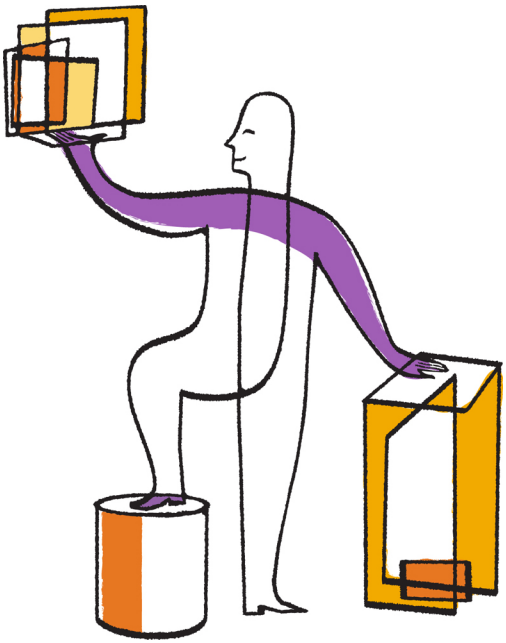
SnapManager® 7.1 for Microsoft® SQL Server®

Installation and Administration Guide



NetApp, Inc.
495 East Java Drive
Sunnyvale, CA 94089
U.S.

Telephone: +1 (408) 822-6000
Fax: +1 (408) 822-4501
Support telephone: +1 (888) 463-8277
Web: www.netapp.com
Feedback: doccomments@netapp.com

Part number: 215-08935_A0
June 2014

Contents

Understanding SnapManager 7
SnapManager overview 7
Terms and technologies 11
How SnapManager for Microsoft SQL Server works 14
How SnapManager works with other backup methods 17
AlwaysOn Availability Group support 17
Preparing to install or upgrade SnapManager 19
Preinstall or preupgrade procedure 19
Backing up system resources and data 20
Windows host system requirements 20
Preparing a Windows host system for SnapManager installation 22
SQL Server service account requirements 23
SnapManager service account requirements 23
SnapManager license requirements 24
Remote servers 25
Verifying storage system requirements 27
Installing or upgrading SnapManager 28
Installing SnapManager on a stand-alone Windows host system 28
System configurations for SnapManager on a Windows cluster using LUNs 33
Installing SnapManager in an existing Windows cluster 35
Upgrading SnapManager 36
Uninstalling SnapManager 40
Reinstalling SnapManager 43
Configuration and volume mount points 44
Preparing to Migrate SQL Server Databases 44
SQL Server configuration rules with SnapManager 44
SQL Server configurations supported with SnapManager 46
Understanding SnapManager support for volume mount points 54
Understanding the SnapManager GUI 56
SnapManager snap-in 56
Filters to help select databases backups 58
Starting SnapManager for the first time after installation 60

Starting SnapManager for the first time after installation 60
Using the SnapManager Configuration wizard 62
How databases are stored on storage system volumes 62
Understanding the Configuration wizard 63
Understanding control-file based configuration 65
Migrating SQL Server databases to LUNs, SMB shares, or VMDKs 80
Moving multiple SnapInfo directories to a single SnapInfo directory 81
Migrating SQL Server databases back to local disks 83
Setting up a SnapManager share for centralized backups of transaction logs 84
Preparing your environment for data protection 85
Preparing your environment to replicate backups 86
How SnapManager uses SnapMirror 86
Preparing your environment for SnapMirror replication 88
Preparing your environment to archive backups (clustered Data ONTAP) 89
Preparing your environment to archive backups (7-Mode) 90
Understanding dataset and SnapVault integration 90
Integrating dataset and SnapVault to SnapManager 92
Configuring datasets 93
Protecting local backups 95
Understanding SnapManager backup sets 97
How SnapManager backup works 97
How SnapManager backup data is organized 99
Types of backup operations performed using SnapManager 103
How SnapManager checks database integrity in backup sets 105
Ways to manage the number of backup sets kept online 109
When to run a SnapManager backup 111
Protecting databases by backing up, replicating, and archiving 113
How SnapManager backup functions are accessed 113
Backing up, replicating, and archiving databases using SnapManager 115
Managing Availability Group transaction log backups 126
Managing transaction log backups using SnapManager 126
What to do if a SnapManager backup operation fails 136
Performing database verification using SnapManager 138
Scheduling a backup job or a database verification job 145
Integrity verification on SnapMirror destination volumes and SnapVault secondary volumes 147

Configuring or changing verification settings 151
Using backup management groups in backup and verification 152
Explicitly deleting backup sets using SnapManager 154
Deleting archived backups 159
Restoring databases using SnapManager 160
Understanding SnapManager restore 160
How SnapManager Restore works 161
Types of SnapManager restore operations 163
Choosing the type of restore operation to perform 166
Understanding post restore database recovery states 166
Performing a restore operation 167
Retrieving and restoring remote backups 176
Deleting restored Snapshot copies 178
Restoring replicated publisher and subscriber databases 178
Cloning databases 180
Understanding database cloning 180
Cloning databases using SnapManager 180
Understanding cloned database lifecycles 188
Creating a clone replica of an AlwaysOn cluster 191
Using VMDKs with SnapManager for SQL Server 192
Setting up VMDK support 192
Backing up databases on VMDKs 193
Cloning databases on a VMDK 196
Performing disaster recovery of databases on VMDKs 196
Managing SnapManager operational reports 200
Understanding the SnapManager Reports option 200
Managing reports 200
Understanding monitoring and reporting 201
Recovering your SQL Server environment 204
Reseeding a database on an AlwaysOn cluster 204
Recovering SQL Server databases using SnapMirror 204
Recovering SQL Server databases using archives 210
Restoring a database on an AlwaysOn cluster 211
Restoring databases from other SQL Server backups 211
Restoring system databases from SnapManager backup sets 220
SnapManager command-line reference 222

Guidelines for using the command-line utility 222
clone-backup 223
clone-database 230
clone-replica 240
delete-backup 249
delete-clone 251
export-config 253
get-backup 255
import-config 257
new-backup 260
reseed-backup 270
restore-backup 275
verify-backup 283
Configuring SnapManager application settings 290
Overview of SnapManager application settings 290
Connecting to an SQL Server instance 292
Connecting to an AlwaysOn failover cluster node 293
Enabling SnapManager to allow databases on any LUN or VMDK configuration 294
Database integrity verification options 294
SnapManager backup options 298
SnapManager restore options 301
Precommand and postcommand script settings 303
Enabling or disabling database migration back to local disks 313
SnapManager report directory options 314
Event notification options 315
Managing fractional space reservation 318
Viewing fractional space reservation status 318
Configuring fractional space reservation policies 321
Copyright information 324
Trademark information 325
How to send your comments 326
Index 327

Understanding SnapManager

SnapManager overview

SnapManager provides an integrated data management solution for Microsoft SQL Server that dramatically boosts the availability and reliability of SQL Server databases. It is important for you to understand what SnapManager does and does not do and to know its components.

What SnapManager does

SnapManager provides rapid online backup and near-instantaneous restoration of databases by using online Snapshot technology that is part of the Data ONTAP software. SnapManager can also leverage the SnapMirror capabilities of storage systems to provide onsite or offsite SnapManager backup set mirroring for disaster recovery.

Data management: SnapManager supports the following data management capabilities:

- Migrating databases and transaction logs to storage systems
- Backing up databases and transaction logs to storage systems
- Verifying the backed-up databases and transaction logs
- Managing the SnapManager backup sets
- Restoring databases and transaction logs from previously created SnapManager backup sets

You can also create and restore databases from remote backups at a remote location through dataset and SnapVault integration to SnapManager.

Data replication for disaster recovery: When used with SnapMirror, SnapManager provides the ability to automatically replicate databases stored on the source volume to its mirrored target volume situated locally or remotely.

Dataset and SnapVault integration: SnapManager helps you create, restore, and manage remote backups. Dataset and SnapVault technologies together form the basis of this integration.

Data archiving for long-term or remote storage of backups: You can use SnapManager to create offline archives of Snapshot copies to unmanaged media for long-term retention. Three different archive methods are supported:

- Manually initiated archival using Network Data Management Protocol (NDMP) or the storage system's `dump` command
- Manually initiated archival using a Windows backup utility
- Automatic archival using the Run Command Operation feature with your backup operation

Federated full backup: You can use Snapshot copy-based full database backup to back up databases from different instances or different servers at the same time. You can add databases from multiple

servers to the same federated group. Backing up a federated group backs up all databases in that group at the same time.

Federated transaction log backup: Federated transaction log backup enables you to back up transaction logs from multiple servers at the same time, by setting marks on transaction logs of all databases in a backup group.

Restore to mark: Restore to marked transaction operations enable you to restore a database to a marked transaction. Using the marks created on a federated transaction log backup, you can restore databases to the same marked transaction across multiple databases for a synchronous restoration.

Enterprise monitoring and reporting: Enterprise monitoring and reporting enables you to report the status of backup, verification, and clone operations by sending application event log or email notifications.

Backup retention management: Configuring the number of transaction log backups your system retains enables you to balance up-to-the-minute restore needs with storage efficiency.

Clone database lifecycles: A cloned database lifecycle comprises automatic, scheduled cloned database resynchronization and deletion.

What SnapManager does not do

SnapManager for Microsoft SQL Server does not support the following uses:

- SnapManager does not create or restore backups of Microsoft SQL Server databases that are stored on storage devices that are provided by companies other than NetApp.
- SnapManager does not support filegroup backups or filegroup restores of Microsoft SQL Server databases.
- SnapManager does not support differential backup.
- Dataset integration with the NetApp Management Console data protection capability is not available for VMDK over NFS and VMFS datastore.
- SnapManager does not back up tempdb files, but it does back up all other system database files. However, if LUNs that contain system databases are included in a dataset and replicated to the SnapVault secondary volume, SnapManager does not restore the system database files from the secondary volume. You can restore the system database files manually by connecting the LUNs that contain the system databases.

Where you install and run SnapManager

You must install and run SnapManager on all SQL Server computers executing SnapManager operations.

In virtualized SQL Server deployments using VMDKs residing on either VMFS or NFS datastores, Virtual Storage Console for VMware vSphere must be installed (typically on the vCenter Server), and it must be accessible from within the guest system.

Local administration: When you run SnapManager on the computer hosting SQL Server, it is called *SnapManager local administration*. System requirements for a SnapManager local administration are described in *Verifying Windows host system requirements* on page 20.

Remote administration: When you run SnapManager on a computer that is not hosting SQL Server, it is called *SnapManager remote administration*. If you install SnapManager on a computer different from the SQL Server computer, you can run SnapManager remotely to perform any task that you can perform on a locally installed SnapManager system.

System requirements for a SnapManager remote administration system are described in *Remote servers* on page 25.

Remote verification: From a remote administration server that is configured with SnapDrive and SQL Server, you can also perform remote database verification. Remote verification offloads the CPU-intensive database verification operations that can affect the performance of your production SQL Server computer.

System requirements for a local and remote administration system used for remote verification are described in *Remote servers* on page 25.

The setup and use of a remote verification server is described in *Backing up databases using SnapManager* on page 113.

About the SnapManager graphical user interface (GUI)

The SnapManager for SQL Server user interface is a stand-alone graphical user interface based on the Microsoft Management Console 3.0 snap-in framework. The SnapManager GUI enables you to perform all the operations offered by SnapManager.

The GUI enables you to perform the following tasks:

- Manage and administer multiple instances of SnapManager successfully.
- Manage backup and restore operations of database files and transaction log files.
- Schedule backups and verify the integrity of databases in SnapManager backup sets.
- Administer SnapManager on another server computer on the network.

The user interface also enables you to schedule and automate backups and verify the integrity of databases in SnapManager backup sets.

SnapManager user interface includes the following components:

- Configuration Wizard including export and import
- Configuration Wizard Option Settings
- Backup Wizard
- Backup Settings
- Backup Verification Settings
- Clone Wizard
- Replica Wizard
- Restore Wizard

- Reseed Wizard
- Restore Setting
- Run Command Operation
- Fractional Space Reservation Settings
- Monitor Settings
- Notification Settings
- Report Directory Settings
- License Settings
- Delete Backup
- Reconnect Server
- Disconnect Server
- Debug Log Options
- View
- Refresh
- Help

Operations performed through the SnapManager command-line interface

SnapManager includes the following PowerShell cmdlets:

- new-backup
- verify-backup
- restore-backup
- get-backup
- delete-backup
- clone-database
- clone-backup
- delete-clone
- import-config
- export-config
- reseed-database
- clone-replica

For more information, see *SnapManager command-line reference* on page 222.

How you use SnapManager

You can run SnapManager on your SQL Server or on a different computer. When you run SnapManager on a different computer, it is called *remote administration*. Using a SnapManager remote administration system, you can perform all of the tasks that you perform on a locally installed SnapManager system. When you perform database verification on a remote system, it is referred to as *remote verification*.

The following actions describe a typical way to use SnapManager:

- After installing SnapManager, you use the SnapManager Configuration wizard to migrate the database to a storage system.

  This involves unmounting your databases and moving them to a storage system. The Configuration wizard ensures that your databases are placed correctly.
- After you configure data storage, you can use SnapManager Backup to create backups of the databases.
- If the need arises, you can use SnapManager Restore to restore your data (either entire groups of databases or individual databases) from one of the backups.

Using SnapManager's backup facility to begin SnapMirror through SnapDrive, you can create mirror replications of these databases to be used for various purposes, such as disaster recovery.

Terms and technologies

This section defines the terms and technologies referenced in this guide. Each term or technology is described within a SnapManager-specific context.

Availability Group
A part of the Microsoft SQL Server 2012 AlwaysOn feature that combines both database mirroring and log shipping capabilities to enable the following:

- Multidatabase failover
- Multiple secondaries
- Application failover using virtual names
- Readable secondary
- Backup from secondary

The databases as a group can move from one node to another node (failover) within the AlwaysOn set of nodes.

AlwaysOn
A disaster recovery solution available in Microsoft SQL Server 2012 and later. The solution provides both database-level and instance-level availability.

backup set
A backup set consists of metadata located in the SnapInfo directory structure and Snapshot copies. The Snapshot copies are created in volumes containing LUNs, SMB shares, and VMDKs used by databases that are contained in the backup set.

cluster group
A logical group of cluster resources that can be moved from one node to the other while the nodes remain operational. The cluster group can be moved by the administrator, or it can be moved as a result of a cluster resource failure.

database
A database is a collection of logical objects within a physical structure. The physical structure consists of one or more data files, and one or more transaction log files. A database is used either by the SQL Server itself (system database) or by an application (user database).

Database Consistency Checker (DBCC)
The Microsoft SQL Server utility for finding and correcting problems in the consistency of the database.

host system
A computer that accesses storage on a storage system.

log shipping
A process that takes backed-up transaction logs from a primary SQL Server and applies them sequentially on a scheduled basis to another SQL Server database. If a failure occurs, an application could be redirected to the other server, which would be only slightly behind the primary database. Log shipping is a means of protecting organizations if a logical or physical system failure occurs.

MSCS
Microsoft Cluster Services (MSCS) are system services that make it possible to create a virtual system consisting of multiple cluster nodes; each node is an independent physical computer and is a failover resource of other nodes in the cluster. Each node can support one or more virtual SQL Server instances.

multiple-instance cluster
A multinode cluster with multiple virtual SQL Server instances. Each node can be active, running one or more virtual SQL Server instances, or passive. The passive node is an idle system waiting for another node to fail over and thereby becoming an active node. If one system fails, the other system takes over its application services.

quorum disk
A shared disk resource that is used by MSCS to keep track of cluster management information, such as cluster resources and state. The quorum disk should not be used for SQL Server files. The quorum disk is a single point of failure.

recovery model
There are three distinct ways that you can recover your SQL Server databases if a failure occurs. Each model addresses a different need for performance, disk and tape space, and protection against data loss. The three models are summarized as follows:

Simple
Supports only database backup and not transaction log backup. Since there is no log backup, you cannot perform an up-to-the-minute restore operation. SnapManager for SQL Server supports only point-in-time restore operations for databases in simple recovery mode.

Full
All transactions are logged.

Bulk logged
Certain database operations (including SELECT INTO, BULK COPY/BCP, CREATE INDEX, WRITETEXT, and UPDATETEXT) are logged minimally. Database pages changed by committed bulk-logged operation are copied to the backed-up transaction log. The bulk logged model has a higher risk of data loss than the full recovery model.

For more information, see your Microsoft SQL Server documentation.

single-instance cluster (active/passive mode) Refers to an MSCS cluster with SQL Server installed, in which only one active instance of SQL Server is owned by a node and all other nodes of the cluster are in a standby state.

SQL Structured Query Language.

SQL Server A Microsoft relational database system based on the client-server database model.

SQL Server computer The hardware on which a Microsoft SQL Server database system is running.

SQL Server replication A process that is initiated and controlled by the database engine (SQL Server).

system database A type of database that is used internally by SQL Server. System databases are created either during installation or during feature configuration, such as the distribution database.

- **distribution database** A database on the distributor that stores data for replication, including transactions, Snapshot jobs, synchronization status, and replication history information. The database is created when replication is activated.
- **master database** Records the system-level information, SQL Server initialization information, and configuration settings for SQL Server. This database also records all login accounts and the mapping information from the name of a database to its primary file location.
- **tempdb database** A database that is used to fulfill all temporary storage needs, including stored procedures and tables. The tempdb database uses SQL Server during query processing and sorting, and for maintaining row versions used in Snapshot isolation. A clean copy of the tempdb database is re-created with its default size every time SQL Server is started.
- **model database** A template for all other databases on the system, including the tempdb database. When a database is created, the first part of it is created as a copy of the contents of the model database. The rest of the database is filled with empty pages. The model database must exist on the system because it is used to re-create tempdb every time SQL Server is started. You can alter the model database to include user-defined data types, tables, and so on. If you alter the model database, every database you create has the modified attributes.
- **msdb database** A database that holds tables that SQL Server Agent uses for scheduling jobs and alerts and for recording operators (those

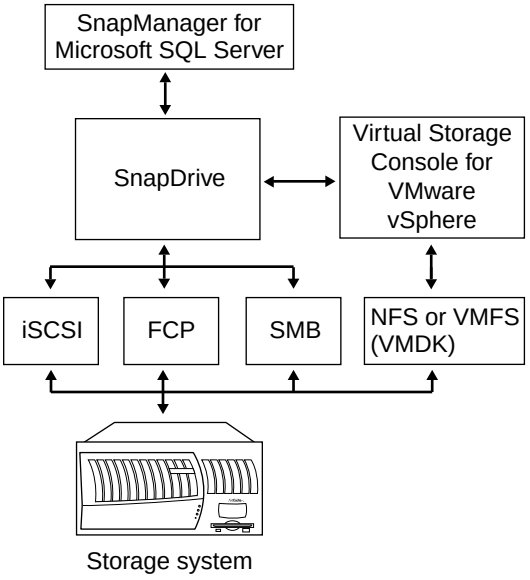
	assigned responsibility for jobs and alerts). This database also holds tables used for log shipping and for backup and recovery.
transaction log	A file that is used as a write-ahead log. All transactional operations are recorded in the transaction log; a transaction is considered committed when the "commit" transaction record has been written to the transaction log. The main purpose of the transaction log is for crash consistency; if there is a system crash, power failure, or similar disastrous event, then the transaction log has enough information to roll forward all committed transactions and roll back all noncommitted transactions.
user database	A database created for and used by an application is considered to be a user database.
Windows Server Failover Cluster (WSFC)	The set of servers (nodes) on which AlwaysOn is configured. The nodes do not share disks, and each node must have an SQL Server instance.

How SnapManager for Microsoft SQL Server works

System overview

SnapManager for Microsoft SQL Server is an SQL Server-aware application that provides backup and restore functionality in an SQL Server environment.

Relationship with other components of an SQL Server installation backed by a storage system
The following illustration shows the relationship between storage systems, SnapDrive, and SnapManager for Microsoft SQL Server.



How SnapManager and SnapDrive work together

SnapDrive provides the underlying layer of support for SnapManager by making storage available to a Windows Server host. SnapDrive software integrates with Windows Volume Manager so that storage systems can serve as virtual storage devices for application data in a Windows Server environment. It can also be used to provision storage for Windows virtual machines hosted on ESX hypervisors.

When to use SnapDrive You can use SnapDrive to automate storage provisioning tasks and manage data in SAN and SMB 3.0 Windows environments. For information about how to perform these tasks using SnapDrive, see the *SnapDrive Administration Guide* for your version of SnapDrive.

When to use SnapManager Use SnapManager to migrate SQL Server databases from a local disk to a LUN, SMB share, or VMDK and perform all operations on the databases.

Having SnapDrive installed on the SQL Server is a requirement for using SnapManager.

SnapManager and Snapshot copies

About SnapManager and Snapshot copies SnapManager uses Snapshot functionality to create real-time, online, read-only copies of databases. A SnapManager backup can consist of several Snapshot copies, depending on how your data is configured.

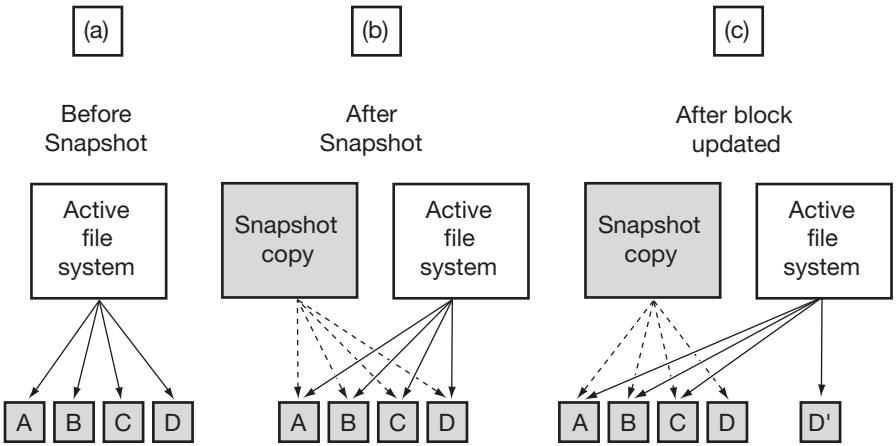
> **Note:** Always use SnapManager to manage SnapManager backups, rather than managing the backup sets using SnapDrive or storage system administration tools.

When to use the various Snapshot copy and backup methods

There are multiple ways to create Snapshot copies or backups in an installation that includes SnapManager. It is important to understand when each of these methods can produce a restorable image and when they cannot.

When one or more databases are mounted, Snapshot copy-based backups should be performed using only SnapManager. Creating Snapshot copies using the storage system console or another tool results in an inconsistent NTFS file system hosted by the LUNs, SMB shares, or VMDKs in the Snapshot copy. Using SnapDrive to create Snapshot copies creates inconsistent database images.

The following illustration provides an example of how Snapshot copies work.



Example

You make a Snapshot copy of a file named `file.txt` that spans four disk blocks in the active file system. Initially, the Snapshot version of `file.txt` and the version in the active file system are identical: the same blocks on the disk store both versions, so the Snapshot copy version of `file.txt` consumes no more disk space.

Now, you make a modification to `file.txt` that affects only one of the four disk blocks. The new data cannot overwrite the original block because that block is needed as part of the Snapshot copy. As a result, the new data is written to a new disk block and the file's inodes are updated accordingly. The active file system inodes now refer to the three original disk blocks that have not been modified since the Snapshot copy, plus the one new block. The Snapshot copy inodes still refer to the original four blocks.

If you delete `file.txt`, the blocks holding its data are no longer part of the active file system. The blocks still remain a part of the Snapshot copy. Deleting `file.txt` from the active file system does not free any disk space until the Snapshot copy is deleted.

Maximum number of Snapshot copies you can retain

Data ONTAP software allows a maximum of 255 Snapshot copies per storage system volume. Because SnapManager backups create Snapshot copies, you must delete old SnapManager backups

because they are no longer needed. Ensure you delete older backups to avoid reaching the limit of 255 Snapshot copies per storage system volume.

Note: The number of Snapshot copies on a volume can be greater than the number of SnapManager backups being retained. For example, if a single volume contains both the SnapInfo directory and the databases, each SnapManager backup generates two Snapshot copies on that volume.

How SnapManager works with other backup methods

SnapManager is a complement to conventional backup processes.

Supplementary backup archive: SnapManager backups are not intended to replace data archiving schemes in place for long-term or permanent data retention. Because SnapManager backups reside on primary disks, you should move your data to alternative media locations, such as secondary storage media. NDMP and the storage system `dump` command are the most efficient methods for creating archives.

SQL Server Management Studio backup utility: Because some types of third-party backup applications truncate transaction logs and interfere with the SnapManager recovery process, you should not perform transaction log backups with any application other than SnapManager.

What SnapManager does not back up: SnapManager does not back up all the files commonly used by an SQL Server computer. You can use Windows Backup (a native backup utility that ships with Windows) to back up the system state and the file systems on hard disks connected to the SQL Server.

Note: You can use Windows Backup to archive SnapManager backup sets to a file, instead of using tape, and store that file on a storage system.

How the storage system safeguards data

Under SnapManager, the database's data and log files reside on a LUN, SMB share, or VMDK created on the storage system and formatted with the New Technology File System (NTFS).

The storage system is a volume manager that stores LUNs, SMB shares, or VMDKs. The storage systems use battery-protected nonvolatile RAM (NVRAM) to protect incoming file system I/O operations. The contents of NVRAM are flushed to disk at regular intervals: more frequently if the NVRAM fills up, even during periods of inactivity. This ensures that the file system is always in a consistent state. The storage system guarantees that the contents of NVRAM are always written to disk, even during a power failure.

AlwaysOn Availability Group support

Starting in Microsoft SQL Server 2012, SQL Server provides you with integrated high availability and disaster recovery solutions through the AlwaysOn feature and enhanced availability for

databases and instances through AlwaysOn Availability Groups and AlwaysOn failover clusters. SnapManager for SQL Server simplifies how you create and manage backups of databases on servers in AlwaysOn Availability Groups.

Using SnapManager, you can do the following with Availability Groups:

- View Availability Group database information on the connected server
- Create and manage Availability Group-level backups
- Restore, reseed, and create clone replicas of Availability Group databases
- Perform up-to-the-minute restore operations to databases in the Availability Group from any backup copy available for that database in the cluster, if transaction log backups were saved to a centralized share

Note: You must install SnapDrive and SnapManager on all nodes of the AlwaysOn cluster.

Preparing to install or upgrade SnapManager

Preinstall or preupgrade procedure

Prerequisites for installing or upgrading SnapManager

Before you begin installing or upgrading SnapManager, you must complete the following tasks.

Task	Process	
1	Back up system resources and databases, as described in *Backing up system resources and data* on page 20.	
2	Determine whether you want to use per-SQL-server SnapManager licensing or per-storage-system SnapManager licensing. For more information, see *Verifying Windows host system requirements* on page 20.	
3	Configure or upgrade your storage system according to the requirements for SnapManager and SnapDrive, described in *Verifying storage system requirements* on page 27.	
4	If...	Then...
	You upgrade SnapManager and you also upgrade underlying *SnapDrive* or Microsoft *iSCSI initiator* versions	Make a note of this now. Later, while preparing to upgrade the SnapManager application (described in *Uninstalling SnapManager* on page 40), you must remove the iSCSI dependency with respect to SnapManager.
	You upgrade only SnapManager	Go to Step 5.
5	Note whether your storage system has multiple IP addresses.	
6	Configure or upgrade your Windows host systems to meet the requirements for SnapDrive and SnapManager, described in *Verifying Windows host system requirements* on page 20.	
7	Be sure that TCP port 808 is open for SnapManager to function.	

Task	Process
8	If you want to use VMDK on a virtual machine, complete the following substeps: **1.** Install Virtual Storage Console for VMware vSphere, as described in *Setting up VMDK support* on page 192. **2.** Set up the virtual machines and install SnapDrive for Windows and SnapManager for SQL on each virtual machine, as described in *Setting up VMDK support* on page 192. **3.** Create a VMDK from an NFS or VMFS datastore and attach it to the virtual machine, as described in *Setting up VMDK support* on page 192.
9	After you complete these tasks, you are ready to install or upgrade SnapManager. Go to *Installing or upgrading SnapManager* on page 28.

Backing up system resources and data

Backing up system resources and data

Before you install SnapManager, you are strongly advised to back up your system resources and data that uses Windows Backup or another industry standard backup utility.

To back up your system resources and data, complete the following steps.

Step	Action
1	Back up the operating system installation on the SQL Server, including the system state.
2	Back up the data on the local drives on the SQL Server.
3	Back up the boot and system drives, and the registry.
4	Use your backup utility to create and maintain a current emergency repair disk (ERD).

Windows host system requirements

In the most basic configuration, SnapManager is installed on the same Windows host system as SQL Server. In addition to this, you can install SnapManager on one or more remote Windows hosts for remote administration of the SQL Server computer or for remote verification of the databases contained in SnapManager backup sets.

The following table lists the minimum requirements for Windows host components. The versions listed for Windows Server, SnapIntegrator for Windows, SQL Server, and OnCommand Unified Manager Core Package are the minimum required versions at the time this document was published. For the latest supported versions, check the Interoperability Matrix.

Windows host component	**Minimum requirements**
Windows Server	• 2012 R2 (Standard and Datacenter) • 2012 (Standard and Datacenter) • 2008 SP2 (Standard, Enterprise, and Datacenter) • 2008 R2 SP1 (Enterprise)
SnapIntegrator for Windows	SnapIntegrator for Windows 7.0.3. For a remote administration server, SnapDrive is optional unless you intend to use the remote administration server to remotely administer SnapDrive. For a remote verification server, SnapDrive is required. If your storage system has multiple IP addresses, you must configure the SnapDrive preferred IP address. If you do not configure a SnapDrive preferred IP address for a storage system that has multiple IP addresses, SnapDrive times out when attempting to simultaneously create multiple Snapshot copies on a storage system. Refer to your SnapDrive documentation for more information.
SQL Server	• 2014 (Standard and Enterprise) • 2012 SP1 (Standard and Enterprise) • 2012 (Standard and Enterprise) • 2008 R2 SP2 (Enterprise and Datacenter) • 2008 SP3 (Standard, Enterprise, and Datacenter) • 2005 SP4 (Standard, Enterprise, and Datacenter)
OnCommand Unified Manager Core Package	OnCommand Unified Manager Core Package 5.2R1. OnCommand Unified Manager Core Package is required for SnapVault support for Data ONTAP operating in 7-Mode. The NetApp Management Console data protection capability must be licensed on the host. You should install the OnCommand Unified Manager Core Package on a dedicated host other than the SQL Server host.
SnapManager licenses	If SnapManager is licensed per-server, a SnapManager license is required on the Windows host system. For per-server SnapManager licensing, you can install SnapManager without specifying a server-side license; after SnapManager has been installed, you can apply the license from the License Settings dialog box.

Windows host component	Minimum requirements
Windows PowerShell	PowerShell 3.0. This is a prerequisite to SnapManager installation.
Microsoft Failover Cluster Automation Server	Microsoft Failover Cluster Automation Server is required when SnapManager is installed on a Windows Server 2012 cluster.

Related information

Interoperability Matrix: support.netapp.com/matrix

Preparing a Windows host system for SnapManager installation

Before you install SnapManager

Before you install SnapManager on a supported Windows host system, complete the following steps.

Step	Action
1	Back up your system resources using your current backup tool or another industry standard backup utility. **1.** Back up the operating system installed on the SQL Server, including the system state, the boot and system drives, and the registry. **2.** Back up your SQL Server databases and transaction log files. Use the Windows Backup utility that is part of the Windows operating system to create and maintain a current Emergency Repair Disk (ERD).
2	Be sure you understand all the installation and configuration steps needed to make SnapManager work in your particular environment. • If you plan to administer SnapManager *locally* from this host, then go directly to Step 3. • If you plan to administer SnapManager *remotely* from this host, review *Remote servers* on page 25 to determine your installation requirements. You do not need to install SnapDrive unless you want to use the remote administration server to remotely administer SnapDrive or you want to use the remote server for verifying backups. • If you plan to use SnapManager on this host only to perform *remote verification*, then review *Remote servers* on page 25 to determine your installation requirements.

Step	Action
3	If you need to install SnapDrive on this system, follow the instructions in the *SnapDrive Installation and Administration Guide* for your version of SnapDrive.
4	Install SnapManager according to the instructions in *How SnapManager for Microsoft SQL Server works* on page 14.

Note: Be sure that TCP port 808 is open for SnapManager to function.

SQL Server service account requirements

The Microsoft SQL Server service account must have permission to write to the file system and the Windows registry. These permissions are required because SnapManager initiates some SQL Server operations that need access to the file system and registry. Because of this requirement, we recommend that you run the SQL Server on a domain account.

Important: You need to change the default service account for SQL Server 2012 and SQL Server 2014 because it is a virtual account that does not have the required permissions. An example format for this virtual account is "NT Service\SQLAgent$INST01" where INST01 is the instance name or a service SID that has limited access to the file system and registry.

SnapManager service account requirements

When you install SnapManager, you specify the Windows account under which SnapManager will run (the SnapManager service account). This account must meet specific requirements.

The SnapManager service account must meet the following requirements:

- The account must have administrator privileges on the SQL Server computer.
- If you want to use Windows Authentication to connect SnapManager to your SQL Servers, the account must have the sysadmin role within each SQL Server instance installed on the server.
- If you want to archive backups to a SnapVault backup (Data ONTAP operating in 7-Mode only), the SnapManager service account must be one of the following:
  - The same account that you setup SnapDrive to use with the DataFabric Manager server (you set this up using the `sdcli dfm_config set` command).
  - A different account that is assigned a role on the DataFabric Manager server with the following capabilities:
    - DFM.DataBase.Read Global
    - DFM.DataSet.Write Global
    - DFM.Policy.Read Global
    - DFM.BackupManager.Backup Global
    - DFM.BackupManager.Read Global

- DFM.BackupManager.Restore Global

You can use the `dfm role create`, `dfm role add`, and `dfm user add` commands to create the role, add the capabilities, and create the user.
Alternatively, you can assign the SnapManager service account full control rights on the DataFabric Manager server. For example:

- Windows:

```
dfm user add -r GlobalFullControl MyDomain\snapuser
```

- UNIX:

```
dfm user add -r GlobalFullControl MyDomain\\snapuser
```

This requirement is necessary because SnapDrive uses the SnapManager service account to edit datasets and request backups.

Group Managed Service Accounts

The SnapManager installer supports Microsoft Windows group Managed Service Accounts. If you want to run SnapManager from a group Managed Service Account, then that account needs rights on the SQL Server. When you install SnapManager using a group Managed Service Account, you do not need to enter a password for the account.

SnapManager service account requirements in workgroup mode

To use SnapManager with Windows in workgroup mode, the SnapManager service account must be a local user account (not a domain account) that meets the requirements described in *SnapManager license requirements* on page 24. For instructions on how to configure SnapDrive in workgroup mode, see the *SnapDrive Installation and Administration Guide* for your version of SnapDrive.

SnapManager license requirements

SnapManager license requirements

SnapManager for Microsoft SQL Server requires that a SnapManager license be enabled on either the SQL Server or on the storage system.

Per-SQL Server license This license is for a specific SQL Server and provides capacity-based utilization for enterprise accounts. This type of license is also called a per-server license or a server-side license. With server-side licensing, no SnapManager license is required on the storage system.

If you are using a per-server SnapManager license, you can enter the license key while you are installing the SnapManager software (in the Customer Information screen of the SnapManager

software installation utility). You can also enter the license key later (from the License Setting action menu), after SnapManager is installed.

Per-storage-system license This license is for a specific storage system and enables you to use SnapManager on the storage system with any number of SQL Server instances and any database sizes. This type of license is also called a per-storage system license or a storage system-side license. If no server-side license is detected, SnapManager checks the storage system for a storage system-side license when a SnapManager operation is initiated. If a storage system-side license is not enabled, the SnapManager operation fails and an error message is written to the Windows event log.

Remote servers

Requirements for a remote administration server

An SQL Server instance that is running SnapManager can be remotely administered from another Windows system that is configured as a remote administration server:

- You do not need to install an iSCSI driver or an HBA driver on the remote administration server.
- SnapDrive does not need to be installed on the remote administration server unless you want to use it to remotely administer SnapDrive.
- SnapManager must be installed on the remote administration server.
- SnapDrive and SnapManager must be installed on the SQL servers that the administration server manages.
- The SnapDrive and SnapManager versions installed on the remote administration server must be the same as the versions on the servers that the administration server manages.

Note: Some limitations apply to using SQL Server authentication as the security authentication method to be used to establish the connection to a remote administration server. For more information, see *Connecting to an SQL Server instance* on page 292.

Note: An SQL Server used for database verification can be a virtual SQL Server.

You can add servers that you want to use through the option “Add servers to be managed” in the Actions pane. For more information, see *Connecting to an SQL Server instance* on page 292.

Requirements for a remote verification server

SnapManager performs remote verification using the same mechanisms used for local verification, except that the verification is performed on a host that is different from the SQL Server that initiated the backup. This is the reason that you need SnapDrive and SnapManager installed on your remote verification server, in addition to connectivity to the storage system.

To run remote database consistency checks, your remote Windows system must meet the following requirements:

- The remote Windows system must have connectivity to the storage system.

A remote verification server can connect to storage systems using a different protocol than the base configuration.

Note: If you are using iSCSI to connect to the storage system on the remote verification server, an iSCSI connection must be created.

- For LUNs:
  - The appropriate LUN driver (iSCSI or FC) must be installed.
  - The remote verification server must have the appropriate initiator to map the LUN.
  - If you have not created a LUN on the remote verification server, you can create an iSCSI session between the remote verification host initiator and the storage system using SnapDrive for Windows “iSCSI Management”.
- When the SQL server is hosted on a virtual machine with VMDK disks, the remote verification server must be a virtual machine.
- The verification virtual machine should also reside on the same vCenter server as the SQL Server virtual machine.
- If you are running verification on a SnapMirror destination volume, or cloning a database on SnapMirror destination volume, the remote virtual machine should not reside on the same vCenter server as the original SQL Server virtual machine.
- SnapDrive must be installed.

Note: Do not try to connect the SQL Server' LUNs to the remote SQL Server.

- SnapManager must be installed, but it does not need to be configured.
- You must specify the user account that you use for the production SQL Server.
- The SnapManager version and SnapDrive version on both the remote computer and host computer must be the same.
- Microsoft SQL Server must be installed.

  The version of SQL Server installed on the remote verification server must be either the same version and patch level as the source SQL Server or a later SQL Server version. For example, the source server is SQL 2008 and the remote verification server is SQL 2012. SnapManager for Microsoft SQL Server does not support using a remote verification server with a SQL Server version earlier than the source version.
- The SQL Server used for database verification can be a virtual SQL Server.

Note: If you cannot use a remote SQL Server instance or do not want to use a local SQL Server instance, you can select to not perform database verification. Alternatively, you can select to verify only online databases before or after a backup.

Note: Although it is possible to restore from an unverified backup, you should restore only from verified backups.

Note: You cannot use a remote physical server as the verification server for an SQL server that is running on a virtual machine.

Verifying storage system requirements

Storage system requirements

To be used with SnapManager, your storage system must meet the following requirements.

Storage system component	Requirements
Data ONTAP	See the SnapDrive software requirements described in the *SnapDrive Installation and Administration Guide* for your version of SnapDrive.
SnapManager license	A SnapManager license is required on the storage system only if you have licensed SnapManager on a per-storage-system basis.
Protocol licenses	• For LUNs, the iSCSI or FCP license • For SMB shares, the CIFS license • For VMDKs, the NFS license
FlexClone license	A FlexClone license is required if any of the following is true: • You want to use SnapManager to create database clones. • You have Data ONTAP operating in 7-Mode and you want to verify databases stored on destination SnapMirror volumes.
SnapRestore	For SnapManager restore operations to work properly, the SnapRestore feature of SnapDrive must be licensed on the storage system that stores the SQL Server databases.
SnapMirror license	If you plan to use the SnapMirror software along with SnapManager, a SnapMirror license is required on both the source and target storage systems. For more information, see the *SnapDrive Installation and Administration Guide* for your version of SnapDrive.
SnapVault license	If you want to archive your backups, you need SnapVault licenses on the primary and the secondary storage systems.

Installing or upgrading SnapManager

You install SnapManager on each Windows host running SQL Server software. Depending on your needs, you might also want to install SnapManager on a dedicated administration or verification server. You can install or upgrade SnapManager by using an interactive wizard or a command.

Important: SnapManager must be installed on every SQL Server node in a Windows Failover Cluster, whether or not the node writes to NetApp storage.

During installation, SnapManager configures the SQL Browser server to start automatically and installs the following Windows components:

- SQL 2005 backward compatibility components
- SQL Server 2012 or 2014 CLR types
- SQL Server 2012 R2 or 2014 Management Objects
- Microsoft Visual C++ 2012 Redistributable Package (x64)
- .NET Framework 4.0

Installing SnapManager on a stand-alone Windows host system

You can install SnapManager on any of the following stand-alone Windows host systems:

- The production SQL Server computer
- A remote administration server
- A remote verification server

You do not need to stop SQL Server instances before or during the SnapManager software installation process.

Modes of installing SnapManager

The software installation utility for SnapManager can be run in either interactive mode or unattended mode. These two modes are described in the following table.

Feature	SnapManager installation mode	
	Interactive	**Unattended**
Access	Requires user interaction and access to the user interface	Allows automated installation by executing a script or command line

Feature	SnapManager installation mode	
	Interactive	**Unattended**
Minimum required input	• SnapManager service account • User name • Password	
Optional input	• Organization name • SnapManager server-side license key • SnapManager installation directory	
After the installation finishes	If a system reboot is required to activate new software, a dialog box appears and prompts you to select whether you want to reboot the target system.	If a system reboot is required to activate new software, a dialog box appears and prompts you to choose whether you want to reboot the target system. You can override this default behavior by including an optional command line parameter.

Installing in interactive mode

To install SnapManager using the software installation utility in interactive mode, complete the following steps.

Step	Action
1	Download the software from the NetApp Support Site at *support.netapp.com* and then launch the installation program. **Attention:** Do not use Terminal Services for any type of SnapManager administration, because you might miss critical information that is displayed only in pop-up boxes at the system console.
2	In the Welcome screen, click **Next**. The installer checks the system for required Microsoft SQL Server components. If they are not present or current, the InstallShield wizard lists the items. You must click **Install** to continue the installation.

Step	Action
3	In the Customer Information screen, specify the user name, the organization name, and the SnapManager license type. See *Verifying Windows host system requirements* on page 20 for details about the two license types. If you have a storage system-side license for SnapManager, select **Per storage system**, and be sure that the SnapManager system-side license is enabled on the storage system. If you have an SQL Server-side license for SnapManager, select **Per Server** and use the License Key box to enter the license key for your server-side license. **Note:** If a server-side license key is unavailable during installation, you can leave the License Key text box empty. After the installation finishes, you can enter the license key through the SnapManager for Microsoft SQL Server GUI by right-clicking the server name and selecting **License Settings**.
4	Click **Next**. The Destination Folder screen appears.
5	In the destination folder, note the full path of the folder in which SnapManager is to be installed. The default installation directory for SnapManager for Microsoft SQL Server is as follows: `C:\Program Files\NetApp\SnapManager for SQL Server\`
6	(Optional) If you want to install SnapManager in a directory other than the default installation directory, do the following: 1. Click **Change** to open the Change Current Destination Folder dialog box. 2. Browse to an alternate installation directory. 3. Click **OK** to close the dialog box. The Destination Folder screen displays the new specified installation directory path. Record the newly specified installation directory path.
7	Click **Next**. The SnapManager Server Identity screen appears.

Step	Action
8	In the Account box of the SnapManager Server Identity screen, specify the user account you want to use to run SnapManager. For information about account requirements, see *SnapManager service account requirements* on page 23. If SnapDrive is installed and configured, the text box is populated with the account for which SnapDrive is configured. Otherwise, browse to find and select the account name. The user account name is specified in either of the following formats: • `DomainName\UserName` • `UserName@DomainName`
9	In the Password box and in the Confirm Password box, enter the user password. Leave the password blank if you entered a group Managed Service Account in the user name field.
10	Click **Next**. The Ready to Install the Program screen appears. All the installation specifications are complete.
11	Optional. Review or change your current installation specifications before proceeding by clicking **Back**.
12	To proceed with the installation using your current specifications, click **Install**. The installation process begins, and the Installing SnapManager for SQL Server screen appears. The screen displays the progress of the installation process.
13	If you are running SQL Server on virtual machine using VMDK, install SnapManager on the virtual machines so you can use SnapManager for SQL Server to back up and restore databases on those virtual machines.
14	After the InstallShield Wizard Completed screen appears, click **Finish** to exit the software installation utility.
15	Proceed to *Starting SnapManager for the first time after installation* on page 60.

After you finish

Fractional space reservation is disabled by default. If you are not using fractional space reservation on the storage system volumes that contain LUNs that are used for SQL Server, the monitoring can be disabled. Doing so improves backup completion times. For details on how to enable and disable monitoring, see *Managing fractional space reservation* on page 318.

Installing SnapManager in unattended installation mode

Installing SnapManager by running the software installation utility from a command line enables you to install the SnapManager software under the control of a script for an unattended installation. You

do not need to stop SQL Server instances before or during the SnapManager software installation process.

How to start the unattended installation: To install SnapManager in unattended mode, enter the following command either directly at the command line or through a script:

```
CommandName /s /v"/qn SILENT_MODE=1 [USERNAME=UserName]
[COMPANYNAME=CompanyName] [ISX_SERIALNUM=LicenseKey]
[INSTALLDIR=InstallationDirectory] SVCUSERNAME=Domain\UserName
SVCUSERPASSWORD=Password SVCCONFIRMUSERPASSWORD=Password
[/L*V DirPath\LogFileName]"
```

The following table describes each of the command parameters.

Command or parameter	Description
`CommandName`	The location and name of the executable
`SILENT_MODE=1`	Runs the installer in silent mode
`USERNAME=`*UserName*	Optional If not specified, the default value is retrieved from the registry.
`COMPANYNAME=`*CompanyName*	Optional If not specified, the default value is retrieved from the registry.
`ISX_SERIALNUM=`*LicenseKey*	Optional Used only to specify an SQL Server-side license for SnapManager.
`INSTALLDIR=`*InstallationDirectory*	Optional If not specified, the default installation directory is used: `C:\Program Files\NetApp\SnapManager for SQL Server\`
`SVCUSERNAME=`*Domain\UserName*	The account from which SnapManager is to be run For information about account requirements, see *SnapManager service account requirements* on page 23. Do not use the password parameters if you entered a group Managed Service Account in the user name field.
`SVCUSERPASSWORD=`*Password*	
`SVCCONFIRMUSERPASSWORD=`*Password*	

Command or parameter	Description
`/L*V DirPath\LogFileName`	Optional If you specify this option, detailed information about the installation is written to the specified log file. This information can be used to investigate details about how a particular instance of SnapManager for Microsoft SQL Server is installed. The asterisk (*) is a wildcard character specifying that all the installation information (such as status messages, nonfatal warnings, and error messages) is to be logged. *DirPath* is the fully qualified name of the directory in which the installation log is created or overwritten. *LogFileName* is the name of the file to which the transaction logs are written.

Example

```
"C:\NetApp\downloads\SMSQL7.1_x64.exe" /s /v"/qn SILENT_MODE=1
SVCUSERNAME=MKTG2\Administrator SVCUSERPASSWORD=****
SVCCONFIRMUSERPASSWORD=*** ISX_SERIALNUM=***"
```

After you finish

- Proceed to *Starting SnapManager for the first time after installation* on page 60.
- Fractional space reservation is disabled by default. If you are not using fractional space reservation on the storage system volumes that contain LUNs that are used for SQL Server, the monitoring can be disabled. Doing so improves backup completion times. For details on how to enable and disable monitoring, see *Managing fractional space reservation* on page 318.

System configurations for SnapManager on a Windows cluster using LUNs

System configurations for SnapManager on a Windows cluster

When installing SnapManager on an MSCS cluster, consider the following system configuration requirements and limitations.

Note: You do not need to stop SQL Server instances before or during the SnapManager software installation process.

SnapManager must be installed on all nodes: SnapManager must be installed on all nodes of the cluster so that SnapManager backup and restore operations can be performed from any node.

Installation on a virtual machine running the installer: If you are running the installer on a virtual machine using VMDKs, SnapDrive must be installed on that machine.

Maximum cluster size: SnapManager supports a maximum cluster size of eight nodes.

Multiple-instance clusters: SnapManager supports multiple-instance clusters, provided that the following additional system requirements are met:

- Each instance must have its own LUNs that cannot be used by other instances.
- Each instance must be created in its own *cluster group*.
- All LUNs assigned to a specified instance must be in the cluster group for that instance and in the SQL Server list of dependencies.

Disk requirements for SnapManager on a Windows cluster

In a clustered environment, SnapManager disk requirements vary, depending on the cluster configuration.

Single-instance cluster example: In an active/passive two-node configuration, there are two clustered nodes and one SQL Server instance. If the active node (the node running SQL Server) fails, the cluster transfers the SQL Server instance to the other (previously passive) node, which then becomes the active node and takes over the LUNs previously used by the failed node.

For a single-instance SQL Server cluster, if your SQL Server data is on a shared resource, your disk requirements are the same as for a stand-alone SQL Server system. A LUN is added for the quorum disk. A minimum of three LUNs are required:

- One LUN for the databases
- One LUN for the SnapInfo directory
- One LUN if a shared quorum disk is used

Multiple-instance cluster example: In an active/active two-node configuration, there are two clustered nodes and an SQL Server instance running on each node. If one node fails, the other node takes over the SQL Server instance running on the failed node. Because both nodes must be able to run an active SQL Server instance, each node requires its own disks, as if it were a self-contained, stand-alone system. In addition, one extra LUN is needed for the quorum disk, if a shared quorum disk is used. Whether you use a hard disk or a LUN as the quorum disk, each configuration requires a minimum of five disks used for the following purposes:

- For node 1
  - One LUN to store the SQL Server databases
  - One LUN to store the SnapInfo directory
- For node 2
  - One LUN to store the SQL Server databases
  - One LUN to store the SnapInfo directory

- One LUN or hard disk to be used as the quorum disk

Each node must be able to own all clustered disk resources in a cluster at any time.

For more information about MSCS clustering with SQL Server, see the *SQL Server 2005 Failover Clustering* document at the following URL: *www.microsoft.com/downloads/details.aspx?FamilyID=818234dc-a17b-4f09-b282-c6830fead499&displaylang=en*.

Installing SnapManager in an existing Windows cluster

Before installing SnapManager on an existing cluster, there are a number of aspects of the cluster you need to check.

To install and configure SnapManager in an existing Windows cluster, complete the following steps.

Note: Before you complete these steps, ensure that you have prepared your system and environment as described in *Preinstall or preupgrade procedure* on page 19.

Note: Before you complete these steps, review system configuration requirements and limitations for a Windows cluster in *System configurations for SnapManager on a Windows cluster* on page 33.

Step	Action
1	Verify that the virtual servers and the cluster services are functioning by moving the virtual server from one cluster node to the other and back. If any errors occur, or if any of the cluster resources do not start, resolve the issue before continuing.
2	From within the Windows Failover Cluster Manager GUI, run a Cluster Validation Report to ensure that the cluster is in a healthy state. If the report identifies any issues, resolve them before you continue.
3	Install or upgrade SnapDrive as required. For details, see your SnapDrive documentation.
4	From the node that owns the cluster group that contains the virtual server, create the shared LUNs to hold the databases and transaction log files. For details, see your SnapDrive documentation. **Note:** These shared LUNs must be physical disk resources in the cluster group that contains the virtual server that uses them.
5	Verify that the System Attendant Resource dependencies are set correctly. **Note:** If the Configuration wizard detects that it is running on a cluster, SnapManager adds the dependencies automatically for all the LUNs that it uses.

Step	Action
6	Verify that the virtual servers and the cluster services are functioning correctly by moving the cluster group containing the newly created virtual server to the other node and back.
7	Install SnapManager on all nodes, starting with the node that currently owns the cluster resources. Use either the interactive installation procedure or the unattended installation procedure for a stand-alone Windows host system. Both procedures are described in *Installing SnapManager on a stand-alone Windows host system* on page 28.
8	Go to *Installing or upgrading SnapManager* on page 28.

After you finish

Fractional space reservation is disabled by default. If you are not using fractional space reservation on the storage system volumes that contain LUNs that are used for SQL Server, the monitoring can be disabled. Doing so improves backup completion times. For details on how to enable and disable monitoring, see *Managing fractional space reservation* on page 318.

Upgrading SnapManager

About this section

This section contains procedures for upgrading SnapManager for Microsoft SQL Server on a Windows host system that is already running a version of SnapManager. Upgrading to SnapManager 7.1 for Microsoft SQL Server is supported from version 5.2.1 and later.

Note: You do not need to stop SQL Server instances before or during the SnapManager software upgrade process.

Pre-upgrade checklist

Before you upgrade to SnapManager for Microsoft SQL Server, be sure that your Windows host system and the storage system are running the supported software versions, as described in *Preparing to install or upgrade SnapManager* on page 19.

Comparison of the two upgrade modes

The software installation utility for SnapManager can be run in either interactive mode or unattended mode. Interactive mode requires user interaction and provides access to the user interface. Unattended mode allows automated upgrade by entering a command or executing a script.

Upgrading using interactive mode

To upgrade SnapManager using the software installation utility in interactive mode, complete the following steps.

Step	Action
1	Exit SnapManager, if you have not already done so. **Note:** You do not need to stop SQL Server instances before or during the SnapManager software upgrade process.
2	If you have not already done so, back up system resources by using an industry-standard backup utility. **1.** Back up the operating system, including the system state, the boot and system drives, and the registry. **2.** Back up your SQL Server databases and transaction log files. Use the Windows Backup utility to create and maintain a current emergency repair disk (ERD).
3	If you have not already done so, verify that your host system meets the minimum requirements. For details, see *Verifying Windows host system requirements* on page 20.
4	If you have not already done so, verify that your storage system meets the minimum requirements. For details, see *Verifying storage system requirements* on page 27.
5	Download the software from the NetApp Support Site at *support.netapp.com* and then launch the installation program. **Attention:** Do not use Terminal Services for any type of SnapManager administration, because you might miss critical information that is displayed only in pop-up boxes at the system console.
6	Click **Next**. The Program Maintenance screen appears.
7	In the Program Maintenance screen, leave the Modify/Upgrade option selected and then click **Next**. The SnapManager Server Identity screen appears.

Step	Action
8	In the SnapManager Server Identity screen, enter the user account you want to use to run SnapManager. For information about account requirements, see *SnapManager service account requirements* on page 23. Specify the user account name in either of the following formats: • *DomainName\UserName* • *UserName@DomainName*
9	In the Password box and in the Confirm Password box, enter the user password and then click **Next**. Leave the password blank if you entered a group Managed Service Account in the user name field.
10	In the Ready to Upgrade the Program screen, click **Upgrade**. The installation begins. When the installation finishes, the InstallShield Wizard Completed screen appears.
11	In the InstallShield Wizard Completed screen, click **Finish** to exit the software installation utility.

After you finish

If you are not using fractional space reservation on the storage system volumes that contain LUNs that are used for SQL Server, the monitoring can be disabled. Doing so improves backup completion times.

Upgrading in unattended mode

This topic describes how to upgrade SnapManager by running the software installation utility from a command line. This enables you to upgrade the SnapManager software under the control of a script for an unattended upgrade.

Step	Action
1	Access the command line of the target host system.
2	Exit SnapManager, if you have not already done so. **Note:** You do not need to stop SQL Server instances before or during the SnapManager software upgrade process.

Step	Action
3	At the command line, enter the following command either directly at the command line or through a script: **`CommandName /s /v"/qn REINSTALLMODE=vomus REINSTALL=ALL SILENT_MODE=1 /L*V DirPath\FileName SVCUSERNAME=Domain\UserName SVCPASSWORD=Password SVCCONFIRMUSERPASSWORD=Password"`**

The following table describes each of the upgrade command parameters.

Command or Parameter	Description
`CommandName`	The location and name of the executable
`REINSTALLMODE=vomus`	The type of reinstall to perform
`REINSTALL=ALL`	Reinstalls the entire product
`SILENT_MODE=1`	Runs the installer in silent mode
`SVCUSERNAME=`*`Domain\UserName`*	The account from which SnapManager is to be run For information about account requirements, see *SnapManager service account requirements* on page 23. Do not use the password parameters if you entered a group Managed Service Account in the user name field.
`SVCPASSWORD=`*`Password`*	
`SVCCONFIRMUSERPASSWORD=`*`Password`*	
`/L*V`*`DirPath\FileName`*	Optional If you specify this option, detailed information about the installation is written to the specified log file. This information can be used to investigate details about how a particular instance of SnapManager for Microsoft SQL Server is installed. The asterisk (*) is a wildcard character specifying that all the installation information (such as status messages, nonfatal warnings, and error messages) should be logged. *DirPath* is the fully qualified name of the directory in which the installation log is created or overwritten. *FileName* is the name of the file to which the installation information is written.

Example

```
SMSQL7.1_x64.exe" /s /v"/qn REINSTALLMODE=vomus REINSTALL=ALL SILENT_MODE=1
SVCUSERNAME=MKTG2\Administrator SVCUSERPASSWORD=password
SVCCONFIRMUSERPASSWORD=password"
```

After you finish

If you are not using fractional space reservation on the storage system volumes that contain LUNs that are used for SQL Server, the monitoring can be disabled. Doing so improves backup completion times.

Uninstalling SnapManager

Before you uninstall SnapManager

Note: Unless it is specified for a particular upgrade path or for a particular troubleshooting situation, you do not need to uninstall SnapManager before upgrading to a newer version. You do need to uninstall SnapManager if you want to reinstall it; otherwise, the installer repairs the existing installation.

If you have used SnapManager to manage your SQL Server databases and *you plan to reinstall* SnapManager later, be sure to *record the drive letter and path of the SnapInfo directory locations before you uninstall* SnapManager.

Single SnapInfo directory: If you have set up a single SnapInfo directory for all databases on this host, you can record the location of the storage that contains a single SnapInfo directory for all SQL Server instances and their associated databases.

Multiple SnapInfo directories: If you have set up multiple SnapInfo directories, you can record the following information:

- The location of the default SnapInfo directory for all SQL Server instances
- The location of the default SnapInfo directory for one or two LUNs shared by multiple databases (if configured)
- The location of the SnapInfo directory for an individual database (if configured)

SnapManager reports record the current SnapInfo directory locations in the most recent logs contained in the *Backup folder* and in the *Config folder*.

After you reinstall SnapManager, be sure to reconfigure SnapManager with the same SnapInfo directory locations that were used by SnapManager previously.

Attention: If you configure SnapManager with different SnapInfo directory locations than used previously, then SnapManager no longer has records of any backups made before the reinstallation of SnapManager occurred. As a result, your prior backup sets could be invalidated or deleted the next time you perform a backup operation.

Comparison of the two uninstallation modes

The software installation utility for SnapManager can be run in either interactive mode or unattended mode. These two modes are described in the following table.

Feature	**SnapManager uninstallation mode**	
	Interactive	**Unattended**
Access	Requires user interaction and access to the user interface You can also uninstall SnapManager by using the .exe file that you used to install the software.	Allows automated uninstallation by executing a script or command line
Tool used	The Windows utility Add or Remove Programs (in Control Panel)	The software installation utility for SnapManager for Microsoft SQL Server
Options	You can also remove the Report directory.	

Uninstalling SnapManager in interactive mode

To uninstall SnapManager for Microsoft SQL Server and all its components by using the Windows Add or Remove Programs utility, complete the following steps.

Step	**Action**
1	If SnapManager is running, stop it. You do not need to stop SQL Server or remove the SQL Server databases before you uninstall SnapManager. SQL Server continues to run during the uninstallation process and after the uninstallation finishes.
2	Use the Control Panel to uninstall SnapManager for Microsoft SQL Server.
3	At the prompt, click **Yes** to proceed with removing the SnapManager software.

Note: In a cluster configuration, be sure to uninstall SnapManager from all nodes of the cluster.

Uninstalling SnapManager in unattended mode

This topic describes how to uninstall SnapManager using the software installation utility in unattended mode. This enables you to uninstall SnapManager under the control of a script for an unattended uninstallation.

Step	**Action**
1	Access the command line of the target host system.

Step	Action
2	At the command line, enter the following command, either directly at the command line or through a script: `CommandName /v"REMOVE=ALL [REMOVEREPORTFOLDER=1] [/L* TempDirPath \LogFileName] /qb"`

Note: In a cluster configuration, be sure to uninstall SnapManager from all nodes of the cluster.

The following table describes each of the uninstallation command parameters.

Command or parameter	Description
`CommandName`	The location and name of the executable
`REMOVE=ALL`	Causes the software installation utility to remove SnapManager (as if you selected the Remove option in the Program Maintenance screen)
`REMOVEREPORTFOLDER=1`	(Optional) Causes the software installation utility to remove the Report folder (as if you selected the Remove Report Folder option in the Remove the Program screen)
`/L* TempDirPath\LogFileName`	(Optional) Writes detailed information about the installation to the specified log file This information can be used to investigate details about how a particular instance of SnapManager for Microsoft SQL Server is installed. The asterisk (`*`) is a wildcard character specifying that all the installation information (such as status messages, nonfatal warnings, and error messages) should be logged. *TempDirPath* is the fully qualified name of the directory in which the installation log is created or overwritten. *LogFileName* is the name of the file to which the installation information is written.

Example:

```
C:\NetApp\downloads\SMSQL7.1_x64.exe /s /v"REMOVE=ALL
[REMOVEREPORTFOLDER=1] /qb"
```

Reinstalling SnapManager

Reinstalling SnapManager

You can reinstall the same version of SnapManager on a Windows host system. This option fixes missing or corrupt files, shortcuts, and registry entries.

Note: You do not need to stop SQL Server instances before or during the SnapManager software reinstallation process.

Unless it is specified for a particular upgrade path or for a particular troubleshooting situation, you do not need to uninstall SnapManager before reinstalling it or upgrading to a newer version.

If you uninstalled SnapManager: If you uninstalled SnapManager, then the reinstallation procedure is identical to a new installation of the software. For installation instructions, see the following topics:

- *Installing SnapManager on a stand-alone Windows host system* on page 28
- *System configurations for SnapManager on a Windows cluster* on page 33

If you had used SnapManager to manage your SQL Server databases before you uninstalled the SnapManager application, then be sure to configure SnapManager with the same SnapInfo directory location or locations that were used by SnapManager before the reinstallation.

Attention: If you configure SnapManager with *different* SnapInfo directory locations than used previously, then SnapManager no longer has records of any backups taken before the reinstallation of SnapManager occurred.

For more information, see *Uninstalling SnapManager* on page 40.

If you did not uninstall SnapManager: See *Upgrading to SnapManager* on page 36.

Configuration and volume mount points

Preparing to Migrate SQL Server Databases

This section describes how to prepare to migrate your SQL Server databases so that you can manage backup and restore operations using SnapManager.

While preparing to migrate your databases, note the following:

- Secondary Availability Group database migration creates a stream-based backup in the SnapInfo folder.
- During a secondary database migration, there should not be any backups made on any node for that database.
- During primary database migration, secondary databases are also affected, so it is best not to perform any operation on the database until the migration completes.
- For all replicas wherever SnapManager is being used, Readable Secondary values should be set to **Yes**.
- Concurrent migration on different replicas of the same Availability Group at the same time is not advisable.
- During secondary database migration, the old database files are removed regardless of the value of the setting, **Delete old database files after migration**.

SQL Server configuration rules with SnapManager

About this section

This section outlines some of the rules governing configuration of SnapManager.

Maximum configurations supported by SnapManager

The following table lists the maximum configuration capacities tested and supported for the SnapManager environment.

