are fragmented. Use the `-f` option to determine the fragmentation levels and either:

  - Exclude files that are highly fragmented and do not have sufficient contiguous extents for Quick I/O use.

  or

  - Create new files with the `qiomkfile` command, rather than converting the files using the `qio_convertdbfiles` command. The new files will be contiguous. You must then move data from the old files to the new files using the DB2 database export/import facility `db2move` or `restore redirect`, and then define the new files to the database.

- `qio_getdbfiles` skips any tablespaces that have a type of *system managed space* (SMS), as these tablespaces are based on a directory format and not suitable for conversion.

- Instead of using the `qio_getdbfiles` command, you can manually create the `mkqio.dat` file containing the DB2 instance filenames that you want to convert to Quick I/O files.

  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.

▼ To extract a list of DB2 containers 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
$ export DB2DATABASE=database_name
$ /opt/VRTSdb2ed/bin/qio_getdbfiles
```

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 DB2 database container 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.

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:

```
/db01/file1         210356
/db01/file2         157996
/db01/file3         38096
/db01/file4         394932
/db01/file5         911784
```

▼ To convert the DB2 database files to Quick I/O files

1. Make the database inactive by either shutting down the instance or disabling user connections.

Caution Running the `qio_convertdbfiles` command while the database is up and running can cause severe problems with your database, including loss of data, and corruption.

2. Run the `qio_convertdbfiles` command from the writable directory where the `mkqio.dat` list resides:

```
$ cd /extract_directory
$ export DB2DATABASE=database_name
$ /opt/VRTSdb2ed/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. Make the database active again.

   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 active, make it inactive by either shutting down the instance or disabling user connections.

2. Run the following command from the writable directory where the `mkqio.dat` list resides:

```
$ cd /extract_directory
$ export DB2DATABASE=database_name
$ /opt/VRTSdb2ed/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.

Understanding 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.

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.

Options

`-al`	Lists all files on a file system, including Quick I/O files and their links.
`-lL`	Shows if Quick I/O was successfully installed and enabled.
`-alL`	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
```

Example

To show the absolute path name created using `qiomkfile` with the `-a` option:

```
$ ls -al d* .d*
-rw-r--r--    1 db2inst1  db2iadm1 104890368Oct 2 13:42.dbfile
lrwxrwxrwx    1 db2inst1  db2iadm1   19    Oct 2 13:42  dbfile -> \
                                                    .dbfile::cdev:vxfs:
```

▼ To determine if a container has been converted to Quick I/O

Use the `ls` command as follows:

```
  $ ls -lL filename
```

Example

To determine if Quick I/O is installed and enabled:

```
$ ls -lL dbfile
crw-r--r--   1 db2inst1   db2iadm1    45,  1  Oct  2 13:42  dbfile
```

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
```

Example

To show how the Quick I/O file name `dbfile` is resolved to that of a raw device:

```
$ ls -alL d* .d*
crw-r--r--   1 db2inst1  db2iadm1      45,   1  Oct 2 13:42   dbfile
-rw-r--r--   1 db2inst1  db2iadm1  104890368  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.

Prerequisites

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

Options

`-e`	Extends the file *by* a specified amount to allow DB2 container resizing.
`-r`	Increases the file *to* a specified size to allow DB2 container resizing.

Usage Notes

- You can also grow VxFS file systems online (provided the underlying disk or volume can be extended) using the `fsadm` command.

Note You must have superuser (`root`) privileges to resize VxFS file systems using the `fsadm` command.

- You can also use the VxDBA Monitoring Agent to monitor file system space and automatically grow file systems. See "Managing Space Usage and the VxDBA Monitoring Agent" on page 305 for information on how to start and use the VxDBA Monitoring Agent, set space alarm thresholds, and automatically grow file systems.
- See the `fsadm_vxfs`(1M) and `qiomkfile`(1M) manual pages for more information.

▼ 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:

   ```
   # /opt/VRTS/bin/fsadm -b newsize /mount_point
   ```

   where:

   - `-b` is the option for changing size
   - *newsize* is the new size of the file system in bytes, kilobytes, megabytes, blocks, or sectors
   - *mount_point* is the file system's mount point

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
   ```

Example

- To grow VxFS file system `/db01` to 500MB and extend the tbs1_cont001 Quick I/O file by 20MB:

  ```
  # /opt/VRTS/bin/fsadm -b 500M /db01
  $ /opt/VRTS/bin/qiomkfile -e 20M /db01/tbs1_cont001
  ```

- To grow VxFS file system `/db01` to 500MB and resize the tbs1_cont001 Quick I/O file to 300MB:

  ```
  # /opt/VRTS/bin/fsadm -b 500M /db01
  $ /opt/VRTS/bin/qiomkfile -r 300M /db01/tbs1_cont001
  ```

Monitoring Tablespace Free Space with DB2 and Extending Tablespace Containers

DB2 does not automatically make use of extended DMS files. When tablespace space needs to be extended, a number of DB2 commands must be run. Unlike raw devices, a Database Administrator can easily extend Quick I/O files online. Using this method, a Database Administrator can monitor the free space available in the DB2 tablespaces and use the `qiomkfile` command to grow the Quick I/O files online as needed (typically when the file is about 80 to 90% 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.

Prerequisites

- Log on as the DB2 instance owner.

Usage Notes

- Monitor the free space available in the Quick I/O file, and grow the file as necessary with the `qiomkfile` command.
- A Database Administrator can grow underlying VxFS file systems online (provided the underlying disk or volume can be extended) using the `fsadm` command. See the `fsadm`(1M) manual page for more information.
- It is strongly recommended that if a DB2 tablespace is running out of space, that all containers are grown by the same amount. It is possible to add extra containers to the tablespace, but this results in DB2 performing a relayout of the data to balance usage across all available containers in the tablespace.
- For more information, see "Tuning for Performance" on page 329, the DB2 Administration Guide section on Managing Storage, and the DB2 SQL Reference Guide for information on the `ALTER TABLESPACE` command.

▼ To monitor the free space available in a DB2 tablespace

Use the following DB2 commands:

```
$ db2 connect to database
$ db2 list tablespaces show detail
$ db2 terminate
```

▼ To extend a Quick I/O file using qiomkfile

Use the `qiomkfile` command to extend the Quick I/O file (if the container is running low on free blocks):

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

▼ To extend a DB2 tablespace by a fixed amount

Use the following DB2 commands:

```
$ db2 connect to database
$ db2 alter tablespace tablespace-name extend (ALL amount)
$ db2 terminate
```

Example

To monitor the free space on the tablespaces in database `PROD`:

```
$ db2 connect to PROD
$ db2 list tablespaces show detail
$ db2 terminate
```

To extend the three DB2 containers owned by tablespace `EMP` by 500 MB using the `qiomkfile` command:

```
$ /opt/VRTS/bin/qiomkfile -e 500M tbsEMP_cont001
$ /opt/VRTS/bin/qiomkfile -e 500M tbsEMP_cont002
$ /opt/VRTS/bin/qiomkfile -e 500M tbsEMP_cont003
```

To notify DB2 that all containers in tablespace `EMP` have grown by 500 MB:

```
$ db2 connect to PROD
$ db2 alter tablespace EMP extend (ALL 500M)
$ db2 terminate
```

To verify the newly allocated space on the tablespace `EMP` in database `PROD`:

```
$ db2 connect to PROD
$ db2 list tablespaces show detail
$ db2 terminate
```

Recreating Quick I/O Files After Recovering a Database

If you need to recover 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.

Prerequisites

- Recover your database before attempting to recreate the Quick I/O files.
- Log in as the DB2 instance owner (typically, the user ID `db2inst1`) 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. If you do not have a `mkqio.dat` file, see "Converting DB2 Containers to Quick I/O Files" on page 74.
- The *`DB2DATABASE`* environment variable must be set.

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

Use the `qio_recreate` command as follows:

```
# /opt/VRTSdb2ed/bin/qio_recreate
```

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 you will see a warning message.
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 you will see a warning message.

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.

Prerequisites

- The file system you are planning to remount must be located in the `/etc/filesystems` file.

▼ To disable Quick I/O

1. If the database is active, make it inactive by either shutting down the instance or disabling user connections.

2. To change Quick I/O files back to regular VxFS files, run the following command from the writable directory where the `mkqio.dat` list resides:

   ```
   $ /opt/VRTSdb2ed/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:

   ```
   # /opt/VRTS/bin/mount -F vxfs -o remount,noqio /mount_point
   ```

Using VERITAS Cached Quick I/O 4

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. Cached Quick I/O is best-suited for use with Oracle8*i*.

This chapter describes how to enable and use Cached Quick I/O for enhanced performance.

Topics covered in this chapter include:

Understanding Cached Quick I/O

How Cached Quick I/O Works

Cached Quick I/O is a specialized external caching mechanism suitable for dynamically changing memory allocation available for caching DB2 tablespace data. Cached Quick I/O can be selectively applied to containers that are suffering an undesirable amount of physical disk I/O due to insufficientDB2 BufferPool reservation. Cached Quick I/O works by taking advantage of the available physical memory that is left over after the operating system reserves the amount it needs and the DB2 BufferPools have been sized to their normal operating capacity. This extra memory serves as a cache to store file data, effectively serving as a second-level cache backing the BufferPool(s).

For example, consider a system configured with 12GB of physical memory, an operating system using 1GB, and a total DB2 BufferPool size of 3.5GB. Unless you have other applications running on your system, the remaining 7.5GB of memory is unused. If you enable Cached Quick I/O, these remaining 7.5GB become available for caching database files.

Note You cannot allocate specific amounts of the available memory to Cached Quick I/O. When enabled, Cached Quick I/O takes advantage of available memory.

Cached Quick I/O is not beneficial for all containers in a database. Turning on caching for all database containers can degrade performance due to extra memory management overhead (double buffer copying). You must use file I/O statistics to determine which individual database containers benefit from caching, and then enable or disable Cached Quick I/O for individual containers.

If you understand the applications that generate load on your database and how this load changes at different times during the day, you can use Cached Quick I/O to maximize performance. By enabling or disabling Cached Quick I/O on a per-container basis at different times during the day, you are using Cached Quick I/O to dynamically tune the performance of a database.

For example, files that store historical data are not generally used during normal business hours in a transaction processing environment. Reports that make use of this historical data are generally run during off-peak hours when interactive database use is at a minimum. During normal business hours, you can disable Cached Quick I/O for database files that store historical data in order to maximize memory available to other user applications. Then, during off-peak hours, you can enable Cached Quick I/O on the same files when they are used for report generation. This will provide extra memory resources to the database server without changing any database configuration parameters. Enabling file system read-ahead in this manner and buffering read data can provide great performance benefits, especially in large sequential scans.

You can automate the enabling and disabling of Cached Quick I/O on a per-container basis using scripts, allowing the same job that produces reports to tune the file system behavior and make the best use of system resources. You can specify different sets of containers for different jobs to maximize file system and database performance.

Note Modifying Cached Quick I/O settings does not require any database level changes. So, unlike modifying existing database global memory or bufferpool settings, Cached Quick I/O does not require the database to be restarted for changes to take effect.

How Cached Quick I/O Improves Database Performance

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.

Overview of How to Set Up Cached Quick I/O

To set up and use Cached Quick I/O:

1. Enable Cached Quick I/O on the underlying file systems used for your database.

2. Exercise the system in your production environment to generate file I/O statistics.

3. Collect the file I/O statistics while the files are in use.

4. Analyze the file I/O statistics to determine which files benefit from Cached Quick I/O.

5. Disable Cached Quick I/O on files that do not benefit from caching.

The rest of this chapter discusses how to set up Cached Quick I/O in more detail.

Enabling Cached Quick I/O on the 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:

```
# chown root:db2iadm1 /opt/VRTS/bin/vxtunefs
# chmod 4550 /opt/VRTS/bin/vxtunefs
```

  where users belonging to the `db2iadm1` group are granted permission to run the `vxtunefs` command. We recommend 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.

If you have correctly enabled Quick I/O on your system, you can proceed to enable Cached Quick I/O as follows:

◆ Set the file system Cached Quick I/O flag, which enables Cached Quick I/O for all files in the file system.

  Setting the file system Cached Quick I/O flag enables caching for all files in the file system. You must disable Cached Quick I/O on individual Quick I/O files that do not benefit from caching to avoid consuming memory unnecessarily. This final task occurs at the end of the enabling process.

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:

```
# /opt/VRTS/bin/vxtunefs -s -o qio_cache_enable=1 /mount_point
```

Example

```
# /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:

```
# /opt/VRTS/bin/vxtunefs -s -o qio_cache_enable=0 /mount_point
```

Example

```
# /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` is the name of a block device
- *dgname* is the name of the disk group
- *volname* is the name of the volume

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.

For 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`:

```
# /opt/VRTS/bin/vxtunefs /mount_point | grep qio_cache_enable
```

Example

```
# /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.

▼ To obtain information on all vxtunefs system parameters

Use the `vxtunefs` command without `grep`:

```
# /opt/VRTS/bin/vxtunefs /mount_point
```

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 = 1048576
discovered_direct_iosz = 262144
max_direct_iosz = 1048576
default_indir_size = 8192
qio_cache_enable = 1
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
```

For a complete description of `vxtunefs` parameters and the tuning instructions, refer to 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.

Prerequisites

- You must enable Cached Quick I/O for the file systems. See "Enabling Cached Quick I/O on the File System" on page 93.

Usage Notes

- See the `qiostat`(1M) manual page for more information.

Collecting I/O Statistics

▼ **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

Example

To collect I/O statistics from all database containers on file system `/db01`:

```
$ /opt/VRTS/bin/qiostat -l /db01/*
```

Analyzing I/O Statistics

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/tbs1_cont001    6        1          21       4         10.0   0.0
                      6        6         0.0

/db01/tbs2_cont00162552   38498     250213   153992      21.9   0.4
                    62567    49060       21.6

/db01/tbs2_cont00262552   38498     250213   153992      21.9   0.4
                    62567    49060       21.6
```

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/tbs1_cont001 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 files /db01/tbs2_* have 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/tbs2_* files, 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/tbs2_cont001` and `/db01/tbs2_cont002` are prime candidates for Cached Quick I/O. For more information on enabling and disabling Cached Quick I/O at the file level, see "Enabling and Disabling Cached Quick I/O for Individual Files" on page 100.

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 container files that contain a particular table, across multiple tablespaces used by a given database, even if the containers in just one of the tablespaces exhibited a high cache hit ratio. In general, we expect all containers in a tablespace to have approximately the same cache hit ratio.

Using 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 for more information.

Setting Cache Advisories for Individual 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
```

Example

To disable Cached Quick I/O for the file `/db01/dbfile`, set the cache advisory to OFF:

```
$ /opt/VRTS/bin/qioadmin -S dbfile=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
```

Example

Running `qiostat` shows the cache hit ratio for the file `/db01/dbfile` reaches a level that would benefit from caching. To enable Cached Quick I/O for the file `/db01/dbfile`, set the cache advisory to ON:

```
$ /opt/VRTS/bin/qioadmin -S dbfile=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
filename,OFF
filename,OFF
filename,OFF
filename,ON
```

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
dbfile01,OFF
dbfile02,OFF
dbfile03,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
```

Example

To display the current cache advisory setting for the file `dbfile` in the `/db01` file system:

```
$ /opt/VRTS/bin/qioadmin -P dbfile /db01
dbfile,OFF
```

Using Storage Mapping 5

Storage mapping is included with VERITAS Storage Foundation *for DB2* Enterprise Edition.

Storage mapping allows a DB2 database administrator to display detailed storage layer topology information for a DB2 container. The database administrator can view the layout of all the containers for a tablespace to identify potential trouble spots. You may obtain and view detailed storage topology information using the `vxstorage_stats` commands or the VERITAS Storage Foundation *for DB2* GUI.

Topics include:

Understanding Storage Mapping

Access to mapping information is important since it allows for a detailed understanding of the storage hierarchy in which files reside, information that is critical for effectively evaluating I/O performance.

Mapping files to their underlying device is straightforward when containers are created directly on a raw device. With the introduction of host-based volume managers and sophisticated storage subsystems that provide RAID features, however, mapping files to physical devices has become more difficult.

With the VERITAS Storage Foundation *for DB2* Storage Mapping option, you can map containers to physical devices. VERITAS Storage Mapping relies on VERITAS Federated Mapping Service (VxMS), a library that assists in the development of distributed SAN applications that must share information about the physical location of files and volumes on a disk.

The VERITAS Storage Foundation *for DB2* Storage Mapping option supports a wide range of storage devices and allows for "deep mapping" into EMC, Hitachi, and IBM Enterprise Storage Server ("Shark") arrays. Deep mapping information identifies the physical disks that comprise each LUN and the hardware RAID information for the LUNs.

You can view storage mapping topology information and I/O statistics using:

- The `vxstorage_stats` command. This command displays the complete I/O topology mapping of specific containers through intermediate layers like logical volumes down to actual physical devices.
- The VERITAS Storage Foundation *for DB2* GUI. The VERITAS Storage Foundation *for DB2* GUI performs file mapping and displays both storage mapping topology information and I/O statistics.

Verifying VERITAS Storage Mapping Setup

Before using the VERITAS Storage Mapping option, verify that the features are set up correctly.

▼ **To verify that your system is using the VERITAS Storage Mapping option**

1. Verify that you have a license key for the storage mapping option.

```
# /opt/VRTS/bin/vxlictest -n "VERITAS Mapping Services" -f \
  "Found_Edi_map"
  Found_Edi_map feature is licensed
```

2. Verify that the VRTSvxmsa package is installed.

```
# pkginfo -l VRTSvxmsa
```

Output similar to the following is displayed:

```
PKGINST:  VRTSvxmsa
     NAME:  VxMS - VERITAS Mapping Service, Application Libraries.
  CATEGORY:  system,utilities
      ARCH:  sparc
   VERSION: 4.2.1-REV=build218_2004.10.29
   BASEDIR:  /opt
    PSTAMP:  oigpsol0920041029112628
  INSTDATE:  Nov 02 2004 15:22
    STATUS:  completely installed
     FILES:     33 installed pathnames
                8 shared pathnames
                14 directories
                15 executables
                2931 blocks used (approx)
```

Using vxstorage_stats

The `vxstorage_stats` command displays detailed storage mapping information and I/O statistics about an individual VxFS file. The mapping information and I/O statistics are recorded only for VxFS files and VxVM volumes.

In `vxstorage_stats` command output, I/O topology information appears first followed by summary statistics for each object.

The command syntax is as follows:

```
/opt/VRTSdb2ed/bin/vxstorage_stats [-m] [-s] [-i interval
-c count] -f filename
```

Prerequisites

- ◆ You must log in as the instance owner or root.

Usage Notes

- ◆ The `-m` option displays the I/O topology for the specified file.
- ◆ The `-s` option displays the file statistics for the specified file.
- ◆ The `-c` *`count`* option specifies the number of times to display statistics within the interval specified by `-i` *`interval`*.
- ◆ The `-i` *`interval`* option specifies the interval frequency for displaying updated I/O statistics.
- ◆ The `-f` *`filename`* option specifies the file to display I/O mapping and statistics for.
- ◆ For more information, see the `vxstorage_stats`(1m) online manual page.

Finding Container Information for DB2 UDB

Currently, DB2 UDB does not offer an interface to link a database page in a table or index to a particular container's offset. However, the `vxstorage_stats` command or the VERITAS Storage Foundation *for DB2* Graphical User Interface can be used to display the I/O topology. No special setup is needed to view the topology.

▼ To locate a table in a tablespace

The steps below use TABLE01 in database PROD as an example:

1. Connect to the database PROD as the instance owner.

```
$db2 connect to PROD

  Database Connection Information
Database server         = DB2/SUN64 7.2.5
SQL authorization ID    = INST1
Local database alias    = PROD
```

2. Enter the following query to display all the tablespaces in this table. In this example, TBSPACE is the primary tablespace for all the table data, INDEX_TBSPACE contains the index (if any), and LONG_TBSPACE is the tablespace to store LONG column.

```
 $db2 “select tbspace, index_tbspace, long_tbspace from
 syscat.tables where tabname='TABLE01'”

 TBSPACE            INDEX_TBSPACE      LONG_TBSPACE
------------------ ------------------ ------------------
 USER1              -                  -
 1 record(s) selected.
```

3. After locating the tablespace name, enter the following command to find the container path name:

```
$db2 list tablespaces
  ...
  Tablespace ID                        = 3
  Name                                 = USER1
  Type                                 = Database managed space
  Contents                             = Any data
  State                                = 0x0000
  Detailed explanation:
  Normal
  ...
```

The output indicates the Tablespace ID is 3 and the container path name is `USER1`:

```
$db2 list tablespace containers for 3

Tablespace Containers for Tablespace 3
Container ID = 0
Name = /data/system01.dbf
Type = Disk
```

Tablespace `USER1` contains only one container:

```
/data/system01.dbf
```

Displaying Storage Mapping Information for DB2 Containers

▼ To display storage mapping information

After the container information has been determined, use the `vxstorage_stats` command with the `-m` option to display storage mapping information:

```
$/opt/VRTSdb2ed/bin/vxstorage_stats -m -f file_name
```

Example

```
$/opt/VRTSdb2ed/bin/vxstorage_stats -m -f \
/data/system01.dbf
```

Output similar to the following is displayed:

```
TY  NAME                          NSUB    DESCRIPTION    SIZE(sectors)  OFFSET(sectors)
PROPERTIES
v   /dev/vx/rdsk/testdg/stripevol 1       MIRROR         204800         0
pl  vxvm:testdg/stripevol-01      3       STRIPE         205056         0
Stripe_size:128
rd  /dev/vx/rdmp/c1t3d0s2         1       PARTITION      68352          0
sd  /dev/rdsk/c1t3d0s2            1       PARTITION      71127180       0
da  c1t3d0                        0       DISK           71127180       0
rd  /dev/vx/rdmp/c1t4d0s2         1       PARTITION      68352          0
sd  /dev/rdsk/c1t4d0s2            1       PARTITION      71127180       0
da  c1t4d0                        0       DISK           71127180       0
rd  /dev/vx/rdmp/c1t5d0s2         1       PARTITION      68352          0
sd  /dev/rdsk/c1t5d0s2            1       PARTITION      71127180       0
da  c1t5d0                        0       DISK           71127180       0
```

Note For file type (`fi`), the `SIZE` column is number of bytes, and for volume (`v`), plex (`pl`), sub-disk (`sd`), and physical disk (`da`), the `SIZE` column is in 512-byte blocks. Stripe sizes are given in sectors.

Displaying I/O Statistics Information

▼ To display I/O statistics information

Use the `vxstorage_stats` command with the `-s` option to display I/O statistics information:

```
$ /opt/VRTSdb2ed/bin/vxstorage_stats -s -f file_name
```

Example

```
$ /opt/VRTSdb2ed/bin/vxstorage_stats -s -f \
/PRODqio/PRODqiotbs
```

Output similar to the following is displayed:

```
                            I/O OPERATIONS     I/O BLOCKS(512 byte)   AVG TIME(ms)

OBJECT                      READ      WRITE    B_READ     B_WRITE     AVG_RD   AVG_WR
/data/system01.dbf          2         2479     8          5068810     0.00     53.28
/dev/vx/rdsk/mydb/myindex   101       2497     1592       5069056     12.18    52.78
vxvm:mydb/myindex-01        101       2497     1592       5069056     12.18    52.76
/dev/rdsk/c3t1d3s3          131       1656     2096       1689696     14.43    39.09
c3t1d3                      131       1656     2096       1689696     14.43    39.09
EMC000184502242:02:0c:02    8480      231019   275952     23296162    -        -
EMC000184502242:31:0c:02    3244      232131   54808      23451325    -        -
/dev/rdsk/c3t1d15s4         0         1652     0          1689606     0.00     46.47
c3t1d15                     0         1652     0          1689606     0.00     46.47
EMC000184502242:01:0c:02    23824     1188997  1038336    32407727    -        -
EMC000184502242:32:0c:02    5085      852384   135672     29956179    -        -
/dev/rdsk/c3t1d2s4          14        1668     200        1689834     18.57    34.19
c3t1d2                      14        1668     200        1689834     18.57    34.19
EMC000184502242:16:0c:02    4406      271155   121368     23463948    -        -
EMC000184502242:17:0c:02    3290      269281   55432      23304619    -        -
```

▼ To display storage mapping and I/O statistics information at repeated intervals

Use the `vxstorage_stats` command with the `-i` *`interval`* and `-c` *`count`* options to display storage mapping and I/O statistics information at repeated intervals. The `-i` *`interval`* option specifies the interval frequency for displaying updated I/O statistics and the `-c` *`count`* option specifies the number of times to display statistics.

```
$ /opt/VRTSdb2ed/bin/vxstorage_stats [-m] [-s] \
[-i interval -c count ] -f file_name
```

Example

To display statistics twice with a time interval of two seconds:

```
$ /opt/VRTSdb2ed/bin/vxstorage_stats -s -i2 -c2 \
-f /data/system01.dbf
```

Output similar to the following is displayed:

```
                             OPERATIONS     FILE BLOCKS(512 byte)      AVG TIME(ms)

OBJECT                         READ    WRITE     B_READ  B_WRITE     AVG_RD    AVG_WR
/data/system01.dbf             615     19        20752   152         3.53      24.74
/dev/vx/rdsk/mapdg/data_vol 19386   33227     895692  1376438     9.27      16.18
vxvm:mapdg/data_vol-01         19386   33227     895692  1376438     9.26      14.03
/dev/rdsk/c1t10d0s2            19386   33227     895692  1376438     9.26      14.03
c1t10d0                        19386   33227     895692  1376438     9.26      14.03
vxvm:mapdg/data_vol-03         0       33227     0       1376438     0.00      14.21
/dev/rdsk/c1t13d0s2            0       33227     0       1376438     0.00      14.21
c1t13d0                        0       33227     0       1376438     0.00      14.21
                             OPERATIONS     FILE BLOCKS(512 byte)      AVG TIME(ms)

OBJECT                         READ    WRITE     B_READ  B_WRITE     AVG_RD    AVG_WR
/data/system01.dbf             0       0         0       0           0.00      0.00
/dev/vx/rdsk/mapdg/data_vol 0       1         0       2           0.00      0.00
vxvm:mapdg/data_vol-01         0       1         0       2           0.00      0.00
/dev/rdsk/c1t10d0s2            0       1         0       2           0.00      0.00
c1t10d0                        0       1         0       2           0.00      0.00
vxvm:mapdg/data_vol-03         0       1         0       2           0.00      0.00
/dev/rdsk/c1t13d0s2            0       1         0       2           0.00      0.00
c1t13d0                        0       1         0       2           0.00      0.00
```

Viewing Storage Mapping Topology and I/O Statistics Using the GUI

▼ To view DB2 container storage mapping topology and I/O statistics using the GUI

1. Start the GUI and connect to the desired host.

2. Expand the DB2 Instances icon in the object tree and then expand the desired database.

3. Under the desired database, select a container in the object tree. (You will need to further expand the tree view to find the icon.)

4. Use one of the following methods to generate datafile statistics.

   - Click on **Container**> **Topology/Statistic**.

     or

   - Right click on the container to bring up a pop-up menu. Then click on **Topology/Statistic**.

     or

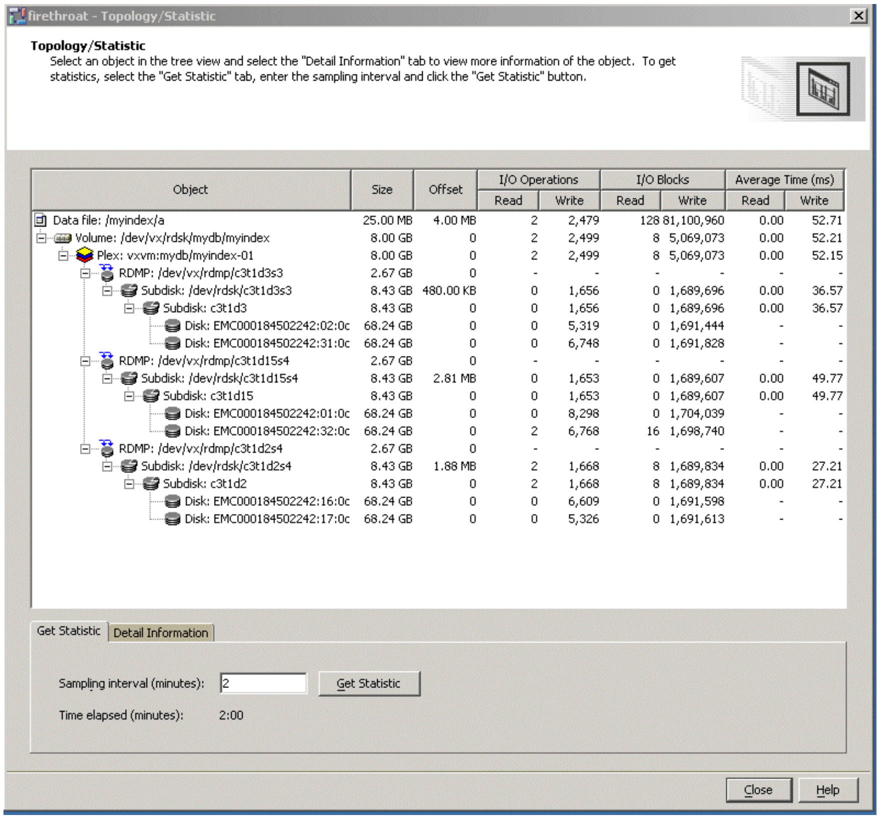
   - Click on the **Topology/Statistic** icon in the main toolbar.

   The **Topology/Statistic** window is displayed.

5. To view more information about an object, select an object in the object tree and then click on the **Detailed Information** tab at the bottom of the **Topology/Statistic** window.

6. Optionally, if you want to view I/O statistics, enter a sampling time (in minutes) in the **Sampling interval (minutes)** field and then press **Get Statistics**.

Information is displayed in the **I/O Operations**, **I/O Blocks**, and **Average Time (ms)** columns of the window.



Topology/Statistic

Select an object in the tree view and select the "Detail Information" tab to view more information of the object. To get statistics, select the "Get Statistic" tab, enter the sampling interval and click the "Get Statistic" button.

Object	Size	Offset	I/O Operations		I/O Blocks		Average Time (ms)	
			Read	Write	Read	Write	Read	Write
Data file: /myindex/a	25.00 MB	4.00 MB	2	2,479	128	81,100,960	0.00	52.71
Volume: /dev/vx/rdsk/mydb/myindex	8.00 GB	0	2	2,499	8	5,069,073	0.00	52.21
Plex: vxvm:mydb/myindex-01	8.00 GB	0	2	2,499	8	5,069,073	0.00	52.15
RDMP: /dev/vx/rdmp/c3t1d3s3	2.67 GB	0	-	-	-	-	-	-
Subdisk: /dev/rdsk/c3t1d3s3	8.43 GB	480.00 KB	0	1,656	0	1,689,696	0.00	36.57
Subdisk: c3t1d3	8.43 GB	0	0	1,656	0	1,689,696	0.00	36.57
Disk: EMC000184502242:02:0c	68.24 GB	0	0	5,319	0	1,691,444	-	-
Disk: EMC000184502242:31:0c	68.24 GB	0	0	6,748	0	1,691,828	-	-
RDMP: /dev/vx/rdmp/c3t1d15s4	2.67 GB	0	-	-	-	-	-	-
Subdisk: /dev/rdsk/c3t1d15s4	8.43 GB	2.81 MB	0	1,653	0	1,689,607	0.00	49.77
Subdisk: c3t1d15	8.43 GB	0	0	1,653	0	1,689,607	0.00	49.77
Disk: EMC000184502242:01:0c	68.24 GB	0	0	8,298	0	1,704,039	-	-
Disk: EMC000184502242:32:0c	68.24 GB	0	2	6,768	16	1,698,740	-	-
RDMP: /dev/vx/rdmp/c3t1d2s4	2.67 GB	0	-	-	-	-	-	-
Subdisk: /dev/rdsk/c3t1d2s4	8.43 GB	1.88 MB	2	1,668	8	1,689,834	0.00	27.21
Subdisk: c3t1d2	8.43 GB	0	2	1,668	8	1,689,834	0.00	27.21
Disk: EMC000184502242:16:0c	68.24 GB	0	0	6,609	0	1,691,598	-	-
Disk: EMC000184502242:17:0c	68.24 GB	0	0	5,326	0	1,691,613	-	-

Sampling interval (minutes): 2

Time elapsed (minutes): 2:00

7. When you are finished, click **Close**.

Configuring Arrays for Storage Mapping and Statistics

VERITAS Storage Foundation *for DB2* provides "deep" mapping information and performance statistics for supported storage arrays. Deep mapping information consists of identifying the physical disks that comprise each LUN and the hardware RAID information for the LUNs.

VERITAS Array Integration Layer (VAIL) software interfaces third-party hardware storage arrays with VERITAS storage software. VAIL providers are software modules that enable VERITAS applications to discover, query, and manage third-party storage arrays.

On Solaris, the following VAIL providers support these third-party storage arrays:

- The `vx_hicommand` provider manages Hitachi arrays.
- The `vx_emc_symmetrix` provider manages EMC Symmetrix arrays.
- The `vx_ibmshark` provider manages IBM ESS (Shark) arrays.

For the most up-to-date array support information, see the appropriate hardware compatibility list (HCL) on the VERITAS Technical Support Web page at:

http://support.veritas.com

If you want to use storage array information accessible through the VAIL providers, install VAIL and perform any required configuration for the storage arrays and VAIL providers. To use deep mapping services and performance statistics for supported storage arrays, you must install both VAIL and VERITAS Mapping Services (VxMS).

You will need to install required third-party array CLIs and APIs on the host where you are going to install VAIL before you install VAIL. If you install any required CLI or API after you install VAIL, rescan the arrays so that VERITAS Storage Foundation *for DB2* can discover them.

For detailed information about supported array models, see the *VERITAS Array Integration Layer Array Configuration Guide*.

Converting Existing Database Configurations to VxFS 6

You can convert existing database configurations to VERITAS File System. This chapter describes how to migrate UFS file systems, earlier version layouts, and raw devices to current VxFS file systems.

Topics include:

Converting From UFS to VxFS With Quick I/O

If you are currently using UFS file systems, you can use the following procedure to upgrade each file system used by the database to a VxFS file system with Quick I/O.

▼ To convert a UFS file system to VxFS with Quick I/O

1. Make the DB2 database inactive by either shutting down the database or disabling all user connections.

2. Create a backup of the UFS file system.

3. Unmount the UFS file system.

4. Remove the entry for the UFS file system from the `/etc/vfstab` directory.

5. Create a VxFS file system of the same size as the original UFS file system, using the mount point where the UFS file system was originally mounted. Use the procedure described in "Creating a VxFS File System" on page 47 to create a VxFS file system.

6. Preallocate Quick I/O files using `qiomkfile`. Use the procedure described in "Creating Database Containers as Quick I/O Files Using qiomkfile" on page 68.

7. Restore the backup created in step 2 on page 116 to the Quick I/O files in the new VxFS file system.

8. Reactivate the DB2 database.

Upgrading From Earlier VxFS Version Layouts

Prerequisites

- Perform a full backup of the file system before upgrading to a new disk layout.

Usage Notes

- The `vxupgrade` command lets you to upgrade the VxFS file system disk layout while the file system is mounted. See the `vxupgrade`(1M) manual page for more details.
- VxFS supports three file system disk layouts: Versions 4, 5, and 6. New file systems are created with the Version 6 (for large file systems) disk layout by default when the current version of VERITAS Storage Foundation *for DB2* is installed on a system. You must minimally upgrade to Version 4 disk layout if you want to use the Storage Rollback.

▼ To upgrade an existing VxFS file system to a new file system disk layout version

Use the `vxupgrade` command to upgrade to Version 4, 5, or 6 disk layout:

```
# /opt/VRTS/bin/vxupgrade -n new_version /mount_point
```

where:

- *`new_version`* is the version of the file system disk layout you want to upgrade to
- */`mount_point`* is the location where the file system is mounted

Example

This is an example of upgrading to disk layout Version 6.

```
# /opt/VRTS/bin/vxupgrade -n 6 /db01
```

▼ To use Quick I/O after upgrading the file system disk layout to Version 4, 5, or 6

1. Make the DB2 database inactive by either shutting down the database or disabling all user connections.
2. Make each database file accessible as a Quick I/O file.

   See "Accessing Regular VxFS Files as Quick I/O Files" on page 72 for more information.
3. Reactivate the DB2 database.

Converting From Raw Devices

If the database is currently using raw disks or volumes, use one of the following procedures to use VxFS with the Quick I/O feature.

If the database is currently using raw disks or volumes, you can do the conversion using DB2 *redirected restore*. DB2 *redirected restore* allows you to redefine or redirect data to new tablespace containers during a restore. Use the following procedure to use VxFS with the Quick I/O feature. For more information on DB2 backup and restore, refer to the DB2 Administration Guide or DB2 Command Reference.

Note The procedure provided assumes the database runs on a single file system after the upgrade.

▼ To convert from raw devices to VxFS with Quick I/O

1. Create a VxFS file system using a size that is 10 percent larger than the original database or total raw device size.

Use the procedure described in "Creating a VxFS File System" on page 47 to create a new VxFS file system. You can create more file systems based on your performance and availability requirements.

2. Verify that the database can perform rollforward recovery. In order to do this, the database parameters LOGRETAIN and/or USEREXIT must be enabled. For example:

```
$ db2 get db cfg for DATABASE | egrep "LOGRET|USEREXIT"
$ db2 update db cfg for DATABASE USING LOGRETAIN ON
$ db2 update db cfg for DATABASE USING USEREXIT ON
```

3. Preallocate Quick I/O files using `qiomkfile`. Use the procedure described in "Creating Database Containers as Quick I/O Files Using qiomkfile" on page 68.

4. Backup the existing database.

For example, use the DB2 `backup database` command to dump all data:

```
$ db2 backup database DATABASE to /dumpdir
```

5. Stop the DB2 instance and then restart.

For example:

```
$ db2 db2stop force
$ db2 db2start
```

6. Restore the database to the newly defined Quick I/O files:

```
$ db2 restore db DATABASE from /dumpdir taken at TIME \
      replace existing redirect
```

7. Redefine the storage that the database is now to use:

```
$ db2 set tablespace containers for TBS_ID using \
      (device '/qio_file_path' TBS_SIZE)
```

8. Continue the restore process:

```
$ db2 restore db DATABASE continue
$ db2 rollforward db DATABASE to end of logs
$ db2 rollforward db DATABASE complete
```

Example (redirected restore)

Create a VM volume on VM group PRODdg (as root)

```
# vxassist -g PRODdg make newdata1 500m
```

Create a new VxFS file system and mount (as root)

```
# mkdir /newdata1
# mkfs -F vxfs /dev/vx/rdsk/PRODdg/newdata1
# mount -F vxfs /dev/vx/rdsk/PRODdg/newdata1 /newdata1
# chown -R db2inst1:db2iadm /newdata1
```

Check on required settings and prepare new containers

```
# su - db2inst1
$ db2 connect to PROD
$ db2 list tablespaces
$ db2 update db cfg for PROD USING LOGRETAIN ON
$ db2 update db cfg for PROD USING USEREXIT ON
$ db2 terminate

$ cd /newdata1
$ /opt/VRTSvxfs/sbin/qiomkfile -s 200m data_container_001
```

Backup the database and restore redirect into the new DMS devices (Quick I/O files)

```
$ db2 backup database PROD to /dump_device
timestamp: 20010828121254

$ db2stop force
$ db2start
$ db2 restore db PROD from /dump_device taken at \
      20010828121254 replace existing redirect
```

Check for tablespaces state (restore pending, storage must/may be defined)

```
$ db2 connect to PROD
$ db2 list tablespaces show detail
```

Issue a SET TABLESPACE CONTAINERS command for each tablespace whose containers must be redefined. For example: tablespace container 5 is going to be redirected to DMS device data_container_001 under /newdata1 directory.

```
$ db2 set tablespace containers for 5 using (device \
      '/newdata1/data_container_001' 50000)
$ db2 restore db PROD continue
$ db2 rollforward db PROD to end of logs
$ db2 rollforward db PROD complete
$ db2 connect to PROD
```

Verify the conversion

```
$ db2 list tablespace containers for 5
```

Note The procedure provided above only converts one tablespace container. Repeat the `set tablespace containers` step as necessary for other tablespaces.

An alternative migration strategy is to use the db2move command to export and import specific tables from a database. This command is used as follows:

Example (db2move migration of table *cust*)

```
$ db2move PROD export -tn cust
$ db2 connect to PROD
$ db2 drop table cust
$ db2 create table cust (cust_no char(6) not null, \
      cust_name char(20) not null) in new_tablespace
$ db2 terminate
$ db2move PROD load -l /data1
```

Using Storage Checkpoints and Storage Rollback 7

The VERITAS Storage Checkpoint feature is available with the Enterprise Edition as part of the VERITAS File System package and is used for the efficient backup and recovery of DB2 databases, including partitioned databases in an SMP environment. Storage Checkpoints can also be mounted, allowing regular file system operations to be performed. This chapter describes what Storage Checkpoints and storage rollback are and how to make use of these technologies through VERITAS Storage Foundation *for DB2*.

Topics covered in this chapter include:

Using Storage Checkpoints and Storage Rollback for Backup and Restore

VERITAS Storage Foundation *for DB2* provides a Storage Checkpoint facility that is similar to the snapshot file system mechanism; however, a Storage Checkpoint persists after a system reboot. A Storage Checkpoint creates an exact image of a database instantly and provides a consistent image of the database from the point in time the Storage Checkpoint was created. The Storage Checkpoint image is managed and available through the GUI or the VERITAS Storage Foundation *for DB2* command line interface (CLI).

Note For more information on creating Storage Checkpoints with the CLI, see "VERITAS Storage Foundation for DB2 Command Line Interface" on page 353.

A direct application of the Storage Checkpoint facility is Storage Rollback. Because each 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 blocks contained in a Storage Checkpoint into the primary file system for restoring the database faster. For more information on Storage Checkpoints and Storage Rollback, see the *VERITAS File System Administrator's Guide*.

Understanding Storage Checkpoints and Storage Rollback

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 unless a Storage Checkpoint allocation policy is assigned. A Storage Checkpoint can be mounted as read-only or read-write, allowing access to the files as if it were a regular file system. A Storage Checkpoint is created using the `db2ed_ckptcreate` command.

Initially, a Storage Checkpoint contains no data—it contains only 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 needed 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 *n* 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.

Storage Rollback restores a database on the primary file systems to the point-in-time image created during a Storage Checkpoint. Storage Rollback is accomplished by copying the “before” images from the appropriate Storage Checkpoint back to the primary file system. As with Storage Checkpoints, Storage Rollback restores at the block level, rather than at the file level. Storage Rollback is executed using the `db2ed_ckptrollback` command.

Note Whenever you change the structure of the database (for example, by adding or deleting containers), you must run `db2ed_update`.

Mountable Storage Checkpoints can be used for a wide range of application solutions, including backup, investigations into data integrity, staging upgrades or database modifications, and data replication solutions.

If you mount a Storage Checkpoint as read-write, the `db2ed_ckptrollback` command and GUI will not allow you to roll back to this Storage Checkpoint. VERITAS Storage Foundation *for DB2* can no longer guarantee that the Storage Checkpoint data still represents the correct point-in-time image. A subsequent rollback from this Storage Checkpoint could corrupt the database. When a Storage Checkpoint is to be mounted as read-write, the `db2ed_ckptmount` command creates a “shadow” Storage Checkpoint and mounts it as read-write. This allows the database to still be rolled back to the original Storage Checkpoint.

Note For more information on mountable Storage Checkpoints, see “VERITAS Storage Foundation for DB2 Command Line Interface” on page 353.

Determining Space Requirements for Storage Checkpoints

To support storage rollback, the file systems need extra disk space to store the Storage Checkpoints. The extra space needed depends on how the Storage Checkpoints are used. Storage Checkpoints that are used to keep track of the block changes contain only file system block maps, and therefore require very little additional space (less than 1 percent of the file system size).

To support Storage Checkpoints and storage rollback, VxFS needs to keep track of the original block contents when the Storage Checkpoints were created. The additional space needed is proportional to the number of blocks that have been changed since a Storage Checkpoint was taken. The number of blocks changed may not be identical to the number of changes. For example, if a data block has been changed many times, only the first change requires a new block to be allocated to store the original block content. Subsequent changes to the same block require no overhead or block allocation.

If a file system that has Storage Checkpoints runs out of space, by default VxFS removes the oldest Storage Checkpoint automatically instead of returning an `ENOSPC` error code (UNIX `errno 28- No space left on device`), which can cause the DB2 database to fail. Removing Storage Checkpoints automatically ensures the expected I/O semantics, but at the same time, eliminates a key recovery mechanism. You can set the Storage Checkpoint to be non-removable when you create it. In this case, an `ENOSPC` will return to DB2. The DBA will need to grow the VxFS file system size to work around the problem.

When restoring a file system that has data-full Storage Checkpoints from tape or other offline media, you need extra free space on the file system. The extra space is needed to accommodate the copy-on-write algorithm needed for preserving the consistent image of the Storage Checkpoints. The amount of free space required depends on the size of the restore and the number of Storage Checkpoints on the file system.

If you are restoring the entire file system, in most cases, you no longer need the existing Storage Checkpoint. You can simply re-make the file system using the `mkfs` command, and then restore the file system from tape or other offline media.

If you are restoring some of the files in the file system, you should first remove the data-full Storage Checkpoints that are no longer needed. If you have very limited free space on the file system, you may have to remove all data-full Storage Checkpoints in order for the restore to succeed.

To avoid unnecessary Storage Checkpoint removal, use the VxDBA utility to set up a Monitoring Agent to monitor file system space usage. When file system space usage exceeds a preset threshold value (say, 95 percent full), the Monitoring Agent alerts the system administrator and optionally grows the volume and the file system. Automatic notifications to the system administrator on the status of space usage and file system resizing are available through electronic mail, the `syslogd`(1M) program, or by logging messages to a simple log file. See "Managing File System Space" on page 306 for more information.

Always reserve free disk space for growing volumes and file systems. You can also preallocate sufficient space for each file system when the file system is first created or manually grow the file system and logical volume where the file system resides. See the `vxassist`(1) and `fsadm_vxfs`(1) manual pages for more information.

Performance of Storage Checkpoints

VERITAS File System attempts to optimize the read and write access performance on both the Storage Checkpoint and the primary file system. Reads from a Storage Checkpoint typically perform at nearly the throughput of reads from a normal VxFS file system, allowing backups to proceed at the full speed of the VxFS file system.

Writes to the primary file system are typically affected by the Storage Checkpoints because the initial write to a data block requires a read of the old data, a write of the data to the Storage Checkpoint, and finally, the write of the new data to the primary file system. Having multiple Storage Checkpoints on the same file system, however, will not make writes slower. Only the initial write to a block suffers this penalty, allowing operations like writes to the intent log or inode updates to proceed at normal speed after the initial write.

The performance impact of Storage Checkpoints on a database is less when the database files are Quick I/O files. A performance degradation of less than 5 percent in throughput has been observed in a typical OLTP workload when the Storage Checkpoints only keep track of changed information. For Storage Checkpoints that are used for storage rollback, higher performance degradation (approximately 10 to 20 percent) has been observed in an OLTP workload. The degradation should be lower in most decision-support or data-warehousing environments.

Reads from the Storage Checkpoint are impacted if the primary file system is busy, because the reads on the Storage Checkpoint are slowed by all of the disk I/O associated with the primary file system. Therefore, performing database backup when the database is less active is recommended.

Storage Checkpoint Allocation Policies

VERITAS File System provides Multi-Volume File Systems (MVS) when used in conjunction with the Volumes Set feature in VERITAS Volume Manager. A volume set is a container for multiple different volumes. MVS enables creation of a single file system over multiple volumes, each volume with properties of its own. This helps administrators specify which data goes on which volume types. For more details about MVS, see *VERITAS Volume Manager Administrator's Guide*. Setting up a storage configuration for MVS operations is a system administrator's responsibility and requires superuser (`root`) privileges.

Multi-Volume File Systems provide, a database administrator, through the SFUA checkpoint administration interface, the ability to create Storage Checkpoint Allocation Policies. A Storage Checkpoint Allocation policy specifies a list of volumes and the order in which to attempt allocations. Once defined, a database administrator can use these policies to:

- Control where the storage checkpoint should be created enabling separation of metadata and data of a storage checkpoint to different volumes.
- Separate storage checkpoints so that data allocated to a storage checkpoint is isolated from the primary file system. This helps control the space used by the checkpoint and prevents the checkpoint from fragmenting the space in the primary fileset.

When policies are assigned to a storage checkpoint, the database administrator must specify the mapping to both metadata and file data. If no policies are specified for the storage checkpoint, the data is placed randomly within the primary file system. Data and metadata of storage checkpoints can have different policies assigned to them or use the same policy to be applied to data and metadata. Multiple checkpoints can be assigned the same checkpoint allocation policy. A *partial* policy is also allowed; a *partial* policy means that the policy does not exist on all file systems used by the database.

Once the policy is assigned to checkpoints, the allocation mechanism attempts to satisfy the request from each device in the policy in the order the devices are defined. If the request cannot be satisfied from any of the devices in the policy, the request will fail, even if other devices exist in the file system which have space. Only those devices can provide allocation that are listed in the policy. This implementation is the mechanism for preventing allocation requests from using space in other devices which are not specified in the policy. It is recommended that you allocate sufficient space for the volumes defined in the Storage Checkpoint policy or update the policy to include additional volumes. This also helps in retaining the old Storage Checkpoints.

Once the assigned policy is deleted, the allocation for metadata and file data for subsequent requests of storage checkpoint will return to the *no policy* assigned state.

Usage Notes

- Since the checkpoint policies feature is associated with MVS file system, it is available only on file systems using disk layout Version 6.
- Storage checkpoint allocation policy requires VxVM Volume Set and VxFS Multi-Volume File Systems features to be enabled. These features are included in the Enterprise Edition of Storage Foundation. Refer to the Multi-Volume File System chapter in the *VERITAS File System Administrator's Guide*, for creating Volume Sets and MVS file systems for the primary file systems used by the database datafiles.
- Data allocation is done by the volumes in the order that was assigned in the policy.
- The maximum length of an checkpoint policy name is 64 characters.

Using Storage Checkpoint Allocation Policies

You can use the `db2ed_ckptpolicy` command to administrate the storage checkpoint allocation policies. For detailed information on administrating Storage Checkpoint Policy via the CLI, see "VERITAS Storage Foundation for DB2 Command Line Interface" on page 353.

Note You cannot administer Storage Checkpoint Allocation policies through the VxDBA utility menu or the GUI.

Usage Notes

See the `db2ed_ckptpolicy(1M)` and `db2ed_ckptcreate(1M)` manual pages for more information. `db2ed_ckptpolicy` command need to be executed by the DB2 database administrator.

In the following example, the file systems for database datafiles are set up as follows:

- Two MVS file systems `/mvsfs/v1` and `/mvsfs/v2` used for database datafiles.
- File system `/mvsfs/v1` is created on volume set `mvsvset1`.
- File system `/mvsfs/v2` is created on volume set `mvsvset2`.
- Volume set `mvsvset1` contains volumes `mvsv1`, `mvsv2`, and `mvsv3` .
- Volume set `mvsvset2` contains volumes `mvsv4` and `mvsv5` .

Usage

Use the `db2ed_ckptpolicy` command with the following options.

```
$ db2ed_ckptpolicy -D DB2DATABASE  [ -I DB2INSTANCE ] \
  [-n] [-h] options
```

Where *options* could be any of the following parameters:

```
-o create|update|remove -p ckpt_sample
-o display [-c ckpt_name | -p ckpt_sample]
-o assign -c ckpt_name -p ckpt_data_policy[,ckpt_metadata_policy]
```

Creating a Storage Checkpoint Allocation policy

▼ To create a Storage Checkpoint allocation policy

Use the `db2ed_ckptpolicy` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptpolicy -D DB2DATABASE \
-o create -p ckpt_policy
```

Note A *partial* policy indicates that the Storage Checkpoint allocation policy does not include all the file systems used by the database.

Example

In the following example, the database name is PROD and the Storage Checkpoint allocation policy that is created is named ckpt_sample.

```
$ db2ed_ckptpolicy -D PROD -o create -p ckpt_sample
```

Output similar to the following is displayed.

```
File System: /mvsfs/v2 (MVS volumes: mvsv4,mvsv5)
Assigned Data Policy: NONE (MVS Volumes: N/A)
Assigned Meta Data Policy: NONE (MVS Volumes: N/A)
Please enter the volume name(s), sperated by space, for the policy
ckpt_sample [skip,quit]: mvsv4

File System: /mvsfs/v1 (MVS volumes: mvsv1,mvsv2,mvsv3)
Assigned Data Policy: NONE (MVS Volumes: N/A)
Assigned Meta Data Policy: NONE (MVS Volumes: N/A)
Please enter the volume name(s), sperated by space, for the policy
ckpt_sample [skip,quit]: mvsv2

The following information will be used to create policy ckpt_sample
ckpt_sample         /mvsfs/v2            mvsv4
ckpt_sample         /mvsfs/v1            mvsv2
```

Assigning a Storage Checkpoint Allocation policy

You can use either of the following methods to assign an allocation policy to a Storage Checkpoint:

- Use the `db2ed_ckptpolicy` command to assign allocation policies to an existing storage checkpoint.
- Use `[ -p ckpt_data_policy[,ckpt_metadata_policy]]` option in `db2ed_ckptcreate` command to supply policies when executed.

Note `db2ed_ckptcreate` command automaticlly assigns the policies when the storage checkpoint is created.

▼ To assign an allocation policy to an existing Storage Checkpoint

The following procedure uses `db2ed_ckptpolicy` to assign an allocation policy to an existing Storage Checkpoint. This example uses PROD as the database name and Checkpoint_1096060202 as a sample Storage Checkpoint.

1. Create an online Storage Checkpoint for database PROD.

```
$ db2ed_ckptcreate -D PROD -o online
```

   As a result, Checkpoint_1096060202 is created.