Configuration capacity	Maximum
SQL Server instances per SQL Server computer or Windows cluster • Windows cluster • Stand-alone Windows host for SQL Server 2005, 2008, and 2012	 25 50
LUNs per SQL Server computer	165

Configuration capacity	Maximum
VMDKs per SQL Server computer	56
Databases per LUN or VMDKs	500
Databases per storage system volume	500
Databases across all SQL Server instances in a stand-alone SQL Server	5000
Databases across all SQL Server Failover Cluster Instances in a Windows Server Failover Cluster	2500
File groups per database	5000
Storage system volumes that can be used to store the following: • A single database • LUNs connected to an individual SQL Server computer • VMDKs connected to an individual SQL Server computer	 30 165 56

Although SnapManager does not prevent you from creating configurations that exceed these limits, such configurations are untested and unsupported. It is best that you do not exceed any of these limits.

SQL Server database configuration restrictions

The SnapManager Configuration Wizard enforces the following for migrating your SQL Server databases to LUNs or VMDKs.

Note: There are no restrictions for databases on SMB shares because they are file based.

Note: SnapManager only supports databases on SMB shares for SQL Server 2012 and later residing on clustered Data ONTAP 8.2 and later.

All the files belonging to a single database For any database that you migrate to LUNs or VMDKs for use with SnapManager, *all the files* (the data files, in addition to the transaction log files) must be migrated to LUNs or VMDKs.

Single database on multiple LUNs or VMDKs The files belonging to an individual database can be spread across two or more LUNs or VMDKs, if those LUNs or VMDKs are not used for storing database files belonging to other databases. This restriction does not apply if the unrestricted database layout option is enabled.

Note: User database files and SnapInfo directory cannot reside on the quorum disk.

No database files on the same LUN or VMDK as the SnapInfo directory You cannot migrate a database to the LUN or VMDK on which the SnapManager SnapInfo directory resides.

No database files or SnapInfo directory on a SAN boot LUN You cannot place database files or a SnapInfo directory on a SAN boot LUN (a LUN configured as a boot device for a SAN host).

No user database files on the LUN or VMDK that hosts the SQL Server You cannot migrate a user database to a LUN or VMDK that hosts the SQL Server.

SQL Server database configurations to avoid

If you add more databases or move databases to different LUNs, SMB shares, or VMDKs without using the Configuration wizard, you can create an invalid configuration that causes SnapManager backup or restore operations to fail.

Note: Always run the Configuration wizard after adding or moving SQL Server databases. The Configuration wizard ensures that the SQL Server databases are stored in valid locations so that SnapManager backup and restore operations can be completed successfully.

Before you migrate your SQL Server databases, note the following recommendations against certain invalid configurations that could be created outside the SnapManager Configuration Wizard.

No other files on LUNs, SMB shares, or VMDKs used by any database files Do not manually store any directories or files (including system paging files) on the LUNs, SMB shares, or VMDKs used for the SQL Server database files. These LUNs, SMB shares, or VMDKs should store only the SQL Server database files (data files and transaction log files) that are placed there by the SnapManager Configuration Wizard.

No other files on the LUN, SMB shares, or VMDK used by the SnapInfo directory Do not manually store any directories or files on the LUN, SMB shares, or VMDK used by the SnapInfo directory. The LUN, SMB shares, or VMDK should store only the directories or files that are placed there by the SnapManager Configuration Wizard.

SQL Server configurations supported with SnapManager

SnapManager databases can be configured on one or more storage systems. This section shows the various ways that you can place the data of your SQL Server on storage system volumes.

Note: If you change the database configuration, you might not be able to perform an up-to-the-minute restore with existing backups. Therefore, perform a backup immediately following any configuration changes.

Any LUN or VMDK configuration is supported if you enable the `unrestricted database layout` option. If you do not enable the `unrestricted database layout` option, you should use only the configurations listed in this section. For details about the `unrestricted database layout` option, see *Enabling SnapManager to allow databases on any LUN or VMDK configuration* on page 294.

If you place your databases on NetApp LUNs without using the configuration wizard, the databases might be in an invalid state if the `unrestricted database layout` option is not enabled and you use an unsupported configuration. When a database is in an invalid state, you cannot use SnapManager to back it up. To correct the issue, you either need to use the Configuration wizard to

move the databases to a supported configuration or you need to enable the `unrestricted database layout` option.

SQL Server configuration requirements for SnapVault backups

If you archive a database to a SnapVault backup (clustered Data ONTAP only), the database and SnapInfo directory must be on separate volumes.

SQL Server configuration requirements for SMB shares

- You can place any number of databases on the same SMB share and you can span a database across multiple SMB shares.
- A complete database must reside on any number of SMB shares.

  You cannot spread a database's files across LUNs and SMB shares.
- The SMB share name used by SQL Server database files must use the CIFS server name on the storage system, instead of the IP address of the management LIF or other data LIF.

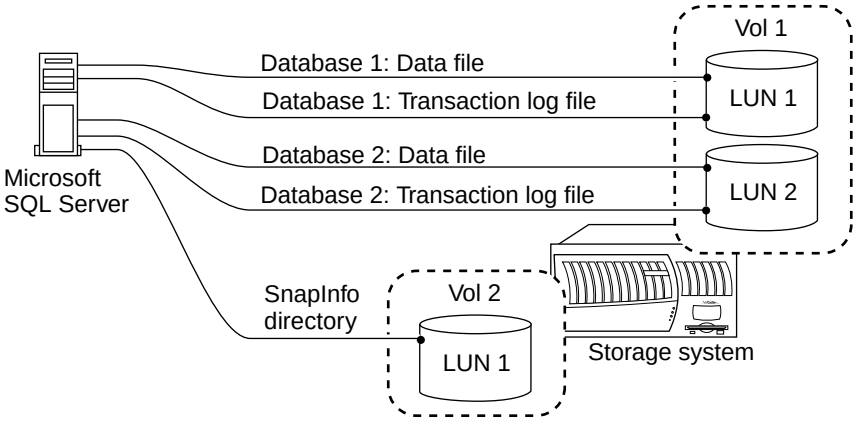
  SnapManager does not recognize a share by the CIFS server's IP address. It recognizes a share by the CIFS server's name.

  For example, the name of a CIFS server is FOX_VS01. The Storage Virtual Machine (SVM) also has an IPv6 data LIF called fd20-8b1e-b255-303---ac11-5b5.ipv6-literal.net. The database files need to use `\\FOX_VS01\sharename` as a file path to the share. The files cannot use `\\fd20-8b1e-b255-303--ac11-5b5.ipv6-literal.net\sharename` as a path to the share.

  If the database already uses a path with an IP address in the share name, manually detach the database, then attach the database using the SMB share path with the CIFS server name in its share name. Since both share paths point to same share, no files are moved.

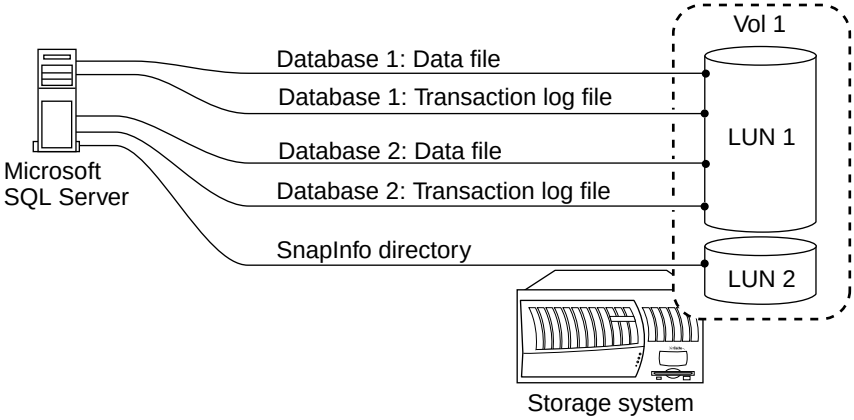
Multiple databases on different LUNs within the same volume

The following supported configuration shows multiple SQL Server databases sharing the same volume but residing on different LUNs.



Multiple databases on one LUN

The following illustration shows multiple SQL Server databases and all their associated files and transaction logs on one LUN.

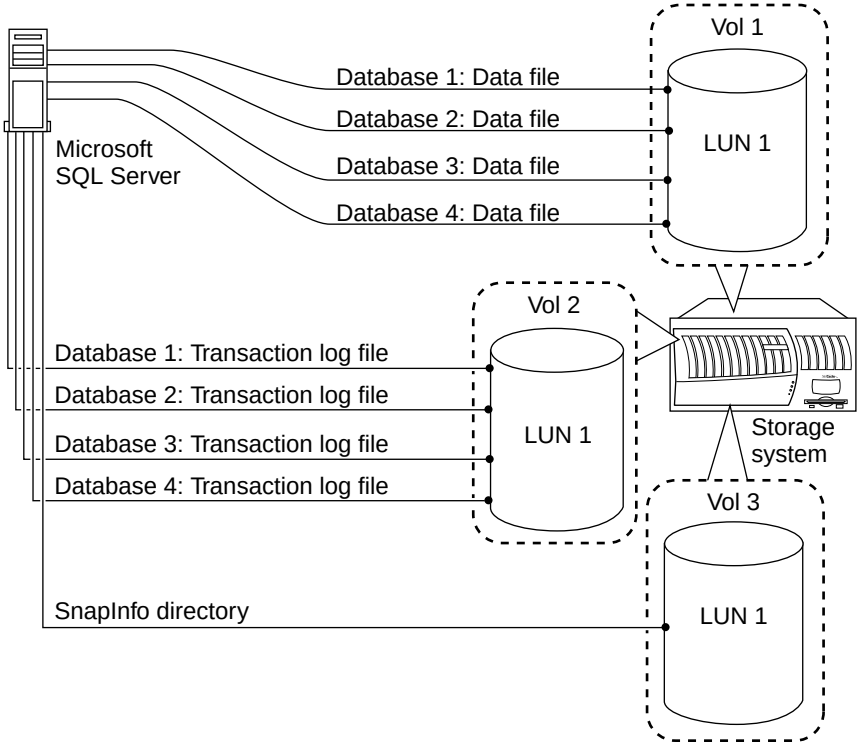


This is a simple configuration, and it can be applied to an SQL Server that supports about 35 databases per volume.

Note: In this configuration, all databases in the shared LUN are backed up at the same time, even if certain databases have not been selected for the backup. However, you have the option to select which databases you want to restore from a multiple-database backup.

Multiple databases sharing two LUNs

The following illustration shows an example of multiple SQL Server databases and all their associated files and transaction logs sharing exactly two LUNs. The database files cannot reside on any other LUNs if the unrestricted database layout option is not enabled. The LUNs can be located on the same or different storage system volumes. The illustration shows an example in which each LUN is located on a different volume.

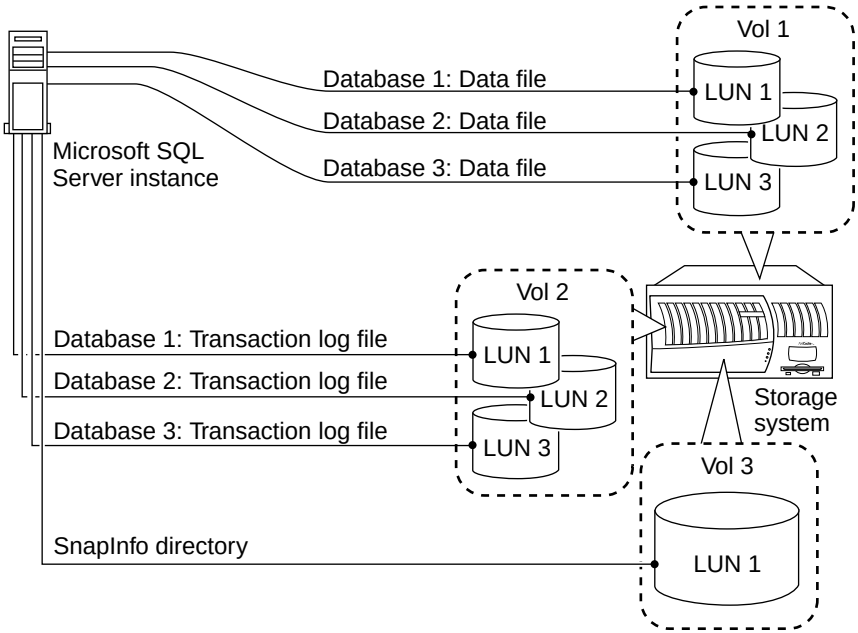


By placing the data files for multiple databases on one LUN and the transaction logs for those databases on the other LUN, SQL database performance is improved by separating the random I/O patterns of the data files from the sequential I/O patterns of the transaction log files.

Note: If you select to restore only a subset of the databases that reside on one or two LUNs shared by multiple databases, then a stream-based restore method is used rather than the online Snapshot restore method.

Single SQL Server and multiple storage system volumes

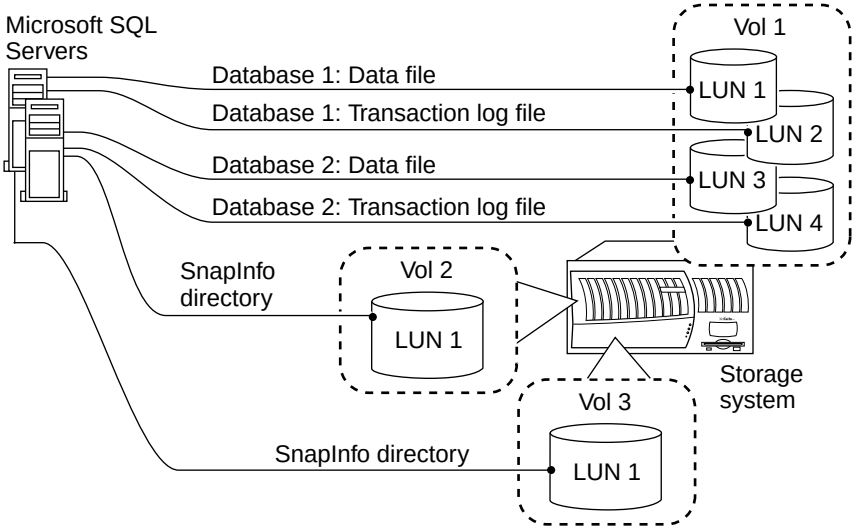
The following illustration shows a configuration in which the data and transaction log files of an SQL Server database reside on separate storage system volumes. Placing all transaction logs on one volume and using another volume for all the database files is partly due to performance. If the volume with the data files fails, it is still possible to back up the log file, restore the last full backup, and then apply all backed-up current transaction logs. This configuration requires another volume for the SnapInfo directory.



Multiple SQL Servers and one storage system volume

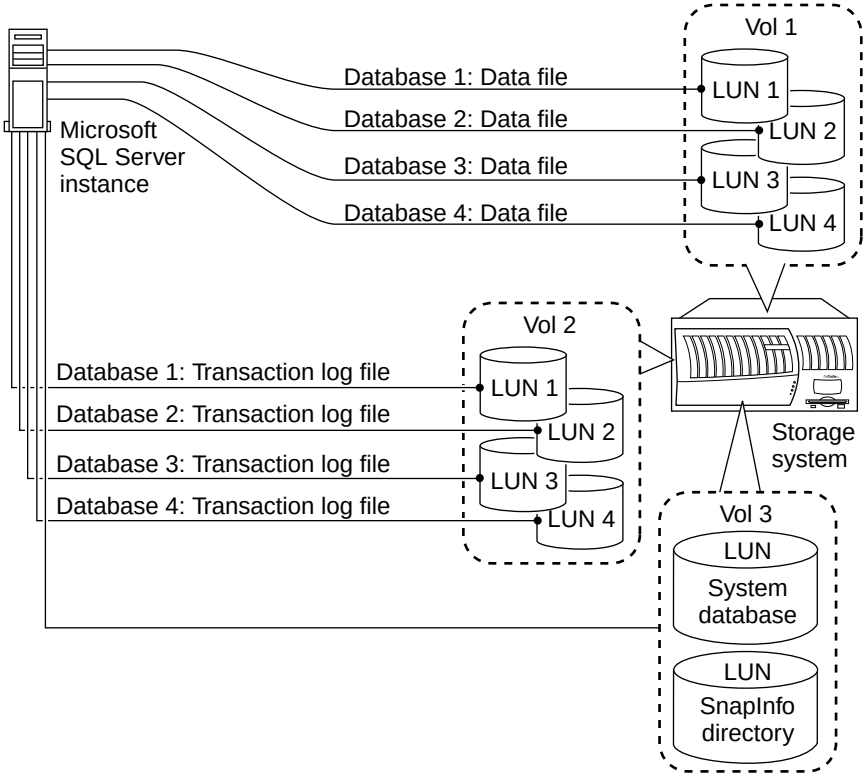
When the SQL Server environment does not generate high I/O load, a single volume can optimize the use of disk and volume space. However, this configuration has two disadvantages:

- If the volume fails, all databases are lost, including the current transaction log files.
- With a single volume housing databases for multiple SQL Server instances, there is an increased possibility of creating a busy Snapshot copy.
- For information about busy Snapshot copies, see "Busy Snapshot error prevents deletion of backup set" in *Explicitly deleting backup sets using SnapManager* on page 154.



Multiple SQL Server instances on the same storage system volume

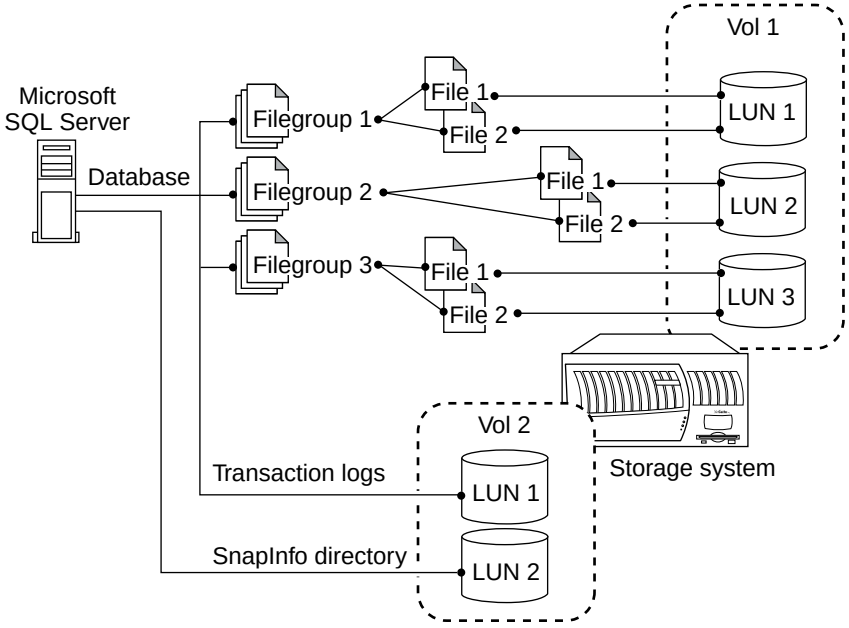
The following illustration shows a storage system volume with LUNs containing the data files of multiple SQL Server instances residing on a storage system volume that is different from the volume on which the LUNs for the transaction log files reside.



Note: Using three volumes prevents the loss of the complete SQL Server environment and makes it quicker to restore from unmanaged media if a volume goes offline.

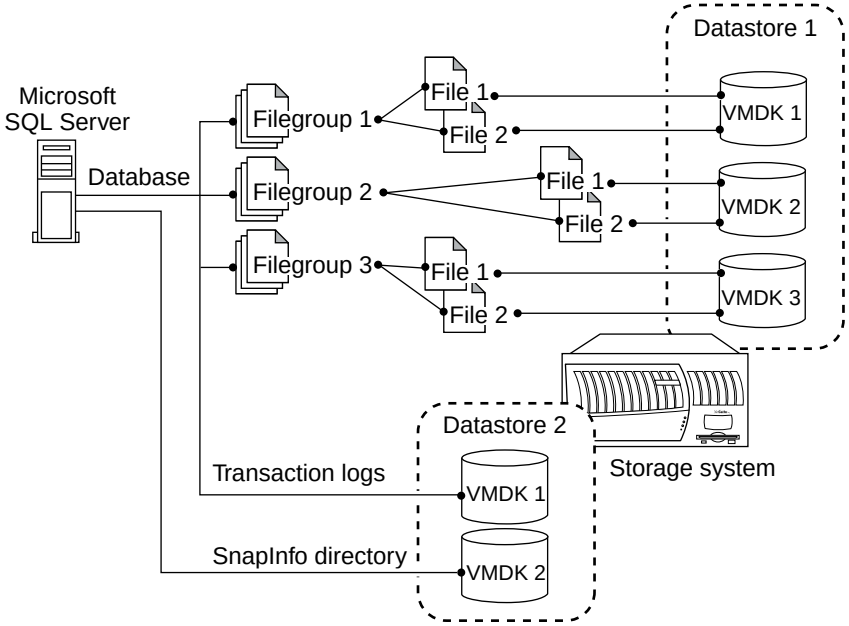
Multiple file groups belonging to the same database on different LUNs

The following illustration shows multiple file groups belonging to the same database residing on different LUNs within the same storage system volume.



Multiple VMDKs belonging to the same database on different volumes

The following illustration shows multiple file groups belonging to the same database residing on different VMDKs on the same datastores.



Understanding SnapManager support for volume mount points

It is useful for you to understand how SnapManager supports volume mount points.

Mounted volume restrictions with SnapManager

An NTFS volume that hosts mount points cannot support SQL Server databases. SnapManager imposes the following restrictions:

- SnapManager does not allow database files or database backup files to exist on an NTFS volume that has mount points.
- The mount point root LUN should not contain SQL database files or transaction log files.
  The mount point root directory can also exist on your local disk.

Using mounted volumes in SnapManager

The path-style representation of a mounted volume can appear in any part of the SnapManager user interface that refers to LUNs and VMDKs accessed by SnapManager:

- Configuration wizard pages that include an Available Disks selection are as follows:
  - Select a database, file, or file group to move to a LUN or VMDK.
  - Setup a SnapInfo directory for all databases

- Select a SnapInfo directory for each server instance

- **LUNs that are referenced more than once:** If the LUN or VMDK is configured with multiple references, each such LUN or VMDK reference has a label that lists any other references to the same LUN or VMDK.

  For example, suppose the *drive letter* `M:` and the *mount point* `C:\Mnt_Pnt\` reference the same LUN. In this case, the Available Disks selection contains two entries for one LUN:

  - `LUN M: <C:\Mnt_Pnt\>`
  - `LUN C:\Mnt_Pnt\ <M>`

  Swap LUNs using a reference mount point: If a database resides in LUN M, create a reference `C:\Mnt_Pnt1\db` to it using SnapDrive. You use the Configuration wizard to migrate the database from the original location LUN M to the reference `C:\Mnt_Pnt1\db` without copying or moving the database files. This operation is called *LUN swapping*.

  - Run the SnapManager for SQL Server Configuration wizard.

    The SnapManager configuration wizard lists all references to the same LUN. In this case, the Available Disks selection contains two entries:

    `LUN M: <C:\Mnt_Pnt1\db\>`

    `LUN C:\Mnt_Pnt1\db\ <M>`
  - Highlight the database on LUN M and click **Reconfigure**.
  - Select `LUN C:\Mnt_Pnt\ <M>` and associate it with the database.
  - Click **Next** to proceed and complete the Configuration wizard.

    The database is now attached to the `C:\Mnt_Pnt1\db` instead of M.

  LUNs that have mounted volumes: If SnapManager accesses a LUN with a volume that is referenced by a mount point, that LUN is listed with a label that indicates this.

  For example, suppose the drive letter `J:` references a LUN that hosts a mount point. In this case, the Available Disks selection lists that LUN as follows:

  `LUN J: (MPRoot)`

  The Configuration wizard does not allow you to store SQL database files on LUNs that host NTFS volume mount points.

- To specify which method is to be used to access database backup sets during database integrity verification, you use the Mount Point option to assign either a drive letter or select a mount point directory path to access the backup Snapshot copy as a mounted LUN.

  You can access this setting from the following locations within the SnapManager user interface:

  - **Configuration Wizard > Backup Verification Settings**
  - **Backup Wizard > Verification Settings**
  - **Restore Wizard > Verification Settings**

  For more information, see “Using the Mount Point tab” in *Database integrity verification options* on page 294.

Understanding the SnapManager GUI

SnapManager snap-in

The SnapManager snap-in is based on Microsoft Management Console 3.0 (MMC 3.0). MMC is a framework that hosts the graphical interface and programming platform to create, open, and save snap-ins.

The SnapManager snap-in allows you to manage the SnapManager application from Microsoft Management Console.

The user interface consists of three panes.

Scope pane This is the left pane in the SnapManager graphical user interface. It lists SnapManager instances. Elements that you select in this pane are automatically associated with possible actions in the Actions pane.

- If you select SnapManager in the Scope pane, all of the actions pertaining to it are displayed.
- If you select a specific server in the Scope pane, all the actions pertaining to it are displayed.
- If you select Backup in the Scope pane, all the actions pertaining to backup operations are displayed.
- If you select Restore in the Scope pane, all the actions pertaining to the restore operations are displayed.
- If you select Scheduled Jobs in the Scope pane, all the actions pertaining to the scheduled jobs are displayed.
- If you select Reports in the Scope pane, all the actions pertaining to reports are displayed.

Result pane This is the center pane in the SnapManager graphical user interface. It displays details of the type of instance that you select in the Scope pane.

Actions pane This is the right pane in the SnapManager graphical user interface. It displays all of the actions that you can perform, based on the instance that you select in the Scope pane.

Possible actions in the Actions pane include the following:

- Select the server and then click *Configuration Wizard* in the Actions pane.
  Use this wizard to configure SnapManager databases, transaction logs, and SnapInfo directories.
- Select the server and then click *Configuration Wizard Options Settings* in the Actions pane.
  Use it to enable or disable database to be migrated to the local disk.
- Select the server and then click *Backup Wizard* in the Actions pane.
  Use this wizard to back up SnapManager databases and transaction logs, as well as to verify databases in the backups.
- Select the server and then click *Backup Settings* in the Actions pane.

Use this option to specify backup settings for databases and transaction logs.

- Select the server and then click *Backup Verification Settings* in the Actions pane.

  Use this option to specify backup verification, mount point and DBCC settings for databases, and transaction logs.
- Select the server and then click *Clone Wizard* in the Actions pane.

  Use this option to clone existing backup sets and active production database. You can also delete clones and specify post restore settings.
- Select the server and then click *Replica Wizard* in the Actions pane.

  Use this option to create an Availability Group database replica using a clone from an existing backup set or an active production database to a remote server under the same AlwaysOn cluster. Once the clone replica completes, a new database replica in an existing availability group is created.
- Select the server and then click *Run Command Settings* in the Actions pane.

  Use this option to automatically run your own program or script before or after a backup or database verification operation, a clone operation, or a restore operation. When used with a backup operation, this is typically used to archive a backup automatically.
- Select *Delete backup* in the Actions pane and select the backups that you want to delete.

  Use this feature to delete SnapManager backups that you do not want to restore, based on the number of backups, the retention period, or the type of backups.
- Select the server and then click *Restore Wizard* in the Actions pane.

  Use this wizard to restore from backups created on same server, on a different server, or from archived backup.
- Select *Fractional Space Reservation Settings* in the Actions pane to monitor the space reservation in the LUN and to set policy settings for LUNs and volumes.

  Note: Fractional space reservation is not supported for VMDKs.

- Select *Notification Settings* in the Actions pane to configure SMTP email, storage system syslog, and AutoSupport for event notification.
- Select *Report Directory* Settings in the Actions pane to specify the report directory path for the server.
- Select *License Settings* in the Actions pane to update the SnapManager per-server license key.
- Select a server and then click *Disconnect Server* in the Actions pane to delete the Server from the SnapManager snap-in console.
- Select a server and then click *Reconnect Server* in the Actions pane to reestablish the connection to the server.
- Select *Restore Setting* to configure recovery, restore, replication, and transaction log backup settings.
- Select *Monitoring and Reporting* to configure email notifications on backup, verification, and clone operations.
- Select *Delete Clone* in the Actions pane to configure delete operation options.

SnapManager Dashboard view

The Dashboard view enables you to view the status of different SnapManager for servers connected to the SnapManager for network. This is a dynamic view that gets refreshed frequently. Dashboard allows you to:

- View the server configuration
- Add new servers

SMSQL Server Configuration-Server Name Click the server in the Scope pane to view Server Configuration details.

The following details are displayed in the Result pane:

- Name of the server instance
- Name of the host

  Note: In the case of clustered configurations, this value must display the host name of the node to which SnapManager is connected.

- Server version
- SnapManager version
- Name of the verification server

Recent Operations The following operations are listed:

- Last backup operation, including a time stamp and a hyperlink to the corresponding report
- Last restore operation, including a time stamp and a hyperlink to the corresponding report
- Last configuration operation, including a time stamp and a hyperlink to the corresponding report

Add new servers

You can add a new server from the Action pane and manage it through SnapManager. To add a new server, see *Connecting to an SQL Server instance* on page 292.

Filters to help select databases backups

When you need to select a server or database you might be able to use filters to help select the correct item; for example, using filters, you can schedule backup jobs on databases with specific properties such as being a primary.

Many actions and wizards provide a filtering capability that helps you select the correct item. The filter shows the relevant criteria, and the selections adjusts as you set and apply filter criteria; for example after selecting **Backup**, all of the databases on the SQL server are shown and the filter shows an SQL Server Instances pane and an AlwaysOn (For Microsoft Server SQL 2012) Availability Groups pane. If you check one of the Availability Groups listed in the filter and click **Filter**, only the databases on the selected server and the selected Availability Group display.

In addition to the dynamic display of information within panes, the filter panes also vary depending on the task; for example, if you select a database and the **Backup and Verify** action, the next screen has a new option, **Availability Group Backup**. If you click on **Availability Group Backup**, a new dialog box opens with a list of all of the Availability Groups present, and options **Preferred Backup Replicas Only** and **Advanced Option**. If you select **Advanced Option**, you then can use the **Replica Type** and **Backup Priority Number** filters.

Starting SnapManager for the first time after installation

Starting SnapManager for the first time after installation

Before you can use SnapManager to create and manage backups, you must connect it to SQL Server instances.

Steps

1. From the Start menu, select **Program Files > NetApp > SnapManager for SQL Server.**
2. In the Actions pane, click **Add Servers to be Managed**.
3. In the **Add SQL Instance to be managed** dialog box, do the following:
   a. Select the SQL Server from the list, type the name, or click the **Browse** button to select the server.

      Note the following:

      - If there is no default instance, specify one of the instance names (Server\Instance) instead of the server name. Even though you specify just one of the instances, SnapManager adds all of the instances on the server.
      - For an Availability Group, you can select any of the servers in the group.

   b. In the **Login Details** box, choose the authentication method that you want SnapManager to use to connect to the SQL Server.

      You can choose from the following two options:

Use Windows authentication	SnapManager connects to the SQL Server by using the Windows account under which SnapManager runs (the SnapManager service account). For more information about the SnapManager service account, see *SnapManager service account requirements* on page 23.
Use SQL Server authentication	SnapManager connects to the SQL Server by using an account defined on the SQL Server. The SQL Server administrator must have sysadmin server role privileges on the SQL Server instance. SnapManager requires that the database administrator have the required privileges to mount and unmount databases and backup and restore data and

transaction log files. The system administrator role fulfills all these permissions requirements.

4. Click **Add**.

   The message box closes and SnapManager launches the Configuration wizard.

5. Click **Cancel**.

Using the SnapManager Configuration wizard

How databases are stored on storage system volumes

About database storage

During the data migration process, the Configuration wizard enforces the following rules for storing your SQL Server database files and transaction log files on storage system volumes.

SQL Server database files

- You cannot spread a database's files across SAN and NAS.
- Database files that cannot be integrated to more than two LUNs or VMDKs cannot be used.

SQL Server transaction log files Transaction logs can reside on the same LUN or VMDK that stores the data files, or they can reside on another LUN or VMDK on the same or different volume. Transaction logs that belong to more than two LUNs or VMDKs cannot be used.

SnapInfo directory The SnapInfo directory must reside on a LUN or VMDK that is different from the LUN or VMDK on which the SQL Server data files and SQL Server transaction logs reside.

Note: This restriction does not apply to SMB shares.

Creating a SnapInfo directory When the Configuration wizard is used to migrate SQL Server databases from a local disk to LUNs, SMB shares, or VMDKs, the Configuration wizard creates a *SnapInfo directory* that stores SnapManager information about the backup sets and the backed-up transaction logs.

If you use the Configuration wizard to move databases, the wizard performs the following tasks:

Stage	Process
1	Detaches the selected databases: • Before the Configuration wizard migrates SQL Server user databases, it detaches them. • While the Configuration wizard is migrating SQL Server system databases, SnapManager stops the SQL Server. **Note:** Migrating SQL Server databases causes them to be taken offline during the move operation.

Stage	Process
2	Moves the SQL Server database files and transaction log files to the correct locations on the specified storage.
3	Reattaches the databases.
4	Brings user databases back online after the migration is complete.

Understanding the Configuration wizard

What the Configuration wizard does

The primary function of the Configuration wizard is to migrate SQL Server databases to LUNs, SMB shares, or VMDKs so that the databases can be backed up and restored using SnapManager. The Configuration wizard enables you to move your SQL Server databases in the following ways:

From local disk to LUN, SMB share, or VMDK This type of move enables management by SnapManager.

- If databases need to be moved, the wizard dismounts the databases, moves the database and transaction log files, and remounts the databases.

  Note: SnapManager takes databases offline during the move operation.

- The wizard creates a SnapInfo directory that SnapManager uses to store information about the backup sets and the backed-up transaction logs.
- The wizard also guides you through several application settings. These settings include enabling notification of SnapManager events using email, and enabling notification of SnapManager events using the storage system Syslog or the AutoSupport feature.

From LUN, SMB share, or VMDK to another LUN, SMB share, or VMDK You might want to make this move if resource management issues require it. For example, consolidating an SQL server on another storage system.

From LUN, SMB share, or VMDK to local disk Even if the databases are no longer managed using SnapManager, you can migrate them back to your local drive.

You can choose whether you want to verify your migrated databases and, if you are migrating your databases to LUNs, SMB shares, or VMDKs, whether you want to delete your original databases after a successful migration.

The Configuration wizard also guides you through several application settings. These settings include setting up the SnapInfo directory, enabling notification of SnapManager events using email, and enabling notification of SnapManager events using the storage system Syslog or the AutoSupport feature.

Note: If you are migrating ReportServer database from your local disk to a LUN, SMB share, or VMDK, ensure that SQL Server Agent and SQL Server Reporting Services are not running.

What the Configuration wizard does not do

Do not use the SnapManager Configuration wizard to migrate replicated databases or databases used in the replication process. For information about configuring replication-specific databases, see your Microsoft SQL Server documentation.

When to use the Configuration wizard

You can use the Configuration wizard in the following situations.

For initial configuration In order to use SnapManager to back up and restore SQL Server databases, you must use the SnapManager Configuration wizard to migrate those databases from your SQL Servers to the storage you configured using SnapDrive. The Configuration wizard also sets up a SnapInfo directory that SnapManager uses to store information about the backup sets and the backed-up transaction logs.

To view or change the database configuration After the initial configuration, you can rerun the Configuration wizard at any time to review or make changes to your SQL Server database configuration.

To validate the database configuration If you add more databases or move databases to different LUNs, SMB shares, or VMDKs, you should run the Configuration wizard to ensure that the databases are stored in valid locations and to create a mapping between those databases and their respective SnapInfo subdirectories.

Attention: Use the SnapManager Configuration wizard to move databases, transaction logs, or database system files. SnapManager ensures that these files are place in locations that meet SnapManager configuration requirements. Incorrectly located database, transaction logs, or database system files impair SnapManager operation. If some other method is used to move the database, transaction logs, or database system files, run the SnapManager Configuration wizard after the operation to ensure that these files are in correct locations.

When to re-run the Configuration wizard

You can back up newly created databases with an existing scheduled job without running the Configuration wizard first. If the scheduled backup job is created when all databases in a server are selected, or you do not specify any database in the `new-backup` cmdlet, the existing backup job can back up those newly created databases. See *new-backup* on page 260 for more information.

About SnapManager components

When you use the Configuration wizard, you are specifying the placement of several components of SQL Server and SnapManager.

Database data files The location of database files. The .mdf file is the primary database file. The .ndf file is the secondary database file.

Transaction log files The location of the transaction logs. The transaction logs contain changes made to the databases since the last backup, enabling an up-to-the-minute restore. The .ldf file is the transaction log file.

SnapInfo directory Contains SnapManager backup information, copies of transaction log backup files, and other data critical to the backup set.

Settings configurable only with the Configuration wizard

The following table lists the SnapManager database management settings that can be configured or changed only through the Configuration wizard. For each setting, the table lists the name of the corresponding screen in the Configuration wizard.

SnapManager setting	Configuration wizard screen
Database migration	Select a database, file group, or file to move
SnapInfo directory location	SnapInfo Directory Type • Single SnapInfo Directory • Advanced SnapInfo Directories
Migration-specific options: • Run DBCC before migration, after migration, or both • Delete original databases after successful migration	Database Migration Options
Microsoft iSCSI Service as a dependency	Add Microsoft iSCSI Service Dependency **Note:** If an FC or iSCSI hardware initiator is present on your system, then the option to add Microsoft iSCSI Service as a dependency is displayed as inactive.
SnapManager shares	Setup SnapManager Share

Understanding control-file based configuration

About the control file

This section describes how to use control-file based configuration to configure basic SnapManager settings. The control file is an XML file that contains SnapManager configuration information. The configuration data is represented in XML format. It can be edited manually.

Note: To avoid syntax errors, use an XML editor to edit control-file configuration.

You can access the control file option from the SnapManager Configuration wizard. You can use the control file as an alternative to the SnapManager Configuration wizard to configure SnapManager. This is especially useful in the following scenarios:

- Multiple SQL Server database servers, databases, and LUNs, SMB shares, or VMDKs
- Disaster recovery
- Mass deployment

The configuration settings contained in the control-file are grouped into the following sections:

- Storage Layout
- Notification settings
- Verification settings
- Report folder setting
- Backup settings
- Run Command Settings
- SnapMirror Volumes
- Scheduled Jobs
- Clone Scheduled Jobs
- Monitoring and Reporting Settings

Importing and exporting configuration settings

The following tasks can be performed using the control file:

- Export the current configuration details to a control file.
- Export a specific section of current configuration to a control file.
- Import configuration details from a control file.
- Import a specific section of configuration information from a control file.

To import or export configuration settings, complete the following steps:

Step	Action
1	If you have not already done so, start SnapManager by accessing the Windows Start menu, and selecting Program Files > NetApp > SnapManager for SQL Server. **Result** The SnapManager for SQL Server console appears.
2	In the Scope pane, double-click SnapManager for SQL Server. **Result** SnapManager displays the Status dashboard in the Result pane.
3	Click the SQL Server database server that you want to configure.
4	In the Actions pane, click Configuration wizard. **Result The Configuration wizard launches and the Welcome window appears.**

Step	Action
5	Select the "Use control-file" check box.
6	Click Next. **Result** The Import or Export Selection window appears.
7	Select either the Import or the Export option. Selecting the Import option will enable the Review current settings in the Configuration wizard check box. Select this check box to review imported settings in the configuration wizard. If you have selected the Import option and unchecked the Review settings in the configuration wizard, SnapManager will proceed to the normal configuration wizards for you to confirm the imported configuration settings. Selecting the Export option causes the Review current settings in the Configuration wizard check box to be grayed out. If you selected the Export option, SnapManager exports the current configuration and settings to the control-file.
8	In the Use control-file check box, select the control file path. SnapManager uses the default path `C:\Program Files\NetApp\SnapManager for SQL Server \SMSQLConfig_mm_dd_yyyy_hh.mm.ss.xml`.
9	Click Advanced.
10	In the Configuration Import/Export Advanced options window, specify the configuration settings that need to be imported or exported.
11	Click OK to confirm the configuration specification or Cancel to go back to the Import or Export Selection window. If SnapManager detects that there is some missing data in any of the selected options, it prompts you if you still want to carry out with the configuration.
12	Click Next to proceed. **Result** The Verification Settings screen appears.
13	Select the verification server and the connection to be used. The connection can be Windows Authentication or SQL Server Authentication.
14	If you selected SQL Server Authentication, enter the login name and password. **Result** SnapManager loads the control-file and validates the imported configuration and settings.

Sample XML schema for the control-file settings

The SnapManager schema file is distributed with the installation package. The following configuration file depicts the SnapManager control-file settings.

Storage layout settings The following schema depicts the storage layout settings section. You can edit the storage layout settings using an XML editor.

```
<?xml version="1.0" ?>
- <SMSQLCONFIG xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<HOST_NAME>SNAPMGR-19</HOST_NAME>
- <STORAGE_LAYOUT>
<MAX_DB_JOB>255</MAX_DB_JOB>
- <SQL_INSTANCES>
- <SQL_INSTANCE>
<SQL_INSTANCE_NAME>SNAPMGR-19</SQL_INSTANCE_NAME>
<SQL_INSTANCE_SNAPINFO_PATH>K:\SMSQL_SnapInfo</
SQL_INSTANCE_SNAPINFO_PATH>
<ADD_MSISIC_DEPENDENCY>false</ADD_MSISIC_DEPENDENCY>
- <DATABASES>
- <DATABASE>
<DATABASE_NAME>master</DATABASE_NAME>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>master</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\master.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>mastlog</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\mastlog.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>tempdb</DATABASE_NAME>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>tempdev</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
empdb.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>templog</FILE_NAME>
```

```
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
emplog.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>model</DATABASE_NAME>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>modeldev</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\model.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>modellog</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\modellog.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>msdb</DATABASE_NAME>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>MSDBData</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\MSDBData.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>MSDBLog</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\MSDBLog.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>DB1</DATABASE_NAME>
<SNAPINFO_PATH>K:\SMSQL_SnapInfo</SNAPINFO_PATH>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
```

```
<FILE_NAME>DB1</FILE_NAME>
<FILE_PATH>K:\MP\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB1.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>DB1_log</FILE_NAME>
<FILE_PATH>K:\MP2\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB1_log.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
- <DB_VOLUMES>
- <DB_VOLUME>
<FILER_NAME>rhine</FILER_NAME>
<VOLUME_NAME>grace2</VOLUME_NAME>
</DB_VOLUME>
</DB_VOLUMES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>DB3</DATABASE_NAME>
<SNAPINFO_PATH>K:\SMSQL_SnapInfo</SNAPINFO_PATH>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>DB3</FILE_NAME>
<FILE_PATH>I:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB3.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>DB3_log</FILE_NAME>
<FILE_PATH>I:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB3_log.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
- <DB_VOLUMES>
- <DB_VOLUME>
<FILER_NAME>rhine</FILER_NAME>
<VOLUME_NAME>grace1</VOLUME_NAME>
</DB_VOLUME>
</DB_VOLUMES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>DB2</DATABASE_NAME>
<SNAPINFO_PATH>K:\SMSQL_SnapInfo</SNAPINFO_PATH>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
```

```
- <DATABASE_FILE>
<FILE_NAME>DB2</FILE_NAME>
<FILE_PATH>K:\MP2\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB2.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>DB2_log</FILE_NAME>
<FILE_PATH>K:\MP\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB2_log.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
- <DB_VOLUMES>
- <DB_VOLUME>
<FILER_NAME>rhine</FILER_NAME>
<VOLUME_NAME>grace2</VOLUME_NAME>
</DB_VOLUME>
</DB_VOLUMES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>DB4</DATABASE_NAME>
<SNAPINFO_PATH>K:\SMSQL_SnapInfo</SNAPINFO_PATH>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>DB4</FILE_NAME>
<FILE_PATH>I:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB4.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>DB4_log</FILE_NAME>
<FILE_PATH>I:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB4_log.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
- <DB_VOLUMES>
- <DB_VOLUME>
<FILER_NAME>rhine</FILER_NAME>
<VOLUME_NAME>grace1</VOLUME_NAME>
</DB_VOLUME>
</DB_VOLUMES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>DB5</DATABASE_NAME>
<SNAPINFO_PATH>K:\SMSQL_SnapInfo</SNAPINFO_PATH>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
```

```
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>DB5</FILE_NAME>
<FILE_PATH>I:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB5.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>DB5_log</FILE_NAME>
<FILE_PATH>I:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\DATA
\DB5_log.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
- <DB_VOLUMES>
- <DB_VOLUME>
<FILER_NAME>rhine</FILER_NAME>
<VOLUME_NAME>grace1</VOLUME_NAME>
</DB_VOLUME>
</DB_VOLUMES>
</DATABASE>
</DATABASES>
</SQL_INSTANCE>
- <SQL_INSTANCE>
<SQL_INSTANCE_NAME>SNAPMGR-19\MARS</SQL_INSTANCE_NAME>
<SQL_INSTANCE_SNAPINFO_PATH>K:\SMSQL_SnapInfo</
SQL_INSTANCE_SNAPINFO_PATH>
<ADD_MSISIC_DEPENDENCY>false</ADD_MSISIC_DEPENDENCY>
- <DATABASES>
- <DATABASE>
<DATABASE_NAME>master</DATABASE_NAME>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>master</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\DATA
\master.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>mastlog</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\DATA
\mastlog.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>tempdb</DATABASE_NAME>
- <FILE_GROUPS>
- <FILE_GROUP>
```

```
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>tempdev</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\DATA
empdb.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>templog</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\DATA
emplog.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>model</DATABASE_NAME>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>modeldev</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\DATA
\model.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>modellog</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\DATA
\modellog.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
</DATABASE>
- <DATABASE>
<DATABASE_NAME>msdb</DATABASE_NAME>
- <FILE_GROUPS>
- <FILE_GROUP>
<GROUP_NAME>PRIMARY</GROUP_NAME>
- <DATABASE_FILES>
- <DATABASE_FILE>
<FILE_NAME>MSDBData</FILE_NAME>
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\DATA
\MSDBData.mdf</FILE_PATH>
</DATABASE_FILE>
</DATABASE_FILES>
</FILE_GROUP>
</FILE_GROUPS>
- <LOG_FILES>
- <LOG_FILE>
<FILE_NAME>MSDBLog</FILE_NAME>
```

```
<FILE_PATH>C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\DATA
\MSDBLog.ldf</FILE_PATH>
</LOG_FILE>
</LOG_FILES>
</DATABASE>
</DATABASES>
</SQL_INSTANCE>
</SQL_INSTANCES>
</STORAGE_LAYOUT>
```

Notification settings: The following schema depicts the notification settings section. You can edit the notification settings using an XML editor.

```
-<COMMON_SETTINGS>
-<NOTIFICATION>
-<SEND_EMAIL_NOTIFICATION>
<SMTP_SERVER>SNAPMGR-19</SMTP_SERVER>
<FROM>SMSQLAutoSender</FROM>
<TO>autosupport@netapp.com</TO>
<SUBJECT>SnapManager for SQL Server</SUBJECT>
<NOTIFY_AUTO>true</NOTIFY_AUTO>
<LONG_MSG>false</LONG_MSG>
<AS_ATTACHMENT>false</AS_ATTACHMENT>
<SEND_ON_FAILURE>true</SEND_ON_FAILURE>
</SEND_EMAIL_NOTIFICATION>
<EMS_ENABLED>true</EMS_ENABLED>
<ASUP_ENABLED>true</ASUP_ENABLED>
<ASUP_ON_FAIL>true</ASUP_ON_FAIL>
</NOTIFICATION>
```

Verification settings The following schema depicts the verification settings section. You can edit the verification settings using an XML editor.

```
-<VERIFICATION>
-<VERIFICATION_CLIENT_SETTING>
<VERIFICATION_SERVER>SNAPMGR-19</VERIFICATION_SERVER>
<VER_SERVER_NTAUTH>true</VER_SERVER_NTAUTH>
<VER_DBCC_NOINDEX>false</VER_DBCC_NOINDEX>
<VER_DBCC_ALL_ERROR_MSG>false</VER_DBCC_ALL_ERROR_MSG>
<VER_DBCC_NO_INFO_MSGS>false</VER_DBCC_NO_INFO_MSGS>
<VER_DBCC_TABLOCK>false</VER_DBCC_TABLOCK>
<VER_DBCC_PHYSICAL_ONLY>false</VER_DBCC_PHYSICAL_ONLY>
<VER_DBCC_ATTACH_DB>false</VER_DBCC_ATTACH_DB>
<VER_DBCC_BEFORE_MIGRATION>true</VER_DBCC_BEFORE_MIGRATION>
<VER_DBCC_AFTER_MIGRATION>false</VER_DBCC_AFTER_MIGRATION>
<VER_DELETE_DB_FILE_ORIG>true</VER_DELETE_DB_FILE_ORIG>
<VER_RUN_UPDATE_STATISTICS>true</VER_RUN_UPDATE_STATISTICS>
</VERIFICATION_CLIENT_SETTING>
-<VERIFICATION_SERVER_SETTING>
<AUTO_DRIVELETTER>true</AUTO_DRIVELETTER>
<MP_DIR>C:\Program Files\NetApp\SnapManager for SQL Server
\SnapMgrMountPoint</MP_DIR>
```

```
</VERIFICATION_SERVER_SETTING>
</VERIFICATION>
```

Monitoring directory settings The following schema depicts the monitoring directory settings. You can edit the monitoring directory settings using an XML editor.

```
- <MONITORING.>
<REPORT_BACKUP> true</REPORT_BACKUP>
<REPORT_CLONE>false</REPORT_CLONE>
<INTERVAL_HOURS>1</INTERVAl_HOURS>
<REPORT_CLOCK>23:15:00</REPORT_CLOCK>
</MONITORING>
```

Report directory settings The following schema depicts the report directory settings section. You can edit the report directory settings using an XML editor.

```
<REPORT_DIRECTORY>C:\Program Files\NetApp\SnapManager for SQL Server
\Report</REPORT_DIRECTORY>
```

Backup settings The following schema depicts the backup settings section. You can edit the backup settings using an XML editor.

```
-<BACKUP>
-<BACKUP_CLIENT_SETTING>
<NAMING_CONVENTION>0</NAMING_CONVENTION>
<BACKUP_SET_TO_KEEP>3</BACKUP_SET_TO_KEEP>
<BACKUP_SET_TO_KEEP_IN_DAYS>0</BACKUP_SET_TO_KEEP_IN_DAYS>
<LOG_BACKUP_SET_TO_KEEP>7</
LOG_BACKUP_SET_TO_KEEP><LOG_BACKUP_SET_TO_KEEP_IN_DAYS>0</
LOG_BACKUP_SET_TO_KEEP_IN_DAYS><DELETE_BACKUPS_OPTION>0</
DELETE_BACKUPS_OPTION>
<DELETE_LOG_BACKUPS_OPTION>0</
DELETE_LOG_BACKUPS_OPTION><BACKUP_SET_TO_VERIFY>0</BACKUP_SET_TO_VERIFY>
<BACKUP_SET_TO_KEEP_UTM>8</BACKUP_SET_TO_KEEP_UTM>
<BACKUP_SET_TO_KEEP_IN_DAYS_UTM/>
<DELETE_BACKUPS_OPTION_UTM>0</DELETE_BACKUPS_OPTION_UTM></
BACKUP_CLIENT_SETTING>
-<BACKUP_SERVER_SETTING>
<RUN_CMD_HOST>SNAPMGR-19</RUN_CMD_HOST> <RUN_CMD_PATH>notepad.exe</
RUN_CMD_PATH>
<RUN_CMD_ARGUMENT>$SqlSnapshot $InfoSnapshot</RUN_CMD_ARGUMENT>
</BACKUP_SERVER_SETTING>
</BACKUP>
```

SnapMirror volumes settings: The following schema depicts the SnapMirror relationship settings section. You can edit the SnapMirror relationship settings using an XML editor.

```
-<VERIFICATION_ON_DESTINATION>
-<SELECTED_DESTINATIONS>
-<SELECTED_DESTINATION>
<SOURCE_FILER>rhine</SOURCE_FILER>
<SOURCE_VOLUME>grace2</SOURCE_VOLUME>
<DESTINATION_FILER>rhine</DESTINATION_FILER>
<DESTINATION_VOLUME>grace2_mir</DESTINATION_VOLUME>
</SELECTED_DESTINATION>
-<SELECTED_DESTINATION>
<SOURCE_FILER>rhine</SOURCE_FILER>
<SOURCE_VOLUME>grace2</SOURCE_VOLUME>
<DESTINATION_FILER>rhine</DESTINATION_FILER>
<DESTINATION_VOLUME>grace2_mir</DESTINATION_VOLUME>
</SELECTED_DESTINATION>
</SELECTED_DESTINATIONS>
</VERIFICATION_ON_DESTINATION>
```

Schedule job settings The following schema depicts the schedule job settings section. You can edit the schedule job settings using an XML editor.

```
-<VERIFICATION_ON_DESTINATION>
-<SELECTED_DESTINATIONS>
-<SELECTED_DESTINATION>
<SOURCE_FILER>rhine</SOURCE_FILER>
<SOURCE_VOLUME>grace2</SOURCE_VOLUME>
<DESTINATION_FILER>rhine</DESTINATION_FILER>
<DESTINATION_VOLUME>grace2_mir</DESTINATION_VOLUME>
</SELECTED_DESTINATION>
-<SELECTED_DESTINATION>
<SOURCE_FILER>rhine</SOURCE_FILER>
<SOURCE_VOLUME>grace2</SOURCE_VOLUME>
<DESTINATION_FILER>rhine</DESTINATION_FILER>
<DESTINATION_VOLUME>grace2_mir</DESTINATION_VOLUME>
</SELECTED_DESTINATION>
</SELECTED_DESTINATIONS>
</VERIFICATION_ON_DESTINATION>
-<SCHEDULE_JOBS>
-<JOB>
<SCHEDULER>Windows Task Scheduler</SCHEDULER>
<JOB_NAME>bkup1</JOB_NAME>
<HOST_NAME>snapmgr-19</HOST_NAME>
<START_DIR>C:\Program Files\NetApp\SnapManager for SQL Server\</
START_DIR> <APPLICATION_NAME>C:\Program Files\NetApp\SnapManager for SQL
Server\SMSQLJobLauncher.exe</APPLICATION_NAME>

<COMMAND>new-backup ñsvr 'SNAPMGR-19' -db 'SNAPMGR-19', '8', 'DB1',
'DB2', 'DB3', 'DB4', 'DB5', 'master', 'model', 'msdb',
'SNAPMGR-19\MARS', '3', 'master', 'model', 'msdb' -ver ñversvr
'SNAPMGR-19' -del -rtbkups 2 -lgbkafbk -noutm -uniq ñmgmt standard</
COMMAND>
<START_TIME>11/6/2007 1:32:00 PM</START_TIME>
-<SCHEDULES>
-<WEEKLY_TRIGGERS>
```

```
-<WEEKLY_TRIGGER>
-<TASK_TRIGGER>
<TriggerSize>48</TriggerSize>
<Reserved1>0</Reserved1>
<BeginYear>2007</BeginYear>
<BeginMonth>10</BeginMonth>
<BeginDay>27</BeginDay>
<EndYear>0</EndYear>
<EndMonth>0</EndMonth>
<EndDay>0</EndDay>
<StartHour>13</StartHour>
<StartMinute>32</StartMinute>
<MinutesDuration>0</MinutesDuration>
<MinutesInterval>0</MinutesInterval>
<Flags>0</Flags>
<Type>TIME_TRIGGER_WEEKLY</Type>
-<Data>
-<daily>
<DaysInterval>1</DaysInterval>
</daily>
-<weekly>
<WeeksInterval>1</WeeksInterval>
<DaysOfTheWeek>4</DaysOfTheWeek>
</weekly>
-<monthlyDate>
<Days>262145</Days>
<Months>0</Months>
</monthlyDate>
-<monthlyDOW>
<WhichWeek>1</WhichWeek>
<DaysOfTheWeek>4</DaysOfTheWeek>
<Months>0</Months>
</monthlyDOW>
</Data>
<Reserved2>0</Reserved2>
<RandomMinutesInterval>0</RandomMinutesInterval>
</TASK_TRIGGER>
</WEEKLY_TRIGGER>
</WEEKLY_TRIGGERS>
</SCHEDULES>
</JOB>
-<JOB>
<SCHEDULER>SQL Server Agent</SCHEDULER>
<JOB_NAME>bkupSqlAgt</JOB_NAME>
<HOST_NAME>SNAPMGR-19</HOST_NAME>
<START_DIR>C:\Program Files\NetApp\SnapManager for SQL Server\</
START_DIR> <APPLICATION_NAME>C:\Program Files\NetApp\SnapManager for SQL
Server\SMSQLJobLauncher.exe</APPLICATION_NAME>
<COMMAND>ackup ñsvr 'SNAPMGR-19' -db 'SNAPMGR-19', '8', 'DB1', 'DB2',
'DB3', 'DB4', 'DB5', 'master', 'model', 'msdb', 'SNAPMGR-19\MARS', '3',
'master', 'model', 'msdb' -ver ñversvr 'SNAPMGR-19' -del -rtbkups 2 -
lgbkafbk -noutm -uniq ñmgmt standard</COMMAND>
<START_TIME>11/7/2007 1:00:00 AM</START_TIME>
-<SQLAGENTSCHEDULES>
<START_DATE_TIME>11/5/2007 12:00:00 AM</START_DATE_TIME>
<START_TIME_OF_DAY>01:00:00</START_TIME_OF_DAY>
<END_DATE_TIME>12/31/9999 12:00:00 AM</END_DATE_TIME>
```

```
<END_TIME_OF_DAY>23:59:59</END_TIME_OF_DAY>
<FREQUENCY_TYPE>Daily</FREQUENCY_TYPE>
<FREQUENCY_INTERVAL>1</FREQUENCY_INTERVAL>
<FREQUENCY_SUBDAY_TYPE>Once</FREQUENCY_SUBDAY_TYPE>
<FREQUENCY_SUBDAY_INTERVAL>0</FREQUENCY_SUBDAY_INTERVAL>
<FREQUENCY_RELATIVE_INTERVAL>First</FREQUENCY_RELATIVE_INTERVAL>
<FREQUENCY_RECURRENCE_FACTOR>0</FREQUENCY_RECURRENCE_FACTOR>
</SQLAGENTSCHEDULES>
</JOB>
-<JOB>
<SCHEDULER>SQL Server Agent</SCHEDULER>
<JOB_NAME>bkupSqlAgtMars</JOB_NAME>
<HOST_NAME>SNAPMGR-19\MARS</HOST_NAME>
<START_DIR>C:\Program Files\NetApp\SnapManager for SQL Server\</
START_DIR> <APPLICATION_NAME>C:\Program Files\NetApp\SnapManager for SQL
Server\SMSQLJobLauncher.exe</APPLICATION_NAME>
<COMMAND>backup ñsvr 'SNAPMGR-19' -db 'SNAPMGR-19', '8', 'DB1', 'DB2',
'DB3', 'DB4', 'DB5', 'master', 'model', 'msdb', 'SNAPMGR-19\MARS', '3',
'master', 'model', 'msdb' -ver ñversvr 'SNAPMGR-19' -del -rtbkups 2 -
lgbkafbk -noutm -uniq ñmgmt standard</COMMAND>
<START_TIME>11/11/2007 2:00:00 AM</START_TIME>
-<SQLAGENTSCHEDULES>
<START_DATE_TIME>11/5/2007 12:00:00 AM</START_DATE_TIME>
<START_TIME_OF_DAY>02:00:00</START_TIME_OF_DAY>
<END_DATE_TIME>12/31/9999 12:00:00 AM</END_DATE_TIME>
<END_TIME_OF_DAY>23:59:59</END_TIME_OF_DAY>
<FREQUENCY_TYPE>Weekly</FREQUENCY_TYPE>
<FREQUENCY_INTERVAL>1</FREQUENCY_INTERVAL>
<FREQUENCY_SUBDAY_TYPE>Once</FREQUENCY_SUBDAY_TYPE>
<FREQUENCY_SUBDAY_INTERVAL>0</FREQUENCY_SUBDAY_INTERVAL>
<FREQUENCY_RELATIVE_INTERVAL>First</FREQUENCY_RELATIVE_INTERVAL>
<FREQUENCY_RECURRENCE_FACTOR>1</FREQUENCY_RECURRENCE_FACTOR>
</SQLAGENTSCHEDULES>
</JOB>
</SCHEDULE_JOBS>
</COMMON_SETTINGS>
</SMSQLCONFIG>
```

Clone job settings The following schema depicts the clone job settings section. You can edit the schedule job settings using an XML editor.

```
<?xml version="1.0"?>
<SMSQLCONFIG xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <HOST_NAME>W2K8R2SP1X64</HOST_NAME>
  <COMMON_SETTINGS>
    <SCHEDULE_JOBS>
      <JOB>
        <SCHEDULER>SQL Server Agent</SCHEDULER>
        <JOB_NAME>CloneAutoDel__abc__09-17-2012_21-38-44</JOB_NAME>
        <HOST_NAME>W2K8R2SP1X64</HOST_NAME>
        <START_DIR>"C:\Program Files\NetApp\SnapManager for SQL Server\
        </START_DIR>
        <APPLICATION_NAME>
          "C:\Program Files\NetApp\SnapManager for SQL Server
```

```
          \SmsqlJobLauncher.exe
        </APPLICATION_NAME>
        <COMMAND>delete-clone –svr 'W2K8R2SP1X64' -inst 'W2K8R2SP1X64'
          -d 'abc__Clone' -JobInstance 'W2K8R2SP1X64'
          -ResyncCloneJob
          'CloneResync__abc__09-17-2012_21-38-44'</COMMAND>
        <START_TIME>9/18/2012 9:38:37 PM</START_TIME>
        <SQLAGENTSCHEDULES>
          <START_DATE_TIME>20120918</START_DATE_TIME>
          <START_TIME_OF_DAY>213837</START_TIME_OF_DAY>
          <END_DATE_TIME>99991231</END_DATE_TIME>
          <END_TIME_OF_DAY>235959</END_TIME_OF_DAY>
          <FREQUENCY_TYPE>OneTime</FREQUENCY_TYPE>
          <FREQUENCY_INTERVAL>0</FREQUENCY_INTERVAL>
          <FREQUENCY_SUBDAY_TYPE>Unknown</FREQUENCY_SUBDAY_TYPE>
          <FREQUENCY_SUBDAY_INTERVAL>0</FREQUENCY_SUBDAY_INTERVAL>
          <FREQUENCY_RELATIVE_INTERVAL>First</
FREQUENCY_RELATIVE_INTERVAL>
          <FREQUENCY_RECURRENCE_FACTOR>0</FREQUENCY_RECURRENCE_FACTOR>
        </SQLAGENTSCHEDULES>
      </JOB>
      <JOB>
        <SCHEDULER>SQL Server Agent</SCHEDULER>
        <JOB_NAME>CloneResync__abc__09-17-2012_21-38-44</JOB_NAME>
        <HOST_NAME>W2K8R2SP1X64</HOST_NAME>
        <START_DIR>
          "C:\Program Files\NetApp\SnapManager for SQL Server\
        </START_DIR>
        <APPLICATION_NAME>"C:\Program Files\NetApp\SnapManager for
          SQL Server\SmsqlJobLauncher.exe</APPLICATION_NAME>
        <COMMAND>clone-database –svr
          'W2K8R2SP1X64' –inst 'W2K8R2SP1X64' -d 'abc'
          -tgInst 'W2K8R2SP1X64' -tgDb 'abc__Clone'
          -tgmpdir 'C:\Program Files\NetApp
          \SnapManager for SQL
            Server\SnapMgrMountPoint' -Resynchronize
          -ForceTerminateConnection -ver
          –verInst 'W2K8R2SP1X64' -mp
          –mpdir 'C:\Program Files\NetApp
          \SnapManager for SQL
          Server\SnapMgrMountPoint' -RetainShareBackups 7
          –mgmt standard </COMMAND>
        <START_TIME>9/18/2012 9:38:37 PM</START_TIME>
        <SQLAGENTSCHEDULES>
          <START_DATE_TIME>20120917</START_DATE_TIME>
          <START_TIME_OF_DAY>213837</START_TIME_OF_DAY>
          <END_DATE_TIME>99991231</END_DATE_TIME>
          <END_TIME_OF_DAY>235959</END_TIME_OF_DAY>
          <FREQUENCY_TYPE>Daily</FREQUENCY_TYPE>
          <FREQUENCY_INTERVAL>1</FREQUENCY_INTERVAL>
          <FREQUENCY_SUBDAY_TYPE>Hour</FREQUENCY_SUBDAY_TYPE>
          <FREQUENCY_SUBDAY_INTERVAL>12</FREQUENCY_SUBDAY_INTERVAL>
          <FREQUENCY_RELATIVE_INTERVAL>First</
FREQUENCY_RELATIVE_INTERVAL>
          <FREQUENCY_RECURRENCE_FACTOR>0</FREQUENCY_RECURRENCE_FACTOR>
        </SQLAGENTSCHEDULES>
      </JOB>
```

```
    </SCHEDULE_JOBS>
  </COMMON_SETTINGS>
</SMSQLCONFIG>
```

Migrating SQL Server databases to LUNs, SMB shares, or VMDKs

Migrating using the Configuration wizard

To migrate your SQL Server databases from local disks to LUNs, SMB shares, or VMDKs or from one LUN, SMB share, or VMDK to another, complete the steps in the following table.

Attention: After an SQL Server database has been migrated using SnapManager, do not use a utility outside of SnapManager to move any other data in the SQL Server system. Doing so might prevent SnapManager from functioning correctly.

Before you begin

- The name of databases must not include a bracket ([or]).
- If you want to migrate databases using any LUN or VMDK configuration, you must have enabled the `unrestricted database layout` option. For details about the `unrestricted database layout` option, see *Enabling SnapManager to allow databases on any LUN or VMDK configuration* on page 294.

Step	Action
1	If you have not already done so, start the SnapManager application. To do this, go to the Windows Start menu and select `Program Files > NetApp > SnapManager for SQL Server`. **Result:** The SnapManager for SQL Server console root opens.
2	Add a server from the Actions pane by clicking **Add Servers to be Managed**, selecting a server from the list, and clicking **Add.** You can also use the Browse option to select a server. **Note:** You can also add server instances using this option, because there can be many server instances within a domain or a single physical server. For more information, see *Starting SnapManager for the first time after installation* on page 60.
3	In the left pane, double-click the name of the SnapManager host that you want to connect to.

Step	Action
4	The Configuration wizard is launched and the Welcome screen appears. Click **Next.** **Result:** If you cleared the **Use control-file** option, the Database Verification Server and Account Authentication screen appears.
5	Use the Database Verification Server panel to specify the SQL Server to be used to perform SQL Server database verification and account authentication; then click **Next**. **Note:** For optimal performance, use an SQL Server that is different from the source (production) server. If you want to specify the database verification server now, select an SQL Server in the Verification Server list and choose the security authentication method to be used to connect to that server. If you choose SQL Server authentication, you must also specify the login name and password. If you want to specify the database verification server later, select the option **Select a verification server later using the Options menu**. Later, when you are ready to select a database verification server, you can do so by selecting **Options > Database Verification Settings** to open the Verification Settings dialog box. For a detailed description of the **Verification Settings** dialog box, see *Database integrity verification options* on page 294. You can set up a different verification server for each SQL Server host in the SnapManager GUI.
6	In the Database Selection panel, do the following and then click **Next**: • In the Database Selection pane, select the database that you want to migrate. • In the Disk Selection pane, select a location for the database. **Note:** A VMDK that is a system drive appears in the Disk Selection pane. • Click <=>. You can view the changes that you made in the Database Location Results pane.
7	Complete the remaining wizard pages.
8	Click **Finish**.

Moving multiple SnapInfo directories to a single SnapInfo directory

Moving multiple SnapInfo directories to a single SnapInfo directory

If you previously configured multiple SnapInfo directories, you can rerun the Configuration wizard to move them to a single SnapInfo directory.

The Configuration wizard enables you to create multiple SnapInfo directories in the following ways:

- A default SnapInfo directory per SQL Server instance
- A separate SnapInfo directory for multiple databases on one or two LUNs, SMB shares, or VMDKs
- A different (non-default) SnapInfo directory for a database in an instance

If you currently have multiple SnapInfo directories, you can choose to combine them into a single directory.

To change your configuration from multiple SnapInfo directories to a single SnapInfo directory for all SQL Server instances and associated databases, complete the following steps.

Step	Action
1	Start the Configuration wizard, then step through the following screens without specifying any configuration changes: • Verification Settings and Account Authentication • Database Selection
2	In the SnapInfo Settings screen, select the Single SnapInfo Directory option, and then click Next. **Result** The Specify a Single SnapInfo Directory screen appears. Note that, in the Current SnapInfo Directory list, all the current SnapInfo directories are selected by default.
3	In the Available Disks window, select the LUN, SMB share, or VMDK to which you want to move all the SnapInfo directories.
4	Click the move (<=>) button. **Result** The Result SnapInfo Directory box displays the path for the SnapInfo directory. Note that the default directory name is `SMSQL_SnapInfo`.
5	If you want to specify a different location or name, modify the information in the Result SnapInfo Directory box. **Note:** The Configuration wizard will allow you to create the SnapInfo directory only in valid locations.
6	Step through the remaining screens of the Configuration wizard without specifying any further configuration changes: • Database Migration Options • Add Microsoft iSCSI Service Dependency • Configure Automatic Event Notification
7	In the Completing the Configuration Wizard screen, verify the changes you specified, and then click Finish.

Step	Action
8	In the Configurator Status dialog box, click Start Now to close the dialog box, and then begin the Configuration wizard tasks you specified.
9	When a message box appears and notifies you that the configuration changes were completed successfully, click OK to close the message.
10	Click Close to close the Configurator Status dialog box.

Migrating SQL Server databases back to local disks

If, for any reason, you choose not to use SnapManager as your data management tool, you can migrate your databases back to local disks.

To migrate your databases back to local disks, complete the following steps.

Step	Action
1	From the SnapManager Actions pane, select Configuration Wizard Option Settings > Enable databases to be migrated back to local disk. **Note:** By default, the Configuration wizard does not list any local drives unless you explicitly enable the migration to a local disk feature.
2	Click OK.
3	Launch the SnapManager Configuration Wizard.
4	Click Next at the Welcome screen.
5	In the "Select a database, file group or file to move" screen, select the database that you want to move back to local disk from the Database Location Results list. • To select multiple databases or files, select the first entry, and then click and hold the Shift button on your keyboard while you make additional selections. • To select a range of databases or files, select the first entry in the range, and then click and hold the Ctrl button on your keyboard while you select the last entry in the range.
6	Click Reconfigure.
7	In the database list, select the database or databases you just re configured. **Note:** In the "Select a Database..." list, the Disk column for this entry lists `Reconfig` instead of the database location.
8	In the Disk Selection Pane, select a local drive, and then click the <=> button.
9	Click Next.

Step	Action
10	In the Select SnapInfo File screen, click Next. **Note:** Both SnapInfo directories must remain on the LUNs, SMB shares, or VMDKs on which you placed them during the original migration. They cannot be moved to local disk.
11	In the Database Migration Options screen, click Next.
12	In the Completing the Configuration Wizard screen, click Finish.
13	Click Start Now to migrate your databases back to local disk.

Setting up a SnapManager share for centralized backups of transaction logs

A centralized network share makes sure that copies of transaction logs are available to all replicas within the Availability Group and non-Availability Group configurations, providing a centralized backup of transaction logs. If the transaction log backups are created on more than one Availability Group replica, those transaction logs are still accessible and can be used for tasks such as up-to-the-minute restores, database reseeding, and using a clone as a replica.

About this task

If you set up a SnapManager share, SnapManager copies any transaction log backups taken on Availability Group databases to the share, including the log backups taken during secondary Availability Group database migration. These log backups are required to perform an up-to-the-minute restore on another node's Availability Group databases.

You can use the Configuration Wizard to set a network location as a centralized location for copies of transaction logs. At the time logs are backed up, the backups are copied to this share. You can do this either as a step in the initial configuration, or as a separate task.

Steps

1. Start the **Configuration Wizard**.
2. Click **Next** until you reach the **Setup SnapManager Share** window.
3. Check **Enable SnapManager share** and enter or browse to an accessible network share.

Preparing your environment for data protection

SnapManager protects your data by creating backups of your databases (Snapshot copies). You can increase your data protection by using built-in mirroring and archiving technologies (SnapMirror and SnapVault) or by archiving backups to third-party tape devices. You need to prepare your environment if you want to use SnapMirror or SnapVault with SnapManager.

You can use any combination of the following methods to protect your data on secondary storage.

Method	Description
Replicating volumes using SnapMirror	SnapMirror technology mirrors a Snapshot copy of data on a source volume to one or more destination volumes. After source and destination relationships are established, a SnapMirror baseline transfer initializes the mirror to create a replica of the source on the destination.
Archiving backups using SnapVault	SnapVault is a disk-to-disk backup and recovery solution. It leverages the efficiencies of Snapshot copies and protects data at the block level. After the initial full backup is complete, only changed blocks are replicated to the secondary storage system.
Archiving backups to third-party tape devices	You can use NDMP, the storage system's `dump` command, or a Windows backup utility to archive backups to tape.

The following table describes the advantages and disadvantages of each method.

Method	Advantages	Disadvantages
Replicating volumes using SnapMirror	• Restoring from a SnapMirror destination is significantly faster than restoring from tape. • The destination can be updated more frequently than by using tape, resulting in more current data.	• Requires another storage system in the remote location. • Requires WAN connectivity to the remote location, with enough bandwidth. • Mirrors only backup sets that are on the source storage system.

Method	Advantages	Disadvantages
Archiving backups using SnapVault	• Restoring from a SnapVault archive is faster than archiving from tape. • You can create and restore remote backup and archives. • The destination can be updated more frequently than by using tape. • Backup sets that are no longer available on the primary storage can be retained.	• Requires another storage system in the remote location. • Requires WAN connectivity to the remote location, with sufficient bandwidth. • Recovery requires data to be replicated back to the original storage system.
Archiving backups to third-party tape devices	• Tape backups require fewer resources to maintain. • You can place the archives in a more secure place than you can place a storage system. • You can recover data from any release of Data ONTAP.	• Restoring data from tape takes a long time. • Finding a particular file or directory on tape is time-consuming.

Related tasks

Preparing your environment to replicate backups on page 86
Preparing your environment to archive backups (clustered Data ONTAP) on page 89
Preparing your environment to archive backups (7-Mode) on page 90

Preparing your environment to replicate backups

How SnapManager uses SnapMirror

SnapManager backs up your SQL Server data by creating Snapshot copies of the databases and transaction logs. SnapMirror can be used to replicate the volumes that host the Snapshot copies to mirrored volumes on a remote storage system.

How SnapManager uses SnapMirror

SnapMirror replication for SnapManager is volume replication When you use SnapMirror to replicate SnapManager backups, you can replicate only volumes, not qtrees. SnapManager does not support SnapMirror qtree replication.

Backup Snapshot can trigger SnapMirror updates SnapManager Backup uses Snapshot copy functionality to back up your SQL Server data to a storage system volume managed by SnapDrive. If the volume has been configured as a SnapMirror source volume with one or more appropriately configured destination volumes, then (upon successful completion of a Snapshot copy backup operation) SnapManager can send a request to SnapDrive to begin a SnapMirror update of each destination volume.

All three types of SnapManager Backup operations can be configured to trigger SnapMirror updates:

- Full database backup, with or without transaction log backup or database verification
- Transaction log backup only
- Database verification only

Note: The result of the database verification operation (the database integrity status) is written to the backup set, in the SnapInfo directory. By replicating the backup set to the mirrored volume, this status information is kept current on the mirrored volume.

SnapMirror replication for SnapManager is asynchronous

When it is configured directly on the storage system, SnapMirror can be used to perform synchronous replication or asynchronous replication.

When you use SnapMirror to replicate SnapManager backups, however, the replication is performed asynchronously. That is, changes made to the databases between SnapManager backups are not replicated in the SnapMirror destination volume. Therefore, any restore from the destination volume restores the databases to their state at the time of the last SnapManager backup; subsequent changes to the database that occur on the source volume after the most recent SnapMirror replication update are not available if a catastrophic failure occurs before the next update of the SnapMirror destination volume.

SnapMirror scheduling When it is configured directly on the storage system, SnapMirror uses its own replication schedule as configured by a Data ONTAP administrator.

When SnapMirror is used by SnapManager, however, the SnapMirror replication schedule must be disabled on the storage system; SnapMirror updates are instead initiated by SnapDrive on the completion of a SnapManager backup operation.

Requirements for using SnapMirror with SnapManager

To use SnapMirror with SnapManager, you must have previously configured SnapMirror on both the source volume to be replicated and its destination volumes. How to complete these tasks is explained in your SnapDrive documentation. Your configuration must satisfy the following requirements:

- There must be one or more SnapMirror source volumes.
- There must be one or more SnapMirror destination volumes for each source volume.
- The size of the destination volumes must be equal to or greater than the size of the source volume.
- SnapMirror licenses must be enabled on both the source and destination storage systems.

- You must manually configure and initialize the SnapMirror replication between source and destination volumes.
- You must disable the SnapMirror replication schedule on your storage system.
- You must configure the SnapMirror replication as asynchronous.

Process overview

If your SQL Server databases reside on a storage system volume that is configured as a SnapMirror source volume, then the SnapMirror destination volume is optionally updated after the SnapManager backup operation finishes.

The following sequence provides an overview of how SnapMirror destination replication works:

1. A SnapManager backup is initiated.
2. SnapManager completes all Snapshot copies required for the backup, then requests a SnapMirror volume update through SnapDrive.
3. If any volume whose data is captured in the backup is a SnapMirror source volume, SnapDrive requests information about all SnapMirror destination volumes of that source volume.
4. SnapDrive sends a SnapMirror destination update request to all the related destination volumes.
5. SnapMirror updates the destination volumes to reflect incremental changes to the source volume.

Preparing your environment for SnapMirror replication

SnapManager can mirror copies of backup sets on a SnapMirror destination volume when it backs up your databases. Before you can use SnapManager's integrated SnapMirror technology, you must configure a data protection relationship between the source and destination volumes, and then initialize that relationship.

Before you begin

To verify a SnapMirror destination volume running on clustered Data ONTAP, the source and destination Storage Virtual Machines (SVMs) on which the volumes reside must have two LIFs:

- A management LIF that has the role set to `data`, the protocol set to `none`, and the firewall policy set to `mgmt`.

  The LIF must have the same DNS name as the SVM.
- A data LIF that has the role set to `data` and the protocol set to `cifs`, `iscsi`, or `fcp`.

About this task

The following steps provide a high-level overview of how to configure a SnapMirror relationship. For more detailed information, see the *Data Protection Guide* for your version of Data ONTAP or the online Help for OnCommand System Manager.

Steps

1. Create or identify the destination volumes to which you want to replicate data.
2. Create a mirror relationship between the primary volume and the destination volumes.
3. Initialize the relationship to start a baseline transfer.
4. Disable the SnapMirror replication schedule, if it is enabled.

   SnapManager defines the mirror schedule when you create the backup schedule.

After you finish

Use SnapManager to back up your databases and select the option to update SnapMirror destination volumes.

Related concepts

Protecting databases by backing up, replicating, and archiving on page 113

Preparing your environment to archive backups (clustered Data ONTAP)

SnapManager can archive backups to a SnapVault secondary volume, which contains a set of read-only backup copies that you can use to restore older copies of your databases. Before you can use SnapManager's integrated SnapVault technology, you must configure a data protection relationship between source and destination volumes, and initialize the relationship.

Before you begin

- The source and destination volumes must be in peered clusters with peered Storage Virtual Machines (SVMs).

  For more information, see the *Cluster and SVM Peering Express Guide*.
- The language setting of the destination SVM must match the language setting of the source SVM.

About this task

Clustered Data ONTAP SnapVault support is provided through SnapDrive. The NetApp Management Console is not used with SnapManager for clustered Data ONTAP SnapVault support, like it is for Data ONTAP operating in 7-Mode.

The following steps provide a high-level overview of how to configure a SnapVault relationship. For more detailed information, see the *SnapVault Express Guide* or the online Help for OnCommand System Manager.

You should not define a cron schedule or Snapshot copy policy through Data ONTAP or System Manager. SnapManager does that for you when you create a backup schedule and select the option to archive backups to secondary storage.

Steps

1. Identify the secondary storage to which you want to archive the backups.
2. Create a destination volume with the volume type `DP`.
3. On the destination SVM, create a SnapVault policy.
4. Add rules to the policy that are labeled as follows:
   - Daily
   - Weekly
   - Monthly
   - Hourly
   - Unlimited

   These are fixed labels that SnapManager uses. You select one of these labels when you archive a backup.
5. Create an `XDP` relationship between the source and destination volumes and apply the policy to that relationship.

   The `XDP` relationship type defines the relationship as a SnapVault relationship.
6. Initialize the relationship to start a baseline transfer.

After you finish

Use SnapManager to back up your databases and select the option to archive backups to secondary storage.

Related concepts

Protecting databases by backing up, replicating, and archiving on page 113

Preparing your environment to archive backups (7-Mode)

Understanding dataset and SnapVault integration

Why dataset and SnapVault integration is required

A dataset is a collection of storage sets with identical data protection requirements on the primary storage system. It is a data management concept introduced in NetApp Management Console and gives you extensive remote backup and archival capabilities.

Datasets are composed of the following three elements:

- Database
- Protection policies

- Resource pool

The protection policies determine how the data is protected. The resource pool includes the backups and replica of the primary data and its configuration information.

By replicating Snapshot copies to secondary storage, SnapVault provides you with a centralized disk-based backup solution. It enables you to keep weeks of backup information online for faster restore operations. Through datasets, SnapManager integrates with SnapVault to archive backups to secondary storage.

SnapManager uses Data ONTAP Snapshot technology to create and restore local backups. Dataset and SnapVault integration with SnapManager provides an integrated, rapid solution to create and restore remote backup copies and archives.

SnapManager manages backup on the primary location, but archived backup is managed by NetApp Management Console.

The following capabilities of NetApp Management Console make it a good option for integration with SnapManager:

- Automatic setup of SnapVault relationships and complex replication topologies with resource pools
- Scheduling of remote backups
- Monitoring of data transfer
- Management of remote backup retention

If NetApp Management Console is available, and SnapDrive is configured for the DataFabric Manager server, SnapManager automatically becomes aware of the dataset. If NetApp Management Console is not available, SnapDrive informs SnapManager of its unavailability. SnapManager continues in normal working mode, and remote backup is not supported.

Note that the DataFabric Manager server and SnapManager backups are not coordinated; changing the backup policy in the DataFabric Manager server does not change the policy in SnapManager, and changing the policy in SnapManager does not change the DataFabric Manager server policy.

Available functionality

You can do the following with SnapManager integrated with dataset and SnapVault:

- Create and restore remote backup.
- Select policies related to the dataset created by the NetApp Management Console.
- Protect created datasets by doing the following:
  - Creating a remote backup on the SnapVault secondary storage system.
  - Mirroring the local source volume to SnapVault destination volume.
  - Using topologies supported by the NetApp Management Console.
- Delete individual remote backups based on the backup version.
- Display remote backups that are available for restore.

- Perform temporary restore operations to another location on the secondary storage system using SnapVault remote Snapshot technology.
- Perform remote backup integrity verification.

Limitations

Limitations in integrating SnapManager with dataset and SnapVault are as follows:

- No remote backup and archival facility is present if dataset configuration is not available.
- The administrator cannot control the archived backup retention policy through SnapManager. It is controlled by the NetApp Management Console.
- The dataset cannot be used for disaster recovery or business continuance.
- Multiple LUNs residing on the same storage system qtree, and LUNs not residing on a storage qtree, are not supported.
- You must roll forward archived backed up transaction logs manually.
- System databases are not supported by dataset and SnapVault integration with SnapManager.

Software dependencies

Software dependencies for integrating SnapManager with dataset and SnapVault are as follows:

- OnCommand Unified Manager Core Package 5.2 or later, which includes the NetApp Management Console
- SnapVault (for both primary and secondary locations)
- NDMP

You can upgrade SnapManager from an earlier version that did not support datasets to a later version that supports datasets. You can also revert to the older version without any adverse effects on the system.

Prerequisites

Prerequisites for dataset and SnapVault integration with SnapManager are as follows:

- Two storage systems should be present.
  One should have the SnapVault primary license, and the other should have the SnapVault secondary license. The primary is the archival source; the secondary is the archival destination.
- All LUNs must be created on qtrees, and each qtree should contain only a single LUN.
- You should install the OnCommand Unified Manager Core Package on a dedicated server other than the SQL Server.
- SnapDrive for Windows should be installed.

Integrating dataset and SnapVault to SnapManager

Integrating dataset and SnapVault to SnapManager

Follow this outline of steps to integrate dataset and SnapVault to SnapManager.

1. Install the OnCommand Unified Manager Core Package on your system.
2. Install SnapDrive for Windows and provide the necessary information to enable the data protection capabilities of the OnCommand Unified Manager Core Package.
   See the *SnapDrive for Windows Installation Guide* for more information.
3. Give the SnapManager service account rights on the DataFabric Manager server.
   For more information, see *SnapManager service account requirements* on page 23.
4. Run the Configuration wizard.
5. Select the archived backup sets and protection policies.
6. Assign a resource pool to the dataset using the NetApp Management Console.
7. To test your configuration, run a SnapManager backup operation and a restore operation.

Configuring datasets

About dataset configuration

A storage set grouped with its configuration information makes a dataset. Datasets associate the LUNs used by an SQL Server to the related set of protection policies. This enables the administrator to protect the data through remote backup and relate to the corresponding resource pool. One dataset is created for each SQL Server on the server host.

Datasets are created when the SnapManager Configuration wizard is run for the first time on a system with the NetApp Management Console installed. If SnapManager is upgraded from an earlier version, rerun the Configuration wizard to setup a dataset. Backups scheduled before the configuration of dataset continue to function without interruption.

The names of the datasets cannot be changed. The following is the example for the naming convention for a dataset:

```
SnapMgr_SQL_server1
```

For SQL Server running on Microsoft Clustered Server, a virtual server is used to name the SnapManager dataset.

About protection policies

The dataset policies control the protection of data in dataset. A policy decides the following characteristics:

- Data replication topology
  - SnapVault topology (also called Backup topology)
- Backup retention type
  - Primary (Determined using SnapManager, the NetApp Management Console remains unaffected)
  - Secondary (Determined using the NetApp Management Console)

- Replication lag and throttle

After a dataset policy is set up, it cannot be changed to another policy from SnapManager. If it is changed using the NetApp Management Console, it is automatically picked up by SnapManager.

Each dataset has a policy assigned to it. But a single policy may be applied to many datasets. Hence modifying a policy might affect all the associated datasets.

You can create a new policy by modifying an existing policy using the NetApp Management Console. For more information, see your OnCommand Unified Manager Core Package documentation.

Note: "Remote backups only" policy is the policy that SnapManager currently supports.

Remote backup retention policies

Remote backup retention policies control the backups created at the remote site. The remote backup retention policies are controlled by SnapDrive and the NetApp Management Console, not SnapManager.

Creating a dataset using SnapManager

You can create a dataset to manage protection for data that shares the same protection requirements. For one SQL server, there can be only one dataset. Create this dataset when you run SnapManager Configuration wizard with the NetApp Management Console for the first time.

Before you begin, ensure that you are assigned an administrator role that enables you to create a dataset. Also ensure that the primary databases are configured properly before the archival process is carried out, or it will fail.

Editing a dataset using the NetApp Management Console

After the dataset is created using SnapManager, check the Protection status and the Conformance status of the dataset using the NetApp Management Console. Next, you need to add the resources at the secondary storage system manually using the NetApp Management Console.

SnapVault relationships

After the dataset is created, policies are determined, and secondary resource pools are added to the dataset, The NetApp Management Console creates SnapVault relationships for archiving. A remote backup restore is not possible if the SnapVault relationship is changed or modified.

If you already have an existing SnapVault relationship, the NetApp Management Console cannot use the existing SnapVault relationship for the dataset automatically. Import the existing SnapVault relationship using the NetApp Management Console. For more information, see your OnCommand Unified Manager Core Package documentation.

If you do not import the SnapVault relationship, a new one is created. For more information, see your OnCommand Unified Manager Core Package documentation.

If you have a SnapVault relationship for the LUN that is used by database, deleting the SnapVault base line Snapshot copy will result in a SnapDrive for Windows error.

Dataset member information

The dataset member information is a list of drive letters and mount points related to SnapManager. It is stored and tracked by the NetApp Management Console, and its information is retained even after SnapDrive is uninstalled. The member information is retained on all cluster nodes.

Protecting local backups

By creating remote backups, SnapManager uses datasets to protect the local backups that were created at the primary storage system. The following conditions should be met before SnapManager starts creating backups:

- A dataset is created.
- The "Archive local backup using SnapVault" option is enabled.
- The dataset has the protection status as "Protected" and conformance status as "Conformant."
- If the configuration contains non-SQL LUNs, the qtrees containing the non-SQL database are not updated during archiving. This changes the dataset protection status to "Lag Warning" or "Lag Error". For more information, see your OnCommand Unified Manager Core Package documentation.

Creation of remote backup

The process of remote backup starts after local backups are created. SnapManager conveys the following information to SnapDrive before actuating the remote backup process:

- The version number of the backup
- The version number acts as the time stamp for the backup and is used by SnapManager to retrieve detailed information about the backup during restore.
- The backup management group
- Two types of management groups are available:
  - Local management group

    The local management groups can be standard, daily and weekly
  - Remote management group

    The remote management groups can be hourly, daily, and weekly, monthly, all, and unlimited. The default management group is daily.

    If you select the hourly management group for remote backup, SnapManager shows a message conveying that hourly archived backups are deleted when the NetApp Management Console restarts.
- A list of LUNs with their corresponding Snapshot names

You can defer remote backup for some time after the local backup is created.

In the Backup wizard, if dataset is configured and the archival process is initiated, the generic backup naming convention is automatically changed to the unique backup naming convention. If you choose to keep the naming convention as generic, no archives are created.

Remote backup retention

Remote backup retention capability refers to the number of backups that can be retained at the secondary storage system. You can determine the number by using the backup management groups. Remote backup retention is controlled by the NetApp Management Console. When SnapManager deletes a backup, it deletes the metadata only after confirming with the NetApp Management Console that the archive backup has also been deleted. The SnapInfo directory that retains the backup metadata in the live file system is not deleted, even if the local backup has been deleted.

When the NetApp Management Console applies the remote backup retention policy to the dataset, it deletes the older version of backup. New backups are continuously created. If the number of backups or days exceed the management group setup, the policy deletes the last backup at the secondary storage system.

Understanding SnapManager backup sets

How SnapManager backup works

About SnapManager backup

SnapManager backup uses Snapshot copy functionality to create online, read-only copies of databases. After the selected databases are backed up, the transaction logs that are already committed to the databases captured in the backup are deleted.

Note: Databases that cannot be backed up by SnapManager are greyed out in the Results pane.

Types of backups SnapManager can perform

SnapManager backup performs the following backup operations at the volume level:

Volume-wide backup: When a Snapshot copy is made of a LUN, SMB share, or VMDK for a SnapManager backup, the entire volume is captured in that Snapshot copy. However, that backup is valid only for that server. If data from other servers resides on the same volume, it is not restorable from that Snapshot copy.

Multiple-volume backups: SnapManager performs backups in parallel on all LUNs, SMB shares, or VMDKs that belong to the same server and share a single storage system volume. When a database set spans multiple volumes, the resulting backup set contains multiple database Snapshot copies but is still restorable as a single entity.

Partial backups: If the backup of some of the databases in the database set fails, the databases in the set that were backed up successfully can still be restored. Because each database constitutes its own backup copy, or file, it is restored discretely, independent of backup copies of the others in its backup set, even though the backup of all databases in the set was performed by the one backup job.

What SnapManager backup does

SnapManager performs the following tasks when creating a backup copy:

1. Checks the SnapManager license
2. Renames the most recent SnapInfo directory (if necessary)
3. Renames the most recent Snapshot copy (if necessary)
4. Creates a new directory in the SnapInfo directory for this backup

   Note: During the backup process, SnapManager collects backup metadata that is automatically archived to the SnapInfo directory.

5. Creates a backup set of the storage containing the database files
6. Backs up transaction logs (if specified)
7. Creates a Snapshot copy of the storage that contains the SnapInfo directory
8. Verifies the databases in the backup set (if specified)
9. Deletes the oldest backup sets (if specified)
10. Deletes the oldest Snapshot copy that contains the SnapInfo directory

SnapManager backup requirements and limitations

You should be aware of the following requirements and limitations of SnapManager backup:

- The SQL database property AutoClose must be set to FALSE.
- SnapManager cannot back up an SQL Server database copies.
  SQL Server database copy names are greyed out in the Backup wizard and Backup and Verify dialog box.
- SnapManager does not support database names with any of the characters \ / : * ? " < > | [,] (although SQL Server does support them).
- If a database name ends in a space, you must change the following DWORD (32-bit) registry setting to 1 and restart the SnapManager Service:
  `HKEY_LOCAL_MACHINE\SOFTWARE\Network Appliance\SnapManager for SQL Server \Server\HaveDBWithTrailingSpaces`
- To run SnapManager backup, the account that SnapManager is using must have a system administrator server role on the SQL Server.
- If you rename the SQL database and then need to restore the database from a backup set that was created before the database was renamed, you must restore to a different and nonexistent database name.
- If you use vFiler units, be sure to enable the option `vfiler.vol_clone_zapi_allow` on the storage system, which is disabled by default.
  If you do not enable this option, SnapDrive does not create clones for the database. For more information, see SnapDrive documentation.
- You must run the Configuration wizard on each SQL Server instance that you want to back up.
  Running the Configuration wizard sets up the backup metadata location for the instance.
- If you change the database configuration, any backups made before the configuration change are invalid.
  After you run the Configuration wizard, immediately make a backup copy to reflect the change in your configuration.
- You can run only one full database backup at a time.
  However, you can schedule more than one deferred verification to run at a time. You can also start a full database backup when a deferred verification is already running.

Attention: In a Microsoft SQL Server environment, you should perform backups using only the SnapManager application. Making Snapshot copies of the storage system for the storage console directly is not supported and results in an inconsistent Snapshot copy image of online databases.

However, you can use SnapDrive to make Snapshot copies of SQL Server databases, although you cannot restore these Snapshot copies using SnapManager.

How SnapManager backup data is organized

SnapManager backup data is stored in backup sets. A SnapManager backup set consists of all the data you need to be able to perform a restore, regardless of whether this data exists on the same LUNs, SMB shares, VMDKs, or volumes. SnapManager also enables you to create backups for read-only databases.

SnapManager backup sets

A backup set contains the following items:

- Database data files
- Transaction logs
- SnapInfo directory

SnapInfo directory

The SnapInfo directory stores information about the streaming-based backups of system databases, copies of transaction log files, and the backup set's metadata. You can specify the location of this directory when you run the Configuration wizard. By default, the directory name is `SMSQL_SnapInfo`. However, you can specify a different directory name.

If the database has a very long name, you should use a shorter SnapInfo directory name; otherwise, the backup might fail due to path length limitations when running Microsoft backup APIs. Using a mount point for the SnapInfo directory typically has a longer SnapInfo path.

Every time a SnapManager backup set is created, SnapManager creates a new backup set subdirectory under the SnapInfo directory. SnapManager populates this subdirectory with the transaction logs backed up as part of that backup set, in addition to the recovery information for that specific Snapshot copy. A complete backup set consists of this SnapInfo subdirectory and the corresponding Snapshot copies of the LUNs, SMB shares, or VMDKs that store the SQL Server databases.

Note: The SnapInfo directory cannot reside on the same LUN or VMDK that stores the database files. This restriction does not apply to SMB shares.

SnapInfo subdirectory names

SnapManager backup set names identify the configuration of the backed-up databases.

Configuration	Format of the SnapInfo subdirectory name
Databases belonging to the SQL Server default instance	The SnapInfo directory name is `SQL__` followed by the SQL Server computer host name: `SQL__`*`SqlServerName`* For example, the subdirectory for databases belonging to the default instance of the SQL Server on the Windows host system `CLPUBS-WINSRVR3` would be named as follows: `SQL__ CLPUBS-WINSRVR3`
Databases belonging to an SQL Server named instance	The SnapInfo directory name is `SQL__` followed by the name of the SQL Server instance: `SQL__`*`InstanceName`* For example, the subdirectory for databases that belong to the SQL Server instance `INST2` on the Windows host system `ENGR-WINSRVR7` would be named as follows: `SQL__INST2`

SnapManager backup set names

SnapManager backup set names identify the configuration of the backed-up databases. These names are displayed in the SnapManager Results pane and in the SnapManager Restore wizard.

Configuration	Format of the backup set name
Databases belonging to the SQL Server default instance	The backup set name is the same as the SQL Server computer host name: *`SqlServerName`* For example, a backup set for databases that belong to the default instance of the SQL Server on the Windows host system `CLPUBS-WINSRVR3` would be named as follows: `CLPUBS-WINSRVR3`
Databases belonging to an SQL Server named instance	The backup set name is the name of the SQL Server instance: *`InstanceName`* For example, a backup set for databases that belong to the SQL Server instance `INST2` on the Windows host system `ENGR-WINSRVR7` would be named as follows: `INST2`

SnapManager backup set naming conventions

The Snapshot copies created by SnapManager backup operations are automatically named by SnapManager. The name of each backup set created during a SnapManager backup operation includes information that identifies the Snapshot copy contents.

- SQL Server name
  Database backup set names and SnapInfo directory Snapshot copy names include the name of the SQL Server for which the backup was made (indicated in this document by the variable *SqlServerName*).
- Backup management group
  Database backup set names and SnapInfo directory Snapshot copy names include the backup management group to which you assigned the full database backup. SnapManager provides backup management groups for designating various levels of backup retention: `Standard`, `Daily`, and `Weekly`.
  - If you assign a full database backup to the `Standard` backup management group, the Snapshot copy names for the databases and SnapInfo directory do not include a backup management group name.
  - If you assign a full database backup to the `Daily` or `Weekly` management groups, the Snapshot copy names for the databases and SnapInfo directory include the name of the backup management group (indicated in this document by the variable *BackupMgmtGrp*).

  For more information about using backup management groups, see *Using backup management groups in backup and verification* on page 152.
- Most recent backup
  Earlier versions of SnapManager appended the string `recent` to the name of the most recently created Snapshot copy. This was to allow external scripts, for example, archive scripts, to identify and operate on the most recent backup set. With the addition of the Run Command Settings feature in SnapManager, appending `recent` is no longer necessary because the scripts can be integrated into the backup process.
  If you enable the enhanced Snapshot copy naming convention in **Backup Settings > Full Database Backup > Advanced**, the most recent backup copy per management group (daily, weekly, standard) will have the suffix "__recent". For example, once enhanced Snapshot copy naming is enabled, backup copies will have the following names:
  - `sqlsnap__`*SqlServerName*`__Daily__recent`
  - `sqlsnap__`*SqlServerName*`__Weekly__recent`
  - `sqlsnap__`*SqlServerName*`__recent`

  Note: You cannot disable the Snapshot copy naming convention after you enable it.

SnapManager offers two conventions for naming backup Snapshot copies:

- Unique backup naming
  The most recent Snapshot copy name contains the Snapshot copy creation date and time (indicated by the *date_time* variable) instead of the string `recent`. The most recent backup is identified by the Snapshot copy name with the most recent date and time. This removes the need

to rename the Snapshot copy when the next backup is created. This is the default naming convention for SnapManager.

- Generic backup naming
  The most recent Snapshot copy name contains the string `recent` instead of a date and time stamp. The most recent backup is identified by the Snapshot copy name that includes the string `recent`. This is the Snapshot copy naming convention used by older versions of SnapManager and is the default setting.

Note: When the database resides on a VMDK, you must use the timestamp-based Snapshot copy naming convention when you want to clone Snapshot copies.

When you have a dataset configured in your system, you can either choose to apply the unique backup naming convention with the archival process enabled, or to keep the generic backup naming convention. If you choose to keep the naming convention as generic, no archives of the database to be backed up at the remote location are created.

If you archive the backups using PowerShell, the generic backup naming convention is automatically changed to the unique backup naming convention.

The backup naming convention is selected in the Backup Settings dialog box. For information about using this dialog box, see *SnapManager backup options* on page 298.

Note: You should select the unique naming convention option unless you have legacy scripts that require the presence of a backup with `recent` in its name. You need to select the unique naming convention explicitly (using the **Options** > **Backup Setting** menu or the Backup Naming Convention screen of the Backup wizard) because for backward compatibility, the generic naming convention is selected by default.

SQL Server database backup set names

For SQL Server, backup set names begin with the string `sqlsnap__`.

Backup management group	Format of the SQL Server database backup set name
Standard	Depending on the naming convention selected: • `sqlsnap__`*`SqlServerName_date_time`* • `sqlsnap__`*`SqlServerName`*`__recent`
Daily or Weekly	Depending on the naming convention selected: • `sqlsnap__`*`SqlServerName_date_time__BackupMgmtGrp`* • `sqlsnap__`*`SqlServerName`*`__recent` • `sqlsnap__`*`SqlServerName__BackupMgmtGrp`*`__recent`

SnapInfo directory Snapshot names

For SnapInfo directory backups, Snapshot copy names begin with the string `sqlinfo__`.

Backup management group	Format of the SnapInfo directory Snapshot copy name
Standard	Depending on the naming convention selected: • `sqlinfo__SqlServerName_date_time` • `sqlinfo__SqlServerName__recent`
Daily or Weekly	Depending on the naming convention selected: • `sqlinfo__SqlServerName_date_time__BackupMgmtGrp` • `sqlinfo__SqlServerName__recent`

Types of backup operations performed using SnapManager

Types of SnapManager backup

SnapManager supports two types of backup operations:

- Full database backup
- Transaction log backup

A transaction log backup can be included in a full database backup, or it can be created as a log-only backup set.

Copy-only backup (database and transaction log) can be selected while configuring backup operation.

Full database backup

A full database backup contains a full copy of the databases that you select. The method that SnapManager uses to create the backup depends on the databases that you select. One method involves streaming the content of the databases individually, while the other method consists of creating Snapshot copies of the databases. The method that SnapManager uses to create a particular backup set has implications for how SnapManager restores databases from that backup set. For more information, see *Understanding SnapManager Restore* on page 160.

Stream-based backup method: With this method, SnapManager creates the full database backup by streaming the contents of the databases individually. SnapManager uses the stream-based method to back up the following:

- All *system databases*
- Any *user databases* that reside on the same LUN or VMDK as a system database

All other database backups use the *online Snapshot copy* backup method.

Note: If there is a system database on the LUN or VMDK that hosts the SQL Server, a user database should not reside on that LUN or VMDK. This restriction is enforced by the Configuration wizard.

Full database stream-based backup files are `.fbk` files named using the convention *date_time_databasename*: for example, `050802_0330_xxx.fbk`. This file is equivalent to the `.bak` file directly created by SQL Server.

Online Snapshot copies backup method: With this method, SnapManager creates the backup by creating Snapshot copies of the databases. SnapManager uses the online Snapshot method to back up all user databases that reside on SMB shares and the user databases that do not reside on the same LUN or VMDK as system databases. All other database backups use the stream-based backup method.

When you select a database for a full database backup, SnapManager automatically selects all other databases that reside on the same storage system volume. You can clear databases that reside on a different LUN, SMB share, or VMDK from the databases you selected, even if the LUN, SMB share, or VMDK is on the same storage volume. If the other LUN, SMB share, or VMDK stores only a single database, you can clear or reselect that database individually. If the other LUN, SMB share, or VMDK houses multiple databases, you must clear or reselect those databases as a group.

For a description of the naming convention used by full database online Snapshot backup sets, see "SnapManager backup set names" in *How SnapManager backup data is organized* on page 99.

More about volume-wide backups: In a volume-wide backup, all the databases that reside on a single volume are backed up *concurrently* using Snapshot copies. If the maximum number of concurrent backup databases is 35, then the total number of Snapshot copies created equals the number of databases divided by 35.

Note: When a Snapshot copy is made for a SnapManager backup, the entire storage system volume is captured in that Snapshot copy. However, that backup is valid only for the SQL host server for which the backup was created. If data from other SQL host servers resides on the same volume, that data is not restorable from the Snapshot copy.

About SQL Server Management Studio: Although SnapManager *Snapshot copy* full database backup files are viewable from the SQL Server Management Studio of your SQL Server, you cannot perform any operations on them using the SQL Server Management Studio.

Transaction log backup

A transaction log backup is a record of the committed database changes that have occurred since the last transaction log backup that was truncated after the backup completed. SnapManager supports transaction log backups to provide a more granular level of database backup and to recover the transactions committed since the most recent full backup.

File name and location: SnapManager creates a backup of a transaction log by copying transaction log data to a file in the SnapInfo directory. Transaction log backup files are named using the following convention:

- *date_time_databasename*.trb

This file is equivalent to the .trn file directly created by SQL Server Management Studio. The structure of the SnapInfo directory is described in *Ways to manage the number of backup sets kept online* on page 109.

Ways to start or schedule a transaction log backup: You can back up a transaction log along with the database or alone.

- SnapManager full database backups include the option to also back up the associated transaction logs after the database Snapshot copy backups finish.
  This is described in *Managing transaction log backups using SnapManager* on page 126.
- SnapManager also provides the option to back up transaction logs only, independent of the associated databases.
  This is described in *Managing transaction log backups using SnapManager* on page 126.

About log shipping and other backup solutions: It is best to use SnapManager alone to back up your SQL Server database transaction log files. SnapManager supports log shipping; therefore, if you decide to use a different backup solution, use it alone as well; do not attempt to restore from backup files that were created using different backup solutions. If you use log shipping, you cannot back up the transaction logs for that database.

If *log shipping* is implemented for a particular database, remember the following recommendations:

- When using SnapManager Backup, do not back up the transaction logs for that database.
- When using SnapManager Restore to restore that database, disable the option to create a transaction log backup before the restore operation and do not restore the transaction logs.

About SQL Server Management Studio: SQL Server Management Studio detects transaction log backups made by SnapManager for Microsoft SQL Server, and can be used to restore the database to a further point in time by applying transaction log backups in sequence. However, SQL Server Management Studio cannot restore full database backups of Snapshot copies made by SnapManager for Microsoft SQL Server.

How SnapManager checks database integrity in backup sets

SnapManager uses Database Consistency Checker (DBCC) to verify SQL Server databases. DBCC is a Microsoft SQL Server utility that verifies the page-level integrity of databases.

Ways that SnapManager uses SQL Server DBCC

SnapManager uses the DBCC CHECKDB command to verify the integrity of live databases in addition to databases in SnapManager backup sets.

Verifying the integrity of live databases: Live databases can be verified as a part of database migration and also as a part of a full database backup:

- Using the Configuration wizard, you can verify live databases before and after database migration.
- Using SnapManager Backup, you can verify live databases before and after a full database backup.

  For more information, see "Configuring the profile of a full database backup" in *SnapManager backup options* on page 298.

To perform live verification of databases residing on SMB shares, you must configure the verification settings in the SnapManager console to use the TABLOCK option.

Verifying the integrity of databases in backup sets: Databases in backup sets can be verified on creation, separately, or before a restore operation:

- Using SnapManager Backup, you can verify the databases in full database backup sets as they are created or you can verify the databases in the most recent unverified backup sets.
- Using SnapManager Restore, if you select a backup set on which a consistency check has not been run successfully, SnapManager prompts (but does not require) you to first verify the databases in that backup set.

Attention: The SnapManager Restore Results pane lists the backups that have been made and the backup verification status of each.

Requirements for running SQL Server DBCC against the databases in a backup set

When you verify the databases in a backup set (as opposed to live databases), Microsoft DBCC requires that all the database files be mounted at the same time. At a more granular level, this means that SnapManager, using SnapDrive commands, mounts all the LUNs or VMDKs that contain the backup sets selected for database verification.

Each LUN or VMDK that is mounted requires one available drive letter or a mount point: To run the DBCC CHECKDB command, the verification server (whether local or remote) must have a sufficient number of drive letters available or a mount point to mount all the LUNs or VMDKs that store the database backup sets you are verifying.

- When you run database verification against backup sets that are stored on a single LUN or VMDK, the SQL Server computer that is used as the verification server must have at least one drive letter available or a mount point so that the LUN or VMDK can be mounted during database verification.
- When you run database verification against backup sets that contain multiple database files stored on separate LUNs or VMDKs, SnapManager mounts all those LUNs or VMDKs at the same time.

  Consequently, the SQL Server that is used as the verification server must have enough drive letters available so that SnapManager can mount each of the LUNs or VMDKs simultaneously.

  For example, suppose you want to run database integrity verification against backup sets containing five file groups using three transaction logs stored on eight separate LUNs or VMDKs.

In this case, the verification server would need to have a minimum of eight drive letters or a mount point available.

Lack of available drive letters causes DBCC CHECKDB to fail: If the verification server runs out of available drive letters while attempting to run `DBCC CHECKDB` for a SnapManager operation, the SnapManager operation fails with the following message in the report log:

```
[SnapDrive Error]: There are no remaining drive letters available on the
system.  Please delete or disconnect a drive and retry.
```

Note: Available drive letters are not required if the databases reside on SMB shares.

The SnapManager operations that enable you to verify the databases in backup sets are as follows:

- Full database backup with verification of the databases in the backup set.
  For detailed information, see *Backing up, replicating, and archiving databases using SnapManager* on page 115.
- Verification of the databases in the most recent unverified backup sets.
  For detailed information, see *Performing database verification using SnapManager* on page 138.
- Verification of the databases in an unverified backup set selected for a restore operation.
  For detailed information, see *Performing a restore operation* on page 167.

Ways to separate database verification from database backup

Running database verification on a production SQL Server is CPU-intensive for the host running the verification and also involves a substantial amount of activity on the storage system. For this reason, verification can degrade SQL Server response, particularly during work hours.

By default, a SnapManager full database backup operation runs DBCC immediately after the backup is complete. However, SnapManager provides two options that enable you to separate the process of verification from the backup itself: deferred database verification and remote database verification.

Deferred database verification: Instead of allowing a full database backup to automatically verify the databases when the operation is complete, you can disable that feature. You can then run a separate database verification operation any time after the full database backup operation is complete.

Note: You can schedule more than one deferred verification to run at the same time.

Remote database verification: To prevent database verification from affecting the performance of your production SQL Server computer, you can run verification on another SQL Server computer.

Options for when to verify the databases in a backup set

You can verify the databases in your SnapManager backup sets at various times.

Automatically verify full database backup sets on creation: By default, SnapManager verifies the databases in a backup set at the time the backup is created. This is simple and ensures that each

database in the backup set is verified. However, this method significantly increases the time required to complete the backup. For a detailed description of how to start or schedule a full database backup with automatic database verification, see *Performing database verification using SnapManager* on page 138.

Explicitly start or schedule database verification only: With this method, a single operation can be initiated to verify the databases contained in one or more backup sets that have already been created. You can start the verification immediately, or you can schedule the verification to occur later, when it does not affect performance or delay later backups. For a detailed description of how to start or schedule database verification, see *Performing database verification using SnapManager* on page 138.

Defer verification until you restore from the backup set: If you attempt to restore from a backup set on which a database consistency check has not been run successfully, SnapManager prompts (but does not require) you to first verify the databases in that backup set. See "Importance of verifying databases to be restored" in *How SnapManager Restore works* on page 161.

Options for where to run SQL Server DBCC

Regardless of when you verify the databases in a backup set, the verification can be done on the production SQL Server (the Windows host system running the SQL Server instance used to create the databases) or on a remote verification system (another SQL Server).

From the production SQL Server: In the simplest SnapManager configuration, verification is run from the production SQL Server. However, because the Microsoft DBCC command used for the verification is CPU-intensive, performing the verification on the production SQL Server host system during peak usage could affect SQL Server performance.

From a remote verification server: Performing the verification on a remote system minimizes the impact of verification on SQL Server system resources and backup schedule. The requirements for a remote verification server are described in "Requirements for a remote verification server" in *Remote servers* on page 25. The procedure specifying a different SQL Server as the remote verification server is described in "Selecting the database verification server" *Configuring SnapManager application settings* on page 290.

Note: You can verify a database from a virtual SQL Server. For more information, see "Requirements for a remote verification server" in *Remote servers* on page 25.

Note: If the source, production SQL Server is a virtual machine, you cannot use a physical remote server to perform verifications.

Ways to manage the number of backup sets kept online

When planning your SnapManager backup schedules, you also need to manage the number of backup sets that are stored online.

Maximum number of databases per storage volume

It is strongly recommended that you put fewer than 35 databases on a storage volume, although you can have more than one LUN, SMB share, or VMDK on the same storage volume.

Note: It is possible for the total number of Snapshot copies on a volume to exceed the number of SnapManager backups being retained. For example, if a single volume contains both the SnapInfo directory and the SQL Server databases, each SnapManager backup generates two Snapshot copies on that volume.

SnapManager provides the following ways to manage and delete backups:

- Automatic deletion
- Explicit deletion

These two methods are described in-depth in the following subsections.

Automatic deletion of the oldest backups in a management group

When you start or schedule a full database backup, you can also specify the number of backup sets of that database to be retained for that backup management group. After the backup is complete, SnapManager will automatically delete the oldest backup sets for that database in the specified backup management group, retaining only the number of backups you want to preserve. SnapManager retention policy does not apply expiration days for individual backups, but manages how many backups are retained at any given time.

This is the recommended method for managing the number of backup sets you store on your system.

The procedural details are included in *Managing transaction log backups using SnapManager* on page 126.

Note that if a database is deleted, SnapManager for SQL Server stops actively managing the backups. The backups remain until manually deleted.

For more information about backup management groups, see and *Using backup management groups in backup and verification* on page 152.

Cases in which more backups are preserved

SnapManager does not count backups that failed verification when counting the number of stored backups. Therefore, more backups might be preserved than you specify in the "Delete Oldest Backups In Excess Of" box.

For example, suppose you are backing up databases A and B, which contain the following backup sets.

SnapManager backup set	Description
Database A	
`sqlsnap__orbit3_11-23-2004_16.21.07`	Old backup- good
`sqlsnap__orbit3__recent`	Recent backup- good
Database B	
`sqlsnap__orbit3_11-23-2004_16.21.07`	Old backup- good
`sqlsnap__orbit3__recent`	Recent backup- inconsistent

Also suppose you have set the "Delete Oldest Backups in Excess Of" box to `1` to preserve only one of each backup set, the most recent one.

In order to preserve one good backup for Database B, SnapManager does not delete the Snapshot copy `sqlsnap__orbit3_11-23-2004_16.21.07`. Therefore, two backups for Database B remain instead of one.

Option to retain up-to-the-minute restore ability

If you delete backups that are not the oldest backups in your backup list, the corresponding transaction logs are also deleted. This makes the older remaining backups no longer available for an up-to-the-minute restore. The reason is that the transaction logs are no longer contiguous from the time when the older backup was taken to the present time.

This can happen when you are deleting backups of a particular backup management group.

SnapManager for Microsoft SQL Server enables you to preserve the logs in this case, thereby retaining the ability to use the older backups in an up-to-the-minute restore.

Note: If you do not need to perform an up-to-the-minute restore from the older backups, allow the logs to be deleted to free up more space on the storage system holding the backups.

To balance up-to-the-minute restore needs with storage efficiency, you can configure the number or days of transaction log backups your system retains. See "Configuring the number of transaction log backups your system retains" in *Managing transaction log backups using SnapManager* on page 126.

Explicit deletion of backup sets

In addition to *automatically* deleting the oldest backup sets (an option that you can select when you start or schedule a backup operation), you can *explicitly* delete individual or multiple backup sets.

Explicit deletion of an individual backup With this method, you delete individual selected backup sets for either full database backups or transaction logs. The procedures are described in "Busy Snapshot error prevents deletion of backup set" in *Explicitly deleting backup sets using SnapManager* on page 154.

Explicit deletion of multiple backups With this method, you select a database to be deleted, the types of backup set components to be deleted (full database backups, transaction logs only, or SnapInfo directory backups), and the type of backup management group to be deleted. The procedure is described in "Deleting backups" in *Explicitly deleting backup sets using SnapManager* on page 154.

Note: You can also explicitly delete the LUN, SMB share, or VMDK Snapshot copies that were created during a restore operation. For a description of restore Snapshot copies, see *How SnapManager Restore works* on page 161. For a description of how to view and delete these Snapshot copies, see *Deleting restored Snapshot copies* on page 178.

When to run a SnapManager backup

You must balance frequency of backups against any overhead incurred by the database verification process. In addition, you must ensure that no SnapManager operations overlap with each other.

Backing up databases following data migration

At the end of the SQL Server database and transaction log migration process, the Configuration wizard reminds you to make an immediate backup of the SQL Server databases. Making an immediate backup of the SQL Server databases is critical because any previous non-SnapManager backups are no longer valid.

Frequency of backups

You do not have to perform multiple SnapManager full backups every day, but the more backups you make, the fewer SQL Server transaction logs need to be played forward at restore time. At a minimum, you should perform one SnapManager full database backup every 24 hours.

Recommendations for scheduling backups

The more often you create SnapManager backups, the fewer SQL Server transaction logs there are to be played forward at restore time, resulting in a faster restore. However, for best results, observe the following recommendations for scheduling backups and verifications:

- Do not schedule any SnapManager operations to overlap each other.
  Only one SnapManager operation can be running on the same machine at the same time.
- Do not schedule a backup to occur while a database verification is being performed, even if the verification is performed on a remote verification machine.
  This can result in a backup that cannot be deleted easily. For more information about this problem, see "Busy Snapshot error prevents deletion of backup set" in *Explicitly deleting backup sets using SnapManager* on page 154.
- Do not schedule verifications on the SQL server during peak usage hours.
  The verification process is CPU-intensive and could degrade SQL Server performance if run on the SQL Server during peak usage hours.

Note: To avoid degrading the performance of your production SQL Server, run your database verification operations on a remote server.

One way to conform to the preceding recommendations is to schedule your backups to occur during peak usage hours, and then use the off-peak hours to perform database integrity verifications.

Protecting databases by backing up, replicating, and archiving

SnapManager provides rapid online backup of databases using Snapshot technology that is part of Data ONTAP. When you define your backup options, you can increase your data protection by replicating Snapshot copies using SnapMirror and by archiving Snapshot copies using SnapVault. An alternative is to archive backups to third-party tape devices.

You need to complete preparatory steps before you use SnapMirror and SnapVault.

Related tasks

Preparing your environment for data protection on page 85

How SnapManager backup functions are accessed

To start or schedule a database backup or verification job, you can use either the Backup and Verify option or the SnapManager Backup wizard to specify the details of the operation you want SnapManager to perform. Depending on the specific parameters you select, various default SnapManager settings for backup operations and verification settings also come into play.

Backup and Verify

The SnapManager console root includes a Backup and Verify option that you can use to specify the job-specific parameters of a SnapManager backup operation or database verification.

This option can be used to start or schedule the following operations:

- "Full database backup using Backup and Verify" in *Backing up, replicating, and archiving databases using SnapManager* on page 115
- "Creating a transaction log backup using the Backup and Verify option" in *Managing transaction log backups using SnapManager* on page 126
- "Database verification using the Backup and Verify option" in *Performing database verification using SnapManager* on page 138

SnapManager Backup wizard

An alternative to the Backup and Verify option, the SnapManager Backup wizard guides you through the specification of the backup or verification operation you want performed.

This wizard can be used to start or schedule the following operations:

- "Full database backup using the Backup wizard" in *Backing up, replicating, and archiving databases using SnapManager* on page 115

- "Creating a transaction log backup using the Backup wizard" in *Managing transaction log backups using SnapManager* on page 126
- "Database verification using the Backup wizard" in *Performing database verification using SnapManager* on page 138

Default backup settings

The SnapManager for SQL Server-Backup dialog box enables you to view or change the default settings that pertain to SnapManager backup operations.

Various default values specified in this dialog box are used when you perform a full database backup a transaction log only backup, or a database verification in unverified backup sets. This is described in the following topics:

- "Information you need to specify for a full database backup" in *Backing up, replicating, and archiving databases using SnapManager* on page 115
- "Information you need to specify when creating a transaction log backup" in *Managing transaction log backups using SnapManager* on page 126
- "Information you need to specify for a database verification" in *Performing database verification using SnapManager* on page 138

For more information, see *SnapManager backup options* on page 298.

Default verification settings

The Verification Settings dialog box enables you to view or change the default settings that pertain to the verification of databases in SnapManager backup sets.

Various default values specified in this dialog box are used when you perform a full database backup, or when you verify the databases in unverified backup sets. This is described in the following topics:

- "Information you need to specify for a full database backup" in *Backing up, replicating, and archiving databases using SnapManager* on page 115
- "Information you need to specify for a database verification" in *Performing database verification using SnapManager* on page 138

You can have a different default verification server for each SQL Server host added to the SnapManager GUI.

Before starting a backup and verify the first time, review the verification mount point path. It is possible to exceed the maximum allowed path length, particularly if you accepted the default mount point. If the path length is long, change it to something such as `drive-letter:\mp`.

For more information on default verification settings, see *Database integrity verification options* on page 294.

Backing up, replicating, and archiving databases using SnapManager

SnapManager provides two ways for you to start or schedule a full database backup: using the Backup wizard or using the Backup and Verify option.

Information you need to specify for a full database backup

A full database backup operation is specified using a combination of parameters.

Job-specific parameters

Each time you start or schedule a full database backup, you must specify the following information in either the Backup and Verification window or in the Backup wizard.

- The databases you want to back up
- You can back up databases from the same host or remote hosts, including different SQL instances on the same host. You can configure these options in the "Backup and Verify" window using the "Federated Backup" option.
- When you select one database, SnapManager automatically selects all other databases that are located on the same Data ONTAP storage volume. SnapManager enables you to clear automatically selected databases. For example:
  - You can clear databases that reside on a different LUN from the databases you selected, even if the LUN is on the same storage volume. If the other LUN stores only a single database, you can clear or re-select that database individually. If the other LUN houses multiple databases, you must clear or re-select those databases as a group.
  - In the case of a *stream-based* full database backup, you can clear any automatically selected database. However, unless the selected databases share the same storage with other databases, SnapManager asks you to confirm your selection; backing up only a subset of the databases that reside on the same volume is not recommended. For more information about the stream-based and online Snapshot backup methods, see later in this topic.
- When you select databases *at the SQL Server instance level* and one of the selected databases cannot be backed up for an unexpected reason (such as the database being offline or in a loading state at the time of the backup), the full database backup operation progresses as follows:
  - The *backup report* includes a message at the beginning of the summary section that indicates that the backup was only partially completed because one or more databases were skipped.
  - A warning event is logged to the *event log*. The description field of this event contains the summary section of the report.
  - If *e-mail notification* is enabled, an email notification is sent to the configured email address.

    If the databases are moved to the local disk later, the scheduled backup operation skips backup deletion.

Note: For an instance-level *transaction-log-only* backup operation in which one of the selected databases cannot be backed up, the operation proceeds in the same manner as described above.

- Database has the full-text search option enabled, the *full-text search catalogs* are visible when you click the "+" next to the database name. The full text catalogs can be migrated, backed up, and restored along with the other files or file groups of the database.
- Which backup management group you want to assign to this backup
- For details, see *Using backup management groups in backup and verification* on page 152.
- The operation asks whether you want to automatically run a transaction log backup after the full database backup finishes.
- The operation asks whether you want to automatically delete the oldest full database backups within this backup management group (recommended to manage the number of Snapshot copies)
- For a description of this option, see "Automatic deletion of the oldest backups in a management group" in *Ways to manage the number of backup sets kept online* on page 109.
- If you select to automatically delete the oldest full database backups within this backup management group: the operation asks whether you also want to retain up-to-the-minute restore ability for all backups.
- For a description of this option, see "Option to retain up-to-the-minute restore ability" in *Ways to manage the number of backup sets kept online* on page 109.
- The operation asks whether you want to perform a database integrity verification of the backup set after the full database backup is complete
- You can back up an Availability Group from one or multiple replicas. You configure this in the Backup and Verify window using **Availability Group Backup** option.

  Use the Availability Group backup option to perform Availability Group level database backups. To do this, make the backup target the Availability Group, instead of databases. When a backup is on the Availability Group level, further backup policies can be specified, and the backup can happen potentially on any replica of the Availability Group when the backup is scheduled.

Note: Although it is possible to restore from an unverified backup, you should restore only from verified backups.

- The operation asks whether you want to run a command after the backup is complete (usually done to archive backups)
- The operation asks if the volumes you are backing up to are SnapMirror sources: whether you want the destination volumes to be updated after the full database backup is complete.
- For more information about this option, see *Replicating backups to mirrored volumes* on page 86.
- The operation asks whether you want to run the backup now or schedule it for later

Note: If you want to schedule the backup to run later, you also need to know the job scheduling information.

Backup settings The following list summarizes the backup settings that pertain to full database backups:

- The operation asks whether you want the backup set to be named using generic ("`__recent`") or unique (time-stamped) naming conventions
- For more information, see "SnapManager backup set naming conventions in "*How SnapManager backup data is organized* on page 99.
- The operation asks whether you want to verify the integrity of the live database *before* the backup is performed and whether you want to verify the integrity of the live database *after* the backup is performed

  Note: Verifying the integrity of the live database is a time-consuming operation. By default, neither of these options is selected.

The preceding options are configured using the Full Database Backup option of the SnapManager for SQL Server-Backup dialog box, described in "Configuring the profile of a full database backup" in *SnapManager backup options* on page 298.

The SnapManager for SQL Server-Backup dialog box can be accessed from the Backup and Verify option, as well as from the Backup wizard.

- If you are using the Backup and Verify option, you can open the SnapManager for SQL Server-Backup dialog box from the Actions pane.
- From within the Backup wizard, you can open the Backup Settings dialog box by clicking the Backup Settings button in the Advanced Backup Options screen.

Verification settings: The following list summarizes the settings that pertain to database verification:

- The operation asks which SQL Server is used to perform database verification
- This is configured using the SQL Server option of the Verification Settings dialog box, described in "Selecting the database verification server" in *Database integrity verification options* on page 294. If you will be specifying a remote verification server, be sure it is set up properly, as described in "Requirements for a remote verification server" in *Database integrity verification options* on page 294.

  Note: Be careful not to schedule backups while verification is in progress. Doing so can create a "busy Snapshot" which might cause problems when you attempt to delete some Snapshot copies.

- The following precautions will help you to avoid creating a busy Snapshot situation:
  - Do not schedule backups while a verification is in progress.
  - Do not create backup sets at the volume level or the SnapDrive level.
- For information about busy Snapshot copies, see "Busy Snapshot error prevents deletion of backup set" in *Explicitly deleting backup sets using SnapManager* on page 154.
- The operation asks which DBCC options are used to verify database backup sets
- This is configured using the DBCC Options option of the Verification Settings dialog box, described in "Selecting DBCC options" in *Database integrity verification options* on page 294.

The Verification Settings dialog box can be accessed from the Backup and Verify option, as well as from the Backup wizard:

- If you are using the Backup and Verify option, select "Verify most recent unverified Snapshot backups only" to open the Verification Settings tab.
- At least two of the most recent SnapManager Snapshot copies that were used for SnapVault updates need to be kept online on the primary storage system.
- You can use SnapManager to delete backup sets outside the backup process. When SnapVault is used, keep at least two of the most recent Snapshot copies used for the SnapVault updates until SnapVault updates for a given backup are complete. To check which Snapshot copies are required for the SnapVault relationships, use the SnapVault destinations `-s` command on the SnapVault primary storage system.
- When you use SnapManager to automatically delete older backup sets as part of a backup process, be sure to configure the "Delete backups In excess of" option to a number that is equal to or greater than two. If backups are automatically deleted based on time rather than quantity using the "Delete backups Older than" option, be sure to specify a number of days that will allow at least two of the Snapshot copies used for the SnapVault updates to remain online. This information also applies to deleting backups using the Delete Backup option from the SnapManager Action pane.

  Note: This applies only to Snapshot copies that are used for SnapVault updates.

- The SnapVault option is not available for VMDKs.
- From within the Backup wizard, you can open the Verification Settings dialog box by clicking "Verification Settings" button in the Verification Settings screen.

  Note: The Verification Settings screen appears only if you are specifying a "Full database backup" or a "Verify most recent backup set backups" operation; the SnapManager Backup wizard does not present this screen if you are specifying a "Transaction log backup only" operation.