2. Assign a Storage Checkpoint policy to the Checkpoint_1096060202.

```
$ db2ed_ckptpolicy -D PROD -n -o assign -c Checkpoint_1096060202 \
  -p ckpt_data,ckpt_metadata
```

3. Display the details of the Storage Checkpoint allocation policy assigned to Checkpoint_1096060202.

```
$ db2ed_ckptpolicy -D PROD -n -o display -c Checkpoint_1096060202
```

   Output similar to the following is displayed:

```
Storage Checkpoint   File System   Data Policy   Meta Data Policy
------------------   -----------   -----------   ----------------
Checkpoint_1096060202 /mvsfs/v2    ckpt_data     ckpt_metadata
Checkpoint_1096060202 /mvsfs/v1    ckpt_data     ckpt_metadata
```

▼ **To assign an allocation policy to a new Storage Checkpoint**

1. Use `db2ed_ckpcreate` to assign an allocation policy to a new Storage Checkpoint.

   ```
   $ db2ed_ckptcreate -D PROD -o online -p ckpt_data,ckpt_metadata
   ```

   As a result, a Storage Checkpoint allocation policy is assigned to Checkpoint_1096060122, which is a new Storage Checkpoint.

2. Display the details of the Storage Checkpoint allocation policy assigned to checkpoint 1096060122.

   ```
   $ db2ed_ckptpolicy -D PROD -n -o display -c Checkpoint_1096060122
   ```

Output similar to the following is displayed:

```
Storage Checkpoint   File System   Data Policy   Meta Data Policy
------------------   -----------   -----------   ----------------
Checkpoint_1096060122 /mvsfs/v2    ckpt_data     ckpt_metadata
Checkpoint_1096060122 /mvsfs/v1    ckpt_data     ckpt_metadata
```

Displaying a Storage Checkpoint Allocation policy

▼ **To display a Storage Checkpoint allocation policy**

Use the `-o display` option to list all the Storage Checkpoint allocation policies contained in the file systems used by the database.

Usage

```
$ db2ed_ckptpolicy -D DB2DATABASE -n -o display
```

Output similar to the following is displayed:

```
Policy Name          File System Coverage
------------------   --------------------
ckpt                 Complete
ckpt_data            Complete
ckpt_metadata        Complete
new_ckpt             Partial
ckpt_sample          Complete
```

Note `Partial` in the File System Coverage column indicates that the Storage Checkpoint allocation policy does not include one or more of the file systems used by the database.

▼ To display Storage Checkpoint Allocation Policy information

Use the `-o display -p` *`checkpointpolicy_name`* option to display information related to a specific Storage Checkpoint allocation policy.

Usage

```
$ db2ed_ckptpolicy -D DB2DATABASE -n -o display \
  -p checkpointpolicy_name
```

Output similar to the following is displayed:

```
Policy Name     File System   MVS volumes
------------    -----------   -----------
ckpt_sample     /mvsfs/v2     mvsv4
ckpt_sample     /mvsfs/v1     mvsv2
```

▼ To display the allocation policies assigned to a Storage Checkpoint

Use the `-o display -c` *`checkpoint_xxxxxxxxx`* option to display the allocation policies assigned to the Storage Checkpoint.

Usage

```
$ db2ed_ckptpolicy -D DB2DATABASE -n -o display \
  -c Checkpoint_xxxxxxxxxx
```

Output similar to the following is displayed:

```
Storage Checkpoint     File System   Data Policy   Meta Data Policy
---------------------  -----------   -----------   ---------------
Checkpoint_1095125037  /mvsfs/v2     ckpt_data     ckpt_metadata
Checkpoint_1095125037  /mvsfs/v1     ckpt_data     ckpt_metadata
```

Updating a Storage Checkpoint Allocation Policy

▼ To update a Storage Checkpoint Allocation Policy

Use the `-o update -p` *`checkpoint_policy_name`* option to update an allocation policy.

Example

```
$ dbed_ckptpolicy -D PROD DB2DATABASE -o update \
  -p checkpointpolicy_name
```

Output similar to the following is displayed:

```
File System: /mvsfs/v2 (MVS volumes: mvsv4,mvsv5)
Policy: ckpt_sample (MVS volumes: mvsv4)
Please enter the volume name(s), separated by space, for the policy
ckpt_sample [skip,quit]: mvsv5

File System: /mvsfs/v1 (MVS volumes: mvsv1,mvsv2,mvsv3)
Policy: ckpt_sample (MVS volumes: mvsv2)
Please enter the volume name(s), separated by space, for the policy
ckpt_sample [skip,quit]: mvsv2,mvsv3

The following information will be used to create policy ckpt_sample
ckpt_sample     /mvsfs/v2     mvsv5
ckpt_sample     /mvsfs/v1     mvsv2,mvsv3
where mvsv4 is the volume currently using the allocation policy.
```

Note The output displays the volumes that are currently assigned in the Storage Checkpoint allocation policy.

Note You are prompted to enter any new volumes to which you would want to assign the Storage Checkpoint allocation policy.

Removing a Storage Checkpoint Allocation Policy

▼ To remove a Storage Checkpoint Allocation Policy

Use the following command to remove a Storage Checkpoint allocation policy:

```
$ db2ed_ckptpolicy -D DB2DATABASE -n -o remove \
  -p checkpointpolicy_name
```

Converting from Regular VxFS file system to MVS

The following procedure describes the conversion of a regular VxFS file system to MVS file system and optionally add new volume to it. Converting a regular VxFS to MVS requires superuser (`root`) privileges. For further details on creating and administrating the VxVM Volume Sets and VxFS Multi-Volume Files System, refer to the *VERITAS Volume Manager Administrator's Guide* and *VERITAS File System Administrator's Guide*.

▼ To convert from a regular VxFS file system to MVS:

1. Select a non-MVS file system to convert to MVS and unmount it.

   ```
   # umount /mnt1
   ```

2. Create the volume set.

   ```
   # vxvset -g dgname make myvset old_vol
   ```

3. Mount the volume set.

   ```
   # mount -F vxfs /dev/vx/dsk/dgname/myvset /mnt1
   ```

4. Upgrade the volume set's file system to Version 6 disk layout. See *VERITAS File System Installation Guide* and the `vxfsconvert(1M)` and `vxupgrade(1M)` manual pages for information on upgrading VxFS disk layouts.

   ```
   # vxupgrade -n 6 /mnt1
   ```

5. Add the new volume to the volume set.

   ```
   # vxvset -g dgname addvol myvset new_vol
   ```

6. Add the new volume to the file system. You must specify the size of the volume.

   ```
   # fsvoladm add /mnt1 new_vol 2g
   ```

   where `new_vol` is the name of the newly added volume and `2g` is the size of the volume.

7. Verify the new volume in the file system.

   ```
   # fsvoladm list /mnt1
   ```

Partitioned Databases and Version Checkpoints

VERITAS Storage Foundation *for DB2* provides commands for creating and using Storage Checkpoints in a DB2 Universal Database (UDB) partitioned database environment as well as an Enterprise Edition (EE) environment.

DB2 UDB Extended Enterprise Edition (EEE) and Enterprise Server Edition (ESE) support data partitioning across clusters of computers. You can install DB2 EEE or ESE with multiple partitions on a single system or you can distribute it on multiple systems.

In a partitioned database environment, Storage Checkpoints that apply to multiple partitions in an SMP environment are referred to as *Version Checkpoints*. VERITAS Storage Foundation *for DB2* supports SMP environments. It does not support MPP environments.

VERITAS Storage Foundation *for DB2* commands that apply to multiple partitions end with the suffix `_all`. For example, `db2ed_ckptcreate_all` creates a Version Checkpoint for a partitioned DB2 database by calling `db2ed_ckptcreate` on every database partition.

Backing Up and Recovering the Database Using Storage Checkpoints

Storage Checkpoints can be created by specifying one of the following options: online or offline. To create a Storage Checkpoint with the online option, the instance should be online and you must enable the LOGRETAIN and/or USEREXIT database configuration parameters. For the offline option, the database should be offline.

During the creation of the Storage Checkpoint, the database is placed in suspended mode. You can roll back the entire database to an online or offline Storage Checkpoint. After the rollback is complete, you may roll the database forward to restore the database if you have used an online Storage Checkpoint.

To allow the easiest recovery, always keep the LOGRETAIN and/or USEREXIT database configuration parameters enabled, regardless of whether the database is online or offline when you create Storage Checkpoints.

Verifying a Storage Checkpoint Using the Command Line

After creating a Storage Checkpoint and before using it to back up or restore a database, you can verify that the Storage Checkpoint is free of errors using the procedure below.

Usage Notes

- See the `db2ed_ckptcreate`(1M) and `db2ed_ckptmount`(1M) manual pages for more information.
- For a database environment with multiple partitions in an SMP environment, see the `db2ed_ckptcreate_all`(1M) and `db2ed_ckptmount_all`(1M) manual pages for more information.

▼ To verify that a Storage Checkpoint is error-free using the command line

1. Create and mount a Storage Checkpoint:

```
$ /opt/VRTS/bin/db2ed_ckptcreate -I db2inst -D PROD -o online
Creating online Storage Checkpoint of database PROD.
Storage Checkpoint Checkpoint_903937870 created.

$ mkdir /tmp/ckpt_ro

$ /opt/VRTS/bin/db2ed_ckptmount -I db2inst -D PROD \
-c Checkpoint_903937870 -m /tmp/ckpt_ro
```

Note If the specified mount point directory does not exist, then `db2ed_ckptmount` creates it before mounting the Storage Checkpoint, as long as the DB2 instance owner has permission to create it.

2. Examine the content of the Storage Checkpoint:

```
$ ls -l /tmp/ckpt_ro/db2vol_82/db2inst1
drwxr-xr-x 3  db2inst1dba  1024       Nov 11 2000 .
drwxr-xr-x 3  db2inst1dba512     Nov 16 11:00 ..
-rw-r--r-- 1  db2inst1dba  209747968  Nov 16 10:58 .tstmp
-rw-r--r-- 1  db2inst1dba209747968Nov 16 10:58 .tstab
lrwxrwxrwx 1  db2inst1dba18      Nov 11 2000 tstmp -> \
                                    .tstmp::cdev:vxfs:
lrwxrwxrwx 1  db2inst1dba18       Nov 11 2000 tstab -> \
                                    .tstab::cdev:vxfs:
```

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, 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 kept on a separate medium. The combination of data redundancy (disk mirroring) and Storage Checkpoints is recommended for highly critical data to protect them from both physical media failure and logical errors.

Backing Up Using a Storage Checkpoint

You can back up a database by creating a Storage Checkpoint using the `db2ed_ckptcreate` command, mount the Storage Checkpoint as read-only using the `db2ed_ckptmount` command, and then back it up using tools such as `tar` or `cpio`.

In a DB2 UDB EEE or ESE SMP environment, you can use `db2ed_ckptcreate_all` to create a Version Checkpoint for a partitioned DB2 database. `db2ed_ckptcreate_all` calls `db2ed_ckptcreate` on every database partition. Similarly, you can use `db2ed_ckptmount_all` to mount all Storage Checkpoints for a Version Checkpoint for a partitioned DB2 database.

Usage Notes

- See the `db2ed_ckptcreate`(1M), `db2ed_ckptmount`(1M), `tar`(1), and `cpio`(1) manual pages for more information.
- For a database with multiple partitions in an SMP environment, see the `db2ed_ckptcreate_all`(1M) and `db2ed_ckptmount_all`(1M) manual pages for more information.

▼ To back up a frozen database image using the command line

Note In this example, all the database containers reside on one VxFS file system named `/db01`.

1. Create a Storage Checkpoint using the `db2ed_ckptcreate` command:

```
$ /opt/VRTS/bin/db2ed_ckptcreate -I db2inst -D PROD -o online
Creating online Storage Checkpoint of database PROD.
Storage Checkpoint Checkpoint_903937870 created.
```

2. Mount the Storage Checkpoint using the `db2ed_ckptmount` command:

```
$ /opt/VRTS/bin/db2ed_ckptmount -I db2inst -D PROD -c \
Checkpoint_903937870 -m /tmp/ckpt_ro
```

Note If the specified mount point directory does not exist, then `db2ed_ckptmount` creates it before mounting the Storage Checkpoint, as long as the DB2 instance owner has permission to create it.

3. Use `tar` to back up the Storage Checkpoint:

```
$ cd /tmp/ckpt_ro
$ ls
db01
$ tar cvf /tmp/PROD_db01_903937870.tar ./db01
```

Recovering a Database Using a Storage Checkpoint

Since Storage Checkpoints record the before images of blocks that have changed, you can use them to do a file-system-based storage rollback to the exact time when the Storage Checkpoint was taken. You can consider Storage Checkpoints as backups that are online, and you can use them to roll back an entire database. Rolling back to or restoring from any Storage Checkpoint is generally very fast because only the changed data blocks need to be restored.

Suppose a user deletes a table by mistake right after 4:00 p.m., and you want to recover the database to a state just before the mistake. You created a Storage Checkpoint (`Checkpoint_903937870`) while the database was running at 11:00 a.m., and you have the `LOGRETAIN` and/or `USEREXIT` database configuration parameters enabled.

▼ To recover the database using a Storage Checkpoint

1. Ensure that the affected database is inactive, and use storage rollback to roll back the database from the Storage Checkpoint you created at 11:00 a.m.

```
$ /opt/VRTS/bin/db2ed_ckptrollback -I db2inst1 -D PROD \
  -c Checkpoint_903937870
```

Note In a DB2 UDB EEE or ESE environment, `db2ed_ckptrollback_all` can be used to roll back a partitioned DB2 database to a Version Checkpoint. `db2ed_ckptrollback_all` calls `db2ed_ckptrollback` on every partition.

2. Start up the instance if it is down.

```
$ db2start
```

3. To re-apply archive logs to the point before the table was deleted to recover the database to 4:00 p.m, enter:

```
$ db2 rollforward database PROD to isotime
```

   where `ISOTIME` is the point in time where all committed transactions are to be rolled forward. The time must be expressed in local time or Coordinated Universal Time (UTC). The UTC format is `yyyy-mm-dd-hh.mm.ss.nnnnnn` (year, month, day, hour, minutes, seconds, microseconds).

4. To complete the roll forward, enter:

```
$ db2 rollforward database PROD complete
```

Cloning the DB2 Database Using db2ed_clonedb

You can use the VERITAS Storage Foundation *for DB2* `db2ed_clonedb` command to clone a DB2 database using mountable and writable Storage Checkpoints to the same or different database so the databases can coexist. You can also create a clone database using a Storage Checkpoint that is not mounted.

You have the option to manually or automatically recover the DB2 database when using the `db2ed_clonedb` command:

- Interactive recovery, which requires using the `-i` option, of the clone instance allows the user to control the degree of recovery by specifying which archive log files are to be replayed.
- Non-interactive recovery recovers the entire database and replays all of the archive logs. You will not be prompted for any archive log names.

Prerequisites

- Before using `db2ed_clonedb` to clone a database either within the same instance or across instances, ensure that your locale is the same as the database locale. If you do not know what locale to set, refer to the file `/opt/VRTSdb2ed/lib/DB2CODPAGE.tbl` for a mapping between the locale and the database Code Page.
- Make sure you have enough space to create a clone instance on your system.

  A clone database takes up as much memory as the primary database.
- You must first create a Storage Checkpoint. (See "Creating Storage Checkpoints Using db2ed_ckptcreate" on page 366.)
- If you choose to use an existing Storage Checkpoint to create the clone database, the Storage Checkpoint needs to be online.

Usage Notes

- The `db2ed_clonedb` command is used to create a copy of an DB2 database, cloning all existing database files to new locations. This is required when using mountable, writable Storage Checkpoints, where a new DB2 database needs to be started on the same host as an existing database.
- The utility requires that the current environment be configured correctly for the existing DB2 database which has had a Storage Checkpoint created underneath it.
- It is assumed that the user has a basic understanding of the DB2 recovery process.
- See the `db2ed_clonedb`(1M) manual page for more information.

Note When cloning a database using `db2ed_clonedb`, any database within the target instance that has the same name as the source database to be cloned will be temporarily uncataloged and therefore unavailable. If the source database is being cloned within the same instance, it will be temporarily uncataloged while `db2ed_clonedb` is running. The database will be recataloged on completion of `db2ed_clonedb`.

Options

`-I`	Specifies the source DB2 instance. If the instance is not specified here, it is set as the current user.
`-S`	Specifies the name of the source DB2 database.
`-T`	Specifies the name of the new DB2 database that will be created.
`-c`	Indicates the name of the Storage Checkpoint to use for creating the new database. The Storage Checkpoint is mounted automatically during the cloning process.
`-m`	Indicates the location of the database containers to be mounted.
`-l`	Specifies the redo log directory of the target database.
`-i`	Runs the command in interactive mode where you must respond to prompts by the system. The default mode is non-interactive. (Optional)
`-a`	This option requires the argument `RECOVER_LOG_LOCATION`. If this option is specified, a minimal database recovery will occur automatically after the clone is created. (Optional)
`-d`	This option is only for use 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.
`-o`	The `-o umount` option shuts down the clone database and unmounts the Storage Checkpoint file system. The `-o restartdb` option mounts the Storage Checkpoint file system and starts the clone database.

▼ To clone a DB2 database with manual DB2 recovery

1. If you will be cloning a database across instances, follow the procedure in "Permission Change Required in the Active Log Directory" on page 145.

2. Use the `db2ed_clonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_clonedb -S source_db -T target_db1 \
-c Checkpoint_1049927758 -m /db2clone/target_db1
Clone database succeeded.
```

▼ To clone a DB2 database with automatic DB2 recovery

1. If you will be cloning a database across instances, follow the procedure in “Permission Change Required in the Active Log Directory” on page 145.

2. Use the `db2ed_clonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_clonedb -S source_db -T target_db2 \
-c Checkpoint_1049927758 -m /db2clone/target_db2 -a \
/db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/
Clone database succeeded.

The database will be rolled forward to 2003-04-09-22.36.02.0000.

/db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/
will be used to be searched for
archived logs during recovery.

                            Rollfo   rward Status

 Inp ut database alias                       = TARGET_DB 2
 Num ber of nodes have returned status       = 1

 Nod e number                                = 0
 Rol lforward status                         = DB   working
 Nex t log file to be read                   = S0000002. LOG
 Log  files processed                        =   -
 L ast committed transaction                =  2003-04-08-20.32.53.000000

DB20000I  The ROLLFORWARD command completed successfully.

                            Rollfo   rward Status

 Inp ut database alias                       = TARGET_DB 2
 Num ber of nodes have returned status       = 1

 Nod e number                                = 0
 Rol lforward status                         = not pendi ng
 Nex t log file to be read                   =
 Log file s processed                      = S00000  02.LOG - S0000002.LOG
 L ast committed transaction                =  2003-04-08-20.32.53.000000

DB20000I  The ROLLFORWARD command completed successfully.
```

▼ To clone a DB2 database with interactive DB2 recovery

1. If you will be cloning a database across instances, follow the procedure in “Permission Change Required in the Active Log Directory” on page 145.

2. Use the db2ed_clonedb command as follows:

```
$ /opt/VRTS/bin/db2ed_clonedb -S source_db -T target_db3 \
-c Checkpoint_1049927758 -m /db2clone/target_db3 -i
/db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/
will be used to be searched for
archived logs during recovery.

Press <Return> to continue
Or <r> to retry
Or <q> to quit:

The database will be rolled forward to 2003-04-09-22.36.02.0000.
The archived logs will be retrieved from
/db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/.
The estimated minimum logs are S0000002.LOG-S0000002.LOG.

You can retrieve all log files by executing
cp /db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/*.LOG
/db2clone/target_db3/REDOLOG
from another session and then

Press q to continue

Or Press Enter to retrieve the minimum logs.

The recovery process will stop
if more logs is required.

Press <Return> to continue ...
    or <q> t  o skip ...

                            Rollfo   rward Status

 Inp ut database alias                 = TARGET_D B3
 Num ber of nodes have returned status  = 1

 Nod e number                          = 0
 Rol lforward status                   = DB   working
 Nex t log file to be read             = S0000002 .LOG
 Log  files processed                  =   -
 L ast committed transaction          =  2003-04-08-20.32.53.000000
```

```
DB20000I  The ROLLFORWARD command completed successfully.

                            Rollfo   rward Status

 Inp ut database alias                     = TARGET_DB 3
 Num ber of nodes have returned status     = 1

 Nod e number                              = 0
 Rol lforward status                       = not pendi ng
 Nex t log file to be read                 =
 Log file s processed                   = S00000  02.LOG - S0000002.LOG
 L ast committed transaction              =  2003-04-08-20.32.53.000000

DB20000I  The ROLLFORWARD command completed successfully.
```

▼ **To clone a DB2 Database to a different instance with manual recovery**

Use the db2edclonedb command as follows:

```
# /opt/VRTS/bin/db2ed_clonedb -I inst1 -S PROD -T clonedb \
  -c Checkpoint -m /clone
```

where `inst1` is the source instance.

Permission Change Required in the Active Log Directory

If you want to duplicate a database across instances, manually change the permissions on the `ActiveLogDir` files for the source instance. This change is required so the new database instance user can access the log files to duplicate the database to the new instance.

Example

The following example shows how to allow `db2inst2` to access the logs from the source instance (`db2inst1`) so the database can be duplicated. This example assumes:

- You are duplicating from `db2inst1` to `db2inst2`.
- Users `db2inst1` and `db2inst2` are in the same UNIX group.

▼ **To allow db2inst2 access the logs from the source instance (db2inst1) for duplication of the database**

1. Log on to the instance owner of the source instance:

```
# su - db2inst1

Sun Microsystems Inc.   SunOS 5.8      Generic Feb ruary 2000
$ pwd
/db2/home/db2inst1
```

2. Connect to the database requiring duplication:

```
$ db2 connect to sample

Database Connection Information
Database server    = DB    2/SUN 8.1.0
SQL authorization ID   = DB2INST1
Local database alias   = SAMPLE
```

3. Use the following command to obtain the log directory:

```
$ db2 get db cfg for sample | grep -i log

Log retain for recovery status               = RECOVERY
User exit for logging status                 = NO
Catalog cache size (4KB)     (CATALOGCA CHE_SZ) = (MAXAPPLS*4)
Log buffer size (4KB)        (LOGBUFSZ)       = 8
Log file size (4KB)          (LOGFILSIZ)      = 1000
Number of primary log files (LOGPRIMARY)      = 3
Number of secondary log files (LOGSECOND)     = 2
Changed path to log files   (NEWLOGPATH)      =
Path to log files = /db2/home/db2inst1/db2inst1/NODE0000
/SQL00002/SQLOGDIR/
Overflow log path            (OVERFLOWLOGPATH) =
Mirror log path              (MIRRORLOGPATH)   =
First active log file = S0000122.LOG
Block log on disk full (BLK_LOG_DSK_FUL) = NO
Percent of max active log space by transaction (MAX_LOG)= 0
Num. of active log files for 1 active UOW(NUM_LOG_SPAN) = 0
Percent log file reclaimed before soft chckpt (SOFTMAX) = 100
Log retain for recovery enabled (LOGRETAIN)       = RECOVERY
User exit for logging enabled (USEREXIT)           = OFF
```

4. Note down the value for the variable *`NEWLOGPATH`*

5. Move to the *`NEWLOGPATH`* location.

```
$ cd /db2/home/db2inst1/db2inst1/NODE0000/SQL00002/SQLOGDIR/
```

6. Use the `chmod` command to provide read access to the LOG Files of the target instance account:

```
$ chmod g+r *.LOG
```

Guidelines for DB2 Recovery

For optimal DB2 recovery, follow these guidelines:

- Ensure that the `LOGRETAIN` and/or `USEREXIT` database configuration parameters are enabled. Enabling these parameters makes the following roll-forward recovery techniques available for use:

  - Point in time recovery using the `ROLLFORWARD DATABASE` command
  - Online processing of `BACKUP` and `RESTORE DATABASE` commands

  When `LOGRETAIN` is enabled, log files are not overwritten when they become inactive. When `USEREXIT` is enabled, DB2 will call the user exit program when a log file is ready for archiving.

- You must have `SYSADM`, `SYSCTRL`, or `SYSMAINT` authority to use the `ROLLFORWARD DATABASE` command.
- A database must be restored successfully (using the `RESTORE DATABASE` command) before it can be rolled forward.
- After Storage Rollback, perform DB2 recovery, applying some or all of the archived redo logs.
- To perform a complete recovery, use:

  ```
  db2rollforward database database_name to end of logs
  ```

- To perform a point-in-time recovery, use:

  ```
  db2rollforward database database_name to PIT
  ```

  Where `PIT` is the point in time you are rolling forward to. The point in time must be specified in UTC or local time.

- To complete the recovery, use:

  ```
  db2rollforward database database_name complete
  ```

See your DB2 documentation for complete information on recovery.

Using the GUI to Perform Storage Checkpoint-Related Operations

Use the GUI to create Storage Checkpoints and then roll back an entire database using any of the previously created Storage Checkpoints. See "Using the VERITAS Storage Foundation for DB2 Graphical User Interface" on page 221 for detailed information about GUI operations.

Using Database FlashSnap for Backup and Off-Host Processing 8

This chapter describes how to use VERITAS Database FlashSnap to create a point-in-time copy of a database for backup and off-host processing. Database FlashSnap allows you to make backup copies of your volumes online with minimal interruption to users.

VERITAS Database FlashSnap is included with VERITAS Storage Foundation Enterprise Edition.

Database FlashSnap lets you capture an online image of an actively changing database at a given instant, known as a snapshot. You can perform backups and off-host processing tasks on snapshots while providing continuous availability of your critical data.

Database FlashSnap commands can be executed from either the command line or the GUI.

Topics covered in this chapter include:

How VERITAS Database FlashSnap Works

You can use VERITAS Database FlashSnap to implement point-in-time copy solutions for enterprise databases. Database FlashSnap offers you a flexible way to efficiently manage multiple point-in-time copies of your data, and reduce resource contention on your business-critical servers.

Database FlashSnap allows database administrators to create a consistent copy of a database without root privileges by creating 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 also allows you to resynchronize your original database volumes from the data in the snapshot if the original volumes become corrupted. This is referred to as *reverse resynchronization*.

Database FlashSnap can significantly reduce the time it takes to backup your database, increase the availability of your production database, and still maintain your production database's performance.

Note To use Database FlashSnap, you must have VERITAS Storage Foundation Enterprise Edition on all systems on which you intend to use Database FlashSnap.

To use Database FlashSnap, you must first configure the volumes used by the database. See "Preparing Hosts and Storage for Database FlashSnap" on page 159, for more information.

Typical Problems Database FlashSnap Solves

Database FlashSnap is designed to enable you to use database snapshots to overcome the following types of problems encountered in enterprise database environments:

- 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, privileges that database administrators do not usually have.
- In some companies, database administrators are granted root privileges, but managing storage is typically not central to their job function or their core competency.

- Creating database snapshots is a complex process, especially in large configurations where thousands of volumes are used for the database. One mistake can render the snapshots useless.

Because it does not require root privileges, Database FlashSnap overcomes these obstacles by enabling database administrators to create consistent snapshots of the database more easily. The snapshots can be utilized for repetitive use.

Database FlashSnap Applications

The following are typical applications of VERITAS Database FlashSnap:

- *Database Backup and Restore*: Enterprises require 24/7 online data availability. They cannot afford the downtime involved in backing up critical data offline. By creating a clone database or a duplicate volume snapshot of data, and then using it to back up your data, your business-critical applications can continue to run without extended down time or impacted performance. After a clone database or snapshot volume is created, it can be used as a source to back up the original database.
- *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 clone database that you have created from snapshots using VERITAS Database FlashSnap, rather than allowing them to compete for access to the primary volume or database. When required, you can quickly resynchronize the clone database with the primary database to get up-to-date information.
- *Application Development and Testing*: Development or service groups can use a clone database created with Database FlashSnap as a test database for new applications. A clone database provides developers, system testers, and quality assurance groups with a realistic basis for testing the robustness, integrity, and performance of new applications.
- *Logical Error Recovery*: Logical errors caused by an administrator or an application program can compromise the integrity of a database. You can recover a database by restoring the database files from a volume snapshot from a clone database created from volume snapshots using Database FlashSnap. These solutions are faster than restoring database files from tape or other backup media.

Using Database FlashSnap

The system administrator needs to configure storage according to the requirements specified in the snapplan. For information on configuring storage, see "Preparing Hosts and Storage for Database FlashSnap" on page 159.

Database FlashSnap allows you to check the storage setup against requirements set forth in the snapplan. Depending on the results, the database administrator may need to modify the snapplan or the system administrator may need to adjust the storage configuration. Properly configuring storage is the only aspect of using Database FlashSnap that requires the system administrator's participation.

To use Database FlashSnap, a database administrator must first define their snapshot requirements. For example, they need to determine whether off-host processing is required and, if it is, which host should be used for it. In addition, it is also important to consider how much database downtime can be tolerated. Database snapshot requirements are defined in a file called a *snapplan*. The snapplan specifies snapshot options that will be used when creating a snapshot image (such as whether the snapshot mode will be `online_snapshot`, `online_mirror`, or `offline`). For further details, see "Creating a Snapplan (db2ed_vmchecksnap)" on page 177.

After creating the snapplan, the database administrator must validate it to ensure that it is correct. During validation the snapplan is copied to the repository before using it to create a snapshot. Depending on the validation results, the database administrator may need to modify the snapplan or the system administrator may need to adjust the storage configuration. Properly configuring storage is the only aspect of using Database FlashSnap that requires the system administrator's participation.

After storage is configured as specified in the snapplan and the snapplan has been validated, the database administrator can create snapshots of the database and create database clones based on the snapshots on either the same host or a secondary one.

A database clone can be used on a secondary host for off-host processing, including decision-support analysis and reporting, application development and testing, database backup, and logical error recovery. After a user has finished using the clone on a secondary host, the database administrator can shut down the clone and move the snapshot database back to the primary host. Regardless of whether a snapshot is used on the primary or secondary host, it can be resynchronized with the primary database using Database FlashSnap. Database FlashSnap utilizes VERITAS Volume Manager FastResync to quickly resynchronize the changed section between the primary and snapshot. Refer to the *VERITAS Volume Manager 4.1 User's Guide* for details about the Volume Manager FastResync.

Database FlashSnap can also be used to recover the primary copy of the database if it becomes corrupted by overwriting it with the snapshot. You can recover the primary database with a snapshot using the reverse resynchronization functionality of Database FlashSnap.

Database FlashSnap Commands

The Database FlashSnap feature consists of three commands:

- `db2ed_vmchecksnap` (used on the primary host)

  Creates and validates the snapshot plan used to create a snapshot image of a DB2 database. You can also use `db2ed_vmchecksnap` to copy, list, or remove a snapplan or make sure the storage is configured properly for the task. `db2ed_vmchecksnap` is also used on the secondary host to list the snapplan.

- `db2ed_vmsnap` (used on the primary host)

  Creates a snapshot image of a DB2 database by splitting the mirror volumes used by the database. You can also use `db2ed_vmsnap` to resynchronize snapshot volumes with their original volumes. The command also allows you to resynchronize the original volumes from the data in the snapshot volumes, which is useful if the original volumes become corrupted. Resynchronizing the original volumes from the snapshot volumes is known as *reverse resynchronization*.

- `db2ed_vmclonedb` (used on the primary or secondary host)

  Mounts and starts a clone database using snapshot volumes. It can also shut down a clone snapshot database and deport its volumes, as well as restart a clone snapshot database that has been shut down. The snapshot image can be brought up on the same host running the primary database or on a secondary host.

All of these commands can be executed by the DB2 database administrator and do not require superuser (`root`) privileges.

Note Database FlashSnap operations can be executed from either the command line or the GUI.

Database FlashSnap Options

Database FlashSnap offers three options for creating database snapshots. The option you choose is specified in the snapplan.

- *online_snapshot*

  Use this option to:

  - create a clone database for decision-support, reporting, and testing purposes.
  - take over the primary database if it becomes corrupted
  - back up the primary database at the operating system level

With this option, the `db2ed_vmsnap` command will first put the database into `WRITE SUSPEND` mode. After `db2ed_vmsnap` finishes creating the snapshot, it will take the database out of `WRITE SUSPEND` mode. The online redo log must be included in the snapshot.

- *online_mirror*

  Use this option to create a "hot backup." The online_mirror option creates a mirror of the database without creating a clone database. Creating a clone database is not allowed with the `online_mirror` option. The mirror database can be used to restore the primary database if media failure occurs.

  With this option, the `db2ed_vmsnap` command will first put the database into `WRITE SUSPEND` mode. After `db2ed_vmsnap` finishes creating the snapshot, it will take the database out of `WRITE SUSPEND` mode. The active log must not be included in the snapshot.

- *offline*

  The `offline` option can be used to clone or back up a database. With this option, the database must be inactive when the snapshot is created. Because the database is inactive, it is not required to be placed into `WRITE SUSPEND` mode. This type of snapshot is a valid database backup. You must include all data and active transaction log space volumes used by its database in the snapshot.

Note In this release of VERITAS Storage Foundation for DB2, Database FlashSnap supports third mirror break-off snapshots only. Third mirror break-off snapshots are fully synchronized, full-sized snapshots. See the *VERITAS Volume Manager Administrator's Guide* for more information.

Planning Considerations

Before using Database FlashSnap, you must first determine your intended application. You will then need to make the following decisions:

- Which snapshot mode is appropriate: `online_snapshot`, `online_mirror`, or `offline`?
- Will you need one or two hosts?

Selecting the Snapshot Mode

Snapshot Mode	**Purpose**
`online_snapshot`	Choose `online_snapshot` if you want to: ◆ Clone a database. A clone database can be used for decision-support analysis, reporting, development, or testing. ◆ Recover the primary database if it is corrupted by either taking over the primary role or performing a reverse resync. With `online_snapshot` you cannot roll forward any logs because the primary and secondary log sequences are different. ◆ Back up the primary database on the same or a different host.
`online_mirror`	Choose `online_mirror` if you want to: ◆ Create a “hot backup” image. ◆ Back up the primary database on the same host at the operating system level using different set of storage. ◆ Perform a point-in-time recovery of the primary database if it becomes corrupted. With `online_mirror`, it is possible to roll forward to recover changes to your database.
`offline`	Choose `offline` if you want to: ◆ Clone your production database if it is offline. A clone database can be used for decision-support analysis, reporting, development, or testing. ◆ Back up your production database if it is offline.

Selecting One or Two Hosts

If maintaining the performance of your primary database is critical, you can offload processing of the snapshots to a secondary host. For off-host processing, storage must be shared between the primary and secondary hosts.

If cost savings is most important, you can choose to do the processing on the same host as the primary database to save on hardware costs.

Preparing Hosts and Storage for Database FlashSnap

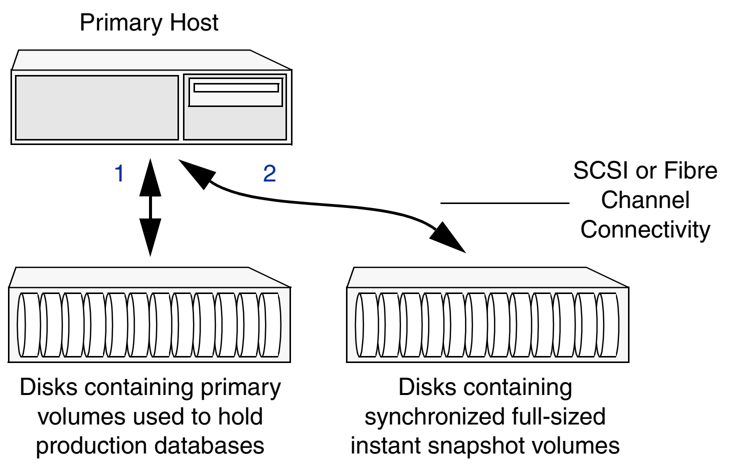
Setting Up Hosts

Database FlashSnap requires sufficient VERITAS Volume Manager disk space, and can be used on the same host that the database resides on (the primary host) or on a secondary host. 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.

Single-Host Configuration

The following figure, "Example of a Database FlashSnap Solution on a Primary Host" on page 159 shows the suggested arrangement for implementing Database FlashSnap solutions on the primary host to avoid disk contention.

Example of a Database FlashSnap Solution on a Primary Host

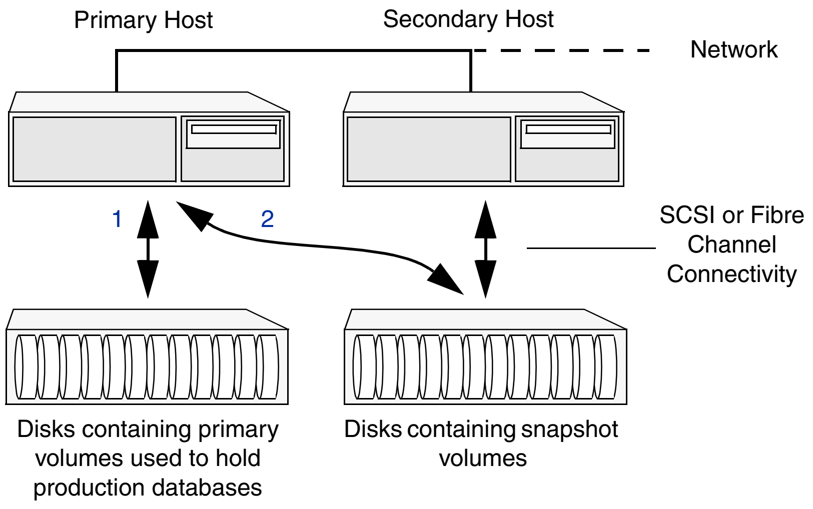

Primary Host
1
2
SCSI or Fibre Channel Connectivity
Disks containing primary volumes used to hold production databases
Disks containing synchronized full-sized instant snapshot volumes

Two-Host Configuration

As shown in the figure below, "Example of an Off-Host Database FlashSnap Solution" on page 160, a Database FlashSnap configuration with two hosts 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. A two-host configuration also allows the snapshot database to avoid contending for I/O resources on the primary host.

For off-host processing applications, both the primary and secondary hosts need to share the storage in which the snapshot database is created. Both the primary and secondary hosts must be able to access the disks containing the snapshot volumes.

Example of an Off-Host Database FlashSnap Solution


Primary Host
Secondary Host
Network
1
2
SCSI or Fibre Channel Connectivity
Disks containing primary volumes used to hold production databases
Disks containing snapshot volumes

Host and Storage Requirements

Before using Database FlashSnap, ensure that:

- All files are on VxFS file systems over VxVM volumes. Raw devices are not supported.
- Symbolic links to containers are not supported.
- DB2 containers and active logs are in the same disk group.

In addition, before attempting to use Database FlashSnap with two hosts, ensure that:

- The versions of VERITAS Storage Foundation *for DB2* on the primary and secondary hosts are the same.
- The same version of DB2 is installed on both hosts the DB2 binaries and datafiles are on different volumes and disks.
- The UNIX login for the database user and group must be the same on both hosts.
- You have a VERITAS Storage Foundation 4.1 *for DB2* Enterprise Edition license on both hosts.

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 primary database. If the primary containers 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 containers before you can create a snapshot of the database. This section describes the procedure used to create snapshot mirrors of volumes.

You can use the `vxsnap` CLI command or the GUI to create a snapshot mirror. Creating a snapshot mirror using the GUI is relatively easy. Because the time required to synchronize a snapshot mirror can be long, using the command line is recommended when resynchronizing snapshot mirrors.

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 *VERITAS Volume Manager 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.

Usage Notes

- Create a separate disk group for DB2 database-related files.
- Do not share volumes between DB2 database files and other software.
- `INSTANCE_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 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.

Caution The procedure given in this section is for existing volumes without existing snapshot plexes or associated snapshot volumes.

▼ To create a snapshot mirror of a volume or volume set

Note In the following procedure, *volume_name* is the name of either a volume or a volume set.

1. To prepare the volume for being snapshot, use the `vxsnap prepare` command:

```
# vxsnap -g diskgroup prepare volume \
alloc="storage_attribute ..."
```

Note 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.

Note For enabling persistent FastResync on a volume in VxVM 4.1, 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
```

Note There is no option for creating multiple mirrors at the same time. Only one mirror can be created at a time.

5. List the available mirrors:

```
# vxprint -g diskgroup -F%name -e"pl_v_name in \"volume_name\""
```

Note The following two steps enable database FlashSnap to locate the correct mirror plexes when creating snapshots.

6. Set the `tag_name` tag for the data plex you want to use for breaking off the mirror.

   ```
   # vxedit -g diskgroup set putil2=tag_name plex_name
   ```

7. Verify that the `tag_name` tag has been set to the desired data plex:

   ```
   # vxprint -g diskgroup -F%name -e"pl_v_name in \"volume_name\" \
   && p2 in \"tag_name\""
   ```

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.

Example

The following example shows the steps involved in creating a snapshot mirror for the volume `data_vol` belonging to the disk group `PRODdg`.

Prepare the volume `data_vol` for mirroring:

```
# vxsnap -g PRODdg prepare data_vol alloc=PRODdg01
```

Verify that FastResync is enabled:

```
# vxprint -g PRODdg -F%fastresync data_vol
on
```

Verify that a DCO and a DCO log are attached to the volume:

```
# vxprint -g PRODdg -F%hasdcolog data_vol
on
```

Create a snapshot mirror of `data_vol`:

```
# vxsnap -g PRODdg addmir data_vol alloc=PRODdg02
```

List the data plexes:

```
# vxprint -g PRODdg -F%name -e"pl_v_name in \"data_vol\""
data_vol-01
data_vol-02
```

Note 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.

Decide which data plex you want to use and set the `tag_name` tag for it:

```
# vxedit -g PRODdg set putil2=PRODtag data_vol-02
```

Verify that the `PRODtag` 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 \"PRODtag\""
data_vol-02
```

To verify that the snapshot volume was created successfully, use the `vxprint -g <dg>` command as follows:

```
# vxprint -g PRODdg

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 with Database FlashSnap for DB2

The procedure described in this section describes how to upgrade a volume created using a version older than VxVM 4.1 so that it can take advantage of database FlashSnap.

Note Snapshots that were created using a version older than VxVM 4.0 are not supported by database FlashSnap.

Note 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.

Note Existing snapshot volumes created by the `vxassist` command are not supported. A combination of snapshot volumes created by `vxassist` and `vxsnap` are not supported.

▼ **To upgrade an existing volume created with an earlier version of VxVM:**

1. Upgrade the disk group that contains the volume, to a 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.

Note 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

In this example, the volume, `data_vol`, is upgraded to make use of VxVM 4.1 features.

Upgrade the disk group, `PRODdg`.

```
# vxdg upgrade PRODdg
```

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
```

Reattach any snapshot volume back to the primary volume to be upgraded.

```
# vxsnap -g PRODdg snapback SNAP-data_vol
```

Turn off FastResync on the volume to be upgraded.

```
# vxvol -g PRODdg set fastresync=off data_vol
```

Disassociate and remove any older DCO object and DCO volumes.

```
# vxassist -g PRODdg remove log  data_vol logtype=dco
```

Upgrade the volume by associating a new DCO object and DCO volume.

```
# vxsnap -g PRODdg prepare data_vol alloc="PRODdg01 PRODdg02"
```

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

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

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

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

# vxprint -g PRODdg

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     -        -       -
```

Summary of Database Snapshot Steps

You can use Database FlashSnap commands to create a snapshot of your entire database on the same host or on a different one. Three types of snapshots can be created: `online_snapshot`, `online_mirror`, or `offline`.

If the `SNAPSHOT_MODE` specified in the snapplan is set to `online_snapshot`, `db2ed_vmsnap` first puts the database into `WRITE_SUSPEND` mode. After the database is in `WRITE_SUSPEND` mode, the snapshot volumes are created by breaking off the mirrors and splitting the snapshot volumes into a separate disk group. The split disk group will have the prefix specified in the snapplan. The database is taken out of `WRITE_SUSPEND` mode using the `WRITE_RESUME` command.

You can run the `db2ed_vmclonedb` command on the secondary host to import the disk group and create a clone database. You can run the `db2ed_vmsnap` command to synchronize the snapshot volumes with the primary database. The snapshot volumes can also be used to restore the primary database if it becomes corrupted. In this case, the `ALLOW_REVERSE_RESYNC` parameter must be set to `yes` in the snapplan.

If the `SNAPSHOT_MODE` is set to `online_mirror`, `db2ed_vmsnap` puts the database into `WRITE_SUSPEND` mode. The snapshot volumes are created by breaking off the mirrors and splitting the snapshot volumes into a separate disk group. The split disk group has the prefix specified in the snapplan. The database is taken out of `WRITE_SUSPEND` mode using the `WRITE_RESUME` command. The database snapshot can be used to restore the primary database if it becomes corrupted.

If the `SNAPSHOT_MODE` is set to `offline`, the database must be inactive before the snapshot is created. The secondary host must be different than the primary host. Since the database is inactive, no active logs exist.

`online_snapshot`, `online_mirror`, and `offline` snapshots provide a valid backup image of the database. You can use the snapshot as a source for backing up the database or creating a clone database (only for `online_snapshot` or `offline`) for decision-support purposes.

The sections that follow explain 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. For more information about the `vxsnap` command, see *VERITAS Volume Manager Administrator's Guide*.

Note Make sure the volumes used by the database are configured properly before attempting to take a snapshot. This requires superuser (`root`) privileges.

Note Anytime you change the structure of the database (for example, by adding or deleting containers), you must run `db2ed_update`.

Note Database FlashSnap commands must be run by the DB2 instance owner.

▼ To create a snapshot image of a database

1. Perform the steps in "Creating a Snapshot Mirror of a Volume or Volume Set Used by the Database" on page 161.

2. Use the `db2ed_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 `db2ed_vmchecksnap` command can also be used to:

   - List all snapplans associated with a specific *`DB2DATABASE`* (`db2ed_vmchecksnap -o list`).
   - Remove the snapplan from the VxDBA repository (`db2ed_vmchecksnap -o remove -f` *`SNAPPLAN`*).
   - Copy a snapplan from the VxDBA repository to your local directory (`db2ed_vmchecksnap -o copy -f` *`SNAPPLAN`*).

   For information about the snapplan file, see "Creating a Snapplan (db2ed_vmchecksnap)" on page 177.

3. Use the `db2ed_vmsnap` command to create snapshot volumes for the database. See "Creating a Snapshot (db2ed_vmsnap)" on page 190 for more information.

4. On the secondary host, use the `db2ed_vmclonedb` command to create a clone database using the disk group deported from the primary host. See "Cloning a Database (db2ed_vmclonedb)" on page 199 for more information.

   If the primary and secondary hosts specified in the snapplan are different, the `db2ed_vmclonedb` command imports the disk group that was deported from the primary host, recovers the snapshot volumes, mounts the file systems, recovers the database, and brings the database online with a different DB2DATABASE name than the primary host. If the secondary host is different, the database name can be same. You can use the `-o recoverdb` option to let `db2ed_vmclonedb` perform an automatic database recovery, or you can use the `-o mount` 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_snapshot`.

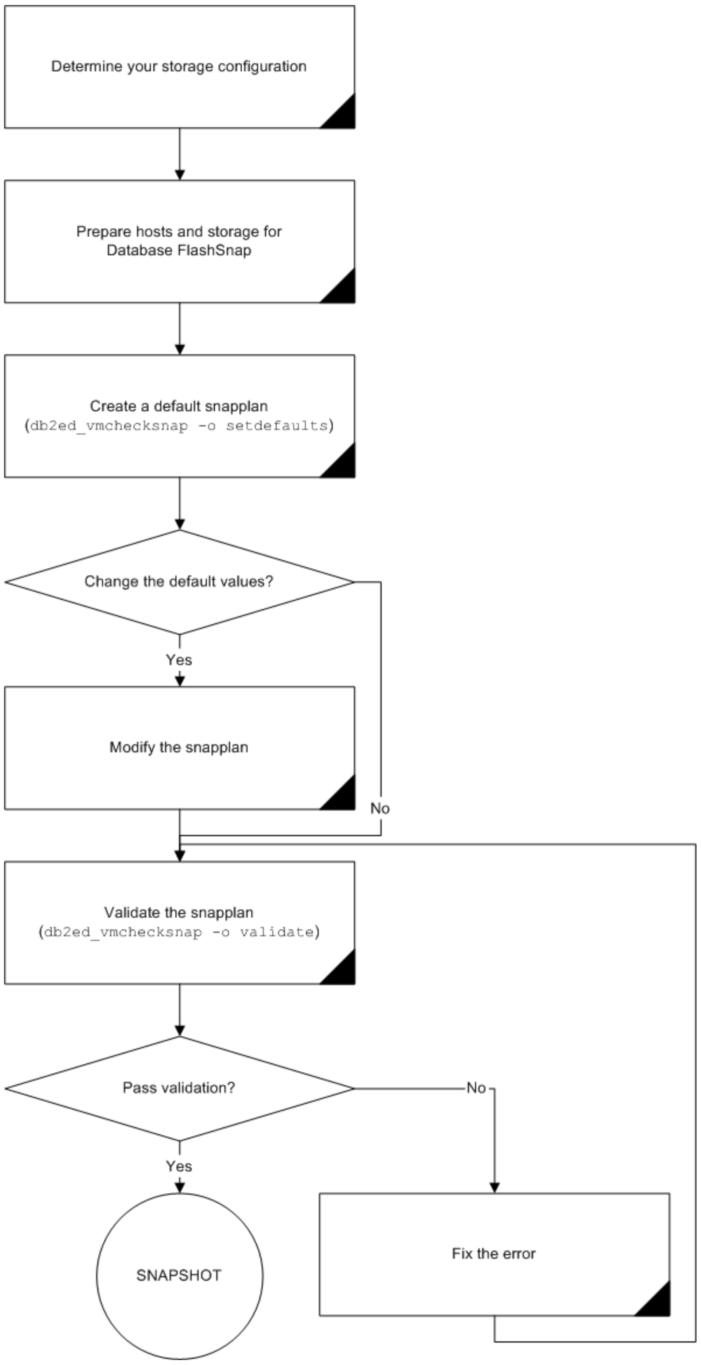
   You can also create a clone on the primary host. Your snapplan settings specify whether a clone should be created on the primary or secondary host.

5. You can now use the clone database to perform database backup and other off-host processing work.

6. 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.

The following flow chart depicts the sequence of steps leading up to taking a snapshot using Database FlashSnap.

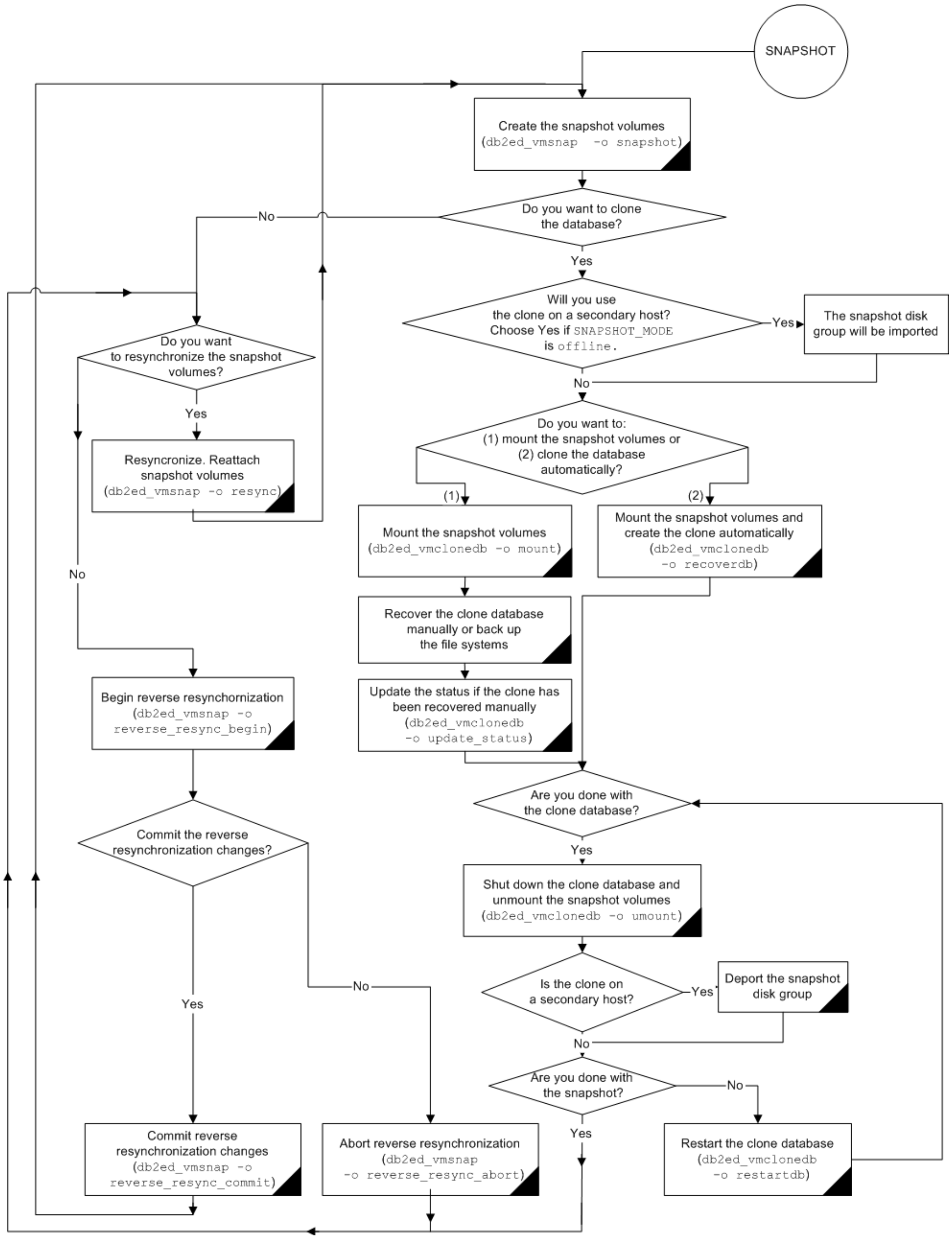
Prerequisites for Creating a Snapshot of your Database


Determine your storage configuration
Prepare hosts and storage for Database FlashSnap
Create a default snapplan (db2ed_vmchecksnap -o setdefaults)
Change the default values?
Yes
No
Modify the snapplan
Validate the snapplan (db2ed_vmchecksnap -o validate)
Pass validation?
Yes
No
SNAPSHOT
Fix the error

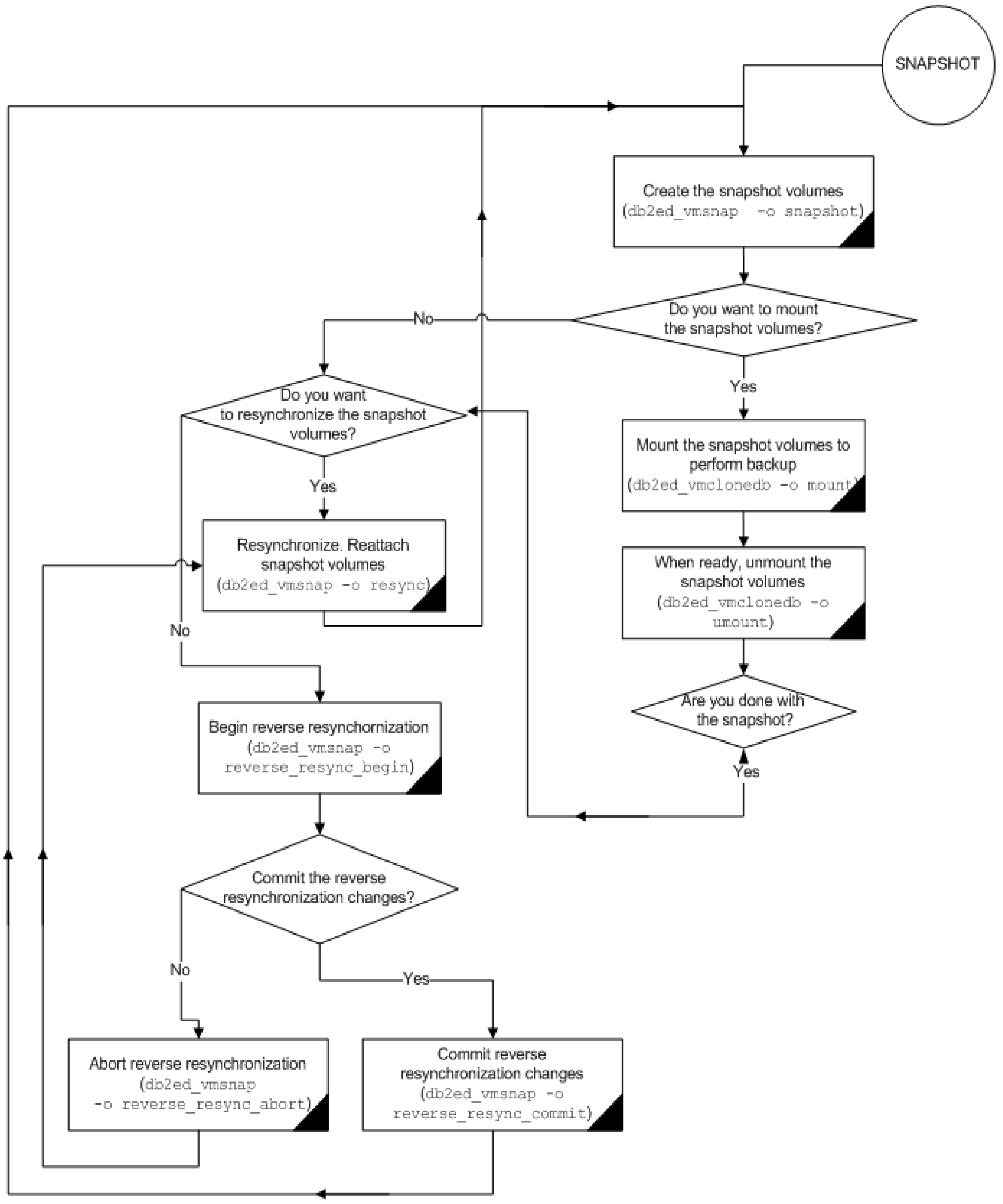
There are many actions you can take after creating a snapshot of your database using Database FlashSnap. You can create a clone of the database for backup and off-host processing purposes. You can resynchronize the snapshot volumes with the primary database. In the event of primary database failure, 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.

Actions you can perform if the SNAPSHOT_MODE Is set to online_snapshot or offline


SNAPSHOT
Create the snapshot volumes (db2ed_vmsnap -o snapshot)
Do you want to clone the database?
No
Yes
Will you use the clone on a secondary host? Choose Yes if SNAPSHOT_MODE is offline.
Yes
The snapshot disk group will be imported
No
Do you want to: (1) mount the snapshot volumes or (2) clone the database automatically?
(1)
(2)
Mount the snapshot volumes (db2ed_vmclonedb -o mount)
Mount the snapshot volumes and create the clone automatically (db2ed_vmclonedb -o recoverdb)
Recover the clone database manually or back up the file systems
Update the status if the clone has been recovered manually (db2ed_vmclonedb -o update_status)
Are you done with the clone database?
Yes
Shut down the clone database and unmount the snapshot volumes (db2ed_vmclonedb -o umount)
Is the clone on a secondary host?
Yes
Deport the snapshot disk group
No
Are you done with the snapshot?
No
Restart the clone database (db2ed_vmclonedb -o restartdb)
Yes
Do you want to resynchronize the snapshot volumes?
Yes
Resyncronize. Reattach snapshot volumes (db2ed_vmsnap -o resync)
No
Begin reverse resynchornization (db2ed_vmsnap -o reverse_resync_begin)
Commit the reverse resynchronization changes?
Yes
No
Commit reverse resynchronization changes (db2ed_vmsnap -o reverse_resync_commit)
Abort reverse resynchronization (db2ed_vmsnap -o reverse_resync_abort)

Actions you can perform if the SNAPSHOT_MODE Is set to online_mirror


SNAPSHOT
Create the snapshot volumes (db2ed_vmsnap -o snapshot)
Do you want to mount the snapshot volumes?
No
Yes
Mount the snapshot volumes to perform backup (db2ed_vmclonedb -o mount)
When ready, unmount the snapshot volumes (db2ed_vmclonedb -o umount)
Are you done with the snapshot?
Yes
Do you want to resynchronize the snapshot volumes?
Yes
Resynchronize. Reattach snapshot volumes (db2ed_vmsnap -o resync)
No
Begin reverse resynchornization (db2ed_vmsnap -o reverse_resync_begin)
Commit the reverse resynchronization changes?
No
Yes
Abort reverse resynchronization (db2ed_vmsnap -o reverse_resync_abort)
Commit reverse resynchronization changes (db2ed_vmsnap -o reverse_resync_commit)

Creating a Snapplan (db2ed_vmchecksnap)

The `db2ed_vmchecksnap` command creates a snapplan that `db2ed_vmsnap` uses to create a snapshot of a DB2 database. The snapplan specifies snapshot scenarios (such as `online_snapshot`, `online_mirror`, or `offline`).

You can name a snapplan file whatever you choose. Each entry in the snapplan file is a line in *`parameter=argument`* format.

When using `db2ed_vmchecksnap` to create or validate a snapplan, the following parameters are set:

Parameter	**Value**
SNAPSHOT_VERSION	Specifies the snapshot version for this major release of VERITAS Storage Foundation *for DB2*.
`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`	The name of the VxVM disk group used by the primary database.
`SNAPSHOT_DG`	The name of the disk group containing the snapshot volumes. 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.
`DB2DATABASE`	The name of the DB2 database.
`DB2HOME`	The home directory of the database.
`REDOLOG_DEST`	The full path of the redo logs.

Parameter	**Value**
SNAPSHOT_MODE	**offline** or **online_snapshot** or **online_mirror** Specifies whether the database should be offline, online_snapshot, or online_mirror when the snapshot is created. By default, the SNAPSHOT_MODE is online_snapshot. If the snapshot is created while the SNAPSHOT_MODE for the database is set to online_snapshot or online_mirror, the db2ed_vmsnap command will first put the database into WRITE SUSPEND mode. After db2ed_vmsnap finishes creating the snapshot, it will take the database out of WRITE SUSPEND mode. If the database is offline, it is not necessary to put the database into WRITE SUSPEND mode. If the SNAPSHOT_MODE is offline, the secondary host must be different than the primary host. If the SNAPSHOT_MODE is online_mirror, the primary and secondary host must be the same, and the ALLOW_REVERSE_RESYNC parameter must be set to yes in the snapplan. If the SNAPSHOT_MODE is online_snapshot, the clone can be created on either the primary or a secondary host.
SNAPSHOT_PLAN_FOR	The default value is **database** and cannot be changed.
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.
SNAPSHOT_MIRROR	Specifies the number of mirrors that will be part of the snapshot. The default value is 1.
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.

Parameter	Value
`ALLOW_REVERSE_RESYNC`	**`yes`** or **`no`** By default, reverse resynchronization is off (set equal to `no`) with the `SNAPSHOT_MODE` set to `online_snapshot`. If the `SNAPSHOT_MODE` is `online_mirror`, reverse resynchronization is set equal to `yes`. If `ALLOW_REVERSE_RESYNC` is set to `yes`, data from the snapshot volume can be used to overwrite the primary volume.

When you first run `db2ed_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.

Note You cannot access Database FlashSnap commands (`db2ed_vmchecksnap`, `db2ed_vmsnap`, and `db2ed_vmclonedb`) with the VxDBA menu utility.

Prerequisites

- Storage must be configured as specified in "Preparing Hosts and Storage for Database FlashSnap" on page 159.
- You must be the DB2 instance owner.
- 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 with LOGRETAIN mode on.

Usage Notes

- The snapplan must be created on the primary host.
- After creating the snapplan using the `db2ed_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 `db2ed_vmchecksnap`(1M) online manual page for more information.
- If the `SNAPSHOT_MODE` for the database is set to `online_snapshot`, the snapshot can be created on either the primary or a secondary host. If the `SNAPSHOT_MODE` is set to `online_mirror`, the snapshot must be created on the primary host. If the `SNAPSHOT_MODE` is set to `offline`, the snapshot must be created on a secondary host.
- Database FlashSnap commands are not supported for multi-partitioned DB2 databases.

▼ To create a snapplan

1. Change directories to the working directory you want to store your snapplan in.