Full database backup using Backup and Verify

To start or schedule a full database backup using Backup and Verify, complete the following steps.

With a full database backup you can choose to also back up the associated transaction logs after the database backup is complete. If you want to back up only transaction logs, see *Managing transaction log backups using SnapManager* on page 126.

You can run only one full database backup at a time. However, you can start a full database backup when a deferred verification is already running.

Note: For a list of information you need to provide as you complete these steps, see the section "Information you need to specify for a full database backup" which appears earlier in this topic.

Step	Action
1	In the SnapManager console root, double-click the server you want to use.
2	Select the Backup option in the Scope pane.

Step	Action
3	In the Results pane, select the databases for which you want to perform a full backup. When you select a database, SnapManager automatically selects all other databases that reside on the same storage system and storage system volume. When you select a database that resides on a VMDK, SnapManager automatically selects all the databases that reside on the VMDKs. For information about clearing any automatically selected databases, see the bullet "Which databases you want to backup" under "Job-specific parameters". **Note:** When SnapManager is running on VMDKs, you cannot select a physical SQL Server as the verification server. When you select databases at the *SQL Server instance level*, SnapManager reports any offline databases as skipped in the backup report. For more information, see "Information you need to specify for a full database backup" in *Managing transaction log backups using SnapManager* on page 126.
4	In the Actions pane, select "Backup and Verify." **Result** The "SnapManager for SQL Server-Backup" window appears.
5	**Optional:** To add databases from remote servers or from different SQL instances on the same server to this backup job, click the Federated Backup button, and complete the following substeps: **1.** Click Browse. A window opens, displaying all connected servers. **2.** Select the server that contains the database you want to add to the backup job. **3.** Enter the login details. **4.** Click Add. The window closes and the server and all databases therein are displayed as a tree in the top pane. **5.** Select the databases that you want to add to the backup job. **6.** Select a federated group. SnapManager does not support SAN and NAS databases from the same host in the same federated group. **7.** Click Add to Group. The databases are moved into the federated group. **Note:** You can add databases from multiple servers to the same federated group. Backing up a federated group backs up all databases in that group at the same time. **8.** Click OK.

Step	Action	
6	**Optional:** To take an Availability Group backup on single or multiple replicas, in the Databases(s) to back up window, select the databases to be backed up, and complete the following substeps: **1.** Click **Availability Group Backup**. **2.** Check either **Preferred backup replica only** or click **Advanced Options**. Check **Preferred backup replica only** to take a backup of only the preferred backup replica. The preferred backup replica is set through the SQL Server 2012 Availability Group properties dialog. Click **Advanced Option** to take backups on multiple replicas. You can back up only a subset of all of the replicas by setting **Replica Type** and setting the desired backup priorities. Backup priorities range from a minimum of 1 to a maximum of 100. **3.** Click **OK**.	
7	In the "Backup management group" option, select the management group for the backup you want to create: Standard, Daily, or Weekly. For more information, see "SnapManager backup set naming conventions" in *How SnapManager backup data is organized* on page 99 and *Using backup management groups in backup and verification* on page 152.	
8	Configure your system's up-to-the-minute restore ability. For more information, see "Configuring the number of transaction log backups your system retains" in *Managing transaction log backups using SnapManager* on page 126.	
9	If...	Then...
	You want the database backup operation to be immediately followed by a transaction log backup	Select the "Run transaction log backup after full database backup" option.
	You want to schedule the transaction log backup yourself later	See *Managing transaction log backups using SnapManager* on page 126. **Note:** When you schedule a transaction log backup, ensure that the full backup and transaction log backup do not coincide.
10	If...	Then...
	You want to delete backups older than a specific number of days	Enter the number of days in the "Older than" field
	You want to delete backups more than a specified number of backups	Enter the number of backups in the "In excess of" field

Step	Action
11	If you want to verify databases after the backup operation, select "Verify databases after backup."
12	If you want to run a command or script prior to performing the backup or after the backup finishes, select the "Run Command" option. You can run a command after performing a backup to automatically archive the backup. **Result** If you select this option, SnapManager displays the Run Command dialog box. For more information, see *Pre-command and post-command script settings* on page 303.
13	Under SnapMirror options, select the corresponding check box if you want to update SnapMirror after operation or to verify available SnapMirror destination volumes. **Note:** If the system is configured for SnapMirror replication, only the databases on mirrored volumes will be updated on the SnapMirror destination volumes.
14	Under Backup archiving options, select the options to archive the backups using SnapVault.
Back up now or schedule for later	
15	You can either run the backup now or schedule it for later. Click one of the following tabs: • Backup Now • Schedule
To complete this procedure by scheduling the backup:	
16	If you select "Schedule...", complete the procedure by scheduling the backup. For details, see *Scheduling a backup job or a database verification job* on page 145.
To complete this procedure by starting the backup:	

Step	Action
17	If you select "Backup Now..." do the following: **1.** Read the items displayed in the Backup Task List. This list shows the progress of the backup operation after you start it. **2.** When you are ready to start the backup operation, click Start Now. **Result** The backup operation is performed, and each item in the Backup Task List is checked off as the task is complete. • You can toggle the Backup Status dialog box between two different views: Backup Task List view and Backup Report view. • The Backup Report view displays detailed progress information as the backup progresses. You can also print this information by using the Print Report button. • If the backup is successful, the Backup Task List view shows the check-off list with the tasks completed. • If Notification is enabled, an email is sent and the event is posted to the Windows Application event log.

Full database backup using the Backup wizard

To start or schedule a full database backup using the Backup wizard, complete the following steps.

Note: For a list of information you need to provide as you complete these steps, see the section "Information you need to specify for a full database backup" which appears earlier in this topic.

Step	Action
1	In the SnapManager for SQL Server console root, click the SnapManager Backup wizard icon. **Result** The SnapManager Backup wizard starts and displays the Welcome screen.
Welcome	
2	Click Next. **Result** The Databases to Backup or Verify screen appears. The *Microsoft SQL Servers* navigation tree in the left panel lists the SQL Server databases that are managed from the current SQL Server. Databases that reside on the same storage system and storage system volume are shown with disk icons of the same color.
Databases to Backup or Verify	

Step	Action	
3	In the left panel, click to select the databases you want to backup. You can back up all databases on an SQL server by selecting the server instead of selecting individual databases. When you select the server, even databases created after configuring the backup are backed up. If you have more than one SQL instance, you must repeat this process for each instance; however, instead of the selecting the server, you should select the instance. When you select a database, SnapManager automatically selects all other databases that reside on the same storage system volume. For information about deselecting any automatically selected databases, see the bullet "Which databases you want to backup" under "Job-specific parameters" in *Managing transaction log backups using SnapManager* on page 126. When you select databases at the *SQL Server instance level*, SnapManager lists any offline databases as skipped in the backup report. For more information, see the section "Information you need to specify for a full database backup" which appears earlier in this topic.	
4	Click Next. **Result** The Backup or Verify Databases and Transaction Logs screen appears.	
Backup or Verify Databases and Transaction Logs		
5	Select the Backup Databases and Transaction Logs option, and then click Next. **Result:** The Select to Apply Backup Option window appears. **Note:** If you do not select the Backup Databases and Transaction Logs option and select to schedule the transaction log backup yourself, ensure that the full backup and transaction log backup operations do not coincide.	
Backup Option		
6	If you want to...	Then...
	Add databases from remote servers or from different SQL instances	Select "Federated Backup Options." For more information, see Step 5 in "Full database backup using Backup and Verify" which appears earlier in this topic.
	Take backups of single or multiple replicas of an Availability Group	Select "Availability Group Database Backup, and then select from Preferred Backup Replica Only, Primary, Secondary, All and optionally, Backup Priority.
	Only back up databases from the selected server	Select "Continue with selected databases."
Select SQL Server backup Type		

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>7</td><td>Select the Full Database Backup option, and then click Next.
Result The "Select backup management group for this backup" screen appears.</td></tr>
<tr><td colspan="2">Select backup management group for this backup</td></tr>
<tr><td>8</td><td>Follow the instructions in the remainder of the Backup wizard screens.
The following screens enable you to specify the details of a full database backup:
• Backup Management Group
• Backup Transaction Log After Full Database Backup
• Delete the Oldest Full Backups
• Retain Up-to-the-Minute Restore Ability for Older Backups
Note: This screen appears only if you selected the Delete the Oldest Backups option in the previous screen.
• Verify the Databases in this Backup
• View or Change Verification Settings
Note: This screen appears only if you chose to automatically verify the databases when the backup is created.
• SnapMirror options and Backup archiving options (SnapVault)
Note: This screen appears only if any SQL Server databases or SnapInfo directories reside on a SnapMirror source volume.
• Advanced Backup Options
• Run a command or script with the current operation
For a list of information you need to provide, see the section "Information you need to specify for a full database backup" which appears earlier in this topic.</td></tr>
<tr><td colspan="2">Backup Finish</td></tr>
<tr><td>9</td><td>The "Backup Finish" screen prompts you to choose whether you want the operation to be performed immediately or scheduled for a later time.
<table><tr><td>If you want to...</td><td>Then...</td></tr><tr><td>Run the backup now</td><td>Go to Step 10.</td></tr><tr><td>Schedule the backup for later</td><td>Go to Step 12.</td></tr></table></td></tr>
<tr><td colspan="2">To run the backup now:</td></tr>
<tr><td>10</td><td>If you want to run the backup, Click Finish.
Result The Backup Status screen appears and displays the backup settings you have selected.</td></tr>
</table>

Step	Action
11	In the Backup Status window, click Start Now. **Result** The backup is performed and the backup set is written to the volume. • The Backup Status dialog box appears and displays a Backup Task List that is used to show the progress of the backup operation after you start it. • SnapManager Backup completes each task and checks it off on the list shown in the Backup Task List view. • You can switch back and forth between the task check-off list and the progress report using the Switch buttons on either view. • If the backup is successful, the Task window shows the check-off list with the tasks completed. • If Notification is enabled, an email is sent and the event is posted to the Windows Application event log.
To schedule the backup for later:	
12	If you want to schedule the backup for later, do the following: **1.** Select the Schedule option. **2.** In the Job Name box, enter a name for this job. **3.** If you want this job name to overwrite a job of the same name (if it exists), select the Replace Job if Exists option. For more information, see *Scheduling a backup job or a database verification job* on page 145.
13	Under "Select the Scheduling Service to Create Job," select either SQL Server Agent or Windows Scheduled Tasks. For more information, see "*Scheduling a backup job or a database verification job* on page 145.
14	Click OK. If your backup job is not scheduled, you are taken to the Microsoft SQL Server Management Studio for scheduling. You cannot use the option Windows Scheduled Tasks if your backup job is not scheduled.
15	After the job is scheduled, SnapManager exits the Backup wizard.

Managing Availability Group transaction log backups

Before SnapManager can manage transaction log backups, you must set a number of options.

Before you begin

The SnapManager connection to the node in the Availability Group is already configured. The node can be primary or secondary.

Note: SnapManager must be configured to connect to a node in the SQL Server cluster, not to the Availability Group endpoint name itself.

Steps

1. Select **Backup** under the server.
2. Select **Backup Settings** in the **Actions** window.
3. Click the **Transaction Log Backup** tab.
4. Under **Repository Log backup Options**, specify options for copying transaction log backups to shares:
   a. Click either **Apply to all Databases** or **Apply to only Availability Group Databases**.
   b. Optional: Click **Delete Share Log backups** and choose how the backups are selected for deletion.

   Note: Copy transaction log backup to share is enabled by default for Availability Group databases.

Managing transaction log backups using SnapManager

Methods of managing transaction log backups

SnapManager provides two ways for you to start or schedule a transaction log backup: using the SnapManager Backup wizard or using the Backup and Verify option. You can also configure how many transaction log backups your system retains in the Up-to-the-minute Restore Ability Settings window.

Note: The topics in this section describe how to start or schedule a SnapManager backup of SQL Server transaction logs only. If you want to backup transaction logs as a follow-on task to a successful full database backup see *Backing up, replicating, and archiving databases using SnapManager* on page 115.

Information you need to specify when creating a transaction log backup

A backup of only transaction logs is specified using a combination of parameters.

Job-specific parameters

Each time you start or schedule a transaction-log-only backup, a full database backup must exist for that database. You can specify the following information in either the Backup and Verification option or in the Backup wizard:

- Which databases you want to back up
- When you select one database, SnapManager automatically selects all other databases that are located on the same Data ONTAP storage volume. You can clear databases that reside on different storage from the databases you selected. For example:
  - If the other LUN stores only a single database, you can clear or reselect that database individually.
  - If the other LUN houses multiple databases, you must clear or reselect those databases as a group.
- When you select databases at the SQL Server instance level and one of the selected databases cannot be backed up for an unexpected reason (such as the database being offline or in a loading state at the time of the backup), the transaction-log-only backup operation happens as follows:
  - The *backup report* includes a message at the beginning of the summary section that indicates that the backup was only partially completed because one or more databases were skipped.
  - A warning event will be logged to the *event log*. The description field of this event contains the summary section of the report.
  - If *e-mail notification* is enabled, an email notification will be sent to the configured email address.
  - When backup is scheduled for future, existing backups are not deleted unless some of the databases in the server instance are first moved to the local disk.

    If the databases are moved to the local disk later, the scheduled backup operation skips backup deletion. The backup is re-created and rescheduled so that for future operations, backups are deleted first.

    Note: For an instance-level *transaction-log-only* backup operation in which one of the selected databases cannot be backed up, the operation will proceed in the same manner as described above.
- The operation asks whether you want to automatically delete the oldest transaction log backups (recommended to manage the disk space occupied by the SnapInfo directory)
- For a description of this option, see "Automatic deletion of the oldest backups in a management group" in *Ways to manage the number of backup sets kept online* on page 109.
- The operation asks whether you want to run a command either before the backup starts or after the backup is complete (frequently done to archive backups)
- This feature is frequently used to archive the backup.

- If the transaction logs you are backing up are for databases and related SnapInfo directories that are located on SnapMirror sources: whether you want the destination volumes to be updated after the transaction log backup is complete
- For more information about this option, see *Replicating backups to mirrored volumes* on page 86.
- The operation asks whether you want to run the backup now or schedule it for later

Note: If you want to schedule the backup to run later, you also need to know the job scheduling information.

Backup settings The following list summarizes the backup settings that pertain to a transaction-log-only backup. These settings enable tasks that can be performed after the transaction log backup finishes successfully.

The Backup settings help you determine the following:

- Whether you want to create a backup set that contains the SnapInfo directory.
- If you create a backup set of the SnapInfo directory after the backup is complete: whether you also want to delete the oldest SnapInfo directory Snapshot copies and retain only a certain number of the most recent backup sets.
- Whether you want to truncate the transaction log itself. Selecting this option enables you to manage the size of the transaction log.
- What the transaction log settings are for a SnapManager share. See *Managing transaction log backups using SnapManager* on page 126 for more information.

The preceding options are configured using the Transaction Log Backup option of the SnapManager for SQL Server-Backup dialog box, described in "Configuring the profile of a transaction log backup" in *SnapManager backup options* on page 298.

The SnapManager for SQL Server-Backup dialog box can be accessed from the Backup and Verify option, as well as from the Backup wizard.

- From within the Backup wizard, you can open the Backup Settings dialog box by selecting the Backup Settings button in the Advanced Backup Options screen.
- If you are using the Backup and Verify option, you can open the Backup Settings dialog box from the Actions pane.

For more details, see "Configuring the profile of a transaction log backup" in *SnapManager backup options* on page 298.

Creating a transaction log backup using the Backup and Verify option

To start or schedule a backup of only transaction logs using the Backup and Verify option, complete the following steps. If you want to back up transaction logs automatically after a full database backup is complete, see *Backing up, replicating, and archiving databases using SnapManager* on page 115 instead.

Note: If you select to delete older backups and do not select the check box "Create snapshot of SnapInfo drive after backup," SnapManager for SQL Server will override the option specified by this check box and automatically create a snapshot of the SnapInfo drive.

Note: For a list of information you need to provide as you complete these steps, see "Information you need to specify when creating a transaction log backup" in *Managing transaction log backups using SnapManager* on page 126.

Step	Action
1	In the SnapManager console root, click the Backup and Verify option.
2	In the Results pane, select the databases you want to backup. When you select a database, SnapManager automatically selects all other databases that reside on the same storage system volume. For information about deselecting any automatically selected databases, see the bullet "Which databases you want to backup" under "Job-specific parameters" in *Managing transaction log backups using SnapManager* on page 126. When you select databases at the SQL Server instance level, SnapManager reports any offline databases as skipped in the backup report. For more information, see "Information you need to specify when creating a transaction log backup" in *Managing transaction log backups using SnapManager* on page 126.
3	In the SnapManager for SQL Server-Backup dialog box, select the "Transaction log backup" option. **Result** SnapManager provides a choice of backup options that pertain to backing up only transaction logs.

Step	Action
4	**Optional:** To add databases from remote servers or from different SQL instances on the same server to this transaction log backup job, click the Federated Backup button. **1.** Click Browse. A window opens, displaying all connected servers. **2.** Select the server that contains the database you want to add to the backup job. **3.** Enter the login details. **4.** Click Add. The window closes and the server and all databases therein are displayed as a tree in the top pane. **5.** Select the databases that you want to add to the backup job. **6.** Select a federated group. SnapManager does not support SAN and NAS databases from the same host in the same federated group. **7.** Click Add to Group. The databases are moved into the federated group. **Note:** You can add databases from multiple servers to the same federated group. Backing up a federated group backs up all databases in that group at the same time. **8.** Click OK.
5	If you want the transaction log backup to be followed by a verification, select the "Verify log backup upon completion" option.
6	If you want to automatically delete older transaction log backups, select the "Delete log backups in excess of" option and select the number of transaction log backups you want to keep. **Note:** If you select the "Delete log backups in excess of" check box and you do not select the check box "Create snapshot of SnapInfo drive after backup", SnapManager for SQL Server will override the option to create a Snapshot copy of the SnapInfo drive and will automatically create a snapshot of the SnapInfo drive. Selecting this option enables you to manage the number transaction log backups.
7	If you want to automatically delete transaction log backups older than a specified number of days, select the "Delete log backups in older than" option and select the number of days. Selecting this option enables you to manage the number transaction log backups.

Step	Action
8	If you want to run a command after the backup finishes, select the Run Commands option. This is usually done to archive backups. **Note:** If you select this option, SnapManager prompts you for the details when you are ready to start or schedule the backup operation.
Back up now or schedule for later	
9	You can either run the backup now or schedule it for later. Click one of the following buttons: • Backup Now... • Schedule...
To complete this procedure by scheduling the backup:	
10	If the Schedule Job dialog box appears, complete this procedure by scheduling the backup. For details, see *Scheduling a backup job or a database verification job* on page 145.
To complete this procedure by starting the backup:	
11	If the Backup Status dialog box appears, do the following: **1.** Read the items displayed in the Backup Task List. This list is used to show the progress of the backup operation after you start it. **2.** When you are ready to start the backup operation, click Start Now. **Result** The backup operation is performed, and each item in the Backup Task List is checked off as the task is complete. • You can toggle the Backup Status dialog box between two different views: Backup Task List view and Backup Report view, by using the Switch button on either view. • The Backup Report view displays detailed progress information as the backup progresses. You can also print this information by using the Print Report button. • If the backup is successful, the Backup Task List view shows the check-off list with the tasks completed. • If Notification is enabled, email is sent and the event is posted to the Windows Application event log.

Creating a transaction log backup using the Backup wizard

To start or schedule a backup of only transaction logs using the Backup wizard, complete the following steps.

Note: For a list of information you need to provide as you complete these steps, see "Information you need to specify when creating a transaction log backup" in *Managing transaction log backups using SnapManager* on page 126.

Step	Action
1	In the SnapManager for SQL Server console root, click the SnapManager Backup Wizard option in the Actions pane. **Result** The SnapManager Backup Wizard starts and displays the Welcome screen.
Welcome	
2	Click Next. **Result** The Databases to Backup or Verify screen appears.
Databases to Backup or Verify	
3	In the left panel, click to select the databases you want to backup, and then click Next. When you select a database, SnapManager automatically selects all other databases that reside on the same storage system volume. For information about deselecting any automatically selected databases, see the bullet "Which databases you want to backup" under "Job-specific parameters" in *Managing transaction log backups using SnapManager* on page 126. When you select databases at the SQL Server instance level, SnapManager reports any offline databases as skipped in the backup report. For more information, see "Information you need to specify when creating a transaction log backup" in *Managing transaction log backups using SnapManager* on page 126. **Result** The Backup or Verify Databases and Transaction Logs screen appears.
Back up or Verify Databases and Transaction Logs	
4	Select the Backup Databases and Transaction logs option, and then click Next. **Result** The SQL Server backup type screen appears.
SQL Server backup Type	
5	Select the Transaction log backup only option, and then click Next. **Result** The "Delete the oldest transaction log backups" screen appears.
Delete the oldest transaction log backups	

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>6</td><td>Follow the instructions in the remainder of the Backup wizard screens. The following screens enable you to specify the details of a transaction log backup only:
Delete the Oldest Transaction Log Backups
Verify the Transaction Logs in this Backup
Note: To view and change the verification settings, click Verification Settings.
Option to Perform SnapMirror Update After Operation
Note: This screen appears only if any SQL Server databases or SnapInfo directories reside on a SnapMirror source volume.
Run a Command After the Operation
For a list of information you need to provide, see "Information you need to specify when creating a transaction log backup" in *Managing transaction log backups using SnapManager* on page 126.</td></tr>
<tr><td colspan="2">Backup Finish</td></tr>
<tr><td>7</td><td>The Backup Finish screen prompts you to choose whether you want the operation to be performed now or scheduled for a later time.<table><tr><td>If you want to...</td><td>Then...</td></tr><tr><td>Run the backup now</td><td>Go to Step 9.</td></tr><tr><td>Schedule the backup for later</td><td>Go to Step 11.</td></tr></table></td></tr>
<tr><td colspan="2">To run the backup now:</td></tr>
<tr><td>8</td><td>If you want to run the backup, click Finish.
Result The Backup Status screen appears and displays the backup settings you have selected.</td></tr>
</table>

Step	Action
9	In the Backup Status window, click Start Now. **Result** The backup is performed. • The Backup Status dialog box appears and displays a Backup Task List that is used to show the progress of the backup operation after you start it. • SnapManager Backup completes each task and checks it off on the list shown in the Backup Task List view. • You can switch back and forth between the task check-off list and the progress report using the Switch buttons on either view. • If the backup is successful, the Task window shows the check-off list with the tasks completed. • If Notification is enabled, an email is sent and the event is posted to the Windows Application event log.
To schedule the backup for later:	
10	If you want to schedule the backup for later, do the following: **1.** Select the Schedule option. **2.** In the Job Name box, enter a name for this job. **3.** If you want this job name to overwrite a job of the same name (if it exists), select the Replace Job if Exists option. For more information, see *Scheduling a backup job or a database verification job* on page 145.
11	Under "Select the Scheduling Service to Create Job," select either SQL Server Agent or Windows Scheduled Tasks. For more information, see *Scheduling a backup job or a database verification job* on page 145.
12	Click OK. If your backup job is not scheduled, you are taken to the Microsoft SQL Server Management Studio for scheduling.
13	After the job is scheduled, SnapManager takes you out of the Backup wizard.

Configuring the number of transaction log backups your system retains

Configuring the number of transaction log backups your system retains enables you to balance up-to-the-minute restore needs with storage efficiency. The more transaction log backups your system retains, the more complete your up-to-the-minute restores can be, but the less efficient your storage is because of the amount of storage consumed.

To configure the number of transaction log backups your system retains, complete the following steps.

Note: You should keep a limited number of transaction logs to avoid running out of storage system disk space.

Step	Action	
1	In the SnapManager console root, double-click the server you want to use.	
2	Select the Backup option in the Scope pane.	
3	In the Actions pane, click Backup and Verify.	
4	In the Backup Settings pane, click Up-to-minute Restore Options. **Result:** The Retain Up-to-minute Restore Ability Settings window opens.	
5	In the Retain Up-to-minute Restore Ability Settings window, choose from the three available options.	
	Option	Description
	Backups generated in the last [days]	Retains all transaction log backups generated during the number of days you define; automatically deletes logs older than the defined limit.
	The most recent [number of backups]	Retains a set number of the most recent transaction log backups; automatically deletes logs exceeding the defined limit.
	All the older backups [Do Not delete any logs]	Retains all transaction log backups.
6	Choose whether to run the backup now or schedule it for later by clicking the applicable button:	
	Schedule	Go to Step 7.
	Backup Now	Go to Step 8.
To schedule the backup:		
7	Follow the instructions in *Scheduling a backup job or a database verification job* on page 145.	
To back up the database now:		
8	Read the items displayed in the Backup Task List. This list shows the progress of the backup operation after you start it.	

Step	Action
9	When you are ready to start the backup operation, click Start Now. **Result** The backup operation is performed, and each item in the Backup Task List is checked off as the task is completed. • You can toggle the Backup Status dialog box between two different views: Backup Task List view and Backup Report view, by using the Switch button on either view. • The Backup Report view displays detailed progress information as the backup progresses. You can also print this information by using the Print Report button. • If the backup is successful, the Backup Task List view shows the check-off list with the tasks completed. • If Notification is enabled, email is sent and the event is posted to the Windows Application event log.

What to do if a SnapManager backup operation fails

If a SnapManager backup fails, check the backup report for details about what SnapManager was trying to do when the failure occurred. SnapManager reports are described in *Managing SnapManager operational reports* on page 200. You can also review the common backup failures that follow.

SnapInfo directory being accessed

Because a SnapManager backup might include renaming a SnapInfo subdirectory and Windows does not allow a directory name to be changed while it is being accessed, accessing the SnapInfo directory with a tool such as Windows Explorer could cause the backup to fail. Make sure that you do not hold any exclusive access to the SnapInfo directory on the SQL Server host system while a backup is progress.

SnapInfo directory out of space

Expand the LUN that contains the SnapInfo directory or, in the case of SMB shares, expand the volume.

Note: When you expand a LUN, ensure that enough space remains in the volume for backup set creation, so that SnapManager can continue to function correctly.

Data does not match

This error occurs if you made changes to your SQL Server database configuration after SnapManager was started. Any of the following actions refresh your SnapManager view:

- Press F5 on your keyboard.

- From the SnapManager console root, click **Action** > **Refresh**.
- Restart SnapManager.

Backup set already exists

Either of the following circumstances might cause this error to occur:

- The system clock on the host running SnapManager might not be synchronized with the clock on the storage system. These two clocks must be synchronized for SnapDrive to function correctly. For more information, see the *Data ONTAP Software Setup Guide* for your version of Data ONTAP.
- If a SnapMirror replication job is running when you attempt to begin a SnapManager backup, the backup can fail. You can avoid this problem by making sure that SnapMirror replications have enough time to finish before you begin another SnapManager backup.

SnapManager server initialization failed

To correct this issue, close the SnapManager for Microsoft SQL Server GUI, restart the SnapManager service from within the services.msc console, and then reopen the SnapManager GUI.

VMDK backup fails when you specify a physical server as the verification server

The backup copy created on the VMDK cannot be verified on a physical server. To resolve this error, select a verification server running on a virtual machine.

Database is not in valid configuration

A SnapManager backup operation might fail with the following message in the backup report:

```
WARNING: Database DatabaseName of ServerName is not in valid configuration,
and will not be included in this backup.
```

This message can appear if any of the following is true:

- The database is not in the online state.
  For example, if autoshrink is enabled on the database, the database is not online during the shrink operation. For information about autoshrink, see *Microsoft article 315512*.
- The database configuration is not supported with SnapManager.
  To verify that your configuration is supported, see “SQL Server configuration rules with SnapManager” and “SQL Server configurations supported with SnapManager.”

[DBMSLPCN] ConnectionRead (WrapperRead())

If the SnapManager host system is running SQL Server 2005, a SnapManager backup operation might fail with the following message in the backup report:

```
[Microsoft][ODBC SQL Server Driver][DBMSLPCN]ConnectionRead
(WrapperRead()).
```

To avoid this problem, install MDAC 2.8 SP1 on the Windows host. See "Windows host system requirements" in *Verifying Windows host system requirements* on page 20.

If Readable Secondary set is to "No"

If any of a replica's secondaries have Readable Secondary set to "No" and that replica is still part of the Availability Group backup, SnapManager displays an error similar to this:

```
The target database, 'test1', is participating in an availability group and
is currently not accessible for queries. Either data movement is suspended
or the availability replica is not enabled for read access. To allow read-
only access to this and other databases in the availability group, enable
read access to one or more secondary availability replicas in the group.
For more information, see the ALTER AVAILABILITY GROUP statement in SQL
Server Books Online.
```

To correct this, change Readable Secondary to Yes for all of the replicas participating in backups.

If the Availability Group was "Cloned as Replica" and the user takes an AG backup across all replicas

SnapManager can clone the AG to the selected replica with the **Readable Secondary** backup option set to "No", but afterwards, if the user tries to select the same AG and tries to take a backup across all replicas, the backup of the cloned AG is skipped, because the databases are mounted as read-only.

The problem occurs because the requested database has not been configured for backup. Instead, the database is marked as "Database is on a LUN backed by snapshot." The message is similar to `[02:14:38.145] [SQL2012HA3] Database requested has not been configured for backup. This database is marked as: Database is on a LUN backed by snapshot. Server: SQL2012HA2 Database:1ag-db1. This Database will be skipped.`

Availability Group database status

SnapManager skips the backup of an Availability Group if any of the replica databases are not in Synchronizing or Synchronized states.

Performing database verification using SnapManager

If you created a full database backup set without automatically verifying the databases at the time the backup was created, you can verify the databases in that backup as a separate operation.

Attention: If you attempt to perform a database verification (or a backup with database verification) when SnapManager is running from a Terminal Services client instead of from a system console, the operation fails.

You can schedule more than one deferred verification to run at the same time.

Information you need to specify for a database verification

A database integrity verification job is specified using a combination of parameters.

Job-specific parameters: Each time you start or schedule a database verification, you must specify the following information in either the Backup wizard or in the Backup and Verify option:

- The databases for which you want to verify any unverified backup sets when you select one database. SnapManager automatically selects all other databases that are located on the same Data ONTAP storage volume. You can clear automatically selected databases. For example:
  - In the case of a virtual machine containing VMDKs, all the databases residing on the VMDKs are selected automatically whether the VMDK resides on the same or different datastores.
  - If the other LUN stores only a single database, you can clear or reselect that database individually.
  - If the other LUN houses multiple databases, you must clear or reselect those databases as a group.
- Within the selected databases, the backup management groups for which you want to verify any unverified backup sets
- For more information, see *Using backup management groups in backup and verification* on page 152.
- For the selected databases and backup management groups, the number of unverified backup sets you want to verify

  Note: If you request verification of a greater number of unverified Snapshot copies than specified by your database and backup management group selections, the verification will proceed, and therefore all backups will be verified.
- Whether you want to run a command after the backup is complete (usually done to archive backups)
- This feature is typically used to automatically archive a backup.
- If the volumes you are backing up to are SnapMirror sources: whether you want the destination volumes to be updated after the database verification is complete

  Note: The result of the database verification operation (the database integrity status) is written to the backup set, in the SnapInfo directory. By replicating the backup set to the mirrored volume, this status information is kept current on the mirrored volume.

  For more information, see *Replicating backups to mirrored volumes.* on page 86.
- Whether you want to run the verification now or schedule it for later

  Note: If you want to schedule the verification to run later, you also need to know the job scheduling information.

Verification settings: The following list summarizes the settings that pertain to database verification:

- Which SQL Server is used to perform database verification
- This is configured using the SQL Server option of the Verification Settings dialog box, described in "Selecting the database verification server" in *Database integrity verification options* on page

294. If you specify a remote verification server, be sure it is set up properly, as described in "Requirements for a remote verification server" in *Database integrity verification options* on page 294.

- Which DBCC options are used to verify database backup Snapshot copies
- This is configured using the DBCC Options option of the Verification Settings dialog box, described in "Selecting DBCC options" in *Database integrity verification options* on page 294.

The Verification Settings dialog box can be accessed from the Backup and Verify option, as well as from the Backup wizard.

- If you are using the Backup and Verify option, you can open the Verification Settings dialog box by selecting "Verify most recent unverified Snapshot backups only".
- From within the Backup wizard, you can open the Verification Settings dialog box by clicking Database Verification Settings in the View or Change Database Verification Options screen.

Note: The View or Change Database Verification Options screen appears only if you are specifying a "Full database backup" operation or a "Verify most recent Snapshot backups" operation; the SnapManager Backup wizard does not present this screen if you are specifying a "Transaction log backup only" operation.

Database verification using the Backup and Verify option

To start or schedule database verification, complete the following steps *from the production SQL Server host system* (not from the remote verification server).

Note: If the FlexClone license is not enabled and you run database verification either remotely or locally, you must be careful not to schedule backups while verification is in progress. Doing so can create a "busy Snapshot" which might cause problems when you attempt to delete some Snapshot copies. For information about busy Snapshot copies, see "Busy Snapshot error prevents deletion of backup set" in *Explicitly deleting backup sets using SnapManager* on page 154.

Step	Action
1	On the production SQL Server, click the Backup and Verify option.
2	Select "Verify most recent unverified Snapshot backups only."
3	Select the number of the most recent unverified backups you want to verify. **Note:** Only unverified backups are counted. For example, if you select **2** , and all the databases contained in the most recent backups have already been verified, then SnapManager verifies the databases in the two previous backups.

Step	Action
4	In the Backup Management Group option, select the backup management group of the backups you want to verify. If you want to verify the most recent backups regardless of their backup management group, select All.
5	If you want to run a command either before the database verification starts or after the database verification finishes, select the Run Command Settings option. **Result** If you select this option, SnapManager displays the Run Command dialog box. For more information, see *Pre-command and post-command script settings* on page 303.
6	If your volume is a SnapMirror source volume and you do not want the destination volume to be updated after this verification is complete, clear the "Update SnapMirror after operation" option. **Note:** The result of the database verification operation (the database integrity status) is written to the backup set, in the SnapInfo directory. By replicating the backup set to the mirrored volume, this status information is kept current on the mirrored volume.
Verify now or schedule for later	
7	You can either run the verification now or schedule it for later. Click one of the following buttons: • Verify Now... • Schedule...
To complete this procedure by scheduling the verification	
8	If the Schedule Job dialog box appears, complete this procedure by scheduling the backup. For details, see *Scheduling a backup job or a database verification job* on page 145.
To complete this procedure by starting the verification:	

Step	Action
9	If the Backup Status dialog box appears, do the following: **1.** Read the items displayed in the Backup Task List. This list is used to show the progress of the verification operation after you start it. **2.** When you are ready to start the verification operation, click Start Now. **Result** The verification operation is performed, and each item in the Backup Task List is checked off as the task is complete. • You can toggle the Backup Status dialog box between two different views, Backup Task List view and Backup Report view, by using the Switch button on either view. • The Backup Report view displays detailed progress information as the verification progresses. You can also print this information by using the Print Report button. • If the backup is successful, the Backup Task List view shows the check-off list with the tasks completed. • If Notification is enabled, email is sent and the event is posted to the Windows Application event log.

Database verification using the Backup wizard

To start or schedule a database verification using the Backup wizard, complete the following steps.

Note: For a list of information you need to provide as you complete these steps, see "Information you need to specify for a database verification" in *Performing database verification using SnapManager* on page 138.

Step	Action
1	In the SnapManager for SQL Server console root, click Backup Wizard. **Result** The SnapManager Backup Wizard starts and displays the Welcome screen.
Welcome	
2	Click Next. **Result** The Databases to Backup or Verify screen appears.
Databases to Backup or Verify	
3	In the left panel, click to select the databases you want to verify, When you select a database, SnapManager automatically selects all other databases that reside on the same storage system volume. For information about deselecting any automatically selected databases, see the bullet "Which databases you want to backup" under "Job-specific parameters" in *Managing transaction log backups using SnapManager* on page 126.

Step	Action
4	Click Next. **Result** The Backup or Verify Databases and Transaction Logs screen appears.
Backup or Verify Databases and Transaction Logs	
5	Specify the number of database backup Snapshot copies you want to verify: **1.** Select the "Verify Database and transaction logs in the" option. **2.** In the "most recent unverified backups" option, select the number of database backup Snapshot copies to verify. **3.** Click Next. **Result** The "Select the backup management group for this backup" screen appears.
Select the backup management group for this backup	
6	Follow the instructions in the remainder of the Backup wizard screens. The following screens enable you to specify the details of a database verification: • Backup Management Group • View or Change Verification Settings • Option to Perform SnapMirror Update After Operation **Note:** This screen appears only if any SQL Server databases or SnapInfo directories reside on a SnapMirror source volume. • Run a Command After the Operation For a list of information you need to provide, see "Information you need to specify for a database verification" in *Performing database verification using SnapManager* on page 138.
Backup Finish	
7	The Backup Finish screen prompts you to choose whether you want the operation to be performed now or scheduled for a later time.

If you want to...	Then...
Run the database verification now	Go to Step 8.
Schedule the database verification for later	Go to Step 11.

To run the database verification now:	
8	If you want to run the database verification immediately, click Verify.

Step	Action
9	After you verify that all the settings in the window are correct, go to the Completing the Backup Wizard dialog box and click Finish. **Result** The Backup wizard closes, and the Backup Status window appears, and displays a Backup Task List that will be used to show the progress of the database verification operation after you start it.
10	In the Backup Status window, click Start Now. **Result** The database verification is performed. • The Backup Status dialog box appears and displays a Backup Task List that is used to show the progress of the backup operation after you start it. • SnapManager Backup completes each task and checks it off on the list shown in the Backup Task List view. • You can switch back and forth between the task check-off list and the progress report using the Switch buttons on either view. • If the verification is successful, the Task window shows the check-off list with the tasks completed. • If Notification is enabled, email is sent and the event is posted to the Windows Application event log.
To schedule the database verification for later:	
11	If you want to schedule the database verification for later, do the following: **1.** Select the Later option. **2.** In the Job Name box, enter a name for this job. **3.** If you want this job name to overwrite a job of the same name (if it exists), select the Replace Job if Exists option. **4.** Click Next. For more information, see *Scheduling a backup job or a database verification job* on page 145. **Result** The Select the Scheduling Service screen appears.
12	In the Select the Scheduling Service screen, do the following: **1.** Select either SQL Server Agent or Windows Scheduled Tasks. For more information, see *Scheduling a backup job or a database verification job* on page 145. **2.** Click Next. **Result** The Completing the Backup Wizard screen appears and displays the backup settings you have selected.

Step	Action
13	After you verify that all the settings in the window are correct, go to the Completing the Backup Wizard dialog box and click Finish. **Result** The selected scheduler displays one of two dialog boxes, as follows: • If you chose to schedule the backup job using the SQL Server Agent, the Properties dialog box appears. • If you chose to schedule the backup job using Scheduled Tasks, the Schedule Job dialog box appears.
14	To schedule the job and close the Backup wizard, do the following: **1.** Specify the details of the job. **2.** Click OK.

Scheduling a backup job or a database verification job

About this section

When you specify a SnapManager backup or SnapManager verification operation, you can start the operation immediately, or you can schedule the operation to run later. This topic describes how to schedule a backup or verification job for a later time.

Related topics

- *Backing up, replicating, and archiving databases using SnapManager* on page 115
- *Managing transaction log backups using SnapManager* on page 126
- *Performing database verification using SnapManager* on page 138

Choosing a schedule service

You can use either of the following schedule services to schedule a backup or database verification:

- SQL Server Agent
- Windows Scheduled Task Wizard

Note: Some limitations apply to using SQL Server authentication as the security authentication method to schedule the job. For more information, see *About SQL Server authentication* in *Connecting to an SQL Server instance* on page 292.

Backup newly created databases without rescheduling existing backup job

If you want to be able to create backup on newly created databases without rescheduling an existing backup job, you must select all the databases on the server when the backup job is scheduled. The `new-backup` cmdlet used for the scheduled job will not specify any databases, which indicates the

cmdlet will try to backup all the databases, including the newly created one on the server. You will not need to re-run the configuration wizard.

Note: It is not recommend to change the SnapManager database configuration when a new database is added to the server. For example, if you have only one database in a LUN or VMDK and you add the new database to the same LUN or VMDK, the SnapManager configuration for the original database will change from non-share configuration to share configuration. The old backups for the original database might be deleted, and an up-to-minute restore for old backup might not be possible.

Using the Schedule Job dialog box

Use the Schedule Job dialog box to schedule a backup operation or a database verification operation. The Schedule Job dialog box opens automatically when you finish specifying the backup or verification operation if you have chosen the option to schedule the operation to be run later.

To use the Schedule Job dialog box, complete the following steps from either Backup and Verify or from the Backup wizard after you have specified the details of the backup or verification operation.

Step	Action	
1	Click "Schedule..."	
2	If the Run Commands dialog box appears, specify the command and then Click OK to close the dialog box. For more information, see *Pre-command and post-command script settings* on page 303. **Result** The Schedule Job dialog box appears.	
3	In the Schedule Job Name box, enter a name for your backup job.	
4	Specify what you want to do if a scheduled job of the same name already exists.	
	If you want to...	Then...
	Overwrite the existing job with this one	Select the Replace Job if Exists option.
	Be prompted to specify a different name	Do not select the Replace Job if Exists option.
5	In the "Select the Scheduling Service to Create Job" panel, select the schedule service you want to use. **Note:** If you select SQL Server Agent and the service is stopped, SnapManager will automatically start the SQL Server Agent service for you.	
Using the SQL Server Agent		

Step	Action	
6	In the Server Name box, specify the name of the SQL Server instance that you want to use to run this job.	
	If...	Then...
	You know the server name	Click the Server Name box and enter the server host name.
	You prefer to browse to the server name	Click Browse to use a browse dialog box to select the server host name.
7	Click OK. **Result** The Properties dialog box appears for the job you are specifying. This is an SQL Server Agent dialog box.	
8	In the Properties dialog box, specify the parameters of your job schedule: • When the job is to run • If you want the job to repeat, at what frequency	
9	Click OK to close the Properties dialog box. **Result** The backup job will run at the times you specified in the Properties dialog box. The backup scheduling process is complete.	

Integrity verification on SnapMirror destination volumes and SnapVault secondary volumes

Choosing the volumes

SnapManager enables you to verify the SQL Server databases stored on destination volumes: SnapMirror destination volumes and SnapVault secondary volumes (clustered Data ONTAP only). When verifying the integrity of databases on a destination volume, SnapManager automatically detects the existing relationships in the SQL Server volumes and selects the available relationship for the selected volume.

The "Choose SnapMirror Destination Volumes for Integrity Verification" window shows the relationship between the source and the destination volumes. Each volume is displayed as a tree showing the relationship between the storage and the databases. For each source volume, there is a list of destination volumes. You can select a destination volume for each source volume for which you want to verify integrity.

To select the destination volume, complete the following steps.

Step	Action
1	Click Backup Verification Settings in the Actions pane. **Result:** The Verification Settings dialog box appears.
2	Click the SnapMirror and SnapVault Options tab.
3	Click the "Verification on Destination volumes" button. **Result** SnapManager displays the Choose SnapMirror Destination Volumes for Integrity Verification window. If the volume is not available, SnapManager displays an appropriate error message.
4	Select the destination volumes. By default, SnapManager displays the "Number of Relationships" field. You cannot edit this value. If the SnapMirror destination volume is not in the SnapMirrored state or does not have FlexClone license installed, SnapManager displays an error message when you click Apply. Integrity verification completes for the source verification volume but fails for the destination volume.
5	Click Apply to save the changes. **Result:** SnapManager saves the settings and makes it available whenever you launch the SnapManager application.
6	Click OK.

Understanding the requirements to run integrity verification for SnapMirror destination volumes

To run integrity verification on SnapMirror destination volumes, ensure that the following system configuration requirements are met:

- A SnapMirror license is enabled on the source volume and a FlexClone license is enabled on the destination volume. SnapManager uses SnapDrive to verify that the required licenses are enabled on the source and destination storage system.
- To run integrity verification on the destination volume, the destination volume must have CIFS shares configured, to be accessible by SnapDrive.
- SnapDrive provides access to the SQL Server databases that are stored on the destination volume, and SnapManager performs the integrity verification on backups of those databases.
- Data files and log files should be present on the SnapMirror destination volume.

Note: SnapManager fails the mount operation on a Snapshot copy in the destination volume if the FlexClone license is not enabled on the SnapMirror destination volumes.

Understanding different types of integrity verification for SnapMirror destination volumes

You can run integrity verification on the SnapMirror destination volumes for different SnapManager operations:

- Full database backup verification
- Deferred integrity verification
- Mount Snapshot and Attach Database
- Restore
- Remote verification of full database backup

Full database backup verification When you run integrity verification on the SnapMirror destination volume, SnapManager performs the following operations:

- Creates a Snapshot copy of the database volumes
- Requests SnapMirror update to replicate the data across destination volumes through SnapDrive
- When the SnapMirror update replicates the backup to the selected destination, SnapManager continuously monitors the SnapMirror update activity through SnapDrive, as follows:
  - SnapDrive provides the SnapMirror update progress information continuously to SnapManager during the update.
  - SnapManager logs the SnapMirror update activity to the backup report at every defined interval.

  Note: If the SnapMirror update operation does not have any progress within a defined interval, SnapManager aborts monitoring it and leaves the backup unverified.

- Creates a Snapshot copy for the SnapInfo volume
- Mounts the available database in the Snapshot copy on the destination volumes.
- Verifies the integrity of databases and transaction logs in the selected destination volumes
- Dismounts the database in the Snapshot copy on the destination volumes.
- Updates database integrity verification result to the live backup SnapInfo.
- Request SnapMirror update to replicate the database verification result to the destination volumes.

Deferred integrity verification For deferred integrity verification (that is, verification at some stage after the backup has been created), SnapManager verifies the SnapMirror state of the destination storage system volume and the existence of a backup Snapshot copy on it. If the Snapshot copy does not exist, SnapManager displays an error message.

When you run a deferred integrity verification on available SnapMirror destination volumes, SnapManager performs the following actions:

If the "SnapMirror update after operation" option is not selected, SnapManager updates the verification results only on the source SnapInfo volumes.

If the "SnapMirror update after operation" option is selected, the following actions occur:

- SnapManager verifies the backup on the selected destination.
- SnapManager updates the verification results to the source SnapInfo volumes.
- SnapMirror replicates the verification results on the source SnapInfo volumes to all the destination SnapInfo volumes.

If the backup is not available on the destination volume, SnapManager fails the mount operation for integrity verification and leaves the backup unverified. In this case, SnapManager does not request the SnapMirror update.

Mount Snapshot and Attach Database When you run an integrity verification for the Mount Snapshot and Attach Database operation, SnapManager performs the following tasks:

- If "Run DBCC CHECKDEB" option is selected, it performs integrity verification on the available selected destination volumes. If "Leave database attached after DBCC" option is selected, the database from the Snapshot copy in the FlexClone destination volumes remains online and operational.
- Displays an appropriate error message, if a backup is not available on the destination volume.
- Updates the verification results to the source SnapInfo volumes

Note: SnapManager does not verify the backup on the source volume, if an unverified SnapManager backup is not available on the destination volume. If you try to run integrity verification in such a case, SnapManager displays an error message.

Restore When you run an integrity verification for a restore operation on the destination volume, SnapManager performs the following actions:

- Performs integrity verification on the destination SnapMirror volumes if a backup is available on the destination volume
- Displays an appropriate error message, if a backup is not available on the destination volume
- Updates the verification results to the source Snapinfo volumes
- Logs the additional steps to the Windows Application event log and to the SnapManager restore report

Remote verification When you run an integrity verification on the destination volume present on the remote destination, SnapManager performs the following actions:

- Creates a backup for the selected database
- Requests SnapMirror Update to replicate the new data to the destination volumes
- Creates a backup for the SnapInfo volume
- Requests SnapMirror Update to replicate the new SnapInfo data to the Destination volumes
- Mounts the database in the SnapShot copy on the destination volumes
- Performs database integrity verification
- Dismounts the database in the SnapShot copy on the destination volumes
- Updates database integrity verification result to the live backup SnapInfo
- Requests SnapMirror Update on the SnapInfo volume to replicate the database verification result to the destination volume

Configuring or changing verification settings

You want to use verified copies when, for example, you restore or clone a database. You can set verification settings on a per-server basis, and if you have some databases in failover clusters and others not in failover clusters, you need to be able to apply different settings for different servers.

About this task

You can configure or change verification settings either from the **SnapManager Configuration Wizard** or from **Backup Verification Settings** action. The options are the same through either method. The **Backup Verification Settings** action is shown here.

Steps

1. From the management console, select the standalone server hosting the databases you want to configure or change their verification settings. For example, **Console Root > SnapManager for SQL Server > SMSQL Server 1**
2. Select **Backup Verification Settings** in the **Actions** window.

   The Verification Settings dialog box opens, displaying the **Verification Settings** tab.
3. Select verification server from the drop-down list.
4. Select the authentication connection type.
5. Select how the drive letter is assigned to the LUN, and set the default mount point directory, if choosing an NTFS directory

   These logs are used to build a log chain during a restore.
6. Save the settings.
7. Click **SnapMirror and SnapVault Options**.
8. Choose the destination volumes used for verification.
9. Click **DBCC Options**
10. Set the desired Microsoft Database Consistency Checker options.

    The Microsoft Database Consistency Checker performs the actual database verification.
11. Decide whether to leave the verified database attached after verification completes.

    If the database remains attached, the associated Snapshot copy LUNs remain mounted, and so you must explicitly unmount the LUNs or the Snapshot copy LUNs will be busy during backup operations.
12. Click **OK**.

Using backup management groups in backup and verification

When you create a full database backup, you can assign it to a backup management group. The backup set names and SnapInfo directory backup set names reflect the management group to which you assigned the backup. The purpose of backup management groups is to enable you to designate various levels of backup retention.

How a backup is assigned a backup management group

When you create a backup, you can assign it to any one of the following SnapManager backup management groups:

- Standard
- Daily
- Weekly

When you start or schedule a full database backup, the Backup wizard and the Backup and Verify option populates the backup management group selection field with the Standard group.

For more information about starting or scheduling a full database backup, see *Backing up, replicating, and archiving databases using SnapManager* on page 115.

Note: The type of backup management group combined with the backup set naming convention selected *(unique* or *generic)* affects the name assigned to the backup set. The name of each backup set created during a SnapManager backup operation includes information that identifies the backup set contents. This is described in “SnapManager backup set naming conventions” in *How SnapManager backup data is organized* on page 99.

How backup management groups are used

The primary purpose of backup management group is to facilitate a database backup retention strategy. Backup management groups are used to determine which backups are targeted for automatic deletion of older backups, database verification for unverified backup Snapshot copies, and explicit deletion of backups.

Note: The backup management group neither depends on nor enforces how often backups are performed. Backup management groups are only backup labeling conventions that determine the backup set retention policies.

Options for a full database backup: When you run or schedule a full database backup, you can specify how many of the most recent backups you want to retain. Only backups of the specified backup management group are deleted. The procedural details are included in *Backing up, replicating, and archiving databases using SnapManager* on page 115.

Options for a database verification: When you run or schedule a database verification separate from the full database backup operation, you can limit the backups you want to verify by specifying a particular backup management group. The procedural details are included in *Performing database verification using SnapManager* on page 138.

Options for an explicit deletion of multiple backup sets: When you explicitly delete multiple backups, you can specify that only backups belonging to a certain backup management group can be deleted. The procedural details are included in “Deleting backups” in *Explicitly deleting backup sets using SnapManager* on page 154.

Backup example using backup management groups: Suppose your company wants to make regular backups between 7:30 a.m. and 7:30 p.m. You want to keep the last backup of the day and retain it for a few weeks, and you want to keep one backup per week for several months for archiving.

To achieve this using backup management groups, you could use the Standard backup management group for the backups during the day, and use a separate backup job to create one backup in the Daily management group at the end of the day. Then, once a week, you could use another job to create a backup in the Weekly backup management group.

You could then decide how many backups to retain independently for each backup management group. For example, you can keep 10 Standard backups, 7 Daily backups (one week's worth), and 4 Weekly backups (one month's worth).

If your Daily or Weekly backup job failed for any reason, you could promote the most recent successful Standard backup to replace the Daily or Weekly backup by changing its backup management group.

Changing the backup management group of an existing backup set

You can use the Change Backup Management Group dialog box to change the backup management group to which the selected backup set belongs.

To change the backup management group of an existing backup set, complete the following steps.

Note: You cannot change the backup management group of the most recent backup sets that were created using the Generic naming convention.

Step	Action
1	In the SnapManager console root, click **Restore**.

Step	Action
2	In the Restore panel, locate the backup set whose management group you want to change: • Database Snapshots (Standard group) ◦ `sqlsnap__sqlservername__`*`date_time`* ◦ `sqlsnap__sqlservername__ recent` • Database Snapshots (Daily or Weekly group) ◦ `sqlsnap__sqlservername__`*`date_time__backupmgmtgroup`* ◦ `sqlsnap__sqlservername__`*`backupmgmtgroup__`*`recent`
3	Right-click the name of the backup set to open a context menu and then select **Change Management Group**.
4	Carefully review the backups listed in the Backups sharing this Snapshot list. The backup management group for all these backups is changed if you complete this operation. This is because they share a common backup set.
5	In the New Management Group list, select the backup management group you want to change to. When you change a backup's backup management group, you also change that backup's name, because the name includes the backup management group.
6	Click **OK**. The backup management group for this backup and all backups listed in the All Backups Sharing This Snapshots list is changed. The report for the backup management group change is in the Miscellaneous report directory.

Explicitly deleting backup sets using SnapManager

You can automatically delete older backup sets by specifying the "Delete full backups in excess of" option and the "Delete full backups older than" option in the SnapManager backup facility. This is the recommended method for managing the number of backup sets stored. For more information, see "Automatic deletion of the oldest backups in a management group" in *Ways to manage the number of backup sets kept online* on page 109.

You can also explicitly select the backup sets that you want to delete.

Understanding explicit deletion of backup sets

SnapManager provides three ways for you to explicitly delete backup sets.

If you want to delete...	Then use this method...
A specific full database backup set	In the "Delete backups" dialog box, select the databases, the database component types, and the backup management group (Standard, Daily, Weekly, or All). You can also use the Restore option to delete backup sets.
A specific transaction log backup	In the "Delete backups" dialog box, select the "Log Snapshots only" option.
Snapshot copies of LUNs, SMB shares, and VMDKs created during restore	In the "Delete backups" dialog box, select the "Delete snapshot of LUNs created during restore" option.

Each of the explicit deletion methods enables you to view detailed information about your selection before you proceed with the operation.

Deleting backups

You can delete backups for a specified group of databases by choosing which backup sets you want to delete and whether you want to also delete backup sets created during the restore (if applicable).

Information you need to specify to delete backups An explicit deletion of backups is specified using the following parameters.

- The backup sets you want to delete
- The databases for which you want to delete backups
- The backup set components you want to delete: complete data sets, transaction log backups only, or SnapInfo Snapshot copies only
- The backup management group for which you want to delete backups: Standard, Daily, Weekly, or All
- The number of backups to delete: all the backups in the specified management group or only the oldest backups, retaining only the number of backups specified
- The number of days such that the backups older than the given number of days are deleted.
- Whether you want to also delete backup sets created during the restore

Procedure To delete backups, complete the following steps.

To use the Up-to-the-minute Restore pane to delete backups, see "Configuring the number of transaction log backups your system retains" in *Managing transaction log backups using SnapManager* on page 126.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>In the SnapManager console root, select "Delete Backup" from the Actions pane.
Result The "Delete backups" dialog box appears and displays information about the selected backup set, including all backed-up databases contained in the backup set.</td></tr>
<tr><td>2</td><td>The "Backup component" option narrows the scope of the deletion by specifying the type of backup components to be deleted.
This option is set to Backup Data Sets by default, but you can narrow this selection to transaction log backups only or to SnapInfo Snapshot copies only.</td></tr>
<tr><td>3</td><td>The Management Group option further narrows the scope of the deletion by specifying the backup management group to be deleted.
This option is set to Standard by default, but you can change it to Daily, Weekly, or All.</td></tr>
<tr><td>4</td><td>By default, only backups containing all the selected databases are deleted. You can override this behavior for this particular backup deletion operation only by using the Advanced button.
<table>
<tr><th>If you want to...</th><th>Then...</th></tr>
<tr><td>Delete only backups containing all the selected databases</td><td>Go to Step 7.</td></tr>
<tr><td>Delete backups containing any one or more of the selected databases</td><td>Go to Step 5.</td></tr>
</table></td></tr>
<tr><td>5</td><td>Click Advanced.
Result The Advanced Options dialog box appears.</td></tr>
<tr><td>6</td><td>In the Delete Backups pane, select the given option and click OK to apply your change and close the dialog box.
Note: For this backup deletion operation only, multiple backup deletions delete backups containing any one or more of the selected databases.</td></tr>
</table>

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td rowspan="5">7</td><td colspan="2">Specify which backup sets you want to delete.</td></tr>
<tr><td>If you want to...</td><td>Then...</td></tr>
<tr><td>Delete the oldest backups</td><td>1. Select the "Delete oldest backup in excess of" option.
2. Specify how many of the newest backups you want to preserve.</td></tr>
<tr><td>Delete backups older than a specified number of days</td><td>1. Select the "Delete backups older than" option.
2. Specify how many days of backup you want to preserve.</td></tr>
<tr><td>Delete all the restore Snapshot copies</td><td>Select the "Delete all backups in the specified management group" option.</td></tr>
<tr><td rowspan="4">8</td><td colspan="2">You can delete the selected restore Snapshot copies immediately, or you can first view the list of restore Snapshot copies that are targeted for deletion.</td></tr>
<tr><td>If you want to...</td><td>Then...</td></tr>
<tr><td>View the list of restore Snapshot copies that would be deleted</td><td>Go to Step 9.</td></tr>
<tr><td>Delete the restore Snapshot copies</td><td>Click Delete.
Result The restore Snapshot copies identified by your selections are deleted. When the deletion is complete, a status message is displayed. You have completed this procedure.</td></tr>
<tr><td>9</td><td colspan="2">Click Delete Preview.
Result The Delete backups dialog box appears. After a moment, the dialog box displays a count and list of the backups identified for deletion.
If you want to view a report, click Show Report.</td></tr>
<tr><td rowspan="4">10</td><td colspan="2">Based on the list displayed in the Delete backups dialog box, you can cancel the delete operation or proceed with the delete operation.</td></tr>
<tr><td>If you want to...</td><td>Then...</td></tr>
<tr><td>Cancel the operation</td><td>Click Close to close the Delete backups dialog box.</td></tr>
<tr><td>Delete the backups listed in the preview</td><td>Click Delete on the Delete backups dialog box.</td></tr>
</table>

Busy Snapshot error prevents deletion of backup set

If the FlexClone license is not enabled and you have a backup of a LUN that is backed by another backup set, you get an error stating that the backup set is busy and cannot be deleted.

Definition of a busy backup set A backup set is busy if there are any LUNs backed by data in that backup set. The backup set contains data that is used by the LUN. These LUNs can exist either in the active file system or in some other backup set. For more information about how a backup set becomes busy, see the *Data ONTAP SAN Administration Guide for 7-Mode* for your version of Data ONTAP.

If you attempt to delete a busy backup set If you begin a backup when a LUN backed by a backup set exists, the result is a backup set that cannot be deleted; if you do attempt to delete the backup set, the following events occur:

- SnapManager displays a busy backup set error message.
- SnapDrive logs event 249 in the Windows application event log.

To check whether you have a busy backup set There are two ways to determine whether you have a busy backup set:

- View your Snapshot copies in FilerView.
- Use the following storage system command to list the busy Snapshot copies:
  `snap list usage` *VolumeName BusySnapshotName*
- The full description of the preceding command syntax is described in the *Command Reference* for your version of Data ONTAP.

To delete a busy backup set Delete the more recently taken backup; then delete the older backup. For more information about deleting a busy backup set, see the *Data ONTAP SAN Administration Guide for 7-Mode* for your version of Data ONTAP.

To avoid this situation in the future Avoid performing SnapManager backups while you have any LUNs backed by Snapshot copies.

- During a *database verification*, a LUN in a backup set is mounted and the DBCC utility is run against the database. For this reason, it is important to carefully plan your SnapManager backup and verification schedules. See "Recommendations for scheduling backups" in *When to run a SnapManager backup* on page 111.
- While *archiving from a LUN backed by a backup set*, avoid performing a SnapManager backup.

Automatically delete backup sets

SnapManager can be used to automatically delete backup sets as part of a backup. It can also be used to delete backup sets outside the backup process. SnapManager works with SnapDrive to prevent any accidental deletion of Snapshot copies that are required to keep up to date.

Note: If Snapshot copies are directly deleted from the storage system without using SnapManager or SnapDrive, do not delete Snapshot copies needed during SnapVault update. When using

SnapVault to archive backup sets in SnapManager for SQL Server, at least two of the most recent snapshots that were used for the SnapVault updates should be kept online in SnapManager.

Example Assume that four Snapshot copies are created every day where the first and last Snapshot copies are used for SnapVault updates and the two Snapshot copies in the middle are not used for the updates. When using SnapManager to automatically delete Snapshot copies based on quantity, at least four Snapshot copies would need to be left online. The two Snapshot copies taken in the middle of the day can be deleted individually and manually under the SnapManager restore option by right-clicking the backup set name and selecting Delete.

Deleting archived backups

Deleting archived backups

The process of deleting archived backups is the same as deleting local backups. Be sure to read the following points before deleting archived backups:

- When a local backup is deleted, SnapManager does not delete the backup metadata and SnapInfo file in the SnapInfo directory but deletes the transaction logs in the SnapInfo directory.
- If SnapManager is unable to find both the local and archived backups, it deletes the backup metadata and the SnapInfo directory associated with the backup.

Note: Make sure that the NetApp Management Console is always available, otherwise SnapManager deletes the backup metadata for the archived backups.

Restoring databases using SnapManager

Understanding SnapManager restore

SnapManager Restore restores the SQL Server databases you select.

Related topics

- *Types of SnapManager restore operations* on page 163
- *Choosing the type of restore operation to perform* on page 166

Sources for a restore operation

SnapManager Restore enables you to restore databases from a SnapManager backup set.

Restore from a SnapManager backup set: You can restore databases from SnapManager backup sets created for the same SQL Server instance or created for a different server instance. The LUNs, SMB shares, or VMDKs containing the selected SQL Server's databases are restored from the backup.

Note: You can perform restore operations only from hosts, not from the storage system.

If SQL Server system databases fail, they can be restored from stream-based SnapManager backup sets of those databases. For more information, see “Preparing to restore operation from a SnapManager backup set” in *Performing a restore operation* on page 167.

Restore from unmanaged media: You can also use SnapManager Restore to restore databases from offline archives (unmanaged media) of backup sets.

For information about recovering SQL Server databases from archives, see *Recovering SQL Server databases using archives* on page 210.

Restore a database residing on multiple LUNs, SMB shares, or VMDKs: You can restore databases that reside on multiple LUNs, SMB shares, or VMDKs. The restore operation takes some time to complete, because SnapManager takes one at a time, serially, for the complete database restore operation.

Destinations for a restore operation

You can restore databases to various types of destinations.

Restore to the original location: By default, SnapManager restores to a database to the same location on the same SQL Server instance.

Restore to different database names: You can restore to a different server instance on the same or different server using different database names.

Restore to other location: You can restore a database to a different location on the same SQL Server instance.

Clone to an alternate location: You can use SnapManager Restore to restore an online database as a new database on the same SQL Server instance. However, you *cannot* restore an online database as a new database on a different SQL Server instance. You must clone the database.

Mount at a temporary, alternate location without restoring: The database is mounted at a temporary alternate location, but the transaction logs are not applied. Because the data is not current, you should use this function to view only the layout of the data.

Cloned database in a backup set

The following information applies to databases that have been *cloned* or only *mounted at* temporary, alternate locations using a writable Snapshot copy.