```
$ cd /working_directory
```

2. Create a snapplan with default values using the `db2ed_vmchecksnap` command:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D DB2DATABASE \
-f SNAPPLAN -o setdefaults -t host_name -p PLEX_TAG
```

3. Open the snapplan file in a text editor and modify it as needed.

Note The default setting for `SNAPSHOT_MODE` is `online_snapshot`.

For information on creating a snapplan using the GUI, see "Using the VERITAS Storage Foundation for DB2 Graphical User Interface" on page 221.

Example

In this example, a snapplan, snap1, is created for a snapshot image in a single-host configuration and default values are set. The host is named host1 and the working directory is /export/snap_dir.

```
$ cd /export/snap_dir
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD \
-f snap1 -o setdefaults -t host1 -p PRODtag
Snapplan snap1 for PROD.
=====================================================
SNAPSHOT_VERSION=4.0
PRIMARY_HOST=host1
SECONDARY_HOST=host1
PRIMARY_DG=PRODdg
SNAPSHOT_DG=SNAP_PRODdg
DB2DATABASE=PROD
DB2HOME=/PROD_HOME
REDOLOG_DEST=/PROD_HOME/inst1/NODE0000/SQL00001/SQLOGDIR/
SNAPSHOT_MODE=online_snapshot
SNAPSHOT_PLAN_FOR=database
SNAPSHOT_PLEX_TAG=PRODtag
SNAPSHOT_MIRROR=1
SNAPSHOT_VOL_PREFIX=SNAP_
ALLOW_REVERSE_RESYNC=no
```

Example

In the following example, a snapplan, snap2, is created for a snapshot image in a two-host configuration, and default values are set. The primary host is host1, the secondary host is host2, and the working directory is /export/snap_dir.

```
$ cd /export/snap_dir
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD \
 -f snap2 -o setdefaults PRODtag -t host2
Snapplan snap2 for PROD.
=====================================================
SNAPSHOT_VERSION=4.0
PRIMARY_HOST=host1
SECONDARY_HOST=host2
PRIMARY_DG=PRODdg
SNAPSHOT_DG=SNAP_PRODdg
DB2DATABASE=PROD
DB2HOME=/PROD_HOME
```

```
REDOLOG_DEST=/PROD_HOME/inst1/NODE0000/SQL00001/SQLOGDIR/
SNAPSHOT_MODE=online_snapshot
SNAPSHOT_PLAN_FOR=database
SNAPSHOT_PLEX_TAG=PRODtag
SNAPSHOT_MIRROR=1
SNAPSHOT_VOL_PREFIX=SNAP_
ALLOW_REVERSE_RESYNC=no
```

Creating Multi-mirror Snapshots

To make 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 gets disabled. Snapshot volumes can be mounted and the entire database snapshot is usable even if one of the mirror gets disabled. The multi-mirror snapshots are enabled via `SNAPSHOT_MIRROR=<n>` in the snapplan.

Note There are no changes to the Command Line usage or arguments for the Flashsnap tools.

Note Before taking the snapshot, make sure all tagged snapshot mirrors are in SNAPDONE state.

The following sample explains the setup and the procedure for taking multi-mirror snapshots:

1. Add the second mirror and DCO log. When allocating storage for the second mirror and DCO logs, make sure the snap volumes are splittable. If snap volumes are not splittable, `dbed_vmchecksnap` fails with appropriate errors.

   Tag the newly added mirror with the same tag as that of the first mirror.

   Assume that the volume has `fastresync = on`, has `dcolog = on`, and already has one `SNAPDONE` mirror and is tagged with `db2ed_flashsnap`.

   ```
   # vxsnap -g dg_a  addmir dg_a_vol1 alloc=dg_a03
   # vxedit -g dg_a set putil2=db2ed_flashsnap dg_a_vol1-03
   ```

2. Here is a sample snapplan.

   ```
   SNAPSHOT_VERSION=4.0
   PRIMARY_HOST=host1
   SECONDARY_HOST=host1
   PRIMARY_DG=PRODdg
   SNAPSHOT_DG=SNAP_PRODdg
   DB2Database=PROD
   redolog_dest=/prod_ar
   ```

```
SNAPSHOT_ARCHIVE_LOG=yes
SNAPSHOT_MODE=online
SNAPSHOT_PLAN_FOR=database
SNAPSHOT_PLEX_TAG=db2ed_flashsnap
SNAPSHOT_MIRROR=2
SNAPSHOT_VOL_PREFIX=SNAP_
ALLOW_REVERSE_RESYNC=no
SNAPSHOT_MIRROR=2
```

Validating a Snapplan (db2ed_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 `db2ed_vmchecksnap` command with the `-o validate` option.

Prerequisites

- The database must be up and running while executing the `db2ed_vmchecksnap` command.

Usage Notes

- The `db2ed_vmchecksnap` command must be run as the DB2 instance owner.
- After validating the snapplan, you have the option of modifying the snapplan file to meet your storage configuration requirements.
- When using `db2ed_vmchecksnap` 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.
- The `db2ed_vmchecksnap` command must be used on the primary host.
- See the `db2ed_vmchecksnap`(1M) manual page for more information.

▼ 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 `db2ed_vmchecksnap` command:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D DB2DATABASE \
-f SNAPPLAN -o validate
```

Note In HA environment, you must modify the default snapplan, use the virtual host name defined for the resource group for the PRIMARY_HOST and/or SECONDARY_HOST, and run validation.

For information on validating a snapplan using the GUI, see "Using the VERITAS Storage Foundation for DB2 Graphical User Interface" on page 221.

Example

In the following example, a snapplan, `snap1`, is validated for a snapshot image in a single-host configuration. The primary host is `host1` and the working directory is `/export/snap_dir`.

```
$ cd /export/snap_dir
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD \
-f snap1 -o validate

PRIMARY_HOST is host1

SECONDARY_HOST is host1

The version of PRIMARY_DG-PRODdg is 110.

SNAPSHOT_DB is SNAP_PRODdg

SNAPSHOT_PLAN_FOR is database

Examining DB2 volume and disk layout for snapshot

Volume prod_db on PRODdg is ready for snapshot.
Original plex and DCO log for prod_db is on PRODdg01.
Snapshot plex and DCO log for prod_db is on PRODdg02.

SNAP_PRODdg for snapshot will include: PRODdg02

ALLOW_REVERSE_RESYNC is no

The snapplan snap1 has been created.
```

Example

In the following example, a snapplan, `snap2`, is validated for a snapshot image in a two-host configuration. The primary host is `host1`, the secondary host is `host2`, and the working directory is `/export/snap_dir`.

```
$ cd /export/snap_dir
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD \
-f snap2 -o validate
PRIMARY_HOST is host1

SECONDARY_HOST is host2

The version of PRIMARY_DG-PRODdg is 110.

SNAPSHOT_DB is SNAP_PRODdg

SNAPSHOT_PLAN_FOR is database

Examining DB2 volume and disk layout for snapshot.

Volume arch on PRODdg is ready for snapshot.
Original plex and DCO log for arch is on PRODdg01.
Snapshot plex and DCO log for arch is on PRODdg02.

SNAP_PRODdg for snapshot will include: PRODdg02

ALLOW_REVERSE_RESYNC is yes

The snapplan snap2 has been created.
```

Displaying, Copying, and Removing a Snapplan (db2ed_vmchecksnap)

This section explains how to list all snapplans for a specific DB2 database, display a snapplan file, and copy and remove snapplans.

Usage Notes

- If the local snapplan is updated or modified, you must re-validate it.
- If the database schema or disk group is modified, you must revalidate.

▼ To list all available snapplans for a specific DB2 database

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D DB2DATABASE -o list
```

Example

In the following example, all available snapplans are listed for the database `PROD`.

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD -o list
The following snapplan(s) are available for PROD:

SNAP_PLAN      SNAP_STATUS       DB_STATUS       SNAP_READY
snap1          init_full         init            yes
snap2          init_full         init            yes
```

Note 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 explanations of the various statuses that may appear for `SNAP_STATUS` and `DB_STATUS`, refer to "VERITAS Database FlashSnap Status Information" on page 419

▼ To display detailed information for a snapplan

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D \
DB2DATABASE -f SNAPPLAN -o list
```

Example

In the following example, the snapplan `snap1` is displayed.

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD -f snap1 -o list
SNAPSHOT_VERSION=4.0
PRIMARY_HOST=host1
SECONDARY_HOST=host1
PRIMARY_DG=PRODdg
SNAPSHOT_DG=SNAP_PRODdg
DB2DATABASE=PROD
DB2HOME=/PROD_HOME
REDOLOG_DEST=/PROD_HOME/inst1/NODE0000/SQL00001/SQLOGDIR/
SNAPSHOT_MODE=online_snapshot
SNAPSHOT_PLAN_FOR=database
SNAPSHOT_PLEX_TAG=PRODtag
SNAPSHOT_MIRROR=1
SNAPSHOT_VOL_PREFIX=SNAP_
ALLOW_REVERSE_RESYNC=yes
STORAGE_INFO
PRODdg02
SNAP_PLEX=prod_db-02

STATUS_INFO
SNAP_STATUS=init_full
DB_STATUS=init
LOCAL_SNAPPLAN=/export/snap_dir/snap1
```

▼ To copy a snapplan from the VxDBA repository to your current directory

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 VxDBA repository to your current directory, the snapplan must not already be present in the current directory.

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D DB2DATABASE \
-f SNAPPLAN -o copy
```

Example

In the following example, the snapplan, `snap1`, is copied from the VxDBA repository to the current directory.

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD \
-f snap1 -o copy
Copying 'snap1' to '/export/snap_dir'
```

▼ To remove a snapplan from the VxDBA repository

A snapplan can be removed from a local directory or repository if the snapplan is no longer needed.

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D DB2DATABASE -f\
SNAPPLAN -o remove
```

Example

In the following example, the snapplan, `snap1`, is removed from the VxDBA repository.

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD -f snap1 -o remove
The snapplan snap1 has been removed.
```

Creating a Snapshot (db2ed_vmsnap)

The `db2ed_vmsnap` command creates a snapshot of a DB2 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 `db2ed_vmsnap` is a frozen image of a DB2 database's containers.

For a complete list of all snapshot and database statuses, see "VERITAS Database FlashSnap Status Information" on page 419.

Note You cannot access Database FlashSnap commands (`db2ed_vmchecksnap`, `db2ed_vmsnap`, and `db2ed_vmclonedb`) with the VxDBA menu utility.

Prerequisites

- You must be logged in as the DB2 instance owner.
- You must create and validate a snapplan using `db2ed_vmchecksnap` before you can create a snapshot image with `db2ed_vmsnap`.

Usage Notes

- The `db2ed_vmsnap` command can only be used on the primary host.
- When creating a snapshot volume, create the snapshot on a separate controller and on separate disks from the primary volume.
- The online redo log must be included in the snapshot if a clone database is to be started.
- 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 `db2ed_vmsnap`(1M) manual page for more information.

▼ 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, terminate all connections to the database.

3. Create the snapshot image using the `db2ed_vmsnap` command:

   ```
   $ /opt/VRTS/bin/db2ed_vmsnap -D DB2DATABASE -f SNAPPLAN \
   -o snapshot [-F]
   ```

Note 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 VERITAS Storage Foundation *for DB2* commands. (That is, the volumes were synchronized without using VERITAS Storage Foundation *for DB2* 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.

Note After the snapshot is created, `db2ed_vmsnap` returns values you will need to run `db2ed_vmclonedb`. These values include the snapshot disk group, the snapplan name, and the VxDBA repository volume for a two-host configuration. Make a note of these values so you have them when running `db2ed_vmclonedb`.
You can also use the command `db2ed_vmchecksnap -f snapplan -o list` to access the information regarding the snapshot disk group, the snapplan name, and the VxDBA repository.

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. For details, see “Backing Up the Database from Snapshot Volumes (db2ed_vmclonedb)” on page 194. You can also create another DB2 database for decision-support purposes. See “Cloning a Database (db2ed_vmclonedb)” on page 199 for more information.

For information on creating a snapshot using the GUI, see “Using the VERITAS Storage Foundation for DB2 Graphical User Interface” on page 221.

Example

In this example, a snapshot image of the database, `PROD`, is created for a single-host configuration. In this case, the `SECONDARY_HOST` parameter is set the same as the `PRIMARY_HOST` parameter in the snapplan.

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 -o snapshot

   Database Connection Information

 Database server        = DB2/6000 8.1.0
 SQL authorization ID   = INST01
 Local database alias   = PROD

db2ed_vmsnap started at 2004-05-26 10:58:23
DB20000I  The SET WRITE command completed successfully.
DB20000I  The SET WRITE command completed successfully.
A snapshot of DB2DATABASE PROD is in DG SNAP_DB2dg.
Snapplan snap1 is used for the snapshot.

If -r <relocate_path> is used in db2ed_vmclonedb,
make sure <relocate_path> is created and owned by DB2 Instance
Owner.
Otherwise, the following mount points need to be
created and owned by DB2 Instance Owner:
        /db2/testvol01.
        /db2/testvol02.
        /db2/testvol03.
        /db2/testvol04.
        /db2/testvol05.
        /db2/udb_home.

db2ed_vmsnap ended at 2004-05-26 10:59:25
DB20000I  The TERMINATE command completed successfully.
```

Example

In this example, a snapshot image of the primary database, `PROD`, is created for a two-host configuration. In this case, the `SECONDARY_HOST` parameter specifies a different host name than the `PRIMARY_HOST` parameter in the snapplan.

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap2 -o snapshot

$ db2ed_vmsnap -D PROD -f snap2 -o snapshot

   Database Connection Information

 Database server        = DB2/6000 8.1.0
 SQL authorization ID   = INST01
 Local database alias   = PROD

db2ed_vmsnap started at 2004-05-26 11:02:54
DB20000I  The SET WRITE command completed successfully.
DB20000I  The SET WRITE command completed successfully.
A snapshot of DB2DATABASE PROD is in DG SNAP_PRODdg.
Snapplan snap2 is used for the snapshot.
VxDBA repository volume is SNAP_testvol01.

If -r <relocate_path> is used in db2ed_vmclonedb,
make sure <relocate_path> is created and owned by DB2 Instance
Owner.
Otherwise, the following mount points need to be
created and owned by DB2 Instance Owner:

        /db2/testvol01.
        /db2/testvol02.
        /db2/testvol03.
        /db2/testvol04.
        /db2/testvol05.
        /db2/udb_home.

db2ed_vmsnap ended at 2004-05-26 11:03:57
DB20000I  The TERMINATE command completed successfully.
```

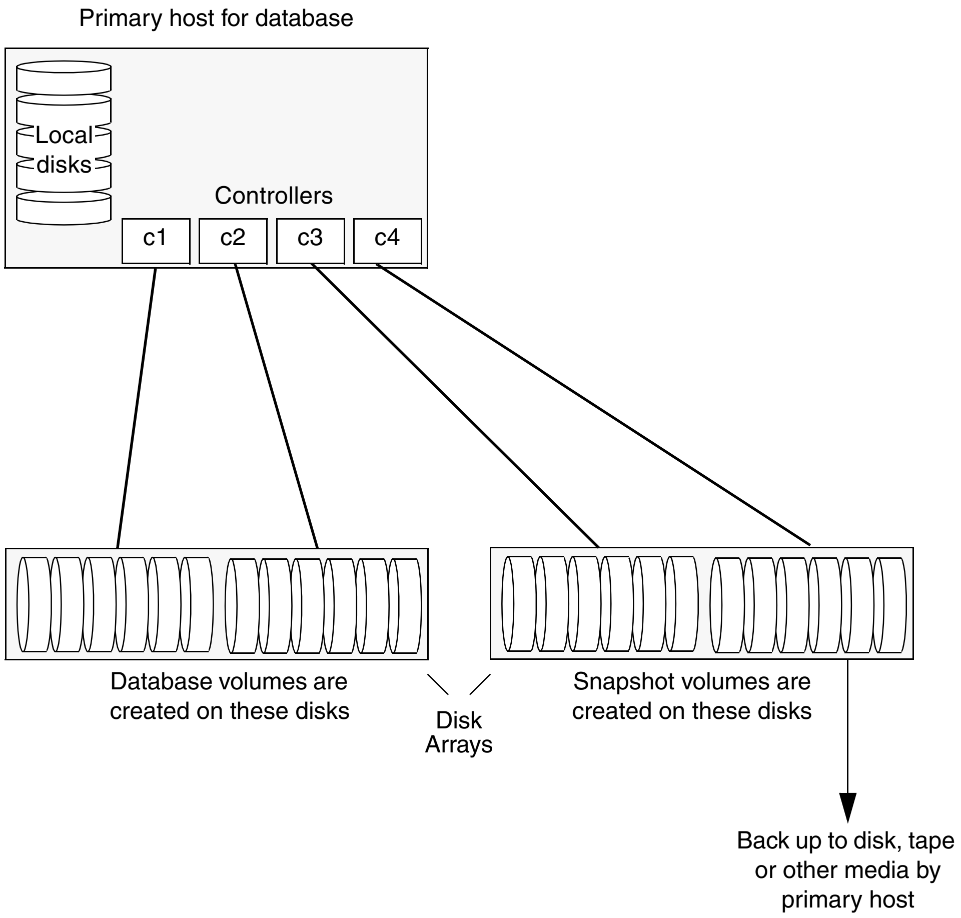
Backing Up the Database from Snapshot Volumes (db2ed_vmclonedb)

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 datafiles to tape using snapshot volumes. You can also use the DB2 Backup utility to backup a snapshot database if all the tablespaces are DMS type.

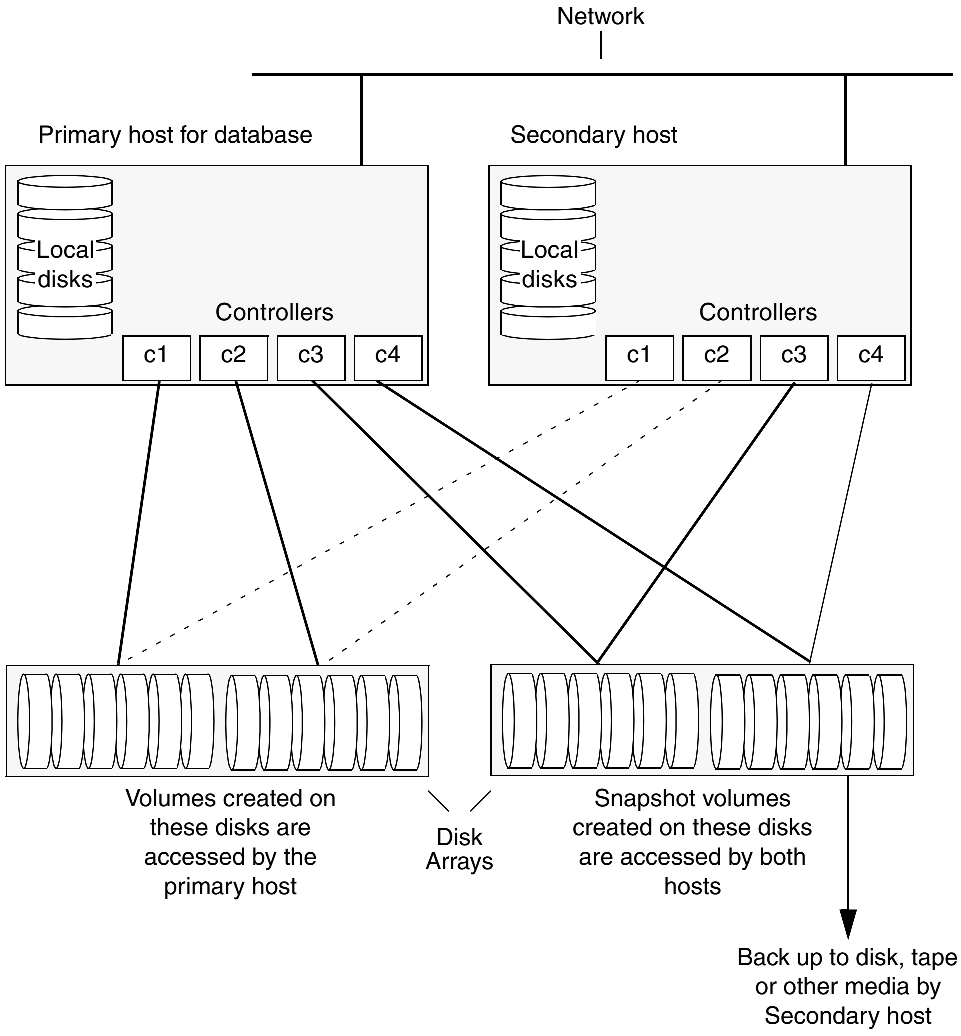
The figure below shows a typical configuration when snapshot volumes are located on the primary host.

Example System Configuration for Database Backup on the Primary Host


Primary host for database
Local disks
Controllers
c1
c2
c3
c4
Database volumes are created on these disks
Disk Arrays
Snapshot volumes are created on these disks
Back up to disk, tape or other media by primary host

The following figure, shows a typical configuration when snapshot volumes are used on a secondary host.

Example System Configuration for Database Backup on a Secondary Host


Network
Primary host for database
Secondary host
Local disks
Controllers
c1
c2
c3
c4
Local disks
Controllers
c1
c2
c3
c4
Volumes created on these disks are accessed by the primary host
Disk Arrays
Snapshot volumes created on these disks are accessed by both hosts
Back up to disk, tape or other media by Secondary host

Prerequisites

- You must be logged in as the DB2 instance owner to use `db2ed_vmclonedb` command.
- Before you can use the `db2ed_vmclonedb` command, you must complete the steps in "Summary of Database Snapshot Steps" on page 170, "Validating a Snapplan (db2ed_vmchecksnap)" on page 184, and "Creating a Snapshot (db2ed_vmsnap)" on page 190.
- The volume snapshot must contain the entire database.
- Before you can use the `db2ed_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 DB2 instance owner.
- The `SNAPSHOT_MODE` must be `online_snapshot` or `offline`. If `SNAPSHOT_MODE` is set to `offline`, a two-host configuration is required.

Usage Notes

- The `db2ed_vmclonedb` command can be used on the secondary host.
- In a single-host configuration, the primary and secondary hosts are the same.
- In a single-host configuration, `-r relocate_path` is required.
- In a two-host configuration, the `vxdbavol=`*`vol_name`* option is required.
-
- See the `db2ed_vmclonedb`(1M) manual page for more information.

Note You cannot access Database FlashSnap commands (`db2ed_vmchecksnap`, `db2ed_vmsnap`, and `db2ed_vmclonedb`) with the VxDBA menu utility.

Mounting the Snapshot Volumes and Backing Up

Before using the snapshot volumes to do a backup, you must first mount them as follows.

▼ To mount the snapshot volumes

Use the `dbed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D DB2DATABASE -g snap_dg \
-o mount,new_db=new_db -f SNAPPLAN \
[-r relocate_path]
```

You can now backup an individual file or a group of files under a directory onto the backup media.

Example

In this example, snapshot volumes are mounted.

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o mount,new_db=NEWPROD -f snap1 -r /clone/single
db2ed_vmclonedb started at 2004-04-02 15:35:41
Mounting /clone/db2/testvol on
/dev/vx/dsk/SNAP_PRODdg/SNAP_testvol.
Mounting /clone/db2/udb_home on
/dev/vx/dsk/SNAP_PRODdg/SNAP_udb_home.
db2ed_vmclonedb ended at 2004-04-02 15:35:50
```

Restoring from Backup

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. See "Resynchronizing Your Database to the Snapshot" on page 209 for more information.

Cloning a Database (db2ed_vmclonedb)

This section explains how to create a clone database using the 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 (db2ed_vmclonedb)" on page 194 for more information. You can backup 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 single-host configuration, the `db2ed_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 single-host 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 a two-host configuration, the `db2ed_vmclonedb` command imports the snapshot disk group `SNAP_dg`, mounts the file systems on the snapshot volumes, 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. When creating the clone off host, `-o vxdbavol=vol_name` is required. `db2ed_vmsnap` picks one of the snapshot volumes to assign metadata information for future use. `db2ed_vmsnap snapshot` displays the selected volume, `SNAP_volume`, and this volume is used as arguments for `vxdbavol`.

Prerequisites

- You must be logged in as the DB2 instance owner.
- Before you can use the `db2ed_vmclonedb` command, you must complete the steps in "Validating a Snapplan (db2ed_vmchecksnap)" on page 184, "Creating a Snapplan (db2ed_vmchecksnap)" on page 177, and "Creating a Snapshot (db2ed_vmsnap)" on page 190.
- 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 `db2ed_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 DB2 instance owner.
- If `SNAPSHOT_MODE` is set to `offline`, a two-host configuration is required.

Usage Notes

- The `db2ed_vmclonedb` command can be used on either the primary or secondary host as specified in the snapplan.
- In a single-host configuration, `-r` *`relocate_path`* is required. This command is also needed if the name of the clone database is different than the primary database.
- In a two-host configuration, the `vxdbavol=`*`vol_name`* option is required.
- See the `db2ed_vmclonedb`(1M) manual page for more information.

Note When using the `-o recoverdb` option to clone a database, any database within the target instance that has the same name as the primary database to be cloned will be temporarily uncataloged and therefore unavailable.

Caution If the primary database is being cloned on the same host and within the same instance, it will be temporarily uncataloged while `db2ed_vmclonedb` is running. The database will be recataloged on completion of `db2ed_vmclonedb`.

▼ To mount a database and recover it manually

1. Start and mount the clone database to allow manual database recovery:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D DB2DATABASE -g snap_dg \
-o mount,new_db=db_name[,vxdbavol=vol_name] -f SNAPPLAN \
[-r relocate_path]
```

2. Recover the database manually.

   a. Create a configuration file with content similar to the following:

```
DB_NAME=old_db,new_db
DB_PATH=old_dbpath,new_dbpath
INSTANCE=old_instance,new_instance
LOG_DIR=old_logdir,new_logdir
```

```
CONT_PATH=old_containerpath1,new_containerpath1
CONT_PATH=old_containerpath2,new_containerpath2
```

b. Run the following command:

```
db2inidb CLONE_DB2DATABASE as snapshot relocate using \
configfile
```

Note When this command is run, DB2 initiates a crash recovery and brings up the clone database.

After this command is run, the primary database is uncataloged.

c. Recatalog the primary database:

```
db2 catalog database DB2DATABASE as snapshot /new_dbpath
```

For detailed information about manual database recovery, refer to your DB2 documentation.

3. Update the snapshot status information for the clone database in the VxDBA repository:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -o update_status,new_db=db_name \
-f SNAPPLAN [-r relocate_path]
```

Example

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/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o mount,new_db=NEWPROD -f snap1 -r /clone
db2ed_vmclonedb started at 2004-04-01 13:30:12
Mounting /clone/testvol on
/dev/vx/dsk/SNAP_PRODdg/SNAP_testvol.
Mounting /clone/udb_home on
/dev/vx/dsk/SNAP_PRODdg/SNAP_udb_home.
db2ed_vmclonedb ended at 2004-04-01 13:30:24
```

The database is recovered manually using `db2initdb`.

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/db2ed_vmclonedb -o update_status,new_db=NEWPROD \
-f snap1 -r /clone
db2ed_vmclonedb started at 2004-04-01 13:37:09
The snapshot status has been updated.
db2ed_vmclonedb ended at 2004-04-01 13:37:23
```

Example

In this example, file systems are mounted *without recovering the clone database*. The clone database must be manually recovered before it can be used. This example is for a clone created on a secondary host.

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D -g SNAP_PRODdg \
-o mount,new_db=NEWPROD,vxdbavol=SNAP_arch -f snap2

db2ed_vmclonedb started at 2004-04-06 09:06:09
Mounting /clone/testvol on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol.
Mounting /clone/udb_home on
/dev/vx/dsk/SNAP_DB2dg/SNAP_udb_home.
db2ed_vmclonedb ended at 2004-04-06 09:06:43
```

The database is recovered manually.

The snapshot status (`database_recovered`) is updated for a clone database on a secondary host after manual recovery has been completed.

```
$ /opt/VRTS/bin/db2ed_vmclonedb -o update_status,new_db=NEWPROD \
-f snap2
db2ed_vmclonedb started at 2004-04-06 09:22:27
The snapshot status has been updated.
db2ed_vmclonedb ended at 2004-04-06 09:22:40
```

▼ To clone the database automatically

Use the `db2ed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D DB2DATABASE -g snap_dg \
-o recoverdb,new_db=db_name[,vxdbavol=vol_name] -f SNAPPLAN \
[-r relocate_path]
```

Where:

- ◆ *`DB2DATABASE`* is the name of the DB2 database used to create the snapshot.
- ◆ *`snap_dg`* is the name of the diskgroup that contains all the snapshot volumes.
- ◆ `db_name` specifies the name of the clone database.

- `vxdbavol` is the volume that contains the snapplan data. This name is provided after you run `db2ed_vmsnap -o snapshot`.
- *`SNAPPLAN`* is the name of the snapplan file.
- *`relocate_path`* is the name of the initial mount point for the snapshot image.

Note When cloning a database on a secondary host, ensure that `PRIMARY_HOST` and `SECONDARY_HOST` parameters in the snapplan file are different.

For information on cloning a database using the GUI, see “Using the VERITAS Storage Foundation for DB2 Graphical User Interface” on page 221.

Example

In the following example, a clone of the primary database is automatically created on the same host as the primary database.

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o recoverdb,new_db=NEWPROD -f snap1 -r /clone
db2ed_vmclonedb started at 2004-04-01 13:18:02
Mounting /clone/testvol on
/dev/vx/dsk/SNAP_PRODdg/SNAP_testvol.
Mounting /clone/udb_home on
/dev/vx/dsk/SNAP_PRODdg/SNAP_udb_home.
Relocating database...
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
Database relocation was successful.
DBT1000I  The tool completed successfully.
DB20000I  The CATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the
directory
cache is refreshed.
db2ed_vmclonedb ended at 2004-04-01 13:18:40
```

Example

In the following example, a clone of the primary database is automatically created on a secondary host.

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o recoverdb,new_db=NEWPROD,vxdbavol=SNAP_arch -f snap2
db2ed_vmclonedb started at 2004-04-06 07:54:09
Mounting /clone/testvol on /dev/vx/dsk/SNAP_DB2dg/SNAP_testvol.
Mounting /clone/udb_home on /dev/vx/dsk/SNAP_DB2dg/SNAP_udb_home.
Relocating database...
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
Database relocation was successful.
DBT1000I  The tool completed successfully.
db2ed_vmclonedb ended at 2004-04-06 07:55:00
```

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 `db2ed_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 `db2ed_vmclonedb -o umount`.

▼ To shut down the clone database and unmount all snapshot file systems

Use the `db2ed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -o umount,new_db=db_name \
-f SNAPPLAN [-r relocate_path]
```

Example

In this example, the clone database is shut down and file systems are unmounted for a clone on the same host as the primary database (a single-host configuration).

```
$ /opt/VRTS/bin/db2ed_vmclonedb -o umount,new_db=NEWPROD \
-f snap1 -r /clone
db2ed_vmclonedb started at 2004-04-02 15:11:22
NOTICE: Umounting /clone/prod_db.
NOTICE: Umounting /clone/prod_ar.
db2ed_vmclonedb ended at 2004-04-02 15:11:47
```

Example

In this example, the clone database is shut down, file systems are unmounted, and the snapshot disk group is deported for a clone on a secondary host (a two-host configuration).

```
$ /opt/VRTS/bin/db2ed_vmclonedb -o umount,new_db=NEWPROD \
-f snap2
db2ed_vmclonedb started at 2004-04-09 23:09:21
NOTICE: Umounting /clone/arch.
NOTICE: Umounting /clone/prod_db.
db2ed_vmclonedb ended at 2004-04-09 23:09:50
```

Restarting a Clone Database

If the clone database is down as a result of using `db2ed_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 `db2ed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D DB2DATABASE -g snap_dg \
-o restartdb,new_db=db_name -f SNAPPLAN  \
[-r relocate_path]
```

For information on restarting the clone database using the GUI, see "Using the VERITAS Storage Foundation for DB2 Graphical User Interface" on page 221.

Example

In this example, the clone database is re-started on the same host as the primary database (a single-host configuration).

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o restartdb,new_db=NEWPROD -f snap1 -r /clone
db2ed_vmclonedb started at 2004-05-26 13:20:58
Mounting /clone/testvol01 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol01.
Mounting /clone/testvol02 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol02.
Mounting /clone/testvol03 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol03.
Mounting /clone/testvol04 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol04.
Mounting /clone/testvol05 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol05.
Mounting /clone/udb_home on /dev/vx/dsk/SNAP_DB2dg/SNAP_udb_home.
Files and control structures were changed successfully.
Database was cataloged successfully.
DBT1000I  The tool completed successfully.
db2ed_vmclonedb ended at 2004-05-26 13:21:48
```

Example

In this example, the clone database is re-started on the secondary host (a two-host configuration).

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o restartdb,new_db=NEWPROD,vxdbavol=SNAP_arch -f snap2
db2ed_vmclonedb started at 2004-05-26 10:14:21
Mounting /clone/testvol01 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol01.
Mounting /clone/testvol02 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol02.
Mounting /clone/testvol03 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol03.
Mounting /clone/testvol04 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol04.
Mounting /clone/testvol05 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol05.
Mounting /clone/udb_home on /dev/vx/dsk/SNAP_DB2dg/SNAP_udb_home.
Files and control structures were changed successfully.
Database was cataloged successfully.
DBT1000I  The tool completed successfully.
db2ed_vmclonedb ended at 2004-05-26 10:15:15
```

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.

There are two choices when resynchronizing the data in a volume:

- Resynchronizing the snapshot from the original volume. This option 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.
- The recovery method is different depending on how the `SNAPSHOT_MODE` is set in the snapplan. See "Selecting the Snapshot Mode" on page 157.

Prerequisites

- You must be logged in as the DB2 instance owner.
- Before you can resynchronize the snapshot image, you must complete the steps in "Summary of Database Snapshot Steps" on page 170, "Validating a Snapplan (db2ed_vmchecksnap)" on page 184, and "Creating a Snapshot (db2ed_vmsnap)" on page 190.
- If a clone database has been created, shut it down and unmount the file systems using the `db2ed_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 204.

Usage Notes

- The `db2ed_vmsnap` command can only be executed on the primary host.
- In a two-host configuration, the `db2ed_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 `db2ed_vmsnap`(1M) manual page for more information.

- You cannot access the Database FlashSnap commands (`db2ed_vmchecksnap`, `db2ed_vmsnap`, and `db2ed_vmclonedb`) with the VxDBA menu utility.

Note If you plan to resynchronize the snapshot image immediately after taking a snapshot, wait for approximately a minute or two after taking the snapshot and before running `db2ed_vmsnap -o resync` to allow time for VERITAS Enterprise Administrator to refresh the objects.

▼ To resynchronize the snapshot image

Use the `db2ed_vmsnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmsnap -D DB2DATABASE -f SNAPPLAN -o resync
```

For information on resynchronizing the snapshot using the GUI, see "Using the VERITAS Storage Foundation for DB2 Graphical User Interface" on page 221.

Example

In this example, the snapshot image is resynchronized with the primary database.

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 -o resync
db2ed_vmsnap started at 2004-04-02 16:19:05
The option resync has been completed.
db2ed_vmsnap ended at 2004-04-02 16:19:26
```

Now, you can again start creating snapshots.

Resynchronizing Your Database to the Snapshot

If your database becomes corrupted, you can use reverse resynchronization to recover the database from a clone. The reverse resynchronization feature of VERITAS Database FlashSnap enables you to resynchronize the primary database or volume with a clone database or snapshot volume.

Reverse resynchronization requires the primary database to be inactive so that it remains unchanged.

Caution Upon completion of reverse resynchronization, the content of the original database is discarded. Storage Checkpoints taken on either the original database or the clone database *after* the snapshot was created are discarded. Storage Checkpoints taken *before* the snapshot was created are preserved. The `db2ed_vmsnap -o reverse_resync_commit` command cannot be undone and should be used with extreme caution.

Prerequisites

◆ You must be logged in as the DB2 instance owner.

◆ Before you can reverse resynchronize the snapshot image, you must complete the steps in “Summary of Database Snapshot Steps” on page 170 and “Creating a Snapshot (db2ed_vmsnap)” on page 190.

◆ The mount point for the primary database must be created by and owned by the DB2 instance owner before mounting the VxFS file system.

◆ If a clone database has been created, you must shut it down and unmount the file systems using the `db2ed_vmclonedb -o umount` command before you can reverse resynchronize the snapshot image. 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 204.

◆ The primary database must be inactive.

Usage Notes

◆ The `db2ed_vmsnap` command can only be executed on the primary host.

Note You cannot access Database FlashSnap commands (`db2ed_vmchecksnap`, `db2ed_vmsnap`, and `db2ed_vmclonedb`) from the VxDBA menu utility.

▼ To begin reverse resynchronization

The `-o reverse_resync_begin` option of the `db2ed_vmsnap` command imports the disk group that was deported from the secondary host (in a two-host configuration) and joins it back to the original disk group. The command unmounts the original volumes, mounts the snapshot volumes with the file systems that are configured for the primary database, and brings up the database snapshot image as the primary database. This operation requires the primary database to be offline so that its contents remain unchanged.

To begin reverse resynchronization, use the `-o reverse_resync_begin` option as follows:

```
$ /opt/VRTS/bin/db2ed_vmsnap -D DB2DATABASE -f SNAPPLAN \
-o reverse_resync_begin
```

Note Any mounted storage checkpoints must be unmounted before running `db2ed_vmsnap -o reverse_resync`.

After executing `reverse_resync_commit`, checkpoints created on the original database or clone database will be deleted.

Limitation

Reverse resynchronization requires the primary database to be down. However, in an HA environment, when VCS detects that the primary database is down, it starts the fail over process, VxDBA repository gets umounted, `db2ed_vmsnap` dies.

Use the following workaround:

1. To temporarily freeze the VCS Resource Group for the database, enter:

   ```
   # hagrp -freeze ResourceGroup
   ```

2. Shutdown the primary database.

3. Run `reverse_resync`.

4. When `reverse_resync` is completed, start up the database. Make sure it is in `archivelog` mode.

5. To unfreeze the Resource Group, enter:

   ```
   # hagrp -unfreeze ResourceGroup
   ```

▼ To abort reverse resynchronization

The `-o reverse_resync_abort` option aborts `-o reverse_resync_begin`, unmounts the snapshot volumes, and mounts the original volumes back with the file systems that are configured to use the volume. This operation is only allowed after `-o reverse_resync_begin` has been executed and cannot be used after reverse resynchronization has been committed (`-o reverse_resync_commit`).

To abort reverse resynchronization, use the `-o reverse_resync_begin` option as follows:

```
$ /opt/VRTS/bin/db2ed_vmsnap -D DB2DATABASE -f SNAPPLAN \
-o reverse_resync_abort
```

Note If your snapshot was created with `SNAPSHOT_MODE` set to `online_mirror`, you cannot run `db2ed_vmsnap -o reverse_resync_begin` after running `db2ed_vmsnap -o reverse_resync_abort`. You must take a new snapshot before performing reverse resynchronization again.

▼ To commit reverse resynchronization changes

The `-o reverse_resync_commit` option commits the reverse resynchronization changes after you have verified that they are acceptable. The operation resynchronizes the original volume from the data in the snapshot.

To commit the reverse resynchronization changes, use the `-o reverse_resync_commit` option as follows:

```
$ /opt/VRTS/bin/db2ed_vmsnap -D DB2DATABASE -f SNAPPLAN \
-o reverse_resync_commit
```

For information on resynchronizing the snapshot using the GUI, see "Using the VERITAS Storage Foundation for DB2 Graphical User Interface" on page 221.

Example

Note The following example is valid only for the `online_snapshot` mode.

Reverse resynchronization is started on the primary host.

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 \
-o reverse_resync_begin
db2ed_vmsnap started at 2004-05-26 13:37:11
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
db2ed_vmsnap ended at 2004-05-26 13:37:48
```

Reverse resychronization is aborted on the primary host.

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 \
-o reverse_resync_abort
db2ed_vmsnap started at 2004-05-26 13:38:16
DB20000I  The FORCE APPLICATION command completed successfully.
DB21024I  This command is asynchronous and may not be effective
immediately.

DB20000I  The UNCATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the
directory cache is refreshed.
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
DB20000I  The UNCATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the
directory cache is refreshed.
DB20000I  The CATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the
directory cache is refreshed.
The option reverse_resync_abort has been completed.
db2ed_vmsnap ended at 2004-05-26 13:39:31
```

Reverse resychronization changes are committed on the primary host.

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 \
-o reverse_resync_commit
db2ed_vmsnap started at 2004-05-26 13:40:32
DB20000I  The FORCE APPLICATION command completed successfully.
DB21024I  This command is asynchronous and may not be effective
immediately.

DB20000I  The UNCATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the
directory cache is refreshed.
DB20000I  The CATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the
directory cache is refreshed.
db2ed_vmsnap ended at 2004-05-26 13:41:35
```

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 root.
- 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:

   a. 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:

   b. 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/db2ed_vmchecksnap -D db -f snapplan -o remove
   ```

Example

To 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
```

Using Database FlashSnap in an HA Environment

VERITAS Storage Foundation 4.1 *for DB2* supports FlashSnap in the HA environment.

When using FlashSnap in the HA environment, observe the following limitations:

- Modify the default snapplan to use the virtual host name defined for the database resource group for the `PRIMARY_HOST` and/or the `SECONDARY_HOST` parameters and validate the snapplan before creating a snapshot by running the following command:

  ```
  db2ed_vmchecksnap -D DB2DATABASE \
  -f SNAPPLAN -o validate
  ```

- The primary database must be down before you perform reverse resychronization (`db2ed_vmsnap -D DB2DATABASE -f SNAPPLAN -o reverse_resync_begin`). When VERITAS Cluster Server (VCS) detects that the primary database is down, it starts the failover process and the VxDBA repository is unmounted and the `db2ed_vmsnap` command is aborted.

To avoid the VCS failover process, perform the following steps:

1. As root, temporarily freeze the VCS resource group for the database:

   ```
   # hagrp -freeze ResourceGroup
   ```

2. Shut down the primary database.

3. Run reverse resynchronization:

   ```
   # db2ed_vmsnap D DB2DATABASE -f SNAPPLAN -o\
    reverse_resync_begin
   ```

4. After reverse reynchronization changes are committed (`-o reverse_resync_commit`), verify that the database has been started in `ARCHIVELOG` mode. This information is provided in the status messages that appear after running committing reverse resynchronization changes.

5. Unfreeze the resource group:

   ```
   # hagrp -unfreeze ResourceGroup
   ```

Using VERITAS NetBackup for Database Backup 9

VERITAS NetBackup™ is a separately licensed product and is not included with VERITAS Storage Foundation *for DB2*. The information included here is for reference only.

With VERITAS NetBackup, you can perform high performance, online (hot) backups of databases that must be available on a 24x7 basis. NetBackup supports the Extended Edition (EE) and the Enterprise Extended Edition (EEE) environments. NetBackup also supports DPF for DB2 8.1 and higher.

Topics include:

For complete information on any of these topics and supported EE and EEE versions, see the *VERITAS NetBackup for DB2 System Administrator's Guide for UNIX*.

VERITAS NetBackup Features

VERITAS NetBackup for DB2 has the following features:

- Media and device management
- Scheduling facilities
- Multiplexed backups and restores
- Transparent execution of both DB2 and regular file system backup and restore operations
- Shared devices and tapes used during other file backups
- Centralized and networked backup operations
- Parallel backup and restore operations
- Incremental backups of DB2 databases

Using VERITAS NetBackup for Backup and Restore

VERITAS NetBackup lets you back up and restore database files and directories. You can set up schedules for automatic, unattended 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. The topics below summarize NetBackup's backup and restore capabilities for DB2. For more information on backup and restore procedures, see "Using NetBackup for DB2 on UNIX" in the *VERITAS NetBackup for DB2 System Administrator's Guide for UNIX*.

Performing a Backup

There are two types of DB2 backups: database and archive logs. These two types of backups can be performed by NetBackup automatically, manually, or by using the `DB2 BACKUP DATABASE` command. For more information on performing a backup, see "Performing a Backup" in the *VERITAS NetBackup for DB2 System Administrator's Guide for UNIX*.

Backing up a DB2 Policy

You can back up a DB2 policy automatically or manually. The most convenient way to back up your database is to set up schedules for automatic backups.

Using DB2 to Perform a Restore

The procedure for restoring a DB2 database depends on the database involved and the problems that you have on your system. See the *DB2 UDB Administration Guide Data Recovery and High Availability Guide* for a complete description of how to recover a DB2 database. You can browse the backups using the `db2 list history` command or using the NetBackup `bplist` command before restoring.

Configuring VERITAS NetBackup for a DB2 EEE (DPF) Environment

VERITAS NetBackup can be configured for DB2 in an Extended Edition (EE), Extended-Enterprise Edition (EEE), or Database Partitioning Feature (DPF) environment. Two types of DB2 backup policies are required. One is used to backup the catalog nodes and the other is used to backup all the nodes, including the catalog node. For detailed information and instructions on configuring DB2 for EEE, see "Configuration for a DB2 EEE (DPF) Environment" in the *VERITAS NetBackup for DB2 System Administrator's Guide for UNIX*.

Using NetBackup to Backup and Restore Quick I/O Files

VERITAS 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.

For example:

```
$ ls -la /db01
total 2192
drwxr-xr-x2 root   root 96      Oct 20 17:39 .
drwxr-xr-x9 root   root 8192    Oct 20 17:39 ..
-rw-r--r--1 db2    dba  1048576 Oct 20 17:39 .dbfile
lrwxrwxrwx1 db2    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 tightly integrated with the VERITAS Database Edition *for DB2*, 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 of or both of the two components are missing, NetBackup creates or overwrites as needed.

Using the VERITAS Storage Foundation for DB2 Graphical User Interface 10

You can access VERITAS Storage Foundation *for DB2*, VERITAS Volume Manager, and VERITAS File System functions through the VERITAS Storage Foundation *for DB2* graphical user interface (GUI). This chapter describes only how to use the GUI to perform various storage management tasks for your DB2 database.

Topics covered in this chapter include:

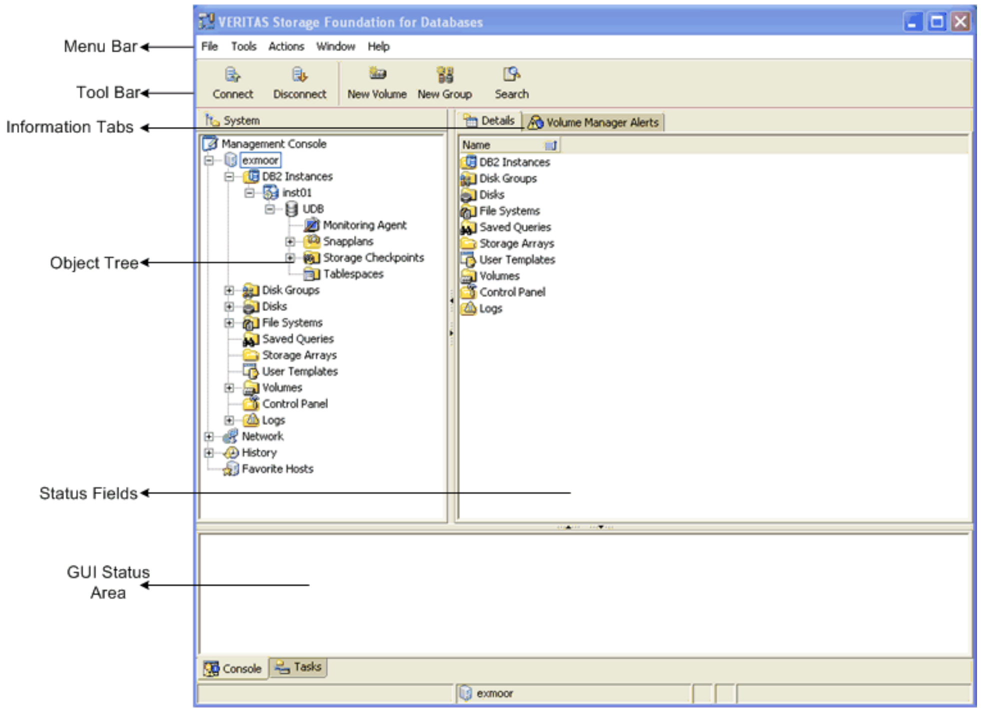
VERITAS Storage Foundation for DB2 GUI

The graphical user interface (GUI) allows you to perform storage management duties for DB2, such as monitoring the database and setting up monitoring agents to monitor file system growth. This chapter describes the components of the GUI.

Note The GUI uses VxDBA to perform many actions. For information on setting up VxDBA and the GUI in a high availability environment, see "Setting Up VxDBA in an HA Environment" on page 327.

The GUI runs in a client-server environment. The server is located on a host that runs VERITAS Storage Foundation *for DB2*. The client can run on any Solaris, Windows NT, Windows 98, Windows 2000, Windows Me, and Windows XP machine that supports the Java Runtime Environment.

Within the GUI, you can perform tasks from the main menu bar or you can right click an object on the navigational (left) side of the screen, as seen in the following graphic:



There are multiple sections within the GUI. The following list describes the sections shown in the graphic, which depicts the Main Window:

- **Menu Bar** - allows you to perform various VxDBA operations. The options in the Menu Bar will vary according to the object in the object tree that you have selected.

Note To access online help from the Menu Bar, click **Help** > **Contents**.

- **Tool Bar** - provides shortcuts to various operations available in the Menu Bar. The Tool Bar is icon-based and dynamically changes when you select something from the Object Tree. When you use your mouse to point at an icon, a description of the icon appears.
- **Information Tabs** - allow you to view different information about the same object in the Object Tree. For example, if you are viewing details about a database, you can click a different tab to view different information about that database.
- **Object Tree** - is a dynamic hierarchical display of VERITAS Storage Foundation *for DB2*, VERITAS Volume Manager, and VERITAS File System objects and other objects on the system.
- **Status Fields** - indicate the status of the object you are viewing. To change your view, click one of the Information Tabs at the top of the window.
- **GUI Status Area** - displays GUI status, which is provided through VERITAS Enterprise Administrator. See the VERITAS Enterprise Administrator documentation for more information.

Within the GUI, options that are not available are displayed as grayed-out and are not clickable. Also, you can point to an object on the screen and a description of the object is displayed in a pop-up field.

The following table describes terms associated with the use of the mouse:

Term	Definition
Click	Press and release the mouse button.
Double-click	Click the mouse button twice (quickly).
Right-click	Press and release the right mouse button.
Press and Hold	Press and continue to hold down the mouse button.
Point	Move the tip of the pointer onto an item on the screen.

Term	Definition
Select	Click the mouse button while the pointer is directly over the item to be selected.
Drag	Slide the mouse while pressing a mouse button.

Overview of GUI Functions

The GUI allows you to perform many storage management tasks for DB2. This section offers an overview of those tasks. The tasks that you can perform dynamically from the main menu bar and pop-up menu depend on what is highlighted on the object tree. For example, if you have the DB2 instance folder highlighted in the tree, you can open an instance from the DB2 menu.

Note Some operations require that you must be logged in as `root`. Please read all "Prerequisites" and "Usage Notes" before starting a procedure.

GUI functionality includes:

- Instance operations

  You can start or stop a DB2 instance through the GUI.

- Storage Checkpoint Management

  You can create and roll back to Storage Checkpoints. You can also mount, unmount, and remove Storage Checkpoints. Storage Checkpoints can be used to clone the database.

- Database FlashSnap

  Database FlashSnap functionality is available through the GUI. With Database FlashSnap, you can create online or offline snapshots of your database, which can be used as backups. You can also use these snapshots to recover your database if it becomes corrupt.

- System configuration maintenance

  You can convert containers to or from Quick I/O files and you can view topology and statistic information. You can also check and save your system configuration.

- Monitoring Agent administration

  You can use the GUI to set up, configure, start, and stop a monitoring agent, which can be used to send alarm notifications and automatically grow file systems when space usage exceeds user-defined thresholds.

- Rescan System Information

  You can rescan, or refresh, your system information, such as DB2 instances and tablespaces. You can also determine the rescan intervals so that automatic rescans happen as regularly as you determine.

Prerequisites

- Before running the GUI, you must have appropriate permission to access the GUI. You must make permission changes to allow database administrators to access these tools. The default settings for the `/opt/VRTSdb2ed` directory at installation time allows only superuser (`root`) access to the directory. To make the appropriate permission changes, you will need to use the `vxdb2edusr` utility. See the steps for granting administrative permissions in the section “Starting VxDBA” on page 295 for more information.

VERITAS Enterprise Administrator Service

To use the VERITAS Storage Foundation *for DB2* GUI, VERITAS Enterprise Administrator (VEA) Service must be running on the server. The VEA Service is started when you install the software. Occasionally, you may need to manually start or stop the service or add users to the VEA Service console registry.

Prerequisites

- Use the `/opt/VRTS/bin/vxdb2edusr` utility to create login names for anyone who needs to run the GUI. To run `/opt/VRTS/bin/vxdb2edusr`, you must have superuser (`root`) privileges. For more information, see "Adding Users to the VERITAS Enterprise Administrator Service Console Registry" on page 227.
- You must have superuser (`root`) privileges to execute the `vxsvc` command.

Note You must update the database using the `db2ed_update` command at least once on each database before you can manage the database(s) through the GUI.

Adding Users to the VERITAS Enterprise Administrator Service Console Registry

You may want to add users to the VEA Service console registry to allow access to the interface to users other than `root`. You also have the option to give database administrators `root` privileges.

Having `root` privileges means that you can access the volume, disk, and file system objects in the system.

▼ **To add users other than root to the VERITAS Enterprise Administrator Service console registry**

1. Make sure that the optional GUI package was installed.

```
# pkginfo -l VRTSd2gui

 PKGINST:  VRTSd2gui
    NAME:  VERITAS Storage Foundation Graphical User Interface for
            DB2
CATEGORY:  application
    ARCH:  Sparc
 VERSION:  4.1
  VENDOR:  VERITAS Software
    DESC:  VERITAS Storage Foundation Graphical User Interface for
            DB2
```

```
     PSTAMP:  test.020322155315
   INSTDATE:  Dec 03 2004 15:55
    HOTLINE:  1-800-342-0652
      EMAIL:  support@veritas.com
     STATUS:  completely installed
      FILES:     29 installed pathnames
                  8 shared pathnames
                 13 directories
                  4 executables
              13514 blocks used (approx)
PKGINST: VRTSd2gui
```

2. To give `root` privileges within the GUI to the database administrator, use the `vxdb2edusr` command as follows.

   ```
   # /opt/VRTS/bin/vxdb2edusr -a user [-A] [-f] -n user_name
   ```

   where:

   `-a user` adds a user to the registry.

   `-A` grants the user `root` access.

   `-f` allows the user to be a user other than the `/opt/VRTSdb2ed` owner.

   `-n` indicates the name of the user that will be added or removed.

   For example, to add a database administrator with the name "db2inst1" as a user with `root` privileges, enter the following:

   ```
   # /opt/VRTS/bin/vxdb2edusr -a user -A -f -n db2inst1
   ```

3. To add a user without `root` privileges, use the `vxdb2edusr` command as follows.

   ```
   # /opt/VRTS/bin/vxdb2edusr -a user -n user_name
   ```

   where `-a` adds a user to the registry.

   For example, to add "db2inst1" as a user, enter the following:

   ```
   # /opt/VRTS/bin/vxdb2edusr -a user -n db2inst1
   ```

4. To add a group to the console registry, use the `vxdb2edusr` command as follows.

   ```
   # /opt/VRTS/bin/vxdb2edusr -a group -n group_name
   ```

   where `-a` adds the user group to the registry.

   For example, to add "dba" as a group, enter the following:

   ```
   # /opt/VRTS/bin/vxdb2edusr -a group -n dba
   ```

Note Make sure you shutdown and then restart the VEA server to update the registration information.

Removing Users from the VERITAS Enterprise Administrator Service Console Registry

You may need to restrict access to the VEA Service console registry. You can remove users or user groups from the registry if they have been previously added.

Note You cannot remove `root` from the VEA Service console registry.

▼ **To remove users other than root from the VERITAS Enterprise Administrator Service console registry**

1. Make sure that the optional GUI package was installed.

```
  # pkginfo -l VRTSd2gui
 PKGINST: VRTSd2gui
    NAME:  VERITAS Storage Foundation Graphical User Interface for
            DB2
CATEGORY:  application
    ARCH:  Sparc
 VERSION:  4.1
  VENDOR:  VERITAS Software
    DESC:  VERITAS Storage Foundation Graphical User Interface for
            DB2
  PSTAMP:  test.020322155315
INSTDATE:  Dec 03 2004 15:55
 HOTLINE:  1-800-342-0652
   EMAIL:  support@veritas.com
```

```
 STATUS:  completely installed
  FILES:      29 installed pathnames
               8 shared pathnames
              13 directories
               4 executables
           13514 blocks used (approx)
PKGINST: VRTSd2gui
```

2. To remove a user, use the `vxdb2edusr` command as follows.

   ```
   # /opt/VRTS/bin/vxdb2edusr -r user -n user_name
   ```

   where `-r` removes the user from the registry.

   For example, to remove the user "db2inst1," enter the following:

   ```
   # /opt/VRTS/bin/vxdb2edusr -r user -n db2inst1
   ```

3. To remove a group, use the `vxdb2edusr` command as follows.

   ```
   # /opt/VRTS/bin/vxdb2edusr -r group -n group_name
   ```

   where `-r` removes the user group from the registry.

   For example, to remove the group "dba," enter the following:

   ```
   # /opt/VRTS/bin/vxdb2edusr -r group -n dba
   ```

Note Make sure you restart the VEA server to update the registration information.

Starting the VERITAS Enterprise Administrator Service

▼ To manually start VERITAS Enterprise Administrator Service

1. Verify the status of the VEA Service:

   ```
   # /opt/VRTS/bin/vxsvcctrl status
   Current state of server : NOT RUNNING
   ```

2. Start the VEA Service:

   ```
   # /opt/VRTS/bin/vxsvcctrl start
   Initializing Storage Foundation Provider 4.1 for DB2
   ```

3. Again verify the status of the VEA Service:

```
# /opt/VRTS/bin/vxsvcctrl status
Current state of server : RUNNING
```

Note If the DB2 databases do not show up in the GUI after starting or restarting the VEA service, you may need to run the `db2ed_update` command. The databases should be displayed after a few seconds.

Shutting Down the VERITAS Enterprise Administrator Service

Although the VEA Service should remain running, you may need to shut it down manually.

▼ **To manually shut down VERITAS Enterprise Administrator Service**

1. Verify the status of the VEA Service:

```
# /opt/VRTS/bin/vxsvcctrl status
Current state of server : RUNNING
```

2. Stop the VEA Service:

```
# /opt/VRTS/bin/vxsvcctrl stop
Successfully unloaded the Storage Foundation Provider 4.1
for DB2
```

3. Again verify the status of the VEA Service:

```
# /opt/VRTS/bin/vxsvcctrl status
Current state of server : NOT RUNNING
```

Opening and Closing the VERITAS Storage Foundation for DB2 GUI

You can run the GUI from a Windows or UNIX client machine. You must have the client software installed before you can use the GUI.

Prerequisites

- Use the `/opt/VRTS/bin/vxdb2edusr` utility to create login names for anyone (other than `root`) who needs to run the GUI. To run `/opt/VRTS/bin/vxdb2edusr`, you must have superuser (`root`) privileges.
- The `db2ed_update` command must be run at least once before you can manage a database through the GUI.

Opening the VERITAS Storage Foundation for DB2 GUI from a Windows Client

▼ **To start the VERITAS Storage Foundation for DB2 GUI from a Windows client**

1. Click on **Start**, then select **Programs** > **VERITAS** > **VERITAS Enterprise Administrator**.

2. In the Connection pop-up window, enter the host name for the server to which you are connecting and press the **Tab** key.

3. Enter your login name and password. Then click **OK**.

Note The VEA Service must be running on the server. If you need to start the VEA Service, see “Starting the VERITAS Enterprise Administrator Service” on page 230.

Opening the VERITAS Storage Foundation for DB2 GUI from a UNIX Client

▼ **To start the VERITAS Storage Foundation for DB2 GUI from a UNIX client**

1. From an open terminal window, type **`/opt/VRTSob/bin/vea`** and press Enter.

Note You can run this command only from `root`.

---

2. In the Connection pop-up window, enter the host name for the server to which you are connecting and press the **Tab** key.

3. Enter your login name and password. Then click **OK**.

Note VEA Service must be running on the server. If you need to start VEA Service, see "Starting the VERITAS Enterprise Administrator Service" on page 230.

Closing the VERITAS Storage Foundation for DB2 GUI

▼ **To close the VERITAS Storage Foundation for DB2 GUI**

1. From the menu bar, select **File** > **Exit**.

2. The GUI displays a message indicating that you will be disconnected from the host if you continue. Click **Yes** to continue or, click **No** to keep the GUI running.

Note Stopping the GUI is the same for a Windows or UNIX client.

Starting a DB2 Instance

You can start a DB2 instance from the GUI. You must know the UNIX user name and password of the instance owner.

The steps to restart a DB2 database instance are the same.

▼ To start a DB2 instance

1. Click the **DB2 Instances** icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to start the instance.

   ◆ From the menu bar, select **DB2** > **Start Instance**.

   or

   ◆ Right click the **DB2 Instances** icon to bring up a pop-up menu. Then click **Start Instance**.

   The Start Instance wizard screen is then displayed.

3. Enter the instance name and home directory. Verify your UNIX user name and enter your password. The UNIX user name is the same as the instance name.

4. Click **Start Instance**.

5. At the confirmation prompt, click **Yes** to confirm that you want to start the database.

   If the DB2 instance was successfully started, you will receive a confirmation message. Click **OK** to continue.

Creating a DB2 Snapshot Database

If you created a snapshot using the Database FlashSnap feature, you can create a DB2 snapshot database from the secondary host via the GUI. This is the equivalent of cloning the database from the primary host. You must know the UNIX user name and password of the instance owner and there must be an existing snapshot.

For more information on the Database FlashSnap feature within the GUI, see "Managing Snapshots with Database FlashSnap" on page 268.

Prerequisites

- Make sure you have enough space to create a clone database on your system.
- You must have an existing snapshot.

▼ To create a DB2 snapshot database

1. Click the DB2 instance in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to start the snapshot database.

   - From the menu bar, select **DB2 Instances** > `Instance Name` > **Create Snapshot Database**.

     or

   - Right click the DB2 instance to bring up a pop-up menu. Then click **Create Snapshot Database**.

   The Create Snapshot Database wizard screen is then displayed.

3. In the Authentication section, enter the password for the UNIX user name. The user name is a read-only field.

4. For the "Primary database information" section, enter the DB2 Instance and DB2 Database information.

5. Select whether you are creating the database or restarting the database by clicking on the appropriate radio button.

   Select "Create database" if you are creating a new snapshot database based on a snapshot. Select "Restart database" if you are restarting a snapshot database that has already been created.

6. For the Snapshot database information section, enter the New DB2 Database, Snapplan file, VxDBA volume name, Snapshot disk group name, and Relocate path in the appropriate fields.

   Note To start an on-host snapshot database, the Relocate path is required.To start an off-host snapshot database, the VxDBA volume name field is required. You can obtain this information by viewing the details after you create a snapshot. If you did not retain the information from the snapshot creation, you can use the `db2ed_vmchecksnap` command or use View Log via the GUI to retrieve the information. For more information, see “Creating and Working with Snapplans Using db2ed_vmchecksnap” on page 387.

7. Click **Start Database**.

8. At the confirmation prompt, click **Yes** to confirm that you want to start the database.

   If the snapshot database was successfully started, you will receive a confirmation message. Click **OK** to continue.

Restarting a DB2 Instance

You can restart a DB2 instance from the GUI. You must have a UNIX user name and password.

▼ **To restart a DB2 instance**

1. Click the actual DB2 instance in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to start the database.

   ◆ From the menu bar, select **DB2** > **Start Instance**.

   or

   ◆ Right click the instance to bring up a pop-up menu. Then click **Start Instance**.

   The Start Instance wizard screen is then displayed.

3. Verify the DB2 instance information, such as the instance name, home directory, and instance owner. Then, click **Next** to continue.

4. Enter your password, then click **Start Instance**.

5. At the confirmation prompt, click **Yes** to confirm that you want to start the database.

   If the DB2 instance was successfully restarted, you will receive a confirmation message. Click **OK** to continue.

Shutting Down a DB2 Instance

The GUI lets you shut down a DB2 instance. For example, you must shut down the database to perform a Storage Rollback of an entire database.

▼ **To shut down a DB2 instance**

1. Click the DB2 instance in the object tree. (You may need to expand the tree view to find the icon.)
2. Select one of the following methods to stop the instance.
   - From the menu bar, select **DB2** > **Shutdown Instance**.

     or
   - Right click the instance to bring up a pop-up menu. Then click **Shutdown Instance**.
3. Verify the DB2 instance information, such as the instance name, home directory, and instance owner. Then, click **Next** to continue.
4. Verify your UNIX user name, then enter your password in the **Password** field.
5. In the dialog box, select the type of shutdown you want to perform:
   - **Normal**

     Use this option to shut down the DB2 instance in normal situations. If you shut down using this option, no new instance connections are allowed. DB2 waits for all currently connected users to disconnect from the instance, and then closes and dismounts the database before shutting down the instance. The next instance start up does not require an instance recovery.
   - **Force**

     Use this option to forcibly shut down the DB2 database when there are existing connections to the database. When this operation is selected, no new database connections are allowed. DB2 immediately terminates all sessions and shuts down the database. The next database startup may require an instance recovery.
6. Click **Shutdown instance** at the bottom of the screen.
7. At the confirmation prompt, click **Yes** to confirm that you want to shut down the instance.

   If the instance was successfully shut down, you will receive a confirmation message. Click **OK** to continue.

Creating Clone Database

You can create a Clone Database using Storage Checkpoints or Volume Manager FlashSnap. Cloning the database helps you perform operations without affecting your production database.

Using Storage Checkpoint

▼ **To create a clone database using a Storage Checkpoint**

1. In the object tree, click a specific Storage Checkpoint in the Storage Checkpoints folder. You may need to expand the tree view to find the Storage Checkpoint.

2. Select one of the following methods to create a clone database:

   - Click a Storage Checkpoint and click **Create Clone Database** in the menu bar.

     or

   - Right click the Storage Checkpoint object to bring up a pop-up menu. Then, click **Create Clone Database**.

   The Create Clone Database wizard is displayed. Click **Next**.

3. The **Create Clone Database** screen is displayed with read-only information about the current database. Please review the information.

   Click **Next**.

4. The **New Database Information** dialog is displayed. Enter the target information. This information will be the basis for your duplicated database. You need to enter or select the following:

   - Instance name
   - Password
   - Database name
   - Mount point
   - Redo log directory (Optional. If you do not specify any options, minimum recovery will be performed.)

   Check Restart Database if you want to restart a Storage Checkpoint Clone Database that is already created. Otherwise, you are creating a new Storage Checkpoint Clone Database.

Note The mount point in the target information is not required for restarting a Storage Checkpoint Clone Database.

5. Enter the source information. This is the information from your original database, which is the one that the Storage Checkpoint represents. You need to enter or verify the following:

   - Instance name (This field is read-only.)
   - Database name (This field is read-only.)
   - Database log directory (By default, this field is populated. Modify it *only* if the information is incorrect.)

6. Click **Create Clone Database**.

   If the database was successfully cloned, you will receive a confirmation message. Click **OK** to continue.

Using Volume Manager FlashSnap

Creating a clone database using Volume Manager FlashSnap goes through the following dialogs:

1. Create Snapplan

2. Validate/Modify Snapplan

3. Create Snapshot

4. Startup Clone Database

▼ To create a clone database using Volume Manager FlashSnap

1. In the object tree, click a DB2 database.

2. . Select one of the following methods to create a clone database:

   - From the menu bar, click a DB2 database, **DB2** > **Create Clone Database**.

     or

   - Right click the DB2 database object to bring up a pop-up menu. Then, click **Create Clone Database**.

   If you choose **New Snapplan**, the **Create Snapplan** dialog is displayed. For details, see "Creating a Clone Database with a new snapplan" on page 241.

If you choose **Existing Snapplan**, you are required to select a snapplan from the drop-down list, see "Creating a Clone Database with an Existing Snapplan" on page 244.

Creating a Clone Database with a new snapplan

▼ **To create a Clone Database with a new snapplan**

1. Confirm the Database Name. This is a read-only field.
2. Enter the following values:
    - Full path of the snapplan file
    - Name of the secondary host
    - Snapshot plex tag
3. Click **Next** to continue. The **Validate/Modify Snapplan** screen is displayed with default values set.
4. If needed, modify any incorrect settings. Then, click **Next** to validate the snapplan.

Note For more information regarding the snapplan parameters, see "Creating and Working with Snapplans Using db2ed_vmchecksnap" on page 387.

If the snapplan was successfully validated, you will receive a confirmation message.

5. To see the snapplan details, click the **Show details** checkbox. The snapplan details are displayed in the pop-up window.

   When you have finished reviewing them, click **OK**. The following example is a sample of the snapplan details that you should see:

```
SNAPSHOT_MODE is online_snapshot

PRIMARY_HOST is host1

SECONDARY_HOST is host2

The version of PRIMARY_DG-DB2dg is 120.

SNAPSHOT_DG is SNAP_DB2dg

SNAPSHOT_PLAN_FOR is database

Examining DB2 volume and disk layout for snapshot.

Volume testvol01 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol01 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol01 is on DB2dg03.
```

```
Volume testvol02 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol02 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol02 is on DB2dg03.

Volume testvol03 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol03 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol03 is on DB2dg03.

Volume testvol04 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol04 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol04 is on DB2dg03.

Volume testvol05 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol05 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol05 is on DB2dg03.

Volume udb_home on DB2dg is ready for snapshot.
Original plex and DCO log for udb_home is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for udb_home is on DB2dg03.

SNAP_DB2dg for snapshot will include: DB2dg03

ALLOW_REVERSE_RESYNC is yes

The snapplan sp1 has been created.
```

6. In the **Create Snapshot** dialog, verify the snapplan location. (This is a read-only field.)

7. If you need to force the snapshot creation, click the **Force snapshot creation** checkbox.

Note **Force snapshot creation** can be used after a snapshot operation has failed and the problem was fixed without using VERITAS Storage Foundation *for DB2* commands. (That is, the volumes were synchronized without using VERITAS Storage Foundation *for DB2* commands.) In this situation, the status of the snapplan will appear as unavailable for creating a snapshot. The **Force snapshot creation** option ignores the unavailable status, checks for the availability of volumes, and creates the snapshot after the volumes pass the availability check.

---

8. Click **Next** to continue.

9. If the snapshot creation was successful, you will receive a confirmation message.

10. Click the **Show details** checkbox to view the actions taken. The following example is a sample of the snapplan details that you should see:

```
db2ed_vmsnap started at 2004-06-08 11:41:39
DB20000I  The SET WRITE command completed successfully.
DB20000I  The SET WRITE command completed successfully.
```

```
A snapshot of DB2DATABASE UDB is in DG SNAP_DB2dg.
Snapplan sp1 is used for the snapshot.

If -r <relocate_path> is used in db2ed_vmclonedb,
make sure <relocate_path> is created and owned by DB2 Instance
Owner.
Otherwise, the following mount points need to be
created and owned by DB2 Instance Owner:

/db2/testvol01.
/db2/testvol02.
/db2/testvol03.
/db2/testvol04.
/db2/testvol05.
/db2/udb_home.

db2ed_vmsnap ended at 2004-06-08 11:42:14
```

11. Click **OK** to continue. You can now use the snapshot as a database backup.

Note You will need this information when you start your snapshot database.

12. The **Startup Snapshot Database** is displayed. In the **Authentication** section, enter the instance owner name and password for the UNIX user name. The user name is a read-only field.

Caution Make sure that the VEA service is running on the secondary host. Also, make sure that the database owner of the secondary host is registered on the VEA Service Console Registry. See "Adding Users to the VERITAS Enterprise Administrator Service Console Registry" on page 227 and "Starting the VERITAS Enterprise Administrator Service" on page 230 for further details.

13. For the "Primary database information" section, verify the DB2 Instance and DB2 Database information.

14. Select whether you are creating the database or restarting the database by clicking the appropriate radio button.

    Select "Create database" if you are creating a new snapshot database based on a snapshot. Select "Restart database" if you are restarting a snapshot database that has already been created.

15. For the Snapshot database information section, enter the New DB2 Database. Verify the Snapplan file, VxDBA volume name, Snapshot disk group name, and Relocate path in the appropriate fields.

    Note To start an on-host snapshot database, the Relocate path is required.To start an off-host snapshot database, the VxDBA volume name field is required. You can obtain this information by viewing the details after you create a snapshot. If you did not retain the information from the snapshot creation, you can use the `db2ed_vmchecksnap` command or use View Log via the GUI to retrieve the information. For more information, see "Creating and Working with Snapplans Using db2ed_vmchecksnap" on page 387.

16. Click **Create Clone Database**.

    If the snapshot database was successfully started, you will receive a confirmation message. Click **OK** to continue.

Creating a Clone Database with an Existing Snapplan

▼ To create a Clone Database with an existing Snapplan

Select **Existing Snapplan**. You are required to select a snapplan from the drop-down list. The snapplan displays its status, for example, `sp00 [init_db] /ora_home/snapplan/sp00`, where INIT_DB is the status of the snapplan.

- If you select a snapplan with INIT_DB status, the **Validate/Modify Snapplan** dialog is displayed. Follow the steps from step 4 on page 241 in the "Creating a Clone Database with a new snapplan" section.
- If you select a snapplan with INIT_FULL status, the **Create Snapshot** dialog is displayed. Follow the steps from step 6 on page 242 in the "Creating a Clone Database with a new snapplan".
- If you select a snapplan with SNAPSHOT_END status displays the **Startup Snapshot Database** dialog is displayed. Follow the steps from step 12 on page 243 in the "Creating a Clone Database with a new snapplan" section.

Removing a Clone Database

In releases prior to 4.1, for removing a cloned database you were required to go through separate menus to shut down and unmount clone databases. VERITAS Storage Foundation *for* DB2 4.1, provides the Remove Clone Database wizard, that can be accessed from the database instance node. The Remove Clone Database wizard helps in shutting down and unmounting the clone database. It also removes the Storage Checkpoint or resynchronizes the snapshot, if needed.

Removing a VM FlashSnap Clone Database

Removing a VM FlashSnap clone database unmounts the FlashSnap database. If needed, it also resynchronizes the snapshot.

1. In the object tree, click a clone database that was created using VM FlashSnap. You may need to expand the object tree to find the clone database.

2. Select one of the following methods to remove a clone database.

   From the menu bar, click DB2 > Remove Clone Database.

   or

   Right click the clone database object to bring up a pop-up menu. Then, click Remove Clone Database.

3. The **Remove Clone Database** dialog is displayed. In the Authentication section, verify the UNIX user name and enter the password.

   In the **Snapshot Database Information** section, verify the Snapplan File name and the Relocate path.

   You may check the **Resync Snapshot** option to resynchronize the snapshot.

   In the **Primary Database Information** section, verify the following information:

   - Primary Host Name
   - DB2 Database Name
   - Instance Owner Name

   Note You are required to enter the Password only if you check the **Resync Snapshot** option.

4. Click the Remove Clone Database button.

5. In the **Confirmation** dialog, click Yes to proceed with removing the clone database.

Removing a Storage Checkpoint Clone Database

Removing a Storage Checkpoint clone database allows you to unmount the clone database. It also gives you the option to remove the checkpoint.

1. In the object tree, click a clone database that was created using a Storage Checkpoint. You may need to expand the object tree to find the clone database.

2. Select one of the following methods to remove the clone database.

   From the menu bar, click **DB2** > **Remove Clone Database**.

   or

   Right click the database object to bring up a pop-up menu. Then, click **Remove Clone Database**.

3. The **Remove Clone Database** screen is displayed. Verify the read-only information. Click **Next** to continue.

4. In the **Storage Checkpoint Clone Database** section, verify the Instance Name and enter the Password. Also, verify the Database Name and Mount Point.

   In the **Primary Database Information** section, verify the Instance Name and Database Name.

5. Check the **Remove Storage Checkpoint** option if you want to remove the Storage Checkpoint along with removing the cloned database.

6. Click the **Remove Clone Database** button.

7. In the **Confirmation** dialog, click Yes to proceed.

Using the Monitoring Agent

You can use a Monitoring Agent to manage and monitor VxFS file systems, DB2 tablespaces, and container space usage. The Monitoring Agent feature is available for DB2 EE instances, but not for DB2 EEE instances.

The Monitoring Agent monitors the file system space, and when the space usage reaches a configured threshold value, a predefined action script grows the file system automatically.

The agent can be enabled at boot-time. Each file system monitored has three settings that the Monitoring Agent needs to know about:

- *Warning Threshold* is a percent value (% of file system used) that determines when the agent begins warning the administrator of space shortage
- *Grow Threshold* is a percent value (% of file system used) that determines when the agent is to attempt to grow the file system (when space usage is at a critical level)
- *Amount* is either a percentage or a value in megabytes by which to grow file systems when the Grow Threshold is reached or exceeded

The VxDBA Monitoring Agent operations are driven from the following files:

- `/opt/VRTSd2bed/lib/dbed_mon_config.base`
- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODEnum0000/ \`
  `db2ed_mon_config.DB2.$DB2INSTANCE.$DB2DATABASE.NODEnum0000`
- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODEnum0000/ \`
  `db2ed_mon_fslist.DB2.$DB2INSTANCE.$DB2DATABASE.NODEnum0000`
- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODEnum0000/ \`
  `db2ed_mon_db2list.DB2.$DB2INSTANCE.$DB2DATABASE.NODEnum0000`
- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODEnum0000/include`

Understanding the Monitoring Agent Files

The `/opt/VRTSdb2ed/lib/db2ed_mon_config.base` file contains the site-level configuration settings for monitoring all file systems and databases recognized. This configuration file specifies how often to check for file system and database configuration changes, how often to check the file space usage, where space usage information gets logged, and the thresholds for warning and automatically growing the file system.

For example, if you are monitoring a database named `PROD` under the instance `inst01`, the database-specific file would be `/etc/vx/vxdba/DB2.inst01.PROD/NODE0000/db2ed_mon_config.DB2.inst0.PROD.NODE0000`. This is the first file opened when the agent is started and contains the default settings for monitoring file systems at the database level. The Monitoring Agent cannot start without this file. Modify this configuration file if you want to change the preconfigured settings carried over from the `db2ed_mon_config.base` file to maintain a different set of settings at the database level.

The files `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODEnum0000/db2ed_mon_fslist.DB2.$DB2INSTANCE.$DB2DATABASE.NODEnum0000` and `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODEnum0000/db2ed_mon_db2list.DB2.$DB2INSTANCE.$DB2DATABASE.NODEnum0000` are created and are used for restarting the Monitoring Agent. These files specify the status of the database. The files also specify the space monitoring and alarm information for each file system, tablespace, and container. You can edit these files manually to change settings, and then restart the Monitoring Agent.

The Monitoring Agent uses the `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODEnum0000/include` file to check that all files are up-to-date and are being monitored. This file is created by VERITAS Storage Foundation for DB2 and should not be edited.

Occasionally, Monitoring Agents ignore Storage Checkpoints. This happens when a Storage Checkpoint is not owned by the current DB2 instance. These Storage Checkpoints will not be used to calculate thresholds and potential removal candidates. Storage Checkpoints that are not considered part of the current instance's data set are logged as such in the file `/var/log/db2ed_mon.prune_ckpt_log.DB2.$DB2INSTANCE.$DB2DATABASE.NODEnum0000` when the Monitoring Agent is looking for potential removal candidates. A Storage Checkpoint must have an entry in the `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/checkpoint_dir` directory before it is considered owned by the database. This is done automatically by the provided `VxDBA`(1M) and `db2ed_ckptcreate`(1M) utilities and ensures that, if multiple databases share the same file system(s), the policy for one database does not affect another.

Starting a Monitoring Agent

Prerequisites

- The `db2ed_update` command must be run at least once before you can use the monitoring agent utility.
- Make sure you are logged on as `root` to perform Monitoring Agent operations.
- Make sure you are running a DB2 EE instance.

▼ To start a Monitoring Agent

1. Click the **Monitoring Agent** icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to start the Monitoring Agent.

   ◆ From the menu bar, select **Monitoring Agent** > **Start Agent**.

   or

   ◆ Right click the Monitoring Agent to bring up a pop-up menu. Then click **Start Agent**.

   You will see a message prompting you for your `root` password.

3. Enter your `root` password. Then click **OK**.

   If the Monitoring Agent was successfully started, you will receive a confirmation message. Click **OK** to continue.

Enable / Disable the Monitoring Agent at Boot-Time

4. Optionally, if you would like to set up the Monitoring Agent to become enabled at boot-time, right click the Monitoring Agent in the object tree and select **Enabled at Boot Time**.

   or

   If you would like to set up the Monitoring Agent to become disabled at boot-time, right click the Monitoring Agent in the object tree and select **Disabled at Boot Time**.

5. Enter your `root` password. Then click **OK**.

   If the Monitoring Agent was successfully changed to be either enabled or disabled at boot time, you will receive a confirmation message. Click **OK** to continue.

   You will receive a confirmation message that the agent will either be enabled or disabled when you boot the system.

6. Click **OK** to continue.

Note To change the default Monitoring Agent values, see “Viewing or Changing Monitoring Agent Values” on page 250.

Stopping a Monitoring Agent

Prerequisites

- The `db2ed_update` command must be run at least once before you can use the monitoring agent utility.
- Make sure you are logged on as `root` to perform Monitoring Agent operations.
- Make sure you are running a DB2 EE instance.

▼ To stop a monitoring agent

1. Click the **Monitoring Agent** icon in the object tree. (You may need to expand the tree view to find the icon.)
2. Select one of the following methods to stop the Monitoring Agent.
   - From the menu bar, select **Monitoring Agent** > **Stop Agent**.

     or
   - Right click the Monitoring Agent to bring up a pop-up menu. Then click **Stop Agent**.

   You will then see a message prompting you for your `root` password.
3. Enter your `root` password. Then click **OK**.

   If the Monitoring Agent was successfully stopped, you will receive a confirmation message. Click **OK** to continue.

Viewing or Changing Monitoring Agent Values

You can view the settings of your active Monitoring Agent. You also have the option to change the default values of the Monitoring Agent, if needed.

Prerequisites

- The `db2ed_update` command must be run at least once before you can use the monitoring agent utility.
- Make sure you are logged on as `root` to perform Monitoring Agent operations.
- Make sure you are running a DB2 EE instance.

▼ To view the Monitoring Agent values

Click the **Monitoring Agent** icon in the object tree. (You may need to expand the tree view to find the icon.)

The values associated with the Monitoring Agent are displayed on the screen.

▼ To change the Monitoring Agent values

1. Click the **Monitoring Agent** icon in the object tree. (You may need to expand the tree view to find the icon.) The Monitoring Agent values appear on the right side of the screen.

2. On the right side of the screen, double-click the field you want to change and enter a new value. You may continue to double-click the fields you want to change and enter new values.

3. Click **Save** to save your changes.

Managing Storage and Version Checkpoints

A Storage Checkpoint is like an online backup of a database, including partitioned databases, that contains a point-in-time database image. Storage Checkpoints can later be used to restore the image of the entire database to any earlier state recorded by the Storage Checkpoints.

Note For DB2 you cannot recover a container or a tablespace.

Storage Checkpoints are used in a DB2 UDB EE environment. DB2 UDB EEE (SMP only) supports data partitioning across clusters of massively parallel computers. In a partitioned EEE environment, Storage Checkpoints that apply to multiple partitions are referred to as *Version Checkpoints*.

VERITAS Storage Foundation *for DB2* uses the VxDBA repository to determine the list of tablespaces, containers, and file systems for Storage Checkpoint creation and removal.

For more information about Storage Checkpoints, see "Understanding Storage Checkpoints and Storage Rollback" on page 122 and "Partitioned Databases and Version Checkpoints" on page 135.

Creating a Storage Checkpoint

For a DB2 EE instance, you can create an offline or online Storage Checkpoint for a DB2 database. To create an offline Storage Checkpoint, the instance can be either running or stopped and the database must be inactive. To create an online Storage Checkpoint, the instance must be running and the database can be either active or inactive, but it must be in archive log enabled mode.

Prerequisites

- Enable `ARCHIVELOG` mode before taking online Storage Checkpoints. See "Backing Up and Recovering the Database Using Storage Checkpoints" on page 136and"Using Storage Checkpoints and Storage Rollback for Backup and Restore" on page 122.

▼ To create a Storage Checkpoint

1. Click the **Storage Checkpoints** icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to create a Storage Checkpoint.

   - From the menu bar, select **Storage Checkpoints** > **Create Storage Checkpoint**.

     or

   - Right click the **Storage Checkpoints** icon to bring up a pop-up menu. Then, click **Create Storage Checkpoint**.

   The Create a Storage Checkpoint wizard is displayed.

3. Verify the database name. This is a read-only field.

4. Select one of the following options:

   - To create an online Storage Checkpoint, click the **Online** button.
   - To create an offline Storage Checkpoint, click the **Offline** button.

5. If you want to remove the Storage Checkpoint when the file system becomes full, click the **Remove This Storage Checkpoint** button.

   or

   If you want to retain the Storage Checkpoint when the file system becomes full, click the **Retain This Storage Checkpoint** button.

6. Click **Create** to continue.

7. At the prompt, click **Yes** to proceed with creating the Storage Checkpoint.

   If the Storage Checkpoint was successfully created, you will receive a confirmation message. To see details, click the **Show Details** checkbox. The details are displayed in a pop-up window.

   Click **OK** to continue.

Viewing Storage Checkpoint Details

▼ To view the details of a Storage Checkpoint

1. Click a specific **Storage Checkpoint** in the object tree. (You may need to expand the tree view to find the Storage Checkpoint.)

   The Storage Checkpoint information is displayed on the right side of the window.

2. To view file system quota information for the Storage Checkpoint, click the **File System Quota** tab at the top of the window, just above the Storage Checkpoint details.

Note The file system quota information is only available for disk layout version 6.

Mounting a Storage Checkpoint

You can mount and write to Storage Checkpoints just as you can do with any file system. See "Using Storage Checkpoints and Storage Rollback for Backup and Restore" on page 122 for more information.

▼ To mount a Storage Checkpoint

1. Click a specific Storage Checkpoint in the object tree. (You may need to expand the tree view to find the Storage Checkpoint.)

2. Select one of the following methods to mount the Storage Checkpoint.

   ◆ From the menu bar, select **Storage Checkpoint** > **Mount Storage Checkpoint**.

   or

   ◆ Right click the Storage Checkpoint you want to mount to bring up a pop-up menu. Then click **Mount Storage Checkpoint**.

   The Mount a Storage Checkpoint wizard then displays.

3. Verify that you are mounting the correct Storage Checkpoint and click **Next** to continue. If you selected the wrong Storage Checkpoint, click **Cancel**. The information on this screen is read-only.

4. On the second screen, enter the mount point (absolute path) where the Storage Checkpoint should be mounted.

Note The directory containing the mount point must be writable by the database administrator group. You should have created this group during installation. If not, create the group before mounting the Storage Checkpoint.

5. To mount the Storage Checkpoint as read-only, click **Read Only**.

   or

   To mount the Storage Checkpoint as read-write, click **Read/Write**. This will allow you to make changes to the Storage Checkpoint.

Note When you select the **Read/Write** option, the GUI creates an identical Storage Checkpoint with the same name plus a `wr001` suffix, where *001* is a sequential number. The GUI mounts the new Storage Checkpoint and leaves the original Storage Checkpoint unmounted. This allows you to roll back to the original Storage Checkpoint.

6. Click **Mount** to mount the Storage Checkpoint.

7. At the prompt, click **Yes** to proceed with mounting the Storage Checkpoint.

Unmounting a Storage Checkpoint

▼ To unmount a Storage Checkpoint

1. Click a specific Storage Checkpoint in the object tree. (You may need to expand the tree view to find the Storage Checkpoint.)

Note If you want to unmount a Storage Checkpoint that was originally mounted with the **Read/Write** option, you should unmount the new Storage Checkpoint that was automatically created by the GUI, which is the Storage Checkpoint that contains the `wr001` suffix, where *001* is a sequential number, at the end of the name.

2. Select one of the following methods to unmount the Storage Checkpoint.

   ◆ From the menu bar, select **Storage Checkpoint** > **Unmount a Storage Checkpoint**.

     or

   ◆ Right click the Storage Checkpoint you want to unmount to bring up a pop-up menu. Then click **Unmount Storage Checkpoint**.

3. Verify that you are unmounting the correct Storage Checkpoint and click **Unmount** to continue. If you selected the wrong Storage Checkpoint, click **Cancel**. The information on this screen is read-only.

4. At the prompt, click **Yes** to proceed with unmounting the Storage Checkpoint.

5. A confirmation dialog is displayed. Click **OK** to continue.

Removing a Storage Checkpoint

Occasionally, you may need to manually remove Storage Checkpoints that are no longer needed. For example, you can remove a Storage Checkpoint on a file system to free up needed space.

Prerequisites

- ◆ Before you can remove a mounted Storage Checkpoint, you must first unmount it.

▼ To remove a Storage Checkpoint

1. Click a specific Storage Checkpoint in the object tree. (You may need to expand the tree view to find the Storage Checkpoint.)

2. Select one of the following methods to remove the Storage Checkpoint.

   - ◆ From the menu bar, select **Storage Checkpoint** > **Remove Storage Checkpoint**.

     or

   - ◆ Right click the Storage Checkpoint you want to remove to bring up a pop-up menu. Then click **Remove Storage Checkpoint**.

3. At the prompt, click **Yes** to remove the Storage Checkpoint.

   If the Storage Checkpoint was successfully removed, you will receive a confirmation message. Click **OK** to continue.

Rolling Back to a Storage Checkpoint

You can roll back the entire database to a Storage Checkpoint.

Note The GUI does not automatically roll back any logs associated with a Storage Checkpoint. See "Guidelines for DB2 Recovery" on page 148 for information on database recovery.

Note You must be the Database Administrator to perform Storage Rollback operations. For an online Storage Rollback, the instance should be running, but the database must be in archive mode (either active or inactive). For an offline Storage Rollback, the instance can be running or inactive, but the database must be inactive. See "Backing Up and Recovering the Database Using Storage Checkpoints" on page 136 and"Guidelines for DB2 Recovery" on page 148 for more information.

Storage Checkpoints can only be used to roll back files that are damaged due to a software error or a human error (for example, accidental deletion of a table). Because Storage Checkpoints reside on the same physical disks as the primary file system, when a file is corrupted due to a media failure, the file on the Storage Checkpoints will not be available either. In this case, you need to restore files from a tape backup.

After the files are rolled back, you may need to follow the recovery procedure described in your DB2 manuals to recover the database before the database can be used.

Rolling Back the Database to a Storage Checkpoint

Rolling back the entire database rolls back all the containers used by the database to a Storage Checkpoint.

Note While the Storage Rollback process is running, it creates a temporary file, `/`*`filesystem`*`/.VRTSstrb.lock`, in each file system. Do not remove these temporary lock files.

▼ **To rollback the database to a Storage Checkpoint**

1. Make the DB2 database inactive by either shutting down the instance or disabling all user connections.

2. Click the a specific Storage Checkpoint in the object tree. (You may need to expand the tree view to find the Storage Checkpoint.)

3. Select one of the following methods to rollback to the selected Storage Checkpoint.

   - From the menu bar, select **Storage Checkpoint** > **Roll Back Storage Checkpoint**.

     or

   - Right click the Storage Checkpoint to which you want to rollback to bring up a pop-up menu. Then, click **Roll Back Storage Checkpoint**.

   The Rollback a Storage Checkpoint wizard is displayed.

4. Verify that you are rolling back to the correct Storage Checkpoint and click **Next** to continue. If you selected the wrong Storage Checkpoint, click **Cancel**. The information on this screen is read-only.

5. On the second screen, use the drop-down menu to select the appropriate buffer size in the Rollback Buffer Size field. The default buffer size is 128K.

Note The buffer size configured for reads and writes when performing a Storage Rollback can affect performance. Vary the size to determine the optimal setting for your system.

6. Use the drop-down menu to select the appropriate number of threads in the Number of Threads field. The default number of threads is four.

Note Depending on the number of CPUs available on your system and the type of volume on which the file system is located, this default setting may specify too few or too many threads.

7. Click the **Roll back a database** button to indicate that you are rolling back the entire database to the Storage Checkpoint.

8. Click **Next** to continue. Click **Roll Back** to continue.

   If the Storage Rollback was successful, a confirmation message is displayed.

9. To see the Storage Rollback details, click the **Show details** checkbox. The details are displayed in the pop-up window.

   When you have finished viewing the details, click **OK**.

10. Click **Yes** to roll back the tablespace or tablespaces. Perform any necessary DB2 recovery. See "Guidelines for DB2 Recovery" on page 148 for more information. (You cannot recover your database through the GUI.)

11. Activate the database again.

Using a Storage Checkpoint Policy for Space Management

A Storage Checkpoint policy establishes how many Storage Checkpoints you would like to keep in the event your file system becomes full. You can also determine what to do with old Storage Checkpoints when you run out of space.

The Storage Checkpoint Policy feature is available for DB2 EE instances, but it is not available for DB2 EEE instances.

Prerequisites

◆ You must enable the Monitoring Agent to use the Storage Checkpoint policy.

◆ You must be running a DB2 EE instance.

Creating a Storage Checkpoint Policy

▼ **To create a Storage Checkpoint policy**

1. Click the **Storage Checkpoints** icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to create a Storage Checkpoint policy.

   Right click the Storage Checkpoint icon to bring up a pop-up menu. Then, click **Policy**.

   The **Storage Checkpoint Policy** window is displayed.

3. Enter the maximum number of Storage Checkpoints to keep on the file system in the Maximum field.

4. To enable the policy, click the **Enable** button.

5. To remove old Storage Checkpoints if the maximum number is exceeded, click **Remove Old Storage Checkpoints**.

   or

   If you would like to keep old Storage Checkpoints after the maximum number is exceeded, click **Retain Old Storage Checkpoints**. (This option will prevent any new Storage Checkpoints from being created if the file system runs out of space.)

6. Click **Policy Update** to save your changes.

   If the Storage Checkpoint policy was successfully created, you will receive a confirmation message. Click **OK** to continue.

Note If your Monitoring Agent is disabled, you will receive warning stating that you must active the Monitoring Agent in order to use the Storage Checkpoint policy.

Disabling a Storage Checkpoint Policy

▼ **To disable a Storage Checkpoint policy**

1. Click the **Storage Checkpoints** icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to disable a Storage Checkpoint policy.

   Right click the Storage Checkpoint icon to bring up a pop-up menu. Then, click **Policy**.

   The **Storage Checkpoint Policy** window is displayed.

3. To disable the policy, click the **Disable** button.

4. Click **Policy Update** to save your changes.

   If the Storage Checkpoint policy was successfully disabled, you will receive a confirmation message. Click **OK** to continue.

Creating a Version Checkpoint

For a DB2 EEE instance, you can create an offline or online Version Checkpoint for a DB2 database. To create an offline Version Checkpoint, the instance can be either running or stopped and the database must be inactive. To create an online Storage Checkpoint, the instance must be running and the database can be either active or inactive, but must be in archive log enabled mode.

Note The GUI does not automatically roll back any logs associated with a Version Checkpoint. See "Guidelines for DB2 Recovery" on page 148 for information on database recovery.

Prerequisites

- Enable ARCHIVELOG mode before taking Version Checkpoints. See "Backing Up and Recovering the Database Using Storage Checkpoints" on page 136 and "Using Storage Checkpoints and Storage Rollback for Backup and Restore" on page 122.

▼ To create a Version Checkpoint

1. Click the **Version Checkpoints** icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to create a Storage Checkpoint.

   - From the menu bar, select **Version Checkpoints** > **Create Version Checkpoint**.

     or

   - Right click the **Version Checkpoints** icon to bring up a pop-up menu. Then, click **Create Version Checkpoint**.

   The Create a Version Checkpoint wizard is displayed.

3. Verify the database name. This is a read-only field.

4. Select one of the following options:

   - To create an online Version Checkpoint, click the **Online** button.
   - To create an offline Version Checkpoint, click the **Offline** button.

5. If you want to remove the Version Checkpoint when the file system becomes full, click the **Remove This Version Checkpoint** button.

   or

   If you want to retain the Version Checkpoint when the file system becomes full, click the **Retain This Version Checkpoint** button.

6. Click **Create** to continue.

7. At the prompt, click **Yes** to proceed with creating the Version Checkpoint.

   To see the details, click the **Show details** checkbox. The details are displayed in the pop-up window.

   When you have finished viewing the details, click **OK**.

8. If the Version Checkpoint was successfully created, you will receive a confirmation message. Click **OK** to continue.

Viewing Version Checkpoint Details

▼ To view the details of a Version Checkpoint

1. Click a specific Version Checkpoint in the object tree. (You may need to expand the tree view to find the Version Checkpoint.)

   The Version Checkpoint information is displayed on the right side of the window. You can view information from the Details and Checkpoint Attributes tables at the top of the window.

2. To view individual Storage Checkpoint details, click the specific **Storage Checkpoint**.

3. To view file system quota information for the Storage Checkpoint, click the **File System Quota** tab at the top of the window, just above the Storage Checkpoint details.

Note The file system quota information is only available for disk layout 6.

Mounting a Version Checkpoint

You can mount, access, and write to Version Checkpoints just as you can any file system. See "Using Storage Checkpoints and Storage Rollback for Backup and Restore" on page 122 for more information.

▼ To mount a Version Checkpoint

1. Click a specific Version Checkpoint in the object tree. (You may need to expand the tree view to find the Version Checkpoint.)

2. Select one of the following methods to mount the Version Checkpoint.

   - From the menu bar, select **Version Checkpoint** > **Mount Version Checkpoint**.

     or

   - Right click the Version Checkpoint you want to mount to bring up a pop-up menu. Then click **Mount Version Checkpoint**.

   The Mount a Version Checkpoint wizard then displays.

3. Verify that you are mounting the correct Version Checkpoint and click **Next** to continue. If you selected the wrong Version Checkpoint, click **Cancel**. The information on this screen is read-only.

4. On the second screen, enter the mount point (absolute path) where the Version Checkpoint should be mounted.

Note The directory containing the mount point must be writable by the database administrator group. You should have created this group during installation. If not, create the group before mounting the Version Checkpoint.

---

5. To mount the Version Checkpoint as read-only, click **Read Only**.

   or

   To mount the Version Checkpoint as read-write, click **Read/Write**. This will allow you to make changes to the Version Checkpoint.

Note When you select the **Read/Write** option, the GUI creates an identical Version Checkpoint with the same name plus a `wr001` suffix, where *001* is a sequential number. The GUI mounts the new Version Checkpoint and leaves the original Version Checkpoint unmounted. This allows you to roll back to the original Version Checkpoint.

---

6. Click **Mount** to mount the Version Checkpoint.

7. At the prompt, click **Yes** to proceed with mounting the Version Checkpoint.

8. To see the details, click the **Show details** checkbox. The details are displayed in the pop-up window.

   When you have finished viewing the details, click **OK**.

Unmounting a Version Checkpoint

▼ To unmount a Version Checkpoint

1. Click a specific Version Checkpoint in the object tree. (You may need to expand the tree view to find the Version Checkpoint.)

Note If you want to unmount a Version Checkpoint that was originally mounted with the **Read/Write** option, you should unmount the new Version Checkpoint that was automatically created by the GUI, which is the Version Checkpoint that contains the `wr001` suffix, where *001* is a sequential number, at the end of the name.

---

2. Select one of the following methods to unmount the Version Checkpoint.
   - From the menu bar, select **Version Checkpoint** > **Unmount Version Checkpoint**.

     or
   - Right click the Version Checkpoint you want to mount to bring up a pop-up menu. Then click **Unmount Version Checkpoint**.
3. Verify that you are unmounting the correct Version Checkpoint and click **Unmount** to continue. If you selected the wrong Version Checkpoint, click **Cancel**. The information on this screen is read-only.
4. At the prompt, click **Yes** to proceed with unmounting the Version Checkpoint.
5. To see the details, click the **Show details** checkbox. The details are displayed in the pop-up window.

   When you have finished viewing the details, click **OK**.

Removing a Version Checkpoint

Occasionally, you may need to manually remove Version Checkpoints that are no longer needed. For example, you can remove a Version Checkpoint on a file system to free up needed space.

Prerequisites

- Before you can remove a mounted Version Checkpoint, you must first unmount it.

▼ To remove a Version Checkpoint

1. Click a specific Version Checkpoint in the object tree. (You may need to expand the tree view to find the Version Checkpoint.)
2. Select one of the following methods to remove the Version Checkpoint.
   - From the menu bar, select **Version Checkpoint** > **Remove Version Checkpoint**.

     or
   - Right click the Version Checkpoint you want to remove to bring up a pop-up menu. Then click **Remove Version Checkpoint**.

3. At the prompt, click **Yes** to remove the Version Checkpoint.

   To see the details, click the **Show details** checkbox. The details are displayed in the pop-up window.

   When you have finished viewing the details, click **OK**.

4. If the Version Checkpoint was successfully removed, you will receive a confirmation message. Click **OK** to continue.

Rolling Back to a Version Checkpoint

You can roll back the entire database to a Version Checkpoint.

Note You must be the Database Administrator to perform Storage Rollback operations. For an online Storage Rollback, the instance should be running, but the database must be in archive mode (either active or inactive). For an offline Storage Rollback, the instance can be running or inactive, but the database must be inactive. See "Backing Up and Recovering the Database Using Storage Checkpoints" on page 136 and "Guidelines for DB2 Recovery" on page 148 for more information.

Version Checkpoints can only be used to roll back files that are damaged due to a software error or a human error (for example, accidental deletion of a table). Because Version Checkpoints reside on the same physical disks as the primary file system, when a file is corrupted due to a media failure, the file on the Version Checkpoints will not be available either. In this case, you need to restore files from a tape backup.

After the files are rolled back, you may need to follow the recovery procedure described in your DB2 manuals to recover the database before the database can be used.

Rolling Back the Database to a Version Checkpoint

Rolling back the entire database rolls back all the containers used by the database to a Version Checkpoint.

Note While the Storage Rollback process is running, it creates a temporary file, `/`*`filesystem`*`/.VRTSstrb.lock`, in each file system. Do not remove these temporary lock files.

▼ To rollback the database to a Version Checkpoint

1. Make the DB2 database inactive by either shutting down the instance or disabling all user connections.

2. Click a specific Version Checkpoint in the object tree. (You may need to expand the tree view to find the Version Checkpoint.)

3. Select one of the following methods to rollback to the selected Version Checkpoint.

   - From the menu bar, select **Version Checkpoint** > **Roll Back Version Checkpoint**.

     or

   - Right click the Version Checkpoint to which you want to rollback to bring up a pop-up menu. Then, click **Roll Back Version Checkpoint**.

   The Rollback a Version Checkpoint wizard is displayed.

4. Verify that you are rolling back to the correct Version Checkpoint and click **Next** to continue. If you selected the wrong Version Checkpoint, click **Cancel**. The information on this screen is read-only.

5. On the second screen, use the drop-down menu to select the appropriate buffer size in the Rollback Buffer Size field. The default buffer size is 128K.

Note The buffer size configured for reads and writes when performing a Storage Rollback can affect performance. Vary the size to determine the optimal setting for your system.

6. Use the drop-down menu to select the appropriate number of threads in the Number of Threads field. The default number of threads is four.

Note Depending on the number of CPUs available on your system and the type of volume on which the file system is located, this default setting may specify too few or too many threads.

7. Click the **Rollback** button to indicate that you are rolling back the entire database to the Version Checkpoint.

8. At the prompt, click **Yes** to continue rolling back the entire database.

   If the operation was successful, you will receive a confirmation message. Click **OK** to continue. You are then returned to the rollback window.

9. Perform any necessary DB2 recovery. See “Guidelines for DB2 Recovery” on page 148 for more information. (You cannot recover your database through the GUI.)

10. Activate the database again.

Managing Snapshots with Database FlashSnap

With VERITAS Database FlashSnap, you can create a point-in-time copy of a database for backup and off-host processing. For more information, see “Using Database FlashSnap for Backup and Off-Host Processing” on page 151.

From the GUI, you can create snapshots of your database using snapplans. You can also resynchronize snapshots to your database and your database to a snapshot. The following sections describe the Database FlashSnap features within the GUI.

Database FlashSnap functionality is not supported on duplicated (cloned) databases.

Note Database FlashSnap is available for DB2 EE instances only.

Prerequisites

- ◆ Before you can create a snapshot, a snapshot mirror of a volume must exist. For details, see “Creating a Snapshot Mirror of a Volume or Volume Set Used by the Database” on page 161.

Creating a Snapplan

A snapplan specifies snapshot scenarios for a DB2 (such as online, offline, and instant). The snapplan is used as a basis for creating a snapshot. You must either create or validate a snapplan before you can create the snapshot image.

Prerequisites

- ◆ You must be logged in as the DB2 database administrator.
- ◆ The disk group must be version 110 or higher.
- ◆ Persistent FastResync must be enabled.
- ◆ To set up your snapplan for online snapshots, the database must be in archive log mode.

Usage Notes

- ◆ Database FlashSnap functionality is not supported on cloned (duplicated) databases.
- ◆ The snapplan name is user-defined.
- ◆ Each entry in the snapplan is a line in *parameter=argument* format.
- ◆ See the `db2ed_vmsnapplan`(1M) and `db2ed_vmchecksnap`(1M) manual page for more information.

▼ To create a snapplan

1. Click the **Snapplans** icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to create a snapshot plan.

   - From the menu bar, select **Snapplans** > **Create Snapplan**.

     or

   - Right click the Snapshot Plans icon to bring up a pop-up menu. Then click **Create Snapplan**.

   The Create Snapplan wizard is then displayed.

3. Confirm the Database Name. This is a read-only field.

4. Enter the full path of the snapplan file.

5. Enter the name of the secondary host.

6. Enter the snapshot plex tag.

7. Click **Next** to continue.

   The Validate/Modify Snapplan screen is displayed with default values set.

8. If needed, modify any incorrect settings. Then, click **Validate** to ensure that the settings have been configured correctly.

Note For more information regarding the snapplan parameters, see "Creating and Working with Snapplans Using db2ed_vmchecksnap" on page 387.

9. At the confirmation prompt, click **Yes** to confirm that you want to validate the snapplan.

   If the snapplan was successfully validated, you will receive a confirmation message.

10. To see the snapplan details, click the **Show details** checkbox. The snapplan details are displayed in the pop-up window.

    When you have finished reviewing them, click **OK**. The following example is similar to what you should see:

```
SNAPSHOT_MODE is online_snapshot

PRIMARY_HOST is host1
```

```
SECONDARY_HOST is host2

The version of PRIMARY_DG-DB2dg is 110.

SNAPSHOT_DG is SNAP_DB2dg

SNAPSHOT_PLAN_FOR is database

Examining DB2 volume and disk layout for snapshot.

Volume testvol01 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol01 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol01 is on DB2dg03.

Volume testvol02 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol02 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol02 is on DB2dg03.

Volume testvol03 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol03 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol03 is on DB2dg03.

Volume testvol04 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol04 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol04 is on DB2dg03.

Volume testvol05 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol05 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol05 is on DB2dg03.

Volume udb_home on DB2dg is ready for snapshot.
Original plex and DCO log for udb_home is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for udb_home is on DB2dg03.

SNAP_DB2dg for snapshot will include: DB2dg03

ALLOW_REVERSE_RESYNC is yes

The snapplan sp1 has been created.
```

Validating or Modifying a Snapplan

Before creating a snapshot, it is necessary to validate your snapplan to make sure the information is correct and that the snapshot will succeed. Occasionally, you may also need to modify a snapplan. For example, you would need to modify your snapplan if you wanted to change the primary disk group that is included in the snapshot.

Prerequisites

- You must be logged in as the DB2 database administrator.
- A DCO log must be attached to the snapshot plex.

Usage Notes

- Each entry in the snapplan is a line in *parameter=argument* format.

▼ To validate or modify a snapplan

1. Click a specific snapplan in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to validate or modify a snapshot plan.

   - From the menu bar, select **Snapplan** > **Modify/Validate Snapplan**.

     or

   - Right click the snapplan to bring up a pop-up menu. Then click **Modify/Validate Snapplan**.

   The Modify/Validate Snapplan wizard is displayed.

3. If incorrect, enter the path and file name of the snapplan in the **Snapplan file** field. Then, click **Next** to continue.

4. If you are modifying your snapplan, enter any new parameters that should be updated. You are not required to do anything if no changes are required.

Note For more information regarding the snapplan parameters, see "Creating and Working with Snapplans Using db2ed_vmchecksnap" on page 387.

5. Click **Validate** to check that the settings have been configured correctly.

6. At the confirmation prompt, click **Yes** to confirm that you want to validate the snapplan.

   If the snapplan was successfully validated, you will receive a confirmation message.

7. To see the snapplan details, click the **Show details** checkbox. The snapplan details are displayed in the pop-up window.

   When you have finished reviewing them, click **OK**. The following example is similar to what you should see:

```
SNAPSHOT_MODE is online_snapshot

PRIMARY_HOST is host1

SECONDARY_HOST is host2

The version of PRIMARY_DG-DB2dg is 110.

SNAPSHOT_DG is SNAP_DB2dg

SNAPSHOT_PLAN_FOR is database

Examining DB2 volume and disk layout for snapshot.

Volume testvol01 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol01 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol01 is on DB2dg03.

Volume testvol02 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol02 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol02 is on DB2dg03.

Volume testvol03 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol03 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol03 is on DB2dg03.

Volume testvol04 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol04 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol04 is on DB2dg03.

Volume testvol05 on DB2dg is ready for snapshot.
Original plex and DCO log for testvol05 is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for testvol05 is on DB2dg03.

Volume udb_home on DB2dg is ready for snapshot.
Original plex and DCO log for udb_home is on DB2dg01 DB2dg02.
Snapshot plex and DCO log for udb_home is on DB2dg03.

SNAP_DB2dg for snapshot will include: DB2dg03

ALLOW_REVERSE_RESYNC is yes

The snapplan sp1 has been created.
```

Removing a Snapplan

You may need to remove a snapplan that is no longer necessary. However, you cannot create a snapshot if you do not have a snapplan.

Prerequisites

- ◆ You must be logged in as the DB2 database administrator.

▼ To remove a snapplan

1. Click the snapplan you want to remove in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to remove the snapplan.

   - ◆ From the menu bar, select **Snapplan** > **Remove Snapplan**.

     or

   - ◆ Right click the snapplan to bring up a pop-up menu. Then click **Remove Snapplan**.

     A pop-up window appears asking you to confirm whether you want to remove the snapplan.

3. At the prompt, click **Yes** to continue with removing the snapplan.

   If the snapplan was successfully removed, you will receive a confirmation message similar to the following:

   ```
   Successfully removed snapplan: sp1
   ```

4. Click the **Show details** checkbox to view the actions taken. You should see information similar to the following:

   ```
   The snapplan <filename> has been removed.
   ```

   Click **OK** to continue.

Creating a Snapshot

After having created and validated a snapplan, you can create a snapshot of your database. You can use the snapshot as a database backup or as a test database to perform operations without affecting your production database.

Prerequisites

- You must have a validated snapplan before creating a snapshot image.
- You must be logged in as the DB2 database administrator.

Usage Notes

- See the `db2ed_vmsnap`(1M) manual page for more information.
- If you are creating an offhost snapshot, perform these steps on the secondary host.

▼ To create a snapshot

1. Click a snapplan in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to create a snapshot.

   - From the menu bar, select **Snapplan** > **Create Snapshot**.

     or

   - Right click the snapplan to bring up a pop-up menu. Then click **Create Snapshot**.

   The Create Snapshot wizard is displayed.

3. Verify the snapplan location. (This is a read-only field.)

4. If you need to force the snapshot creation, click the **Force snapshot creation** checkbox.

   Note **Force snapshot creation** can be used after a snapshot operation has failed and the problem was fixed without using VERITAS Storage Foundation *for DB2* commands. (That is, the volumes were synchronized without using VERITAS Storage Foundation *for DB2* commands.) In this situation, the status of the snapplan will appear as unavailable for creating a snapshot. The **Force snapshot creation** option ignores the unavailable status, checks for the availability of volumes, and creates the snapshot after the volumes pass the availability check.

5. Click **Finish** to continue.

6. If the snapshot creation was successful, you will receive a confirmation message.

7. Click the **Show details** checkbox to view the actions taken. You should see information similar to the following:

   ```
   db2ed_vmsnap started at 2004-06-08 11:41:39
   DB20000I  The SET WRITE command completed successfully.
   DB20000I  The SET WRITE command completed successfully.
   A snapshot of DB2DATABASE UDB is in DG SNAP_DB2dg.
   Snapplan sp1 is used for the snapshot.

   If -r <relocate_path> is used in db2ed_vmclonedb,
   make sure <relocate_path> is created and owned by DB2 Instance
   Owner.
   Otherwise, the following mount points need to be
   created and owned by DB2 Instance Owner:

   /db2/testvol01.
   /db2/testvol02.
   /db2/testvol03.
   /db2/testvol04.
   /db2/testvol05.
   /db2/udb_home.

   db2ed_vmsnap ended at 2004-06-08 11:42:14
   ```

   Click **OK** to continue.

   You can now use the snapshot as a database backup.

Note You will need this information when you start your snapshot database.

Creating a Clone Database With a Snapshot

After having created a snapshot, you can use the snapshot to create a clone database. The cloned database can be used for decision-making and testing that cannot be done on your production database.

Prerequisites

- Make sure you have enough disk space to create a clone database on your system.
- You must have an existing snapshot.

▼ To create a clone database with a snapshot

1. Click the **DB2 instances** icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to start the snapshot database.

   - From the menu bar, select **DB2 instances** > **Create Snapshot Database**.

     or

   - Right click the **DB2 instances** icon to bring up a pop-up menu. Then click **Create Snapshot Database**.

   The Create Snapshot Database wizard screen is then displayed.

3. In the “Authentication” section, enter the Unix user name and password.

4. For the primary database, enter the DB2 Instance and DB2 Database information.

5. Select whether you are creating the database or restarting the database by clicking the appropriate radio button.

   Select **Startup database** if you are restarting a new snapshot database based on a snapshot. Select **Restart database** if you are restarting a snapshot database that has already been created.

6. For the snapshot database, enter the new DB2 Database, Snapplan file, VxDBA volume name, Snapshot disk group name, and Relocate path in the appropriate fields.

   Note To start an on-host snapshot database, the Relocate path is required. To start an off-host snapshot database, the VxDBA volume name field is required. You can obtain this information by viewing the details after you create a snapshot. If you did not retain the information from the snapshot creation, you can use the `db2ed_vmchecksnap` command or use Viewlog via the GUI to retrieve the information. For more information, see “Creating and Working with Snapplans Using db2ed_vmchecksnap” on page 387.

7. Click **Start Database**.

8. At the confirmation prompt, click **Yes** to confirm that you want to start the database.

   If the snapshot database was successfully started, you will receive a confirmation message. Click **OK** to continue.

Resynchronizing a Snapshot to a Database

Resynchronizing a snapshot to a database will refresh the snapshot so that it contains the most recent changes made to your production database.

Prerequisites

- You must be logged in as the DB2 database administrator.
- You must shut down the clone database and unmount the file systems.

Usage Notes

- See the `db2ed_vmsnap`(1M) manual page for more information.

▼ To resynchronize a snapshot to a database

1. Click a snapplan, located under the **Snapplans** icon, in the object tree. (You may need to expand the tree view to find the icon.)
2. Select one of the following methods to resynchronize the snapshot.
   - From the menu bar, select **Snapplan** > **Resync Snapshot**.

     or
   - Right click the snapplan to bring up a pop-up menu. Then click **Resync Snapshot**.
3. At the confirmation prompt, click **OK** to continue resynchronizing the snapshot to the database.
4. If the resynchronization was successful, you will receive a confirmation message. Click **OK** to continue.
5. Status information similar to the following example is displayed. Click **OK** when you have finished viewing the information.

   ```
   db2ed_vmsnap started at 2003-05-13 17:20:05
   The option resync has been completed.
   db2ed_vmsnap ended at 2003-05-13 17:20:41
   ```

   To see the details, click the Show details checkbox. The details are displayed in a pop-up window.

Resynchronizing a Database to a Snapshot

Resynchronizing your database to a snapshot, also known as reverse resynchronization, reverts your database to a snapshot. Use this option if your database becomes corrupted and you need to restore your database to a previous point-in-time.

Prerequisites

- You must be logged in as the DB2 primary database administrator.
- You must shut down the primary database and the clone database; also, unmount the file systems.

Usage Notes

- See the `db2ed_vmsnap`(1M) manual page for more information.

▼ To resynchronize a database to a snapshot

1. Click a snapplan, located under the **Snapplans** icon, in the object tree. (You may need to expand the tree view to find the icon.)
2. Select one of the following methods to resynchronize the database to a snapshot.
   - From the menu bar, select **Snapplan** > **Reverse Resync Snapshot**.

     or
   - Right click the snapplan to bring up a pop-up menu. Then click **Reverse Resync Snapshot**.

   The Reverse Resync wizard is displayed.
3. Verify the Database name and snapplan information. These are read-only fields.
4. Click the Begin button, then click **Finish** to start the reverse resynchronization process. To view details, click the **Show details** checkbox. The details are displayed in a pop-up window.
5. At the confirmation prompt, click **Yes** to continue.

   The Begin option performs the following actions:
   - Imports the disk group that was deported from the secondary host and joins it back to the original disk group.

- Mounts the snapshot volumes.
- Mounts the file systems that are configured for the primary database.
- Brings up the database snapshot image as the primary database.

Note The primary database must be offline to perform this action.

---

6. If the begin action was successful, you will receive a confirmation message. Click **Show details** to see the actions completed. When you are through, click **OK** to continue.

7. Again, click the snapplan on which you want to perform the reverse resynchronization.

8. Select one of the following methods to resynchronize the database to a snapshot.

   - From the menu bar, select **Snapplan** > **Reverse Resync Snapshot**.

     or

   - Right click the snapplan to bring up a pop-up menu. Then click **Reverse Resync Snapshot**.

   The Reverse Resync wizard is displayed.

9. Click the **Commit** button, then click **Finish** to commit the reverse resynchronization process.

10. At the confirmation prompt, click **Yes** to continue.

    The Commit option performs the following actions:

    - Commits the reverse resynchronization changes.
    - Resynchronizes the original volume from the data in the snapshot and then discards the content of the original database.

Note This action cannot be undone.

---

11. If the commit action was successful, you will receive a confirmation message. Click **Show details** to see the actions completed. When you are through, click **OK** to continue.

Aborting the Reverse Resychronization Operation

Occasionally, you may need to stop the reverse resynchronization process after you have begun. You can only abort the reverse resynchronization process after you have completed the Begin operation and before performing a Commit operation.

▼ To abort the reverse resynchronization operation

1. Click the snapplan for which the reverse resynchronization was begun.

2. Select one of the following methods to resynchronize the database to a snapshot.

   ◆ From the menu bar, select **Snapplan** > **Reverse Resync Snapshot**.

   or

   ◆ Right click the snapplan to bring up a pop-up menu. Then click **Reverse Resync Snapshot**.

   The Reverse Resync wizard is displayed.

3. Verify the Database name and snapplan information. These are read-only fields.

4. Click the Abort button, then click **Finish** to abort the reverse resynchronization process.

5. At the confirmation prompt, click **Yes** to continue.

   The Abort option performs the following actions:

   ◆ Unmounts the snapshot volumes.

   ◆ Mounts the original volumes back with the file systems that are configured to use the volume.

   Note This action can only be performed after a “begin” action has been completed and cannot be used after a reverse resynchronization has been committed.

6. If the abort action was successful, you will receive a confirmation message. Click **Show details** to see the actions completed. When you are through, click **OK** to continue.

Viewing the Snapplan Log

The Snapplan Log displays information about the Snapplan, disk group, snapshot Plex tag. It also displays the VxDBA volume name and the snapshot status.

▼ To view a Snapplan Log

1. In the object tree, click the snapplan whose log you wish to see. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to view the log of a snapplan.

   ◆ From the menu bar, select **Snapplan** > **View Log**.

   or

   ◆ Right click the snapplan to bring up a pop-up menu. Then click **View Log**.

   A pop-up window appears displaying the path of the Snapplan and the Log. The information displayed is similar to the following output:

```
SNAPSHOT_VERSION=4.0
PRIMARY_HOST=exmoor
SECONDARY_HOST=exmoor
PRIMARY_DG=DB2dg
SNAPSHOT_DG=SNAP_DB2dg
DB2DATABASE=UDB
DB2HOME=/db2/udb_home
REDOLOG_DEST=/db2/udb_home/inst01/NODE0000/SQL00001/SQLOGDIR/
SNAPSHOT_MODE=online_snapshot
SNAPSHOT_PLAN_FOR=database
SNAPSHOT_PLEX_TAG=dbed_flashsnap
SNAPSHOT_VOL_PREFIX=SNAP_
ALLOW_REVERSE_RESYNC=yes

STORAGE_INFO
DB2dg03
SNAP_PLEX=testvol01-02 testvol02-02 testvol03-02 testvol04-02
testvol05-02 udb_home-02

STATUS_INFO
SNAP_STATUS=init_full
DB_STATUS=init
LOCAL_SNAPPLAN=/home/inst01/snapplans/sp1
```

3. Click **OK** to continue.

Maintaining Your System Configuration

VERITAS Storage Foundation *for DB2* maintains a repository that stores the pertinent information needed to display configuration information. This repository is located at `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE`. When the database configuration changes, the information stored in the repository may not be up-to-date. You can resynchronize the repository, if needed. You can also rescan, or refresh, instances and tablespaces to make sure system information is up-to-date.

The GUI also allows you to check and save the configuration of each DB2 instance on your system. Information on all volumes, file systems and their types, and disk groups can be displayed.

Resynchronizing the VxDBA Repository

▼ **To resynchronize the VxDBA repository**

1. Click the DB2 database icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to resynchronize the repository.

   ◆ From the menu bar, select **DB2** > **Resync Repository**.

   or

   ◆ Right click the DB2 database icon to bring up a pop-up menu. Then click **Resync Repository**.

3. If the resynchronization was successful, you will receive a confirmation message. Click **OK** to continue.

Rescanning the Instance

1. Click the DB2 instance icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to rescan the instance.

   ◆ From the menu bar, select **DB2** > **Rescan**.

   or

   ◆ Right click the DB2 instance icon to bring up a pop-up menu. Then click **Rescan**.

3. If the rescan was successful, you will receive a confirmation message. Click **OK** to continue.

Displaying and Rescanning Tablespace Information

▼ To display tablespace information

Click a tablespace in the object tree. (You may need to expand the tree view to find the icon.) The tablespace information is displayed on the right side of the window.

▼ To rescan tablespace information

1. Click the **Tablespaces** icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to refresh the tablespace information.

   ◆ From the menu bar, select **Tablespace** > **Rescan**.

   or

   ◆ Right click the Tablespaces icon to bring up a pop-up menu. Then click **Rescan**.

3. If the refresh was successful, you will receive a confirmation message. Click **OK** to continue.

Viewing DB2 Container Topology or Statistics

1. Click a container in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to generate datafile statistics.

   - From the menu bar, select **Container** > **Topology/Statistic**.

     or

   - Right click the container to bring up a pop-up menu. Then click **Topology/Statistic**.

   The Topology/Statistic page is displayed.

3. To view detailed information, click an object in the tree and then click the **Detailed Information** tab at the bottom of the page.

4. Optionally, if you want I/O statistics, enter a sampling time (in minutes) in the **Sampling interval (minutes)** field and then press **Get Statistics**.

   Information is displayed in the Operations, File Blocks, and Average Time (ms) columns of the page. A confirmation dialog is displayed. Click **OK** to continue.

5. When you are through, click the **Close** button to quit.

Converting Regular Containers to Quick I/O Files

VERITAS Storage Foundation *for DB2* provides an option to convert your regular containers to Quick I/O files to improve performance.

Prerequisites

- Files you want to convert must be regular containers on VxFS file systems or links that point to regular VxFS files.

Usage Notes

- Converting existing containers to be Quick I/O files may not be the optimal thing to do if these files are fragmented.
- You can only convert DMS containers. This operation is not supported on SMS containers.

▼ To convert DB2 containers to Quick I/O files

1. Make the database inactive.

2. Click a container in the object tree. (You may need to expand the tree view to find the icon.)

3. Select one of the following methods to generate container statistics.

   ◆ From the menu bar, select **Container** > **Conversion**.

   or

   ◆ Right click the container to bring up a pop-up menu. Then click **Conversion**.

4. At the prompt, click **Yes** to convert the container to a Quick I/O file.

   If the container was successfully converted to a Quick I/O file, you will receive a confirmation message. Click **OK** to view the information.

Converting Quick I/O Files to Regular Containers

VERITAS Storage Foundation *for DB2* provides an option to convert your Quick I/O files to regular DB2 containers.

Prerequisites

◆ Files you want to convert must be Quick I/O files on VxFS file systems or links that point to Quick I/O files.

▼ To convert Quick I/O files to regular containers

1. Make the database inactive.

2. Click a Quick I/O file in the object tree. (You may need to expand the tree view to find the icon.)

3. Select one of the following methods to generate container statistics.

   ◆ From the menu bar, select **Container** > **Conversion**.

   or

   ◆ Right click the Quick I/O file to bring up a pop-up menu. Then click **Conversion**.

4. At the prompt, click **Yes** to convert the Quick I/O files to a regular container.

   If the Quick I/O file was successfully converted to a regular container, you will receive a confirmation message. Click **OK** to view the information.

Updating Rescan Intervals

VERITAS Storage Foundation *for DB2* periodically scans the system for updated information. You can modify the rescan intervals to be faster or slower. A partial scan is a scan of existing known objects and a full scan is a scan of known and unknown objects, meaning that a search for new information is performed.

▼ To update a rescan interval

1. Click the DB2 Instances icon in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to update the rescan interval times.

   ◆ From the menu bar, select **DB2** > **Update Rescan Intervals**.

     or

   ◆ Right click the DB2 Instances icon to bring up a pop-up menu. Then click **Update Rescan Intervals**.

3. To update a partial rescan interval, enter a new time (in minutes) in the appropriate field.

4. To update a full rescan interval, enter a new time (in minutes) in the appropriate field.

5. Click **Save** to save your changes. A confirmation dialog is displayed. Click **OK** to continue.

Checking the System Configuration

You can check the System Configuration only when using VERITAS Storage Foundation *for DB2* Enterprise Edition.

▼ To scan the system configuration of a database

1. Click the DB2 database in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to view the database configuration information.

   ◆ From the menu bar, select **DB2** > **Check System Configuration**.

   or

   ◆ Right click the database to bring up a pop-up menu. Then click **Check System Configuration**.

3. When the system has scanned the database, it will display a confirmation message. Click **Yes** or **No** to continue. The Scan Configuration Output window then displays. You should see output similar to the following:

```
Examining File System and DB2 Container attributes.

Total of 0 containers over 0 file systems.
WARNING: No file systems to examine.
Examining Quick I/O settings.
Examining Cached Quick I/O settings.
NOTICE: No file systems have Cached Quick I/O enabled.

The database has:
 3 SMS Containers
 1 DMS File Containers
 0 DMS Device Containers

Examining DB2 container fragmentation.

NOTICE: Could not examine DB2 container fragmentation.

Examining File System tunable settings.

NOTICE: Parameters for all VxFS file systems used by PROD.

Examining DB2 Volume layout and attributes.

WARNING: Data for database PROD is spread over multiple volume
groups.
```

```
Examining DB2 internal information.

DB2 Version is 8.1.

Examining DB2 logging mode.

The database has transaction logs in directory
/udb_home/prod/inst01/NODE0000/SQL00001/SQLOGDIR/.

WARNING: Transaction log directory is not mirrored using VxVM.

The database is in circular log mode (not archivelog).
WARNING: Database recovery is affected by this mode.

The database is archiving logs in the default location.
All archived logs remain under
/udb_home/prod/inst01/NODE0000/SQL00001/SQLOGDIR/.

Examining DB2 Database Free Space.

DB20000I  The SQL command completed successfully.

 Name                                 = SYSCATSPACE
 Type                                 = System managed space
 Total pages                          = 5570
 Used pages                           = 5570
 Free pages                           = Not applicable
 Page size (bytes)                    = 4096

 Name                                 = TEMPSPACE1
 Type                                 = System managed space
 Total pages                          = 1
 Used pages                           = 1
 Free pages                           = Not applicable
 Page size (bytes)                    = 4096

 Name                                 = USERSPACE1
 Type                                 = System managed space
 Total pages                          = 1
 Used pages                           = 1
 Free pages                           = Not applicable
 Page size (bytes)                    = 4096

 Name                                 = TBS1
 Type                                 = Database managed space
 Total pages                          = 5000
```

```
Used pages                            = 160
Free pages                            = 4800
Page size (bytes)                     = 4096
```

4. Click **OK** to return to the main window.

Saving the System Configuration

You can save the System Configuration only when using VERITAS Storage Foundation *for DB2* Enterprise Edition.

▼ To save the system configuration of a database

1. Click the DB2 database in the object tree. (You may need to expand the tree view to find the icon.)

2. Select one of the following methods to view the database configuration information.

- ◆ From the menu bar, select **DB2** > **Save System Configuration**.

  or

- ◆ Right click the database to bring up a pop-up menu. Then click **Save System Configuration**.

The System Configuration wizard is then displayed.

3. Enter a path name, or directory, in the Path field to indicate where you would like to save the system configuration information.

4. Click **Save** to save the configuration information. A confirmation dialog is displayed. Click **OK** to continue.

Using the VxDBA Utility 11

This chapter describes how to use the VxDBA utility to support administrative tasks for DB2 database management. This is the last release of Storage Foundation *for DB2* to support the VxDBA utility.

Topics include:

Overview of the VxDBA Menus

VxDBA Main Menu

The VxDBA utility main menu provides access to the following operations:

```
VERITAS Storage Foundation for DB2(DB2DATABASE'PROD')
Menu: Database Main

  1      Database Administration
  2      Display Database/VxDBA Information
  3      Monitoring Agent Administration

  ?      Display Help About the Current Menu
  q      Exit From Current Menu
  x      Exit From VxDBA Utility

  Select Operation to Perform:
```

Type the number, letter, or symbol of the operation you want to perform. You can also use the interrupt key (Ctrl-C) to end or break out of any operation and return to the system prompt.

The VxDBA main menu provides access to the following operations:

◆ **1 - Database Administration**

  Use this menu to perform basic database management operations.

  The **Database Administration** menu provides access to the following operations:

  ◆ **Startup Database Instance**

  ◆ **Shutdown Database Instance**

  ◆ **Display/Update Tablespace Information**

- **2 - Display Database/VxDBA Information**

  Use this menu to display information about various aspects of your database environment, as well as examine and save configuration information for database recovery.

  The **Display Database/VxDBA Information** menu provides access to the following operations:

  - **Display Database Information**
  - **Display/Update Tablespace Information**
  - **Display Container/File System Information**
  - **Display VxDBA/Database Configuration Files**
  - **Examine Volume/File System/Database Configuration**
  - **Save Volume/File System/Database Configuration**

- **3 - Monitoring Agent Administration**

  Use this menu to monitor and manage key aspects of your database environment. The primary function of VxDBA's Monitoring Agent is to monitor space usage of your database file systems, tablespaces, and containers. The Monitoring Agent can be configured to send alarm notifications and automatically grow file systems when space usage exceeds user-defined thresholds.

  The **Monitoring Agent Administration** menu provides access to the following operations:

  - **File System Space Administration**
  - **DB2 Tablespace/Container Space Administration**
  - **Configure Monitoring Agent Options**
  - **Start/Stop Monitoring Agent**

Each VxDBA menu has the following standard operational and navigational controls:

- ? - **Display Help About the Current Menu**

  This menu option provides online help for the current VxDBA menu, listing the available operations and a definition of each.

- q - **Exit From Current Menu**

  This menu option returns you to the main menu if you are in one of the administration submenus or exits VxDBA if you are at the main menu level.

- x - **Exit From VxDBA Utility**

  This menu option exits the VxDBA utility.

VxDBA Submenu Operations

Most of the operations available from the VxDBA submenus are run as the DB2 database administrative user (typically, user ID `db2inst1`), which allows the VxDBA utility permission to connect directly to the database and gather information from the system catalog. This information includes the status of the database (for example, ACTIVE or INSTANCE DOWN), the number of tablespaces and containers in the current DB2 database, and the number of file systems that these containers are spread across. When available, VxDBA displays these fields as header information on the submenus. For example:

```
Database Status : ACTIVE
# File Systems  : (1)
# Tablespaces   : (3)
# Containers    : (4)
```

A subset of the VxDBA submenu operations requires superuser (`root`) privileges to interact with the more secure or "privileged" operations of the VERITAS Storage Foundation *for DB2* product (for example, the VxDBA Monitoring Agent). When you run VxDBA operations as `root`, VxDBA cannot connect to and obtain information directly from the database, so the submenu **Database Status** header reports a permission error. For example:

```
Database Status : Instance Down
# File Systems  :  (1)
# Tablespaces   :  (3)
# Containers    :  (4)
```

Because VxDBA cannot obtain information directly from the database at this time, the values for the number of file systems, tablespaces, and containers are enclosed in parentheses to indicate that this was the last value VxDBA was able to obtain from the system catalog.

When executing various options of the `vxdba` command, at times the database status indicates **Database Down** as shown below.

```
VERITAS Storage Foundation for DB2 (DB2DATABASE 'PROD')
Menu: Database Administration
Database Status : Database Down

# File Systems : (1)
# Tablespaces : (3)
# Containers : (4)
```

The **Database Down** status means that the database is inactive, and the instance is up. If the instance is down, all databases of that instance are inactive, and the database status is **Instance Down**.

Starting VxDBA

Most VxDBA utility operations can be run by the DB2 Instance Owner (typically, the user ID `db2inst1`) of the database instance. Some VxDBA utility operations, like many of the file system space management operations, require superuser (`root`) privileges. VxDBA prompts you for the `root` password when required.

Prerequisites

Before running VxDBA, you must:

- Define the environment variable *`$DB2DATABASE`*.
- Have appropriate permission to run the utility. VxDBA requires permission changes to allow database administrators to access these tools. The default settings for the `/opt/VRTSdb2ed` directory at installation time allows only superuser (`root`) access to the directory. If you did not make these permission changes when prompted during installation, you can grant administrators access to VxDBA now.

▼ To grant administrative access to VxDBA

1. Use the `chown` and `chmod` commands to allow single user access to VxDBA. For example:

```
# chown db2inst1 /opt/VRTSdb2ed
# chmod 500 /opt/VRTSdb2ed
```

2. Use the `chgrp` and `chmod` commands to allow group user access to VxDBA. For example:

```
# chgrp db2iadm1 /opt/VRTSdb2ed
# chmod 550 /opt/VRTSdb2ed
```

▼ To start VxDBA

1. Before Starting VxDBA, verify the repository information exists in `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE`. If it does not exist, update the repository using the following command:

```
$ db2ed_update -D $DB2DATABASE
```

2. To start VxDBA, enter the following command at the administrative prompt:

```
$ /opt/VRTSdb2ed/bin/db2ed_vxdba
```

VxDBA starts up and displays the main menu containing the available operations.

Using VxDBA to Perform Administrative Operations

The rest of this chapter details the administrative operations available through the VxDBA utility.

Managing Your Database

Use the **Database Administration** menu to perform basic database operations.

This operation displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2(DB2DATABASE 'PROD')
Menu: Database Administration

       Database Status : ACTIVE
       # File Systems  : 1
       # Tablespaces   : 4
       # Containers    : 4

 1     Startup Database Instance
 2     Shutdown Database Instance
 3     Display/Update Tablespace Information

 ?     Display Help About the Current Menu
 q     Exit From Current Menu
 x     Exit From VxDBA Utility

Select Operation to Perform:
```

Select from the following operations:

Startup Database Instance–Use this menu option to start up the database.

Shutdown Database Instance–Use this menu option to shut down the database.

Display/Update Tablespace Information–Use this menu option to update the tablespaces of a DB2 database and their associated containers.

Starting Up a Database Instance

The VERITAS Storage Foundation *for DB2* software package includes adaptable scripts that are run automatically when starting the database using the VxDBA utility. You can modify these scripts to run other tools and applications or to start and stop other services before and/or after database startup. For example, you could modify the scripts to ensure the database auxiliary program is also correctly started.

The following scripts are included in the `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE`*num* directory:

- startup.pre
- startup.post

These scripts are copied from the system-wide default files, located in `/opt/VRTSdb2ed/lib` directory:

- startup.pre.base
- startup.post.base

Use the **Startup Database Instance** menu option to bring the DB2 database online. You must define the `$DB2DATABASE` and `$DB2INSTANCE` environment variables before attempting to start up the database instance. If you do not define the `$DB2INSTANCE` variable, the DB2 database takes the `$DB2INSTANCE` as the current user ID.

This operation displays a screen similar to the following:

```
-------------------------------------------------------------
VxDBA: Startup Database Instance -inst01
-------------------------------------------------------------


Database pre-startup script completed.

DB20000I  The DB2START command completed successfully.

VxDBA: DB2 instance inst01 successfully started.

Database post-startup script completed.


Press <Return> to continue ...
```

Shutting Down a Database Instance

The VERITAS Storage Foundation *for DB2* software package includes adaptable scripts that are automatically run when stopping the database using the VxDBA utility. You can modify these scripts to run other tools and applications or to start and stop other services before and/or after database shutdown. For example, you could modify the scripts to ensure the database TCP listener is also correctly stopped. The following scripts are included in the `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE`*`num`* directory:

- shutdown.pre
- shutdown.post

These scripts are copied from the system-wide default files, located in the `/opt/VRTSdb2ed/lib` directory:

- shutdown.pre.base
- shutdown.post.base

To perform some administrative tasks (for example, conversion of tablespace containers), the database must be offline. Use the **Shutdown Database Instance** menu option to bring the database instance down.

This operation displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2(DB2DATABASE 'PROD')
Menu: Shutdown Database Instance

  1       Shutdown NORMAL
  2       Shutdown FORCE

  ?       Display Help About the Current Menu
  q       Exit From Current Menu
  x       Exit From VxDBA Utility

Select Operation to Perform: 1
```

Select from the following shutdown methods:

Shutdown NORMAL. Use this menu option to shut down the DB2 database in normal situations. When this operation is selected, no new database connections are allowed. DB2 waits for all currently connected users to disconnect from the database, and then closes and dismounts the database before shutting down the instance. The next database startup does not require an instance recovery.

Shutdown FORCE. Use this menu option to forcibly shut down the DB2 database when there are existing connections to the database. When this operation is selected, no new database connections are allowed. DB2 immediately terminates all sessions and shuts down the database. The next database startup may require an instance recovery.

The **Shutdown NORMAL** operation displays a screen similar to the following:

```
-------------------------------------------------------------
VxDBA: Shutdown (NORMAL) Database Instance - inst01
-------------------------------------------------------------


Database pre-shutdown script completed.

DB20000I  The DB2STOP command completed successfully.

Database post-shutdown script completed.


DB2 instance inst01 successfully shut down.

Press <Return> to continue ...
```

After the DB2 database is shut down, VxDBA displays the **Database Administration** menu.

Displaying Database/VxDBA Information

Use the **Display Database/VxDBA Information** menu option to display information about various aspects of your database environment.

This operation displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2(DB2DATABASE'PROD')
Menu: Display Database/VxDBA Information

 1     Display Database Information
 2     Display/Update Tablespace Information
 3     Display Container/File System Information
 4     Display VxDBA/Database Configuration Files
 5     Examine Volume/File System/Database Configuration
 6     Save Volume/File System/Database Configuration

 ?     Display Help About the Current Menu
 q     Exit From Current Menu
 x     Exit From VxDBA Utility

Select Operation to Perform:
```

Select from the following operations:

Display Database Information. Use this menu option to display the database instance and status information.

Display/Update Tablespace Information. Use this menu option to display and update the tablespaces of DB2 databases and their associated containers.

Display Container/File System Information. Use this menu option to display the list of database files and the file systems used by the DB2 database.

Display VxDBA/Database Configuration Files. Use this menu option to display and view the contents of various VxDBA and DB2 configuration files.

Examine Volume/File System/Database Configuration. Use this menu option to display general system configuration and database information, such as Quick I/O files, groups, log files, containers, and the layout and version of the file systems.

Save Volume/File System/Database Configuration. Use this menu option to save important system hardware, operating system, kernel tunables, database layout and control files, volume and file system configuration, packaging, and license information.

Displaying Database Information

Use the **Display Database Information** menu option to display the DB2 database name (`$DB2DATABASE`), the DB2 instance (`$DB2INSTANCE`), the DB2 release level, and the status of the DB2 database.

This operation displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2(DB2DATABASE 'PROD')
Menu: Display Database Information

       DB2DATABASE      : PROD
       DB2 INSTANCE     : db2inst1
       DB2 Version      : 8.1
       Database Status  : INSTANCE DOWN
       # File Systems   : 1
       # Tablespaces    : 4
       # Containers     : 4

 1     Refresh Status

 ?     Display Help About the Current Menu
 q     Exit From Current Menu
 x     Exit From VxDBA Utility

Select Operation to Perform:
```

If the display screen has been idle for a while, the status of the DB2 database may have changed. Use the **Refresh Status** option to refresh the display with the up-to-date status of the DB2 database.

Displaying and Updating Tablespace Information

Use the **Display/Update Tablespace Information** menu option to display the list of tablespaces of DB2 database and their associated containers. This menu option is also available on the **Database Administration** menu.

This operation displays a screen similar to the following:

```
-------------------------------------------------------------
VxDBA: Display/Update Tablespace Information -
DB2.db2inst1.PROD/NODE0000
-------------------------------------------------------------

 ---- Table Space ----
 Name           Type       Data         Container Name
 ----------- ---   ----
-------------------------------------------------------
SYSCATSPACE     SMS        ANY         /eeevol1/system1.con
SYSCATSPACE     SMS        ANY        /eeevol1/system2.con
SYSCATSPACE     SMS        ANY        /eeevol1/system3.con
TEMPSPACE1      SMS        SYSTMP    /eeevol1/db/db2inst1/NODE0000
                                     /SQL00001/SQLT0001.0
USERSPACE1      SMS        ANY       /eeevol1/db/db2inst1/NODE0000
                                     /SQL00001/SQLT0002.0
TBS1            DMS        ANY       /suzfs/tbs1
TBS2            DMS        ANY       /suzfs/tbs2.dbf
TEMPORARY1      SMS        SYSTMP    /suzfs/temp1.con

All database configuration files are up-to-date.
```

VxDBA maintains a repository that stores the pertinent information needed to display configuration information. This repository is located at `/etc/vx/vxdba/$/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000`.

When the database configuration changes, the information stored in the repository may not be up-to-date. Use the `db2ed_update_all` or `db2ed_update` commands to update the repository.

Displaying Container and File System Information

Use the **Display Container/File System Information** menu option to display the list of database files and the file systems used by the DB2 database.

This operation displays a screen similar to the following:

```
----------------------------------------------------------------
VxDBA: Display Container/File System Information - PROD
----------------------------------------------------------------

        Database Status : Instance Down
        # File Systems  : (11)
        # Tablespaces   : (8)
        # Containers    : (13)

Container                                               ---- Table Space ----   ---- File  System ----
                                                        Name        Type  Data  Type   Name
------------------------------------------------------- ----------- ---   ----  -----  -------------
/tmp/udb01f/udb_home/inst01_home/udb01a_system1/udb01a_system1.con SYSCATSPACE SMS   ANY   vxfs
/tmp/udb01f/udb_home/inst01_home/udb01a_system1
/tmp/udb01f/udb_home/inst01_home/udb01a_system2/udb01a_system2.con SYSCATSPACE SMS   ANY   vxfs
/tmp/udb01f/udb_home/inst01_home/udb01a_system2
/tmp/udb01f/udb_home/inst01_home/udb01a_system3/udb01a_system3.con SYSCATSPACE SMS   ANY   vxfs
/tmp/udb01f/udb_home/inst01_home/udb01a_system3
/tmp/udb01f/udb_home/inst01_home/udb01a_user1_1/udb01a_user1_1.dbf USER1       DMS   ANY   vxfs
/tmp/udb01f/udb_home/inst01_home/udb01a_user1_1
/tmp/udb01f/udb_home/inst01_home/udb01a_user2_1/udb01a_user2_1.dbf USER2       DMS   ANY   vxfs
/tmp/udb01f/udb_home/inst01_home/udb01a_user2_1
/tmp/udb01f/udb_home/inst01_home/udb01a_user2_2/udb01a_user2_2.dbf USER2       DMS   ANY   vxfs
/tmp/udb01f/udb_home/inst01_home/udb01a_user2_2
/tmp/udb01f/udb_home/inst01_home/udb01a_indx1_1/udb01a_indx1_1.dbf INDX1       DMS   ANY   vxfs
/tmp/udb01f/udb_home/inst01_home/udb01a_indx1_1
/tmp/udb01f/udb_home/inst01_home/udb01a_user3_1/udb01a_user3_1.con USER3       SMS   ANY   vxfs
/tmp/udb01f/udb_home/inst01_home/udb01a_user3_1


Press <Return> to continue ...
        or <q> to skip ...
```

The container and file system information is also stored in VxDBA's repository. When the configuration changes, use the `db2ed_update_all` or `db2ed_update` commands to update the repository.

Displaying VxDBA and Database Configuration Files

Use the **Display VxDBA/Database Configuration Files** menu option to display and then view the contents of various VxDBA and DB2 or script files.

This operation displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2 (DB2DATABASE 'PROD')
Menu: Display VxDBA/Database Configuration Files

 1       Database Pre-Startup Script
 2       Database Post-Startup Script
 3       Database Pre-Shutdown Script
 4       Database Post-Shutdown Script
 5       VxDBA Settings File
 6       VxDBA Monitoring Agent Configuration File
 7       Database Manager Configuration-Instance
 8       Current Database Configuration

 ?       Display Help About the Current Menu
 q       Exit From Current Menu
 x       Exit From VxDBA Utility

Select Operation to Perform:
```

When you select an item from the list, VxDBA first displays the location of the file and then its contents. Use the space bar to page through the file contents. If VxDBA cannot find a configuration file, you will see a `(missing)` designation after the selection option description.

Note You cannot edit the configuration files from within VxDBA; you can only view their contents.

Examining Database Environment Information

Use the **Examine Volume/File System/Database Configuration** menu option to display system configuration and database information that can assist you in determining if your database is configured properly on top of VERITAS products and the operating system.

Note The database must be running in archivelog mode.

By default, only users with superuser (`root`) privileges have permission to run `vxtunefs` and `vxdisk` commands. To allow database administrators to use these commands, you need to change these commands as follows:

```
# chown root:db2iadm1 /opt/VRTSvxfs/sbin/vxtunefs
# chmod 4550 /opt/VRTSvxfs/sbin/vxtunefs
# chown root:db2iadm1 /usr/sbin/vxdisk
# chmod 4550 /usr/sbin/vxdisk
```

This operation displays a screen similar to the following:

```
 -------------------------------------------------------------
 VxDBA: Examine Volume/File System/Database Configuration
-------------------------------------------------------------
       Database Status : ACTIVE
        # File Systems  : 1
        # Tablespaces   : 1
        # Datafiles     : 1
 Examining file system attributes.
 NOTICE: All file systems are VxFS.
 NOTICE: All file systems are VxFS Version 4 or higher layout.
 Examining Quick I/O settings.
 NOTICE: 1 files are not configured to use Quick I/O.
 Examining datafiles fragmentation.
 NOTICE: 1 files are fragmented.
 Examining File System tunable settings.
 NOTICE: Parameters for all VxFS file systems used by PROD.
 Filesystem i/o parameters for /DB2
 rfcl_winterval = 3600
 Press <Return> to continue ...
        or <q> to skip ...
```

Managing Space Usage and the VxDBA Monitoring Agent

Use the **Monitoring Agent Administration** menu to:

- Manage and monitor VxFS file system, DB2 tablespace, and container space usage
- Start and stop the Monitoring Agent

This operation displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2(DB2DATABASE 'PROD')
Menu: Monitoring Agent Administration

 1    File System Space Administration
 2    DB2 Tablespace/Container Space Administration
 3    Configure Monitoring Agent Options
 4    Start/Stop Monitoring Agent

 ?    Display Help About the Current Menu
 q    Exit From Current Menu
 x    Exit From VxDBA Utility

 Select Operation to Perform:
```

Managing File System Space

The VxDBA Monitoring Agent monitors the file system space, and when the space usage reaches a configured threshold value, a predefined action script grows the file system automatically. The agent can be enabled or disabled to start at boot-time. Each file system monitored has three settings that the Monitoring Agent needs to know about:

- *Warning Threshold* is a percent value (% of file system utilized) that determines when the agent begins warning the administrator of space shortage
- *Grow Threshold* is a percent value (% of file system utilized) that determines when the agent is to attempt to grow the file system (when space usage is at a critical level)
- *Amount* is either a percentage or a value in megabytes by which to grow file systems when the Grow Threshold is reached or exceeded

The VxDBA Monitoring Agent operations are driven from the following files:

- `/opt/VRTSd2bed/lib/dbed_mon_config.base`
- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000 \`
  `/db2ed_mon_config.DB2.$DB2INSTANCE.$DB2DATABASE.NODE0000`
- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000 \`
  `/db2ed_mon_fslist.DB2.$DB2INSTANCE.$DB2DATABASE.NODE0000`
- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000 \`
  `/db2ed_mon_db2list.DB2.$DB2INSTANCE.$DB2DATABASE.NODE0000`

- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000/include`

The `/opt/VRTSdb2ed/lib/db2ed_mon_config.base` file contains the site-level configuration settings for monitoring all file systems and databases recognized. This configuration file specifies how often to check for file system and database configuration changes, how often to check the file space usage, where space usage information gets logged, and the thresholds for warning and automatically growing the file system. By default, the monitoring agent log file is located under the `/var/log/db2ed_mon` directory.

During file system and database configuration, the `dbed_mon_config.base` file gets copied into each database-specific directory as the file `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000 \ /db2ed_mon_config.$DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000`. For example, if you are monitoring a database named `PROD`, the database-specific file would be `/etc/vx/vxdba/PROD/db2ed_mon_config.PROD`. This is the first file opened when the agent is started and contains the default settings for monitoring file systems at the database level. The VxDBA Monitoring Agent cannot start without this file. Modify this configuration file if you want to change the preconfigured settings carried over from the `dbed_mon_config.base` file to maintain a different set of settings at the database level.

The following two files:

- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000 \`
  `/db2ed_mon_fslist.DB2.$DB2INSTANCE.$DB2DATABASE.NODE0000`
- `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000 \`
  `/db2ed_mon_db2list.DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000`

are created by the VxDBA utility and are used for restarting the VxDBA Monitoring Agent. These files specify the status of the database. The files also specify the space monitoring and alarm information for each file system, tablespace, and container. You can edit these files manually to change settings, and then use VxDBA to restart the Monitoring Agent.

The VxDBA Monitoring Agent uses the `/etc/vx/vxdba/$DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000/include` file to check that all files are up-to-date and are being monitored. This file is created by the VxDBA utility and should not be edited.

Use the **File System Space Administration** menu to monitor the space usage of the file system. You can also use the menu to enable or disable the VxDBA Monitoring Agent.

This operation displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2(DB2DATABASE 'PROD')
Menu: File System Space Administration

       Database Status : ACTIVE# File Systems    : 1
       # Tablespaces     : 4
       # Containers      : 4

 1     Display File System Space Usage
 2     Display File System Space Alarm Settings
 3     Enable/Disable/Modify Space Alarm Settings

 ?     Display Help About the Current Menu
 q     Exit From Current Menu
 x     Exit From VxDBA Utility

Select Operation to Perform:
```

Select from the following File System Space Alarm Administration operations:

Display File System Space Usage

Use this menu option to display file system space usage information.