Backup set label: A database cloned or mounted at a temporary alternate location is listed in the Backup and Verify option with the following label:

`SnapLUN`

Backup set name: If the database name already exists on the server, or if the backup set consists of more than one database, the database is listed using the following naming convention:

databasename`__Clone`

Avoiding a busy Snapshot condition: Explicitly detach the temporary database and dismount the backup set as soon as you finish viewing the data or data layout. Otherwise, you might encounter a *busy Snapshot* condition when you attempt to delete the backup set.

> **Note:** If the storage system has a FlexClone license installed, then FlexClone is used for verification. In this case, you do not encounter the busy Snapshot condition.

If the database was *restored* with a post-restore state of *Read-Only* or is in the *loading* state and you cannot bring the database into read-write mode, use SQL Server Management Studio.

How SnapManager Restore works

If the "Create transaction log backup before restore" option is selected, the transaction log is backed up before the restore is performed.

If you are cloning the database using a writable Snapshot copy, a transaction log backup is *not* created before the restore, even if this option is selected. If you want to create a transaction log backup, do so as a separate operation before you restore to the alternate location.

For reasons to clear this option, see "Understanding the restore options" in *SnapManager restore options* on page 301.

SnapManager restores the databases that you select to the active file system. The restore method used by SnapManager depends on (1) the method that was used to create the backup set and (2) the specific subset of databases you choose to restore from the backup set.

SnapManager uses the *stream-based* restore method if you are restoring from a stream-based backup set. With this method, each of the databases is restored individually. Depending on the composition of the backup set, a stream-based restore can require additional time and free space on the storage system as compared to an online Snapshot copy restore.

SnapManager uses LUN, SMB share, and VMDK cloning if you are restoring from a backup set that contains multiple databases that reside on the same LUN, SMB share, or VMDK.

SnapManager uses the *copy-based* restore method if any of the following conditions are true:

- The backup set contains only a subset of the databases that reside on the same LUN, SMB share, or VMDK (not recommended).
- You select only a subset of the databases contained in the backup set.
- A new database was added to the same LUN, SMB share, or VMDK after the backup was created.

In a volume-wide backup, all the databases that reside on a single volume are backed up concurrently using Snapshot copies. Because the maximum number of databases supported per storage system volume is 35, the total number of Snapshot copies created equals the number of databases / 35.

If the database has transaction log backups, SnapManager Restore can apply the transaction log backups (if necessary).

Depending on the database restore option selected, SnapManager Restore performs a *point-in-time* restore or an *up-to-the-minute* restore.

Restore Snapshot copies Every time you perform a restore operation using SnapManager, SnapManager first creates a Snapshot copy on each storage system volume that contains files for the databases you will be restoring. That way, in the unlikely event that a catastrophic failure occurs during a restore, you have recent Snapshot copies of the LUNs, SMB shares, or VMDKs that can be used to re-create those databases as they existed prior to the start of the failed restore operation.

Each restore Snapshot copy is named using the following naming convention:

`rstsnap__`*`SqlServerName_date_time`*

The Snapshot copy name contains the name of the SQL Server instance to which the backup was restored (indicated by the variable *SqlServerName*) and the Snapshot copy creation date and time (indicated by the variable *date_time*).

After you verify that a restore was completed successfully and you are satisfied with the results, you can delete the restore Snapshot copy.

SQL Server cluster group state during a restore SnapManager can restore databases in a Windows cluster without taking the SQL Server cluster group offline.

Cluster failure during a restore operation If a cluster failure (a cluster group move operation) occurs during the restore operation, for example if the node that owns the resources goes down, you must reconnect to the SQL Server instance and then restart the restore operation.

Transaction log restore operations A SnapManager transaction log restore uses the SQL recovery process to play forward transactions from the log backup into the restored database.

Importance of verifying databases to be restored

The database verification process protects you from restoring a backup that contains any physical-level corruption. Physical-level database corruption can occur silently in SQL Server databases. The only way to know whether a particular database backup incurred physical-level corruption is to run database verification on that backup.

Before allowing a restore operation to proceed, SnapManager enables you to check that the selected backup set was verified through the use of DBCC CHECKDB.

Backup verification status SnapManager Restore shows you a list of the backups that have been taken. For each backup, the date and time of the backup is displayed, as well as an icon that indicates the backup verification status.

Icon description	Backup verification status
Circled check mark	The databases in this backup have been verified.
Circled question mark	The databases in this backup have not been verified.

If you select a database on which a consistency check has not been run successfully, SnapManager prompts (but does not require) you to run DBCC before performing a restore. Running database consistency checking as part of recovery increases the time the recovery takes.

Types of SnapManager restore operations

You can use SnapManager to perform any of the following types of restore operations:

- Up-to-the-minute restore operation
- Point-in-time restore operation
- Marked transaction restore operation

Related topics

- *Understanding SnapManager Restore* on page 160
- *Choosing the type of restore operation to perform* on page 166

Up-to-the-minute restore operation

In an up-to-the-minute restore, databases are recovered up to the point of failure. SnapManager accomplishes this by performing the following sequence:

The last active transaction log is automatically backed up.

The databases are restored from the full database sets you select.

All the transaction logs that were not committed to the databases, including transaction logs from the *backup sets*, from the time the backup set was created up to the most current time, are played forward and applied to the databases (if selected).

An up-to-the-minute restore requires a *contiguous set of transaction logs*. The up-to-the-minute restore type is selected by default. For more information, see *Choosing the type of restore operation to perform* on page 166.

Because SnapManager cannot restore transaction logs from *log-shipping* backup files, you might not be able to restore the database using an up-to-the-minute restore. For this reason, you should use SnapManager only to back up your SQL Server database transaction log files.

If you do not need to retain up-to-the-minute restore capability for all backups, you can configure your system's transaction log backup retention through Up-to-minute Restore Options, located in the Backup and Verify window. Details about configuring transaction log backup retention are in "Configuring the number of transaction log backups your system retains" in *Managing transaction log backups using SnapManager* on page 126.

Example You run SnapManager Backup every day at noon, and on Wednesday at 4:00 p.m. you need to restore from a backup. For some reason, the backup set from Wednesday lunch time failed verification, so you decide to restore from the Tuesday lunch time backup. If the After that backup is restored, all the transaction logs are played forward and applied to the restored databases, starting with those that were not committed when you created Tuesday's backup set and continuing through the latest transaction log written on Wednesday at 4:00 p.m. (if the transaction logs were backed up).

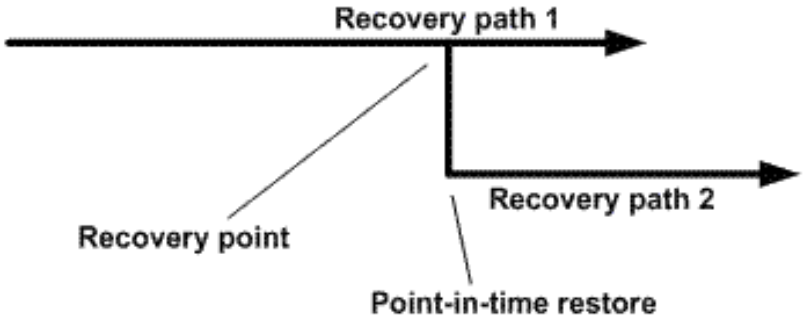
Point-in-time restore operation

In a point-in-time restore, databases are restored only to a point-in-time from the past. A point-in-time restore occurs in two restore scenarios:

- The database is restored to a given time in a backed up transaction log.
- The database is restored and only a subset of backed up transaction logs are applied to it.

Note: When you restore a database to a point in time, it results in a new recovery path.

The following image illustrates the potential problems when a point-in-time restore is performed.



In the image, Recovery path 1 consists of a full backup followed by the number of transaction log backups. The database administrator restores the database to a point in time. New transaction log backups are created after the point-in-time restores which results in Recovery path 2. The new transaction log backups are created without creating a new full backup. Due to data corruption or other problems, if you need to restore the current database, you will not be able to restore it because a new full backup was not created. Also, it is not possible to apply the transaction logs created in Recovery path 2 to the full backup belonging to Recovery path 1.

Note: Ensure that you always create a full backup after restoring a database to a point in time.

If you apply transaction log backup sets, you can also specify a particular date and time at which you want to stop the application of backed up transactions. To do this, you specify a date and time within the available range and SnapManager will roll back any transactions that were not committed prior to that point in time. You can use this method to restore databases back to a point in time before a corruption occurred, or to recover from an accidental database, or table deletion.

Example Suppose you take full database backups once at midnight and a transaction log backup every hour. The database crashes at 9:45 a.m., but you still back up the transaction logs of the failed database. You can choose from among three point-in-time restore scenarios:

- Restore the full database backup taken at midnight and accept the loss of the database changes made afterward.
- Restore the full database backup and apply all the transaction log backups until 9:45 a.m.
- Restore the full database backup and apply transaction log backup sets. Specifying the time you want the transactions to restore from the last set of transaction log backups.

In this case, you would calculate the date and time where a certain error was reported. Any transactions that were not committed prior to the date and time specified in the `Restore` command are rolled back.

Marked transaction restore operation

Restore to marked transaction operations enable you to restore a database to a marked transaction. Using the marks created on a federated full backup, you can restore a backup to a marked transaction across multiple databases for a synchronous restoration.

Note: You can use *either* restore to mark *or* restore to point-in-time. They do not work simultaneously.

These transaction marks are recorded in the transaction log and included in the logs of the affected database.

Choosing the type of restore operation to perform

Use following general guidelines to help you decide whether to use a point-in-time restore or an up-to-the-minute restore.

Related topics

- *Understanding SnapManager Restore* on page 160
- *Types of SnapManager restore operations* on page 163

Capabilities

Often you can choose a restore type based on the particular restore capabilities needed.

If you want to roll forward all the transactions up to the most current time, use an up-to-the-minute restore.

If you want to recover the databases as they were at a particular point in time, for example, at the point when the most recent backup was created, use a point-in-time restore.

Requirements and limitations

Before choosing a restore type, be aware of the requirements and limitations of each.

For to an up-to-the-minute restore to succeed, a contiguous set of all required transaction log backups must be in the SnapInfo folder.

After a point-in-time restore of a backup that is not the most recent one, all existing backups become point-in-time restorable only. Backups created after the point-in-time restore operation will be available for both up-to-the-minute and point-in-time restore operations.

Understanding post restore database recovery states

The following table describes the post restore database states from which you can select.

Database state	Description
Operational	All of the following apply: • No more transaction logs can be restored. • The database is ready to use. This database state is selected by default.
Non-Operational	More transaction logs can be restored.
Read-Only	All the following apply: • More transaction logs can be restored. • The undo file is enabled. If more transaction logs are restored, any changes can be rolled back if the restoration of the transaction log is unsuccessful. **Note:** If you restore a database to a temporary, alternate location using a writable Snapshot copy with this option enabled, the Detach Database and Dismount Snapshot LUN(s) function is unavailable for this database.

Performing a restore operation

You can restore SQL Server databases from a SnapManager backup in two ways: using the SnapManager Restore option or using the SnapManager Restore Wizard. You can restore from local backups or archived backups.

Preparing to restore operation from a SnapManager backup set

Before you restore from a SnapManager backup set, review the following checklist:

- The SQL Server must be online and running before a SnapManager Restore can take place.
  This applies to both user database restore operations and system database restore operations.
- Be sure that the target databases are detached or in a suspect state.
- You can perform a restore of the SQL Server databases with the databases online, but this requires that the online restore option be enabled.
  To view or change this option and other options pertaining to the SnapManager restore operation, go to the SnapManager console root and select **Options > Restore Settings**.
- If you restore multiple databases to the same SQL Server instance, ensure that you do not assign the same target database name for multiple databases.

Follow these guidelines when restoring a SnapManager backup set:

- Although not necessary, you should always restore from the most recent Snapshot copy.
- If you rename a database, make sure that you back it up as soon as possible.

- If you use SnapManager to restore a backup that is not the most recent one, that backup's sequential backup sets are still available for future restore.
- An earlier version of SQL Server cannot restore databases that were created in a later version of SQL Server, but databases created by an earlier version of SQL server can be restored by a later version of SQL Server.
- When multiple versions of SQL Server are installed on the same server, ensure that the latest version of SQL Server is selected in `Add Server to be Managed` option to connect to the specified SQL Server.

Preparing to restore an online database as a new database

You must detach the online database before you begin restoring it as a new database on the *same* SQL Server instance.

Each time you restore a SnapManager backup set, you must specify the following information in either the SnapManager Restore option or in the Restore Wizard:

Backup set from which the databases are to be restored You can select an *unverified* backup set, but SnapManager asks you to confirm your selection. You should restore only from verified backup sets.

If for some reason you do not have a verified backup set available when you need to perform a restore and you do not want to wait for a verification to be completed before you perform the restore, you might find it necessary to restore directly from an unverified backup set.

If you must restore from an unverified backup set, you are strongly recommended to perform an *up-to-the-minute* restore operation. This way, if you discover later that the backup set was corrupted, you can restore the database from a different backup set.

If an SQL Server 2005 database has the full-text search option enabled, the *full-text search catalogs* are visible when you click the "+" next to the database name. The full text catalogs can be migrated, backed up, and restored along with the other files or filegroups of the database.

Databases to be restored from the backup set: All databases in the backup set (the default setting)

A subset of the databases in the backup set To choose only a subset of the databases in the selected backup set, highlight any database in the right pane and then select the **Unselect All Databases and Logs** item from the context menu. This deselects all databases in the backup set. You can then choose the individual databases that you want to restore.

Database target:

- Original database (the default setting)
- A database of a different name

SQL Server instance to which the backup set is to be restored:

- Original SQL Server (the default setting)
- A different SQL Server (only on the same host)

Restore type:

- Up-to-the-minute (the default setting)
- Point-in-time
- Restore to mark

For more information, see *Types of SnapManager restore operations* on page 163 and *Choosing the type of restore operation to perform* on page 166.

You cannot restore multiple databases with different restore options in a single restore operation.

Restore location:

- Original database location (the default setting)
- Other location

If you are restoring a *log-shipped* database, do not restore the transaction logs. Restoring the transaction logs to a log-shipped database causes the SnapManager operation to fail.

How filestreams are supported by SnapManager for SQL Server

Filestreams are fully supported by SnapManager for SQL Server, and filestream objects configured within SQL appear in the SnapManager for SQL Server GUI. If you have a database that includes filestream objects, SnapManager for SQL Server can back up and restore the filestream objects along with the database. For more information about how to enable the Filestream option, see SQL Server online documentation.

Verification settings

The following list summarizes the settings that pertain to database restore operations:

- Which SQL Server is used to perform database verification
- This is configured using the SQL Server option of the Verification Settings dialog box. See "Selecting the database verification server" in *Database integrity verification options* on page 294.
- Which DBCC options are used to verify database backup sets
- This is configured using the DBCC Options option of the Verification Settings dialog box. See "Selecting DBCC options" in *Database integrity verification options* on page 294.
- The Verification Settings dialog box can be accessed from the Restore wizard.

Restore settings

The following restore settings determine how SnapManager is to restore database backup sets:

- Recover databases without restoring at the end of the restore if needed
- Restore databases even if existing databases are online
- Retain SQL database replication settings
- Create transaction log backup before restore

If you are restoring a *log-shipped* database, disable the option to create a transaction log backup before the restore.

- Abort database restore if transaction log backup before restore fails

These settings are configured using the Restore Settings dialog box, described in "Configuring the profile of a restore operation" in *SnapManager restore options* on page 301.

Using the Find Backups Wizard

You can restore backups that were created previously using the Find Backups wizard. Follow these steps to restore backups created previously.

Step	Action
1	Click Restore in the Scope pane.
2	Click Find Backups in the Actions pane. **Result** The SnapManager for SQL Server Find Backups Wizard starts.
3	Follow the steps as instructed in the wizard and click Finish.

Using this wizard, you can restore backups that were created on the same SQL Server, or restore from unmanaged media or restore backups that were created on a different server by selecting the relevant option in the wizard. You need to enter the SnapInfo directory path if you want to restore from unmanaged media or restore backups that were created on a different server.

Restoring using the SnapManager Restore option

To restore an SQL Server database from a backup set using SnapManager Restore, complete the following steps.

Step	Action
1	Review the list in "Preparing to restore operation from a SnapManager backup set" in *Performing a restore operation* on page 167.
2	Make sure that all Windows Explorer instances are closed on the SQL Server computer that is running SnapManager.
3	Disable any SnapManager operations that are scheduled to run against the SQL Server data you are restoring, including any jobs scheduled on remote management or remote verification servers.
4	In the SnapManager console root, click the SQL Server you want to perform the restore operation on. **Note:** You can only restore from hosts, not from storage systems.
5	In the Scope pane, click **Restore**. You can now navigate the backup sets.
Specify the source	

Step	Action
6	If you want to restore databases to this SQL Server using SnapManager backup sets that were created for other SQL Servers, follow the procedure described in "Restoring from other SQL Server backups using SnapManager Restore" in *Restoring databases from other SQL Server backups* on page 211.
7	In the Restore panel, double-click to select the backup set from which you want to restore.
8	In the Actions panel, click **Restore**. The SnapManager for SQL Server-Restore window appears. **Note:** When SnapManager takes transaction logs on multiple nodes of a Windows Server Failover Cluster, the Restore window displays multiple transaction logs that have the same timestamp, but different indexes. For example: "04-25-2014_05.57.08 (1)" and "04-25-2014_05.57.08 (2)"
9	If you want to restore as a database with a different name than the original database, follow these steps: **1.** Click the tab marked **...** next to **Restore as Database**. **2.** The "Individual Database Restore As..." dialog box appears. **3.** In the Restore as Database box, enter the database name to which you want the backup restored. This database name must not already exist on the SQL Server instance to which you will be restoring the database. **4.** Click **OK** to apply your change and close the dialog box.
10	Click the tab marked "..." next to "Restore to Server (instance)".
11	Select or enter the server name that you want the database to be restored to.
12	Choose the Connection by selecting the **Use Windows Authentication** or **Use SQL Server Authentication** radio button.
13	Click **OK** to apply your change and close the dialog box.
Specify the restore type	

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td rowspan="3">14</td><td>If...</td><td>Then...</td></tr>
<tr><td>You want to restore to a point-in-time backup</td><td>1. Click the tab marked "..." next to "Point-in-Time Restore."
The Point-in-Time dialog box opens.
2. In the Point-in-Time Restore dialog box, specify the date and time after which transaction logs are not applied to the restored database.
3. Click OK to apply your change and close the dialog box.
Note: A point-in-time restore halts the restoration of transaction log entries that were recorded after the specified date and time.</td></tr>
<tr><td>You want to restore to a marked transaction</td><td>1. Click the tab marked "..." next to "Marked Transaction."
The Marked Transaction dialog box opens.
2. In the Marked Transaction dialog box, select which marked transaction at which to stop the restore operation.
3. Click OK.</td></tr>
<tr><td>15</td><td colspan="2">If you want to run a command or script prior to performing the restore operation or after the restore operation finishes, select the Run Command Settings option.
Result If you select this option, SnapManager displays the Run Command dialog box. For more information, see *Pre-command and post-command script settings* on page 303.</td></tr>
</table>

Step	Action
16	If you want to restore the database to a different location, do the following: **1.** Click the **...** tab next to **Restore to Other Location**. **2.** To edit the location, select and modify the **Restore To** field for each row or select the **...** tab and browse for the location. Note the following requirements for the location: • If you restore a database to a different path and that path is an SMB share, the SMB share must be accessible from SnapDrive. • If you chose to restore from unmanaged media, enter the location of the mounted disk where the database files are available. • If you chose to restore from unmanaged media, ensure that you place the database files in the restoring location. • You cannot spread a database's files across SAN and NAS environments. **Note:** If the alternate location does not have enough space, the restore will fail. If this happens, delete the partially copied database files.
17	To start the restore operation, click Restore. **Result** SnapManager begins to restore your databases from the backup you selected. SnapManager Restore completes each task and checks it off the list shown in the Restore Task List view. You can switch back and forth between the task check-off list and the progress report using the Switch buttons on either window. If the restore is successful, the Task window shows the check-off list with the tasks completed, and a dialog box reports that the restore was successful. **Note:** If Notification is enabled, email is sent and the event is posted to the Windows event log.
18	After all the restore tasks are finished, click OK. **Result** Your restore is complete and your SQL Server computer comes back online.
19	After the restore is complete, you should perform a full backup and verification to verify that your restored database is free of physical-level corruption. This step is especially important if you restored a database to a different path that is shared by existing databases.

Other restore options in the Actions pane

You can change the management group of the database to be restored using the option "Change Management Group." You can also mount Snapshot copies, run the DBCC functionality, and attach a copy of databases to Snapshot copies using the option "Mount Attach Db..."

Restoring using the SnapManager Restore Wizard

To restore an SQL Server database from a backup set using the SnapManager Restore Wizard, complete the following steps.

Step	**Action**	
1	Review the list in "Preparing to restore operation from a SnapManager backup set" in *Performing a restore operation* on page 167.	
2	Make sure that all Windows Explorer instances are closed on the SQL Server computer running SnapManager.	
3	Disable any SnapManager operations that are scheduled to run against the SQL Server data you are restoring, including any jobs scheduled on remote management or remote verification servers.	
4	To launch the SnapManager Restore Wizard, select the server you want to restore to in the Scope pane.	
5	Select "Restore Wizard" from the Actions pane. **Result** The Restore Wizard appears and displays the Welcome screen.	
Start		
6	Click Next. **Result** The SnapManager for SQL Server Restore screen appears.	
SQL Server		
7	By default, SnapManager restores from backups that were created on the same server that you run the Restore wizard on.	
	If...	Then...
	You want to restore from backups that were created on the same SQL Server	Select "Restore SnapManager backups that were created on the same SQL Server". The "Backup Set" screen appears. See Step 8.
	You want to restore from backups that were created on a different SQL Server	Select "Restore backup created on a different server". Follow the procedure described in "Restoring from other SQL Server backups using the SnapManager Restore wizard" in *Restoring databases from other SQL Server backups* on page 211.
	You want to restore from an unmanaged media	Select "Restore from Unmanaged Media".
Backup Set		

Step	Action
8	Double-click to select the backup under the database you want to restore. Click Next.
9	Follow the instructions in the Restore wizard as you proceed.
Restore Database As	
10	If you want to restore the database to a different location, do the following: **1.** Click the **...** tab next to **Restore to Other Location**. **2.** To edit the location, select and modify the **Restore To** field for each row or select the **...** tab and browse for the location. Note the following requirements for the location: • If you restore a database to a different path and that path is an SMB share, the SMB share must be accessible from SnapDrive. • If you chose to restore from unmanaged media, enter the location of the mounted disk where the database files are available. • You cannot spread a database's files across SAN and NAS environments. **Note:** If the alternate location does not have enough space, the restore fails. If this happens, delete the partially copied database files.
Completing the Restore Wizard	
11	After you verify that all the settings in the screen are correct, click Finish. **Result** The Restore wizard closes and the Restore Status dialog box appears and displays the Restore Task List, which is used to show the progress of the restore operation after you start it.
Restore Status	
12	To start the restore operation, click Start Now. **Result** SnapManager begins to restore your databases from the backup you selected. SnapManager Restore completes each task and checks it off on the list shown in the Restore Task List view. You can switch back and forth between the task check-list and the progress report using the Switch buttons on either window. If the restore is successful, the Task window shows the check-list with the tasks completed, and a dialog box reports that the restore was successful. **Note:** If Notification is enabled, email is sent to the specified address. All events are posted to the Windows event log, even if notification is not enabled.

Step	Action
13	After the restore is complete, click OK to close the dialog box. **Result** Your restore is now complete and your SQL Server computer comes back online.
14	After the restore is complete, you should perform a full backup and verification to verify that your restored database is free of physical-level corruption. This step is especially important if you restored a database to a different path that is shared by existing databases.

Retrieving and restoring remote backups

Retrieving remote backups

To retrieve a remote backup, SnapManager uses the name of the dataset and the SnapInfo directory to create a list of databases that need to be restored.

Restoring remote backups

The process of restoring a remote backup is almost the same as restoring a local backup, except that the remote backup needs to be restored from the archived backup and the backed up transaction logs have to be applied manually.

To restore an archived database from the Restore wizard, follow these steps.

Step	Action
1	Select the SQL Server from the Scope pane.
2	Select Restore Wizard from the Actions pane. **Result** The Restore wizard opens.
3	Click Next. **Result** The "Which SQL Server Created the Backups" window appears.
4	In the "Which SQL Server Created the Backups" window, select the option "Restore from archive."
5	Continue with the instructions provided in the Restore wizard.
6	Click Finish to restore the database from the archived backups.

Deferred database integrity verification

Deferred integrity verification can be performed on the local and the remote backup. Deferred integrity verification can be carried out in two ways:

- Deferred integrity verification at the local location
- SnapManager archives the local backup first, then verifies the backup on the SnapVault secondary storage system.
- Both "Archive backup to secondary storage" and "Verify on archived backup on secondary storage" should be enabled.
- Both local and remote backup management groups are used. The local backups of the local backup management group are archived using the remote backup management group.
- Deferred integrity verification at the archived location (secondary storage system)
- SnapManager runs verification on the backups already archived on the SnapVault secondary storage.
- Only "Verify on archived backup on secondary storage" should be enabled, and only the backups of the remote backup management group are verified.

Deferred integrity verification runs on the management group that you selected previously. The remote backup management group can always be changed after remote backup is created.

You can perform deferred verification on the SnapVault secondary storage system from both the local application server and the remote verification server.

Types of restore operations supported with dataset and SnapVault integration

The following table describes the restore operations supported with dataset and SnapVault integration with SnapManager.

Operation	Backup type	Restore from archive
Restore system databases	Not supported	Not supported
Restore user databases	Full backup	Automatic
	Transaction log backup	Manual
Backup verification during restore	Full backup	Automatic
	Transaction log backup	Not supported
Restore when the system databases and the user databases share one LUN	Full backup and transaction log backup	Not supported
Clone user databases	Full backup	Automatic
	Transaction log backup	Manual

Deleting restored Snapshot copies

Deleting restored Snapshot copies

To explicitly delete the oldest Snapshot copies created during previous restore operations, complete the following steps.

Step	Action
1	In the SnapManager Actions pane, click Delete Backup. **Result** The Delete backups dialog box is displayed.
2	Select the "Delete Snapshot of LUNs created during restore" option. For more information, see *Explicitly deleting backup sets using SnapManager* on page 154.

Restoring replicated publisher and subscriber databases

Restoring replicated publisher and subscriber databases

If you are restoring replicated publisher and subscriber databases, follow these steps:

Step	Action
1	Perform the backup operation on the distribution and replicated database.
2	Restore the following database strictly in the given order: **1.** Distribution database **2.** Publisher database **3.** Subscriber database **Note:** If you do not restore the distribution database first, the replication settings are not maintained and you will have to restart the replication.
3	While restoring replicated databases, stop the running SQL Agent.
4	Take the publisher and subscriber database offline.
5	In the Action pane, click Restore Settings> Restore SQL database replication settings.
6	Select the options "Retain SQL database replication settings" and "Restore database even if existing databases are online".

Step	Action
7	If you have multiple replication sets, restore the most recent distribution database to maintain the replication settings for all of the other replicated databases.
8	Reinitialize the restored publisher or subscriber databases because they are out of sync with the latest distribution database.

Cloning databases

Understanding database cloning

What database cloning is

Database cloning is the process of creating a point-in-time copy of a production database or its backup set.

Cloned databases can be used for multiple purposes:

- During application development cycles for testing functionality that has to be implemented using the current database structure and content.
- By data extraction and manipulation tools for populating data warehouses.
- For recovering data that was mistakenly deleted or changed.

The database cloning feature enables you to clone all databases simultaneously or select specific databases out of many. You can either rename a cloned database or accept the default name provided. You can select the SQL Server instance either from a host on which the database resides or from a remote host. You cannot perform a database clone on a remote physical server when the database resides on a VMDK.

Note: The remote host must be connected to the storage system containing the database files.

You should delete cloned databases that are no longer relevant.

Completion of a current database cloning operation generates two reports: a backup report and a restore report.

Cloning databases using SnapManager

Tasks performed by the Clone wizard

SnapManager contains a Clone wizard that provides a convenient interface for performing the following cloning operations:

- Clone databases from a local backup or an archived backup
- Clone active production databases
- Clone database on a SnapMirror destination volume
- Delete cloned database

Note: Cloning using the Clone wizard provides you with a complete set of cloning options. Cloning using the Actions pane in SnapManager Restore gives you quick cloning with fewer options than the Clone wizard.

Note: If a database resides on a virtual machine with a VMDK disk, it is not possible to clone the database to a physical server.

Cloning a database from a local backup or an archived backup

Cloning the backup of a database is probably the most commonly used cloning feature. The cloned database can serve as a baseline for developing new applications, or to isolate application errors that occur in the production environment. It could also be used for recovery from soft database errors.

Step	Action
1	In the SnapManager console root, select a server.
2	In the Actions pane, click Clone Wizard.
3	In the Start page, click Next.
4	In the Clone Type page, select Clone Databases from existing Backup Set and click Next.
5	In the Backup Selection page, double-click the backup from which you want to create the clone and then click Next. **Note:** The first time you select a database that resides on a LUN, SnapManager automatically selects all other databases on the same storage. You can then de-select any databases that you do not want to be cloned.
6	In the Restore Settings page, do the following and click Next: • Select backups to restore. • Choose where to apply point-in-time settings. • Choose a point-in-time or marked transaction. For more information about these options, see *Types of SnapManager restore operations* on page 163.
7	Click Next.
8	SnapManager displays the list of databases to be cloned. By default, SnapManager provides the same name to the clone as the original database. You should rename the cloned database. Click Next.
9	In the Restore Settings page, specify the clone database name and click Next.

Step	Action
10	In the Clone to Server page, specify the clone server name, choose whether you will use a letter drive or a mount point, and click Next. If you choose a mount point, specify the mount point directory or accept the default. If you specify a mount point, make sure the directory is empty. If there is a database in the directory, after the mount the database will be in an invalid state. For more information about mount point settings, see *Database integrity verification options* on page 294
11	In the Verification Settings page, you can choose to do the following: • Update SnapMirror after the clone operation completes. • Archive the clone to a SnapVault backup. • Clone from a SnapVault backup.
12	In the Restore Settings page, do any of the following and click Next: • Click Clone Restore Settings to configure advanced settings. • Choose whether you want to clone the database on an available SnapMirror destination volume. • Choose whether you want to change the clone database paths based on the new database name.
13	In the Clone Life Cycle Management page, you can choose to resynchronize the clone and to automatically delete the clone. For more information about these options, see *Understanding cloned database lifecycles* on page 188.
14	In the Restore Settings page, select the state of the database you want after restore and click Next. If you select "Leave the database in read-only mode and available for restoring additional transaction logs, the "Undo file directory" option activates. **Note:** The default path for the SnapInfo directory in the "Undo file directory" option is that of the source host.
15	If you want to run a command or script prior to performing the clone operation or after the clone operation finishes, select the "Run Command Settings" option and click Next. **Result** If you select this option, SnapManager displays the Run Command dialog box. For more information, see *Pre-command and post-command script settings* on page 303.
16	Click Finish. **Result** The Clone Status window is displayed that shows the Clone task list and the Clone Report.

Step	Action
17	Click Start Now to start cloning. The operation is performed, and each item in the Clone Task List is checked off as the task is complete. A message appears indicating the successful completion of the cloning operation.

Cloning a database that is in production

The clone of a database that is in production is used when a new application or function has to be tested with the latest database content as the final step before the application is taken into production.

A current database that is in production must be selected for cloning. Cloning a current database involves two steps. The first step is creating the backup of the selected database and the second step is to restore the database from the just created backup set. The whole cloning process is managed by the Cloning wizard. Options made visible by the Cloning wizard are similar to options available in the Backup and Restore wizard.

To clone a current database, perform the following steps:

Step	Action
1	In the SnapManager console root, select a server.
2	In the Actions pane, click Clone Wizard.
3	In the Start page, click Next.
4	In the Clone Type page, select Clone Active Production Datbases and click Next. **Note:** If you select "Run Through Clone QuickStart Wizard", the wizard applies default options for most of the settings.
5	In the Database Selection page, double-click the backup from which you want to create the clone and then click Next. **Note:** The first time you select a database that resides on a LUN, SnapManager automatically selects all other databases on the same storage. You can then de-select any databases that you do not want to be cloned.
6	Continue with the next steps, as instructed in the wizard.
7	If you want to rename the new database clone's paths based on the name of the new database, select the appropriate check box in the wizard. **Note:** You cannot specify database paths for a clone.
8	To perform a clone on a SnapMirror destination volume, select the "Clone on available SnapMirror destination volumes" check box.

Step	Action
9	If you want to run a command or script prior to performing the clone operation or after the clone operation finishes, select the "Run Command Settings" option. **Result:** If you select this option, SnapManager displays the Run Command dialog box. For more information, see *Pre-command and post-command script settings* on page 303.
10	The wizard takes you to the final option that displays the SnapManager clone task list. Click Start Now to begin the specified tasks. **Result:** The operation is performed, and each item in the Clone Task List is checked off as the task is complete. A message appears indicating the successful completion of the cloning operation.

Cloning using the Clone option in SnapManager Restore

To clone an SQL Server database from a backup set using SnapManager Restore, complete the following steps.

Step	Action
1	Review the list in *Understanding SnapManager Restore* on page 160.
2	Ensure that all Windows Explorer windows are closed on the SQL Server computer that is running SnapManager.
3	Disable any SnapManager operations that are scheduled to run against the SQL Server data you are restoring, including any jobs scheduled on remote management or remote verification servers.
4	In the SnapManager console root, click the SQL Server you want to perform the restore operation on.
5	In the Scope pane, click Restore. **Result** It enables you to navigate the backup sets.
6	In the Results pane, double-click to select the backup set that you want to clone.
7	In the Actions pane, click Clone. **Result** The SnapManager for SQL Server-Clone window appears.
Specify the destination	

Step	Action
8	If you want to clone a database on a server other than the original server, follow these steps: **1.** Click the "..." button next to "Clone to Server (Instance)." **2.** The "Select SQL Server Agent" dialog box appears. **3.** Select the SQL Server to which you want to restore the database. **4.** Choose the Connection by selecting the "Use Windows Authentication" or "Use SQL Server Authentication" option. **5.** Click OK to apply your change and close the dialog box.
9	If you want to clone a database with a different name, follow these steps: **1.** Click the option marked "..." next to "Clone as Database." The "Individual Database Restore As..." dialog box appears. **2.** In the Restore as Database box, enter the database name to which you want the backup restored. This database name must not already exist on the SQL Server instance to which you will be restoring the database. **3.** Click OK to apply your change and close the dialog box. **Note:** When you have completed viewing the data, detach the database and dismount the Snapshot copy.
Specify the restore type	

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td rowspan="3">10</td><td>If...</td><td>Then...</td></tr>
<tr><td>You want to restore to a point-in-time backup</td><td>1. Click the tab marked "..." next to "Point-in-Time Restore"
The Point-in-Time Restore dialog box opens.
2. In the Point-in-Time Restore dialog box, specify the date and time after which transaction logs are not applied to the restored database.
3. Click OK to apply your change and close the dialog box.
Note: A point-in-time restore halts the restoration of transaction log entries that were recorded after the specified date and time.</td></tr>
<tr><td>You want to restore to a marked transaction</td><td>1. Click the tab marked "..." next to "Marked Transaction."
The Marked Transaction dialog box opens.
2. In the Marked Transaction dialog box, select which marked transaction at which to stop the restore operation.
3. Click OK.</td></tr>
<tr><td>11</td><td colspan="2">To start the clone operation, click Clone.
Result SnapManager begins to clone your databases from the backup you selected. SnapManager Clone completes each task and checks it off the list shown in the Restore Task List view.
You can switch back and forth between the task check-off list and the progress report using the Switch buttons on either window.
If the clone is successful, the Task window shows the check-off list with the tasks completed, and a dialog box reports that the restore was successful.
Note: If Notification is enabled, email is sent to the specified address. Events are posted to the Windows event log even if notification is not enabled.</td></tr>
<tr><td>12</td><td colspan="2">After all the restore tasks are finished, click OK.
Result Your restore is complete and your SQL Server computer comes back online.</td></tr>
</table>

Deleting cloned databases

You can delete a cloned database that has outlived its purpose. Deleting the cloned database implies disconnecting the Snapshot copy. To delete cloned databases automatically as part of clone lifecycle management, see "Configuring automatic clone deletion in the Clone wizard" in *Understanding cloned database lifecycles* on page 188.

To delete a cloned database, complete the following steps:

Step	Action
1	If you have not already done so, start SnapManager for SQL Server by accessing the Windows Start menu, and selecting `Program Files > NetApp > SnapManager for SQL Server`. **Result:** The SnapManager for SQL Server console appears.
2	In the Scope pane, double-click SnapManager for SQL Server. **Result:** SnapManager displays the SQL Server database servers running.
3	Click the SQL Server database server that you want to configure. **Result:** SnapManager displays the Status dashboard in the Result pane.
4	In the Actions pane, click Clone Wizard. **Result:** The Clone wizard launches and the Welcome window appears.
5	Click Next. **Result:** SnapManager displays an option for selecting the operation that you want to perform.
6	Select the operation you want to perform, and click Next. **Result:** SnapManager displays the Database to clone window listing the available cloned databases. Select the cloned databases that you want to delete.
7	In the Delete clone summary screen, verify the settings selected in the previous steps and click Finish.
8	This takes you to the final option that displays the Delete clone task list. Click start now to begin the specified tasks. **Result:** The operation is performed, and each item in the Clone Task List is checked off as the task is complete. A message appears indicating the successful completion of the delete clone operation.
9	Click Close to close the Clone Status dialog box.

You can also delete clones through Delete Clone in the Actions pane.

Understanding cloned database lifecycles

What clone database lifecycles are

A cloned database lifecycle management is comprised of automatic, scheduled cloned database resynchronization and deletion. Cloned database resynchronization syncs the cloned database with the live database and automates cloned database deletion.

Managing cloned database lifecycles improves performance in the following ways:

- Relieves manual responsibility for database clone management by automating the process
- Improves resource and storage efficiency by deleting unnecessary clones
- Automates clone database synchronization and deletion

To configure a cloned database lifecycle, complete both "Configuring clone resynchronize in the Clone wizard" and "Configuring automatic clone deletion in the Clone wizard".

Configuring clone resynchronize in the Clone wizard

Clone resynchronize enables you to automate clone database synchronization on a regular schedule.

To enable and configure cloned database resynchronize, complete the following:

Step	Action
1	If you have not already done so, start SnapManager for SQL Server by accessing the Windows Start menu, and selecting **Program Files > NetApp > SnapManager for SQL Server**. **Result: The SnapManager for SQL Server console appears.**
2	In the Scope pane, double-click SnapManager for SQL Server. **Result:** SnapManager displays the SQL Server database servers running.
3	Click the SQL Server database server that you want to configure. **Result:** SnapManager displays the Status dashboard in the Result pane.
4	In the Actions pane, click Clone Wizard. **Result:** The Clone wizard launches and the Welcome window appears.
5	Click Next. **Result:** SnapManager displays an option for selecting the operation that you want to perform.
6	Click Clone Life Cycle Management. **Result:** The pane opens and displays two options: Clone Resynchronize Option and Clone Auto Deletion Option.

Step	Action
7	Click Next in all the subsequent panes until Clone Life Cycle Management pane appears. **Result:** SnapManager displays an option for selecting the operation that you want to perform.
8	Select the Clone Resynchronize Option check box.
9	Enter a schedule for refresh cycles by entering numerical values in the "days," "hours," or "minutes" fields. For example, if you enter "4" in the hours field, SnapManager refreshes the database clone every four hours. **Note:** Clone Resynchronize runs indefinitely at the specified interval if you do not enable Clone Auto Deletion.
10	Set the "Start at" time. For example, if you wanted to begin the clone resynchronize cycle at 12:00 a.m., you would enter "12:00:00". **Result:** The `__clone` suffix is added to the database clone name when the first database clone is created. The `__clone__temp` suffix is added to the database clone name when database clones are created during subsequent resynchronization operations.
11	**Optional:** If you want to protect user connections to the clone database from disconnection during clone resynchronize, deselect the "Terminate connection to the clone database during clone refresh..." option. **Note:** Disabling this option causes the clone resynchronize operation to fail if there are any user connections to the clone database.
12	Click Finish.
13	The Schedule Job Creation window displays. Specify whether you want to create a job using the SQL agent or the Windows Task Scheduler. Enter the appropriate user credentials and click OK.

Note: Changes to the cloned database are lost when a clone is refreshed.

Configuring automatic clone deletion in the Clone wizard

Clone Auto Deletion enables you to automate clone database deletion on a regular schedule.

To enable and configure automatic clone deletion, complete the following steps:

Step	Action
1	If you have not already done so, start SnapManager for SQL Server by accessing the Windows Start menu, and selecting **Program Files > NetApp > SnapManager for SQL Server**. **Result:** The SnapManager for SQL Server console appears.
2	In the Scope pane, double-click SnapManager for SQL Server. **Result:** SnapManager displays the SQL Server database servers that are running.
3	Click the SQL Server database server that you want to configure. **Result:** SnapManager displays the Status dashboard in the Result pane.
4	In the Actions pane, click Clone Wizard. **Result:** The Clone wizard launches and the Welcome window appears.
5	Click Next. **Result:** SnapManager displays an option for selecting the operation that you want to perform.
6	Click Clone Life Cycle Management. **Result:** The pane opens and displays two options: Clone Resynchronize and Clone Auto Deletion.
7	Select the Clone Auto Deletion check box.
8	Set the "Delete at" time. For example, if you wanted to delete the clone database at 12:00 a.m., you would enter "12:00:00."
9	Enter a schedule for clone deletion by entering numerical values in the "days," "hours," or "minutes" fields. For example, if you enter "4" in the hours field and "12:00:00" in the Start from field, SnapManager deletes the clone database four hours after 12:00:00. **Note:** The minimum automatic clone deletion period is five minutes.
10	Click Finish.

Creating a clone replica of an AlwaysOn cluster

If you need a failover copy of the databases in an AlwaysOn cluster you can use the SnapManager for SQL Server Availability Group Replica Wizard to create them.

About this task

The SnapManager for SQL Server Availability Group Replica Wizard uses Snapshot copies to quickly clone the databases to a remote server and then join the clone databases to an Availability Group as a new availability group database replica. After the clone replica is completed, you have a new secondary replica created on the existing availability group, as a normal secondary replica. A Clone replica should be used in same manner as a clone database, only for temporary purposes.

Steps

1. From the management console, select the server hosting the database you want to clone. For example, **Console Root > SnapManager for SQL Server > AlwaysOn Cluster1**
2. Select **Replica Wizard** in the **Actions** window.
3. Follow the steps in the **SnapManager for SQL Server Availability Group Replica Wizard** to specify the source, settings for the replica, and the destination for the replica. Notice that in the **Quick Clone Replica** step you can click **Run through Quick Availability Group Clone Replica Wizard** so that the wizard automatically sets the mandatory settings.

Using VMDKs with SnapManager for SQL Server

Setting up VMDK support

Overview of VMDK setup

Before you begin installing or upgrading SnapManager, you must complete certain prerequisites. See *Preinstall or preupgrade procedure* on page 19. If your system will include VMDK disks, before you install SnapManager, you must also complete these tasks:

- Installing Virtual Storage Console for VMware vSphere
- Creating a virtual machine
- Creating VMDK disks

Installing Virtual Storage Console for VMware vSphere

To support VMDK disks, you must install VSC for VMware vSphere.

To install VSC, complete the following steps.

Step	Action
1	Determine where to install VSC for VMware vSphere. VSC can reside on the vCenter Server or a dedicated server.
2	Install VSC for VMware vSphere on the server. Refer to VSC documentation for installation requirements and steps.
3	Configure VSC for VMware vSphere to discover your storage environment.

Creating a virtual machine

To support VMDK disks, create one or more virtual machines.

Step	Action
1	Create an NFS, iSCSI, FC, or VMFS datastore.
2	Create a virtual machine on the datastore you just created.
3	Install the guest operating system on the virtual machine.

Step	Action
4	Install SnapDrive for Windows on the virtual machine and enter the IP address of the management server and credentials used to communicate with VSC on the management server. **Note:** If you will not be using RDM LUNs on the virtual machine, do not enter the ESX/VSC credentials.

Creating VMDK disks

For SQL Server on VMDK, you must create one or more VMDKs from an NFS or VMFS datastore and attach it to the virtual machine.

To create a VMDK disk and attach it to the virtual machine, complete the following steps.

Step	Action
1	Create a VMDK on an NFS or VMFS datastore on the management server.
2	On the virtual machine, open the Disk Management console, initialize the new disk, and assign a drive letter or mount point to the disk. **Note:** See the *SnapDrive Installation and Administration Guide* or the *Virtual Storage Console for VMware vSphere Installation and Administration Guide* for information about how to create a VMDK.

After you complete the preceding tasks, you can continue preparing to install or upgrade SnapManager. Go to *Installing or upgrading SnapManager* on page 28.

Related information

Documentation on the NetApp Support Site: support.netapp.com

Backing up databases on VMDKs

Backing up databases on local servers

Backing up databases on VMDKs from local servers is similar to backing up databases on LUNs and SMB shares. For information about performing backups and verifications to local servers, see *Backing up, replicating, and archiving databases using SnapManager* on page 115. However, you should be aware of the following behaviors and limitations:

- When you select a database that resides on a VMDK, SnapManager automatically selects all the databases that reside on the VMDK.
- SnapVault integration is not supported for VMDK disks.
- The backup created on the VMDK cannot be verified on a remote physical server.

- However, you can select a verification server running on a virtual machine.

Limitations for verification on destination volumes

SnapManager does not support verification on SnapMirror destination volumes, if the source and destination volumes contain VMFS datastores. This limitation applies to clustered Data ONTAP only.

Prerequisites for verification and cloning on destination volumes

You can verify and clone from destination volumes when the database hosted on the VMDKs is replicated to a site by SnapMirror and the configuration meets the following requirements:

- The virtual machine is installed on the ESX server on the secondary site.
- SQL Server, SnapDrive, and SnapManager for SQL are installed on the virtual machine.
- The ESX server is managed by another vCenter Server and the VSC server on the secondary site.
- SnapDrive is installed on the secondary virtual machine that is pointing to the VSC server on the secondary site.
- On the primary site, you have selected the SQL Server on the secondary site as the remote verification server.
- On both the primary and secondary VSC servers, you have created a Windows share on the VSC repository folder where the backup metadata file resides.
- The SnapManager service account has read permission on the share at the primary site and write and modify permissions at the share on the secondary site.
- The primary VSC server has discovered the destination storage system.
- For NFS datastores residing on clustered Data ONTAP, a datastore must exist on the SnapMirror destination volume and the name of the destination datastore must be specified in the change list file.
- On the primary virtual machine where the backup is initiated, the following registry settings and values must be defined in `HKEY_LOCAL_MACHINE\SOFTWARE\Network Appliance \SnapManager for SQL Server\Server`:

SMVITransformEnable	dword:00000001
SMVITransformScript	SMVI_Metadata_update.exe
SMVIDestinationServer	*destination SMVI server name*
SMVISourceBackupXmlUNC	*source SMVI server name**SMVI repository share name*\backups.xml
SMVIDestinationBackupXmlUNC	*destination SMVI server name**SMVI repository share name*\backups.xml
SMVIChangeListFile	*change list file name*

Requirements for the change list file

The change list file is a text file that contains information about source and destination volumes. The file is required for backup verification on SnapMirror destination volumes. The contents of the file must be in the following format, with fields separated by a space and each datastore beginning a new line:

```
DatastoreType SourceDatastoreName DestinationDatastoreName
SourceVirtualMachineName DestinationVirtualMachineName
SourceVirtualMachineUUID DestinationVirtualMachineUUID
SourceVirtualMachineDirectoryName DestinationVirtualMachineDirectoryName
SourceStorageName DestinationStorageName SourceVolumeName
DestinationVolumeName [SourceDatastoreUUID DestinationDatastoreUUID]
```

In this format, *`Datastore Type`* is either NFS or VMFS and *`Datastore UUID`* is not required for an NFS volume.

Note: For NFS datastores residing on clustered Data ONTAP, *`SourceStorageName`* and *`DestinationStorageName`* must be the IP addresses of the source and destination NFS data LIFs.

The following example shows the contents of a change list file:

```
NFS ds-nfs1 ds-nfs1-dest snapmgr-05-vm2 snapmgr-54-vm1 4211945a-124a-b7c9-
ae63-cacc07f3f4f8 420f010b-7e5a-e66e-7ed1-7bef6a357cca snapmgr-05-vm1
snapmgr-54-vm1 172.17.233.24 172.17.232.74 snapmgr05_vmw1
snapmgr05_vmw1_mir
```

Verification on destination volumes

Backing up databases on VMDKs and verifying on destination volumes is different from backing up and verifying databases on LUNs and SMB shares. You can create full database backups when the databases are hosted on the VMDK that is replicated to a site by SnapMirror.

If you want to perform verification on a destination volume, you must specify a remote verification server and ensure that the configuration meets the requirements stated in "Prerequisites for verification on destination volumes and cloning on destination volumes." Only the backups created in this type of configuration can be verified from the destination volume on a remote SQL server.

See the following sections for performing a backup and a verification separately:

- *Backing up, replicating, and archiving databases using SnapManager* on page 115
- *Performing database verification using SnapManager* on page 138

Remote verification on source volumes

To perform a remote verification on a source volume, the remote verification server must be running on a virtual machine.

As with the physical server, SQL Server, SnapDrive, and SnapManager for SQL must be installed on the remote server. But SnapDrive must be pointing to the same VSC server as the local server.

Cloning databases on a VMDK

Cloning databases from source volumes

Cloning databases residing on VMDKs using a source volume is similar to cloning databases from LUNs and SMB shares. For information about performing clones to local servers, see *Cloning databases using SnapManager* on page 180. However, you should be aware of the following limitations and behaviors:

- You cannot perform a database clone on a remote physical server when the database resides on a VMDK.
- You cannot clone a database on a SnapVault secondary because there is no remote backup available for clone operation.

Cloning databases from destination volumes

Cloning databases on VMDKs to destination servers is different from cloning databases from SnapMirror destination volumes on physical servers.

When the databases are hosted on the VMDKs that are replicated to a site by SnapMirror, you cannot clone databases from a SnapMirror destination volume to the local SQL Server. However, you can clone databases from destination volumes to an SQL Server running on a remote virtual machine. You must ensure the configuration meets the requirements in the prerequisites section of *Backing up databases on VMDKs* on page 193 before backing up the databases. Only the backups created in such configuration can be cloned from the destination volume to a remote SQL server.

SnapManager does not support cloning on SnapMirror destination volumes, if the source and destination volumes contain VMFS datastores. This limitation applies to clustered Data ONTAP only.

Performing disaster recovery of databases on VMDKs

Overview

The disaster recovery of databases on VMDKs involves the disaster recovery of the virtual infrastructure by Virtual Storage Console for VMware vSphere. Refer to the *SnapManager for Virtual Infrastructure Best Practices Guide* for more information.

Preparing primary site for disaster recovery

Prepare the primary site for disaster recovery by completing the following step:

Step	Action
1	Before creating a backup by SnapManager for SQL Server on the primary side, modify the registry keys by completing the following substeps. This enables SnapManager for SQL Server to update the metadata from the primary VSC for VMware vSphere server to the secondary VSC server. **1.** On the primary server, navigate to the location of the registry keys at: HKEY_LOCAL_MACHINE\SOFTWARE\NetworkAppliance\SnapManager for SQL Server\Server **2.** Change the registry keys as follows: SMVIChangeListFile: The change list file path on the Primary virtual machine (for example, C:\DR\dr_info.txt). SMVIDestinationBackupXmIUNC: The path of the secondary SMVI server's backups.xml path (for example, \\DestinationSMVIServer\repository\backups.xml). SMVIDestinationServer: The name or IP of the destination VSC server. SMVISourceBackupXmIUNC: The path of the primary SMVI server's backups.xml path (for example, \\PrimarySMVIServer\repository\backups.xml). SMVITransformEnable: 1.

Preparing the disaster recovery standby site

Complete the following steps for configuring the standby site:

Step	Action
1	Install VSC for VMware vSphere.
2	Configure VSC to use the volumes on the destination side (secondary site) storage systems.
3	Enter the vCenter server and storage system IP addresses or names in the VSC Setup window.
4	Run the `smvi servercredential set` command from the CLI, if necessary.
5	Stop the VSC service in Windows.
6	Establish the SnapMirror relationship on the underlying volume from the primary site to secondary site. Volumes used for VSC on the destination side storage should be used as the SnapMirror destination volumes.

Step	Action
7	Create a Windows share on the repository of both the primary and secondary VSC servers. Make sure that the SnapManager service account has Read permission on the share at the primary site and Write and Modify permissions at the share on the disaster recovery site.

8	Create a text file and save the following list information in the file.
	You can use the format provided in "Prerequisites for verification on destination volumes and cloning on destination volumes" in *Backing up databases on VMDKs* on page 193.
	`datastore type datastore name of both sites virtual machine name of both sites`
	`virtual machine uuid of both sites virtual machine directory name of both sites storage system name or IP address of both sites volume name of both sites| datastore uuid of both sites in case of VMFS type of datastore`
	Ensure that all of the above information is in one line per datastore. Each field is separated with a space.
	Note: The virtual machine name and its uuid can be the same if there is no preinstalled standby virtual machine on the disaster recovery site.

9	Save this file to any folder on the primary virtual machine or any other server that the SnapManager service can access via Windows share.

Performing disaster recovery of databases on VMDKs

To perform disaster recovery of databases residing on VMDKs to destination servers, complete the following steps:

Step	Action
1	Break the SnapMirror relationship from the storage system.
2	Bring online the SnapMirror destination volumes on which the datastores reside.
3	Create an NFS export for the NFS storage on the storage system for the destination volume.
4	Add the new NFS export to each of the destination VSC servers on the ESX.
5	Right click on the data store and select **`Browse data store`**.
6	In the left pane, click the virtual machine's name.

Step	Action
7	In the right pane, right click the virtual machine's VMX file and select the option **`Add to Inventory.`**
8	Follow the steps in the wizard to add the virtual machine to the ESX server.
9	Power on the virtual machine.
10	Log into the virtual machine.
11	From the command prompt, enter the following: `sdcli smvi_config list` The command lists the primary VSC server.
12	Switch to the secondary VSC server by entering the following command: `sdcli smvi_config set -host <IP of the secondary SMVI Server>`

13	Restart the SnapDrive for Windows service using the following commands:
	`net stop swsvc`
	`net start swsvc`

14	After the SnapDrive for Windows serve starts successfully, check if all of the VMDKs are available by entering the following command: `sdcli disk list`
15	On the recovered virtual machine, run SnapManager for SQL Server restore to recover SQL databases.

Managing SnapManager operational reports

Understanding the SnapManager Reports option

Understanding the SnapManager Reports option

Use the SnapManager Reports option in the Scope pane to access the operational reports that are automatically created for SnapManager configuration, backup, restore, backup set deletion, and other miscellaneous operations. Each report is a log file that includes step-by-step details of the operation, the final status of the operation, and any error messages encountered during the operation.

The SnapManager Reports option consists of a *navigation panel* and a *display panel*. The navigation panel contains a tree structure that enables you to navigate the *folders* into which the individual *reports* are organized. Each report is a log file that is named in the format *mm-dd-yyyy_hh.mm.ss* that serves to time stamp the creation of the report. Note that *hh* represents the hour expressed in military time. The display area displays the contents of the selected log file.

The following paragraphs describe the folders that contain the SnapManager reports.

Backup Contains a log file for every backup set (full database backup or transaction log backup) created by SnapManager.

Config Contains a log file for each time SnapManager migrates a database.

Debug Contains a debugging log in SnapManager when debug logging is enabled.

Delete Backup Set Contains a log file for every delete backup operation.

Miscellaneous Contains log files for all other operations.

Monitor Contains a log file for SnapManager monitoring features.

Restore Contains a log for every restore operation (whether it is a stream-based restore, a copy-based restore, or an online Snapshot restore) performed on an SQL Server that is configured using SnapManager.

Managing reports

Viewing reports

To view a SnapManager report, complete the following steps.

Step	Action
1	In the SnapManager console root, click the SnapManager Reports option.
2	In the navigation panel, click to expand the appropriate reports folder and select the report you want to display in the display panel.

Managing reports

To manage your reports, perform the actions listed in the table below:

If you want to...	Then...
Refresh a report that you are viewing	Click "Refresh" in the Actions pane.
Delete a specific report	Select the report and click "Delete" in the Actions pane.
Delete all reports of a folder	Select the folder in the Results pane and click "Delete All" in the Actions pane.
Delete selected reports	Press Ctrl or Shift, and then click the reports you want to delete. Click "Delete" in the Actions pane.
Open the report in a new window	Select the report and click "New window" in the Actions pane.
Open the report in Notepad	Select the report and click "Open in Notepad" in the Actions pane.
Print the report	Select the report and click "Print" in the Actions pane.
Print selected reports	Press Control or Shift, and then click the reports you want to print. Click "Print" in the Actions pane.
Preview the print layout of the report	Select the report and click "Print Preview" in the Actions pane.
Find a word in a report	Select the report and click "Find in Report" in the Actions pane.

Understanding monitoring and reporting

About monitoring and reporting

You can receive automatic, scheduled email notifications on overall status of all backup, verification, and clone operations. You can use these email notifications to summarize information about both successful and failed operations. Each email notification can include the following information:

- Summary of operations

- Summary of individual SQL Server instances, with the number of successful and failed operations
- List of all operations performed on individual SQL Server instances

  Note: The SnapManager monitoring and reporting reports the full backup as two operations: one for backup and one for verification.

- Summary of successful, failed, and "not run" operations on individual SQL Server instances

  Note: For backup and verification operations, incomplete or "not run" operations are logged as an error in the Windows event log, but are logged with an informational message for clone operations.

- List of all failed operations for individual SQL Server instance

Configuring monitoring and reporting settings

You can use the monitoring and reporting functions to receive email notifications for status of backup, verification, and clone operations. You can choose which operations you receive notifications for, how frequently the reports are sent, and at what time the report operations begins.

To configure your monitoring and reporting settings, complete the following steps:

Step	Action
1	From the Start menu, select **Program Files > NetApp > SnapManager for SQL Server**.
2	Select the SQL Server you want to manage.
3	In the Actions pane, click Monitor Settings. **Result:** The Monitoring and Reporting Settings window opens.
4	Select the Enable Monitoring and Reporting check box.
5	Choose which operations to be monitored by SnapManager. • For Backup • For Verification • For Clone Resync **Example:** If you select only "For Backup" and "For Clone Resync," you receive email notifications on all backup and clone operations that SnapManager for SQL Server performs, but not on verification operations.
6	In the Select Reporting Interval pane, enter the interval at which you want to receive notifications. **Example:** If you want to receive email notifications once per day, enter 1 in the "days" field.

Step	Action
7	In the "Report operations start at" field, set the time at which the report operations begins.
8	Click OK.

Recovering your SQL Server environment

Reseeding a database on an AlwaysOn cluster

If a replica database gets out of sync with the primary database in an Availability Group, you can reseed the replica database from an existing SnapManager backup. Using the reseed operation, any unhealthy secondary database can be recovered and brought back in synch with its primary database. This is quicker than copying the primary database to the replica and requires less network bandwidth.

Before you begin

A common SnapManager share for all the replica nodes and the share retention settings configured using **Actions > Backup Settings > Transaction log backup > Repository Log backup Options**.

About this task

Note that you cannot use a stream-based backup to reseed a database, and so stream-based backups are not displayed by the Reseed Wizard.

Primary databases and non-Availability Group databases are skipped during the reseed operation.

Steps

1. From the management console, select the standalone server hosting the Availability group databases you want to use to reseed. For example, **Console Root > SnapManager for SQL Server > AlwaysOn Cluster 1**.
2. Select **Reseed Wizard** in the **Actions** window.
   The Reseed Wizard opens.
3. Follow the steps in **Reseed Wizard** to select the database, logs, and optionally a custom command, before verifying the reseed settings and starting the reseed operation.

Recovering SQL Server databases using SnapMirror

Preparing for disaster recovery using mirrored volumes

After the failure of a storage system or a volume on a storage system, you can recover SQL Server databases that are mirrored using SnapMirror. To be able to recover SQL Server databases using SnapMirror, you must complete the following disaster recovery preparation tasks:

Configure SnapMirror to replicate SQL Server database backups to mirrored volumes: You can configure SnapMirror to use a destination volume on the same storage system or a different,

remote storage system. Whether the destination volume is on the same or a different storage system as the failed volume, the disaster recovery procedure is largely the same. For more information about configuring the Data ONTAP SnapMirror feature, see the *Data ONTAP Data Protection Guide* for your particular version of Data ONTAP.

Note the drive letter mappings: For each LUN on the SnapMirror source volume, note the relevant drive letter mapping on the SQL Server computer. You must use the same mappings during the disaster recovery procedure when you use SnapDrive to connect to the corresponding LUNs on the SnapMirror destination volume.

Note the name of the SMB share: You must use the name of the original share.

Procedure overview

The procedure entails recovering databases from the SnapMirror destination volume.

Basic procedure: Use SnapDrive for Windows to connect to the data file backups, transaction log backups, and SnapInfo LUNs, and then perform a SnapManager-based restore operation on the database.

If you cannot attach the database to the SnapMirror destination volume and the transaction log files are intact, follow the steps that describe how to minimize data loss by ensuring that SnapManager restore operations automatically back up the last active transaction log of the database.

If you succeed in recovering the necessary LUNs and SMB shares, you can use SnapManager to restore from the most recent backup set. Some of the details of this phase of the recovery procedure depend on the nature of the storage system or volume failure:

- Only SQL Server database data files were lost
- Only SQL Server transaction log files were lost
- Both SQL Server database files and transaction log files were lost

If the transaction log files were lost: The recovery procedure includes special steps you must take if the transaction files were lost:

- If the transaction log files were lost, the active transactions were lost and are unrecoverable. Because the active transactions are unavailable, you must use SnapManager to perform a point-in-time restore and not an up-to-the minute restore.
- For descriptions of the two types of restore operations, see *Types of SnapManager restore operations* on page 163.
- In addition, the SnapManager Restore from the SnapMirror destination volume must be performed with the transaction log backup option disabled if the transaction log files were lost.
- If you fail to disable this transaction log backup, subsequent SnapManager backup sets reside on a recovery path that is inconsistent with that of the database.

  Such backup sets cannot be applied to the database; attempts to restore the database from such backup sets results in failure, with the following error message in the restore log:

  ```
  Failed with error code 0x800410df
  ```

If this occurs, perform the restore operation again, but do not apply transaction logs.

Performing disaster recovery using mirrored volumes

To recover SQL Server databases whose backup sets have been mirrored using SnapMirror, complete the following steps.

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td rowspan="2">1</td><td>If...</td><td>Then...</td></tr>
<tr><td>You are recovering SQL Server 2005 and SQL Server 2008 databases</td><td>Go to Step 2.</td></tr>
<tr><td>2</td><td colspan="2">If any LUNs from the failed source volume still appear to be connected, disconnect them.</td></tr>
<tr><td rowspan="3">3</td><td>If...</td><td>Then...</td></tr>
<tr><td>You have LUNs</td><td>Use SnapDrive to connect to the corresponding LUNs in the SnapMirror destination volume. Use the same drive letters for connecting to the mirrored LUNs that were used on the source volume.
For each mirrored volume, SnapDrive breaks the replica and restores the LUN using the most recent Snapshot copy generated by SnapDrive or SnapManager.</td></tr>
<tr><td>You have SMB shares</td><td>1. Manually disable the mirror relationship using the Data ONTAP CLI or a management tool.
2. Create a new share that matches the name of the original share.</td></tr>
<tr><td>4</td><td colspan="2">Restart SQL Server if it has been stopped.</td></tr>
<tr><td rowspan="4">5</td><td colspan="2">Use SQL Server Management Studio to attach the database located on the associated LUNs or SMB shares in the SnapMirror destination volume, as follows:</td></tr>
<tr><td>If...</td><td>Then...</td></tr>
<tr><td>You succeeded in attaching the database</td><td>Complete this procedure as described in Step 6.</td></tr>
<tr><td>The database could not be accessed using SQL Server Management Studio</td><td>Complete this procedure as described beginning with Step 10.</td></tr>
<tr><td></td><td>You were unable to attach the database</td><td>Complete this procedure as described beginning with Step 10.</td></tr>
</table>

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td colspan="3">If you attached the database on the SnapMirror destination volume</td></tr>
<tr><td>6</td><td colspan="2">The steps for restoring the database depend on whether the transaction log volume was lost:</td></tr>
<tr><td></td><td>If...</td><td>Then...</td></tr>
<tr><td></td><td>You lost only the data files of the database</td><td>Complete this procedure as described in Step 7.</td></tr>
<tr><td></td><td>If you lost only the transaction log files of the database</td><td>Complete this procedure as described in Step 8.</td></tr>
<tr><td></td><td>If you lost both the data files and the transaction log files of the database</td><td>Complete this procedure as described in Step 8.</td></tr>
<tr><td colspan="3">Performing the restore if only the data volume was lost</td></tr>
<tr><td>7</td><td colspan="2">Run SnapManager and use the newest full database backup copy to perform either an up-to-the-minute restore or a point-in-time restore:
For an up-to-the-minute restore, SnapManager automatically backs up the last active transaction log before performing the restore operation.
For a point-in-time restore, select the backup set, a combination of transaction log backups to be restored, or both.
For more information, see the following topics:
For information about identifying the most recent full database backup, see "SnapManager backup set names" in How SnapManager backup data is organized on page 99.
For information about identifying the transaction logs, see "SnapInfo subdirectory names" in How SnapManager backup data is organized on page 99 and "Transaction log backup" in Types of backup operations performed using SnapManager on page 103.
For information about performing a restore operation, see "Restoring using the SnapManager Restore option" in Performing a restore operation on page 167.
The procedure is now complete.</td></tr>
<tr><td colspan="3">Performing the restore operation if the transaction log volume was lost</td></tr>
</table>

Step	Action
8	Disable the option to back up the transaction log before performing the restore operation: **1.** From the SnapManager menu bar, select **Options > Restore Settings**. **2.** In the Restore Settings dialog box, clear **Create transaction log backup before restore**. **3.** Click **OK**. The reason you must disable this restore option is that the active transactions were lost due to the failure of the volume containing the transaction log.
9	Run SnapManager and use the newest full database backup copy to perform a point-in-time restore. **Note:** Because the transaction log volume was lost, an up-to-the minute restore is not possible. For a point-in-time restore, select the backup set, a combination of transaction log backups to be restored, or both. For more information, see the following topics: • For information about identifying the most recent full database backup, see "SnapManager backup set names" in *How SnapManager backup data is organized* on page 99. • For information about identifying the transaction logs, see "SnapInfo subdirectory names" in *How SnapManager backup data is organized* on page 99 and "Transaction log backup" in *Types of backup operations performed using SnapManager* on page 103. • For information about performing a restore operation, see "Restoring using the SnapManager Restore option" in *Performing a restore operation* on page 167. The procedure is now finished.
If you did not attach the database on the SnapMirror destination volume	

Step	Action
10	If you cannot attach the database on the SnapMirror destination volume and none of the transaction log files were lost, then, to reduce the loss of data, ensure that the last active transaction log of the database is automatically backed up by SnapManager Restore: • See Microsoft KB article 253817, "HOW TO: Back up the Last Transaction Log When the Master and the Database Files Are Damaged." This article describes how you can back up the currently active transaction log even if the SQL Server database file is damaged, provided that the transaction log file is still accessible. • Use this same Microsoft KB article as a general guide for gaining access to the last active transaction log of the database on the SnapMirror destination volume. While referring to the steps in that article, observe the following key points: ◦ When you create a similar database that contains the same number of data and transaction log files as the original database on the SnapMirror destination volume, you are creating the database you will be restoring using SnapManager. ◦ Instead of using the SQL Server `Backup Log` command to back up the transaction log (as described in the Microsoft article), go to the next step in this procedure. • For information about identifying the transaction logs, see "SnapInfo subdirectory names" in *How SnapManager backup data is organized* on page 99 and "Transaction log backup" in *Types of backup operations performed using SnapManager* on page 103. **Attention:** Do not use SQL Server Management Studio to back up the last active transaction log. Due to file formatting differences between SnapManager backups and Management Studio backups, SnapManager cannot perform a restore operation using transaction logs that were backed up using Management Studio. If any of the transaction log files were lost, no workaround is possible and you cannot minimize data loss.

Step	Action
11	Use SnapManager Restore to automatically back up the last active transaction log of the database. **1.** Start the SnapManager application. **2.** Select **Options > Restore Settings**, and ensure that **Create transaction log backup before restore** is enabled. This causes SnapManager Restore to automatically back up the last active transaction log before actually performing the restore portion of the operation. **3.** Use the newest full database backup copy to perform either an up-to-the-minute restore or a point-in-time restore to the new database you created in the previous step. For information about identifying the most recent full database backup, see "SnapManager backup set names" in *How SnapManager backup data is organized* on page 99. For general information about performing a restore operation, see "Restoring using the SnapManager Restore option" in *Performing a restore operation* on page 167.

Recovering SQL Server databases using archives

System prerequisites

To restore SQL Server databases from archives, the following prerequisites must be met.

Storage system The storage system must be up and running and ready for data to be restored.

Backup media The backup media must be available and ready to be used for restore.

Database If the database is still mounted, detach it, using SQL Server Enterprise Manager or SQL Server Management Studio.

Windows Server You must restore the Windows system and all services required by the SQL Server. SnapManager, the SQL Server, and the storage systems depend on Microsoft Windows infrastructure elements such as DNS and Active Directory.

LUNs Disconnect the LUNs from the Windows host machines.

Information needed

Before you begin restoring your SQL Server databases from archive, you need the following information.

Backup and restore method You should be familiar with the backup and restore method you are using for the LUNs, SMB shares, and SnapInfo directory (either the storage system `dump` command or an NDMP-based backup). See the product documentation specific to the backup application.

Supporting documentation Have the supporting documentation for SnapDrive, Data ONTAP, and your backup application available for reference.

LUN drive letters You need to know the original drive letters used by the LUNs because LUN objects restored from archive must be reconnected using the same drive letters.

SMB share name The SMB shares must use the same share name.

Procedure summary

The following steps represent a high-level overview of the "Restore from Unmanaged Media" process:

1. Recover the archived LUNs and SMB shares containing the full backup dataset to the active file system of the storage system.
2. Reconnect the LUNs to the original drive letters and give the hosts access to the shares.

Restoring a database on an AlwaysOn cluster

SnapManager streamlines and simplifies the restoration of a database on an AlwaysOn cluster. You can restore a database from the same SQL Server, from unmanaged media, and from a backup created on a different server. You can also verify the restore.

Before you begin

If the restore is from a database on a different SQL server, the source storage must have been made available to the current SQL server.

Steps

1. From the management console, select the standalone server hosting the databases you want to use to reseed. For example, **Console Root > SnapManager for SQL Server > AlwaysOn Cluster 1**
2. Select **Restore Wizard** in the **Actions** window.
3. Follow the steps in the **Restore Wizard** to select the database, logs, database state after the restore, optional new name, target location, and optionally adjust the restore settings.

Restoring databases from other SQL Server backups

About this section

You can restore databases to the current SQL Server using SnapManager backup sets that were created for a different SQL Server. If the original SQL Server fails, this feature enables you to recover its databases using a different SQL Server.

You can perform a restore operation from other SQL Server backups using the SnapManager Find Backups option or the SnapManager Restore wizard.

Restoring from other SQL Server backups using SnapManager Find Backups

To use the SnapManager Find Backups option to restore databases to an SQL Server using backup sets created for other SQL Servers, complete the following steps.

Step	Action	
1	If you have LUNs, and the source LUNs for the failed databases are still online and mapped on the primary storage, do the following: **1.** Note the LUN drive letter assignments. **2.** Unmap the LUNs using FilerView or the `lun` command on the storage system console. **3.** In MSCS configurations, remove any cluster resource dependencies you might have configured on these LUNs.	
2	If you have LUNs, reconnect the restored LUN objects with the SnapDrive MMC interface, using the original drive letters. Consult the SnapDrive documentation for details. Ensure that the LUNs are accessible on the hosting SQL Server.	
3	Use SQL Server Management Studio to attach the database located on the LUNs and SMB shares.	
	If...	Then...
	You succeeded in attaching the database	Complete this procedure as described beginning with Step 4.
	It is not possible to attach the database	Complete this procedure as described beginning with Step 9.
If you attached the database		
4	Start the SnapManager for Microsoft SQL Server application.	
5	Click **Restore** in the Actions pane. The SnapManager for SQL-Restore dialog box appears.	
6	In the Restore to Server box, select the SQL Server whose backup sets you want to use to restore databases to this SQL Server.	
7	Click the **...** tab next to the Point-in-time option. The Point-in-time dialog box appears.	

Step	Action
8	Use the Point-in-time option to perform an up-to-the-minute restore operation or a point-in-time restore operation: • For an up-to-the-minute restore operation, back up the most recent transactions and select them for restore by selecting **Most recent backup selected**. • For a point-in-time restore operation, select the backup set, a combination of transaction log backups to be restored, or both, and select **Committed transactions at the specified time**. For detailed information, follow the steps in “Restoring using the SnapManager Restore option” in *Performing a restore operation* on page 167. For more information, see the following topics: • For information about identifying the most recent full database backup, see "SnapManager backup set names" in *How SnapManager backup data is organized* on page 99. • For information about identifying the transaction logs, see "SnapInfo subdirectory names" in *How SnapManager backup data is organized* on page 99. The procedure is now complete.
If you did not attach the database	

Step	Action
9	If you cannot attach the database, then, to reduce the loss of data, ensure that the last active transaction log of the database is automatically backed up by SnapManager Restore: • See Microsoft KB article 253817, "HOW TO: Backup the Last Transaction Log When the Master and the Database Files Are Damaged." This article describes how you can back up the currently active transaction log even if the SQL Server database file is damaged, provided that the transaction log file is still accessible. • Use this same Microsoft KB article as a general guide for gaining access to the last active transaction log of the database. While referring to the steps in that article, observe the following key points: ◦ When you create a similar database that contains the same number of data and transaction log files as the original database, you are creating the database you will be restoring using SnapManager. ◦ Instead of using the SQL Server `Backup Log` command to back up the transaction log (as described in the Microsoft article), proceed to the next step in this procedure. • For information about identifying the transaction logs, see "SnapInfo subdirectory names" in *How SnapManager backup data is organized* on page 99 and "Transaction log backup" in *Types of backup operations performed using SnapManager* on page 103. **Attention:** Do not use SQL Server Management Studio to back up the last active transaction log. Due to file formatting differences between SnapManager Backup and Management Studio backups, SnapManager cannot perform a restore operation using transaction logs that were backed up using Management Studio.
10	Start the SnapManager for Microsoft SQL Server application.
11	Click **Restore** in the Actions pane. The SnapManager for SQL-Restore dialog box appears.
12	In the Restore to Server box, select the SQL Server whose backup sets you want to use to restore databases to this SQL Server.

Step	Action
13	Use the Point-in-time option to perform an up-to-the-minute restore operation or a point-in-time restore operation: • For an up-to-the-minute restore operation, back up the most recent transactions and select them for restore by selecting **Most recent backup selected**. • For a point-in-time restore operation, select the backup set, a combination of transaction log backups to be restored, or both, and select **Committed transactions at the specified time**. For detailed information, follow the steps in "Restoring using the SnapManager Restore option" in *Performing a restore operation* on page 167. For more information, see the following topics: • For information about identifying the most recent full database backup, see "SnapManager backup set names" in *How SnapManager backup data is organized* on page 99. • For information about identifying the transaction logs, see "SnapInfo subdirectory names" in *How SnapManager backup data is organized* on page 99 and "Transaction log backup" in *Types of backup operations performed using SnapManager* on page 103. The procedure is now finished.

Restoring from other SQL Server backups using the SnapManager Restore wizard

To use the SnapManager Restore wizard to restore databases to this SQL Server using backup sets created for other SQL Servers, complete the following steps.

Note: Before you can restore databases to the current SQL Server using backups created for a different SQL Server, you must first remap the source LUNs to this SQL Server, using the same drive letter assignments that were used for the original SQL Server.

Step	Action
1	If you have LUNs and the source LUNs for the failed databases are still online and mapped on the primary storage, do the following: **1.** Note the LUN drive letter assignments. **2.** Unmap the LUNs using FilerView or the `lun` command on the storage system console. **3.** In MSCS configurations, remove any cluster resource dependencies you might have configured on these LUNs.

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td>2</td><td colspan="2">If you have LUNs, reconnect the restored LUN objects with the SnapDrive MMC interface using the original drive letters.
Consult the SnapDrive documentation for details. Ensure that the LUNs are accessible on the hosting SQL Server.</td></tr>
<tr><td rowspan="3">3</td><td colspan="2">Use SQL Server Management Studio to attach the database located on the LUNs and SMB shares.</td></tr>
<tr><td>If...</td><td>Then...</td></tr>
<tr><td>You succeeded in attaching the database</td><td>Complete this procedure as described beginning with Step 4.</td></tr>
<tr><td>It is not possible to attach the database</td><td>Complete this procedure as described beginning with Step 14.</td></tr>
<tr><td colspan="3">If you attached the database</td></tr>
<tr><td>4</td><td colspan="2">Start the SnapManager for Microsoft SQL Server application.</td></tr>
<tr><td>5</td><td colspan="2">Ensure that all other Windows Explorer windows are closed on the SQL Server computer running SnapManager.</td></tr>
<tr><td>6</td><td colspan="2">Disable any SnapManager operations that are scheduled to run against the SQL Server data you are restoring, including any jobs scheduled on remote management or remote verification servers.</td></tr>
<tr><td>7</td><td colspan="2">To launch the SnapManager Restore wizard, go to the Actions pane and click Restore Wizard.
The SnapManager Restore wizard appears and displays the Welcome screen.</td></tr>
<tr><td>8</td><td colspan="2">In the Welcome screen, click Next.
The SQL Server screen appears.</td></tr>
<tr><td>9</td><td colspan="2">In the SQL Server screen, do the following:
1. Select Restore SnapManager backups that were created on a different SQL Server.
2. Click Next.
The SQL Server Where the Backups were Created screen appears.</td></tr>
</table>

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td>10</td><td colspan="2">In the SQL Server Where the Backups were Created screen, do the following:
1. Select the SQL Server whose backup sets you want to use to restore databases to this SQL Server.
2. In the SnapInfo Directory Path box, enter or browse to the name of the SnapInfo directory for those backup sets.
3. Leave the Use this server's SnapInfo directory check box cleared.
4. Click Next.
The Backup Set screen appears and lists the backup sets for the other SQL Server you specified.</td></tr>
<tr><td rowspan="3">11</td><td colspan="2">You can choose to restore to a point-in-time or marked transaction:</td></tr>
<tr><td>To restore to a point-in-time</td><td>Go to Step 12.</td></tr>
<tr><td>To restore to a marked transaction</td><td>Go to Step 13.</td></tr>
<tr><td>12</td><td colspan="2">Use the Point-in-time option in the Transaction Logs screen to perform an up-to-the-minute restore operation or a point-in-time restore operation:
• For an up-to-the-minute restore, backup the most recent transactions and select them for restore by selecting Most recent backup selected.
• For a point-in-time restore, select the backup set, a combination of transaction log backups to be restored, or both, and select Committed transactions at the specified time.
For detailed information, follow the steps in "Restoring using the SnapManager Restore option" in Performing a restore operation on page 167.
For more information, see the following topics:
• For information about identifying the most recent full database backup, see "SnapManager backup set names" in How SnapManager backup data is organized on page 99.
• For information about identifying the transaction logs, see "SnapInfo subdirectory names" in How SnapManager backup data is organized on page 99 and "Transaction log backup" in Types of backup operations performed using SnapManager on page 103.
The procedure is now complete.</td></tr>
<tr><td>13</td><td colspan="2">Use the Marked Transaction option in the Transaction Logs screen to restore to a marked transaction.
For detailed information, follow the steps in "Restoring using the SnapManager Restore option" in Performing a restore operation on page 167.
The procedure is now complete.</td></tr>
</table>

Step	Action
If you did not attach the database	
14	If you cannot attach the database, then, to reduce the loss of data, ensure that the last active transaction log of the database is automatically backed up by SnapManager Restore: • See Microsoft KB article 253817, "HOW TO: Back up the Last Transaction Log When the Master and the Database Files Are Damaged." This article describes how you can back up the currently active transaction log even if the SQL Server database file is damaged, provided that the transaction log file is still accessible. • Use this same Microsoft KB article as a general guide for gaining access to the last active transaction log of the database. While referring to the steps in that article, observe the following key points: ◦ When you create a similar database that contains the same number of data and transaction log files as the original database, you are creating the database you will be restoring using SnapManager. ◦ Instead of using the SQL Server `Backup Log` command to back up the transaction log (as described in the Microsoft article), proceed to the next step in this procedure. • For information about identifying the transaction logs, see "SnapInfo subdirectory names" in *How SnapManager backup data is organized* on page 99 and "Transaction log backup" in *Types of backup operations performed using SnapManager* on page 103. **Attention:** Do not use SQL Server Management Studio to back up the last active transaction log. Due to file formatting differences between SnapManager backup and Management Studio backups, SnapManager cannot perform a restore operation using transaction logs that were backed up using Management Studio.
15	Start the SnapManager for Microsoft SQL Server application.
16	Make sure that all other Windows Explorer windows are closed on the SQL Server computer running SnapManager.
17	Disable any SnapManager operations that are scheduled to run against the SQL Server data you are restoring, including any jobs scheduled on remote management or remote verification servers.
18	To launch the SnapManager Restore wizard, go to the Actions pane and select **Restore Wizard**. The SnapManager Restore wizard appears and displays the Welcome screen.
19	In the Welcome screen, click **Next**. The SQL Server screen appears.

Step	Action
20	In the SQL Server screen, do the following: **1.** Select **Restore SnapManager backups that were created on a different SQL Server**. **2.** Click **Next**. The SQL Server Where the Backups were Created screen appears.
21	In the SQL Server Where the Backups were Created screen, do the following: **1.** Select the SQL Server whose backup sets you want to use to restore databases to this SQL Server. **2.** In the SnapInfo Directory Path box, enter or browse to the name of the SnapInfo directory for those backup sets. **3.** Leave the **Use this server's SnapInfo directory** check box cleared. **4.** Click **Next**. The Backup Set screen appears and lists the backup sets for the other SQL Server you specified.
22	Use the Point-in-time option in the Transaction Logs screen to perform an up-to-the-minute restore operation or a point-in-time restore operation: • For an up-to-the-minute restore, backup the most recent transactions and select them for restore by selecting **Most recent backup selected**. • For a point-in-time restore, select the backup set, a combination of transaction log backups to be restored, or both, and select **Committed transactions at the specified time**. For detailed information, follow the steps in "Restoring using the SnapManager Restore option" in *Performing a restore operation* on page 167. For more information, see the following topics: • For information about identifying the most recent full database backup, see "SnapManager backup set names" in *How SnapManager backup data is organized* on page 99. • For information about identifying the transaction logs, see "SnapInfo subdirectory names" in *How SnapManager backup data is organized* on page 99 and "Transaction log backup" in *Types of backup operations performed using SnapManager* on page 103. The procedure is now complete.

Restoring system databases from SnapManager backup sets

Prerequisites for restoring system databases

After the failure of your SQL Server system databases (distribution, master, model, and msdb databases), you can restore them from SnapManager backup sets for default and named SQL Server instances.

Before you can restore system databases from SnapManager backup sets:

1. The system databases must be migrated to LUNs. For more information, see *How databases are stored on storage system volumes* on page 62.
2. SnapManager must be used to create stream-based backup sets of those databases. For more information, see *Types of backup operations performed using SnapManager* on page 103.

Procedures for restoring system databases

The procedure to restore your SQL Server system database depends on whether the database is still functional. If the database is no longer functional, you must rebuild the system databases first.

Restoring system databases that are still functional If you are restoring system databases that are still functional, you only need to use SnapManager to restore the system databases from SnapManager backup sets.

For more information, see *Restoring databases using SnapManager* on page 160.

Restoring system databases that are no longer functional If you are restoring system databases because they are no longer functional, you must first rebuild the system databases using an SQL Server utility:

- For Microsoft SQL Server 2005 and SQL Server 2008, use the `setup.exe` utility to rebuild the system databases. This utility is located in the directory `C:\Program Files\Microsoft SQL Server\90\Setup Bootstrap`.
  For more information, see your Microsoft SQL Server documentation.

To rebuild and then restore SQL Server system databases from SnapManager backup sets, complete the following steps.

Step	Action
1	Create a new LUN on the same drive letter as the original LUN.
2	Use `Rebuildm.exe` or `setup.exe` to create base system databases. For more information, see your SQL Server documentation. **Result** The system databases are created in the default location.

Step	Action
3	Migrate the system databases from the default location back to the LUN. For more information, see *Understanding control-file based configuration* on page 65.
4	Use SnapManager to restore the system databases from SnapManager backup sets. For more information, see the following topics: • *Understanding SnapManager Restore* on page 160. • *Types of SnapManager restore operations* on page 163. • *Restoring using the SnapManager Restore option* in *Performing a restore operation* on page 167.

SnapManager command-line reference

Guidelines for using the command-line utility

Location of the SnapManager PowerShell

To launch SnapManager PowerShell, go to `Start > Programs > NetApp > SnapManager PowerShell`.

PowerShell syntax for backup operations performed in the GUI

When you use the SnapManager GUI to back up a database, SnapManager logs the PowerShell syntax to the Backup report. You might use this information to create scripts or troubleshoot issues.

Common parameters used

The following are the ubiquitous (common) parameters in PowerShell:

Debug (-db) This parameter displays the debug information for the cmdlet used.

ErrorAction Action Preference (-ea) Scripting blocks use this parameter. The following are the examples that explain the usage of this parameter.

- `SilentlyContinue:` Continue without printing.
- `Continue:` Print and then continue (This is the default setting.)
- `Stop:` Halt the command or script.
- `Inquire`: Ask the user what to do.

ErrorVariable (-ev) This parameter displays the error data in the specified variable.

OutVariable (-ov) This parameter displays the output data string.

OutBuffer (-ob) This parameter displays the output buffer.

Whatif This parameter gives you a preview of an operation.

Confirm This parameter prompts you for confirmation before the actual deletion operation starts.

Verbose (-vb) This parameter displays the report content for backup, restore, configuration, and verification options

Tips for using the command-line interface

Observe the following guidelines when using the SnapManager command-line functionality:

- All parameters and options are case-insensitive. For example, if you use the option `-Daily`, it achieves the same results as you get if you use `daily`.
- Some of the options must be invoked in a particular order. For best results, use the order specified in the syntax for all options.
- When a parameter value string contains spaces, be sure to enclose it in double quotes. For example, use `"First Backup Set"` rather than `First Backup Set`.
- Press Ctrl-D to cancel a running operation. Closing the PowerShell window does not cancel the running operation.

If the execution policies in your system are restricted, you might be unable to load the PowerShell snap-in. To check and reset the execution policies on your system, follow these steps:

Step	Action
1	Enter the command `get-executionpolicy` in PowerShell.

2	If the policy displayed is "Allsigned" or "Restricted", enter any of the following commands:
	`set-executionpolicy unrestricted`
	or
	`set-executionpolicy remotesigned`

clone-backup

Name

clone-backup

Synopsis

Use this cmdlet to clone databases from an existing backup or archive using the SnapManager SQL Server PowerShell command-line interface. You can also use this cmdlet to add (by cloning) a database to an Availability Group.

Syntax

```
clone-backup [-Server <String>] [-UserName <String>] [-Password <String>]
[-ServerInstance <String[]>] -Database <String[]> [-Backup <String>] [-
RestoreLastBackup <Int32>] [-TransLogsToApply <Int32[]>] [-ForceRestore
[<Boolean>]] [-ClusterAware] [-TargetDatabase <String[]>] [-
TargetServerInstance <String[]>] [-TargetServerMountPointDir <String>] [-
PointInTime <String[]>] [-SnapInfoDirectory <String>] [-MarkName
<String[]>] [-MarkTime <String[]>] [-RestoreBeforeMark [<Boolean>]] [-
RecoverDatabase <Boolean[]>] [-StandbyPath <String>] [-apicontext] [-
```

```
RestoreArchivedBackup] [-SnapVaultSecondary] [-CloneOnMirrorDestination] [-
ChangeClonePath] [-CloneMirrorDestVolumes <String[]>] [-PreCommand] [-
PreCommandPath <String>] [-PreCommandArguments <String>] [-PreCommandHost
<String>] [-PreCommandErrors <EnumHandleCmdError[]>] [-PostCommand] [-
PostCommandPath <String>] [-PostCommandArguments <String>] [-
PostCommandHost <String>] [-PostCommandErrors <EnumHandleCmdError[]>] [-
AvailabilityGroup] [-IgnoreRepLogs] [-WhatIf] [-Confirm]
[<CommonParameters>]
```

Description

You can use this cmdlet to clone a live database or a database that is already backed up in a backup set. This cmdlet restores the database from the existing backup set, to clone the database to an alternate temporary writable LUN location, or to an Availability Group for further use.

You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL server on which the SQL server instances reside. SnapManager takes the local computer name as the default server name.

Using this parameter, you can also specify a particular SQL server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

-UserName <String> - Short Form: -usr

This parameter denotes the SQL Server account name. If the login name is not specified, SnapManager uses Windows NT Authentication.

-Password <String> - Short Form: -pwd

This parameter is the SQL Server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-ServerInstance <String[]> - Short Form: -inst

This parameter specifies the SQL Server instance where the database is backed up originally. SnapManager takes the local computer name as the default server instance.

If multiple databases reside on the same LUN but are owned by different SQL server instances when you backed them up originally, use the following format:

```
-Inst "SQLServerInstance1","SQLServerInstance2"
```

The first database specified in the -Database parameter refers the first server instance in the -ServerInstance parameter, the second database in the -Database parameter refers to the second server instance in the -ServerInstance parameter and so on.

-Database <String[]> - Short Form: -d

Use this option to specify the databases that need to be cloned. Use a comma-separated list of strings:

```
-d Database 1, Database 2, Database 3, Database 4,....
```

Multiple database names should be specified only if those databases share a single LUN or multiple LUNs together. For a multiple database restore, all the selected databases should be present in the selected Snapshot copy.

You cannot restore a database with a new name if you specify multiple databases. If you want to restore with a new name, restore those databases one by one. In case of restore to alternate location, specify only one database name.

-Backup <String> - Short Form: -bkup

Use this option to specify the name of the backup set. This is a mandatory parameter. The following example illustrates the usage:

-bkup sqlsnap__SYMNASQLDEV170_04-11-2007_15.22.27

-RestoreLastBackup <Int32> - Short Form: -lastBkup

Use this parameter to restore backups without specifying the name. If you try to use the `Backup` and `RestoreLastBackup` parameters together, SnapManager ignores the `RestoreLastBackup` parameter and uses the `Backup` parameter during restore operation. A typical usage example of the `restorelastbackup` parameter is as follows:

```
restore-backup -restorelastbackup 1 -backup <backup name>
```

Note: If the value for RestoreLastBackup parameter is 0, SnapManager restores the latest backup. If the value is 1, SnapManager restores second-to-latest backup and so on.

-TransLogsToApply <Int32[]> - Short Form: -translogs

This parameter specifies the list of transactions logs that need to be applied. SnapManager applies all transaction logs of the databases specified in the -Database parameter by default. You can specify the number of transaction logs to be applied for every database mentioned in the -Database parameter. The list of number of transaction logs that have to applied has to be listed in the same sequence as the databases listed in the -Database parameter. For example,

```
restore-backup -svr MACHINE1\INST1 -database db1,db2 -TransLogsToApply 3,7
```

-ForceRestore [<Boolean>] - Short Form: -force

Use this parameter to force the restore of a database based on its state. SnapManager sets its value to "true" by default.

-ClusterAware - Short Form: -cl

Use this parameter to specify that the cmdlet runs solely on the active node in a cluster environment.

-TargetDatabase <String[]> - Short Form: -tgDb

Use this parameter to restore a database with a new name. The following example illustrates the usage:

```
-tgDb "NewDatabaseName1"," NewDatabaseName2"," NewDatabaseName3"
```

The parameter defines the new database name to which the original database is restored. The old database name is defined at the same position in the -Database parameter.

If no new database name is given, the database is restored to the original database name the database had during backup. If this original name already exists, the name is modified to: originalDbName__clone, or originalDbName__mount.

-TargetServerInstance <String[]> - Short Form: -tgInst

This parameter specifies the name of the new SQL server if you want to restore the database to a new SQL Server. SnapManager takes the source SQL server instance as the default.

-TargetServerMountPointDir <String> - Short Form: -tgmpdir

Use this parameter to specify the mount point path or directory of the target server instance in which the backups are cloned or mounted.

-PointInTime <String[]> - Short Form: -pit

Use this switch to restore databases until a specific point in time. The format for the point-in-time string is yyyy-mm-ddThh:mm:ss, with time specified in a 24-hour format.

In case of multiple databases you should specify the point-in-time values for every database separated by a comma. The number of values after the parameter name should equal the number of databases selected. The first value will be applied to the first database specified after the -Database parameter, the second value to the second database, and so on. The following example illustrates the usage:

```
-pit 2008-10-22T11:50:00, 2008-11-25T22:50:00
```

Note: The parameter correspondence is one-to-one, that is, the first point-in-time parameter value specified after the parameter -pit is applied to the first database specified in the parameter -Database and the second point-in-time parameter value to second database and so on. The values should conform to the required PointInTime regular expression.

-SnapInfoDirectory <String> - Short Form: -snapinfo

Use this parameter to specify the SnapInfo directory path of the archived backup set.

-MarkName <String[]> - Short Form: -mark

This parameter indicates the marked transaction at which to stop the transaction log recovery.

-MarkTime <String[]> - Short Form: -mktm

This parameter specifies a unique timestamp to guarantee the uniqueness of the input restored mark.

-RestoreBeforeMark [<Boolean>] - Short Form: -beforemk

This true or false value indicates whether the specified marked transaction log should be included in the restore.

-RecoverDatabase <Boolean[]> - Short Form: -recoverdb

This parameter indicates whether the database fully recovered or left in a partially recovered state after the cmdlet finishes, to facilitate future SQL transaction log restores. This is an array of booleans, so it must match the same number of elements of the -database array. If the it does not match the number of elements of the -database array, an error is given. This defaults to $true for all databases unless the -standbyPath is given, in which case it defaults to $false for all databases.

-StandbyPath <String> - Short Form: -standby

This parameter indicates the path to the standby recovery file where incomplete transactions are stored after restoring a full database and its transaction logs. There is no default if you specify this parameter. The path must be to the standby directory if more than one database shares a LUN. If the database is on a dedicated LUN, then it must be a specific file. If the -standbypath parameter is given, the -RecoveryDatabase given must be -RecoverDatabase $False, otherwise it defaults to $false for all databases if no _RecoverDatabase parameter is specified.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-RestoreArchivedBackup - Short Form: -rstarchbkup

Use this parameter to specify using remote backup to clone the database.

-SnapVaultSecondary - Short Form: -vaultsec

This optional parameter identifies the backup vault from which you want to clone a database. If you do not specify this parameter, SnapManager chooses one of the backup vaults. You use this parameter in conjunction with the `-RestoreArchivedBackup` parameter. If you specify this parameter with the `-AvailabilityGroup` parameter, then the Availability Group databases must be spread across the same volumes. Otherwise, do not specify this parameter and SnapManager will choose one of the backup vaults. This parameter applies to clustered Data ONTAP only.

The syntax for this parameter is as follows:

```
-SnapVaultSecondary n, Vserver:volume
```

Where n is the number of Storage Virtual Machine (SVM, formerly known as Vserver):volume pairs.

Example: `-SnapVaultSecondary 3, Vserver1:volume1, Vserver2:volume2, Vserver3:volume3`

-CloneOnMirrorDestination - Short Form: -cloneonmir

This parameter indicates to clone a database based on the Snapshot copy on the SnapMirror destination volume. Ensure that the SnapMirror relationship exists and SnapMirror was updated when using this option.

-ChangeClonePath (Boolean Parameter) - Short Form: -chgpath

Use this parameter to change clone database paths based on the new database clone name.

-CloneMirrorDestVolumes <String[]> - Short Form: -clonemir

Use this parameter to specify cloning using the Snapshot copy on the SnapMirror destination volume.

-PreCommand <String> - Short Form: -precmd

This parameter indicates to run a command before the current operation.

Note: You cannot have more than one space between items that may be parsed in this parameter's value.

-PreCommandPath <String> - Short Form: -precmdpath

This parameter specifies the operating system path to the command to be run before the SnapManager operation starts.

-PreCommandArguments <String> - Short Form: -precmdargs

This parameter contains a list of strings of SnapManager operation-specific information or user-defined arguments to be passed to the program or script. The default is to pass no parameters to the script. If the parameter contains white spaces (tabs or spaces) you need to enclose it in double quotes. This parameter is processed only if the parameters -PreCommand and -PreCommandPath are specified.

-PreCommandHost <String> - Short Form: -precmdhost

This parameter specifies the host machine name on which the command is run before the operation starts. The default is to run on the current machine. This parameter is considered only if the parameters -PreCommand and -PreCommandPath are specified.

-PreCommandErrors <EnumHandleCmdError[]> - Short Form: -precmnderrors

This parameter specifies how to handle errors on the pre-command. The ContinueOnError value (the default) indicates that the SnapManager operation executes even if an error is detected during the pre-command launch. The StopOnPreCmdError value indicates that if a pre-command script gets an error, the remaining SMSQL operation is not attempted. This parameter is considered only if the parameters -PreCommand and -PreCommandPath are specified.

-PostCommand - Short Form: -postcmd

This parameter indicates to run a command after the current operation is complete.

Note: You cannot have more than one space between items that may be parsed in this parameter's value.

-PostCommandPath <String> - Short Form: -postcmdpath

Use this parameter to specify the operating system path to the command to be run after the SnapManager operation starts.

-PostCommandArguments <String> - Short Form: -postcmdargs

This parameter contains a list of strings of SnapManager operation-specific information or user-defined arguments to be passed to the program or script. The default is to pass no parameters to the script. If the parameter contains white spaces (tabs or spaces) you enclose it in double quotes. This parameter is processed only if the parameters -PostCommand and -PostCommandPath are specified.

-PostCommandHost <String> - Short Form: -postcmdhost

This parameter specifies the host machine name on which the command is run after the operation is complete. The default is to run on the current machine. This parameter is considered only if the parameters -PostCommand and -PostCommandPath are specified.

-PostCommandErrors <EnumHandleCmdError[]> - Short Form: -postcmderrors

This parameter specifies how to handle SMSQL operation errors on the post-command run. The ContinueOnError value (the default) indicates that the SMSQL operation executes even if an error is detected during the post-command launch. The StopOnPostCmdError value indicates that if a post-command script gets an error, the remaining SMSQL operation is not attempted. This parameter is considered only if the parameters -PostCommand and -PostCommandPath are specified.

-AvailabilityGroup <String> - Short Form: -ag

Use this parameter to reseed databases belonging to the given Availability group.

-IgnoreRepLogs: - Short Form: -nosharelogs

Use this parameter to ignore the transaction logbackups from SnapManager Repository Share.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual operation starts.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1: `clone-backup -Server win-225-165 -Database DB2 -Inst win-225-165 -Backup sqlsnap__win-225-165_09-06-2008_13.44.51`

This command creates a clone of the specified backup.

Example 2: `clone-backup -Server win-225-165 -Database DB2 -Inst win-225-165 -RestoreLastBackup 0`

This command restores the most recent clone that was created.

Example 3: `clone-backup -Server win-225-165 -Inst win-225-165 -AvailabilityGroup Ag1 -RestoreLastBackup 0`

This command restores the most recent clone of the Availability Group that was created.

Example 4: `clone-backup -svr win-225-165 -Database DB2 -Inst win-225-165 -Backup sqlsnap__win-225-165_09-06-2008_13.44.51 -RestoreArchivedBackup -vaultsec 2,vserver1:volume1,vserver2:volume2`

This command creates a clone of database DB2 from the specified SnapVault secondary volume.

clone-database

Name

clone-database

Synopsis

This cmdlet enables you to clone a live database or a database that is already backed up in a backup set using the SnapManager SQL Server PowerShell command-line interface.

Syntax

```
clone-database [-Server <String>] [-UserName <String>] [-Password <String>]
[-LogBkup] [-Verify] [-VerifyServerInstance <String>] [-VerSvrLogin
<String>] [-VerSvrPassword <String>] [-VerDestVolume] [-VerifyOnDestVolumes
<String[]>] [-DBCCOption <EnumDbccOption[]>] [-CloneOnMirrorDestination] [-
ChangeClonePath] [-Resynchronize] [-ForceTerminateConnection] [-
ClusterAware] [-CloneMirrorDestVolumes <String[]>] [-VerifyDisable] [-
UseMountPoint] [-MountPointDir <String>] [-UseDriveAvailable] [-
RetainBackups <Int32>] [-RetainBackupDays <Single>] [-AttachDB] [-
UpdateMirror] [-NoRetainUTM] [-ManagementGroup <String>] [-LogBkupOnly] [-
BkupSIF] [-RetainSnapofSnapInfo <Int32>] [-RetainSnapofSnapInfoDays
<Single>] [-TruncateSqlLog [<Boolean>]] [-TruncateLogs] [-PreCommand] [-
PreCommandPath <String>] [-PreCommandArguments <String>] [-PreCommandHost
<String>] [-PreCommandErrors <EnumHandleCmdError[]>] [-PostCommand] [-
PostCommandPath <String>] [-PostCommandArguments <String>] [-
PostCommandHost <String>] [-PostCommandErrors <EnumHandleCmdError[]>] [-
RunDBCCAfter] [-RunDBCCBefore] [-GenericNaming] [-ArchiveBackup] [-
VerifyArchiveBackup] [-ArchivedBackupRetention <String>] [-ServerInstance
<String[]>] -Database <String[]> [-TransLogsToApply <Int32[]>] [-
ForceRestore [<Boolean>]] [-TargetDatabase <String[]>] [-
TargetServerInstance <String[]>] [-TargetServerMountPointDir <String>] [-
MarkName <String[]>] [-MarkTime <String[]>] [-RestoreBeforeMark
```

```
[<Boolean>]] [-RecoverDatabase <Boolean[]>] [-StandbyPath <String>] [-
apicontext] [-RestoreArchivedBackup] [-RetainShareBackups] [-
RetainShareBackupDays] [-AvailabilityGroup] [-IgnoreRepLogs] [-WhatIf] [-
Confirm] [<CommonParameters>]
```

Description

This cmdlet enables you to clone a live database or a database that is already backed up in a backup set. It creates a backup set of the database and uses the backup set to clone the database. This cmdlet provides you various verification options, DBCC, recovery after restore, retaining backups, management groups and many other options.

You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short form: -svr

This parameter denotes the name of the host SQL Server on which the SQL Server instances reside. SnapManager takes the local computer name as the default server name.

Using this parameter, you can also specify a particular SQL server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

-Username <String> - Short form: -usr

This parameter denotes the SQL Server account name. If the login name is not specified, SnapManager uses Windows NT Authentication.

-Password <String> - Short form: -pwd

This parameter is the SQL Server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-Logbkup - Short form: -lb

Use this option to specify that the transaction logs also need to be backed up after a full backup.

-Verify - Short form: -ver

Use this parameter if you wish to verify the backed up databases and logs.

-VerifyServerInstance <String> - Short form: -verInst

This parameter specifies the separate SQL server that is used to run the Database Consistency Check (DBCC) utility. If you have not specified the -verify parameter, SnapManager ignores this parameter.

The following example illustrates the usage:

```
-verInst win-225-161
```

Here the SQL server instance is the local or remote SQL server instance to verify on. SnapManager takes the configured SQL server instance that is used for verify in client configuration (registry) as the default SQL server instance.

-VerSvrLogin <String> - Short form: -verlogin

This parameter specifies that SQL Server authentication is used. If not specified, the default Windows NT Authentication mechanism is used.

-VerSvrPassword <String> - Short form: -verpwd

This parameter is used to input the verification server password. SnapManager ignores this parameter if the parameter -VerSvrLogin is not specified.

-VerDestVolume - Short form: -verdest

Use this parameter to verify the database on the SnapMirror destination volume. SnapManager sets it to "false" by default.

-VerifyOnDestVolumes <String[]> - Short form: -vermirror

Specify this parameter in order to override the default SnapMirror relationships. Enter the source and destination storage systems and volumes as a comma-separated list. SnapManager sets it to "false" by default.

-DBCCOption <EnumDbccOption[]> - Short form: -dbccopt

This parameter specifies options to the DBCC SQL command that are used to validate and verify the database that is being processed. When you use this parameter, you are explicitly requesting DBCC options, and the system does read the registry to determine the default DBCC options. The security access issues for the registry are bypassed when you use this cmdlet option. The parameter uses the following values:

NOOPTION

NOINDEX

ALL_ERRORMSGS

NO_INFOMSGS (default)

TABLOCK

PHYSICAL_ONLY (default)

For more information about these options, see your Microsoft SQL Server documentation.

-CloneOnMirrorDestination - Short form: -cloneonmir

Use this parameter to clone a database based on the Snapshot copy on the SnapMirror destination volume. Ensure that the SnapMirror relationship exists and SnapMirror was updated when using this option.

-ChangeClonePath - Short form: -chgpath

Use this parameter to change clone database paths based on the new database clone name.

-Resynchronize - Short form: -resync

Use this parameter to specify that the existing clone is refreshed with the live database.

-ForceTerminateConnection - Short form: -ftc

Use this parameter to specify that all the connections to the existing clone are terminated during clone resynchronize.

-ClusterAware - Short form: -cl

Use this parameter to specify that the cmdlet runs solely on the active node in a cluster environment.

-CloneMirrorDestVolumes <String[]> - Short form: -clonemir

Use this parameter to specify cloning using the Snapshot copy on the SnapMirror destination volume.

-VerifyDisable - Short form: -verDis

This parameter overrides verification and can disable verification even if the database was not verified after backup.

-UseMountPoint - Short form: -mp

This parameter specifies that the Snapshot copy must be mounted to an NTFS directory.

During a SnapManager verification operation, Snapshot copies are mounted to the default NTFS directory for database verification. The option is effective when there are no available drive letters to mount the Snapshot copies. It overrides pre-configured SnapManager verification settings.

-MountPointDir <String> - Short form: -mpdir

Use this parameter to specify the mount point directory on which a backup set will be mounted during database verification. Use this parameter with the parameter -UseMountPoint.

-UseDriveAvailable - Short form: -drvavail

Use this parameter to indicate that you should use available drive letter as mount point on which a backup set is mounted during database verification.

-RetainBackups <Int32> - Short form: -tgInst

Use this parameter to specify the number of backups to be retained after the delete operation.

-RetainBackupDays <Single> - Short form: -rtdays

Use this parameter to specify the number of days you want to retain the backups for. SnapManager deletes backups older than the specified number of days. The parameters RetainBackups and RetainBackupDays are mutually exclusive and cannot be specified together.

-AttachDB - Short form: -attdb

If the operation includes a database or transaction log verification, use this option when you want to specify that the databases are to be attached after the verification operation completes.

-UpdateMirror - Short form: -updmir

Use this option to update the SnapMirror destination after a backup or verification operation ends, if the operation uses backups that reside on volumes configured as SnapMirror sources.

-NoRetainUTM - Short form: -noutm

Use this option if you do not want to retain up-to-the-minute restore ability for older backups in other management groups.

-ManagementGroup <String> - Short form: -mgmt

This parameter denotes the backup or verify operation that SnapManager performs on daily, or weekly, or standard basis. The default management group is standard.

-LogBkupOnly - Short form: -lgbkonly

Use this option to back up your SQL Server transaction log files only. No full snapshot backup will be done.

-BkupSIF - Short form: -bksif

Use this option to create a Snapshot copy of the SnapInfo directory after the backup of the transaction log completes. The backup type should be a transaction log backup only.

-RetainSnapofSnapInfo <Int32> - Short form: -rtsifsnap

Use this option if you want to delete the oldest Snapshot copies in the SnapInfo directory, specified that the backup type is a transaction log backup only. It has an integer value. The following example illustrates the usage of this parameter: -rtsifsnap Number of SnapInfo Snapshots to keep

Note: This option is valid only if you specify the parameter - BkupSIF.

-RetainSnapofSnapInfoDays <Single> - Short form: -rtsifsnapdays

Use this parameter to delete SnapInfo Snapshot copies older than the specified number of days. This parameter is mutually exclusive with the parameter RetainSnapofSnapinfo and they cannot be specified together in the same cmdlet.

-TruncateSqlLog [<Boolean>] - Short form: -truncLog

This parameter specifies whether to truncate the SQL transaction logs. SQL transaction logs are truncated by default. Valid values are $true or $false. This parameter only works if -LogBkup or -LogBkupOnly are true.

-TruncateLogs - Short form: -trlog

This obsolete parameter (now replaced by TruncateSqlLog) specifies whether to truncate the SQL transaction logs. SQL transaction logs are not truncated by default. This parameter only works if -LogBkup or -LogBkupOnly are true. In SMSQL 5.2 and later, if neither -TruncateLogs or -TruncateSqlLog is specified, the default behavior is to truncate the logs.

-PreCommand <String> - Short form: -precmd

This parameter indicates to run a command before the current operation.

Note: You cannot have more than one space between items that may be parsed in this parameter's value.

-PreCommandPath <String> - Short form: -precmdpath

This parameter specifies the operating system path to the command to be run before the SnapManager operation starts.

-PreCommandArguments <String> - Short form: -precmdargs

This parameter contains a list of strings of SnapManager operation-specific information or user-defined arguments to be passed to the program or script. The default is to pass no parameters to the script. If the parameter contains white spaces (tabs or spaces) you need to enclose it in double quotes. This parameter is processed only if the parameters -PreCommand and -PreCommandPath are specified.

-PreCommandHost <String> - Short form: -precmdhost

This parameter specifies the host machine name on which the command is run before the operation starts. The default is to run on the current machine. This parameter is considered only if the parameters -PreCommand and -PreCommandPath are specified.

-PreCommandErrors <EnumHandleCmdError[]> - Short form: -precmnderrors

This parameter specifies how to handle errors on the pre-command. The ContinueOnError value (the default) indicates that the SMSQL operation executes even if an error is detected during the pre-command launch. The StopOnPreCmdError value indicates that if a pre-command script gets an error, the remaining SMSQL operation is not attempted. This parameter is considered only if the parameters -PreCommand and -PreCommandPath are specified.

-PostCommand - Short form: -postcmd

This parameter indicates to run a command after the current operation is complete.

Note: You cannot have more than one space between items that may be parsed in this parameter's value.

-PostCommandPath <String> - Short form: -postcmdpath

This parameter specifies the operation system path for the command to be run after the SMSQL operation is complete.

-PostCommandArguments <String> - Short form: -postcmdargs

This parameter contains a list of strings of SnapManager operation-specific information or user-defined arguments to be passed to the program or script. The default is to pass no parameters to the script. If the parameter contains white spaces (tabs or spaces) you enclose it in double quotes. This parameter is processed only if the parameters -PostCommand and -PostCommandPath are specified.

-PostCommandHost <String> - Short form: -postcmdhost

This parameter specifies the host machine name on which the command is run after the operation is complete. The default is to run on the current machine. This parameter is considered only if the parameters -PostCommand and -PostCommandPath are specified.

-PostCommandErrors <EnumHandleCmdError[]> - Short form: -postcmderrors

This parameter specifies how to handle SMSQL operation errors on the post-command run. The ContinueOnError value (the default) indicates that the SMSQL operation executes even if an error is detected during the post-command launch. The StopOnPostCmdError value indicates that if a post-command script gets an error, the remaining SMSQL operation is not attempted. This parameter is considered only if the parameters -PostCommand and -PostCommandPath are specified.

-RunDBCCAfter - Short form: -dbccaf

If the operation includes a database backup, use this parameter if you want to verify the live database after the backups are performed.

-RunDBCCBefore - Short form: -dbccbf

If the operation includes a database backup, use this parameter if you want to verify the live database before the backups are performed.

-GenericNaming - Short form: -gen

This parameter specifies that the backups must follow the Generic backup naming convention.

-ArchiveBackup - Short form: -arch

Use this parameter to archive database to a secondary storage system during the backup phase of the operation.

-VerifyArchiveBackup - Short form: -verarch

Use this parameter to verify database archived at the secondary storage system.

-ArchivedBackupRetention <String> - Short form: -archret

Use this parameter to specify whether you want to retain backups at the archived location on a daily, hourly, weekly, monthly or unlimited basis.

-ServerInstance <String[]> - Short form: -inst

This parameter specifies the SQL server instance where the database is backed up originally. SnapManager takes the local computer name as the default server instance.

You can specify multiple server instance names here as a comma-separated list. If multiple databases reside on the same LUN but are owned by different SQL server instances when you backed them up originally, use the following format:

```
-Inst "SQLServerInstance1","SQLServerInstance2"
```

The first database specified in the -Database parameter refers the first server instance in the -ServerInstance parameter, the second database in the -Database parameter refers to the second server instance in the -ServerInstance parameter and so on.

-Database <String[]> - Short form: -d

Use this option to specify the databases that need to be cloned. Use a comma-separated list of strings:

```
-d Database 1, Database 2, Database 3, Database 4,....
```

Multiple database names should be specified only if those databases share a single LUN or multiple LUNs together. For a multiple database restore, all the selected databases should be present in the selected Snapshot copy.

You cannot restore a database with a new name if you specify multiple databases. If you want to restore with a new name, restore those databases one by one. In case of restore to alternate location, specify only one database name.

-TransLogsToApply <Int32[]> - Short form: -translogs

This parameter specifies the count of transactions logs that need to be applied to each database restored. If the TransLogsToApply parameter is not given, then all transaction logs that apply to the full backup restored are applied by default (just as the GUI does). You can specify the number of transaction logs to be applied for every database mentioned in the -Database parameter. The list of number of transaction logs that are applied must be listed in the same sequence as the databases listed in the -Database parameter. For example:

-Database db1,db2

might correspond to:

-TransLogsToApply 1,8

which means 1 transaction log backup will be applied to db1, and 8 will be applied to db2.

-ForceRestore [<Boolean>] - Short form: -force

Use this parameter to force the restore of a database based on its state. SnapManager sets it's value to "true" by default.

-TargetDatabase <String[]> - Short form: -tgDb

Use this parameter to restore a database with a new name. The following example illustrates the usage:

```
-tgDb "NewDatabaseName1"," NewDatabaseName2"," NewDatabaseName3"
```

The parameter defines the new database name to which the original database is restored. The old database name is defined at the same position in the -Database parameter.

If no new database name is given, the database is restored to the original database name the database had during backup. If this original name already exists, the name is modified to: originalDbName__clone, or originalDbName__mount.

-TargetServerInstance <String[]> - Short form: -tgInst

This parameter specifies the name of the new SQL server if you want to restore the database to a new SQL server. SnapManager takes the source SQL server instance as the default.

-TargetServerMountPointDir <String> - Short form: -tgmpdir

Use this parameter to specify the mount point path or directory of the target server instance in which the databases are to be cloned or mounted.

-MarkName <String[]> - Short form: -mark

This parameter indicates the marked transaction at which to stop the transaction log recovery.

-MarkTime <String[]> - Short form: -mktm

This parameter specifies a unique timestamp to guarantee the uniqueness of the input restored mark.

-RestoreBeforeMark [<Boolean>] - Short form: -beforemk

This true or false value indicates whether the specified marked transaction log should be included in the restore.

-RecoverDatabase <Boolean[]> - Short form: -recoverdb

This parameter indicates whether the database fully recovered or left in a partially recovered state after the cmdlet finishes, to facilitate future SQL transaction log restores. This is an array of booleans, so it must match the same number of elements of the -database array. If the it does not match the number of elements of the -database array, an error is given. This defaults to $true for all databases unless the -standbyPath is given, in which case it defaults to $false for all databases.

-StandbyPath <String> - Short form: -standby

This parameter indicates the path to the standby recovery file where incomplete transactions are stored after restoring a full database and its transaction logs. There is no default if you specify this parameter. The path must be to the standby directory if more than one database shares a LUN. If the database is on a dedicated LUN, then it must be a specific file. If the -standbypath parameter is given, the -RecoveryDatabase given must be -RecoverDatabase $False, otherwise it defaults to $false for all databases if no -RecoverDatabase parameter is specified.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-RestoreArchivedBackup - Short form: -rstarchbkup

Use this parameter to specify using remote backup to perform the clone operation.

-RetainShareBackups <Integer> - Short form: -rtsharebackups

Use this parameter to specify the number of log backups retained in the SnapManager for SQL repository share.

-RetainShareBackupDays <Integer> - Short form: -rtsharedays

Use this parameter to specify for how many days log backups are retained in the SnapManager Repository Share.

-AvailabilityGroup <String> - Short form: -ag

Use this parameter to reseed databases belonging to the given Availability group.

-IgnoreRepLogs: - Short form: -nosharelogs

Use this parameter to ignore the transaction logbackups from SnapManager Repository Share.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual operation starts.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1: `clone-database -svr sql1 -Database "Db1"`

This command clones database Db1 located on SQL Server sql1.

Example 2: `clone-database -svr win-225-166 -Inst win-225-166 -Database dbtest1 -Verify -verinst win-225-166 -RecoverDatabase`

This example enables database cloning with a default name for a default instance.

Example 3: `clone-database -svr win-225-166 -Inst win-225-166 -Database dbtest1 -Verify -verinst win-225-166 -TargetDatabase dbtest1_Clone -RecoverDatabase`

This example enables database cloning with a new name for a default instance.

Example 4: `clone-database -svr win-225-166 -Inst win-225-166\Named -Database dbtest2 -Verify -verinst win-225-166 -RecoverDatabase`

This example enables database cloning with a default name for a named instance.

Example 5: `clone-database -svr win-225-166 -Inst win-225-166\Named -Database dbtest2 -Verify -verinst win-225-166 -TargetDatabase dbtest2_Clone -RecoverDatabase`

This example enables database cloning with a new name for a named instance.

Example 6: `clone-database -svr 'SNAPMGR-19' -inst 'SNAPMGR-19', 'SNAPMGR-19', 'SNAPMGR-19' -d 'DB3', 'DB4', 'DB5' -tgInst 'SNAPMGR-19' -tgDb 'DB3__Clone', 'DB4__Clone', 'DB5__Clone' -tgmpdir 'E:\Program Files \NetApp\SnapManager for SQL Server\SnapMgrMountPoint' -ClusterAware -Resynchronize -ForceTerminateConnection -RetainBackups 3 -lb -mgmt standard`

This example creates a new backup on database "DB3," "DB4," and "DB5" and refreshes the cloned databases on the active node.

Example 7: `clone-database -svr 'venudhar-2k8vm2' -inst 'venudhar-2k8vm2\heitz' -ag 'testag'`

This command clones all the databases belonging to the specified Availability group.

clone-replica

Name

clone-replica

Synopsis

Use this cmdlet to create an Availability Group replica by cloning existing Availability Group databases to a specified server, which then becomes a secondary.

Syntax

```
clone-replica [-Server <String>] [-UserName <String>] [-Password <String>]
[-LogBkup] [-Verify] [-VerifyServerInstance <String>] [-VerSvrLogin
<String>] [-VerSvrPassword <String>] [-VerDestVolume] [-VerifyOnDestVolumes
<String[]>] [-DBCCOption <EnumDbccOption[]>] [-CloneOnMirrorDestination] [-
ChangeClonePath] [-Resynchronize] [-ForceTerminateConnection] [-
ClusterAware] [-CloneMirrorDestVolumes <String[]>] [-VerifyDisable] [-
UseMountPoint] [-MountPointDir <String>] [-UseDriveAvailable] [-
RetainBackups <Int32>] [-RetainBackupDays <Single>] [-AttachDB] [-
UpdateMirror] [-NoRetainUTM] [-ManagementGroup <String>] [-LogBkupOnly] [-
BkupSIF] [-RetainSnapofSnapInfo <Int32>] [-RetainSnapofSnapInfoDays
<Single>] [-TruncateSqlLog [<Boolean>]] [-TruncateLogs] [-PreCommand] [-
PreCommandPath <String>] [-PreCommandArguments <String>] [-PreCommandHost
<String>] [-PreCommandErrors <EnumHandleCmdError[]>] [-PostCommand] [-
PostCommandPath <String>] [-PostCommandArguments <String>] [-
PostCommandHost <String>] [-PostCommandErrors <EnumHandleCmdError[]>] [-
RunDBCCAfter] [-RunDBCCBefore] [-GenericNaming] [-ArchiveBackup] [-
VerifyArchiveBackup] [-ArchivedBackupRetention <String>] [-ServerInstance
<String[]>] -Database <String[]> [-TransLogsToApply <Int32[]>] [-
ForceRestore [<Boolean>]] [-TargetDatabase <String[]>] [-
TargetServerInstance <String[]>] [-TargetServerMountPointDir <String>] [-
MarkName <String[]>] [-MarkTime <String[]>] [-RestoreBeforeMark
[<Boolean>]] [-RecoverDatabase <Boolean[]>] [-StandbyPath <String>] [-
apicontext] [-RestoreArchivedBackup] [-WhatIf] [-Confirm] -
AvailabilityGroup [-SynchronousCommit] [-FailoverMode] [-ReadableSecondary]
[<CommonParameters>]
```

Description

The cmdlet uses Snapshot technology to quickly replicate databases to a remote cluster SQL instance, and then groups them in an Availability Group. The replicated databases are associated with instances in the same cluster so that Availability Group failover can take place when required or requested.

An Availability Group supports up to three synchronous commit replicas and up to two automatic failover replicas.

You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short form: -svr

This parameter denotes the name of the host SQL Server on which the SQL Server instances reside. SnapManager takes the local computer name as the default server name.

Using this parameter, you can also specify a particular SQL server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

-Username <String> - Short form: -usr

This parameter denotes the SQL Server account name. If the login name is not specified, SnapManager uses Windows NT Authentication.

-Password <String> - Short form: -pwd

This parameter is the SQL Server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-Logbkup - Short form: -lb

Use this option to specify that the transaction logs also need to be backed up after a full backup.

-Verify - Short form: -ver

Use this parameter if you wish to verify the backed up databases and logs.

-VerifyServerInstance <String> - Short form: -verInst

This parameter specifies the separate SQL server that is used to run the Database Consistency Check (DBCC) utility. If you have not specified the -verify parameter, SnapManager ignores this parameter.

The following example illustrates the usage:

```
-verInst win-225-161
```

Here the SQL server instance is the local or remote SQL server instance to verify on. SnapManager takes the configured SQL server instance that is used for verify in client configuration (registry) as the default SQL server instance.

-VerSvrLogin <String> - Short form: -verlogin

This parameter specifies that SQL Server authentication is used. If not specified, the default Windows NT Authentication mechanism is used.

-VerSvrPassword <String> - Short form: -verpwd

This parameter is used to input the verification server password. SnapManager ignores this parameter if the parameter -VerSvrLogin is not specified.

-VerDestVolume - Short form: -verdest

Use this parameter to verify the database on the SnapMirror destination volume. SnapManager sets it to "false" by default.

-VerifyOnDestVolumes <String[]> - Short form: -vermirror

Specify this parameter in order to override the default SnapMirror relationships. Enter the source and destination storage systems and volumes as a comma-separated list. SnapManager sets it to "false" by default.

-DBCCOption <EnumDbccOption[]> - Short form: -dbccopt

This parameter specifies options to the DBCC SQL command that are used to validate and verify the database that is being processed. When you use this parameter, you are explicitly requesting DBCC options, and the system does read the registry to determine the default DBCC options. The security access issues for the registry are bypassed when you use this cmdlet option. The parameter uses the following values:

NOOPTION

NOINDEX

ALL_ERRORMSGS

NO_INFOMSGS (default)

TABLOCK

PHYSICAL_ONLY (default)

For more information about these options, see your Microsoft SQL Server documentation.

-CloneOnMirrorDestination - Short form: -cloneonmir

Use this parameter to clone a database based on the Snapshot copy on the SnapMirror destination volume. Ensure that the SnapMirror relationship exists and SnapMirror was updated when using this option.

-ChangeClonePath - Short form: -chgpath

Use this parameter to change clone database paths based on the new database clone name.

-Resynchronize - Short form: -resync

Use this parameter to specify that the existing clone is refreshed with the live database.

-ForceTerminateConnection - Short form: -ftc

Use this parameter to specify that all the connections to the existing clone are terminated during clone resynchronize.

-ClusterAware - Short form: -cl

Use this parameter to specify that the cmdlet runs solely on the active node in a cluster environment.

-CloneMirrorDestVolumes <String[]> - Short form: -clonemir

Use this parameter to specify cloning using the Snapshot copy on the SnapMirror destination volume.

-VerifyDisable - Short form: -verDis

This parameter overrides verification and can disable verification even if the database was not verified after backup.

-UseMountPoint - Short form: -mp

This parameter specifies that the Snapshot copy must be mounted to an NTFS directory.

During a SnapManager verification operation, Snapshot copies are mounted to the default NTFS directory for database verification. The option is effective when there are no available drive letters to mount the Snapshot copies. It overrides pre-configured SnapManager verification settings.

-MountPointDir <String> - Short form: -mpdir

Use this parameter to specify the mount point directory on which a backup set will be mounted during database verification. Use this parameter with the parameter -UseMountPoint.

-UseDriveAvailable - Short form: -drvavail

Use this parameter to indicate that you should use available drive letter as mount point on which a backup set is mounted during database verification.

-RetainBackups <Int32> - Short form: -tgInst

Use this parameter to specify the number of backups to be retained after the delete operation.

-RetainBackupDays <Single> - Short form: -rtdays

Use this parameter to specify the number of days you want to retain the backups for. SnapManager deletes backups older than the specified number of days. The parameters RetainBackups and RetainBackupDays are mutually exclusive and cannot be specified together.

-AttachDB - Short form: -attdb

If the operation includes a database or transaction log verification, use this option when you want to specify that the databases are to be attached after the verification operation completes.

-UpdateMirror - Short form: -updmir

Use this option to update the SnapMirror destination after a backup or verification operation ends, if the operation uses backups that reside on volumes configured as SnapMirror sources.

-NoRetainUTM - Short form: -noutm

Use this option if you do not want to retain up-to-the-minute restore ability for older backups in other management groups.

-ManagementGroup <String> - Short form: -mgmt

This parameter denotes the backup or verify operation that SnapManager performs on daily, or weekly, or standard basis. The default management group is standard.

-LogBkupOnly - Short form: -lgbkonly

Use this option to back up your SQL Server transaction log files only. No full snapshot backup will be done.

-BkupSIF - Short form: -bksif

Use this option to create a Snapshot copy of the SnapInfo directory after the backup of the transaction log completes. The backup type should be a transaction log backup only.

-RetainSnapofSnapInfo <Int32> - Short form: -rtsifsnap

Use this option if you want to delete the oldest Snapshot copies in the SnapInfo directory, specified that the backup type is a transaction log backup only. It has an integer value. The following example illustrates the usage of this parameter: -rtsifsnap Number of SnapInfo Snapshots to keep

Note: This option is valid only if you specify the parameter - BkupSIF.

-RetainSnapofSnapInfoDays <Single> - Short form: -rtsifsnapdays

Use this parameter to delete SnapInfo Snapshot copies older than the specified number of days. This parameter is mutually exclusive with the parameter RetainSnapofSnapinfo and they cannot be specified together in the same cmdlet.

-TruncateSqlLog [<Boolean>] - Short form: -truncLog

This parameter specifies whether to truncate the SQL transaction logs. SQL transaction logs are truncated by default. Valid values are $true or $false. This parameter only works if -LogBkup or -LogBkupOnly are true.