Display File System Space Alarm Settings

Use this menu option to display the space alarm information on the file systems used by the DB2 database. The menu operation provides the boot-time status and current status of the VxDBA Monitoring Agent, as well as the list of file systems with their associated space alarm settings and status (ENABLED or DISABLED).

Enable/Disable/Modify Space Alarm Settings

Use this menu option to control the Monitoring Agent activity and modify the current space alarm settings on the file systems used by the DB2 database.

Displaying File System Space Usage

This operation displays the space usage of the file system daily database space consumption.

Display File System Usage displays a screen similar to the following:

```
-------------------------------------------------------
VxDBA: Display File System Space Usage - PROD
-------------------------------------------------------

       DB2DATABASE      : PROD
       DB2INSTANCE      : db2inst1
       DB2 Version      : 8.1
       Database Status : ACTIVE# File Systems  : 1
       # Tablespaces    : 4
       # Containers     : 4

File System                   FS Size      Used   Avail   %Full
--------------------------------------  --------- -------  -----
/db01                           929MB     850MB   79MB     91%


Press <Return> to continue...
```

Displaying File System Space Alarm Settings

This operation displays the information about the space alarm settings defined for the file systems used by the DB2 database.

The space alarm relies on the VxDBA Monitoring Agent. The agent daemon processes must be running first. If the agent daemons are not running, a message is displayed asking you to start the agent daemons.

```
-------------------------------------------------------------
VxDBA: Display File System Space Alarm Settings - PROD
-------------------------------------------------------------


       DB2DATABASE      : PROD
       DB2 INSTANCE     : db2inst1
       DB2 Version      : 8.1
       DATABASE STATUS : ACTIVE

       # Tablespaces   : 4
       # Containers    : 4

Monitoring Agent is DISABLED at system boot time.

Monitoring Agent is not running.

The Monitoring Agent daemon must be running for
the file system space alarm to work. You can start
the Monitoring Agent using the VxDBA utility. From
the File System Space Administration menu, select
menu item 3 Enable/Disable/Modify Space Alarm
Settings to configure and start the Monitoring
Agent daemon. You can also start the Monitoring
Agent daemon automatically at system boot time
using this menu item.

Press <Return> to continue...
```

After you start the VxDBA Monitoring agent, **Display File System Space Alarm Settings** displays the list of file systems and the space alarm status:

```
-------------------------------------------------------------
VxDBA: Display File System Space Alarm Settings - PROD
-------------------------------------------------------------


        DB2DATABASE       :  PROD
        DB2INSTANCE       : db2inst1


        DB2 Version       : 8.1
        Database Status   : ACTIVE
        # Tablespaces     : 4
        # Containers      : 4

Monitoring Agent is DISABLED at system boot time.

Monitoring Agent is running as pid 6991.

Press <Return> to continue...

-------------------------------------------------------------
VxDBA: Display File System Space Alarm Settings - PROD
-------------------------------------------------------------

 File System                        Status   Thresholds Grow By
                                             Warn Grow
 ---------------------------------- -------- ---- ----  ------
 /db01                               enabled  70     90     5%
```

Enabling, Disabling, or Modifying Space Alarm Settings

When the file system runs out of space, VxFS automatically removes Storage Checkpoints to free up space. This could happen when DB2 is processing update transactions such that original data blocks are saved in the Storage Checkpoints. Enabling the space alarm allows VxDBA to monitor space usage and grow the file systems automatically, so that Storage Checkpoints are not unnecessarily removed.

Note Only users with superuser (`root`) privileges can perform this operation.

The **Enable/Disable/Modify Space Alarm Settings** operation first checks to see if you are logged in as `root`. If you are not logged in as `root`, VxDBA prompts you for the `root` password:

```
---------------------------------------------------------------
VxDBA: Enable/Disable/Modify Space Alarm Settings - PROD
---------------------------------------------------------------

You must be root to access the space alarm.

If you can enter the root password, you can continue.

Continue? [y,n,q,?] (default: y) y
Password:
```

After you enter the `root` password, **Enable/Disable/Modify Space Alarm Settings** displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2(DB2DATABASE 'PROD')
Menu: Enable/Disable/Modify Space Alarm Settings

       Database Status :Instance Down
       # File Systems  : (1)
       # Tablespaces   : (4)
               # Containers    : (4)


 1     Enable or Disable Boot-Time Start of Monitoring Agent
 2     Set Monitoring/Expansion Policy for All DB2 Tablespaces  3
Set Monitoring Policy Per DB2 Tablespace  4Re-Read Configuration
File for Monitoring Agent
 5     Start/Stop Monitoring Agent

 ?     Display Help About the Current Menu
 q     Exit From Current Menu
 x     Exit From VxDBA Utility

Select Operation to Perform:
```

Note When you run VxDBA operations as `root`, VxDBA cannot connect to and obtain information directly from the database, so the submenu **Database Status** header reports a permission error, and the number of tablespaces and containers are enclosed in parentheses.

Select from the following file system space alarm operations:

Enable or Disable Boot-Time Start of Monitoring Agent. Use this menu option to change the boot-time start activity of the VxDBA Monitoring Agent. You are provided with the current setting (ENABLED or DISABLED), and then prompted for changes.

While this is not recommended, you can disable the boot-time start activity of the VxDBA Monitoring Agent that monitors the space usage of the file systems used by the DB2 database.

Set Monitoring/Expansion Policy for All DB2 Tablespaces. Use this menu option to configure the monitoring and expansion policy for all DB2 tablespaces. You are prompted for the Warning Threshold for space usage, the Grow Threshold for space usage, and the Amount as a percentage or a value in megabytes by which to grow the tablespace. These three policy values are then used for all tablespaces unless a per tablespace policy is set.

Set Monitoring/Expansion Policy Per DB2 Tablespace. Use this menu option to configure the monitoring and expansion policy for a particular DB2 tablespace. After displaying the current policy per tablespace, you are asked if you want to change the policies and are then prompted for the new values for the Warning Threshold for space usage, the Grow Threshold for space usage, and the Amount as a percentage or a value in megabytes by which to grow the tablespace. If you want to disable monitoring of a particular file system, set the Grow Threshold and the amount to Grow By values to zero. By default, the expansion of file systems is disabled and must be enabled by the user.

Re-Read Configuration File for Monitoring Agent. Use this menu option to re-read the `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000 / /db2ed_mon_fslist.$DB2INSTANCE.$DB2DATABASE.NODE0000`file. Select this operation if you manually edit this file.

Start/Stop Monitoring Agent. Use this menu option to start or stop the monitoring agent. After displaying the current status of the Monitoring Agent (running or not running), you can either start or stop the Monitoring Agent.

Managing DB2 Tablespace and Container Space

Use the **DB2 Tablespace/Container Space Administration** menu to monitor the space usage of DB2 tablespaces and containers, and to display or modify the VxDBA Monitoring Agent's space alarm settings. You can also use the menu to enable or disable the VxDBA Monitoring Agent.

This operation displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2(DB2DATABASE 'PROD')
Menu: DB2 Tablespace/Container Space Administration

       Database Status : ACTIVE
       # File Systems  : 1
       # Tablespaces   : 4
       # Containers    : 4

 1     Display DB2 Tablespace/Container Space Usage
 2     Display DB2 Tablespace/Container Space Alarm Settings
 3     Enable/Disable/Modify Space Alarm Settings

 ?     Display Help About the Current Menu
 q     Exit From Current Menu
 x     Exit From VxDBA Utility

Select Operation to Perform:
```

Select from the following File System Space Alarm Administration operations:

Display DB2 Tablespace/Container Space Usage. Use this menu option to display container space usage information. Total size of DB2 objects, free space available, and DB2 blocks are displayed. DB2 objects and free space available are given in MB.

Display DB2 Tablespace/Container Space Alarm Settings. Use this menu operation to display the space alarm information on the DB2 tablespaces. The menu display provides the boot-time and current status of the VxDBA Monitoring Agent and the list of DB2 tablespaces with their associated space alarm settings and status (ENABLED or DISABLED).

Enable/Disable/Modify Space Alarm Settings. Use this menu option to change the boot-time start activity of the VxDBA Monitoring Agent. You are provided with the current setting (ENABLED or DISABLED), and then prompted for changes.

Displaying DB2 Tablespace/Container Space Usage

This operation displays the space usage of DB2 tablespaces and containers. When examining space usage for SMS tablespaces, note that since they are not preallocated, all pages that have been allocated to an SMS tablespace are in use. Since DB2 stripes data across containers in a tablespace, the free space per containers, (assuming all containers, are the same size) is calculated by the tablespace free space, divided by the number of containers, that compose the tablespace.

Display DB2 Tablespace/Container Space Usage displays a screen similar to the following:

```
---------------------------------------------------------------
VxDBA: Display DB2 Tablespace/Container Space Usage - PROD
---------------------------------------------------------------

       DB2DATABASE      :  PROD
       DB2INSTANCE      : db2inst1
       DB2 Version      : 8.1
       Database Status : Instance Down
       # File Systems  : 1
       # Tablespaces   : 4
               # Containers    : 4

  Name        Type  Data  State  Total_MB  Free_MB   Containers
------------ ----- ----- --------------  -------   ----------
 SYSCATSPACE  SMS   ANY   NORMAL     16       0      1
 TEMPSPACE1   SMS   SYSTMPNORMAL     1        0      1
 USERSPACE1   SMS   ANY   NORMAL     1        0      1
 TBS1         DMS   ANY   NORMAL     19       18     1
 TBS2         DMS   ANY   NORMAL     19       18     1
 TBS01        DMS   ANY   NORMAL     7        7      1
 TBS02        DMS   ANY   NORMAL     7        7      1
 TBS03        DMS   ANY   NORMAL     7        7      1

Press <Return> to continue...
       or <q> to skip...
  Name        Type  Data  State  Total_MB  Free_MB   Containers
------------ ----- ----- --------------  -------   ----------
      TBS04   SMS   ANY   NORMAL     7        7      1
      TBS05   SMS   ANY   NORMAL     7        7      1

Press <Return> to continue...
```

Displaying DB2 Tablespace/Container Space Alarm Settings

This operation displays the information about the space alarm settings defined for the DB2 tablespaces and containers.

The space alarm relies on the VxDBA Monitoring Agent. The agent daemon processes must be running first. If the agent daemons are not running, a message is displayed asking you to start the agent daemons.

The expansion of DB2 Tablespaces is not currently supported. The display for settings on the grow threshold and the grow amount always shows N/A.

```
------------------------------------------------------------
VxDBA: Display DB2 Tablespace/Container Space Alarm Settings -  PROD
------------------------------------------------------------


       DB2DATABASE      : PROD
       DB2INSTANCE      : db2inst1
       Database Status : INSTANCE DOWN

       # File Systems  : 1
       # Tablespaces   : 4
       # Containers    : 4


Monitoring Agent is DISABLED at system boot time.

Monitoring Agent is not running.

The Monitoring Agent daemon must be running for
the DB2 tablespace/container space alarm to work.
You can start the Monitoring Agent using the VxDBA
utility. From the DB2 Tablespace/Container Space
Administration menu, select menu item 3 Enable/Disable
/Modify Space Alarm Settings to configure and start
the Monitoring Agent daemon. You can also start the
Monitoring Agent daemon automatically at system boot
time using this menu item.

Press <Return> to continue...
```

Once you start the VxDBA Monitoring agent, **Display DB2 Tablespace /Container Space Alarm Settings** displays the list of tablespaces and containers and the space alarm status:

```
-------------------------------------------------------------
VxDBA: Display DB2 Tablespace/Container Space Alarm Settings
-------------------------------------------------------------



      DB2DATABASE     : PROD
      DB2INSTANCE     : db2inst1
      Database Status : ACTIVE
      # File Systems  : 1
      # Tablespaces   : 4
     # Containers     : 4

Monitoring Agent is ENABLED at system boot time.

Monitoring Agent is running as pid 6991.

Press <Return> to continue...

-------------------------------------------------------------
VxDBA: Display DB2 Tablespace/Container Space Alarm Settings PROD
-------------------------------------------------------------

Tablespace                          Status   Thresholds  Grow By
                                             Warn Grow
----------------------------------- -------- ---- ---- -------
SYSCATSPACE                         enabled   80    N/A N/A
TEMPSPACE1                          enabled   80    N/A N/A
USERSPACE1                          enabled   80    N/A N/A
DATATBS001                          enabled   80    N/A N/A




Press <Return> to continue...
```

Enabling, Disabling, or Modifying DB2 Space Alarm Settings

Enabling the DB2 space alarm allows VxDBA to monitor tablespace and container space usage. The warning is sent to the log file `db2ed_mon.logfile.DB2.$DB2INSTANCE.$DB2DATABASE.NODE0000`

Note Only users with superuser (`root`) privileges can perform this operation.

The **Enable/Disable/Modify DB2 Space Alarm Settings** operation first checks to see if you are logged in as `root`. If you are not logged in as `root`, VxDBA prompts you for the `root` password:

```
---------------------------------------------------------------
VxDBA: Enable/Disable/Modify Space Alarm Settings - PROD
---------------------------------------------------------------

You must be root to access the space alarm.

If you can enter the root password, you can continue.

Continue? [y,n,q,?] (default: y) y
Password:
```

After you enter the `root` password, **Enable/Disable/Modify Space Alarm Settings** displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2(DB2DATABASE 'PROD')
Menu: Enable/Disable/Modify Space Alarm Settings

       Database Status : INSTANCE DOWN
       # File Systems  : (1)
       # Tablespaces   : (4)
       # Containers    : 4

 1     Enable or Disable Boot-Time Start of Monitoring Agent
 2     Set Monitoring Policy for All DB2 Tablespaces
 3     Set Monitoring Policy Per DB2 Tablespaces
 4     Re-Read Configuration File for Monitoring Agent
 5     Start/Stop Monitoring Agent

 ?     Display Help About the Current Menu
 q     Exit From Current Menu
 x     Exit From VxDBA Utility

Select Operation to Perform:
```

Note When you run VxDBA operations as `root`, VxDBA cannot connect to and obtain information directly from the database, so the submenu **Database Status** header reports a permission error, and the number of tablespaces and containers are enclosed in parentheses.

Select from the following DB2 space alarm operations:

Enable or Disable Boot-Time Start of Monitoring Agent. Use this menu option to change the boot-time start activity of the monitoring agent. The message provides you with the current setting and prompts you for changes.

Set Monitoring Policy for All DB2 Tablespaces. Use this menu option to configure the monitoring policy for all DB2 tablespaces. The menu prompts you for the Warning Threshold for space usage.

Set Monitoring Policy Per DB2 Tablespaces. Use this menu option to configure the monitoring policy for a particular DB2 tablespace. After displaying the current policy per DB2 tablespace, the program asks if you want to change the policies and then prompts you for the new values for the Warning Threshold.

Re-Read Configuration File for Monitoring Agent. Use this menu option to re-read the `/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000 \` `/db2ed_mon_config.DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000` file. Select this operation if you manually edited this file.

Start/Stop Monitoring Agent. Use this menu option to start or stop the monitoring agent.

Configuring Monitoring Agent Options

Use this menu operation to modify current default settings for the Monitoring Agent. The agent configuration is saved under the following:

```
/etc/vx/vxdba/DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000 \
/dbed_mon_config.DB2.$DB2INSTANCE.$DB2DATABASE/NODE0000
```

When the configuration file is modified, restart the Monitoring Agent for the changes to take effect. This operation displays a screen similar to the following:

```
-------------------------------------------------------------
VxDBA: Configure Monitoring Agent Options -
-------------------------------------------------------------


You will now be prompted to modify current VxDBA
Monitoring Agent settings.

For detailed information on these settings, see
the db2ed_mon(1) manual page and the Monitoring
Agent configuration file in the following location:

/etc/vx/vxdba/DB2.inst01.PROD/NODE0000/db2ed_mon_config.DB2.inst01.
PROD.NODE0000
You will now be prompted to modify current VxDBA
Monitoring Agent settings.

For detailed information on these settings, see
the db2ed_mon(1M) manual page and the Monitoring
Agent configuration file in the following location:

/etc/vx/vxdba/DB2.inst01.PROD/NODE0000/db2ed_mon_config.DB2.inst01.
PROD.NODE0000


Continue [y,n,q,?] (default: y)
NOTICE: Default setting for CHECK is 300.

Do you want to change the setting for LOGFREQ [y,n,q,?] (default: n)
n

NOTICE: Default setting for DEF_WARN is 85.

Do you want to change the setting for DEF_WARN [y,n,q,?] (default: n)
n
Do you want to change the setting for CHECK [y,n,q,?] (default: n) y
Enter new setting for CHECK [current 300] : 60
Changing variable CHECK from 300 to 60

NOTICE: Default setting for LOGFREQ is 0.

Do you want to change the setting for LOGFREQ [y,n,q,?] (default: n)
```

```
NOTICE: Default setting for DEF_GROW is 90.
 Do you want to change the setting for DEF_GROW [y,n,q,?] (default:
n)
 NOTICE: Default setting for DEF_GROWBY is 0%.
 Do you want to change the setting for DEF_GROWBY [y,n,q,?] (default:
n)
 n
 NOTICE: Default setting for DEF_DB2_WARN is 85.
 Do you want to change the setting for DEF_DB2_WARN [y,n,q,?]
(default:
 n) n
 WARNING: No default setting for LOGPATH variable.
 Do you want to change the setting for LOGPATH [y,n,q,?] (default: n)
n
 WARNING: No default setting for LOG_EMAIL variable.
 Do you want to change the setting for LOG_EMAIL [y,n,q,?] (default:
n) n
 WARNING: No default setting for SYSLOG_FACILITY variable.
 Do you want to change the setting for SYSLOG_FACILITY [y,n,q,?]
(default: n)
 WARNING: No default setting for SYSLOG_PRIORITY variable.
 Do you want to change the setting for SYSLOG_PRIORITY [y,n,q,?]
 (default: n) n
 NOTICE: Default setting for STATS is 0.
 Do you want to change the setting for STATS [y,n,q,?] (default: n) y
 Changing variable STATS from 0 to 1800
 Changing variable STATS from 0 to 1800
Enter new setting for STATS [current 0] : 1800
```

```
 WARNING: No default setting for FSSTATSPATH variable.

 Do you want to change the setting for FSSTATSPATH [y,n,q,?]
(default: n)
 n

 NOTICE: Default setting for DB2STATS is 0.

 Do you want to change the setting for DB2STATS [y,n,q,?] (default:
n)
 Enter new setting for DB2STATS [current 0] : 1800
 Changing variable DB2STATS from 0 to 1800

 WARNING: No default setting for DB2 STATSPATH variable.

 Do you want to change the setting for DB2STATSPATH [y,n,q,?]
 (default:n) n

 The following changes are to be made to the Monitoring
 Agent configuration file.

 CHECK 60
 DB2STATS 1800


 Do you want to commit the changes [y,n,q,?] (default: n)y
 Changes committed.

 You need to restart the Monitoring Agent for these
 configuration changes to take effect.

 Do you want to restart the Monitoring Agent [y,n,q,?] (default: y)
```

Starting and Stopping the Monitoring Agent

Note Only users with superuser (`root`) privileges can perform this operation.

The **Start/Stop Monitoring Agent** operation first checks to see if you are logged in as `root`. If you are not logged in as `root`, VxDBA prompts you for the `root` password:

```
---------------------------------------------------------------
VxDBA: Start/Stop Monitoring Agent- PROD
---------------------------------------------------------------

You must be root to start or stop the Monitoring Agent.

If you can enter the root password, you can continue.

Continue? [y,n,q,?] (default: y) y
Password:
```

After you enter the `root` password, **Start/Stop Monitoring Agent** displays a screen similar to the following:

```
VERITAS Storage Foundation for DB2 (DB2DATABASE 'PROD')
Menu: Start/Stop Monitoring Agent

 VxDBA: Monitoring Agent is running as pid 6991.

 1     Start Monitoring Agent
 2     Stop Monitoring Agent

 ?     Display Help About the Current Menu
 q     Exit From Current Menu
 x     Exit From VxDBA Utility

Select Operation to Perform:
```

Select from the following menu operations:

Start Monitoring Agent. Use this menu option to start the monitoring activity of the VxDBA Monitoring Agent.

Stop Monitoring Agent. Use this menu option to stop the monitoring activity of the VxDBA Monitoring Agent.

Note VxDBA keeps a record of the Monitoring Agent process ID. To avoid any inconsistent Monitoring Agent status (running or not running), do not stop the Monitoring Agent outside of VxDBA (for example, using the `kill`(1M) command).

Setting Up VxDBA in an HA Environment

VxDBA puts its repository and lock files in the directory `/etc/vx/vxdba`. The repository for a particular database (noted by `$DB2DATABASE`) is under directory `/etc/vx/vxdba/DB2.DB2INSTANCE.DB2DATABASE`. If you are configuring your database in a high availability or cluster environment, you must configure this directory for failover.

In a VERITAS Cluster Server (VCS) environment, you should make the directory `/etc/vx/vxdba/DB2.DB2INSTANCE.DB2DATABASE` the mount point of a file system that mounts on a shared disk device. Also, set up the proper configuration for the mount agent. See the *VERITAS Cluster Server Installation Guide* for more information.

Tuning for Performance 12

This chapter provides tuning tips that you can use to improve database performance.

Topics covered in this chapter include:

- "Tuning VxVM" on page 330
- "Tuning VxFS" on page 332
- "Tuning DB2 Databases" on page 341
- "Tuning Solaris" on page 348

Use the tuning tips and information provided in this chapter in conjunction with other more in-depth publications, such as:

- *IBM Configuration and Performance RedBooks* at http://www.redbooks.ibm.com/
- *DB2 UDB V7.2 Performance Tuning Guide* (IBM)
- *DB2 UDB V8.1 Performance Tuning Guide* (IBM)
- *DB2 High Performance Design and Tuning* (Prentice Hall ; ISBN: 0132037955)
- *VERITAS Volume Manager Administrator's Guide*, chapter on "VxVM Performance Monitoring"

Tuning 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. For information on tuning VERITAS Volume Manager, refer to the “Tuning VxVM” section of the “Performance Monitoring and Tuning” chapter in the *VERITAS Volume Manager Administrator’s Guide*.

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 log2              0       6312         0     79836            .0       0.2
vol db02        2892318    3399730   0283759   7852514          20.6      25.5
```

The "Performance Monitoring" section of the "Performance Monitoring and Tuning" chapter in the *VERITAS Volume Manager Administrator's Guide* provides detailed information 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. Refer to the `vxstat`(1M) manual page for more information.

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.

Monitoring Free Space

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 for a description of online file system expansion.

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 -o s` commands. 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.

Tuning VxFS I/O Parameters

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. For more details, see the vxtunefs(1M) and tunefstab(4) manual pages. The vxtunefs command can be used to print the current values of the I/O parameters.

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`	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.

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         0         0         0       0.0       0.0
/db01/file2        0         0         0         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         0         0         0       0.0       0.0
```

For detailed information on available options, see the `qiostat`(1M) manual page.

Using 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        1          0        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        0          0        0        0.0      0.0
/db01/temp1       0        0          0        0        0.0      0.0
/db01/ware1       3       14         12       56       23.3     44.3
/logs/log1        0        0          0        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
```

Interpreting 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.

Tuning DB2 Databases

To achieve optimal performance on your DB2 database, the database needs to be tuned to work with VxFS. This section describes some of the DB2 parameters that you can tune to improve your DB2 database performance when using Quick I/O.

DB2_STRIPED_CONTAINERS (DB2 v7.x only)

If multiple containers are defined, then the data will be striped at both the DB2 container and the volume level. This double lot of striping can affect performance adversely and make determining and tuning performance metrics more difficult. In addition, if only a single container is defined and grown as necessary, the database does not have to perform restriping when additional containers are added.

The `DB2_STRIPED_CONTAINERS` variable is set using the `db2set` command.

```
$ db2set DB2_STRIPED_CONTAINERS=ON
$ db2stop ;
$ db2start
```

Any DMS container that is created (with `CREATE TABLESPACE` or `ALTER TABLESPACE`) after this setting comes into effect will have new containers with tags taking up a full extent. Existing containers will remain unchanged. To stop creating containers with this attribute, reset the `DB2_STRIPED_CONTAINERS` variable, and then stop and restart your instance.

When using hardware or software RAID devices underneath a DB2 database, volumes are striped underneath the database containers. If you are using DMS device containers in a tablespace, then this striping will also affect the values required for extent size and prefetching of data.

DB2_USE_PAGE_CONTAINER_TAG

By default, DB2 stores a container tag in the first extent of each DMS container, whether it is a file or a device. The container tag is the metadata for the container. (Before DB2 v8.1, the container tag was stored in a single page, so it required less space in the container.) It is recommended that you keep this variable set to OFF.

The `DB2_USE_PAGE_CONTAINER_TAG` variable is set using the `db2set` command.

```
$ db2set DB2_USE_PAGE_CONTAINER_TAG=OFF
$ db2stop ;
$ db2start
```

If you set this registry variable to ON when you use RAID devices for containers, I/O performance might degrade. Because for RAID devices you create table spaces with an extent size equal to or a multiple of the RAID stripe size, setting the DB2_USE_PAGE_CONTAINER_TAG to ON causes the extents not to line up with the RAID stripes. As a result, an I/O request might need to access more physical disks than would be optimal. Users are strongly advised against enabling this registry variable.

DB2_PARALLEL_IO

This setting is used to force parallel I/O to occur on tablespaces. This is important in combination with the DB2_STRIPED_CONTAINERS setting, as RAID devices have more than one physical disk and therefore can sustain a greater I/O load than non-RAID devices. DB2 achieves this parallelism by enabling multiple prefetch threads on enabled tablespaces.

The DB2_PARALLEL_IO variable is set using the db2set command. To enable parallel I/O on all tablespaces, you would run the commands:

```
$ db2set DB2_PARALLEL_IO=*
$ db2stop ; db2start
```

To enable parallel I/O on a subset of all tablespaces, you need to know the tablespace identifying number and supply a list of tablespace ids, comma separated, to the db2set command:

```
$ db2 connect to PROD
$ db2 list tablespaces
$ db2 terminate
$ db2set DB2_PARALLEL_IO=3,4,8,9
$ db2stop ; db2start
```

As per the examples, you must stop and restart your instance after modifying the DB2_PARALLEL_IO setting. It is also recommended that DB2_PARALLEL_IO be enabled for tablespaces residing on RAID devices when PREFETCHSIZE > EXTENTSIZE.

PREFETCHSIZE and EXTENTSIZE

Prefetching is a behavior that increases database performance in DSS type environments, or environments where data are large enough that they cannot be maintained in the database memory. The extentsize is important in environments where DB2 tablespaces and containers reside upon RAID devices. In general, the EXTENTSIZE should always be equal to or a multiple of the RAID stripe size

By setting DB2_PARALLEL_IO, the tablespace PREFETCHSIZE takes on special meaning. PREFETCHSIZE is divided by the EXTENTSIZE to arrive at the degree of I/O parallelism. Without this environment variable set, the degree of I/O parallelism is normally derived from the number of containers. Because RAID often has only one container, it is important to set the PREFETCHSIZE as a multiple of the EXTENTSIZE, to provide a sufficient number of IO_SERVERS (at least one per physical disk), and to assign the tablespace to a BufferPool that is sufficiently large to accommodate to prefetch requests.

In the general case, we calculate EXTENTSIZE based on the physical attributes of the volume. PREFETCHSIZE should be at least EXTENTSIZE * the number of containers in order to obtain a good I/O parallelism. When dealing with RAID devices however, we may have only a single container within a tablespace and so the number of containers would be substituted with the number of devices or columns in the volume.

When using DMS device containers, such as Quick I/O files, the operating system does not perform any prefetching or caching. When you need to have a greater control over when and where memory is allocated to caching and prefetching of DB2 tablespace data, use Cached Quick I/O (See "Using VERITAS Cached Quick I/O" on page 89). If you prefer to assign more system memory permanently to DB2 BufferPools, set PREFETCHSIZE and the DB2_PARALLEL_IO settings for tablespaces.

Example

We have a VxVM RAID0 volume striped across 10 physical disks with a stripe column size of 64k. We have created a VxFS file system on this volume and are about to create a tablespace of DMS containers:

```
$ qiomkfile -s 1G /db2_stripe/cont001
$ db2 connect to PROD
$ db2 create tablespace DATA1 managed by database \
    using (device '/db2_stripe/cont001' 128000 ) \
    pagesize 8k extentsize 8 prefetchsize 80
$ db2 terminate
```

In this example, we ensure that each read of an extent will span 1 physical drive (column width is 64k and our extentsize is 8 * 8k pagesize). When prefetching, we take a full stripe read at a time (there are 10 disks in the stripe, so 10 * an extent is 80 pages). Observe that the PREFETCHSIZE remains a multiple of the EXTENTSIZE. These settings would provide a good environment for a database which in general uses clusters of data around 640k or less. For larger database objects or more aggressive prefetch on data, the specified PREFETCHSIZE can be multiplied.

If the database's main workload requires good sequential I/O performance, such as a DSS workload, then the settings for Cached Quick I/O and PREFETCHSIZE becomes even more important.

There are some cases where setting the PREFETCHSIZE to large values or having prefetching at all may degrade performance. In OLTP environments where data access is very random, you may need to turn off prefetching on a tablespace, or minimize the effect by setting PREFETCHSIZE equal to EXTENTSIZE. It is still very important in these types of environment to ensure that access to indexes is very fast and preferably all heavily accessed indexes are cached by Cached Quick I/O or in BufferPool memory.

INTRA_PARALLEL

The INTRA_PARALLEL setting is usually set on machines with multiple CPUs when large and complex queries are being executed. This may not provide any performance advantage in OLTP environments, as queries in these types of environments are normally very simple, short and highly repetitive. However, for DSS or OLAP environments, enabling this option may provide significant performance improvements.

NUM_IOCLEANERS

Specifies the number of async page cleaners. The cleaners flush dirty pages from the buffer pool, freeing the space for the threads pulling data in from storage. Important to tune this if the PREFETCH settings for the database are being modified. To avoid I/O wait, set this parameter higher if insert/update/delete is heavy or prefetch large.

NUM_IOSERVERS

Specifies the number of I/O servers for the database. These servers implement prefetch and async I/O operations. Should be set to at least the number of physical devices on the host system in order to maximize I/O parallelism.

CHNGPGS_THRESH

Specifies the threshold at which the IOCLEANERS start flushing dirty pages. A lower value indicates that cleaning should being earlier.

Other DB2 Tuning Considerations

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 are requested by DB2 are readily available in the system buffer cache and are simply copied to the DB2 buffer pool. 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, two methods are available to improve performance:

- Modify the DB2 PREFETCH setting to force reading in data before it is required.
- The second method is to enable Cached Quick I/O for the files that would be read by the DB2 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. A major advantage of using Cached Quick I/O is that it does not require any database level changes and so does not require the database to be restarted for changes to take effect.

Asynchronous I/O

Asynchronous I/O allows the DB2 database to schedule multiple I/Os without waiting for the I/O to complete. When the I/O completes, the kernel notifies the `DB2` using an interrupt.

Quick I/O supports kernel asynchronous I/O (KAIO), which reduces CPU utilization and improves transaction throughput. The DB2 database engine, by default, will make use of asynchronous I/O when using DMS containers.

Tuning Buffer Pools

The UNIX buffer cache plays an important role in performance when using UFS, HFS, or JFS in buffered I/O mode. However, when using Quick I/O, the database buffer pools must be tuned as if raw devices are being used. You can allocate more memory to the database buffer pools 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 pools. Adjusting the size and number of buffer pools requires restarting the database. Cached Quick I/O can be used to dynamically modify memory allocation to the database without requiring a restart. See "Using VERITAS Cached Quick I/O" on page 89 for more information on using Cached Quick I/O with your database.

Memory Allocation

Never configure DB2 to use more than 75% of the physical memory available on the system. DB2 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 DB2 simplifies the tuning and monitoring issues and ensures best performance.

TEMPORARY Tablespaces

When more than one TEMPORARY tablespace exists in the database, they will be used in round-robin fashion in order to balance their usage. See the Administration Guide for information on using more than one tablespace, rebalancing and recommended values for EXTENTSIZE, PREFETCHSIZE, OVERHEAD, and TRANSFERRATE.

Sizing DMS Containers

When you have more than one container in a DMS tablespace, it is important to ensure that all containers are the same physical, and logically declared, size. DB2 stripes data across available containers in a tablespace, writing in a round-robin fashion. If containers are not sized the same, then once the tablespace becomes sufficiently full, all I/O activity could be occurring to one physical file or device. This will incur a heavy performance penalty, especially when coupled with high values of the `NUM_IOCLEANERS`, `NUM_IOSERVERS` and `PREFETCHSIZE` configuration settings. When extending tablespace containers using the `qiomkfile` command, ensure that you maintain this equal length across all containers in a tablespace.

Separate Data, Indexes and Logs

It is always important to separate database data and log files. The write patterns for these types of object are very different and so mixing them on the same device will adversely affect performance. Log writes are always sequential and high bandwidth, whereas writes to data tablespaces can range from random to large and sequential. It is important to ensure that log writes are fast and do not suffer from device latency in order to provide the highest performing database environment.

When using SMS tablespaces, it is not possible to separate data and indexes onto different devices. This means that there is no way to reduce contention for I/O and memory between these two types of database object. However, when using DMS devices, it is possible to place the data and indexes of tables into different tablespaces. This can provide much improved performance in environments which have very heavy usage of indexes and/or constrained memory. In addition to being able to separate and therefore easily monitor I/O to the data and indexes, assigning indexes to a separate tablespace allows you to assign a dedicated bufferpool to the indexes or enable Cached Quick I/O on the index containers as required. This can greatly improve performance in environments where you want to ensure that indexes are always in memory and that there is no contention between data and indexes for a single bufferpools resources.

Update Database Statistics Regularly

The DB2 database maintains internal information and statistics about the physical layout of data in the database. These internal statistics are used by the prefetch and I/O scheduling threads to plan operations in advance and can therefore have a very large impact on performance. With regular database activity, the statistics can become incorrect and therefore begin to have an adverse affect on I/O planning. This is especially true after major loads of new data, creating indexes on tables and heavy table activity involving large numbers of delete or update queries.

DB2 provides several tools to assist in updating these statistics and therefore enable continued and accurate I/O planning. These tools can be run from the db2 command prompt and are called RUNSTATS, REORG and REORGCHK tools. They should be run regularly to ensure optimal database performance. For more information, see the System Catalog Statistics section in the *DB2 Administration Guide* and the section on CLP commands in the *DB2 Command Reference*.

Tuning Solaris

To achieve optimal performance using VERITAS Storage Foundation *for DB2*, 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.

The rest of this section describes the important tuning parameters that DB2 depends on for optimal performance. 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 DB2 *
set msgsys:msginfo_msgmax=65535
set msgsys:msginfo_msgmnb=65535
set msgsys:msginfo_msgseg=16384
set msgsys:msginfo_msgssz=16
set msgsys:msginfo_msgmap=258
set msgsys:msginfo_msgmni=256
set msgsys:msginfo_msgtql=512
*
set shmsys:shminfo_shmmax=134217728
set shmsys:shminfo_shmseg=16
set shmsys:shminfo_shmmni=300
*
set semsys:seminfo_semmni=256
set semsys:seminfo_semmap=258
set semsys:seminfo_semmns=512
set semsys:seminfo_semmnu=512
* end DB2 *
```

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. Refer to the book *Quick Beginnings for DB2 Connect Enterprise Edition* for your version of DB2 to obtain precise settings for your hardware configuration.

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 DB2 concurrently, you may need to increase this 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
   ```

msgmax

This parameter sets the maximum size (in bytes) of a single message. In general for Solaris systems, the value should not exceed 65535. See your DB2 documentation for the recommended value.

msgmnb

This parameter limits the total number of message bytes than can be on a single message queue at one time. The value should be some multiple of the *`msgmax`* setting. See your DB2 documentation for the recommended value.

msgseg

This parameter sets the number of *msgssz* segments available to all segments on all message queues. Memory for these segments is pre-allocated from kernel memory. See your DB2 documentation for the recommended value.

msgssz

This parameter sets the size of each message segment (in bytes) allocated for a message. The operating system allocates a total of *`msgseg * msgssz`* segments for use by application messages. The *`msgssz`* parameter is highly application dependent. See your DB2 documentation for the recommended value.

msgmap

This parameter sets the size of the message queue resource map. Each block of available, contiguous message segments requires one *msgmap* entry. See your DB2 documentation for the recommended value.

msgmni

This parameter sets the number of message queue identifiers available. Each message queue requires one message queue identifier and each identifier structure is approximately 144 bytes in size. See your DB2 documentation for the recommended value.

msgtql

This parameter sets the number of message queue headers available. Each queued message requires one message queue header and each header structure is approximately 12 bytes in size. See your DB2 documentation for the recommended value.

shmmax

This parameter sets the maximum size (in bytes) of a single shared memory segment. See your DB2 documentation for the recommended value.

shmmni

This parameter sets the number of shared memory identifiers. See your DB2 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 DB2 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 DB2 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 DB2 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 DB2 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 DB2 documentation for the recommended value of `semmnu`.

VERITAS Storage Foundation for DB2 Command Line Interface

A

VERITAS Storage Foundation *for DB2* provides a command line interface (CLI) to many key operations also supplied from within the VERITAS Storage Foundation GUI application. The command line interface lets you incorporate command operations into scripts and other administrative processes.

VERITAS Storage Foundation *for DB2* also provides commands specific to the Database FlashSnap feature. These commands can be executed using the CLI or the VERITAS Storage Foundation *for DB2* GUI. At this time, there are no VxDBA menu equivalents for Database FlashSnap operations.

Note The VERITAS Storage Foundation *for DB2* command line interface depends on certain tablespace and container information that is collected and stored in VxDBA's repository. Some CLI commands update the repository by default. It is also important to regularly ensure the VxDBA repository is up-to-date by using **Display/Update Tablespace Information** from either the **Database Administration** or **Display Database/VxDBA Information** VxDBA submenus, the update repository option from the GUI (right click on the database and select **Update Repository**), or the `db2ed_update` command.

Topics include:

- "Overview of Commands" on page 354
- "Examples of Using the Command Line Interface" on page 359

Overview of Commands

All VERITAS Storage Foundation *for DB2* commands supported in the command line interface are located in the `/opt/VRTS/bin` directory. Exceptions are `vxstorage_stats` and `edgetmsg2`, which are located in `/opt/VRTSdb2ed/bin`. Online manual pages are located in the `/opt/VRTS/man` directory. Follow the installation instructions provided in the *VERITAS Storage Foundation for DB2 Installation Guide* to ensure you can use these commands and view the online manual pages.

Note The commands ending with `_all` apply to multiple partitions of a partitioned DB2 database on the same host.

The following table summarizes the commands available to you from the command line:

VERITAS Storage Foundation for DB2 Commands

Command	Description
`db2ed_update` and `db2ed_update_all`	Creates or updates the VxDBA repository in VERITAS Storage Foundation *for DB2*. `db2ed_update_all` calls `db2ed_update` on every database partition for a partitioned DB2 database in a symmetric multiprocessing (SMP) environment. Performs the same operation from the command line, as right clicking on a database and selecting **Resync Repository** in the GUI.
`db2ed_checkconfig` and `db2ed_checkconfig_all`	Checks the configuration of a DB2 database in a VERITAS Storage Foundation *for DB2* environment. `db2ed_checkconfig_all` calls `db2ed_checkconfig` on every database partition for a partitioned DB2 database in an SMP environment. Performs the same operation from the command line, as **Display Database/VxDBA Information** > **Examine Volume/File System/Database Configuration** in the VxDBA utility menus. Performs the same operation from the command line, as right clicking on a database and selecting **Check System Configuration** in the GUI.

VERITAS Storage Foundation for DB2 Commands

Command	Description
`db2ed_saveconfig` and `db2ed_saveconfig_all`	Saves the configuration of a DB2 database in a VERITAS Storage Foundation *for DB2* environment. `db2ed_saveconfig_all` calls `db2ed_saveconfig` on every database partition for a partitioned DB2 database in an SMP environment. Performs the same operation from the command line, as **Display Database/VxDBA Information** > **Save Volume/File System/Database Configuration** in the VxDBA utility menus. Performs the same operation from the command line, as right clicking on a database and selecting **Save System Configuration** in the GUI.
`db2ed_ckptcreate and db2ed_ckptcreate_all`	Creates a Storage Checkpoint for a DB2 database. `db2ed_ckptcreate_all` creates a Version Checkpoint for a partitioned DB2 database. `db2ed_ckptcreate_all` calls `db2ed_ckptcreate` on every database partition. Performs the same operation from the command line, as **Storage Checkpoints** > **Create Storage Checkpoint** in the GUI.
`db2ed_ckptdisplay and db2ed_ckptdisplay_all`	Displays Storage Checkpoints for a DB2 database. `db2ed_ckptdisplay_all` displays Version Checkpoints for the current partitioned DB2 database. Performs the same operation from the command line, as **Storage Checkpoint** > **Properties** in the GUI. (Clicking on **Storage Checkpoint** alone lists all Storage Checkpoints.)
`db2ed_ckptmount and db2ed_ckptmount_all`	Mounts a Storage Checkpoint for a DB2 database. `db2ed_ckptmount_all` mounts all Storage Checkpoints for a Version Checkpoint for a partitioned DB2 database. Performs the same operation from the command line, as **Storage Checkpoint** > **Mount Storage Checkpoint** in the GUI.
`db2ed_ckptpolicy`	Creates and administers Storage Checkpoint allocation policies for a Multi-Volume File System (MVS). You can display, create, update, assign, and remove Storage Checkpoint allocation policies using this command. This option is not available through the VxDBA utility menu or the GUI.
`db2ed_ckptquota`	Administers quotas for Storage Checkpoints. This option is not available through the VxDBA utility menu or the GUI.

VERITAS Storage Foundation for DB2 Commands

Command	**Description**
`db2ed_ckptremove` and `db2ed_ckptremove_all`	Removes a Storage Checkpoint for a DB2 database. `db2ed_ckptremove_all` removes a Version Checkpoint for a partitioned DB2 database. Performs the same operation from the command line, as **Storage Checkpoint** > **Remove Storage Checkpoint** in the GUI.
`db2ed_ckptrollback` and `db2ed_ckptrollback_all`	Rolls back a DB2 database to a Storage Checkpoint point-in-time image. `db2ed_ckptrollback_all` rolls back a a partitioned DB2 database to a Version Checkpoint. The `db2ed_ckptrollback_all` command calls `db2ed_ckptrollback` on every partition. Performs the same operation from the command line, as **Storage Checkpoint** > **Rollback Storage Checkpoint** in the GUI.
`db2ed_ckptumount` and `db2ed_ckptumount_all`	Unmounts a Storage Checkpoint for a DB2 database. `db2ed_ckptumount_all` unmounts a Version Checkpoint for a partitioned DB2 database. Performs the same operation from the command line, as **Storage Checkpoint** > **Unmount Storage Checkpoint** in the GUI.
`db2ed_clonedb`	Creates a copy of a DB2 database by cloning all existing database containers. This cloned database can only be started on the same host as the existing database. Performs the same operation from the command line, as **DB2** > **Create Clone Database** in the GUI.
`qio_convertdbfiles`	Converts DB2 container files to Quick I/O files. Performs the same operation from the command line, as **Container** > **Conversion** in the GUI.
`qio_getdbfiles`	Extracts information on files used by the database and stores the names of these files in `mkqio.dat`. This command is not available through the GUI.
`qio_recreate`	Automatically recreates Quick I/O files when the database is recovered. The command expects to find a mkqio.dat file in the directory where the `qio_recreate` command is run. This command is not available through the GUI.

VERITAS Storage Foundation for DB2 Commands

Command	Description
`db2ed_vmchecksnap`	Creates and validates a snapplan that the `db2ed_vmsnap` command uses to create a volume snapshot of a DB2 database. The snapplan specifies snapshot scenarios (such as `online_snapshot`, `online_mirror`, or `offline`). The command can also be used to validate, list, copy, and remove a snapplan. This option is not available through the VxDBA utility menu. Performs the same operation from the command line, as the options within **Snapplans > Create Snapplan...** and **Snapplan > Modify/Validate Snapplan** in the GUI.
`db2ed_vmsnap`	Creates a snapshot image of a DB2 database by splitting the mirror volumes used by the database. You can also use this command to resynchronize the snapshot image back to the current database. The command also allows you to reverse resynchronize a snapshot image of an DB2 database. This option is not available through the VxDBA utility menu. Performs the same operation from the command line, as the options within **Snapplan > Create Snapshot** in the GUI.
`db2ed_vmclonedb`	Mounts the file systems on the snapshot volumes and starts a clone database from snapshot volumes. You can also use this command to shut down or restart the clone database, unmount the file systems, or deport the clone database's volumes. This option is not available through the VxDBA utility menu. Performs the same operation from the command line, as the options within **DB2 instances > Create Snapshot Database...** in the GUI.
`edgetmsg2`	Manages message log files. You can use this utility to display and list message log files. You can also use this utility to write a message to a log file or to the console, or read the log file and print to the console. This option is not available through the VxDBA utility menu or through the GUI.
`vxstorage_stats`	Displays storage object I/O statistics. This option is not available through the VxDBA utility menu. Performs the same operation from the command line, as the options within **Containers > Topology Statistics...** in the GUI.

Command Support

All commands ending with `_all` are supported in an SMP environment. These commands apply to multiple partitions.

The following commands *are not supported* in an SMP environment:

- `db2ed_clonedb`
- `db2ed_ckptpolicy`
- `db2ed_ckptquota`
- `db2ed_mon`
- `db2ed_vmchecksnap`
- `db2ed_vmsnap`
- `db2ed_vmclonedb`
- VxDBA menu utility `(db2ed_vxdba)`
- `qio_getdbfiles`
- `qio_convertdbfiles`
- `qio_recreate`
- `vxstorage_stats`

The following commands apply at the partition level:

- `qio_getdbfiles`
- `qio_convertdbfiles`
- `qio_recreate`

The following command applies at the host level:

- `edgetmsg2`
- `vxstorage_stats`

Examples of Using the Command Line Interface

This section provides examples for using the VERITAS Storage Foundation *for DB2* command line interface to perform administrative operations. For more detailed information about the commands and their syntax and available options, see the individual manual pages.

Creating or Updating VxDBA's Repository Using db2ed_update

You can use the VERITAS Storage Foundation *for DB2* `db2ed_update` command to create or update the repository for VxDBA.

Any time you change the structure of the database (for example, by adding or deleting containers), you must run `db2ed_update`.

Prerequisites

- You must be logged on as the instance owner (typically, the user ID `db2inst1`).

Usage Notes

- The `db2ed_update` command creates a repository in the `/etc/vx/vxdba/DB2.`*`$DB2INSTANCE.$DB2DATABASE/NODE0000`* directory where information used by VERITAS Storage Foundation *for DB2* is kept. If the repository already exists, the command will refresh the information.
- The database must be up and running, and the `$DB2DATABASE` variable argument must be specified with the `-D` option. The `$DB2INSTANCE` argument is optional.
- See the `db2ed_update`(1M) manual page for more information.

Note For multiple partitions, use the `db2ed_update_all` command. `db2ed_update_all` calls `db2ed_update` on every database partition for a partitioned DB2 database.

▼ To update the VxDBA repository

Use the `db2ed_update` command as follows:

```
$ /opt/VRTS/bin/db2ed_update -D PROD
```

▼ To view the status of the VxDBA repository

Use the `db2ed_update` command with the `-n` option as follows:

```
$ /opt/VRTS/bin/db2ed_update -D PROD -n
VxDBA repository is up to date.
```

Checking DB2 Configuration Environment Using db2ed_checkconfig

You can use the VERITAS Storage Foundation *for DB2* `db2ed_checkconfig` command to verify and report on a DB2 environment from the command line.

Prerequisites

- ◆ You must be logged on as the instance owner (typically, the user ID `db2inst1`).

Usage Notes

- ◆ The `db2ed_checkconfig` command is used to verify various elements of the DB2 database environment. The utility attempts to use certain basic rules on DB2 settings, file system and volume parameters and layout options to verify how resilient and well configured a configuration is. The information provided is valid for the supplied DB2 database.
- ◆ See the `db2ed_checkconfig`(1M) and `db2ed_checkconfig_all`(1M) manual pages for more information.

Note For multiple partitions, use the `db2ed_checkconfig_all` command. `db2ed_checkconfig_all` checks the configuration for a partitioned DB2 database by calling `db2ed_checkconfig` on every database partition.

▼ To check the DB2 configuration environment

Use the `db2ed_checkconfig` command as follows:

```
$ /opt/VRTS/bin/db2ed_checkconfig -D PROD
Examining File System and DB2 Container attributes.

Total of 13 containers over 2 file systems.
All file systems are VxFS.
Examining Quick I/O settings.
Examining Cached Quick I/O settings.
No file systems have Cached Quick I/O enabled.
SFDB2 db2ed_checkconfig WARNING V-81-3508: The VxFS layout of
/udb_home/db2inst1/prod_temp is not
version 4.
SFDB2 db2ed_checkconfig WARNING V-81-3508: The VxFS layout of
/udb_home is not version 4.

VxFS file systems not in the version 4 layout do
not support all of the features of VxFS.
```

```
The database has:
  5 SMS Containers
  3 DMS File Containers
  0 DMS Device Containers

13 containers are not configured to use Quick I/O.

Examining DB2 container fragmentation.


0 SMS containers skipped fragmentation check.
11 files are fragmented.

Examining File System tunable settings.

Parameters for all VxFS file systems used by PROD.
Filesystem i/o parameters for /udb_home/db2inst1/prod_temp
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 = 1048576
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_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 = 31775
fcl_keeptime = 0
fcl_winterval = 3600
Filesystem i/o parameters for /udb_home
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 = 1048576
discovered_direct_iosz = 262144
max_direct_iosz = 1048576
default_indir_size = 8192
qio_cache_enable = 0
write_throttle = 0
max_diskq = 1048576
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 = 222425
fcl_keeptime = 0
fcl_winterval = 3600

Examining DB2 Volume layout and attributes.

Data for database PROD is contained in one volume group.

SFDB2 db2ed_checkconfig WARNING V-81-3419: File system
/udb_home/db2inst1/prod_temp is not
mirrored using VxVM.

SFDB2 db2ed_checkconfig WARNING V-81-3419: File system /udb_home is
not mirrored using VxVM.

Examining Volume Group versions.
Volume Group version for db2dg is:
110


See the vxdg(1M) man page for more information on Volume Group
versions.

Examining DB2 internal information.

DB2 Version is 8.1.

Examining DB2 logging mode.
```

```
The database has transaction logs in directory
/udb_home/db2inst1/prod_re.
SFDB2 db2ed_checkconfig WARNING V-81-3322: Transaction log
directory
is not mirrored using VxVM.

The database is running in archivelog mode.

SFDB2 db2ed_checkconfig WARNING V-81-3322: The database has a
user-exit defined for archiving logs. Cannot determine where logs
are archived to.

Examining DB2 Database Free Space.

DB20000I  The SQL command completed successfully.

Name                                 = SYSCATSPACE
Type                                 = System managed space
Total pages                          = 1745
Total pages                          = 1745
Used pages                           = 1745
Free pages                           = Not applicable
Page size (bytes)                    = 4096

Name                                 = USER1
Type                                 = Database managed space
Total pages                          = 12800
Used pages                           = 160
Free pages                           = 12608
Page size (bytes)                    = 4096

Name                                 = INDX1
Type                                 = Database managed space
Total pages                          = 12800
Used pages                           = 96
Free pages                           = 12672
Free pages                           = 12672
Page size (bytes)                    = 4096

Name                                 = TEMPORARY1
Type                                 = System managed space
Total pages                          = 1
Used pages                           = 1
Free pages                           = Not applicable
Page size (bytes)                    = 4096
```

Saving the DB2 Configuration Environment Using db2ed_saveconfig

You can use the VERITAS Storage Foundation *for DB2* `db2ed_saveconfig` command to save configuration information on DB2, VERITAS products, and system hardware from the command line.

Prerequisites

◆ You must be logged on as the instance owner (typically, the user ID `db2inst1`).

Usage Notes

◆ The `db2ed_saveconfig` command is used to collect and record configuration information on DB2, VERITAS products, and system hardware. Information is gathered in the context of a specified DB2 database. The utility attempts to gather enough information to allow an administrator to reconstruct a system and database from scratch, in the case of a complete system failure.

  Information collected is in the form of many system configuration files and the results of querying the system hardware, VERITAS products, and DB2. The location where configuration information has been saved is displayed as output from the `db2ed_saveconfig` command. Alternatively, you can use the `-l` option to designate this location.

◆ See the `db2ed_saveconfig`(1M) and `db2ed_saveconfig_all`(1M) manual pages for more information.

Note For multiple partitions, use the `db2ed_saveconfig_all` command. `db2ed_saveconfig_all` saves the configuration for a partitioned DB2 database by calling `db2ed_saveconfig` on every database partition.

▼ To save the DB2 configuration environment

Use the `db2ed_saveconfig` command as follows:

```
$ /opt/VRTS/bin/db2ed_saveconfig -D PROD
System configuration information saved to directory:
/tmp/vxdba.DR.1148
```

Creating Storage Checkpoints Using db2ed_ckptcreate

You can use the VERITAS Storage Foundation *for DB2* `db2ed_ckptcreate` command to create a Storage Checkpoint from the command line. Storage Checkpoints can be either online or offline. If `online` is specified, the database is put into write suspend mode when the Storage Checkpoint is created. If `offline` is specified, the database is expected to be down.

Prerequisites

- You must be logged on as the instance owner (typically, the user ID `db2inst1`).
- For best recoverability, always keep the LOGRETAIN and/or USEREXIT database configuration parameters enabled when you create Storage Checkpoints.

Usage Notes

- `db2ed_ckptcreate` stores Storage Checkpoint information under the following directory:

`/etc/vx/vxdba/DB2.`*`$DB2INSTANCE.$DB2DATABASE/NODEnum/`*`checkpoint_dir`

- See the `db2ed_ckptcreate`(1M) manual page for more information.

Note For multiple partitions, use the `db2ed_ckptcreate_all` command. `db2ed_ckptcreate_all` creates a Version Checkpoint for a partitioned DB2 database by calling `db2ed_ckptcreate` on every database partition.

▼ To create Storage Checkpoints while the database is online

Use the `db2ed_ckptcreate` command as follows:

```
 $ /opt/VRTS/bin/db2ed_ckptcreate -D PROD
-o online
 An online Storage Checkpoint Checkpoint_971672042 is created at GMT
 2004-05-01-18.22.34.0000.
```

▼ To create Storage Checkpoints without updating the VxDBA repository while the database is online

Use the `db2ed_ckptcreate` command as follows:

```
 $ /opt/VRTS/bin/db2ed_ckptcreate -D PROD  -o online -n
 An online Storage Checkpoint Checkpoint_971672043 is created at GMT
 2004-05-01-18.22.34.0000.
```

▼ To create Storage Checkpoints while the database is offline

1. Terminate the database before creating the offline Storage Checkpoint:

   ```
   db2 terminate
   ```

2. Use the `db2ed_ckptcreate` command as follows:

   ```
   $ /opt/VRTS/bin/db2ed_ckptcreate -D PROD -o offline
   An offline Storage Checkpoint Checkpoint_971672044 is created at
   GMT
   2004-05-01-18.22.34.0000.
   ```

Note The default option is `online`.

▼ To assign a Storage Checkpoint allocation policy to a Storage Checkpoint

Use the `db2ed_ckptcreate` command as follows:

```
  $ /opt/VRTS/bin/db2ed_ckptcreate -D PROD  -o online -p
ckpt_data,ckpt_metadata
  Creating online Storage Checkpoint of database PROD.
  Storage Checkpoint Checkpoint_971672044 created.
```

▼ To create Storage Checkpoints on multiple partitions

As the DB2 instance owner, use the `db2ed_ckptcreate_all` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptcreate_all -I db2inst -D PROD
Creating Version Checkpoint Version_ckpt_1088639094 on all
partitions.
rah: omitting logical node 0

Creating checkpoint on partition 1.
An online Storage Checkpoint Checkpoint_1088639125 is created
at GMT 2004-06-30-23.45.29.0000
db2ed_ckptcreate -I ... completed ok

Creating checkpoint on partition 0.
An online Storage Checkpoint Checkpoint_1088639159 is created
at GMT 2004-06-30-23.46.03.0000
db2ed_ckptcreate -I ... completed ok
```

where `db2inst` is the instance name.

Note The command output will contain “`rah: omitting logical node 0`”. This is the normal behavior.

Displaying Storage Checkpoints Using db2ed_ckptdisplay

You can use the VERITAS Storage Foundation *for DB2* `db2ed_ckptdisplay` command to display the Storage Checkpoints associated with a DB2 instance from the command line. You can also use it to display fileset quota values.

Prerequisites

- You may be logged in as either the instance owner or `root`. If you execute the command as `root`, you must set up `$DB2INSTANCE` as the instance owner.

Usage Notes

- In addition to displaying the Storage Checkpoints created by VxDBA, `db2ed_ckptdisplay` also displays other Storage Checkpoints (for example, Storage Checkpoints created by the NetBackup).
- See the `db2ed_ckptdisplay`(1M) manual page for more information.
- The **Status** field identifies if the Storage Checkpoint is partial (P), complete (C), invalid (I), mounted (M), read-only (R), or writable (W).

Note For multiple partitions, use the `db2ed_ckptdisplay_all` command. `db2ed_ckptdisplay_all` displays Version Checkpoints associated with the current partitioned DB2 database.

▼ To display Storage Checkpoints created by VERITAS Storage Foundation for DB2

Use the `db2ed_ckptdisplay` command as follows to display information for Storage Checkpoints created by VERITAS Storage Foundation *for DB2*:

```
$ /opt/VRTS/bin/db2ed_ckptdisplay -D PROD
Checkpoint_975876659            Sun Apr 3 12:50:59 2004    P+R
Checkpoint_974424522_wr001      Thu May 16 17:28:42 2004   C+R
Checkpoint_974424522            Thu May 16 17:28:42 2004   P+R
```

▼ To display other Storage Checkpoints

Use the `db2ed_ckptdisplay` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptdisplay -D PROD  -o other
NetBackup_incr_PROD_955133480    NBU      /db01
NetBackup_full_PROD_955132952    NBU      /db01
```

▼ To display other Storage Checkpoints without updating the VxDBA repository

Use the `db2ed_ckptdisplay` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptdisplay -D PROD  -o other -n
NetBackup_incr_PROD_955133480    NBU      /db01
NetBackup_full_PROD_955132952    NBU      /db01
```

▼ To display all Storage Checkpoints

Use the `db2ed_ckptdisplay` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptdisplay -D PROD  -o all
Checkpoint_971672042             Sun May 15 13:55:53 2004   C+R
Checkpoint_903937870             Fri May 13 22:51:10 2004   C+R
Checkpoint_901426272             Wed May 11 16:17:52 2004   P+R
NetBackup_incr_PROD_955133480    NBU      /db01
NetBackup_full_PROD_955132952    NBU      /db01
```

▼ To display all Storage Checkpoints without updating the VxDBA repository

Use the `db2ed_ckptdisplay` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptdisplay -D PROD  -o all -n
Checkpoint_971672042             Sun May 15 13:55:53 2004   C+R
Checkpoint_903937870             Fri May 13 22:51:10 2004   C+R
Checkpoint_901426272             Wed May 11 16:17:52 2004   P+R
NetBackup_incr_PROD_955133480    NBU      /db01
NetBackup_full_PROD_955132952    NBU      /db01
```

▼ To display fileset quota values

Use the `db2ed_ckptdisplay` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptdisplay -D PROD -c Checkpoint_903937870 -Q
Checkpoint_975876659              Wed Mar 19 9:12:20 2003      C+R
 Filesystem                        HardLim   SoftLim    CurrentUse
 /udb_home/db2inst1/indx1_1       100000     50000          2028
 /udb_home/db2inst1/indx1_1       100000     50000          2028
 /udb_home/db2inst1/temp          150000     80000          2142
 /udb_home/db2inst1/system1       150000     70000          3092
```

Scheduling Storage Checkpoints Using db2ed_ckptcreate and cron

You can use the VERITAS Storage Foundation *for DB2* `db2ed_ckptcreate` command to schedule Storage Checkpoint creation in a `cron` job or other administrative script.

Prerequisites

- You must be logged on as the instance owner (typically, the user ID `db2inst1`).

Usage Notes

- Create a new `crontab` file or edit an existing `crontab` file to include a Storage Checkpoint creation entry with the following space-delimited fields:

  ```
  minute hour day_of_month month_of_year day_of_week \
    /opt/VRTS/bin/db2ed_ckptcreate
  ```

  where:

  - *minute* - numeric values from 0-59 or *
  - *hour* - numeric values from 0-23 or *
  - *day_of_month* - numeric values from 1-31 or *
  - *month_of_year* - numeric values from 1-12 or *
  - *day_of_week* - numeric values from 0-6, with 0=Sunday or *

  Each of these variables can either be an asterisk (meaning all legal values) or a list of elements separated by commas. An element is either a number or two numbers separated by a hyphen (meaning an inclusive range).

- See the `db2ed_ckptcreate`(1M), `cron`(1M), and `crontab`(1) manual pages for more information.

Scheduling Storage Checkpoint Creation in a cron Job

- To create a Storage Checkpoint twice a day, at 5:00 a.m. and 7:00 p.m., every Monday through Friday, include the following entry in your `crontab` file:

```
0 5,19 * * 1-5 /opt/VRTS/bin/db2ed_ckptcreate -D PROD \
  -I db2inst1 -o online
```

- To create a Storage Checkpoint at 11:30 p.m., on the 1st and 15th day of each month, include the following entry in your `crontab` file:

```
30 23 1,15 * * /opt/VRTS/bin/db2ed_ckptcreate -D PROD \
  -I db2inst1 -o online
```

- To create a Storage Checkpoint at 1:00 a.m. every Sunday while the database is offline, include the following entry in your `crontab` file:

```
0 1 * * 0 /opt/VRTS/bin/db2ed_ckptcreate -D PROD \
  -I db2inst1 -o offline
```

Mounting Storage Checkpoints Using db2ed_ckptmount

You can use the VERITAS Storage Foundation *for DB2* `db2ed_ckptmount` command to mount a Storage Checkpoint for a DB2 database from the command line.

Prerequisites

- You may be logged in as either the instance owner or `root`. If you execute the command as `root`, you must set up `$DB2INSTANCE` as the instance owner.

Usage Notes

- The `db2ed_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.
- The DB2 instance owner must have permission to mount the Storage Checkpoint on the mount point.
- 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, `_wr`*`xxx`*, where *`xxx`* is an integer, will be appended to the Storage Checkpoint name.
- If the specified mount point directory does not exist, then `db2ed_ckptmount` creates it before mounting the Storage Checkpoint, as long as the DB2 instance owner has permission to create it.
- For multiple partitions, use the `db2ed_ckptmount_all` command. `db2ed_ckptmount_all` mounts all Storage Checkpoints for a Version Checkpoint for a partitioned DB2 database.
- See the `db2ed_ckptmount` and `db2ed_ckptmount_all`(1M) manual pages for more information.

▼ To mount Storage Checkpoints with the read/write option

Use the `db2ed_ckptmount` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptmount -D PROD -c Checkpoint_971672042 \
  -m /tmp/ckpt_rw -o rw
Creating Storage Checkpoint on \
/tmp/ckpt_rw/udb_home/udb01a_home \
with name Checkpoint_971672042_wr001
```

▼ To mount Storage Checkpoints with the read-only option

Use the `db2ed_ckptmount` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptmount -D PROD -c Checkpoint_971672042 \
  -m /tmp/ckpt_ro -o ro
```

Unmounting Storage Checkpoints Using db2ed_ckptumount

You can use the VERITAS Storage Foundation *for DB2* `db2ed_ckptumount` command to unmount a Storage Checkpoint for a DB2 instance from the command line.

Prerequisites

- You may be logged in as either the instance owner or `root`. If you execute the command as `root`, you must set up `$DB2INSTANCE` as the instance owner.

Usage Notes

- The `db2ed_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 `db2ed_ckptumount`(1M) manual page for more information.

Note For multiple partitions, use the `db2ed_ckptumount_all` command.

▼ To unmount Storage Checkpoints

Use the `db2ed_ckptumount` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptumount -D PROD \
  -c Checkpoint_971672042_wr001
```

Creating and Working with Storage Checkpoint Allocation Policies Using db2ed_ckptpolicy

You can use the VERITAS Storage Foundation *for DB2* `db2ed_ckptpolicy` command to create and administer Storage Checkpoint allocation policies for Multi-Volume File Systems (MVSs). Storage Checkpoint allocation policies specify a list of volumes and the order in which to allocate data to them.

Prerequisites

- You must be logged on as the instance owner (typically, the user ID `db2inst1`).

Usage Notes

- All volumes must be MVSs.
- The `db2ed_ckptpolicy` command can be used only on file systems using disk layout Version 6.
- The VxVM volume set and VxFS Multi-Volume File System features must be enabled to use Storage Checkpoint allocation policies.
- The status of a Storage Checkpoint allocation policy is either `partial` or `complete`. A *partial policy* is one that does not exist on all file systems used by the database. A *complete policy* is one that exists on all file systems.
- After an allocation policy is assigned to a Storage Checkpoint, the allocation mechanism attempts to satisfy requests from each device in the order specified in the allocation policy. If the request cannot be satisfied from any of the devices in the allocation policy, the request will fail, even if other devices that have space exist in the file system. Only devices listed in the policy can be allocated.
- See the `db2ed_ckptpolicy`(1M) manual page for more information.

▼ To create a Storage Checkpoint allocation policy

Use the `db2ed_ckptpolicy` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptpolicy -D DB2DATABASE \
  -o create -p ckpt_policy
```

Output similar to the following is displayed. This example assumes the following:

- Two MVS file systems `/mvsfs/v1` and `/mvsfs/v2` are used for datafiles.
- File system `/mvsfs/v1` is created on volume set mvsvset1.
- File system `/mvsfs/v2` is created on volume set mvsvset2.
- Volume set `mvsvset1` contains volumes `mvsv1`, `mvsv2`, and `mvsv3`.

- ◆ Volume set mvsvset2 contains volumes mvsv4 and mvsv5.

```
File System: /mvsfs/v2 (MVS volumes: mvsv4,mvsv5)
Assigned Data Policy: NONE (MVS Volumes: N/A)
Assigned Meta Data Policy: NONE (MVS Volumes: N/A)
Please enter the volume name(s), sperated by space, for the policy
ckpt_policy [skip,quit]: mvsv4

File System: /mvsfs/v1 (MVS volumes: mvsv1,mvsv2,mvsv3)
Assigned Data Policy: NONE (MVS Volumes: N/A)
Assigned Meta Data Policy: NONE (MVS Volumes: N/A)
Please enter the volume name(s), separated by space, for the policy
ckpt_policy [skip,quit]: mvsv2

The following information will be used to create policy ckpt_sample
ckpt_sample             /mvsfs/v2                         mvsv4
ckpt_sample             /mvsfs/v1                         mvsv2
```

▼ To display Storage Checkpoint allocation policy within the database

Use the db2ed_ckptpolicy command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptpolicy -D DB2DATABASE \
  -n -o display [-c storage_ckpt | -p ckpt_policy]
```

If -p *ckpt_policy* and -c *storage_ckpt* options are not specified, output similar to the following is displayed:

```
Policy Name          File System Coverage
-------------------- --------------------
 ckpt                Complete
 ckpt_data           Complete
 ckpt_metadata       Complete
 new_ckpt            Partial
 ckpt_sample         Complete
```

If -p *ckpt_policy* option is specified, output similar to the following is displayed:

```
Policy Name      File System  MVS volumes
---------------- ------------ -----------
ckpt_sample      /mvsfs/v2    mvsv4
ckpt_sample      /mvsfs/v1    mvsv2
```

If the -c *storage_ckpt* option is specified, output similar to the following is displayed:

```
Storage Checkpoint     File System  Data Policy  Meta Data Policy
---------------------- ------------ -----------  ----------------
Checkpoint_1095125037 /mvsfs/v2    ckpt_data    ckpt_metadata
Checkpoint_1095125037 /mvsfs/v1    ckpt_data    ckpt_metadata
```

▼ To update a Storage Checkpoint allocation policy

Use the `db2ed_ckptpolicy` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptpolicy -D DB2DATABASE \
  -n -o update -p ckpt_policy
```

Output similar to the following is displayed:

```
File System: /mvsfs/v2 (MVS volumes: mvsv4,mvsv5)
Policy: ckpt_sample (MVS volumes: mvsv4)
Please enter the volume name(s), separated by space, for the policy
ckpt_sample [skip,quit]: mvsv5

File System: /mvsfs/v1 (MVS volumes: mvsv1,mvsv2,mvsv3)
Policy: ckpt_sample (MVS volumes: mvsv2)
Please enter the volume name(s), separated by space, for the policy
ckpt_sample [skip,quit]: mvsv2,mvsv3

The following information will be used to create policy ckpt_sample
ckpt_sample            /mvsfs/v2                            mvsv5
ckpt_sample           /mvsfs/v1                           mvsv2,mvsv3
```

▼ To assign a Storage Checkpoint allocation policy

Use the `db2ed_ckptpolicy` command as follows to assign an allocation policy to a specified Storage Checkpoint:

```
$ /opt/VRTS/bin/db2ed_ckptpolicy -D DB2DATABASE \
  -n -o assign -c ckpt_name -p ckpt_policy[,ckpt_metadata_policy]
```

▼ To remove a Storage Checkpoint allocation policy

Use the `db2ed_ckptpolicy` command as follows to remove an allocation policy from a specified Storage Checkpoint:

```
$ /opt/VRTS/bin/db2ed_ckptpolicy -D DB2DATABASE \
  -n -o remove -p ckpt_policy
```

Administering Storage Checkpoint Quotas db2ed_ckptquota

You can use the VERITAS Storage Foundation *for DB2* `db2ed_ckptquota` command to administer file system quotas for Storage Checkpoint for a DB2 database from the command line.

Prerequisites

- ◆ You must be logged on as the instance owner (typically, the user ID `db2inst1`).
- ◆ The VxDBA repository entry for the database must exist and the DBA must be the owner of all file systems to be affected.

Usage Notes

- ◆ See the `db2ed_ckptquota`(1M) manual page for more information.

▼ To set quota limits for all file systems in the database and enable quota enforcement

Use the `db2ed_ckptquota` command as follows to set the hard and soft limits for all file systems in the database and enable quota enforcement:

```
$ /opt/VRTS/bin/db2ed_ckptquota -D DB2DATABASE \
-o set=hardlimit,softlimit,enable
```

▼ To set quota limits for all file systems specified in a list file

Use the `db2ed_ckptquota` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptquota -D DB2DATABASE \
-o set=hardlimit,softlimit -f listfile
```

▼ To disable quota limits for a file system

Use the `db2ed_ckptquota` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptquota -D DB2DATABASE -o disable \
filesystem_path
```

▼ To display quota values for all file systems in the database

Use the `db2ed_ckptquota` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptquota -D DB2DATABASE -o display
```

Output similar to the following is displayed:

```
Filesystem                  Hardlimit     Softlimit     CurrentUse
/db2/udb_home               50000         40000         136
/db2/testvol01              25000         20000         128
/db2/testvol02              50000         40000         128
/db2/testvol03              50000         40000         0
/db2/testvol04              25000         20000         128
/db2/testvol05              50000         40000         128
```

Note `CurrentUse` displays the number of filesystem blocks currently used by all Storage Checkpoints in the filesystem. If there are no Storage Checkpoints, or if quotas have been disabled, `CurrentUse` will display 0.

Performing Storage Rollback Using db2ed_ckptrollback

You can use the VERITAS Storage Foundation *for DB2* `db2ed_ckptrollback` command to rollback a DB2 instance to a Storage Checkpoint.

Prerequisites

- You may be logged in as either the instance owner.

Usage Notes

- The `db2ed_ckptrollback` rolls a DB2 database back to a specified Storage Checkpoint. You can perform a Storage Rollback for the entire database.
- Database rollback for the entire database requires that the database be inactive before Storage Rollback commences. The `db2ed_ckptrollback` command will not commence if the DB2 database is active. See the `db2ed_ckptrollback`(1M) manual page for more information.

Note For multiple partitions, use the `db2ed_ckptrollback_all` command. `db2ed_ckptrollback_all` rolls back a partitioned DB2 database to a Version Checkpoint. The `db2ed_ckptrollback_all` command calls `db2ed_ckptrollback` on every partition.

▼ To roll back a DB2 instance to a Storage Checkpoint

Use the `db2ed_ckptrollback` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptrollback -D PROD  \
  -c Checkpoint_903937870
```

Removing Storage Checkpoints Using db2ed_ckptremove

You can use the VERITAS Storage Foundation *for DB2* `db2ed_ckptremove` command to remove a Storage Checkpoint for a DB2 instance at the command line.

Prerequisites

- You may be logged in as either the instance owner or `root`. If you execute the command as `root`, you must set up `$DB2INSTANCE` as the instance owner.

Usage Notes

- The `db2ed_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 VxDBA GUI, or the `db2ed_ckptcreate`(1M) command.
- See the `db2ed_ckptremove`(1M) manual page for more information.
- You must unmount the Storage Checkpoint before you can remove it.

Note For multiple partitions, use the `db2ed_ckptremove_all` command. `db2ed_ckptremove_all` removes a Version Checkpoint for a partitioned DB2 database.

▼ To remove Storage Checkpoints

Use the `db2ed_ckptremove` command as follows:

```
$ /opt/VRTS/bin/db2ed_ckptremove -D PROD \
  -c Checkpoint_971672042_wr001
```

Cloning the DB2 Database Using db2ed_clonedb

You can use the VERITAS Storage Foundation *for DB2* `db2ed_clonedb` command to clone a DB2 database using a Storage Checkpoint. Cloning an existing database using a Storage Checkpoint must be done on the same host.

You have the option to manually or automatically recover the DB2 database when using the `db2ed_clonedb` command:

- Manual (Interactive) recovery, which requires using the `-i` option, of the clone instance allows the user to control the degree of recovery by specifying which archive log files are to be replayed.
- Automatic (Non-interactive) recovery recovers the entire database and replays all of the archive logs. You will not be prompted for any archive log names.

Prerequisites

- Before using `db2ed_clonedb` to clone a database either within the same instance or across instances, ensure that your locale is the same as the database locale. If you do not know what locale to set, refer to the file `/opt/VRTSdb2ed/lib/DB2CODPAGE.tbl` for a mapping between the locale and the database Code Page.
- Make sure you have enough space and system resources to create a clone instance on your system.

  A clone database takes up as much memory as the primary database.
- You must first create a Storage Checkpoint. (See "Creating Storage Checkpoints Using db2ed_ckptcreate" on page 366.)
- If you choose to use an existing Storage Checkpoint to create the clone database, the Storage Checkpoint needs to be online.

Usage Notes

- The `db2ed_clonedb` command is used to create a copy of a DB2 database, cloning all existing database files to new locations.
- For DB2 8.1 and earlier versions, `db2ed_clonedb` works when the instance is either up or down. For DB2 8.2 and later, `db2ed_clonedb` only works when the instance is up.
- It is assumed that the user has a basic understanding of the DB2 recovery process.
- See the `db2ed_clonedb`(1M) manual page for more information.

Note When cloning a database using `db2ed_clonedb`, any database within the target instance that has the same name as the source database to be cloned will be temporarily uncataloged and therefore unavailable. If the source database is being cloned within the same instance, it will be temporarily uncataloged while `db2ed_clonedb` is running. The database will be recataloged on completion of `db2ed_clonedb`.

Options

`-I` *`SOURCE_INSTANCE`*	Specifies the source DB2 database. If the instance is not specified here, it is set as the current user.
`-S` *`SOURCE_DATABASE`*	Specifies the name of the source DB2 database.
`-T` *`TARGET_DATABASE`*	Specifies the name of the new DB2 database that will be created.
`-c` *`CKPT_NAME`*	Indicates the name of the Storage Checkpoint to use for creating the new database. The Storage Checkpoint is mounted automatically during the cloning process.
`-m` *`MOUNT_POINT`*	Indicates the location of the database containers to be mounted.
`-l`	Requires the argument *`TARGET_DATABASE_REDOLOG_DIRECTORY`*. Specifies the redo log directory of the target database.
`-i`	Runs the command in interactive mode where you must respond to prompts by the system. The default mode is non-interactive. (Optional)
`-a`	Requires the argument `RECOVERY_LOG_LOCATION`. If this option is specified, a minimal database recovery will occur automatically after the clone is created. (Optional)
`-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.

▼ To clone a DB2 database with manual DB2 recovery

Use the `db2ed_clonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_clonedb -S source_db -T target_db1 \
-c Checkpoint_1049927758 -m /db2clone/target_db1
Clone database succeeded.
```

▼ To clone a DB2 database with automatic DB2 recovery

Use the `db2ed_clonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_clonedb -S source_db -T target_db2 \
-c Checkpoint_1049927758 -m /db2clone/target_db2 -a \
/db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/
Clone database succeeded.

The database will be rolled forward to 2003-04-09-22.36.02.0000.

/db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/
will be used to be searched for
archived logs during recovery.

                               Rollforward Status

 Input database alias                   = TARGET_DB2
 Number of nodes have returned status   = 1

 Node number                            = 0
 Rollforward status                     = DB  working
 Next log file to be read               = S0000002.LOG
 Log files processed                    =  -
 Last committed transaction             = 2003-04-08-20.32.53.000000

DB20000I  The ROLLFORWARD command completed successfully.

                               Rollforward Status
 Input database alias                       = TARGET_DB2
 Number of nodes have returned status       = 1

 Node number                                = 0
 Rollforward status                         = not pending
 Next log file to be read                   =
 Log files processed                       = S0000002.LOG - S0000002.LOG
 Last committed transaction                 = 2003-04-08-20.32.53.000000

DB20000I  The ROLLFORWARD command completed successfully.
```

▼ To clone a DB2 database with interactive DB2 recovery

Use the `db2ed_clonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_clonedb -S source_db -T target_db3 \
-c Checkpoint_1049927758 -m /db2clone/target_db3 -i
/db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/
will be used to be searched for
archived logs during recovery.

Press <Return> to continue
Or <r> to retry
Or <q> to quit:

The database will be rolled forward to 2003-04-09-22.36.02.0000.
The archived logs will be retrieved from
/db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/.
The estimated minimum logs are S0000002.LOG-S0000002.LOG.

You can retrieve all log files by executing
cp /db2clone/db2inst1/NODE0000/SQL00002/SQLOGDIR/*.LOG
/db2clone/target_db3/REDOLOG
from another session and then

Press q to continue

Or Press Enter to retrieve the minimum logs.

The recovery process will stop
if more logs is required.

Press <Return> to continue ...
      or <q> to skip ...

                               Rollforward Status
```

```
 Input database alias                   = TARGET_DB3
 Number of nodes have returned status   = 1

 Node number                            = 0
 Rollforward status                     = DB  working
 Next log file to be read               = S0000002.LOG
 Log files processed                    =  -
 Last committed transaction             = 2003-04-08-20.32.53.000000

DB20000I  The ROLLFORWARD command completed successfully.

                               Rollforward Status

 Input database alias                   = TARGET_DB3
 Number of nodes have returned status   = 1

 Node number                            = 0
 Rollforward status                     = not pending
 Next log file to be read               =
 Log files processed                   = S0000002.LOG - S0000002.LOG
 Last committed transaction             = 2003-04-08-20.32.53.000000

DB20000I  The ROLLFORWARD command completed successfully.
```

▼ **To shut down the clone database and unmount the Storage Checkpoint**

Use the db2ed_clonedb command as follows:

```
$ /opt/VRTS/bin/db2ed_clonedb -T target_db3 -o umount
```

▼ **To mount a Storage Checkpoint file system and start the clone database**

Use the db2ed_clonedb command as follows:

```
$ /opt/VRTS/bin/db2ed_clonedb -T target_db3 -o restartdb
```

▼ **To delete a clone database and the Storage Checkpoint used to create it**

Use the db2ed_clonedb command as follows:

```
$ /opt/VRTS/bin/db2ed_clonedb -T target_db3 -o umount -d
```

Creating and Working with Snapplans Using db2ed_vmchecksnap

A snapplan specifies snapshot scenarios for a DB2 database (such as `online_snapshot`, `online_mirror`, or `offline`). You can name a snapplan file whatever you choose. You can use the `db2ed_vmchecksnap -o setdefaults` option to create the snapplan and set default values for the parameters. You may then modify the snapplan file using a text editor.

You can also use the command to validate, copy, list, or remove a snapplan and check the storage to make sure it is configured appropriately for the Database FlashSnap feature. For more information on snapplans, see "Validating a Snapplan (db2ed_vmchecksnap)" on page 184.

Note You must have the Enterprise Edition of VERITAS Storage Foundation *for DB2* to use this command.

Snapplan Parameters

When using `db2ed_vmchecksnap -o setdefaults` option to create the snapplan, the following parameters are set:

Parameter	Value
`SNAPSHOT_VERSION`	Specifies the snapshot version for this release of VERITAS Storage Foundation *for DB2*
`PRIMARY_HOST`	Specifies the name of the host where the primary database resides.
`SECONDARY_HOST`	Specifies the name of the host where the clone database will reside. If the primary and secondary hosts are the same, the snapshot volumes will not be deported.
`PRIMARY_DG`	Specifies the name of the Volume Manager disk group used by the primary database.
`SNAPSHOT_DG`	Specifies the name of the disk group containing the snapshot volumes. The snapshot volumes will be put into this disk group on the primary host and deported if the primary and secondary hosts are different. The secondary host will import this disk group to start a clone database.
`DB2DATABASE`	Specifies the name of the DB2 database.

Parameter	Value
DB2HOME	Specifies the home directory of the database.
REDOLOG_DEST	Specifies the full path of the redo logs.
SNAPSHOT_MODE	**online_snapshot**, **online_mirror**, or **offline** Specifies whether the snapshot mode should be online_snapshot, online_mirror, or offline when the snapshot is created. By default, the SNAPSHOT_MODE is online_snapshot. If the snapshot is created while the SNAPSHOT_MODE for the database is set to online_snapshot or online_mirror, the db2ed_vmsnap command will first put the database into WRITE SUSPEND mode. After db2ed_vmsnap finishes creating the snapshot, it will take the database out of WRITE SUSPEND mode. If the database is offline, it is not necessary to put the database into WRITE SUSPEND mode. If the SNAPSHOT_MODE is offline, the secondary host must be different than the primary host. If the SNAPSHOT_MODE is online_mirror, the primary and secondary host must be the same, and the ALLOW_REVERSE_RESYNC parameter must be set to yes in the snapplan. If the SNAPSHOT_MODE is online_snapshot, the snapshot can be created on either the primary or a secondary host.
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 name of the tag set to the plexes that will be used by db2ed_vmsnap to take the snapshot. The db2ed_vmchecksnap command will use this tag name to search if all the volumes in the database have the plexes with this tag name set.
SNAPSHOT_MIRROR	Specifies the number of plexes to be snapshot. The default value is 1.
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.

Parameter	Value
`ALLOW_REVERSE_RESYNC`	**`yes`** or **`no`** By default, reverse resynchronization is off (set equal to `no`). If it is set to `yes`, this parameter allows you to restore the original volume from a snapshot. The original database, however, must be down for this operation.

Creating a Snapplan

Prerequisites

- You must be the DB2 instance owner.
- 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 with `LOGRETAIN` mode on.

Usage Notes

- The snapplan must be created on the primary host.
- After creating the snapplan using the `db2ed_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 `db2ed_vmchecksnap`(1M) online manual page for more information.
- If the `SNAPSHOT_MODE` for the database is set to `online_snapshot`, the snapshot can be created on either the primary or a secondary host. If the `SNAPSHOT_MODE` is set to `online_mirror`, the snapshot must be created on the primary host. If the `SNAPSHOT_MODE` is set to `offline`, the snapshot must be created on a secondary host.

- Database FlashSnap commands are not supported for partitioned DB2 databases in an SMP or MPP environment.

Options

`-D` *`DB2DATABASE`*	Specifies the name of the DB2 database for which a snapshot image will be created.
`-I` *`DB2INSTANCE`*	Specifies the name of the DB2 instance.
`-f SNAPPLAN`	Specifies the local path or the full path of the snapplan that you are creating.
`-o setdefaults`	Creates a default snapplan. This option can be used with the `-o validate` option to validate that the configuration is correct. See "Summary of Database Snapshot Steps" on page 170 for descriptions of the snapplan parameters.
`-o validate`	Validates each parameter in the snapplan and checks whether the snapshot volumes have been configured correctly for creating snapshots, and copies the snapplan to the repository.
`-o list`	Lists all the snapplans associated with a specific *$DB2DATABASE*.
`-o copy`	Copies the snapplan from the repository to your current local directory.
`-o remove`	Removes the snapplan from the repository.
`-t` *`SECONDARY_HOST`*	Specifies the name of the host to which the snapshot image will be deported. If it is the same as the primary server, the snapshot volumes will not be deported. This argument is required if `-o setdefaults` is used. It is ignored if specified for `-o validate`.
`-p` *`PLEX_TAG`*	Specifies the tag name for the plexes used to create the snapshot. This argument is required if `-o setdefaults` is used.

▼ To create a snapplan and set the default values for a single host

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD  \
-f snap1 -o setdefaults -t host1 -p PRODtag
Snapplan snap1 for PROD.
=====================================================
SNAPSHOT_VERSION=4.0
PRIMARY_HOST=host1
SECONDARY_HOST=host1
PRIMARY_DG=PRODdg
SNAPSHOT_DG=SNAP_PRODdg
DB2DATABASE=PROD
DB2HOME=//PROD_HOME
REDOLOG_DEST=//PROD_HOME/inst1/NODE0000/SQL00001/SQLOGDIR/
SNAPSHOT_MODE=online_snapshot
SNAPSHOT_PLAN_FOR=database
SNAPSHOT_PLEX_TAG=PRODtag
SNAPSHOT_MIRROR=1
SNAPSHOT_VOL_PREFIX=SNAP_
ALLOW_REVERSE_RESYNC=no
```

▼ To create a snapplan and set the default values in a two-host configuration

Use the **`db2ed_vmchecksnap`** command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD \
-f snap2 -o setdefaults -t host2 -p PRODtag
Snapplan snap2 for PROD.
=====================================================
SNAPSHOT_VERSION=4.0
PRIMARY_HOST=host1
SECONDARY_HOST=host2
PRIMARY_DG=PRODdg
SNAPSHOT_DG=SNAP_PRODdg
DB2DATABASE=PROD
DB2HOME=//PROD_HOME
REDOLOG_DEST=//PROD_HOME/inst1/NODE0000/SQL00001/SQLOGDIR/
SNAPSHOT_MODE=online_snapshot
SNAPSHOT_PLAN_FOR=database
SNAPSHOT_PLEX_TAG=PRODtag
SNAPSHOT_MIRROR=1
SNAPSHOT_VOL_PREFIX=SNAP_
ALLOW_REVERSE_RESYNC=no
```

Validating a Snapplan

You can use the `db2ed_vmchecksnap` command with the `-o validate` option to validate a snapplan and check the storage to make sure it is configured appropriately for the Database FlashSnap feature.

▼ To validate a snapplan for a snapshot image to be used on the primary host

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD \
-f snap1 -o validate
PRIMARY_HOST is host1

SECONDARY_HOST is host1

The version of PRIMARY_DG-PRODdg is 110.

SNAPSHOT_DB is SNAP_PRODdg

SNAPSHOT_PLAN_FOR is database

Examining DB2 volume and disk layout for snapshot

Volume prod_db on PRODdg is ready for snapshot.
Original plex and DCO log for prod_db is on PRODdg01.
Snapshot plex and DCO log for prod_db is on PRODdg02.

SNAP_PRODdg for snapshot will include: PRODdg02

ALLOW_REVERSE_RESYNC is yes

The snapplan snap1 has been created.
```

▼ To validate a snapplan for a snapshot image to be used on the secondary host

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD \
-f snap2 -o validate
PRIMARY_HOST is host1

SECONDARY_HOST is host2

The version of PRIMARY_DG-PRODdg is 110.

SNAPSHOT_DB is SNAP_PRODdg
```

```
SNAPSHOT_PLAN_FOR is database

Examining DB2 volume and disk layout for snapshot.

Volume prod_db on PRODdg is ready for snapshot.
Original plex and DCO log for prod_db is on PRODdg01.
Snapshot plex and DCO log for prod_db is on PRODdg02.

SNAP_PRODdg for snapshot will include: PRODdg02

ALLOW_REVERSE_RESYNC is yes

The snapplan snap2 has been created.
```

Listing and Viewing Snapplans Using db2ed_vmchecksnap

▼ To list all available snapplans for a specific DB2 database

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD -o list

The following snapplan(s) are available for PROD:

SNAP_PLAN      SNAP_STATUS      DB_STATUS      SNAP_READY
snap1          init_full        init           yes
snap2          init_full        init           yes
snap3          init_full        init           yes
```

Note 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 explanations of the various statuses that may appear for `SNAP_STATUS` and `DB_STATUS`, refer to "VERITAS Database FlashSnap Status Information" on page 419.

▼ To view a snapplan

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD -f snap1 -o list
SNAPSHOT_VERSION=4.0
PRIMARY_HOST=host1
SECONDARY_HOST=host1
PRIMARY_DG=PRODdg
SNAPSHOT_DG=SNAP_PRODdg
```

```
DB2DATABASE=PROD
DB2HOME=//PROD_HOME
REDOLOG_DEST=//PROD_HOME/inst1/NODE0000/SQL00001/SQLOGDIR/
SNAPSHOT_MODE=online_snapshot
SNAPSHOT_PLAN_FOR=database
SNAPSHOT_PLEX_TAG=PRODtag
SNAPSHOT_MIRROR=1
SNAPSHOT_VOL_PREFIX=SNAP_
ALLOW_REVERSE_RESYNC=yes

STORAGE_INFO
PRODdg02
SNAP_PLEX=prod_db-02

STATUS_INFO
SNAP_STATUS=init_full
DB_STATUS=init
LOCAL_SNAPPLAN=/export/snap_dir/snap1
```

Copying or Removing a Snapplan Using db2ed_vmchecksnap

▼ To copy a snapplan from the VxDBA repository to your local directory

To copy a snapplan from the VxDBA repository to your local directory, the snapplan must not already be present in your local directory.

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD -f snap1 -o copy
Copying 'snap1' to '/export/snap_dir'
```

▼ To remove a snapplan

Use the `db2ed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmchecksnap -D PROD -f snap1 -o remove
The snapplan snap1 has been removed from VxDBA repository.
```

Creating, Resynchronizing, or Reverse Resynchronizing a Snapshot Database Using db2ed_vmsnap

You can use the VERITAS Storage Foundation *for DB2* `db2ed_vmsnap` command to create a snapshot image of a DB2 database. The snapshot can be used locally or on another host that is physically attached to the shared storage. You can also resynchronize the snapshot image back to the primary database.

Prerequisites

- You must be logged in as the DB2 instance owner.
- You must create and validate a snapplan using `db2ed_vmchecksnap` before you can create a snapshot image with `db2ed_vmsnap`.

Usage Notes

- The `db2ed_vmsnap` command can only be used on the primary host.
- When creating a snapshot volume, create the snapshot on a separate controller and on separate disks from the primary volume.
- The online redo log must be included in the snapshot if a clone database is to be started.
- 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 `db2ed_vmsnap`(1M) manual page for more information.

Options

`-D` *`DB2DATABASE`*	Specifies the name of the DB2 database for which a snapshot image will be created.
`-f` *`SNAPPLAN`*	Specifies the name of the snapplan you are using.
`-o snapshot [-F] | resync`	Specifies whether to create a snapshot or synchronize the snapshot image with the current database image. The -F option prepares the volumes for being snapshot and forces snapshot creation.
`-o reverse_resync_begin`	Begins reverse resynchronization.

`-o reverse_resync_commit`	Commits the reverse resynchronization changes after you have verified that they are acceptable.
`-o reverse_resync_abort`	Aborts reverse resynchronization and mounts the original volumes back with the file systems that are configured to use the volume.

▼ To create a snapshot image on the primary host

Use the `db2ed_vmsnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 -o snapshot
db2ed_vmsnap started at 2004-05-26 10:58:23
DB20000I  The SET WRITE command completed successfully.
DB20000I  The SET WRITE command completed successfully.
A snapshot of DB2DATABASE PROD is in DG SNAP_DB2dg.
Snapplan snapplan is used for the snapshot.

If -r <relocate_path> is used in db2ed_vmclonedb,
        make sure <relocate_path> is created and owned by DB2
        Instance Owner. Otherwise, the following mount points need
        to be created and owned by DB2 Instance Owner:

        /db2/testvol01.
        /db2/testvol02.
        /db2/testvol03.
        /db2/testvol04.
        /db2/testvol05.
        /db2/udb_home.

db2ed_vmsnap ended at 2004-05-26 10:59:25
DB20000I  The TERMINATE command completed successfully.
```

▼ To resynchronize a snapshot to your database

Use the `db2ed_vmsnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 -o resync
db2ed_vmsnap started at 2004-05-27 10:39:21
DB20000I  The SET WRITE command completed successfully.
DB20000I  The SET WRITE command completed successfully.
A snapshot of DB2DATABASE PROD is in DG SNAP_DB2dg.
Snapplan snapplan is used for the snapshot.
```

```
If -r <relocate_path> is used in db2ed_vmclonedb,
      make sure <relocate_path> is created and owned by DB2 Instance
      Owner.
      Otherwise, the following mount points need to be
      created and owned by DB2 Instance Owner:

      /db2/testvol01.
      /db2/testvol02.
      /db2/testvol03.
      /db2/testvol04.
      /db2/testvol05.
      /db2/udb_home.

db2ed_vmsnap ended at 2004-05-27 10:41:01
DB20000I  The TERMINATE command completed successfully.
```

▼ **To resynchronize your database to a snapshot**

Note To run this command successfully, the mount point for the primary database must be created by and owned by the DB2 instance owner before mounting the VxFS file system.

Use the `db2ed_vmsnap` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 \
-o reverse_resync_begin
db2ed_vmsnap started at 2004-05-26 13:37:11
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
db2ed_vmsnap ended at 2004-05-26 13:37:48
```

▼ **To abort resynchronizing your database to a snapshot**

This option is only allowed when `reverse_resync_begin` has been run. It is not allowed if `reverse_resync_commit` has been executed.

Note If your snapshot was created with the snapshot mode set to `online_mirror`, you cannot run `db2ed_vmsnap -o reverse_resync_begin` after running `db2ed_vmsnap -o reverse_resync_abort`. You must take a new snapshot before performing reverse resynchronization again.

Use the db2ed_vmsnap command as follows:

```
$ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 -o \
reverse_resync_abort
db2ed_vmsnap started at 2004-05-26 13:38:16
DB20000I  The FORCE APPLICATION command completed successfully.
DB21024I  This command is asynchronous and may not be effective
immediately.

DB20000I  The UNCATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the directory
cache is refreshed.
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
DB20000I  The UNCATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the directory
cache is refreshed.
DB20000I  The CATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the directory
cache is refreshed.
The option reverse_resync_abort has been completed.
db2ed_vmsnap ended at 2004-05-26 13:39:31
```

▼ To commit reverse resynchronization changes

This option is only allowed after reverse_resync_begin has been run.

Caution Upon completion of reverse resynchronization, the content of the original database is discarded. Storage Checkpoints taken on either the original database or the clone database *after* the snapshot was created are discarded. Storage Checkpoints taken *before* the snapshot was created are preserved. The db2ed_vmsnap -o reverse_resync_commit command cannot be undone and should be used with extreme caution.

Use the `db2ed_vmsnap` command as follows:

```
  $ /opt/VRTS/bin/db2ed_vmsnap -D PROD -f snap1 -o
reverse_resync_commit
  db2ed_vmsnap started at 2004-05-26 13:40:32
  DB20000I  The FORCE APPLICATION command completed successfully.
  DB21024I  This command is asynchronous and may not be effective
  immediately.

  DB20000I  The UNCATALOG DATABASE command completed successfully.
  DB21056W  Directory changes may not be effective until the directory
  cache is refreshed.
  DB20000I  The CATALOG DATABASE command completed successfully.
  DB21056W  Directory changes may not be effective until the directory
  cache is refreshed.
  db2ed_vmsnap ended at 2004-05-26 13:41:35
```

Creating or Shutting Down a Clone Database Using db2ed_vmclonedb

You can use the VERITAS Storage Foundation *for DB2* `db2ed_vmclonedb` command to create or shutdown a clone database on either the primary or secondary host using snapshot volumes from the primary host.

Prerequisites

- You must be logged in as the DB2 instance owner to use `db2ed_vmclonedb` command.
- Before you can use the `db2ed_vmclonedb` command, you must complete the steps in "Summary of Database Snapshot Steps" on page 170, "Validating a Snapplan (db2ed_vmchecksnap)" on page 184, and "Creating a Snapshot (db2ed_vmsnap)" on page 190.
- 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 `db2ed_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 DB2 instance owner.
- The `SNAPSHOT_MODE` must be `online_snapshot` or `offline`. If `SNAPSHOT_MODE` is set to `offline`, a two-host configuration is required.

Usage Notes

- The `db2ed_vmclonedb` command can be used on the secondary host.
- In a single-host configuration, the primary and secondary hosts can be the same.
- In a single-host configuration, `-r` *`relocate_path`* is required.
- In a two-host configuration, the `vxdbavol=`*`vol_name`* option is required.
- See the `db2ed_vmclonedb`(1M) manual page for more information.

Options

`-D` *`DB2DATABASE`*	Specifies the name of the DB2 database for which a snapshot image will be created.
`-I` *`DB2INSTANCE`*	Specifies the name of the DB2 instance used to create the snapshot. This is required when cloning a database under a different instance.
`-g` *`snap_dg`*	Specifies the name of the disk group that contains all snapshot volumes.
`-o mount`	Mounts the file systems so you can use them to do a backup.
`-o recoverdb`	Automatically recovers the database.
`-o restartdb`	Restarts the database if the clone database is shut down. A clone database must exist to use the `-o restartdb` option.
`-o update_status`	Updates the database status information in the VxDBA repository.
`-o umount`	Shuts down the clone database and unmounts all snapshot files.
`vxdbavol=`*`vol_name`*	Specifies the volume that contains snapplan data. This name is not determined by the user. It is provided after you run `vmsnap -o snapshot`. This parameter is required when creating the clone in a two-host configuration.
`-f SNAPPLAN`	Indicates the name of the snapplan that you are using.

`-r` *`relocate_path`*	Specifies the initial mount point for the snapshot image. If you are creating a clone in a single-host configuration, -r is required. Otherwise, it is an optional argument. The system administrator needs to create and change the owner of this mount point to the DB2 instance owner. The `db2ed_vmclonedb` command will fail if the DB2 instance owner does not have access rights to this mount point. The `-r` *`relocate_path`* option is not allowed if `SNAPSHOT_MODE=offline`. If `-r` *`relocate_path`* is used in `-o mount | recoverdb`, then it is required to restart or unmount the clone database.

▼ To clone a database automatically

In a single-host configuration:

To create a clone of the primary database on the same host as the primary database, use the `db2ed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o recoverdb,new_db=NEWPROD -f snap1 -r /clone
db2ed_vmclonedb started at 2004-04-01 13:18:02
Mounting /clone/testvol on
/dev/vx/dsk/SNAP_PRODdg/SNAP_testvol.
Mounting /clone/udb_home on
/dev/vx/dsk/SNAP_PRODdg/SNAP_udb_home.
Relocating database...
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
Database relocation was successful.
DBT1000I  The tool completed successfully.
DB20000I  The CATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the directory
cache is refreshed.
db2ed_vmclonedb ended at 2004-04-01 13:18:40
```

In a two-host configuration:

To create a clone of the primary database on a secondary host, use the db2ed_vmclonedb command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o recoverdb,new_db=NEWPROD,vxdbavol=SNAP_arch -f snap2
db2ed_vmclonedb started at 2004-04-06 07:54:09
Mounting /clone/testvol on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol.
Mounting /clone/udb_home on
/dev/vx/dsk/SNAP_DB2dg/SNAP_udb_home.
Relocating database...
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
Database relocation was successful.
DBT1000I  The tool completed successfully.
db2ed_vmclonedb ended at 2004-04-06 07:55:00
```

▼ To clone a database manually

In a single-host configuration:

To mount file systems and create a clone of the primary database on the same host as the primary database *without recovering the clone database*, use the db2ed_vmclonedb command as follows. The clone database must be manually recovered before it can be used:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o mount,new_db=NEWPROD -f snap1 -r /clone
db2ed_vmclonedb started at 2004-04-01 13:30:12
Mounting /clone/testvol on
/dev/vx/dsk/SNAP_PRODdg/SNAP_testvol.
Mounting /clone/udb_home on
/dev/vx/dsk/SNAP_PRODdg/SNAP_udb_home.
db2ed_vmclonedb ended at 2004-04-01 13:30:24
```

Recover the database manually.

Update the snapshot status (database_recovered) for the clone database on the primary host after manual recovery has been completed:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -o
update_status,new_db=NEWPROD -f snap1 -r /clone
db2ed_vmclonedb started at 2004-04-01 13:37:09
The snapshot status has been updated.
db2ed_vmclonedb ended at 2004-04-01 13:37:23
```

In a two-host configuration:

To mount file systems and create a clone of the primary database on a secondary host *without recovering the clone database,* use the `db2ed_vmclonedb` command as follows. The clone database must be manually recovered before it can be used:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
 -o mount,new_db=NEWPROD,vxdbavol=SNAP_arch -f snap2
db2ed_vmclonedb started at 2004-04-06 09:06:09
Mounting /clone/testvol on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol.
Mounting /clone/udb_home on
/dev/vx/dsk/SNAP_DB2dg/SNAP_udb_home.
db2ed_vmclonedb ended at 2004-04-06 09:06:43
```

Recover the database manually.

Update the snapshot status (`database_recovered`) for the clone database on the secondary host after manual recovery has been completed:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -o update_status,new_db=NEWPROD \
-f snap2
db2ed_vmclonedb started at 2004-04-06 09:22:27
The snapshot status has been updated.
db2ed_vmclonedb ended at 2004-04-06 09:22:40
```

▼ To shut down the clone database and unmount all snapshot file systems

In a single-host configuration:

To shut down the clone database and unmount file systems for a clone on the same host as the primary database, use the `db2ed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -o umount,new_db=NEWPROD \
-f snap1 -r /clone
db2ed_vmclonedb started at 2004-04-02 15:11:22
Umounting /clone/prod_db.
Umounting /clone/prod_ar.
db2ed_vmclonedb ended at 2004-04-02 15:11:47
```

In a two-host configuration:

To shut down the clone database, unmount file systems, and deport the snapshot disk group for a clone on a secondary host, use the `db2ed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -o umount,new_db=NEWPROD \
-f snap2
db2ed_vmclonedb started at 2004-04-09 23:09:21
Umounting /clone/arch.
Umounting /clone/prod_db.
db2ed_vmclonedb ended at 2004-04-09 23:09:50
```

▼ To restart a clone database

In a single-host configuration:

To start the clone database on the same host as the primary database, use the `db2ed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o restartdb,new_db=NEWPROD -f snap1 -r /clone
db2ed_vmclonedb started at 2004-05-26 13:20:58
Mounting /clone/testvol01 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol01.
Mounting /clone/testvol02 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol02.
Mounting /clone/testvol03 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol03.
Mounting /clone/testvol04 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol04.
Mounting /clone/testvol05 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol05.
Mounting /clone/udb_home on
/dev/vx/dsk/SNAP_DB2dg/SNAP_udb_home.
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
db2ed_vmclonedb ended at 2004-05-26 13:21:48
```

In a two-host configuration:

To start the clone database on the secondary host, use the `db2ed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/db2ed_vmclonedb -D PROD -g SNAP_PRODdg \
-o restartdb,new_db=NEWPROD,vxdbavol=SNAP_arch -f snap2
db2ed_vmclonedb started at 2004-05-26 10:14:21
Mounting /clone/testvol01 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol01.
Mounting /clone/testvol02 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol02.
Mounting /clone/testvol03 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol03.
Mounting /clone/testvol04 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol04.
Mounting /clone/testvol05 on
/dev/vx/dsk/SNAP_DB2dg/SNAP_testvol05.
Mounting /clone/udb_home on
/dev/vx/dsk/SNAP_DB2dg/SNAP_udb_home.
Files and control structures were changed successfully.
Database was catalogued successfully.
DBT1000I  The tool completed successfully.
db2ed_vmclonedb ended at 2004-05-26 10:15:15
```

Managing Log Files Using edgetmsg2

You can use the `edgetmsg2` utility to manage message log files. You can use the `edgetmsg2` utility to write a message to a log file or to the console, read the log file and print to the console, and display the available log files.

Prerequisites

- You must be logged in as the instance owner or root to use this command.

Usage Notes

- The default log file for a database is located in the following directory:

  `/etc/vx/vxdba/logs/sfua_`*`database`*`.log`

  where *`database`* is the *`db2database`*.
- By default, only messages with a severity greater than `ERROR` will be logged.
- The `-s` *`set_num`* option specifies the message catalogue set number. The default is 1.
- The `-M` *`msgid`*`[:`*`severity`*`]` option specifies the message ID and severity to be printed.
- The `-f` *`msg_catalog`* `|` *`logfile`* `|` *`log_directory`* option specifies the message catalogue path, log file, or log directory.
- The `-v` *`severity`* `|` *`severity`* option is used to overwrite the minimum log severity or to create a severity filter. The severity values are either `0-8` or `100-108`.
- The `-p` option pauses the cursor at the end of a display message. By default, a line feed is added to each display message. Use the `-p` option to indicates that no line feed is to be added.
- The `-m` *`value`* option is for internal use only. This option overwrites certain internal settings for the `edgetmsg2` utility.
- The `-o` *`list`*`[,`*`suppress_time`*`]` option displays the content of a log file. You can specify `,`*`suppress_time`* to exclude time information in the utility output.
- The `-o` *`report`*`[,`*`no_archive`*`]` displays the available log files. You can specify `,`*`no_archive`* to exclude log files from the utility output.
- The `-t` *`from_time`*`[,`*`to_time`*`]` reduced the length of the utility output by specifying the time range to include. This option must be used together with the `-o list` option. Use the following format: *`yyy-mm-dd HH:MM:SS`*.
- The `-I` *`db2instance`* option specifies the DB2 instance.
- The `-D` *`db2database`* option specifies the DB2 database.

- "`default format string`" option specifies the C language `printf()` format string.
- [*`args`*] specifies arguments for the format string conversion characters.
- See the `edgetmsg2`(1M) manual page for more information.

▼ To print a message

Use the `edgetmsg2` command as follows:

```
$ /opt/VRTSdb2ed/bin/edgetmsg2 [-s set_num] [-M msgid[:severity]] \
[-f msg_catalog] [-v severity] [-p] [-m value] \
["default format string" [args]]
```

▼ To read a message log file

Use the `edgetmsg2` command as follows:

```
$ /opt/VRTSdb2ed/bin/edgetmsg2 -o list[,suppress_time] \
-I db2instance -D db2database | [-f logfile] \
[-v severity] [-t from_time,to_time]
```

Output similar to the following is displayed:

```
$ edgetmsg2 -o list -S t10 -t "2004-10-14 13:44:45,2004-10-14 14:46:31"
2004-10-14 13:44:45 SFDB2 vxsnapadm ERROR V-81-5542 Snapback datavol failed.
2004-10-14 13:44:45 SFDB2 vxsnapadm ERROR V-81-5542 Snapback archvol failed.
2004-10-14 13:44:45 SFDB2 db2ed_vmsnap ERROR V-81-5617 snapback failed.
2004-10-14 13:56:46 SFDB2 db2ed_vmsnap ERROR V-81-5255 sp2 does not exist.
2004-10-14 14:46:16 SFDB2 db2ed_vmclonedb ERROR V-81-4833 There already appears to
be a database T1 running.
2004-10-14 14:46:31 SFDB2 db2ed_vmclonedb ERROR V-81-5713 The database name exceeds
the maximum length of 8 characters.
```

▼ To list available log files

Use the `edgetmsg2` command as follows:

```
$ /opt/VRTSdb2ed/bin/edgetmsg2 -o report[,no_archive] \
[-f log_directory]
```

Output similar to the following is displayed:

```
dbid      begin                end                  size(K) file
----      -----                ---                  ----    ----
default   2004-10-25 09:50:17  2004-10-25 11:04:00  0       /etc/vx/vxdba/logs/sfua_default.log
t10       2004-09-09 08:54:59  2004-10-21 14:19:35  10      /etc/vx/vxdba/logs/sfua_t10.log
clone2    2004-10-11 09:22:35  2004-10-11 09:22:35  0       /etc/vx/vxdba/logs/sfua_clone2.log
T1        2004-10-14 14:51:33  2004-10-14 14:51:33  0       /etc/vx/vxdba/logs/sfua_T1.log
Tiff1     2004-10-21 14:24:27  2004-10-21 14:24:27  0       /etc/vx/vxdba/logs/sfua_Tiff1.log
clone     2004-08-05 09:23:19  2004-10-21 17:45:19  11      /etc/vx/vxdba/logs/sfua_clone.log
```

Displaying I/O Mapping and Statistics Using vxstorage_stats

You can use the VERITAS Storage Foundation *for DB2* `vxstorage_stats` command to display I/O mapping and statistics about VERITAS File System files one file at a time. The statistics are recorded only for VxFS files and VxVM volumes. These statistics show I/O activity.