-TruncateLogs - Short form: -trlog

This obsolete parameter (now replaced by TruncateSqlLog) specifies whether to truncate the SQL transaction logs. SQL transaction logs are not truncated by default. This parameter only works if -LogBkup or -LogBkupOnly are true. In SMSQL 5.2 and later, if neither -TruncateLogs or -TruncateSqlLog is specified, the default behavior is to truncate the logs.

-PreCommand <String> - Short form: -precmd

This parameter indicates to run a command before the current operation.

Note: You cannot have more than one space between items that may be parsed in this parameter's value.

-PreCommandPath <String> - Short form: -precmdpath

This parameter specifies the operating system path to the command to be run before the SnapManager operation starts.

-PreCommandArguments <String> - Short form: -precmdargs

This parameter contains a list of strings of SnapManager operation-specific information or user-defined arguments to be passed to the program or script. The default is to pass no parameters to the script. If the parameter contains white spaces (tabs or spaces) you need to enclose it in double quotes. This parameter is processed only if the parameters -PreCommand and -PreCommandPath are specified.

-PreCommandHost <String> - Short form: -precmdhost

This parameter specifies the host machine name on which the command is run before the operation starts. The default is to run on the current machine. This parameter is considered only if the parameters -PreCommand and -PreCommandPath are specified.

-PreCommandErrors <EnumHandleCmdError[]> - Short form: -precmnderrors

This parameter specifies how to handle errors on the pre-command. The ContinueOnError value (the default) indicates that the SMSQL operation executes even if an error is detected during the pre-command launch. The StopOnPreCmdError value indicates that if a pre-command script gets an error, the remaining SMSQL operation is not attempted. This parameter is considered only if the parameters -PreCommand and -PreCommandPath are specified.

-PostCommand - Short form: -postcmd

This parameter indicates to run a command after the current operation is complete.

Note: You cannot have more than one space between items that may be parsed in this parameter's value.

-PostCommandPath <String> - Short form: -postcmdpath

This parameter specifies the operation system path for the command to be run after the SMSQL operation is complete.

-PostCommandArguments <String> - Short form: -postcmdargs

This parameter contains a list of strings of SnapManager operation-specific information or user-defined arguments to be passed to the program or script. The default is to pass no parameters to the script. If the parameter contains white spaces (tabs or spaces) you enclose it in double quotes. This parameter is processed only if the parameters -PostCommand and -PostCommandPath are specified.

-PostCommandHost <String> - Short form: -postcmdhost

This parameter specifies the host machine name on which the command is run after the operation is complete. The default is to run on the current machine. This parameter is considered only if the parameters -PostCommand and -PostCommandPath are specified.

-PostCommandErrors <EnumHandleCmdError[]> - Short form: -postcmderrors

This parameter specifies how to handle SnapManager operation errors on the post-command run. The ContinueOnError value (the default) indicates that the SMSQL operation executes even if an error is detected during the post-command launch. The StopOnPostCmdError value indicates that if a post-command script gets an error, the remaining SMSQL operation is not attempted. This parameter is considered only if the parameters -PostCommand and -PostCommandPath are specified.

-RunDBCCAfter - Short form: -dbccaf

If the operation includes a database backup, use this parameter if you want to verify the live database after the backups are performed.

-RunDBCCBefore - Short form: -dbccbf

If the operation includes a database backup, use this parameter if you want to verify the live database before the backups are performed.

-GenericNaming - Short form: -gen

This parameter specifies that the backups must follow the Generic backup naming convention.

-ArchiveBackup - Short form: -arch

Use this parameter to archive database to a secondary storage system during the backup phase of the operation.

-VerifyArchiveBackup - Short form: -verarch

Use this parameter to verify database archived at the secondary storage system.

-ArchivedBackupRetention <String> - Short form: -archret

Use this parameter to specify whether you want to retain backups at the archived location on a daily, hourly, weekly, monthly or unlimited basis.

-ServerInstance <String[]> - Short form: -inst

This parameter specifies the SQL server instance where the database is backed up originally. SnapManager takes the local computer name as the default server instance.

You can specify multiple server instance names here as a comma-separated list. If multiple databases reside on the same LUN but are owned by different SQL server instances when you backed them up originally, use the following format:

```
-Inst "SQLServerInstance1","SQLServerInstance2"
```

The first database specified in the -Database parameter refers the first server instance in the -ServerInstance parameter, the second database in the -Database parameter refers to the second server instance in the -ServerInstance parameter and so on.

-Database <String[]> - Short form: -d

Use this option to specify the databases that need to be cloned. Use a comma-separated list of strings:

```
-d Database 1, Database 2, Database 3, Database 4,....
```

Multiple database names should be specified only if those databases share a single LUN or multiple LUNs together. For a multiple database restore, all the selected databases should be present in the selected Snapshot copy.

You cannot restore a database with a new name if you specify multiple databases. If you want to restore with a new name, restore those databases one by one. In case of restore to alternate location, specify only one database name.

-TransLogsToApply <Int32[]> - Short form: -translogs

This parameter specifies the count of transactions logs that need to be applied to each database restored. If the TransLogsToApply parameter is not given, then all transaction logs that apply to the full backup restored are applied by default (just as the GUI does). You can specify the number of transaction logs to be applied for every database mentioned in the -Database parameter. The list of number of transaction logs that are applied must be listed in the same sequence as the databases listed in the -Database parameter. For example:

-Database db1,db2

might correspond to:

-TransLogsToApply 1,8

which means 1 transaction log backup will be applied to db1, and 8 will be applied to db2.

-ForceRestore [<Boolean>] - Short form: -force

Use this parameter to force the restore of a database based on its state. SnapManager sets it's value to "true" by default.

-TargetDatabase <String[]> - Short form: -tgDb

Use this parameter to restore a database with a new name. The following example illustrates the usage:

```
-tgDb "NewDatabaseName1"," NewDatabaseName2"," NewDatabaseName3"
```

The parameter defines the new database name to which the original database is restored. The old database name is defined at the same position in the -Database parameter.

If no new database name is given, the database is restored to the original database name the database had during backup. If this original name already exists, the name is modified to: originalDbName__clone, or originalDbName__mount.

-TargetServerInstance <String[]> - Short form: -tgInst

This parameter specifies the name of the new SQL server if you want to restore the database to a new SQL server. SnapManager takes the source SQL server instance as the default.

-TargetServerMountPointDir <String> - Short form: -tgmpdir

Use this parameter to specify the mount point path or directory of the target server instance in which the databases are to be cloned or mounted.

-MarkName <String[]> - Short form: -mark

This parameter indicates the marked transaction at which to stop the transaction log recovery.

-MarkTime <String[]> - Short form: -mktm

This parameter specifies a unique timestamp to guarantee the uniqueness of the input restored mark.

-RestoreBeforeMark [<Boolean>] - Short form: -beforemk

This true or false value indicates whether the specified marked transaction log should be included in the restore.

-RecoverDatabase <Boolean[]> - Short form: -recoverdb

This parameter indicates whether the database will be fully recovered or left in a partially recovered state after the cmdlet finishes to facilitate future SQL transaction log restores. This is an array of booleans, so it must match the same number of elements of the -database array. If the it does not match the number of elements of the -database array, an error is given. This defaults to $true for all databases unless the -standbyPath is given, in which case it defaults to $false for all databases.

-StandbyPath <String> - Short form: -standby

This parameter indicates the path to the standby recover file where incomplete transactions are stored after restoring a full database and its transaction logs. There is no default if you specify this parameter. The path must be to the standby directory if more than one database shares a LUN. If the database is on a dedicated LUN, then it must be a specific file. If the -standbypath parameter is given, the -RecoveryDatabase given must be -RecoverDatabase $False, otherwise it defaults to $false for all databases if no -RecoverDatabase parameter is specified.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-RestoreArchivedBackup - Short form: -rstarchbkup

Use this parameter to specify using remote backup to perform the clone operation.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual operation starts.

-AvailabilityGroup <String> - Short form: -ag

This parameter specifies the name of the source Availability Group.

-SynchronousCommit - Short form: -syncCommit

This parameter specifies that replica databases are synchronized to their primary. If not specified, false is assumed.

-FailoverMode - Short form: -flMd

This parameter specifies that failover occur to the preferred replica, if the primary replica becomes unavailable. If not specified, false is assumed.

-ReadableSecondary - Short form: -readsec

This parameter specifies read-only access for all of the new secondary databases. If not specified, false is assumed.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1: `clone-replica -svr 'SQL2012HA2' -inst 'SQL2012HA2\INST2' -ag 'snapmgr2012 ' -tgInst 'SQL2012HA1\INST1'`

This command creates a secondary replica for the Availability Group "snapmgr2012" on the secondary "SQL2012HA1\INST1". Values for -SynchronousCommit, FailoverMode, and ReadableSecondary are not specified, so the default, false, is used.

Example 2: `clone-replica -svr 'SQL2012HA2' -inst 'SQL2012HA2\INST2' -ag 'snapmgr2012 ' -tgInst 'SQL2012HA1\INST1' -SychronousCommit -FailoverMode -ReadableSecondary`

This command creates a secondary replica for the Availability Group "snapmgr2012" on the secondary "SQL2012HA1\INST1". The replica is created with the properties synchronous commit, failover mode, and readable secondary set to true.

delete-backup

Name

delete-backup

Synopsis

This cmdlet enables you to delete the SnapManager backup sets using the SnapManager SQL Server PowerShell command-line interface.

Syntax

```
delete-backup [-Server <String>] [-UserName <String>] [-Password <String>]
[-ServerInstance <String>] -Database <String> -Backup <String> [-
apicontext] [-ArchiveBackup] [-SnapVaultSecondary] [-WhatIf] [-Confirm]
[<CommonParameters>]
```

Description

This cmdlet enables you to delete a database depending on the input criteria specified in the command-line interface. It deletes the specified backup set if it contains the specified database name.

You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL Server on which the SQL Server instances reside. SnapManager takes the local computer name as the default server name. In case of a clustered configuration, the virtual server name is the default server name.

Using this parameter, you can also specify a particular SQL Server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

-UserName <String> - Short Form: -usr

This parameter denotes the SQL Server account name. If the login name is not specified, SnapManager uses Windows NT Authentication.

-Password <String> - Short Form: -pwd

This parameter is the SQL Server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-ServerInstance <String> - Short Form: -inst

This parameter specifies the SQL Server instance where the database is backed up originally. SnapManager takes the local computer name as the default server instance.

-Database <String> - Short Form: -d

This is a mandatory parameter that specifies a database.

-Backup <String> - Short Form: -bkup

Use this parameter to specify the backup set that needs to be deleted. It is a mandatory parameter.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-ArchiveBackup - Short Form: -arcbk

Use this parameter to specify the archived backup set that needs to be deleted.

Note: This parameter is mandatory if you delete archived backup sets.

-SnapVaultSecondary - Short Form: -vaultsec

This optional parameter identifies the backup vault from which you want to delete the Snapshot copy. If you do not specify this parameter, all backups are deleted from the related backup vaults. You use this parameter in conjunction with the `-ArchiveBackup` parameter. This parameter applies to clustered Data ONTAP only. The syntax for this parameter is as follows:

```
-SnapVaultSecondary n, Vserver:volume
```

Where n is the number of Storage Virtual Machine (SVM, formerly known as Vserver):volume pairs.

Example: `-SnapVaultSecondary 3, Vserver1:volume1, Vserver2:volume2, Vserver3:volume3`

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual deletion operation starts.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see about_CommonParameters (http://technet.microsoft.com/library/hh847884.aspx).

Examples

Example 1: `delete-backup -d "Db1" -bk "Db1bkup"`

This command deletes the backup set Db1bkup where DB1 is the cloned database.

Example 2: `delete-backup -d "Db1" -bk "Db1bkup" -ArchiveBackup -vaultsec 2,sn_vserver_dev:test_volume_sec,sn_vserver_dev:test_volume_sec2`

This command deletes the backup set Db1bkup from the specified SnapVault secondary volume.

delete-clone

Name

delete-clone

Synopsis

This cmdlet enables you to delete a cloned database.

Syntax

```
delete-clone [-Server <String>] [-UserName <String>] [-Password <String>]
[-ServerInstance <String>] -Database <String[]> [-JobInstance <String>] [-
ResyncCloneJob <String>] [-ClusterAware] [-TerminateConnection] [-
apicontext] [-WhatIf] [-Confirm] [ <CommonParameters>]
```

Description

This cmdlet helps you delete a cloned database using the SnapManager PowerShell command-line interface. Before deleting a clone, make sure all connections to the cloned database are disconnected.

You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL Server on which the SQL Server instances reside. SnapManager takes the local computer name as the default server name.

Using this parameter, you can also specify a particular SQL server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

-UserName <String> - Short Form: -usr

This parameter denotes the SQL Server account name. If the login name is not specified, SnapManager uses Windows NT Authentication.

-Password <String> - Short Form: -pwd

This parameter is the SQL server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-ServerInstance <String> - Short Form: -inst

This parameter specifies the SQL server instance where the database is backed up originally. SnapManager takes the local computer name as the default server instance.

-Database <String[]> - Short Form: -d

This is a mandatory parameter that specifies the list of cloned databases to be deleted. Enter the cloned database names in a comma separated list.

-JobInstance <String> - Short Form: -jobinst

This parameter is followed by the name of the SQL Server instance on which the clone resync job is created.

-ResyncCloneJob <String> - Short Form: -rcjob

This parameter is followed by the name of the clone resync job for the specified cloned database.

-ClusterAware - Short Form: -cl

Use this parameter to specify that the cmdlet runs solely on the active node in a cluster environment.

-TerminateConnection - Short form: -terminate

Use this parameter to terminate open database connections.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual deletion operation starts.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1: `delete-clone -svr sql1 -d "Db1"`

This command deletes the clone Db1 on Server sql1.

Example 2:
```
delete-clone -svr 'SNAPMGR-25' -inst 'SNAPMGR-25' -d
'DB1__Clone', 'DB2__Clone' -ClusterAware -ResyncCloneJob
"CloneResync_VDISK_W_07-08-2011_13-02-29" -JobInstance "SNAPMGR-19\MARS"
```

This example deletes the clone of database "DB1" and "DB2" from SQL Server "SNAPMGR-25" and the corresponding clone refresh job "CloneResync_VDISK_W_07-08-2011_13-02-29" from SQL agent instance "SNAPMGR-19\MARS".

export-config

Name

export-config

Synopsis

This cmdlet enables you to export the existing configuration information of an SQL server to a control-file using SnapManager PowerShell command-line interface.

Syntax

```
export-config [-Server <String>] [-ControlFilePath <String>] [-Section
<String[]>] [-apicontext] [-exportobject] [-WhatIf] [-Confirm]
[<CommonParameters>]
```

Description

This cmdlet enables you to export the existing configuration information of an SQL server to a control-file using SnapManager PowerShell command-line interface.

You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL server on which the SQL server instances reside. SnapManager takes the local computer name as the default server name.

Using this parameter, you can also specify a particular SQL server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

-ControlFilePath <String> - Short Form: -config

This parameter specifies the name of the control-file and its path. SnapManager takes the current directory as the control-file path by default.

-Section <String[]> - Short Form: -sect

This parameter lists section names that are to be imported (separated by commas). If you do not specify any particular section, the default value of all sections is applied. The valid section names that can be applied are as follows: storage, notification, verification, report, backup, scheduledjob, runcommand, snapmirrorvolume, monitor, and clonejob.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-exportobject - Short form: none

Use this parameter is to publish the configuration information as objects either shown on the output screen or to be piped to another cmdlet. This facilitates easy communication with SMSPS. Without this parameter, the default behavior is to export configuration information as an .xml file.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual operation starts.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1 `export-config -Server win-225-166 -ControlFilePath "C:\Program Files\NetApp\SnapManager for SQL Server\SMSQLConfig_16July_test4.xml" -Section storage,notification`

This cmdlet exports all sections of the existing configuration and settings to the specified control-file.

get-backup

Name

get-backup

Synopsis

This cmdlet allows you to list the backup sets made by SnapManager for Microsoft SQL Server.

Syntax

```
get-backup [-Server <String>] [-BackupServer <String>] [-UserName <String>] [-Password <String>] [-ServerInstance <String>] [-Database <String>] [-SnapInfoDirectory <String>] [-apicontext] [-WhatIf] [-Confirm] [<CommonParameters>]
```

Description

This cmdlet enables you to list the backup sets of a particular database by specifying an SQL Server, an SQL Server instance, or a database set. You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL Server on which the SQL Server instances reside. SnapManager takes the local computer name as the default server name.

Using this parameter, you can also specify a particular SQL Server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

For virtual server instances, specify the virtual server name. For example:

```
get-backup -server <virtual_server> -ServerInstance <virtual_instance> -d
aa1
```

-BackupServer <String> - Short Form: -bksvr

Use this parameter to specify where the backup was originally created. Use the host name or cluster name where the SQL Server instance resides. This parameter cannot be an SQL Server instance name. This parameter is optional, and is mainly used for a restore backup created from a different server. For example, this parameter can be used for DR using SnapMirror. By default, the backup server is the server currently connected, specified by -Server parameter. For example:

```
-Server win2k8-248-137 -backupserver 'SQL2K8VI1' -inst 'SQL2K8VI1\DE1' -
TargetServerInstance win2k8-248-137
```

The server is connected to a new server where the restore will be performed. But the backup was originally created on 'SQL2K8VI1', and the instance was 'DE1'.

-Username <String> - Short Form: -usr

This parameter denotes the SQL Server account name. If the login name is not specified, SnapManager uses Windows NT Authentication.

-Password <String> - Short Form: -pwd

This parameter is the SQL Server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-ServerInstance <String> - Short Form: -inst

This parameter specifies the SQL Server instance where the database is backed up originally. SnapManager takes the local computer name as the default server instance. For named SQL Server instances, enter the instance in the following format: `HostName\InstanceName`

-Database <String> - Short Form: -d

This is a mandatory parameter that specifies the database. If you do not specify the database parameter, the cmdlet backs up all of the SQL Server instances that are peer instances of the SQL server in the `-Server` parameter.

-SnapInfoDirectory <String> - Short Form: -sif

This parameter enables you to list the system and user databases on a remote server.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual operation starts.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1 `get-backup -svr 'VM-VS-1' -inst vm-vs-1 -d 'ds_test7'`

This example retrieves the backed up database on a server instance of the specified server.

Example 2 `get-backup -svr snapmgr-62 -inst snapmgr-63\FEDERATED -snapinfo \ \172.17.233.163\ G$\SMSQL_SnapInf`

This example shows all the server and user databases on the remote server.

import-config

Name

import-config

Synopsis

This cmdlet enables you to import the configuration information from a SnapManager for SQL control-file using SnapManager PowerShell command-line interface.

Syntax

```
import-config [-Server <String>] [-ControlFilePath <String>] [-Section
<String[]>] [-ValidateAndApply] [-AllowLocal] [-UserName <String>] [-
Password <String>] [-ClusterAware] [-DBCCBefore [<Boolean>]] [-DBCCAfter
[<Boolean>]] [-DeleteOriginalDBFile [<Boolean>]] [-UpdateStatisticsTable
[<Boolean>]] [-apicontext] [-WhatIf] [-Confirm] [<CommonParameters>]
```

Description

This cmdlet enables you to import the configuration information from a SnapManager for SQL control-file using SnapManager PowerShell command-line interface. You can import sections like storage, notification, verification, report, backup, scheduled job, snapmirror volume and so on. You can also control DBCC integrity verification and update statistics table using this cmdlet.

You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL server on which the SQL server instances reside. SnapManager takes the local computer name as the default server name.

Using this parameter, you can also specify a particular SQL server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

-ControlFilePath <String> - Short Form: -config

This parameter specifies the name of the control-file and its path. SnapManager takes the current directory as the control-file path by default.

-Section <String[]> - Short Form: -sect

This parameter lists section names that are to be imported (separated by commas). If you do not specify any particular section, the default value of all sections is applied. The valid section names that can be applied are as follows: storage, notification, verification, report, backup, scheduledjob, runcommand, snapmirrorvolume, monitor, and clonejob.

-ValidateAndApply - Short Form: -apply

This parameter applies the imported storage and notification settings data to the current system after validation. If you specify this parameter and validation is successful the imported data will be applied. If you do not specify this parameter only validation occurs.

-AllowLocal - Short Form: -tolocal

This parameter specifies that the migration of databases to the local disk is permitted. Its value is set to "false" by default.

-UserName <String> - Short Form: -usr

This parameter denotes the SQL Server account name. If the login name is not specified, SnapManager uses Windows NT Authentication. This parameter is mandatory if you import a scheduled job.

-Password <String> - Short Form: -pwd

This parameter is the SQL server account password. SnapManager ignores this parameter if the parameter -UserName is not specified. This parameter is mandatory if you import a scheduled job.

-DBCCBefore [<Boolean>] - Short Form: -dbcc

This parameter runs the DBCC physical integrity verification before migration. Its value is set to "true" by default.

-DBCCAfter [<Boolean>] - Short Form: -dbcc2

This parameter runs the DBCC physical integrity verification after migration. Its value is set to "false" by default.

-DeleteOriginalDBFile [<Boolean>] - Short Form: -deletedbfile

This parameter deletes the copy of the migrated database at original location. Its value is set to "true" by default.

-UpdateStatisticsTable [<Boolean>] - Short Form: -updatestatistics

This parameter runs "Update statistics" on tables before detaching the databases. Its value is set to "true" by default.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual operation starts.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1: `import-config -server "sql1" -ControlFilePath "C:\Program Files \NetApp\SnapManager for SQL\SMSQLConfig_01_23_2007_23.10.20.xml" -Section backup`

This cmdlet validates the backup settings in the control-file. It does not apply the settings to the SQL server.

Example 2 `import-config -Server win-225-166 -Section storage,notification -ControlFilePath "C:\Program Files\NetApp\SnapManager for SQL Server \SMSQLConfig_16July_test4.xml" -ValidateAndApply -AllowLocal`

This cmdlet validates the imported storage and notification settings from control-file and applies it to the system.

new-backup

Name

new-backup

Synopsis

This cmdlet enables you to back up the SQL server databases in the SnapManager PowerShell command-line interface.

Syntax

```
new-backup [-Server <String>] [-UserName <String>] [-Password <String>] [-
Database <String[]>] [-FederatedGroups <String[]>] [-Mark <String>] [-
MarkDesc <String>] [-LogBkup] [-Verify] [-VerifyServerInstance <String>] [-
VerSvrLogin <String>] [-VerSvrPassword <String>] [-RetainBackups <Int32>]
[-RetainBackupDays <Single>] [-RetainUtmBackups <Int32>] [-RetainUtmDays
<Single>] [-UseMountPoint] [-MountPointDir <String>] [-UseDriveAvailable]
[-AttachDB] [-UpdateMirror] [-NoRetainUTM] [-VerDestVolume] [-
ManagementGroup <String>] [-LogBkupOnly] [-BkupSIF] [-RetainSnapofSnapInfo
<Int32>] [-RetainSnapofSnapInfoDays <Single>] [-TruncateSqlLog [<Boolean>]]
[-TruncateLogs] [-Command] [-RunCommand <String>] [-CommandArguments
<String>] [-CommandServer <String>] [-PreCommand] [-PreCommandPath
<String>] [-PreCommandArguments <String>] [-PreCommandHost <String>] [-
PreCommandErrors <EnumHandleCmdError[]>] [-PostCommand] [-PostCommandPath
<String>] [-PostCommandArguments <String>] [-PostCommandHost <String>] [-
PostCommandErrors <EnumHandleCmdError[]>] [-RunDBCCAfter] [-RunDBCCBefore]
[-DBCCOption <EnumDbccOption[]>] [-GenericNaming] [-GenericNamingAdvanced]
[-VerifyOnDestVolumes <String[]>] [-apicontext] [-ArchiveBackup] [-
VerifyArchiveBackup] [-ArchivedBackupRetention <String>] [-ClusterAware] [-
WhatIf] [-Confirm] [-AvailabilityGroup] [-BackupPriority] [-Primary] [-
Secondary] [-CopyOnly] [-PreferredBackupReplica] [-CopyOnlyLogBackup] [-
CopyLogBackupToShare] [-RetainShareBackups] [-RetainShareBackupDays]
[<CommonParameters>]
```

Description

This cmdlet enables you to begin the backup-only and backup-with-verify operations. SnapManager provides a separate cmdlet for verification. You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL Server on which the SQL server instances reside. SnapManager takes the local computer name as the default server name. If no default host exists, SnapManager attempts to use the following as the default:

- The VerifyServerInstance specified by the user
- The configured verification server for the current machine (in the registry) done in the configuration wizard, or backup verification settings
- The VerificationServerInstance from the SQL Server being backed up as the verification server
- The current machine

Using this parameter, you can also specify a particular SQL Server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

To back up all instances on a server that has a default instance, specify the following:

```
-server <server_name>
```

To back up all instances on a server that does not have a default instance, specify one of the named instances on the server in the following format:

```
-server <host\instance>
```

To back up all databases on specified instances, use the following format:

```
-server <SQL_server_name or host\instance> -d <host\instance>, 0
```

For example:

```
–server 'sql1' -d 'sql1\instance1', '0', 'sql1\instance2', '0'
```

-Username <String> - Short Form: -usr

This parameter denotes the SQL Server account name. If the login name is not specified, SnapManager uses Windows NT Authentication.

-Password <String> - Short Form: -pwd

This parameter is the SQL Server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-Database <String[]> - Short Form: -d

Use this parameter to specify the original database that you want to back up. You can also specify multiple database names, but only if the databases share a single LUN or multiple LUNs. In this case, list the databases followed by -Database in following format:

```
-database sql-server-instance, count-of-databases, "database1"," database2"
```

If you do not specify the database parameter explicitly, the cmdlet backs up all the databases from all the SQL Server instances in the host. If storage other than NetApp storage exists on your system, the cmdlet omits databases located on that storage. Databases incompletely configured or databases in incompatible states are omitted when not explicitly provided with this parameter.

-FederatedGroups <String[]> - Short Form: -g

This parameter specifies the original federated groups to back up. If you specify multiple federated groups, the items in the list are separated by commas. If you do not specify the FederatedGroups parameter, the cmdlet backs up only the databases specified in the Database parameter. If neither parameter is specified, the cmdlet backs up all SQL server instances that are peer instances of the SQL server in the -Server parameter.

-Mark <String> - Short Form: -m

Use this parameter to specify a mark name when backing up transaction logs. If you do not specify a name, the default mark name "snapmgr_sqlbackup_[timestamp]" is used.

-MarkDesc <String> - Short Form: -md

Use this parameter to specify a mark description when backing up transaction logs. If you do not specify a name, the default mark description "snapmanager sql backup mark generated at [timestamp]" is used.

-Logbkup - Short form: -lb

Use this option to specify that the transaction logs also need to be backed up after a full backup.

-Verify - Short form: -ver

Use this parameter to verify the backed up databases and logs.

-VerifyServerInstance <String> - Short form: -verInst

This parameter specifies the separate SQL server that is used to run the Database Consistency Check (DBCC) utility. If you have not specified the -verify parameter, SnapManager ignores this parameter.

The following example illustrates the usage:

```
-verInst win-225-161
```

In this example, the SQL server instance is the local or remote SQL server instance to verify on. SnapManager takes the configured SQL server instance that is used for verify in client configuration (registry) as the default SQL server instance.

-VerSvrLogin <String> - Short form: -verlogin

This parameter specifies that SQL Server authentication is used. If not specified, the default Windows NT Authentication mechanism is used.

-VerSvrPassword <String> - Short form: -verpwd

This parameter is used to input the verification server password. SnapManager ignores this parameter if the parameter -VerSvrLogin is not specified.

-RetainBackups <Int32> - Short Form: -rtbackups

Use this parameter to specify the number of backups to be retained after the delete operation.

-RetainBackupDays <Single> - Short Form: -rtdays

Use this parameter to specify the number of days you want to retain the backups. SnapManager deletes backups older than the specified number of days. The parameters RetainBackups and RetainBackupDays are mutually exclusive.

-RetainUTMBackups <Single> - Short Form: -rubackups

Specifies the number of the most recent backup copies to retain up-to-the-minute restore ability after a SnapManager backup operation.

-RetainUTMDays <Single> - Short Form: -rudays

Specifies the number of days that the backups created within the time period retain up-to-the-minute restore ability. The backups older than the specified number of days lose up-to-the-minute restore ability and are for point-in-time restore only. Use this option in conjunction with RetainUtmBackups. Absence of this option denotes a up-to-the-minute restore ability retain policy based on backup count.

-UseMountPoint - Short Form: -mp

This parameter is a switch that specifies that the Snapshot copy must be mounted to an NTFS directory. During a SnapManager verification operation, Snapshot copies are mounted to the default NTFS directory for database verification. The option is effective when there are no available drive letters to mount the Snapshot copies. It overrides preconfigured SnapManager verification settings.

-MountPointDir <String> - Short Form: -mpdir

Use this parameter to specify the mount point directory on which a backup set is mounted during database verification. This parameter should be used along with the parameter -UseMountPoint.

Note: This option is valid only if you specify the parameter -BkupSIF.

-UseDriveAvailable - Short Form: -drvavail

Use this parameter to specify the mount point with available drive on which a backup set is to be mounted during database verification.

-AttachDB - Short Form: -attdb

If the operation includes a database or transaction log verification, use this option when you want to specify that the databases are to be attached after the verification operation completes.

-UpdateMirror - Short Form: -updmir

Use this option to update the SnapMirror destination after the backup or verification operations are complete, if you are using backups that reside on volumes configured as SnapMirror sources.

-NoRetainUTM - Short Form: -noutm

Use this option if you do not want to retain up-to-the-minute restore ability for older backups in other management groups.

-VerDestVolume - Short Form: -verdest

Use this parameter to verify the database on the SnapMirror destination volume. SnapManager sets it to “false” by default.

-ManagementGroup <String> - Short form: -mgmt

This parameter denotes the backup or verify operation that SnapManager performs on a daily, weekly, or standard basis. The default management group is standard.

-LogBkupOnly - Short form: -lgbkonly

Use this option to back up your SQL Server transaction log files only. No full Snapshot copy is made.

-BkupSIF - Short form: -bksif

Use this option to create a Snapshot copy of the SnapInfo directory after the backup of the transaction log finishes. The backup type should be a transaction log backup only.

-RetainSnapofSnapInfo <Int32> - Short form: -rtsifsnap

Use this option if you want to delete the oldest Snapshot copies in the SnapInfo directory, specified that the backup type is a transaction log backup only. It has an integer value. The following example illustrates the use of this parameter: -rtsifsnap Number of SnapInfo Snapshots to keep

Note: This option is valid only if you specify the parameter - BkupSIF.

-RetainSnapofSnapInfoDays <Single> - Short form: -rtsifsnapdays

Use this parameter to delete SnapInfo Snapshot copies older than the specified number of days. This parameter is mutually exclusive with the parameter RetainSnapofSnapinfo.

-TruncateSqlLog [<Boolean>] - Short form: -truncLog

This parameter specifies whether to truncate the SQL transaction logs. SQL transaction logs are truncated by default. Valid values are $true or $false. This parameter is valid only if -LogBkup or -LogBkupOnly are true.

-TruncateLogs - Short form: -trlog

This obsolete parameter (now replaced by TruncateSqlLog) specifies whether to truncate the SQL transaction logs. SQL transaction logs are not truncated by default. This parameter is valid only if -LogBkup or -LogBkupOnly are true. In SMSQL 5.2 and later, if neither -TruncateLogs or -TruncateSqlLog is specified, the default behavior is to truncate the logs.

-Command - Short form: -cmd

This switch parameter runs a command after the backup or verify operation.

-RunCommand - Short form: -runcmd

This parameter runs the specified command after the SnapManager backup or verification operation is complete. It defines the complete path for the command to be run after the backup or verify operation is complete. There is no default.

-CommandArguments <String> - Short form: -cmdargs

This option contains the string of SnapManager operation-specific information to be passed to your program or script. It is considered only if Command and RunCommand are specified. There is no default.

-CommandServer <String> - Short form: -cmdsvr

This obsolete parameter (now replaced by PostCommandHost) was used to indicate the machine where the desired command should run after the operation is complete. The default was to run on the current machine. This was only considered if -command and -RunCommand were specified.

-PreCommand <String> - Short form: -precmd

This parameter indicates to run a command before the current operation.

-PreCommandPath <String> - Short form: -precmdpath

This parameter specifies the operating system path to the command to be run before the SnapManager operation starts.

-PreCommandArguments <String> - Short form: -precmdargs

Use this parameter to specify a list of strings of SnapManager operation-specific information or user-defined arguments to be passed to the program or script.

-PreCommandHost <String> - Short form: -precmdhost

Use this parameter to specify the host machine name on which the command is to be run before the operation starts.

-PreCommandErrors <EnumHandleCmdError[]> - Short form: -precmnderrors

Use this parameter to specify how to handle errors on the precommand and the following SMSQL operation. The ContinueOnError value indicates that the following SMSQL operation is to be executed anyway. The StopOnPreCmdError value indicates that if a precommand script results in an error, the remaining SMSQL operation is not attempted.

-PostCommand - Short form: -postcmd

Use this parameter to run a command after the current operation.

-PostCommandPath <String> - Short form: -postcmdpath

Use this parameter to specify the operating system path to the command to be run after the SMSQL operation starts.

-PostCommandArguments <String> - Short form: -postcmdargs

Use this parameter to specify a list of strings of SnapManager operation-specific information or user defined arguments to be passed to the program or script.

-PostCommandHost <String> - Short form: -postcmdhost

Use this parameter to specify the host name on which the command is to be run after the operation is complete.

-PostCommandErrors <EnumHandleCmdError[]> - Short form: -postcmderrors

Use this parameter to specify how to handle errors on the next postcommand run. The ContinueOnError value indicates that the following SMSQL operation is to be executed anyway. The StopOnPostCmdError value indicates that if a postcommand script results in an error, the remaining SMSQL operation is not attempted.

-RunDBCCAfter - Short form: -dbccaf

If the operation includes a database backup, use this parameter if you want to verify the live database after the backups are performed.

-RunDBCCBefore - Short form: -dbccbf

If the operation includes a database backup, use this parameter if you want to verify the live database before the backups are performed.

-DBCCOption <EnumDbccOption[]> - Short form: -dbccopt

This parameter specifies options to the DBCC SQL command that are used to validate and verify the database that is being processed. When you use this parameter, you are explicitly requesting DBCC options, and the system reads the registry to determine the default DBCC options. The security access issues for the registry are bypassed when you use this cmdlet option. The parameter uses the following values:

NOOPTION

NOINDEX

ALL_ERRORMSGS

NO_INFOMSGS (default)

TABLOCK

PHYSICAL_ONLY (default)

For more information about these options, see your Microsoft SQL Server documentation.

-GenericNaming - Short Form: -gen

This parameter sets the naming convention for new backups as generic.

-GenericNamingAdvanced - Short Form: -genadv

This parameter sets the naming convention for new backups as enhanced.

After you use or enable the enhanced naming convention, it is permanent. You should continue to use -genadv and not revert to using only -gen.

-VerifyOnDestVolumes <String[]> - Short form: -vermirror

Specify this parameter to override the default SnapMirror relationships. Enter a comma-separated list of the source storage system, the source volume, the destination storage system, and the destination volume.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-ArchiveBackup - Short Form: -arch

Use this parameter to archive the database to a secondary storage system.

-VerifyArchiveBackup - Short Form: -verarch

Use this parameter to verify the database archived at the secondary storage system.

-ArchivedBackupRetention <String> - Short Form: -archret

Use this parameter to specify whether you want to retain backups at the archived location on a daily, hourly, weekly, monthly, or unlimited basis.

-ClusterAware - Short form: cl

Use this parameter to specify that the cmdlet runs solely on the active node in a cluster environment.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual operation starts.

-AvailabilityGroup <String> - Short Form: -ag

Use this parameter to specify one or more names of Availability Groups to which this backup applies.

-BackupPriority <Integer,Integer> - Short Form: -bp

Use this parameter to specify a set of secondary Availability Groups on a cluster by specifying a range of backup priorities. The operation applies to all replicas with backup priorities in that range. The maximum priority must be in the range of 1 to 100. The minimum backup priority must be less than or equal to the maximum priority.

-Primary - Short form: -prm

If this parameter is defined, then the backup is only made on the primary replica. If BackupPriority is also defined, then the primary replica must also satisfy the BackupPriority values.

-Secondary - Short form: -sec

If this parameter is defined, then the backup is made on all secondary replicas. If BackupPriority is also defined, then the secondary replicas must also satisfy the BackupPriority values.

-CopyOnly - Short form: -cpyonly

If this parameter is defined, a full backup is made as a copy-only full backup.

-PreferredBackupReplica - Short form: -preferbkreplica

Use this parameter to specify that only the preferred backup replica is backed up. The preferred backup replica is set from the Availability Group properties in the SQL Server 2012 Management Studio.

-CopyOnlyLogBackup - Short form: -cpyonlylgBk

Use this parameter to specify that transaction log backups are made as copy-only log backups.

-CopyLogBackupToShare <EnumBackupToShareType[]> - Short Form: -cpylgbkshare

Use this parameter to specify which transaction log backups are copied to the predefined repository share. The possible values are one of NOTHING_TOSHARE, COPYLOG_TOSHARE, or COPYLOG_TOSHARE_AGONLY. The repository share is set by the SnapManager for SQL repository share option.

-RetainShareBackups <Integer> - Short Form: -rtsharebackups

Use this parameter to specify the number of log backups retained in the SnapManager for SQL repository share.

-RetainShareBackupDays <Integer> - Short Form: -rtsharedays

Use this parameter to specify for how many days log backups are retained in the SnapManager Repository Share.

If you specify -PreferredBackupReplica along with -Primary, -Secondary, or -BackupPriority, the -PreferredBackupReplica value is used, and the others are ignored.

<CommonParameters>

This cmdlet supports the common parameters Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1: `new-backup -Server 'DBServer1' -Verify - VerifyServerInstance 'Snapmgr-50'`

This command creates a backup of all databases on the host DBServer1 and verifies the backups using the remote server Snapmgr-50.

Example 2: `new-backup -svr 'VM-VS-1' -d 'VM-VS-1', '4', 'ds_test1', 'ds_test2', 'ds_test6', 'ds_test7' -ver -verInst 'ZEUS-VM1\VERSERVER' -rtbackups 7 -lb -bksif -rtsifsnap 8 -trlog -noutm -mgmt standard -ArchiveBackup -VerifyArchiveBackup -ArchivedBackupRetention daily`

This example illustrates the creation of a new backup with verification of local backups and archive backups.

Example 3: `new-backup -svr 'VM-VS-1' -d 'VM-VS-1', '2', 'model', 'sm_test' -ver -verInst 'ZEUS-VM1\VERSERVER' -rtbackups 7 -lb -bksif -rtsifsnap 8 -trlog -noutm -gen -mgmt standard`

This example creates a new backup with the generic naming convention.

Example 4: `new-backup -svr 'VM-VS-1' -d 'VM-VS-1', '2', 'model', 'sm_test' -ver -verInst 'ZEUS-VM1\VERSERVER' -rtbackups 7 -lb -bksif -rtsifsnap 8 -trlog -noutm -mgmt standard`

This example creates a new backup with the unique naming convention.

Example 5: `new-backup -Server 'SNAPMGR-63' -Database 'SNAPMGR-63\SQL63INSTANCE1', '2', 'master', 'testdb2', 'SNAPMGR-63\SQL63INSTANCE2', '1', 'testdb1', 'SNAPMGR-19\SQLINSTANCE', '3', 'testdb1', 'testdb2', 'testdb3'`

This example creates a new backup with the federated backup feature.

Example 6: `new-backup -Server 'SNAPMGR-63' -Database 'SNAPMGR-63\SQL63INSTANCE1', '2', 'testdb4', 'testdb5'-FederatedGroups 2, 'SNAPMGR-63\SQL63INSTANCE1', '1', 'testdb1', 'SNAPMGR-19\SQLINSTANCE', '1', 'testdb2', 3, 'SNAPMGR-63\SQL63INSTANCE1', '2', 'testdb2', testdb3', 'SNAPMGR-63\SQL63INSTANCE2', '1', 'testdb1', 'SNAPMGR-19\SQLINSTANCE', '2', 'testdb1', 'testdb3',1, 'SNAPMGR-63\SQL63INSTANCE3', 0`

This example creates backups on all replicas.

Example 7: `new-backup -svr 'SQL2012HA2' -ag snapmgr2012 -prm -sec -mgmt standard`

This example creates backups on all replicas, because the default is all replicas.

Example 8: `new-backup -svr 'SQL2012HA2' -ag snapmgr2012 -mgmt standard`

This example creates backups on all replicas with backup priorities within the range of 50 to 70, because the default is all replicas.

Example 9: `new-backup -svr 'SQL2012HA2' -ag snapmgr2012 -prm -sec -bp 50,70 -mgmt standard`

This example creates a backup of the preferred replica.

Example 10: `new-backup -svr 'SQL2012HA2' -ag snapmgr2012 -PreferredBackupReplica -mgmt standard`

reseed-backup

Name

reseed-backup

Synopsis

This command enables you to reseed databases from SnapManager backups.

Syntax

```
reseed-backup [-Server <String>] [-UserName <String>] [-Password <String>]
[-ServerInstance <String[]>] -Database <String[]> [-Backup <String>] [-
RestoreLastBackup <Int32>] [-VerifyServerInstance <String>] [-VerSvrLogin
<String>] [-VerSvrPassword <String>] [-VerifyDisable] [-DBCCOption
<EnumDbccOption[]>] [-apicontext] [-PreCommand] [-PreCommandPath <String>]
[-PreCommandArguments <String>] [-PreCommandHost <String>] [-
PreCommandErrors <EnumHandleCmdError[]>] [-PostCommand] [-PostCommandPath
<String>] [-PostCommandArguments <String>] [-PostCommandHost <String>] [-
PostCommandErrors <EnumHandleCmdError[]>] [-AvailabilityGroup] [-
RestoreArchivedBackup] [-SnapVaultSecondary] [-IgnoreRepLogs] [-WhatIf] [-
Confirm] [<CommonParameters>]
```

Description

This cmdlet enables you to reseed a secondary database or a secondary Availability group replica. It has many other options. You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL Server on which the SQL Server instances reside. SnapManager takes the local computer name as the default server name.

Using this parameter, you can also specify a particular SQL server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

-Username <String> - Short Form: -usr

UserName is the SQL Server account name. It is specified if the SQL Server computer is accessed using a different account from that used to access the production SQL Server. If not specified, the Windows NT Authentication user name is used.

-Password <String> - Short Form: -pwd

This parameter is the SQL Server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-ServerInstance <String[]> - Short Form: -inst

This parameter specifies the SQL Server instance on which the database is backed up originally. SnapManager takes the local computer name as the default server instance.

You can specify multiple server instance names in one list, separated by commas. If multiple databases reside on the same LUN but are owned by different SQL Server instances when you backed them up originally, use the following format:

```
-Inst "SQLServerInstance1","SQLServerInstance2"
```

The first database specified in the -Database parameter refers the first server instance in the -ServerInstance parameter, the second database in the -Database parameter refers to the second server instance in the -ServerInstance parameter, and so on.

-Database <String[]> - Short Form: -d

Use this parameter to specify the original database that you want to reseed. You can specify multiple database names using this option if the databases share a single LUN or multiple LUNs; also, the backups for multiple databases must all have the same name. Use the following format:

```
-Database "DatabaseName1"," DatabaseName2"
```

Note: All the databases selected should be present in the selected Snapshot copy.

-Backup <String> Short Form: -bkup

Use this option to specify the name of the backup set. The following example illustrates the usage:

```
-bkup sqlsnap__SYMNASQLDEV170_04-11-2007_15.22.27
```

-RestoreLastBackup <Int32> - Short Form: -lastBkup

Use this parameter to restore backups without specifying the name. If you try to use the Backup and RestoreLastBackup parameters together, SnapManager ignores the RestoreLastBackup parameter and uses the Backup parameter during restore operation. A typical usage example of the restorelastbackup parameter is as follows:

```
restore-backup -restorelastbackup 1
```

Note: If the value for RestoreLastBackup parameter is 0, SnapManager reseeds the latest backup. If the value is 1, SnapManager reseed the second-to-latest backup, and so on.

-VerifyServerInstance <String> - Short Form: -verInst

This parameter specifies the separate SQL Server that is used to run the Database Consistency Check (DBCC) utility. If you have not specified the -verify parameter, SnapManager ignores this parameter.

The following example illustrates the usage:

```
-verInst win-225-161
```

In this example, the SQL Server instance is the local or remote SQL Server instance to verify on. SnapManager takes the configured SQL Server instance that is used for verify in client configuration (registry) as the default SQL Server instance.

-VerSvrLogin <String> - Short Form: -verlogin

This parameter specifies that SQL Server authentication is used. If not specified, the default Windows NT Authentication mechanism is used.

-VerSvrPassword <String> - Short Form: -verpwd

SnapManager ignores this parameter if the parameter -VerSvrLogin is not specified.

-VerifyDisable - Short Form: -verDis

This parameter overrides verification and can disable verification even if the database was not verified after backup.

-DBCCOption <EnumDbccOption[]> - Short form: -dbccopt

This parameter specifies options to the DBCC SQL command that are used to validate and verify the database that is being processed. When you use this parameter, you are explicitly requesting DBCC options, and the system does read the registry to determine the default DBCC options. The security access issues for the registry are bypassed when you use this cmdlet option. The parameter uses the following values:

NOOPTION

NOINDEX

ALL_ERRORMSGS

NO_INFOMSGS (default)

TABLOCK

PHYSICAL_ONLY (default)

For more information about these options, see your Microsoft SQL Server documentation.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-PreCommand <String> - Short form: -precmd

This parameter indicates to run a command before the current operation.

-PreCommandPath <String> - Short form: -precmdpath

This parameter specifies the operating system path to the command to be run before the SnapManager operation starts.

-PreCommandArguments <String> - Short form: -precmdargs

Use this parameter to specify a list of strings of SnapManager operation-specific information or user defined arguments to be passed to the program or script.

-PreCommandHost <String> - Short form: -precmdhost

Use this parameter to specify the host machine name on which the command is to be run before the operation starts.

-PreCommandErrors <EnumHandleCmdError[]> - Short form: -precmnderrors

Use this parameter to specify how to handle errors on the precommand and the following SMSQL operation. The ContinueOnError value indicates that the following SMSQL operation are to be executed anyway. The StopOnPreCmdError value indicates that if a precommand script get an error, the remaining SMSQL operation is not attempted.

-PostCommand - Short form: -postcmd

Use this parameter to run a command after the current operation.

-PostCommandPath <String> - Short form: -postcmdpath

Use this parameter to specify the operating system path to the command to be run after the SMSQL operation starts.

-PostCommandArguments <String> - Short form: -postcmdargs

Use this parameter to specify a list of strings of SnapManager operation-specific information or user-defined arguments to be passed to the program or script.

-PostCommandHost <String> - Short form: -postcmdhost

Use this parameter to specify the name of the host on which the command is to be run after the operation is complete.

-PostCommandErrors <EnumHandleCmdError[]> - Short form: -postcmderrors

Use this parameter to specify how to handle errors on the next postcommand run. The ContinueOnError value indicates that the following SMSQL operation is executed anyway. StopOnPostCmdError value indicates that if a postcommand script results in an error, the remaining SMSQL operation is not attempted.

-AvailabilityGroup <String> - Short form: -ag

This parameter specifies the name of the Availability Group to which the databases belong.

-RestoreArchivedBackup - Short Form: -rstarchbkup

Use this parameter to specify using a remote backup to reseed the database.

-SnapVaultSecondary - Short Form: -vaultsec

This optional parameter identifies the backup vault from which you want to reseed a database. If you do not specify this parameter, SnapManager chooses one of the backup vaults. You use this parameter in conjunction with the -RestoreArchivedBackup parameter. If you specify this parameter with the -AvailabilityGroup parameter, then the Availability Group databases must be spread across the same volumes. Otherwise, do not specify this parameter and SnapManager chooses one of the backup vaults. This parameter applies to clustered Data ONTAP only.

The syntax for this parameter is as follows, where n is the number of Vserver:volume pairs:

```
-SnapVaultSecondary n, Vserver:volume
```

Example: `-SnapVaultSecondary 3, Vserver1:volume1, Vserver2:volume2, Vserver3:volume3`

-IgnoreRepLogs - Short form: -nosharelogs

This parameter specifies that the log backups from the SnapManager Repository Share should be ignored.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual deletion operation starts.

<CommonParameters>

This cmdlet supports the common parameters Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, and OutVariable. For more information, see about_CommonParameters (http://technet.microsoft.com/library/hh847884.aspx).

Examples

Example 1: `reseed-backup -svr venudhar-2k8vm2 -inst venudhar-2k8vm2 -ag tstag1 -backup sqlsnap__VENUDHAR-2K8VM2_08-26-2012_20.46.42`

This command reseeds Availability Group `tstag1`. Note that only unhealthy databases or databases that are already dropped in the given Availability Group are reseeded.

Example 2: `reseed-backup -svr venudhar-2k8vm2 -inst venudhar-2k8vm2 -d db1,db2,db3 -backup sqlsnap__VENUDHAR-2K8VM2_08-26-2012_20.46.42`

This example reseeds the specific databases db1, db2, and db3.

Example 3: `reseed-backup -svr 'venudhar-2k8vm2' -inst 'venudhar-2k8vm2\heitz' -ag 'testag' -restorelastbackup 0`

This example reseeds all databases that belong to the Availability Group.

Example 4: `reseed-backup -svr 'venudhar-2k8vm2' -inst 'venudhar-2k8vm2' -ag 'testag1' -backup 'sqlsnap__VENUDHAR-2K8VM2_08-26-2012_20.46.42' -RestoreArchivedBackup`

This example reseeds databases that belong to the specified Availability Group from a SnapVault secondary volume chosen by SnapManager. The command does not specify a SnapVault secondary volume because the Availability Group databases are not spread across the same volumes.

restore-backup

Name

restore-backup

Synopsis

This cmdlet enables you to restore databases from SnapManager backups.

Syntax

```
restore-backup [-Server <String>] [-BackupServer <String>] [-UserName
<String>] [-Password <String>] [-ServerInstance <String[]>] -Database
<String[]> [-Backup <String>] [-RestoreLastBackup <Int32>] [-
TransLogsToApply <Int32[]>] [-ForceRestore [<Boolean>]] [-
VerifyServerInstance <String>] [-VerSvrLogin <String>] [-VerSvrPassword
<String>] [-VerDestVolume] [-VerifyOnDestVolumes <String[]>] [-
VerifyDisable] [-DBCCOption <EnumDbccOption[]>] [-TargetDatabase
<String[]>] [-TargetLocation] <String[]>] [-TargetServerInstance
<String[]>] [-PointInTime <String[]>] [-RestoreArchive] [-
RestoreFromUnmanagedMedia] [-SnapInfoDirectory <String>] [-MarkName
<String[]>] [-MarkTime <String[]>] [-RestoreBeforeMark [<Boolean>]] [-
RecoverDatabase <Boolean[]>] [-StandbyPath <String>] [-apicontext] [-
RestoreArchivedBackup] [-SnapVaultSecondary] [-NoAccessToRemoteBackup] [-
ProxyServer <String>] [-PreCommand] [-PreCommandPath <String>] [-
PreCommandArguments <String>] [-PreCommandHost <String>] [-PreCommandErrors
<EnumHandleCmdError[]>] [-PostCommand] [-PostCommandPath <String>] [-
PostCommandArguments <String>] [-PostCommandHost <String>] [-
PostCommandErrors <EnumHandleCmdError[]>] [-AvailabilityGroup] [-
IgnoreRepLogs] [-WhatIf] [-Confirm] [<CommonParameters>]
```

Description

This cmdlet enables you to restore a database. It also gives point-in-time restore, verification, force restore and many other options.

You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL Server on which the SQL Server instances reside. SnapManager takes the local computer name as the default server name.

Using this parameter, you can also specify a particular SQL server instance. The following examples illustrate the usage:

-BackupServer <String> - Short Form: -bkupsvr

Use this parameter to specify where the backup was originally created. Use the host name or cluster name where the SQL Server instance resides. This parameter cannot be an SQL Server instance name. This parameter is optional, and is mainly used for a restore backup created from a different server. For example, this parameter can be used for DR using SnapMirror. By default, the backup server is the server currently connected, specified by `-Server` parameter. For example:

```
-Server win2k8-248-137 -backupserver 'SQL2K8VI1' -inst 'SQL2K8VI1\DE1' -
TargetServerInstance win2k8-248-137 -SnapInfoDirectory 'H:\SMSQL_SnapInfo'
```

The server is connected to a new server where the restore will be performed. But the backup was originally created on 'SQL2K8VI1', and the instance was 'DE1'. The `-SnapInfoDirectory` parameter is required when you specify this parameter.

-Username <String> - Short Form: -usr

UserName is the SQL Server account name. It is specified if the SQL Server computer is accessed using a different account from that used to access the production SQL Server. If not specified the Windows NT Authentication username will be taken.

-Password <String> - Short Form: -pwd

This parameter is the SQL Server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-ServerInstance <String[]> - Short Form: -inst

This parameter specifies the SQL Server instance where the database is backed up originally. SnapManager takes the local computer name as the default server instance.

You can specify multiple server instance names here as a comma-separated list. If multiple databases reside on the same LUN but are owned by different SQL Server instances when you backed them up originally, use the following format:

```
-Inst "SQLServerInstance1","SQLServerInstance2"
```

The first database specified in the -Database parameter refers the first server instance in the -ServerInstance parameter, the second database in the -Database parameter refers to the second server instance in the -ServerInstance parameter and so on.

-Database <String[]> - Short Form: -d

Use this parameter to specify the original database that you want to restore. You can also specify multiple database names only if the databases share a single LUN or multiple LUNs together. In this case, list the databases followed by -Database in the following format:

```
-Database "DatabaseName1"," DatabaseName2"
```

Note: All the databases selected should be present in the selected Snapshot copy. This is a required parameter.

-Backup <String> - Short Form: -bkup

Use this option to specify the name of the backup set. The following example illustrates the usage:

```
-bkup sqlsnap__SYMNASQLDEV170_04-11-2007_15.22.27
```

-RestoreLastBackup <Int32> - Short Form: -lastBkup

Use this parameter to restore backups without specifying the name. If you try to use the Backup and RestoreLastBackup parameters together, SnapManager ignores the RestoreLastBackup parameter and uses the Backup parameter during restore operation. A typical usage example of the restorelastbackup parameter is as follows:

```
restore-backup -restorelastbackup 1 -backup (backup name)
```

Note: If the value for RestoreLastBackup parameter is 0, SnapManager restores the latest backup. If the value is 1, SnapManager restores the second-to-latest backup and so on.

-TransLogsToApply <Int32[]> - Short Form: -translogs

This parameter specifies the list of transactions logs that need to be applied. SnapManager applies all transaction logs of the databases specified in the -Database parameter by default. You can specify the number of transaction logs to be applied for every database mentioned in the -Database parameter. The list of number of transaction logs that have to applied has to be listed in the same sequence as the databases listed in the -Database parameter. For example:

restore-backup -svr MACHINE1\INST1 -database db1,db2 -transLogsToApply 3,7

-ForceRestore [<Boolean>] - Short Form: -force

Use this parameter to force the restore of a database based on its state. SnapManager sets its value to "true" by default.

-VerifyServerInstance <String> - Short Form: -verInst

This parameter specifies the separate SQL Server that is used to run the Database Consistency Check (DBCC) utility. If you have not specified the -verify parameter, SnapManager ignores this parameter.

The following example illustrates the usage:

```
-verInst win-225-161
```

Here the SQL Server instance is the local or remote SQL Server instance to verify on. SnapManager takes the configured SQL Server instance that is used for verify in client configuration (registry) as the default SQL Server instance.

-VerSvrLogin <String> - Short Form: -verlogin

This parameter specifies that SQL Server authentication is used. If not specified, the default Windows NT Authentication mechanism is used.

-VerSvrPassword <String> - Short Form: -verpwd

This parameter is used to input the verification server password. SnapManager ignores this parameter if the parameter -VerSvrLogin is not specified.

-VerDestVolume - Short Form: -verdest

Use this parameter to verify the database on the SnapMirror destination volume. SnapManager sets it to "false" by default.

-VerifyOnDestVolumes <String[]> - Sort form: -verMirror

Specify this parameter to override the default SnapMirror relationships. Enter a comma-separated list of the source storage system, the source volume, the destination storage system, and the destination volume.

-VerifyDisable - Short Form: -verDis

This parameter overrides verification and can disable verification even if the database was not verified after backup.

-DBCCOption <EnumDbccOption[]> - Short form: -dbccopt

This parameter specifies options to the DBCC SQL command that are used to validate and verify the database that is being processed. When you use this parameter, you are explicitly requesting DBCC options, and the system does read the registry to determine the default DBCC options. The security access issues for the registry are bypassed when you use this cmdlet option. The parameter uses the following values:

NOOPTION

NOINDEX

ALL_ERRORMSGS

NO_INFOMSGS (default)

TABLOCK

PHYSICAL_ONLY (default)

For more information about these options, see your Microsoft SQL Server documentation.

-TargetDatabase <String[]> - Short Form: -tgDb

When you want to restore the database with a new name, use this parameter

-TargetLocation - Shot Form: -tgloc

This parameter defines the location to which you want to restore a database.

Syntax: `-TargetLocation Source_Database_Name, n, Logical_FileName_1, Desination_FilePath_1,....,Logical_FileName_n, Desination_FilePath_n`

Where, `Source_Database_Name` represents the source database, `n` represents the number of database files, `Logical_File_1` to `Logical_File_n` represents the database logical file names, `Destination_File_1` to `Destination_File_n` represents the corresponding destination file names for the logical file name, if you want to change the database file name at the target location.

For example, `restore-backup -Database db -TargetDatabase newDB -TargetLocation db,2, DB_FileName, K:\NewDB\NewDB.mdf, LOG_FileName, K:\NewDB\NewDB.ldf`

-TargetServerInstance <String[]> - Short Form: -tgInst

This parameter specifies the name of the new SQL server if you want to restore the database to a new SQL server. SnapManager takes the source SQL server instance as the default.

-PointInTime <String[]> - Short Form: -pit

Use this switch to restore databases until a specific point in time. The format for the point-in-time string is yyyy-mm-ddThh:mm:ss, with time specified in a 24-hour format.

In case of multiple databases you should specify the point-in-time values for every database separated by a comma. The number of values after the parameter name should equal the number of databases selected. The first value will be applied to the first database specified after the -Database parameter, the second value to the second database, and so on. The following example illustrates the usage:

```
-pit 2008-10-22T11:50:00, 2008-11-25T22:50:00
```

> **Note:** The parameter correspondence is one-to-one, that is, the first point-in-time parameter value specified after the parameter -pit is applied to the first database specified in the parameter -Database and the second point-in-time parameter value to second database and so on. The values should conform to the required PointInTime regular expression.

-RestoreArchive - Short Form: -rstarch

Use this parameter to restore a database from an archived backup.

-RestoreFromUnmanagedMedia - Short Form: -rstumm

Use this parameter if you are restoring databases from archived SnapManager backup sets.

-SnapInfoDirectory <String> - Short Form: -snapinfo

Use this parameter to specify the SnapInfo directory path of the archived backup set. Use the parameter only along with the -RestoreFromUnmanagedMedia parameter. This parameter is required when you specify the "-BackupServer" parameter.

-MarkName <String[]> - Short form: -mark

This parameter indicates the marked transaction at which to stop the transaction log recovery.

-MarkTime <String[]> - Short form: -mktm

This parameter specifies a unique timestamp to guarantee the uniqueness of the input restored mark.

-RestoreBeforeMark [<Boolean>] - Short form: -beforemk

This true or false value indicates whether the specified marked transaction log should be included in the restore.

-RecoverDatabase <Boolean[]> - Short Form: -recoverdb

This parameter indicates whether the database fully recovered or left in a partially recovered state after the cmdlet finishes, to facilitate future SQL transaction log restores. This is an array of booleans, so it must match the same number of elements of the -database array. If the it does not match the number of elements of the -database array, an error is given. This defaults to $true for all databases unless the -standbyPath is given, in which case it defaults to $false for all databases.

-StandbyPath <String> - Short Form: -standby

This parameter indicates the path to the standby recovery file where incomplete transactions are stored after restoring a full database and its transaction logs. There is no default if you specify this parameter. The path must be to the standby directory if more than one database shares a LUN. If the database is on a dedicated LUN, then it must be a specific file. If the -standbypath parameter is given, the -RecoveryDatabase given must be -RecoverDatabase $False, otherwise it defaults to $false for all databases if no _RecoverDatabase parameter is specified.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-RestoreArchivedBackup - Short Form: -rstarchbkup

Use this parameter to restore database from an archived backup.

-SnapVaultSecondary - Short Form: -vaultsec

This optional parameter identifies the backup vault from which you want to restore a database. If you do not specify this parameter, SnapManager chooses one of the backup vaults. You use this parameter in conjunction with the `-RestoreArchivedBackup` parameter. If you specify this parameter with the `-AvailabilityGroup` parameter, then the Availability Group databases must be spread across the same volumes. Otherwise, do not specify this parameter and SnapManager will choose one of the backup vaults. This parameter applies to clustered Data ONTAP only.

The syntax for this parameter is as follows:

```
-SnapVaultSecondary n, Vserver:volume
```

Where n is the number of Storage Virtual Machine (SVM, formerly known as Vserver):volume pairs.

Example: `-SnapVaultSecondary 3, Vserver1:volume1, Vserver2:volume2, Vserver3:volume3`

-NoAccessToRemoteBackup - Short Form: -noaccessarchivebkup

This parameter specifies that there is no direct access to the secondary storage system. SnapManager uses the proxy server to access the secondary storage system.

-ProxyServer <String> - Short Form: -pxy

This parameter defines the name of the proxy server. Use it along with the parameter NoAccessToRemoteBackup.

-PreCommand <String> - Short form: -precmd

This parameter indicates to run a command before the current operation.

-PreCommandPath <String> - Short form: -precmdpath

This parameter specifies the operating system path to the command to be run before the SnapManager operation starts.

-PreCommandArguments <String> - Short form: -precmdargs

Use this parameter to specify a list of strings of SnapManager operation-specific information or user defined arguments to be passed to the program or script.

-PreCommandHost <String> - Short form: -precmdhost

Use this parameter to specify the host machine name on which the command is to be run before the operation starts.

-PreCommandErrors <EnumHandleCmdError[]> - Short form: -precmnderrors

Use this parameter to specify how to handle errors on the pre-command and the following SMSQL operation. ContinueOnError value indicates that the following SMSQL operation will be executed anyway. StopOnPreCmdError value indicates that if a pre-command script get an error, the remaining SMSQL operation will not be attempted.

-PostCommand - Short form: -postcmd

Use this parameter to indicate to run a command after the current operation.

-PostCommandPath <String> - Short form: -postcmdpath

Use this parameter to specify the operating system path to the command to be run after the SMSQL operation starts.

-PostCommandArguments <String> - Short form: -postcmdargs

Use this parameter to specify a list of strings of SnapManager operation-specific information or user defined arguments to be passed to the program or script.

-PostCommandHost <String> - Short form: -postcmdhost

Use this parameter to specify the host name on which the command is to be run after the operation is complete.

-PostCommandErrors <EnumHandleCmdError[]> - Short form: -postcmderrors

Use this parameter to specify how to handle errors on the following post-command run. ContinueOnError value indicates that the following SMSQL operation will be executed anyway. StopOnPostCmdError value indicates that if a post-command script get an error, the remaining SMSQL operation will not be attempted.

-AvailabilityGroup <String> - Short Form: -ag

Use this parameter to specify one or more names of Availability Groups for which this backup applies.

-IgnoreRepLogs - Short form: -nosharelogs

Use this parameter to ignore the transaction logbackups from SnapManager Repository Share.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual deletion operation starts.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1: `restore-backup -Server sql1 -Database "Db1"`

This command restores the backup of database Db1 on SQL Server sql1.

Example 2: `restore-backup -svr 'VM-VS-1' -inst vm-vs-1 -d 'ds_test7' -backup sqlsnap__VM-VS-1_07-18-2008_03.19.14__Daily`

This example restores the specified backup on the given server instance.