Prerequisites

- You may be logged in as either the database administrator or `root`.

Usage Notes

- The `-m` option displays the I/O topology for the specified file.
- The `-s` option displays the file statistics for the specified file.
- The `-c` *`count`* option specifies the number of times to display statistics.
- The `-i` *`interval`* option specifies the interval frequency for displaying updated I/O statistics.
- The `-f` *`filename`* option specifies the file to display I/O mapping and statistics for.

Command usage for `vxstorage_stats` is as follows:

```
$ /opt/VRTS/bin/vxstorage_stats [-m] [-s] \
[-i interval -c count ] -f file_name
```

To display I/O mapping information

Use the `vxstorage_stats` command with the `-m` option as follows:

```
$ /opt/VRTS/bin/vxstorage_stats -m -f \
/data/PRODqiotbs
TY NAME                      DESCRIPTION      SIZE       EXT   OFFSET
fi /data/system01.dbf          FILE         536870912      1   1048576
v  /dev/vx/rdsk/db2dg/mydb01_user MIRROR    16777216      0    0
TY NAME                      DESCRIPTION      SIZE       EXT   OFFSET
fi /data/system01.dbf          FILE         536870912      1   1048576
v  /dev/vx/rdsk/db2dg/mydb01_user MIRROR    16777216      0    0
pl vxvm:db2dg/mydb01_user-01     STRIPE     16777600      0    0
rd /dev/vx/rdmp/c5t2d0s2      HOST_DEVICE  0              0    0
sd /dev/rdsk/c5t2d0s2         DEVICE        3355520       0    0
da c5t2d0                     DISK          71120064      0    0
rd /dev/vx/rdmp/c5t3d0s2      HOST_DEVICE  0              0    0
sd /dev/rdsk/c5t3d0s2         DEVICE        3355520       0    0
da c5t3d0                   DISK            71120064      0    0
rd /dev/vx/rdmp/c5t4d0s2    HOST_DEVICE     0             0    0
sd /dev/rdsk/c5t4d0s2       DEVICE          3355520       0    0
da c5t4d0                   DISK            71120064      0    0
```

```
rd /dev/vx/rdmp/c5t0d0s2    HOST_DEVICE    0            0   0
sd /dev/rdsk/c5t0d0s2       DEVICE         3355520      0   0
da c5t0d0                   DISK           71120064     0   0
rd /dev/vx/rdmp/c5t1d0s2    HOST_DEVICE    0            0   0
sd /dev/rdsk/c5t1d0s2       DEVICE         3355520      0   0
da c5t1d0                   DISK           71120064     0   0
The output indicates that the file has a single extent, which usually indicates a
VERITAS Quick I/O file. The volume mydb01_user, where the file system is created,
is a VERITAS Volume Manager Volume with 5 column striping across 5 disks (c5t0 to
c5t4).
```

Note For file type (`fi`), the `SIZE` column is number of bytes, and for volume (`v`), plex (`pl`), sub-disk (`sd`), and physical disk (`da`), the `SIZE` column is in 512-byte blocks. Stripe sizes are given in sectors.

▼ To display the entire I/O mapping and statistics for each I/O stack

Use the `vxstorage_stats` command with the `-m` and `-s` options as follows:

```
$ /opt/VRTS/bin/vxstorage_stats -m -s -f /data/system01.dbf

TY NAME                    NSUB  DESCRIPTION     SIZE            OFFSET
PROPERTIES
fi /data/system01.dbf      1     FILE            262146048       1172128
Extents: 6      Sparse Extents:0
v  data_vol                2     MIRROR           8388608         0
pl vxvm:mapdg/data_vol-01 1     CONCAT_VOLUME    8388608         0
rd /dev/vx/rdmp/c1t10d0s2 1     PARTITION        8388608         0
sd /dev/rdsk/c1t10d0s2    1     PARTITION        35368272        0
da c1t10d0                0     DISK             35368272        0
pl vxvm:mapdg/data_vol-03 1     CONCAT_VOLUME    8388608         0
rd /dev/vx/rdmp/c1t13d0s2 1     PARTITION        8388608         0
sd /dev/rdsk/c1t13d0s2    1     PARTITION        71127180        0
da c1t13d0                0     DISK             71127180        0
                            OPERATIONS     FILE BLOCKS(512 byte)  AVG TIME(ms)

OBJECT                      READ    WRITE    B_READ    B_WRITE    AVG_RD  AVG_WR
/data/system01.dbf          615     19       20752     152        3.53    24.74
/dev/vx/rdsk/mapdg/data_vol 19444   33318    895903    1376825    9.26    16.14
vxvm:mapdg/data_vol-01      19444   33318    895903    1376825    9.24    14.00
/dev/rdsk/c1t10d0s2         19444   33318    895903    1376825    9.24    14.00

c1t10d0                     19444   33318    895903    1376825    9.24    14.00
vxvm:mapdg/data_vol-03      0       33318    0         1376825    0.00    14.18
/dev/rdsk/c1t13d0s2         0       33318    0         1376825    0.00    14.18
c1t13d0                     0       33318    0         1376825    0.00    14.18
```

Example

To display statistics two times with a time interval of two seconds:

```
$ /opt/VRTSdb2ed/bin/vxstorage_stats -s -i2 -c2 \
-f /data/system01.dbf
```

Identifying VxFS Files to Convert to Quick I/O Using qio_getdbfiles

You can use the `qio_getdbfiles` command to identify VxFS files before converting them to Quick I/O files. Only VxFS files may be converted to Quick I/O.

The `qio_getdbfiles` command queries the database and gathers a list of containers to be converted to Quick I/O. The command requires direct access to the database.

Prerequisites

- To use this command for DB2, the `DB2DATABASE` environment variable must be set.
- You must be logged in as the instance owner.

Usage Notes

- The `-T` option forces the behavior for a specific database type. The database options that are supported are `ora`, `syb`, and `db2`. Use this option in environments with more than one type of database.
- See the `qio_getdbfiles`(1M) manual page for more information.

▼ To identify the VxFS files to convert to Quick I/O

1. Use the `qio_getdbfiles` command as follows:

   ```
   $ /opt/VRTSdb2ed/bin/qio_getdbfiles [-T ora|syb|db2]
   ```

   where `-T` forces behavior for a specific database type. Use this option in environments with more than one type of database.

   The `qio_getdbfiles` command stores the filenames and file sizes in bytes in a file called `mkqio.dat`.

2. View the `mkqio.dat` file:

   ```
   $ cat mkqio.dat
   ```

   The `mkqio.dat` file contains the database filenames that can be converted to Quick I/O files. The format of the file is a list of paired file paths and file sizes. For example:

   ```
   /database/dbfiles.001 1024000
   /database/dbfiles.002 2048000
   ```

Converting VxFS Files to Quick I/O Using qio_convertdbfiles

After running `qio_getdbfiles`, you can use the `qio_convertdbfiles` command to convert database files to use Quick I/O. This command is for use with VxFS file systems only.

The `qio_convertdbfiles` command converts regular files or symbolic links that point to regular files on VxFS file systems to Quick I/O. The `qio_convertdbfiles` command converts only those files listed in the `mkqio.dat` file to Quick I/O. The `mkqio.dat` file is created by running `qio_getdbfiles`. It can also be created manually.

Prerequisites

- To use this command for DB2, the `DB2DATABASE` environment variable must be set.
- You must be logged in as the instance owner to use this command.

Usage Notes

- The `-T` option forces the behavior for a specific database type. The database options that are supported are `ora`, `syb`, and `db2`. Use this option in environments with more than one type of database.
- The `qio_convertdbfiles` command expects all files to be owned by the database administrator.
- The `-a` option changes regular files to Quick I/O files using absolute pathnames. Use this option when symbolic links need to point to absolute pathnames. By default, relative pathnames are used.
- The `-f` option reports on current fragmentation levels for files listed in `mkqio.dat`. Fragmentation is reported at four levels: not fragmented, slightly fragmented, fragmented, and highly fragmented.
- Converting existing database files to Quick I/O is not recommended if the files are fragmented. In this case, it is recommended that you create new files with the `qiomkfile` command (these files are guaranteed not to be fragmented) and then convert the data from the old files (using a command such as `dd`).
- The `-h` option displays a help message.
- The `-i` option creates extra links for all database files and log files in the /dev directory to support the SAP `brbackup` command.
- The `-u` option changes Quick I/O files back to regular files.
- Ensure that the database is shut down before running `qio_convertdbfiles`.
- See the `qio_convertdbfiles`(1M) manual page for more information.

▼ To convert VxFS files to Quick I/O files

1. After running the `qio_getdbfiles` command, shut down the database:

Caution Running `qio_convertdbfiles` with any option except `-f` while the database is up and running can cause severe problems for your database, including data loss and corruption. Make sure the database is shut down before running the `qio_convertdbfiles` command.

2. Run the `qio_convertdbfiles` command to convert the list of files in `mkqio.dat` to Quick I/O files:

```
$ /opt/VRTSdb2ed/bin/qio_convertdbfiles
```

Note You must remove any non-VxFS files from `mkqio.dat` before running `qio_convertdbfiles`. The `qio_convertdbfiles` command will display an error message if any of the database files in `mkqio.dat` are not on a VxFS file system.

3. Restart the database to access these database files using the Quick I/O interface.

▼ To undo a previous run of qio_convertdbfiles

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

```
$ /opt/VRTSdb2ed/bin/qio_convertdbfiles -u
.dbfile::cdev:vxfs: --> dbfile
```

Note If the database is up and running, an error message will be displayed stating that you need to shut it down before you can run `qio_convertdbfiles`.

Example

In this example a regular VxFS file named `dbfile` is converted to Quick I/O and then converted back to a regular VxFS file:

Get information about the file:

```
$ /opt/VRTSdb2ed/bin/qio_getdbfiles
$ cat mkqio.dat
dbfile 104800000
```

Shut down the database and convert the file to Quick I/O:

```
$ /opt/VRTSdb2ed/bin/qio_convertdbfiles

$ ls -alL d* .d*
-rw-r--r-- 1 admin dbgrp 104857600 May 2 13:42 .dbfile
crw-r--r-- 1 admin dbgrp 45,1 May 3 12:18 dbfile

$ ls -al d* .d*
-rw-r--r-- 1 admin dbgrp 104857600 May 2 14:42 .dbfile
lrwxrwxrwx 1 admin dbgrp 19 May 3 12:18 dbfile ->
.dbfile::cdev:vxfs:
```

The `qio_convertdbfiles` command renames the file `dbfile` to `.dbfile` and creates a symbolic link to `.dbfile` with the Quick I/O extension. By default, the symbolic link uses a relative path name.

Start up the database.

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

```
$ /opt/VRTSdb2ed/bin/qio_convertdbfiles -u
.dbfile::cdev:vxfs: --> dbfile
```

Recreating Quick I/O Files Using qio_recreate

You can use the `qio_recreate` command to automatically recreate Quick I/O files when the database is recovered.

Prerequisites

- To use this command for DB2, the `DB2DATABASE` environment variable must be set.
- You must be logged in as the instance owner to use this command.

Usage Notes

- The `qio_recreate` command expects to find a file named `mkqio.dat` in the directory where the command is run. The mkqio.dat file contains a list of the Quick I/O files used by the database and their sizes. If the `mkqio.dat` file is not in the directory, you will be prompted to create it using `qio_getdbfiles`. For more information, see "Identifying VxFS Files to Convert to Quick I/O Using qio_getdbfiles" on page 412.

- The qio_recreate command supports conventional Quick I/O files only (that is, Quick I/O files in the following form: file --> .file::cdev:vxfs:). In creating a Quick I/O file, the qio_convertdbfiles command renames the regular VxFS file, file, to .file with the Quick I/O extension (:cdev::vxfs:) and creates a symbolic link to it. By default, the symbolic link uses a relative path name.
- There are no options for the qio_recreate command and no output is returned when the command runs successfully.
- The qio_recreate command follows these rules in recreating Quick I/O files when a database is recovered:
  - If a Quick I/O file (.file::cdev:vxfs:) is missing, then qio_recreate recreates it.
  - If both a symbolic link (file) and its associated Quick I/O file (.file::cdev:vxfs:) are missing, qio_recreate recreates both the symbolic link and the Quick I/O file.
  - If a symbolic link (file) from a regular VxFS file to its associated Quick I/O file (.file::cdev:vxfs:) is missing, then qio_recreate recreates the symbolic link.
  - If a Quick I/O file (.file::cdev:vxfs:) is missing and the regular VxFS file that is symbolically linked to it is not the same one that originally created it, then qio_recreate issues a warning message and does not recreate the Quick I/O file.
  - If a Quick I/O file (.file::cdev: vxfs:) is smaller than the size listed in mkqio.dat, qio_recreate issues a warning message.
- See the qio_recreate(1M) manual page for more information.

▼ To automatically recreate Quick I/O files when the database is recovered

Use the qio_recreate command as follows:

```
$ /opt/VRTSdb2ed/bin/qio_recreate
```

Using Third-Party Software to Back Up Files B

Using third-party software to back up VERITAS Quick I/O files requires special consideration and handling. This appendix discusses these issues. For information about backing up files using VERITAS NetBackup, refer to “Using VERITAS NetBackup for Database Backup” on page 215.

Topics covered in this chapter include:

Using Legato NetWorker to Back Up and Restore Quick I/O Files

When Quick I/O files are created using the command `qiomkfile`, a hidden file with storage space allocated to it and a link are created. The link points to the Quick I/O interface of the hidden file. Using `qiomkfile` ensures that the space for the file is allocated in a contiguous manner, which typically improves performance.

Legato NetWorker does not follow symbolic links during backups because doing so would result in the data getting backed up twice: once using the symbolic link and once as the file itself. As a result, Quick I/O files must be backed up as two separate files and restored accordingly.

Because Legato NetWorker deletes and recreates files before they are restored, the restored files lose their contiguous allocation and could be restored as fragmented files with indirect extents. While this does not impact the integrity of the data being restored, it can degrade performance. Creating the file using `qiomkfile` before doing the backup does not resolve this problem because NetWorker deletes and recreates the file.

To avoid this potential performance degradation, Quick I/O files must be backed up and restored using the same methods used to back up and restore raw devices. This method involves using the NetWorker `rawasm` command to back up or `save` directories containing Quick I/O files. Because of the way the `rawasm` command works, NetWorker follows the Quick I/O symbolic link to back up the actual data in the hidden file. Skip the hidden file to avoid backing up the data twice. During restore, NetWorker looks at the attributes of the saved file and restores it using `rawasm`, bypassing the file deletion and recreation steps. For example:

```
$ ls -al /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 db2inst1 db2iadm11048576   Oct 20 17:39 .dbfile
lrwxrwxrwx   1 db2inst1 db2iadm1     22   Oct 20 17:39 dbfile ->\
                                            .dbfile::cdev:vxfs:
```

The command for backing up the `/db01` directory using `rawasm` would look like:

```
<< /db01 >>
rawasm: dbfile
skip: .dbfile
```

To restore the file, preallocate the Quick I/O file using the `qiomkfile` command and enter:

```
$ cd /db01
$ recover -a /db01/dbfile
```

VERITAS Database FlashSnap Status Information C

The VERITAS Database FlashSnap functionality provides both snapshot status information and snapshot database status information for various stages of snapplan and snapshot procedures. You can view the status information through the CLI and through the GUI.

For more information about Database FlashSnap command line functionality, see "Using Database FlashSnap for Backup and Off-Host Processing" on page 151.

For more information about Database FlashSnap GUI functionality, see "Managing Snapshots with Database FlashSnap" on page 268.

Topics covered in this appendix are:

Obtaining Database FlashSnap Snapshot Status and Database Status Using the CLI

You can obtain both the snapshot status and the database status from the command line using the `db2ed_vmchecksnap` command with the `-o list` option. The snapshot status and database status information may also appear in error messages. The tables in this section provide detailed information about the various status values.

Snapshot Status Details

To view snapshot status information from the command line, use the `db2ed_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`.

For information about the various status values, see the following table:

SNAP_STATUS	Completed Operations	Allowed Operations
`init_full`	◆ `db2ed_vmchecksnap -o validate` (successful) ◆ `db2ed_vmsnap -o resync` (successful) ◆ `db2ed_vmsnap -o reverse_resync_commit` (successful)	◆ `db2ed_vmsnap -o snapshot`
`init_db`	◆ `db2ed_vmchecksnap -o validate` (failed)	◆ `db2ed_vmchecksnap -o validate | list -f` *`snapplan`* ◆ Ensure that your storage configuration has been set up correctly. See "Preparing Hosts and Storage for Database FlashSnap" on page 159.
`invalid`	◆ `db2ed_vmchecksnap -o validate` (failed)	◆ `db2ed_vmchecksnap -o validate`

SNAP_STATUS	Completed Operations	Allowed Operations
`snapshot_start`	◆ `db2ed_vmsnap -o snapshot` (failed)	◆ Contact your system administrator for help. Use VERITAS Volume Manager commands to resynchronize the snapshot volumes, and use `db2ed_vmsnap -o snapshot -F` to force snapshot creation.
`snapshot_end`	◆ `db2ed_vmsnap -o snapshot` (successful) ◆ `db2ed_vmsnap -o reverse_resync_abort` (successful)	◆ `db2ed_vmsnap -o resync` ◆ `db2ed_vmsnap -o reverse_resync_begin` ◆ `db2ed_vmclonedbmount|restartdb`
`resync_start`	◆ `db2ed_vmsnap -o resync` (failed)	◆ Contact your system administrator for help. Use VERITAS Volume Manager commands to resynchronize the snapshot volumes, and use `db2ed_vmsnap -o snapshot -F` to force snapshot creation.
`reverse_resync_begin_start`	◆ `db2ed_vmsnap -o reverse_resync_begin` (failed)	◆ Contact VERITAS support.
`reverse_resync_begin_end`	◆ `db2ed_vmsnap -o reverse_resync_begin` (successful)	◆ `db2ed_vmsnap -o reverse_resync_abort` ◆ `db2ed_vmsnap -o reverse_resync_commit`
`reverse_resync_commit_start`	◆ `db2ed_vmsnap -o reverse_resync_commit` (failed)	◆ Contact VERITAS support.
`mount_start`	◆ `db2ed_vmclonedb -o mount` (failed)	◆ `db2ed_vmclonedb -o umount`
`mount_end`	◆ `db2ed_vmclonedb -o mount` (successful)	◆ `db2ed_vmclonedb -o umount`
`restartdb_start`	◆ `db2ed_vmclonedb -o restartdb` (failed)	◆ `db2ed_vmclonedb -o umount` ◆ Start the snapshot database manually.

SNAP_STATUS	Completed Operations	Allowed Operations
`restartdb_end`	◆ `db2ed_vmclonedb -o restartdb` (successful)	◆ `db2ed_vmclonedb -o umount`
`recoverdb_start`	◆ `db2ed_vmclonedb -o recoverdb` (failed)	◆ Recover the snapshot database manually, then run `db2ed_vmclonedb -o update_status` ◆ `db2ed_vmclonedb -o umount`
`recoverdb_end`	◆ `db2ed_vmclonedb -o recoverdb` (successful)	◆ `db2ed_vmclonedb -o umount`
`umount_start`	◆ `db2ed_vmclonedb -o umount` (failed)	◆ Verify that your file system(s) are not busy and retry the command.
`umount_end`	◆ `db2ed_vmclonedb -o umount` (successful)	◆ `db2ed_vmclonedb -o mountdb` ◆ `db2ed_vmclonedb -o restartdb` ◆ `db2ed_vmsnap -o resync` ◆ `db2ed_vmsnap -o reverse_resync_begin`

Database Status Details

To view snapshot status information from the command line, use the `db2ed_vmchecksnap` command with the `-o list` option to list all available snapplans for a specified database. Database status information is displayed in the command output under the column heading `DB_STATUS`. For information about the various status values, see the following table:

DB_STATUS	Completed Operations
`init`	◆ `db2ed_vmchecksnap -o validate` (successful) ◆ `db2ed_vmsnap -o snapshot` (successful) ◆ `db2ed_vmsnap -o reverse_resync_begin` (successful)
`database_recovered`	◆ `db2ed_vmclonedb -o recoverdb` (successful)

Obtaining Database FlashSnap Snapshot Status and Database Status from the GUI

You can obtain both the snapshot status and the snapshot database status from the GUI. The tables in this section provide detailed information regarding the various status values.

Database FlashSnap Snapshot Status Details

To view snapshot status information from the GUI, click on a specific snapplan in the object tree. The snapshot status can be seen on the right side of the window in the **Snapplan State** field. For information regarding the various status values, see the following table:

Snapshot Status (as seen in the Snapplan State field)	Completed Operations	Allowed Operations
`init_full`	◆ Modify/Validate Snapplan (successful) ◆ Resync Snapshot (successful) ◆ Reverse Resync Snapshot with the **commit** option (successful)	◆ Create Snapshot
`snapshot_start`	◆ Create Snapshot (failed)	◆ If the "Create Snapshot" operation failed, contact your system administrator for help. You can use the VxVM utilities to create a snapshot and resynchronize the snapshot volumes, then use the "Create Snapshot" operation with the **Force snapshot creation** option for the subsequent snapshot.
`snapshot_end`	◆ Create Snapshot (successful)	◆ Resync Snapshot ◆ Reverse Resync Snapshot with the **begin** option ◆ Create Snapshot Database with the **Create database** option

Snapshot Status (as seen in the Snapplan State field)	Completed Operations	Allowed Operations
`resync_start`	◆ Resync Snapshot (failed)	◆ If the “Resync Snapshot” operation failed, contact your system administrator for help. You can use the VxVM utilities to resynchronize the snapshot volumes, then use the “Create Snapshot” operation with the **Force snapshot creation** option for the subsequent snapshot.
`reverse_resync_begin_start`	◆ Reverse Resync Snapshot with the **begin** option (failed)	◆ Contact VERITAS support.
`reverse_resync_begin_end`	◆ Reverse Resync Snapshot with the **begin** option (successful)	◆ Reverse Resync with the **commit** option ◆ Reverse Resync with the **abort** option
`reverse_resync_abort_start`	◆ Reverse Resync Snapshot with the **abort** option (failed)	◆ Contact VERITAS support.
`reverse_resync_abort_end`	◆ Reverse Resync Snapshot with the **abort** option (successful)	◆ Reverse Resync Snapshot with the **begin** option ◆ Resync Snapshot ◆ Create Snapshot Database with the **Restart database** option
`reverse_resync_commit_start`	◆ Reverse Resync Snapshot with the **commit** option (failed)	◆ Contact VERITAS support.
`restartdb_start`	◆ Start Up Snapshot Database with the **Restart database** option (failed)	◆ Try to start the snapshot database manually.
`restartdb_end`	◆ Create Snapshot Database with the **Restart database** option (successful)	◆ Unmount Snapshot Database with the **unmount** option

Snapshot Status (as seen in the Snapplan State field)	Completed Operations	Allowed Operations
`mountdb_start`	◆ `dbed_vmclonedb -o mountdb` command failed from the CLI **Note** This option is not supported in the GUI.	◆ Recover the snapshot database manually, then run the `dbed_vmclonedb -o update_status` command from the CLI **Note** This option is not supported in the GUI.
`mountdb_end`	◆ `dbed_vmclonedb -o mountdb` command from the CLI was successful **Note** This option is not supported in the GUI.	◆ Umount Database **FlashSnap** ◆ `dbed_vmclonedb -o update_status` command from the CLI **Note** This option is not supported in the GUI.
`recoverdb_start`	◆ Create Snapshot Database with the **Restart database** option (failed)	◆ Recover the snapshot database manually, then run the `dbed_vmclonedb -o update_status` command from the CLI **Note** This option is not supported in the GUI.
`recoverdb_end`	◆ Create Snapshot Database with the **Restart database** option (successful)	◆ Unmount Snapshot Database with the **umount** option
`umount_start`	◆ `dbed_vmclonedb -o umount` command failed from the CLI **Note** This option is not supported in the GUI.	◆ Verify that your file system(s) are not busy and retry the command.
`umount_end`	◆ `dbed_vmclonedb -o umount` command from the CLI was successful **Note** This option is not supported in the GUI.	◆ Restart Snapshot Database with the **restart database** option ◆ Resync Snapshot ◆ Reverse Resync Snapshot with the **begin** option

Snapshot Database Status Details

To view snapshot database status information from the GUI, click on a specific snapplan in the object tree. The database status can be seen on the right side of the window in the **Database Status** field. For information regarding the various status values, see the following table:

Database Status (as seen in the Database State field)	Completed Operations
`init Db`	◆ Modify/Validate Snapplan (successful) ◆ Create Snapshot (successful) ◆ Reverse Resync Snapshot (successful)
`database_recovered`	◆ Create Snapshot Database with the **Create database** option (successful)

Accessibility and VERITAS Storage Foundation for DB2 D

VERITAS products meet federal accessibility requirements for software as defined in Section 508 of the Rehabilitation Act:

http://www.access-board.gov/508.htm

VERITAS Storage Foundation *for DB2* Graphical User Interface (GUI) inherits the font size and color settings from the operating system it is running on.

Keyboard shortcuts are available for all major GUI operations and menu items. VERITAS products are compatible with operating system accessibility settings as well as a variety of assistive technologies. All manuals also are provided as accessible PDF files, and the online help is provided as HTML displayed in a compliant viewer.

The following topics detail accessibility features and compliance in VERITAS Storage Foundation *for DB2*:

- "Keyboard Navigation and Shortcuts in VERITAS Storage Foundation for DB2" on page 430
- "Keyboard Shortcuts" on page 434
- "Menu Hot Keys" on page 436
- "Support for Accessibility Settings and Assistive Technologies" on page 439

Keyboard Navigation and Shortcuts in VERITAS Storage Foundation for DB2

All program functions and menu items are accessible using the keyboard exclusively. VERITAS Storage Foundation *for DB2* uses standard operating system navigation keys and keyboard shortcuts.

To see a table of the standard Microsoft navigation keys and keyboard shortcuts, select your version of Microsoft Windows from the drop-down listbox at:

http://www.microsoft.com/enable/products/keyboard.aspx

VERITAS Enterprise Administrator (VEA) has the following exceptions and additions to the Microsoft Keyboard standard:

VEA Exceptions to Microsoft Keyboard Conventions

Convention	**Mnemonic**	**Accelerator**
Cancel	C	ALT+C
OK	O	ALT+O
Refresh	e	None
Yes	Y	ALT+Y
No	N	ALT+N
Shift+F10	None	None

VEA Additions to Microsoft Keyboard Conventions

Convention	**Mnemonic**	**Accelerator**
Connect	C	None
Contents	C	None
Delete Now	D	ALT+ D
Delete Temporary Files on Exit	e	ALT+ E
Disconnect	D	None

VEA Additions to Microsoft Keyboard Conventions

Convention	Mnemonic	Accelerator
Help	H	ALT+ H
Number of Items in History	N	ALT+ N
Rescan	s	None
Set to defaults	S	ALT+ S
Remember Password	R	ALT+ R
Username	U	ALT+ U

VEA Help Additions to Microsoft Keyboard Conventions

Convention	Mnemonic	Accelerator
Find in Topic..	F	Ctrl + F
Find Next	n	F3
Find Previous	P	None
Hide Navigation Tabs	H	None
Index	I	None
Print	P	None
Search	S	None
Synchronize	n	None

General Keyboard Navigation Within the GUI

You can navigate and use VERITAS Storage Foundation *for DB2* with only the keyboard. In the GUI, the current active tree or table has a dark blue highlight, and the current active tab, radio button, or checkbox is enclosed within a rectangle formed by dotted lines. These areas are said to have *focus* and will respond to commands.

All VERITAS GUIs use the following keyboard navigation standards:

- Tab moves the focus to the next active area, field, or control, following a preset sequence. Shift+Tab moves the focus in the reverse direction through the sequence.
- Ctrl+Tab exits any Console area that you internally navigate with Tab.
- Up and Down arrow keys move focus up and down the items of a list.
- ALT in combination with the underlined mnemonic letter for a field or command button shifts the focus to that field or button.
- Enter activates your selection. For example, after pressing Tab to select the **Next** button in a wizard panel, press Enter to display the next screen.

Keyboard Navigation Within Dialog Boxes

Dialog boxes contain groups of controls necessary to set options or settings for programs. Here are some general rules about dialog box navigation:

- Tab moves focus between controls within the dialog box along a preset sequence.
- Controls displaying a mnemonic (an underlined letter) can be selected regardless of focus by typing ALT and the underlined letter.
- A dark border indicates the default command button. Press Enter at any time to choose the button with a dark border.
- ALT C chooses the **Cancel** button if one exists.
- Spacebar chooses a control you select with Tab.
- Spacebar changes the state of a checkbox or radio button that has focus. Typing a mnemonic (if one is available) will move the focus to the checkbox or radio button and change its state.
- Arrow keys move focus within listboxes, sliders, groups of option controls, or groups of page tabs.
- Items that cannot be changed are not visited by the Tab key sequence. Options that are unavailable are grayed-out and can neither be selected nor given focus.

While the controls described here are typically found in dialog boxes, they also can occur in other contexts. The same navigation standards will apply.

Tabbed Dialog Boxes

Some dialog boxes use tabbed pages to subcategorize groups of many options. Each tabbed page contains different groups of controls. Use Tab to move the focus between tabs within a dialog box. Typing the mnemonic for a tab also moves the focus to the tab and displays its page of controls.

The following table lists keyboard navigation rules within tabbed dialog boxes:

Keyboard Navigation within Tabbed Dialog Boxes

Keyboard input	Result
Ctrl+Page Down	Switches to the next tab and displays the page
Ctrl+Page Up	Switches to the previous tab and displays the page
Right arrow or Left arrow	When the focus is on a tab selector, chooses the next or previous tab in the current row and displays the page

Listboxes

Listboxes display a column of available choices. There are different kinds of listboxes with additional navigation conventions:

- *Drop-down listboxes* by default show only the selected item. A small button to the right of the control shows a downward-pointing arrow. Select the arrow to display more items from the listbox. If there are more choices than can fit in the preset listbox area, a slider appears along the side of the listbox. Show or hide the list using F4. Enter selects or deselects an item.
- *Extended selection listboxes* support selecting single items, blocks of items, or combinations of the two. After selecting an item, hold down Shift+ or Ctrl+navigation keys to select or deselect additional items or blocks of items.

Keyboard Shortcuts

All menu items can be selected by using accelerator or mnemonic keyboard shortcuts. An accelerator is a key combination that provides shortcut access to a GUI function. A mnemonic (sometimes referred to as a “hot key”) is a single-key equivalent (used in combination with the ALT key) for selecting GUI components such as menu items. The mnemonic “hot key” letter is underlined in the GUI. For example:

- ALT to go into menu pull-down mode
- F key to access the **File** menu
- O key to activate the open command

Mnemonics are case-insensitive. Keys can be pressed sequentially instead of simultaneously.

Keyboard Shortcuts

Keyboard Input	Action
Tab, Shift-Tab (for reversing the action)	Navigates between main components of the user-interface
Shift-F10	Display Context-sensitive menu
Ctrl-A	Selects all items in list
F3	Find Next
Enter, Return	Activates default button (does not require keyboard focus)

Routine functions such as opening, saving, and printing files can be performed using the standard Microsoft keyboard shortcuts.

Keyboard shortcuts are not case-sensitive. Mnemonic keystrokes may be pressed either sequentially or simultaneously. All menu items have mnemonics, but not all menu items have accelerators.

Keyboard Navigation

The following table lists some of the keys frequently used to navigate with the keyboard:

Keyboard Navigation

Keyboard Input	Result
TAB	Move forward between panes in the active Console window.
SHIFT+TAB	Move backwards between panes in the active Console window.
SHIFT+ UP ARROW	Move up one item in the tree view.
SHIFT+DOWN ARROW	Move down one item in the tree view.
SHIFT+PAGE UP	Move to the top item visible in the tree view.
HOME	Move to the first item in the tree view.
END	Move to the last item in the tree view.
RIGHT ARROW	Expands the highlighted item. If the highlighted item does not contain hidden items, behaves like DOWN ARROW.
LEFT ARROW	Collapses the highlighted item. If the highlighted item does not contain expanded items, behaves like UP ARROW.

Menu Hot Keys

The following table lists the hot keys associated with the different menus:

Menu Hot Keys

Action	Keyboard Input
File	ALT+F to open the menu, then: ◆ C– Display the connection dialog window ◆ D-Display the disconnection dialog window ◆ r–Display the properties of the connected host ◆ u–Page setup for print ◆ w–Print preview ◆ P–Print the page ◆ x –Exit
Tools	ALT+T to open the menu, then: ◆ P–Set the display preferences ◆ M–Manage user profiles ◆ E–Display the error console ◆ H– Customize the table headers ◆ c–Scan the host for devices ◆ S–Search for storage devices
Actions	ALT+ A to open the menu, then: ◆ e–Refresh ◆ s–rescan ◆ g–Foreign Device: ◆A–Add a device ◆R–Remove a device ◆L–List the devices ◆
Window	◆ ALT+ W to open the menu, then: ◆ T–Tear off a component from being shown 1-To tear off details 2-To tear off the system details

Menu Hot Keys

Action	Keyboard Input
Help	ALT+H to open the menu, then:
	◆ C– Display the contents
	◆ b– Display information about the Help viewer
DB2 Instances	ALT+D to open the menu, then:
	◆ S– Start the DB2 instance
	◆ U– Update the rescan intervals to be faster or slower
	◆ R–Rescan system information
DB2 Instance	ALT+D to open the menu, then:
	◆ S– Start the DB2 instance
	◆ D– Shut down the DB2 Instance
	◆ R–Rescan system information
DB2 Database	ALT+D to open the menu, then:
	◆ o–Resync the database repository
	◆ h–Check system configuration
	◆ a–Save system Configuration
	◆ R–Rescan system information
	◆ C–Create Clone Database
Monitoring Agent	Alt+M to open the menu, then:
	◆ S–Start the monitoring agent
	◆ o–Stop the monitoring agent
	◆ E–Enable the agent at boot time
	◆ D–Disable the agent at boot time
Tablespace	◆ R–Rescan a tablespace
Container	◆ Alt+C to open the menu, then:
	◆ T– Topology/Statistics to generate datafile statistics
	◆ C–Conversion, to convert oracle datafiles to Quick I/O files.

Menu Hot Keys

Action	Keyboard Input
<u>S</u>napplans	ALT+ S to open menu, then: ◆ C–Create a snapplan ◆ M–Modify/Validate a snapplan ◆ R–Rescan Snapplans
<u>S</u>napplan	ALT+ S to open menu, then: ◆ M–Modify/Validate a snapplan ◆ R–Remove a Snapplan ◆ C–Create a snapshot using snapplan ◆ y–Resync a snapshot ◆ v–Reverse Resync a snapshot ◆ L–View log for the Snapplan
<u>S</u>torage Checkpoints	ALT+S to open menu, then: ◆ C–Create a Storage Checkpoint ◆ P–Create a Storage Checkpoint policy ◆ R– Rescan system information
<u>S</u>torage Checkpoint	ALT+S to open menu, then: ◆ B–Rollback a storage checkpoint ◆ M–Mount a storage checkpoint ◆ U–Unmount a storage checkpoint ◆ R–Remove a storage checkpoint

Support for Accessibility Settings and Assistive Technologies

VERITAS software responds to operating system accessibility settings.

VERITAS products are compatible with Microsoft's accessibility utilities. In Windows 2000, accessibility options involving keyboard responsiveness, display contrast, alert sounds, and mouse operation can be set through the Control Panel (**Start > Settings > Control Panel > Accessibility Options**) and through the Accessibility Wizard (**Start > Programs > Accessories > Accessibility > Accessibility Wizard**).

Note Though all graphics in VERITAS documentation can be read by screen readers, setting your screen reader to ignore graphics may improve performance.

Glossary

address-length pair

Identifies the starting block address and the length of an extent (in file system or logical blocks).

asynchronous I/O

A format of I/O that performs non-blocking reads and writes. This enables the system to handle multiple I/O requests simultaneously.

atomic operation

An operation that either succeeds completely or fails and leaves everything as it was before the operation was started. If the operation succeeds, all aspects of the operation take effect at once and the intermediate states of change are invisible. If any aspect of the operation fails, then the operation aborts without leaving partial changes.

block map

A file system is divided into fixed-size blocks when it is created. As data is written to a file, unused blocks are allocated in ranges of blocks, called extents. The extents are listed or pointed to from the inode. The term used for the data that represents how to translate an offset in a file to a file system block is the "block map" for the file.

boot disk

A disk used for booting an operating system.

buffered I/O

A mode of I/O operation (where I/O is any operation, program, or device that transfers data to or from a computer) that first transfers data into the Operating System buffer cache.

cache

Any memory used to reduce the time required to respond to an I/O request. The read cache holds data in anticipation that it will be requested by a client. The write cache holds data written until it can be safely stored on non-volatile storage media.

Cached Quick I/O

Cached Quick I/O allows databases to make more efficient use of large system memory while still maintaining the performance benefits of Quick I/O. Cached Quick I/O provides an efficient, selective buffering mechanism to complement asynchronous I/O.

cluster

A set of hosts that share a set of disks.

cluster-shareable disk group

A disk group in which the disks are shared between more than one host.

cold backup

The process of backing up a database that is not in active use.

command launcher

A graphical user interface (GUI) window that displays a list of tasks that can be performed by VERITAS Volume Manager or other objects. Each task is listed with the object type, task (action), and a description of the task. A task is launched by clicking on the task in the Command Launcher.

concatenation

A VERITAS Volume Manager layout style characterized by subdisks that are arranged sequentially and contiguously.

configuration database

A set of records containing detailed information on existing VERITAS Volume Manager objects (such as disk and volume attributes). A single copy of a configuration database is called a configuration copy.

container

A physical storage device that can be identified by a directory name, a device name, or a file name.

copy-on-write

A technique for preserving the original of some data. As data is modified by a write operation, the original copy of data is copied.

Applicable to Storage Checkpoint technology, where original data, at the time of the Storage Checkpoint, must be copied from the file system to the Storage Checkpoint when it is to be overwritten. This preserves the frozen image of the file system in the Storage Checkpoint.

database

A database is a collection of information that is organized in a structured fashion. Two examples of databases are Relational Databases (such as Oracle, Sybase, or DB2), where data is stored in tables and generally accessed by one or more keys and Flat File Databases, where data is not generally broken up into tables and relationships. Databases generally provide tools and/or interfaces to retrieve data.

Decision Support Systems

Decision Support Systems (DSS) are computer-based systems used to model, identify, and solve problems, and make decisions.

defragmentation

The act of reorganizing data to reduce fragmentation. Data in file systems become fragmented over time.

device file

A block- or character-special file located in the `/dev` directory representing a device.

device name

The name of a device file. It represents a device. The `c#t#d#s#` syntax identifies the controller, target address, disk, and partition.

direct I/O

An unbuffered form of I/O that bypasses the kernel's buffering of data. With direct I/O, data is transferred directly between the disk and the user application.

Dirty Region Logging

The procedure by which the VERITAS Volume Manager monitors and logs modifications to a plex. A bitmap of changed regions is kept in an associated subdisk called a *log subdisk*.

disk access name

The name used to access a physical disk, such as `c0t0d0s2`. The `c#t#d#s#` syntax identifies the controller, target address, partition, and disk. The term *device name* can also be used to refer to the disk access name.

disk array

A collection of disks logically and physically arranged into an object. Arrays provide benefits including data redundancy and improved performance.

disk cache

A section of RAM that provides a cache between the disk and the application. Disk cache enables the computer to operate faster. Because retrieving data from hard disk can be slow, a disk caching program helps solve this problem by placing recently accessed data in the disk cache. Next time that data is needed, it may already be available in the disk cache; otherwise a time-consuming operation to the hard disk is necessary.

disk group

A collection of disks that share a common configuration. A disk group configuration is a set of records containing detailed information on existing VERITAS Volume Manager objects (such as disk and volume attributes) and their relationships. Each disk group has an administrator-assigned name and an internally defined unique ID. The root disk group (`rootdg`) is a special private disk group that always exists.

disk name

A VERITAS Volume Manager logical or administrative name chosen for the disk, such as `disk03`. The term *disk media name* is also used to refer to the disk name.

DMP

See “Dynamic Multipathing.”

DSS

See “Decision Support Systems.”

Dynamic Multipathing

Dynamic Multipathing (DMP) is a VERITAS Volume Manager feature that allows the use of multiple paths to the same storage device for load balancing and redundancy.

error handling

Routines in a program that respond to errors. The measurement of quality in error handling is based on how the system informs the user of such conditions and what alternatives it provides for dealing with them.

evacuate

Moving subdisks from the source disks to target disks.

extent

A logical database attribute that defines a group of contiguous file system data blocks that are treated as a unit. An extent is defined by a starting block and a length.

extent attributes

The extent allocation policies associated with a file and/or file system. For example, see "address-length pair."

failover

The act of moving a service from a failure state back to a running/available state. Services are generally applications running on machines and failover is the process of restarting these applications on a second system when the first has suffered a failure.

file system

A collection of files organized together into a structure. File systems are based on a hierarchical structure consisting of directories and files.

file system block

The fundamental minimum size of allocation in a file system.

fileset

A collection of files within a file system.

fixed extent size

An extent attribute associated with overriding the default allocation policy of the file system.

fragmentation

Storage of data in non-contiguous areas on disk. As files are updated, new data is stored in available free space, which may not be contiguous. Fragmented files cause extra read/write head movement, slowing disk accesses.

gigabyte

Approximately one billion bytes. Also GB, Gbyte, G-byte.

high availability (HA)

The ability of a system to perform its function continuously (without significant interruption) for a significantly longer period of time than the combined reliabilities of its individual components. High availability is most often achieved through failure tolerance and inclusion of redundancy; from redundant disk to systems, networks, and entire sites.

hot backup

The process of backing up a database that is online and in active use.

hot pluggable

To pull a component out of a system and plug in a new one while the power is still on and the unit is still operating. Redundant systems can be designed to swap disk drives, circuit boards, power supplies, CPUs, or virtually anything else that is duplexed within the computer. Also known as hot swappable.

hot-relocation

A VERITAS Volume Manager technique of automatically restoring redundancy and access to mirrored and RAID-5 volumes when a disk fails. This is done by relocating the affected subdisks to disks designated as spares and/or free space in the same disk group.

inode list

An inode is an on-disk data structure in the file system that defines everything about the file, except its name. Inodes contain information such as user and group ownership, access mode (permissions), access time, file size, file type, and the block map for the data contents of the file. Each inode is identified by a unique inode number in the file system where it resides. The inode number is used to find the inode in the inode list for the file system. The inode list is a series of inodes. There is one inode in the list for every file in the file system.

intent logging

A logging scheme that records pending changes to a file system structure. These changes are recorded in an *intent log*.

interrupt key

A way to end or break out of any operation and return to the system prompt by pressing Ctrl-C.

kilobyte

One thousand bytes. For technical specifications, it refers to 1,024 bytes. In general usage, it sometimes refers to an even one thousand bytes. Also KB, Kbyte and K-byte.

kernel asynchronous I/O

A form of I/O that performs non-blocking system level reads and writes. This enables the system to handle multiple I/O requests simultaneously.

large file

A file more than two gigabytes in size. An operating system that uses a 32-bit signed integer to address file contents will not support large files; however, the Version 4 disk layout feature of VxFS supports file sizes of up to two terabytes.

large file system

A file system more than two gigabytes in size. VxFS, in conjunction with VxVM, supports large file systems.

latency

The amount of time it takes for a given piece of work to be completed. For file systems, this typically refers to the amount of time it takes a given file system operation to return to the user. Also commonly used to describe disk seek times.

load balancing

The tuning of a computer system, network tuning, or disk subsystem in order to more evenly distribute the data and/or processing across available resources. For example, in clustering, load balancing might distribute the incoming transactions evenly to all servers, or it might redirect them to the next available server.

load sharing

The division of a task among several components without any attempt to equalize each component's share of the load. When several components are load sharing, it is possible for some of the shared components to be operating at full capacity and limiting performance, while others components are under utilized.

Logical Unit Number

A method of expanding the number of SCSI devices that can be placed on one SCSI bus. Logical Unit Numbers address up to seven devices at each SCSI ID on an 8-bit bus or up to 15 devices at each ID on a 16-bit bus.

logical volume

See "volume."

LUN

See "Logical Unit Number."

master node

A computer which controls another computer or a peripheral.

megabyte

One million bytes, or more precisely 1,048,576 bytes. Also MB, Mbyte and M-byte.

metadata

Data that describes other data. Data dictionaries and repositories are examples of metadata. The term may also refer to any file or database that holds information about another database's structure, attributes, processing, or changes.

mirror

A duplicate copy of a volume and the data therein (in the form of an ordered collection of subdisks). Each mirror is one copy of the volume with which the mirror is associated.

The terms *mirror* and *plex* can be used synonymously.

mirroring

A layout technique that mirrors the contents of a volume onto multiple plexes. Each plex duplicates the data stored on the volume, but the plexes themselves may have different layouts.

mount point

The directory path name at which a file system attaches to the file system hierarchy.

multithreaded

Having multiple concurrent or pseudo-concurrent execution sequences. Used to describe processes in computer systems. Multithreaded processes are one means by which I/O request-intensive applications can use independent access to volumes and disk arrays to increase I/O performance.

NBU

See "VERITAS NetBackup (NBU)."

node

One of the hosts in a cluster.

object (VxVM)

An entity that is defined to and recognized internally by the VERITAS Volume Manager. The VxVM objects include volumes, plexes, subdisks, disks, and disk groups. There are two types of VxVM disk objects—one for the physical aspect of the disk and the other for the logical aspect of the disk.

OLTP

See “Online Transaction Processing.”

online administration

An administrative feature that allows configuration changes without system or database down time.

Online Transaction Processing

A type of system designed to support transaction-oriented applications. OLTP systems are designed to respond immediately to user requests and each request is considered to be a single transaction. Requests can involve adding, retrieving, updating or removing data.

paging

The transfer of program segments (pages) into and out of memory. Although paging is the primary mechanism for virtual memory, excessive paging is not desirable.

parity

A calculated value that can be used to reconstruct data after a failure. While data is being written to a RAID-5 volume, parity is also calculated by performing an *exclusive OR* (XOR) procedure on data. The resulting parity is then written to the volume. If a portion of a RAID-5 volume fails, the data that was on that portion of the failed volume can be recreated from the remaining data and the parity.

partition

The logical areas into which a disk is divided.

persistence

Information or state that will survive a system reboot or crash.

plex

A duplicate copy of a volume and its data (in the form of an ordered collection of subdisks). Each plex is one copy of a volume with which the plex is associated. The terms *mirror* and *plex* can be used synonymously.

preallocation

Prespecifying space for a file so that disk blocks will physically be part of a file before they are needed. Enabling an application to preallocate space for a file guarantees that a specified amount of space will be available for that file, even if the file system is otherwise out of space.

Quick I/O

Quick I/O presents a regular VERITAS File System file to an application as a raw character device. This allows Quick I/O files to take advantage of kernel-supported asynchronous I/O and direct I/O to and from the disk device, as well as bypassing the UNIX single-writer lock behavior for most file system files.

Quick I/O file

A regular UNIX file that is accessed using the Quick I/O naming extension (`::cdev:vxfs:`).

RAID

A Redundant Array of Independent Disks (RAID) is a disk array set up with part of the combined storage capacity used for storing duplicate information about the data stored in that array. This makes it possible to regenerate the data if a disk failure occurs.

repository

A repository holds the name, type, range of values, source, and authorization for access for each data element in a database. Pertinent information, needed to display configuration information and interact with the database, is stored in VxDBA's repository. The database maintains a repository for administrative and reporting use.

root disk

The disk containing the root file system.

root disk group

A special private disk group that always exists on the system. The root disk group is named `rootdg`.

root file system

The initial file system mounted as part of the UNIX kernel startup sequence.

script

A file, containing one or more commands that can be run to perform processing.

shared disk group

A disk group in which the disks are shared by multiple hosts (also referred to as a *cluster-shareable disk group*).

sector

A minimal unit of the disk partitioning. The size of a sector can vary between systems. A sector is commonly 512 bytes.

segment

Any partition, reserved area, partial component, or piece of a larger structure.

single threading

The processing of one transaction to completion before starting the next.

slave node

A node that is not designated as a master node.

slice

The standard division of a logical disk device. The terms *partition* and *slice* can be used synonymously.

snapped file system

A file system whose exact image has been used to create a snapshot file system.

snapped volume

A volume whose exact image has been used to create a snapshot volume.

snapshot

A point-in-time image of a volume or file system that can be used as a backup.

snapshot file system

An exact copy of a mounted file system, at a specific point in time, that is used for online backup. A snapshot file system is not persistent and it will not survive a crash or reboot of the system.

snapshot volume

An exact copy of a volume, at a specific point in time. The snapshot is created based on disk mirroring and is used for online backup purposes.

spanning

A layout technique that permits a volume (and its file system or database) too large to fit on a single disk to distribute its data across multiple disks or volumes.

Storage Checkpoint

An efficient snapshot technology for creating a point-in-time image of a currently mounted VxFS file system. A Storage Checkpoint presents a consistent, point-in-time view of the file system by identifying and maintaining modified file system blocks. The Storage Checkpoint facility is an enabling technology for Block-Level Incremental (BLI) Backup and Storage Rollback.

Storage Rollback

On-disk restore capability for faster recovery from logical errors, such as accidentally deleting a file. Because each Storage Checkpoint is a point-in-time image of a file system, Storage Rollback simply restores or rolls back a file or entire file system to a Storage Checkpoint.

stripe

A set of stripe units that occupy the same positions across a series of columns in a multi-disk layout.

stripe unit

Equally sized areas that are allocated alternately on the subdisks (within columns) of each striped plex. In an array, this is a set of logically contiguous blocks that exist on each disk before allocations are made from the next disk in the array.

stripe unit size

The size of each stripe unit. The default stripe unit size for VxVM is 32 sectors (16K). For RAID 0 stripping, the stripe unit size is 128 sectors (64K). For VxVM RAID 5, the stripe unit size is 32 sectors (16K). A *stripe unit size* has also historically been referred to as a *stripe width*.

striping

A layout technique that spreads data across several physical disks using stripes. The data is allocated alternately to the stripes within the subdisks of each plex.

subdisk

A consecutive set of contiguous disk blocks that form a logical disk segment. Subdisks can be associated with plexes to form volumes.

superuser

A user with unlimited access privileges who can perform any and all operations on a computer. In UNIX, this user may also be referred to as the “root” user. On Windows/NT, it is the “Administrator.”

tablespace

A tablespace is a storage structure (containing tables, indexes, large objects, and long data) that allows you to assign the location of database and table data directly onto containers. Tablespaces reside in database partition groups.

terabyte

Shorthand for 1,000,000,000,000 (10^{12}) bytes (or approximately 1000 GB).

throughput

A measure of work accomplished in a given amount of time. For file systems, this typically refers to the number of I/O operations in a given period of time.

UFS

The Solaris name for a file system type derived from the 4.2 Berkeley Fast File System.

unbuffered I/O

I/O that bypasses the file system cache for the purpose of increasing I/O performance (also known as *direct I/O*).

VERITAS Enterprise Administrator

Application that is required to access graphical user interface (GUI) functionality.

VERITAS File Replicator (VFR)

An enterprise data replication solution used to distribute Web or file server data. It enables multi-host processing and protects against critical data loss.

VERITAS NetBackup (NBU)

A product that lets you back up, archive, and restore files, directories, or raw partitions that reside on your client system.

VERITAS Volume Replicator (VVR)

A feature of VERITAS Volume Manager, VVR is a data replication tool designed to contribute to an effective disaster recovery plan.

Version Checkpoint

In a DB2 UDB EEE or ESE environment, a Version Checkpoint is a set of checkpoints that spans multiple partitions. A Version Checkpoint contains one Storage Checkpoint per partition. A Storage Checkpoint is an efficient snapshot technology for creating a point-in-time image of a currently mounted VxFS file system. A Storage Checkpoint presents a consistent, point-in-time view of the file system by identifying and maintaining modified file system blocks.

volume

A logical disk device that appears to applications, databases, and file systems as a physical disk partition. A logical disk can encompass multiple or one to many physical volumes.

volume layout

A variety of layouts that allows you to configure your database to meet performance and availability requirements. This includes spanning, striping (RAID-0), mirroring (RAID-1), mirrored stripe volumes (RAID-0+1), striped mirror volumes (RAID-1+0), and RAID 5.

volume manager objects

Volumes and their virtual components. See "object (VxVM)."

VFR

See "VERITAS File Replicator (VFR)."

VVR

See "VERITAS Volume Replicator (VVR)."

VxDBA

A VERITAS Storage Foundation *for DB2* menu-driven utility or graphical user interface (GUI) that helps you manage your database environment.

vxfs or VxFS

The acronym for VERITAS File System.

vxvm or VxVM

The acronym for VERITAS Volume Manager.

Index