Example 3: `restore-backup -inst 'SNAPMGR-65' -Database 'dbDef_1' -restorelastbackup 0 - standbypath u:\temp\standby -recoverdatabase $false`

This example specifies the path where incomplete transactions are stored after restoring a full database and its transaction logs.

Example 4: `restore-backup -Server snapmgr-62 -FederatedGroups 1, snapmgr-62\SQLEXPRESS62, 1, TestData -Mark mypsmark -MarkDesc "mymark description" -Logbkup`

This example restores the log to a marked transaction.

Example 5: `restore-backup -svr 'venudhar-2k8vm2' -inst 'venudhar-2k8vm2\heitz' -ag 'testag' -restorelastbackup 0`

This command restores all the databases belonging to the specified Availabilty group.

Example 6: `restore-backup -svr 'VM-VS-1' -inst vm-vs-1 -d 'ds_test7' -backup sqlsnap__VM-VS-1_07-18-2008_03.19.14__Daily -RestoreArchivedBackup -vaultsec 2,vserver1:volume1,vserver2:volume2`

This example restores the database ds_test7 from the specified SnapVault secondary volume on the specified server instance.

verify-backup

Name

verify-backup

Synopsis

This cmdlet enables you to verify the SQL Server databases in SnapManager PowerShell command-line interface.

Syntax

```
verify-backup [-Server <String>] [-UserName <String>] [-Password <String>]
[-Database <String[]>] [-VerifyServerInstance <String>] [-VerSvrLogin
<String>] [-AttachDB] [-VerSvrPassword <String>] [-UpdateMirror] [-
VerDestVolume] [-VerifyOnDestVolumes <String[]>] [-ManagementGroup
<String>] [-BackupNo <Int32>] [-MountPointDir <String>] [-UseMountPoint] [-
UseDriveAvailable] [-Command] [-RunCommand <String>] [-CommandArguments
<String>] [-CommandServer <String>] [-PreCommand] [-PreCommandPath
<String>] [-PreCommandArguments <String>] [-PreCommandHost <String>] [-
PreCommandErrors <EnumHandleCmdError[]>] [-PostCommand] [-PostCommandPath
<String>] [-PostCommandArguments <String>] [-PostCommandHost <String>] [-
PostCommandErrors <EnumHandleCmdError[]>] [-DBCCOption <EnumDbccOption[]>]
[-apicontext] [-ArchiveBackup] [-VerifyArchiveBackup] [-
ArchivedBackupRetention <String>] [-ClusterAware] [-WhatIf] [-Confirm] [-
AvailabilityGroup] [-BackupPriority] [-Primary] [-Secondary] [-
PreferredBackupReplica] [<CommonParameters>]
```

Description

This cmdlet enables you to perform verification operations. You can mount the Snapshot copies, manage SnapMirror relationships and destinations, assign management groups for verification and so on.

You can also implement these options with the SnapManager user interface.

Parameters

-Server <String> - Short Form: -svr

This parameter denotes the name of the host SQL Server on which the SQL Server instances reside. SnapManager takes the local computer name as the default server name. If no default host exists, SnapManager attempts to use the following as the default:

- The VerifyServerInstance specified by the user
- The configured verification server for the current machine (in the registry) done in the configuration wizard, or backup verification settings
- The VerificationServerInstance from the SQL Server being backed up as the verification server
- The current machine

Using this parameter, you can also specify a particular SQL Server instance. The following examples illustrate the usage:

```
-svr win-225-161
```

```
-svr sql1
```

-Username <String> - Short Form: -usr

UserName is the SQL Server account name. It is specified if the SQL Server computer is accessed using a different account from that used to access the production SQL Server. If not specified the Windows NT Authentication username will be taken.

-Password <String> - Short Form: -pwd

This parameter is the SQL Server account password. SnapManager ignores this parameter if the parameter -UserName is not specified.

-Database <String[]> - Short Form: -d

Short Form: -d This parameter specifies the original databases to backup. If you specify multiple database names, the list is separated by commas. If you do not specify the database parameter, the cmdlet backs up all SQL server instances that are peer instances of the SQL server in the -Server parameter.

-VerifyServerInstance <String> - Short Form: -verInst

This parameter specifies the separate SQL Server that is used to run the Database Consistency Check (DBCC) utility. If you have not specified the -verify parameter, SnapManager ignores this parameter.

The following example illustrates the usage:

```
-verInst win-225-161
```

Here the SQL Server instance is the local or remote SQL Server instance to verify on. SnapManager takes the configured SQL Server instance that is used for verify in client configuration (registry) as the default SQL Server instance.

-VerSvrLogin <String> - Short Form: -verlogin

This parameter specifies that SQL Server authentication is used. If not specified, the default Windows NT Authentication mechanism is used.

-AttachDB - Short Form: -attdb

If the operation includes a database or transaction log verification, use this option when you want to specify that the databases are to be attached after the verification operation completes.

-VerSvrPassword <String> - Short Form: -verpwd

This parameter is used to input the verification server password. SnapManager ignores this parameter if the parameter -VerSvrLogin is not specified.

-UpdateMirror - Short Form: -updmir

Use this option to update the SnapMirror destination after the backup or verification operations are complete, if you are using backups that reside on volumes configured as SnapMirror sources.

-VerDestVolume - Short Form: -verdest

Use this parameter to verify the database on the SnapMirror destination volume. SnapManager sets it to false by default.

-VerifyOnDestVolumes <String[]> - Short form: -vermirror

Specify this parameter to override the default SnapMirror relationships. Enter a comma-separated list of the source storage system, the source volume, the destination storage system, and the destination volume.

-ManagementGroup <String> - Short form: -mgmt

This parameter denotes the backup or verify operation that SnapManager performs on daily, or weekly, or standard basis. The default management group is standard.

-BackupNo <Int32> - Short Form: -bkno

This option specifies the number of most recent unverified backups to verify. It is an integer with a default value of 1.

-MountPointDir <String> - Short Form: -mpdir

Use this parameter to specify the mount point directory on which a backup set is mounted during database verification. This parameter should be used along with the parameter -UseMountPoint.

Note: This option is valid only if you specify the parameter -BkupSIF.

-UseMountPoint - Short Form: -mp

This parameter specifies that the Snapshot copy must be mounted to an NTFS directory. During a SnapManager verification operation, Snapshot copies are mounted to the default NTFS directory for database verification. The option is effective when there are no available drive letters to mount the Snapshot copies. It overrides preconfigured SnapManager verification settings.

-UseDriveAvailable - Short Form: -drvavail

Use this parameter to specify the mount point with available drive on which a backup set will be mounted during database verification.

-Command - Short form: -cmd

This switch parameter that runs a command after the backup or verify operation.

-RunCommand - Short form: -runcmd

This parameter runs the specified command after the SnapManager backup or verification operation is complete. It defines the complete path for the command to be run after the backup or verify operation is complete. There is no default.

-CommandArguments <String> - Short form: -cmdargs

This option contains the string of SnapManager operation-specific information to be passed to your program or script. It is considered only if Command and RunCommand are specified. There is no default.

-CommandServer <String> - Short form: -cmdsvr

This obsolete parameter (now replaced by PostCommandHost) was used to indicate the machine where the desired command should run after the operation is complete. The default was to run on the current machine. This was only considered if -command and -RunCommand were specified.

-PreCommand <String> - Short form: -precmd

This parameter indicates to run a command before the current operation.

-PreCommandPath <String> - Short form: -precmdpath

This parameter specifies the operating system path to the command to be run before the SnapManager operation starts.

-PreCommandArguments <String> - Short form: -precmdargs

Use this parameter to specify a list of strings of SnapManager operation-specific information or user defined arguments to be passed to the program or script.

-PreCommandHost <String> - Short form: -precmdhost

Use this parameter to specify the host machine name on which the command is to be run before the operation starts.

-PreCommandErrors <EnumHandleCmdError[]> - Short form: -precmnderrors

Use this parameter to specify how to handle errors on the pre-command and the following SMSQL operation. ContinueOnError value indicates that the following SMSQL operation will be executed anyway. StopOnPreCmdError value indicates that if a pre-command script get an error, the remaining SMSQL operation will not be attempted.

-PostCommand - Short form: -postcmd

Use this parameter to indicate to run a command after the current operation.

-PostCommandPath <String> - Short form: -postcmdpath

Use this parameter to specify the operating system path to the command to be run after the SMSQL operation starts.

-PostCommandArguments <String> - Short form: -postcmdargs

Use this parameter to specify a list of strings of SnapManager operation-specific information or user defined arguments to be passed to the program or script.

-PostCommandHost <String> - Short form: -postcmdhost

Use this parameter to specify the host name on which the command is to be run after the operation is complete.

-PostCommandErrors <EnumHandleCmdError[]> - Short form: -postcmderrors

Use this parameter to specify how to handle errors on the following post-command run. ContinueOnError value indicates that the following SMSQL operation will be executed anyway. StopOnPostCmdError value indicates that if a post-command script get an error, the remaining SMSQL operation will not be attempted.

-DBCCOption <EnumDbccOption[]> - Short form: -dbccopt

This parameter specifies options to the DBCC SQL command that are used to validate and verify the database that is being processed. When you use this parameter, you are explicitly requesting DBCC options, and the system does read the registry to determine the default DBCC options. The security access issues for the registry are bypassed when you use this cmdlet option. The parameter uses the following values:

NOOPTION

NOINDEX

ALL_ERRORMSGS

NO_INFOMSGS (default)

TABLOCK

PHYSICAL_ONLY (default)

For more information about these options, see your Microsoft SQL Server documentation.

-apicontext - Short form: none

Use this parameter when calling the cmdlet as an API call.

-ArchiveBackup - Short form: -arch

Use this parameter to archive database to a secondary storage system.

-VerifyArchiveBackup - Short form: -verarch

Use this parameter to verify database archived at the secondary storage system.

-ArchivedBackupRetention <String> - Short form: -archret

Use this parameter to specify whether you want to retain backups at the archived location on a daily, hourly, weekly, monthly, or unlimited basis.

-ClusterAware - Short form: cl

Use this parameter to specify that the cmdlet runs solely on the active node in a cluster environment.

-WhatIf - Short form: -wi

This parameter gives you a preview of an operation.

-Confirm - Short form: -cf

This parameter prompts you for confirmation before the actual deletion operation starts.

-AvailabilityGroup <String> - Short Form: -ag

Use this parameter to specify one or more names of Availability Groups for which this backup applies.

-BackupPriority <Integer,Integer> - Short Form: -bp

Use this parameter to specify a set of secondary Availability Groups on a cluster by specifying a range of backup priorities. The operation applies to all replicas with backup priorities in that range. The maximum priority must be in the range of 1 to 100. The minimum backup priority must be less than or equal to the maximum priority.

-Primary - Short form: -prm

If this parameter is defined, then the backup is only taken on the primary replica. If BackupPriority is also defined, then the primary replica must also satisfiy the BackupPriority values.

-Secondary - Short form: -sec

If this parameter is defined, then the backup is taken on all secondary replicas. If BackupPriority is also defined, then the secondary replicas must also satisfiy the BackupPriority values.

-PreferredBackupReplica - Short form: -preferbkreplica

Use this parameter to specify that only the preferred backup replica is verified. The preferred backup replica is set from the Availability Group properties in the SQL Server 2012 Management Studio.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see about_CommonParameters (http://go.microsoft.com/fwlink/?LinkID=113216).

Examples

Example 1: `verify-backup -svr 'VM-VS-1' -d 'VM-VS-1', '2', 'ds_test6', 'ds_test7' -verInst 'ZEUS-VM1\VERSERVER' -bkno 1 -mgmt standard -ArchiveBackup -VerifyArchiveBackup -ArchivedBackupRetention Daily`

This command initiates deferred verification for the specified database at the specified server, with one unverified most recent backup. The management groups are standard.

Configuring SnapManager application settings

Overview of SnapManager application settings

SnapManager application settings

The following table lists SnapManager application settings that can be configured or changed at any time after SnapManager has been installed. Shaded rows indicate settings that can also be configured using the Configuration wizard.

Application setting	How the setting can be accessed
SQL Server to be managed The first time you start SnapManager, the application automatically opens this dialog box to prompt you for this setting. See *Starting SnapManager for the first time after installation* on page 60.	• Actions pane • Action menu • Configuration wizard
SnapManager server identity	• Add Servers to be Managed • Configuration wizard
Backup verification settings From the Configuration wizard, you can specify only the verification server and security authentication method. In order to access other verification settings (the DBCC Options), you must open the Verification Settings dialog box.	• Backup wizard • Action menu • Configuration wizard
Backup settings	• Action menu • Backup wizard • Backup and Verify option
Clone settings	• Action menu • Clone wizard • Clone option

Application setting	**How the setting can be accessed**
Restore Settings	• Action menu • Actions pane • Restore wizard • Restore option
Fractional Space Reservation settings	• Actions pane • Action menu
Notification settings	• Actions pane • Action menu
Run Commands When the Run Commands dialog box is opened from the Actions menu, only the default settings can be viewed or configured. See *Pre-command and post-command script settings* on page 303. However, from within the context of a specific operation, the default settings are presented and then can be modified for this operation only. As an option, the default settings can be updated.	• Action menu • Within the context of a backup or database verification operation: ◦ Backup wizard ◦ Backup and Verify option
Report directory	• Actions pane • Action menu
License settings	• Actions pane • Action menu
Server connection settings	• Actions pane • Action menu
Find Backups	• Actions pane • Action menu • Restore wizard

Connecting to an SQL Server instance

About this section

If you want to add a server that makes all SQL server instances running on the server visible, use the "Add Servers to be Managed" option. SnapManager enables you to connect to the SQL Server you want to manage and to specify the security authentication method to be used to establish the connection.

About the "Add Servers to be Managed" option

Use the "Add Servers to be Managed" option to connect to add a server that makes all SQL server instances running on the server visible. When you specify or change the settings, SnapManager immediately connects to the specified SQL Server using the specified security authentication method. These settings remain in effect as the defaults until or unless you change them.

When you start SnapManager, the Add server to be managed dialog box opens automatically if a *default SQL Server* has not yet been specified. You cannot proceed to use SnapManager until you have successfully added a server to be managed. Thereafter, whenever the SnapManager application is started, it automatically connects to the default SQL Server using the default security authentication method.

If at a later time you want to manage another server, select "Add Servers to be Managed" to connect to the server running the server account that belongs to the Administrators group on that machine. For more information, see *Connecting to a different SQL server* in this topic.

Connecting to a different SQL server

To connect to a different SQL Server, complete the following steps.

Note: This changes only the SnapManager server identity on the current machine and does not change the SnapManager server identity on the remote host.

Step	Action
1	If SnapManager is already connected to the default SQL Server, click Disconnect Server in the Actions pane.
2	In the Actions pane, click Add Servers to be Managed.

Step	Action
3	Select the SQL Server from the list, type the name, or click the **Browse** button to select the server. Note the following: • If there is no default instance, specify one of the instance names (Server\Instance) instead of the server name. Even though you specify just one of the instances, SnapManager adds all of the instances on the server. • For an Availability Group, you can select any of the servers in the group.
4	Enter the Windows authentication or SQL server authentication under Login Details.
5	Click "Add."
6	If, instead of connecting to the specified SQL Server, SnapManager displays an error message regarding the SQL Server, MDAC, or SnapDrive version on that SQL Server computer, do the following: **1.** See *Verifying Windows host system requirements* on page 20 to determine which software components you need to update on the SQL Server computer. **2.** Close the SnapManager application. **3.** Upgrade the software on the SQL Server computer as needed. **4.** Restart SnapManager.

Connecting to an AlwaysOn failover cluster node

Before you can create and manage backups on an AlwaysOn failover cluster, you must connect to a node in the cluster to the SnapManager configuration.

Steps

1. In the **SnapManager for SQL Server** window, select **SnapManager for SQL Server**.
2. In the **Actions** pane, click **Add Servers to be Managed**

   The Add SQL Instance to manage dialog opens.
3. Use the dialog to choose a node within the Availability Group and establish the type of authentication (Windows authentication or SQL authentication).

   Note that the type of authentication specified here is used for all instances of Microsoft SQL databases on the cluster, so all of the databases must use the same method.

Enabling SnapManager to allow databases on any LUN or VMDK configuration

By default, the SnapManager Configuration wizard enforces database configuration restrictions when you migrate your databases to LUNs or VMDKs. You can enable the `unrestricted database layout` option, which removes layout restrictions imposed during configuration, enabling you to place your databases in any LUN or VMDK configuration.

About this task

The `unrestricted database layout` option removes the restriction of shared disk configurations from multiple databases. Also, when a database is spread across more than two disks, those disks can be reused for other databases. For example, you can spread a database's files across two or more LUNs or VMDKs, even if those LUNs or VMDKs are used for storing database files that belong to other databases.

You cannot disable the `unrestricted database layout` option after you enable it.

Note the following about enabling this option when you have existing LUN or VMDK configurations:

- If you enable the `unrestricted database layout` option and then move databases to a new configuration, you should take a full backup of all databases before any log backups are taken.
  Taking a full backup ensures that you can perform up-to-the-minute restores.
- If you enable the unrestricted database layout option and you have an existing LUN that contains multiple databases, you can continue to restore and clone the entire LUN, as long as the LUN contains the *entire* contents of multiple databases.
  For example, you can no longer restore or clone at the LUN level if you add a database to the LUN by spreading its files across this LUN and other LUNs. The entire database must reside on the LUN.

Steps

1. From the **Actions** pane, click **Configuration Wizard Option Settings**.
2. Select **Enable unrestricted database layout**.
3. Click **OK** to close the dialog box.

Database integrity verification options

You can use the Verification Settings dialog box to specify the verification server and configure database verification options.

Note: When you change the database verification server, this change does not affect any database backup (with verification) or database verification-only jobs that are already scheduled.

Selecting the database verification server

To view or change the verification server, complete the following steps *from the production SQL Server*.

Step	Action
1	If you are specifying a remote verification server, be sure it is set up properly, as described in "Requirements for a remote verification server" in *Database integrity verification options* on page 294.
2	From the production SQL Server (and not from a remote verification server), open the Verification Settings dialog box using any of these methods: • In the Actions pane, select **Backup Verification Settings**. • From the Backup wizard, go to the Verification Settings page and click **Verification Settings**. • From the Restore wizard, go to the Verification Settings page and click **Verification Settings**. **Result:** The Verification Settings dialog box appears. The SQL Server option is active by default and displays the host name of the current verification server.
3	In the SQL Server box, specify the stand-alone or cluster SQL Server instance you want to use as the database verification server. **Note:** If you plan to specify a remote verification server, ensure that the server is set up properly, as described in "Requirements for a remote verification server" in *Database integrity verification options* on page 294.
4	In the Connection panel, choose the security authentication method to be used to connect to the SQL Server: • Windows authentication • SQL Server authentication If you select Windows authentication mode (the default selection), users with a valid Windows account can log in to Microsoft SQL Server without supplying a user name and password. Windows Authentication relies on the user being authenticated by the operating system and takes advantage of Windows user security and account mechanisms. **Note:** Windows Authentication is the authentication mode recommended by Microsoft.

Step	Action
5	If you selected SQL Server authentication, also specify the login name and password. For more information, see *Starting SnapManager for the first time after installation* on page 60.
6	Click **OK**. **Result:** The Verification Settings dialog box closes. **Note:** Until you change these settings, database verification is run from the SQL Server you selected using the options you specified. It does not necessarily run on the system from which you opened the Verification Settings dialog box. It does not affect any database verification jobs that are already scheduled.

Selecting DBCC options

To specify which DBCC options are used to verify database backup Snapshot copies, complete the following steps.

Step	Action
1	Open the Verification Settings dialog box using any of these methods: • In the Actions pane, select **Backup Verification Settings**. • From the Backup wizard, go to the Verification Settings page and click **Verification Settings**. • From the Restore wizard, go to the Verification Settings page and click **Verification Settings**. **Result:** The Verification Settings dialog box appears. The SQL Server option is active by default and displays the host name of the current verification server.
2	Click the **DBCC Options** tab. **Result:** The DBCC Options option displays the selected DBCC options.
3	In the DBCC Options panel, select the options you want to use: • `NOINDEX` • `ALL_ERRORMSGS` • `NO_INFOMSGS` • `TABLOCK` • `PHYSICAL_ONLY` For more information about these options, see your Microsoft SQL Server documentation. **Note:** `PHYSICAL_ONLY` and `NO_INFOMGS` are selected by default.

Step	Action	
4	By default, the option “Leave database attached after verification” is left unchecked and the database is detached after the DBCC utility finishes.	
	If...	Then...
	If you want to detach the database after the verification finishes	Keep the “Leave database attached after verification” option unchecked.
	If you want to leave the database attached after the verification finishes	Select the “Leave database attached after verification” option. If a database verification (with or without a full database backup) is started or scheduled with this option enabled, a message box notifies you that this option is set and prompts you to confirm that you want to continue. Unless you explicitly detach the database and unmount the Snapshot copy after this operation completes, subsequent backup operations on this database encounter busy Snapshot copies.
5	Click **OK**. **Result:** The Verification Settings dialog box closes.	

Using the Mount Point tab

Use the Mount Point tab to specify how SnapManager is to access the database backup Snapshot copies during database integrity verification.

Related topics

- "Selecting the database verification server" in *Database integrity verification options* on page 294

To specify which method to use to access database backup Snapshot copies during the database integration verification, complete the following steps.

Step	Action
1	Open the Verification Settings dialog box using any of these methods: • In the Actions pane, select **Backup Verification Settings**. • From the Backup wizard, go to the Verification Settings page and click **Verification Settings**. • From the Restore wizard, go to the Verification Settings page and click **Verification Settings**.

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td>2</td><td colspan="2">In the Verification Setting tab, assign either a drive letter or a directory path to access the backup Snapshot copy. By default, the default mount directory path appears as follows:
<code>C:\SMSQLMnPt</code>
If you have a configuration that has SMB shares only, keep the default setting. SnapManager does not use the path. However, if you have an FCI configuration with SMB shares only and the verification server is a failover cluster instance, the default setting does not work. You must specify a UNC path. When you specify a UNC path, SnapManager does not enforce the use of a shared disk as the mount point directory. SnapManager does not use the UNC path.</td></tr>
<tr><td></td><td>If you want to...</td><td>Then...</td></tr>
<tr><td></td><td>Mount the Snapshot copy on the next available drive letter</td><td>Select the Automatically assign available drive letter option.</td></tr>
<tr><td></td><td>Mount the Snapshot copy on a specific NTFS mount point</td><td>Do the following:
1. Select the Mount in an empty NTFS directory option.
2. Enter or browse to the directory path of an NTFS mount point.
Note: This mount point is used if SnapManager is configured to use drive letters but runs out of available drive letters.
Note: If you are using a cluster SQL Server instance as the verification server, the mount point directory must be on a shared LUN that is in the same cluster group as the verification server.</td></tr>
<tr><td>3</td><td colspan="2">Click OK.
Result The Verification Settings dialog box closes. After the database verification, the Snapshot copy directory created in the default mount point directory path is unmounted automatically.</td></tr>
</table>

SnapManager backup options

Use the Backup Settings dialog box to configure default settings for SnapManager backup operations.

Configuring the profile of a full database backup

To configure the profile of a full database backup, complete the following steps.

Note: For a complete list of parameters that are applied to a full database backup operation, see "Information you need to specify for a full database backup" in *Backing up, replicating, and archiving databases using SnapManager* on page 115.

Step	Action
1	Open the Backup Settings dialog box using any of the following methods: • From the Actions pane, select Backup Settings. • From the Action menu, click **Backup Settings**. • From the Backup wizard, go to the Backup Options page and click **Backup Settings**. **Result:** The Backup Settings dialog box appears. The Full Database Backup option is active by default and displays the current settings.
2	In the Select a backup naming convention... panel, specify the naming convention you want used to form database backup Snapshot copy names and SnapInfo directory Snapshot copy names: • If you want the most recent backup to be identified by the Snapshot copy name that includes the string `recent`, select "Use Generic...." You can click **Advanced** to enable the enhanced Snapshot copy naming convention, which creates a most recent backup copy per management group (daily, weekly, standard) with the suffix "__recent". You cannot disable this option after you enable it. • If you want all Snapshot copy names, even for the most recent backup, to contain the Snapshot copy creation date and time, select "Use Unique Naming convention." This option is selected by default. For more detailed information, see "SnapManager backup set naming conventions" in *How SnapManager backup data is organized* on page 99.
3	In the Verify mounted online databases panel, select whether you want to run verification against the live database before the backup, after the backup, or both before and after the backup: • If you want to run the DBCC utility against the live database before the database is backed up, select "Run DBCC physical integrity verification before the backup"." • If you want to run the DBCC utility against the live database after the database is backed up, select "Run DBCC physical integrity verification after the backup." **Note:** By default, both options are not selected, because database verification is a time-consuming activity.

Step	Action
4	To apply your changes and close the dialog box, click **OK**. **Result:** The new settings are applied to all subsequent full database backups.

Configuring the profile of a transaction log backup

To configure the profile of a transaction log backup, complete the following steps.

Note: For a complete list of parameters that are applied to a transaction log backup operation, see "Information you need to specify when creating a transaction log backup" in *Managing transaction log backups using SnapManager* on page 126.

Step	Action
1	Open the Backup Settings dialog box using any of the following methods: • From the Actions pane, select **Backup Settings**. • From the Action menu, click **Backup Settings**. • From the Backup wizard, go to the Backup Options page and click **Backup Settings**. **Result:** The Backup Settings dialog box appears. The Full Database Backup option is active by default and displays the current settings.
2	In the dialog box, click the **Transaction Log Backup** tab. **Result:** The Transaction Log Backup tab displays the current settings.
3	In the Transaction Log Backup tab, select **Create snapshot of the SnapInfo drive after backup** to create a Snapshot copy of the SnapInfo directory after the backup operation finishes.
4	Select **Delete SnapInfo Snapshots** to delete SnapInfo Snapshot copies based on their number or the number of retention days.
5	If you want to run the DBCC utility against the live database after the database is backed up, select **Truncate committed transactions in the transaction log.**
6	If you want some or all of the transaction logs copied to the repository share, select **Copy transaction log backups to share** and determine if this applies to all databases or only to Availability Group databases. Then set the retention policy by enabling **Delete Share Log backups** and choose how the backups are selected for deletion. **Note: Copy transaction log backups to share** is enabled by default for Availability Group databases.
7	To apply your changes and close the dialog box, click **OK**. **Result:** The new settings are applied to all subsequent full database backups.

Configuring backup concurrency

Microsoft recommends a maximum of 35 databases per backup Snapshot copy when running out of SQL Server thread resources.

To set the maximum number of databases per Snapshot copy, enter a number in the field.

Note: During backup, SnapManager might use a different number of maximum databases per Snapshot copy than what is configured in the Backup Settings dialog box. This happens because SnapManager tries to use the smallest number of Snapshot copies as possible. For example, if the maximum databases per Snapshot copy setting is 35 and there are 45 databases to back up, SnapManager might back up all 45 databases in the same Snapshot copy operation.

SnapManager restore options

Use the Restore Settings dialog box to configure default settings for SnapManager restore operations.

Understanding the restore options

The following table describes each of the Restore Settings options.

Restore option	Description	Default
Recover database without restoring at the end of restore if needed	If the database is not fully operational and you want to leave it operational after restore, on selecting this option SnapManager skips the restore and performs the recover operation.	Not selected
Restore databases even if existing databases are online	If this option is selected and an existing database is online at the time of the restore operation, SnapManager proceeds with the restore and overwrites the existing database.	Selected
Retain SQL database replication settings	If this option is selected and you are restoring databases for an SQL Server instance that is acting as a Publisher or as a Subscriber in a replication topology, the replication relationship is retained after the SnapManager restore operation finishes.	Not selected
Create transaction log backup before restore	If this option is not selected, SnapManager does not create a transaction log backup before the restore is performed, thereby decreasing overall restore time. Clear this option under the following circumstances: • You are recovering from a mirrored backup for which the transaction log files were lost. Disabling this option avoids subsequent creation of SnapManager backup sets on a recovery path that is inconsistent with that of the database. • You are restoring a log-shipped database.	Selected

Restore option	Description	Default
Abort database restore if transaction log backup before restore fails	If this option is selected and the transaction log backup before restore fails, SnapManager aborts the database restore operations. This option is available when the option "Create transaction log backup before restore" is selected.	Not selected
Ignore Logbackups from SMSQL Repository Share	If this option is selected, log backups on the repository share are not used in the restore.	

Configuring the profile of a restore operation

To configure the profile of a restore operation, complete the following steps.

Note: For a complete list of parameters that are applied to a database restore operation, see *Performing a restore operation* on page 167.

Step	Action
1	Open the Restore Settings dialog box using any of the following options: • From the Actions pane, select Restore Settings. • From the Action menu, select Restore Settings. • From the Restore Wizard, go to the Restore Settings screen and then click the Restore Settings button. **Result** The Restore Settings dialog box appears.
2	Select any combination of the restore options you want to use: • Leave the database in a state where more logs can be applied • Restore databases even if existing databases are online • Retain SQL database replication settings • Create transaction log backup before restore • Abort database restore if transaction log backup before restore fails
3	Click OK to apply your changes and close the dialog box. **Result** The new settings will be applied to all subsequent database restore operations.

Precommand and postcommand script settings

About precommand and postcommand settings

When you start a SnapManager backup, database verification, restore, or clone operation, you can automatically run a command or script either before the operation starts or after the operation is complete. You can also choose how script errors are handled.

Where to specify a command before an operation

You can set up commands or scripts to run before SnapManager operations in the following ways:

- From the backup, restore, or clone wizards
- From the Quick Backup, Quick Restore, and Quick Clone dialog boxes
- From the Run Commands dialog box

How to specify a script to run before an operation from a wizard or quick dialog box

From within the context of a SnapManager backup, verification, restore, or clone operation, you can use the Run Commands dialog box to do the following:

- Specify the details of the command:
  - Whether you want to continue or stop the operation for errors that occur during or prior to the script-command
  - The computer on which you want to run the command
  - The full path to the command
  - The sequence of SnapManager variables that you want to pass to the command
- Specify whether you want to save the current settings as the default settings

Note: If you want to change the default values specified in the Run Commands dialog box without starting an SnapManager for SQL operation like backup, verify, restore, or clone, you can open the Run Commands dialog box from the Option menu.

When you choose to run a script before an operation from a wizard or quick dialog box, you are prompted to specify the following information before the operation begins:

- Whether you want to stop the SnapManager operation (for example, backup, verify, restore, or clone) if the user script has an error.
- The host system from which the command is to be run
- The full path of the command that you want SnapManager to run after the backup or database verification is complete
- Any parameters that are to be passed to the command

- Because the command (your own program or script) is invoked from within the context of a SnapManager operation, you can pass the command information about the components of that operation.

  For example, if the precommand or postcommand is a batch file that is launched, it looks like `c:\bat\mytestCommand.bat $SqlSnapshot $Database`. In this example, inside the batch file, the %1 batch parameter corresponds to the first parameter `$SqlSnapshot` passed to the batch script; the %2 string corresponds to the second `$Database` parameter passed, and so on, because it is running in the context of the SnapManager for SQL operation.

To run a script from a wizard or a quick dialog box, see one of the following procedures:

- "Full database backup using Backup and Verify" in *Backing up, replicating, and archiving databases using SnapManager* on page 115
- "Restoring using the SnapManager Restore Wizard" in *Performing a restore operation* on page 167
- "Cloning a database that is in production" in *Types of clone operations performed using SnapManager* on page 180
- "Cloning a database in a backup set" in *Types of clone operations performed using SnapManager* on page 180

Setting precommand or script defaults

You can use the Run Commands dialog box to configure default values that you want used to prepopulate the Run Commands dialog box when it is opened from either the Backup, Verify, Clone, or Restore wizards and options.

To run a script from the Run Commands dialog box, complete the following steps.

Step	Action
1	From the Actions menu, click the **Run Commands** option. **Result:** SnapManager displays the Run Commands dialog box with the current default settings.
2	Select the list box item value for the operation (for example, backup, verify, restore, or clone) where you want these default Run Commands settings to apply.
3	Make sure that the **Pre-Operation Command** tab is selected.

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td rowspan="4">4</td><td colspan="2">Choose whether to stop the precommand operation if an error occurs.</td></tr>
<tr><td>If...</td><td>Then ..</td></tr>
<tr><td>You want the SnapManager for SQL Server operation to stop when an error occurs in the custom user command</td><td>Select Treat pre command errors as fatal by stopping the remaining SnapManager operation...</td></tr>
<tr><td>You want to ignore errors that occurred during the user command or script, and want to continue to run the SnapManager for SQL Server operation regardless of those script errors</td><td>Do not select Treat pre command errors as fatal by stopping the remaining SnapManager operation...</td></tr>
<tr><td>5</td><td colspan="2">In the “Specify a computer where...” box, enter or browse to the name of the host on which your program or script resides.</td></tr>
<tr><td>6</td><td colspan="2">In the “Specify the full path...” box, browse to your program or script.</td></tr>
<tr><td>7</td><td colspan="2">Form the command input string in the Command Arguments box. You can do this using any combination of the following methods:
• To enter text directly into the Command Arguments box, click in the box and type the desired text.
• To enter a SnapManager variable into the Command Arguments box, do the following:
1. If necessary, click in the Command Arguments box to position the cursor.
2. In the SnapManager Variables list, select the variable you want to enter.
For more information, see "Running a command or script after an operation" and "Command arguments that are operation-specific" later in this section.
3. Click Select.
Note: Several parameters, like the <code>$SnapInfoPath</code> and <code>$LogBackupFile</code> variables, are enclosed within double quotes by default, so that the actual path name can contain spaces without affecting the script invocation on the Windows command line. If you do not want the double quotes to appear in your command line, remove them from the Command Arguments field in the Run Commands dialog box.</td></tr>
<tr><td>8</td><td colspan="2">Repeat Step 4 as needed until the Command Arguments box contains the arguments you want to pass to your program or script.</td></tr>
<tr><td>9</td><td colspan="2">Click OK to apply your changes and close the Run Commands dialog box.
Your changes are saved as the default.</td></tr>
</table>

Running a command or script after an operation

SnapManager enables you to run scripts after database backup, verify, restore, or clone operations. SnapManager also enables you to choose how operation errors are handled. You can choose to stop the SnapManager for SQL Server operation if an error occurs during the script launch.

You can set up commands or scripts to run after SnapManager operations in the following ways:

- From the backup, restore, or clone wizards
- From the Quick Backup, Quick Restore, and Quick Clone dialog boxes
- From the Run Commands dialog box

How to specify a script to run after an operation from a wizard or quick dialog box

From within the context of a SnapManager backup, verification, restore, or clone operation, you can use the Run Commands dialog box to do the following:

- Specify the details of the command:
  - Whether you want the precommand to run regardless of the SQL operation's result
  - The computer on which you want to run the command (your own program or script)
  - The full path to the command
  - The sequence of SnapManager variables that you want to pass to the command
- Specify whether you want to save the current settings as the default settings

Note: If you want to change the default values specified in the Run Commands dialog box without starting or scheduling a database backup, transaction log only backup, or database verification, you can open the Run Commands dialog box from the Option menu.

To run a script from a wizard or a quick dialog box, see one of the following procedures:

- "Full database backup using Backup and Verify" in *Backing up, replicating, and archiving databases using SnapManager* on page 115
- "Restoring using the SnapManager Restore option" in *Performing a restore operation* on page 167
- "Cloning a database that is in production" in *Types of clone operations performed using SnapManager* on page 180

When you choose to run a script after an operation from a wizard or quick dialog box, you are prompted to specify the following information before the operation begins:

- Whether to run the postcommand regardless of the SnapManager for SQL operation's result.
- The host system on which the command is to be run
- The full path of the command that you want SnapManager to run after the backup or database verification is complete
- Any parameters that are to be passed to the command
- Because the command (your own program or script) is invoked from within the context of a specific backup or database verification, you can pass the command information about the components of that operation.

For example, if you run the following batch file script after the operation, following `c:\myPostCmd.bat $Database $SqlInstance`, the first parameter passed, $Database, corresponds to the %1 batch parameter, and the second parameter, $SqlInstance, corresponds to the batch %2 parameter.

After you have completed specifying the command and parameters, you can start the operation.

Setting postcommand or script defaults

You can use the Run Commands dialog box to configure default values that you want used to prepopulate the Run Commands dialog box when it is opened from either the Backup, Verify, Clone, or Restore wizards and options.

To specify the default command or script information to be run after a backup, verify, clone, or restore wizard operation, complete the following steps:

Step	Action	
1	From the Actions menu, click the **Run Commands** option. **Result:** SnapManager displays the Run Commands dialog box with the current default settings.	
2	Select the list box item value for the SnapManager or SQL Server operation where you want these default `run` command settings to apply.	
3	Make sure that the **Post Operation Command** tab is selected.	
4	Choose whether to stop the post-command operation if an error occurs.	
	If...	Then...
	You want the SnapManager for SQL Server operation (for example, backup, verify, restore, or clone) to stop when an error occurs in the custom user command	Select **Treat pre command errors as fatal by stopping the remaining SnapManager operation...**
	You want to ignore errors that occurred during the user command or script, and want to continue to run the SnapManager for SQL Server operation regardless of those script errors	Do not select **Treat pre command errors as fatal by stopping the remaining SnapManager operation...**
5	In the "Specify a computer where..." box, enter or browse to the name of the host on which your program or script resides.	
6	In the "Specify the full path..." box, browse to your program or script.	

Step	Action
7	Form the command input string in the Command Arguments box. You can do this using any combination of the following methods: • To enter text directly into the Command Arguments box, click in the box and type the desired text. • To enter a SnapManager variable into the Command Arguments box, do the following: **1.** If necessary, click in the Command Arguments box to position the cursor. **2.** In the SnapManager Variables list, select the variable you want to enter. For more information, see "Running a command or script after an operation" and "Command arguments that are operation-specific" in this section. **3.** Click **Select**. **Note:** Several parameters, like the `$SnapInfoPath` and `$LogBackupFile` variables, are enclosed within double quotes by default, so that the actual path name can contain spaces without affecting the script invocation on the Windows command line. If you do not want the double quotes to appear in your command line, remove them from the Command Arguments field in the Run Command Operation dialog box.
8	Repeat step 6 as needed until the Command Arguments box contains the arguments you want to pass to your program or script.
9	Click **Save** to save your changes as the default run commands settings for the operation selected, and then close the Run Command dialog box. Repeat Steps 2 through 7 to set up the default for each operation type. Your changes are saved as the default.

Precommand arguments

The following precommand arguments apply to backup, verify, restore, and clone operations.

Variable	Description

`$Database`	Specifies the logical name of the database processes.
	To prevent PowerShell from interpreting the value of this parameter, be sure to enclose the entire parameter value with single quotes: `-PreCmdArg '$Database $ServerInstance'`.
	Example:
	`DatabaseAccounting`
	If you want to have more than one database expanded, repeat the parameter as many times as you want.
	Example:
	`AccountingDB1 AcmeServer1/SqlInst1 FinanceDB2 AcmeServer1/SqlInst2`

`$ServerInstance`	Specifies the name of the SQL server instance that is actually processed. Example: `ACMESERVER1\SQLINSTANCE1`

Postcommand arguments

The following postcommand arguments apply to backup, verify, restore, and clone operations.

To prevent PowerShell from interpreting the value of a parameter, be sure to enclose the entire parameter value with single quotes: `-PostCmdArg ‘$Database $ServerInstance $SqlSnapshot'`.

Variable	Description

`$InfoSnapshot`	Expands to the name of a SnapInfo directory Snapshot copy.
	Examples:
	`sqlinfo__winsrvr2__01-31-2014_15.03.09`
	`sqlinfo__winsrvr2__recent`

Variable	Description
`$LogBackupFile`	Expands to the full path name of the transaction log backup file. Example: `I:\SMSQL_SnapInfo\SQL__WINSRVR2\DB__Northwind \LogBackup\ 11-01-2004_13.34.59__Northwind.TRB`
`$OperationStatus`	Provides the status of the SMSQL operation. Example: 5234
`$PreCommandStatus`	Provides the precommand status to the postcommand if the postcommand is executed based on the status of the earlier precommand. Example: 5234

`$SnapInfoName`	Expands to the name of the SnapInfo directory.
	Examples:
	`WINSRVR2__recent`
	`WINSRVR2_11-23-2013_16.21.07__Daily`
	If you use this variable, you must also provide the correct path to the directory.

`$SnapInfoPath`	Expands to the name of the SnapInfo subdirectory. This argument is used in backup and verification operations.
	Example:
	`I:\SMSQL_SnapInfo\SQL__WINSRVR2\DB__Northwind`
	For restore and clone operations, this argument specifies the path to the Snapshot copy information metadata that is being used for the database restore.
	Example:
	`U:\SMSQL_SnapInfo\VDISK__E\FG__ \05-14-2010_15.33.41\SnapInfo__05-14-2010_15.33. 41.sml`

Variable	Description

`$SqlSnapshot`	Expands to the name of an SQL Server database Snapshot copy. This argument is used for backup and verification operations.
	Examples:
	`sqlsnap__winsrvr2__01-31-2014_15.03.09`
	`sqlsnap__winsrvr2__recent`
	The number of database Snapshot copies in a SnapManager backup set depends on the number of volumes used to store the databases included in the backup.
	For restore and clone operations, this argument specifies the name of the Snapshot copy to be restored.
	Example:
	`sqlsnap__winsrvr2__01-31-2014_15.03.09`
	`sqlsnap__winsrvr2__recent`

Several parameters, like the `$SnapInfoPath` and `$LogBackupFile` variables, are enclosed within double quotes by default, so that the actual path name can contain spaces without affecting the script invocation on the Windows command line. If you do not want the double quotation marks to appear in your command line, remove them from the Command Arguments field in the Run Commands dialog box.

The following postcommand arguments apply only to restore and clone operations.

Variable	Description
$StandbyFile	This is the full file system path of the SQL standby file used on a restore. This file path is calculated during the restore-clone operation as a temporary file when incomplete transactions are removed from the database and stored in the file for later use. The user requests to generate a standby (or *undo*) file in a certain directory, but the full file name path actually used is not known until the restore or clone operation is launched. This happens when several databases are restored at the same time to the same LUNs. By default, this is created in the `snap-info` directory. Example: `U:\SMSQL_SnapInfo\VDISK__E\UNDO_SECLOCSYS_db5.dat`
`$TargetDatabase`	Specifies the destination name of the database to restore. Example: `DatabaseAccountingRestoredCopy`
`$TargetServerInstance`	Specifies the destination SQL Server instance to be used. Example: `ACMESERVER2\SQLINSTANCECOPY`
`$TargetDatabaseFile`	Specifies the target file system database path to be used. Example: `Z:\MNT\NETAPP1\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Data\DatabaseAccounting.mdf`

Command arguments that are operation-specific

Each SnapManager operation that supports the Run Commands feature parses only the variables that apply to the operation as you have specified it.

The following table shows which of the command variables are available to the Run Commands feature, depending on which SnapManager operation is used to invoke the feature.

Variable	SnapManager operation that is used to invoke the Run Commands feature				
	Full backup	**Transaction log backup**	**Verification of full backup**	**Restore**	**Clone**
$Database	Parsed	Parsed	Parsed	Parsed	Parsed
`$InfoSnapshot`	Parsed	Parsed	—	—	—
`$LogBackupFile`	Parsed	Parsed	—	—	—
$ServerInstance	Parsed	Parsed	Parsed	Parsed	Parsed
$OperationStatus	Parsed	Parsed	Parsed	Parsed	Parsed
$PreCommandStatus	Parsed	Parsed	Parsed	Parsed	Parsed
`$SnapInfoName`	Parsed	Parsed	Parsed	—	—
`$SnapInfoPath`	Parsed	Parsed	Parsed	—	—
`$SqlSnapshot`	Parsed	—	Parsed	Parsed	Parsed
$StandbyFile	—	—	—	Parsed	Parsed
$TargetDatabase	—	—	—	Parsed	Parsed
$TargetDatabaseFile	—	—	—	Parsed	Parsed
$TargetServerInstance	—	—	—	Parsed	Parsed

1. Any argument can be repeated, and if multiple databases or servers are backed up, they are substituted in order.

   If no value corresponds to the $NNNN parameter, then a string that reads NULL is substituted for the $NNNN parameter.

2. Full backup with the Run Transaction Log Backup option selected: The `$LogBackupFile` variable is parsed only when the transaction logs are backed up after full backup.

Enabling or disabling database migration back to local disks

The primary function of the Configuration wizard is to migrate SQL Server databases to LUNs, SMB shares, or VMDKs so that the databases can be backed up and restored using SnapManager. If you choose to stop using SnapManager as your data management tool, you can also use the Configuration Wizard to migrate your databases back to local disks.

However, by default, the Configuration wizard does not list any local drives unless you enable an option.

Related topics

- *Understanding control-file based configuration* on page 65

To enable or disable the ability to migrate databases back to local disks, complete the following steps.

Step	Action	
1	From the Actions pane, click "Configuration Wizard Option Settings".	
2	The option "Enable databases to be migrated back to local disk" appears.	
	If...	Then...
	You need to enable database migration back to local disks	Select the option "Enable databases to be migrated back to local disk"
	You need to disable database migration back to local disks	Clear the option "Enable databases to be migrated back to local disk".
3	Click OK to close the dialog box.	

SnapManager report directory options

Default report directory

By default, the SnapManager reports are stored in a subdirectory named `Report` under the directory in which the SnapManager application is installed. If you installed SnapManager in its default installation directory, then the default report directory path is as follows:

```
C:\Program Files\NetApp\SnapManager for SQL Server\Report
```

Reasons to change the report directory

Reasons for changing the location of the SnapManager report directory are described in the following paragraphs.

Limited space If you find you have limited space in the current report directory, you can change the report directory to a different location that has more available disk space.

Clustered environment If you are running SQL Server and SnapManager in an MSCS cluster, storing the SnapManager reports in the default location (in a directory named `Report` under the SnapManager installation directory) would not allow the report directory to be shared between the nodes in the cluster. Furthermore, you would not see the same reports from different nodes.

To avoid these problems, you can move the report directory to a disk that belongs to the same group as your SQL Server virtual server. This needs to be performed from every SnapManager node.

Accessing reports created in a previous directory

If you change the name or location of the SnapManager report directory, you cannot use the SnapManager Reports option to view or print any reports that were created in that report directory.

However, assuming the previous report directory was not explicitly changed or removed, any reports created in that directory are still accessible. In order to view or print those older reports, you must change the report directory back to its previous location.

Using the Report Directory Setting dialog box

Use the Report Directory Setting dialog box to view and change the directory where your SnapManager reports are stored.

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td>1</td><td colspan="2">From the SnapManager Actions pane, select Report Directory Setting.
Result The Report Directory dialog box appears and displays the current location of the report directory.</td></tr>
<tr><td rowspan="4">2</td><td colspan="2">Specify the new location for the report directory. The directory cannot be located on a CIFS share.
Attention: Do not use a disk that contains SQL Server or SnapManager data for the report directory; it is restored from the Snapshot copy when you perform a SnapManager Restore.</td></tr>
<tr><td>If...</td><td>Then...</td></tr>
<tr><td>You know the full directory path name</td><td>Click in the Report Directory box and modify the path name.</td></tr>
<tr><td>You prefer to browse to the new location</td><td>Click Browse to use a browse dialog box to select the new location.</td></tr>
<tr><td>3</td><td colspan="2">To apply your changes and close the Report Directory Setting dialog box, click OK.</td></tr>
<tr><td>4</td><td colspan="2">To refresh the information displayed in the SnapManager Reports option, go the SnapManager Actions pane and select Refresh.</td></tr>
</table>

Event notification options

You can use either the Configuration Wizard or the Notification Settings dialog box to enable and configure the SnapManager event notification services.

Understanding SnapManager event notification options

The following event notification options can be configured from either the Configuration Wizard or from the Notification Settings dialog box.

SnapManager email notification SnapManager can notify you through email (using SMTP) about the success or failure of the following types of events:

- SnapManager backup
- Database integrity verification
- SnapManager restore
- SnapManager clone
- SnapManager configuration

SnapManager event logging If AutoSupport is enabled on the storage system, the SnapManager events can be posted to the storage system event log. This option is enabled by default. SnapManager also sends system configuration data, such as the number of databases on the system, to the storage system in the form of AutoSupport. SnapManager always sends this kind of AutoSupport information because AutoSupport is independent of SnapManager event notification settings.

AutoSupport notification If AutoSupport is enabled on both the storage system and SnapManager, technical support receives automatic email notification about any SnapManager events or storage system problems that might occur. This option is enabled by default.

The AutoSupport daemon monitors the storage system's operations and sends automatic messages to technical support to alert them to potential storage system problems. If necessary, technical support contacts you at the email address that you specified to help resolve a potential system problem.

The following information is sent to AutoSupport each time the SQL Server database is enumerated in SMSQL:

- Number of databases on the host
- Number of the SQL Server instance
- Total number of clones on the host
- A log entry each time a clone is created indicating the success or failure of the creation

The AutoSupport daemon is enabled by default on the storage system. For additional information, see the *Data ONTAP SAN Administration Guide for 7-Mode* for your version of Data ONTAP.

Limit event logging to failure events If AutoSupport is enabled on the storage system, you can limit the SnapManager events that are posted to the storage system event log and AutoSupport (if enabled for SnapManager) to failure events only. The option to limit event logging to failure events is enabled by default.

Using the Notification Settings dialog box

To configure automatic event notification settings for SnapManager, complete the following steps.

Step	Action
1	From the Actions pane, select **Notification Settings**. **Result** The Notification Settings dialog box appears. **Note:** The Configuration Wizard presents the same options in the Configure Automatic Event Notification screen.
Configure Email Notification	
2	To enable email notification, select the **Send e-mail notification** option. By default, the automatic email notification feature is disabled. **Note:** SnapManager relies on and requires an external mail host at your site to send mail. The mail host is a host that runs a mail server that listens on the SMTP port (25).
3	In the four text boxes in the top half of the tab, enter the following information: **SMTP Server** The host name or the IP address of the SMTP email server or gateway to be used. **From** The email address of the sender of the notification. By default, the name *SMSQLAutoSender* is used. To specify a sender other than the default, use following email address format: SenderName@SenderDomain.com **To** The email address of the recipient to whom the notification is to be sent. For more than one recipient, use a semicolon (;) to separate the addresses. Each recipient must be in the RecipientName@RecipientDomain.com format. **Subject** The text to be appended to the following standard subject line, which is included in all notification messages: `Backup status at` *`mm_dd_yyyy-hh.mm.ss`* `from` *MachineName* By default, `SnapManager for SQL Server` is used for the appended subject string.
4	Click **Advanced**. **Result** The Advanced Event Notification Settings dialog box appears.
Advanced Event Notification Settings	
5	To disable email notification, select the Exclude warning notification option. Enable this option if you do not want to receive emails for clone dismounts. By default, the warning email notification feature is enabled.

Step	Action
6	In the E-mail Message Content panel, select one of the following types of body messages to include in the body of the email: • Send operation results summary • Send verbose operation results **Note:** If you choose to send the operational results in summary format rather than in verbose format, you can also select the **Include SnapManager operation report as an attachment** option.
7	Click **OK** to commit your settings.
8	Click **Send a Test Email**. **Result:** SnapManager sends the email notification, using the settings you specified, and displays a notification.
Configure Event Logging and AutoSupport	
9	If you want to enable posting of SnapManager events to the storage system event log, select the **Log SnapManager events to storage system syslog** option.
10	If SnapManager event logging is enabled, you can also enable automatic email notification about any SnapManager or storage system problems to technical support. To do this, select the **Send AutoSupport Notification** option.
11	If you want to limit SnapManager event logging to failure events, select the **On failure only** option.
12	Click **OK**. **Note:** If you are using the Configuration Wizard instead of the Notification Settings dialog box, click **Next**.

Managing fractional space reservation

Viewing fractional space reservation status

In the Fractional Space Reservation Settings dialog box, use the Current Settings tab to view the current space consumption in the storage system volumes that contain LUNs that store SQL Server data or SnapInfo directories.

Drive Letter or Mountpoint

A SnapManager configuration setting for the LUN. The drive letter or NTFS mount point on which the LUN is mounted.

Fractional Reserve (%)

The amount of space reserved for overwrites on the storage system volume that contains this LUN. Expressed as a percentage of the total size of all space-reserved LUNs in the volume.

Backup Autodelete Trigger (%)

A SnapManager fractional space reservation policy setting for the storage system volume that contains the LUN. The percentage of overwrite reserve utilization that triggers automatic deletion of SQL Server backup sets.

Disable Database Trigger (%)

A SnapManager fractional space reservation policy setting for the storage system volume that contains the LUN. The percentage of overwrite reserve utilization that triggers automatic disabling of SQL Server databases.

Used Reserve

For the storage system volume that contains this LUN, the amount of overwrite reserve *in use*. Expressed in two ways: as a percentage of the total size of all space-reserved LUNs in the volume and in megabytes.

Available Reserve (MB)

For the storage system volume that contains this LUN, the amount of overwrite reserve *available*.

Snapshot Autodelete

For the storage system volume that contains this LUN, the state of the Data ONTAP Snapshot copy automatic deletion feature: enabled or disabled. If this LUN stores SQL Server data files and is contained in a storage system volume for which the Data ONTAP Snapshot copy automatic deletion feature is enabled, disable this feature on that volume or ensure that it is configured so that it does not delete SnapManager backup set components.

To view the current space consumption information about each LUN, complete the following steps.

Step	Action
1	Select **Fractional Space Reservation Settings** in the SnapManager Actions pane.

Step	Action
2	In the Current Status tab, note the space consumption status for each LUN that stores SQL Server data or SnapInfo directories. • The following columns display SnapManager configuration information: ◦ Drive Letter or Mount Point ◦ Fractional Overwrite Reserve (%) ◦ Backup Autodelete Trigger (%) ◦ Disable Database Trigger (%) The SnapManager fractional space reservation policy triggers (listed above) are not applicable to fully space-reserved LUNs. • The following columns display the fractional overwrite reserve settings and status: ◦ Used Overwrite Reserve (%) ◦ Used Overwrite Reserve (MB) ◦ Used Reserve (MB) ◦ Available Reserve (MB) ◦ Storage System Snapshot Autodelete If Fractional Overwrite Reserve (%) is 100, the LUN is contained in a fully space-reserved volume rather than a in a fractionally space-reserved volume. The information displayed in this tab is automatically refreshed every 60 seconds. Only the Drive Letter or Mount Point column displays LUN-specific information. All other columns in this tab display information that applies across the storage system volume that contains the LUN.

Step	Action
3	If the Snapshot Autodelete column is enabled, investigate the cause and take preventive action if necessary. **Attention:** If the Storage Snapshot Autodelete column is enabled, the LUN is contained in a FlexVol volume that has overwrite reserve set to less than 100 percent and that also has the Data ONTAP automatic Snapshot copy deletion feature enabled and configured to trigger when the overwrite reserve is nearly full. If SQL Server data or SnapManager SnapInfo directories are stored on LUNs contained in a volume with these characteristics, the Data ONTAP Snapshot copy automatic deletion policy might delete SQL Server backup set components. Take one of the following actions on the volume: • Disable the Data ONTAP Snapshot copy automatic delete feature. • Ensure that the Data ONTAP Snapshot copy automatic delete feature is configured in such a way that it does not delete SQL Server backup set components. For details about the `snap autodelete` storage system command, see the *Data ONTAP SAN Administration Guide for 7-Mode*. The SnapManager fractional space reservation policy includes a separate, SQL Server-aware automatic deletion feature. For details, see *Configuring fractional space reservation policies* on page 321. The SnapManager automatic deletion feature can be used in place of or in conjunction with the Data ONTAP automatic deletion feature; you can also select to disable the SnapManager automatic deletion feature.
4	To close the dialog box, click **OK.**

Configuring fractional space reservation policies

In the Fractional Space Reservation Settings dialog box, you can use the Policy Settings tab to view or customize the default policy and to configure custom policies for individual fractional space-reserved LUNs.

SnapManager automatically applies the default policy to every storage system volume that contains fractional space-reserved LUNs that store SQL Server database files or SnapInfo directories. Therefore, your storage is protected from an out-of-space condition, without requiring you to explicitly enable or configure any fractional space reservation policies.

To configure the fractional space reservation policy, complete the following steps.

Step	Action
1	Select **Fractional Space Reservation Settings** in the SnapManager Actions pane. The Fractional Space Reservation Settings window is displayed.
2	Click the **Policy Settings** tab.

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td colspan="3">Choose to specify either the default policy or a volume-specific policy</td></tr>
<tr><td rowspan="4">3</td><td colspan="2">In the left navigation tree, select the scope of the policy you want to view or change in the main panel on the right side of the tab:</td></tr>
<tr><td>If you want to view or change...</td><td>Then do this...</td></tr>
<tr><td>The default policy</td><td>In the navigation tree, select Default Policy.</td></tr>
<tr><td>A volume-specific policy</td><td>In the navigation tree, select the storage system and then the volume.</td></tr>
<tr><td colspan="3">Enable or disable fractional space reservation monitoring</td></tr>
<tr><td rowspan="3">4</td><td>If you want to...</td><td>Then do this...</td></tr>
<tr><td>Enable fractional space reservation monitoring</td><td>Select the Enable Fractional Space Reservation Monitoring check box.</td></tr>
<tr><td>Disable fractional space reservation monitoring</td><td>Clear the Enable Fractional Space Reservation Monitoring check box.</td></tr>
<tr><td colspan="3">Disable or configure automatic deletion of SQL Server backup Snapshot copies</td></tr>
<tr><td rowspan="5">5</td><td colspan="2">Use the Automatic deletion of backups panel to disable, enable, or configure automatic deletion of SQL Server backup Snapshot copies in fractional space-reserved LUNs on the volume.
Although automatic deletion of SQL Server backup Snapshot copies does not necessarily prevent an out-of-space condition on the volume, it is a best practice to enable this feature for every volume that contains fractional space-reserved LUNs that store SQL Server data.</td></tr>
<tr><td>If you want to...</td><td>Then do this...</td></tr>
<tr><td>Enable automatic deletion of SQL Server backup Snapshot copies</td><td>Select the Delete backups that include LUNs which have less than 100% overwrite reservation check box and then go to Step 8.</td></tr>
<tr><td>Disable automatic deletion of SQL Server backup Snapshot copies</td><td>Clear the Delete backups that include LUNs which have less than 100% overwrite reservation check box and to to Step 6.</td></tr>
<tr><td colspan="2">Data ONTAP includes a separate Snapshot copy automatic deletion feature. For details, see Viewing fractional space reservation status on page 318. You can use the SnapManager automatic deletion feature in place of or in conjunction with the Data ONTAP automatic deletion feature.</td></tr>
</table>

Step	Action
6	In the “Trigger point for overwrite reserve utilization” field, enter the level of overwrite reserve utilization (in percentage of total reserve) that is to trigger deletion of SQL Server backup Snapshot copies. The value must be a non-negative integer that is less than the “Trigger point for overwrite reserve utilization” value in the Automatic dismount of databases panel.
7	In the “Number of most recent backups to retail” field, enter the number of backups to be retained if automatic backup set deletion is triggered. The value must be an integer from 1 through 256 and should be based on the backup creation and verification schedule.
Configure automatic dismounting of SQL Server databases	
8	Use the Automatically dismount databases panel to configure automatic dismounting of SQL Server databases in fractional space-reserved LUNs on the volume. Because automatic deletion of SQL Server backup Snapshot copies does not necessarily prevent an out-of-space condition on the volume, SnapManager does not allow you to disable dismounting of databases for any fractional space reservation policy. In the “Trigger point for overwrite reserve utilization” field, enter the level of overwrite reserve utilization (in percentage of total reserve) that is to trigger dismounting of SQL Server databases. The value must be an integer from 0 through 99. If Snapshot copy automatic deletion is enabled, SnapManager requires that this threshold be set to a higher level than the threshold that triggers automatic Snapshot copy deletion. This ensures that Snapshot copy automatic deletion is triggered first.
Apply the changes to the default or volume-specific policy	
9	To apply your changes and close the dialog box, click **OK**.

Copyright information

Copyright © 1994–2014 NetApp, Inc. All rights reserved. Printed in the U.S.

No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademark information

NetApp, the NetApp logo, Network Appliance, the Network Appliance logo, Akorri, ApplianceWatch, ASUP, AutoSupport, BalancePoint, BalancePoint Predictor, Bycast, Campaign Express, ComplianceClock, Customer Fitness, Cryptainer, CryptoShred, CyberSnap, Data Center Fitness, Data ONTAP, DataFabric, DataFort, Decru, Decru DataFort, DenseStak, Engenio, Engenio logo, E-Stack, ExpressPod, FAServer, FastStak, FilerView, Fitness, Flash Accel, Flash Cache, Flash Pool, FlashRay, FlexCache, FlexClone, FlexPod, FlexScale, FlexShare, FlexSuite, FlexVol, FPolicy, GetSuccessful, gFiler, Go further, faster, Imagine Virtually Anything, Lifetime Key Management, LockVault, Manage ONTAP, Mars, MetroCluster, MultiStore, NearStore, NetCache, NOW (NetApp on the Web), Onaro, OnCommand, ONTAPI, OpenKey, PerformanceStak, RAID-DP, ReplicatorX, SANscreen, SANshare, SANtricity, SecureAdmin, SecureShare, Select, Service Builder, Shadow Tape, Simplicity, Simulate ONTAP, SnapCopy, Snap Creator, SnapDirector, SnapDrive, SnapFilter, SnapIntegrator, SnapLock, SnapManager, SnapMigrator, SnapMirror, SnapMover, SnapProtect, SnapRestore, Snapshot, SnapSuite, SnapValidator, SnapVault, StorageGRID, StoreVault, the StoreVault logo, SyncMirror, Tech OnTap, The evolution of storage, Topio, VelocityStak, vFiler, VFM, Virtual File Manager, VPolicy, WAFL, Web Filer, and XBB are trademarks or registered trademarks of NetApp, Inc. in the United States, other countries, or both.

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. A complete and current list of other IBM trademarks is available on the web at *www.ibm.com/legal/copytrade.shtml*.

Apple is a registered trademark and QuickTime is a trademark of Apple, Inc. in the United States and/or other countries. Microsoft is a registered trademark and Windows Media is a trademark of Microsoft Corporation in the United States and/or other countries. RealAudio, RealNetworks, RealPlayer, RealSystem, RealText, and RealVideo are registered trademarks and RealMedia, RealProxy, and SureStream are trademarks of RealNetworks, Inc. in the United States and/or other countries.

All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such.

NetApp, Inc. is a licensee of the CompactFlash and CF Logo trademarks.

NetApp, Inc. NetCache is certified RealSystem compatible.

How to send your comments

You can help us to improve the quality of our documentation by sending us your feedback.

Your feedback is important in helping us to provide the most accurate and high-quality information. If you have suggestions for improving this document, send us your comments by email to *doccomments@netapp.com*. To help us direct your comments to the correct division, include in the subject line the product name, version, and operating system.

You can also contact us in the following ways:

- NetApp, Inc., 495 East Java Drive, Sunnyvale, CA 94089 U.S.
- Telephone: +1 (408) 822-6000
- Fax: +1 (408) 822-4501
- Support telephone: +1 (888) 463-8277

Index

*__recent Snapshot names *97*, *109*
*.bak files *97*, *109*
*.fbk files *97*, *109*
*.trb files *97*, *109*
*.trn files *97*, *109*

A

allow databases on any LUN or VMDK configuration
  enabling SnapManager *294*
AlwaysOn
  SnapManager support for *17*
AlwaysOn failover cluster node
  connecting to *293*
AlwaysOn feature of SQL Server 2012
  what it enables you to do *17*
application settings
  configurable outside of the Configuration wizard *303*
archives
  disaster recovery guidelines *204*, *210*, *211*, *220*
  overview *85*
  recovering SQL Server databases *204*, *210*, *211*, *220*
  restoring databases from *204*, *210*, *211*, *220*
archiving SnapManager backup sets
  preparation for SnapVault (clustered Data ONTAP) *89*
authentication method
  SQL Server *145*
Auto Shrink option, SQL Server database *113*, *115*, *126*, *136*, *138*, *154*
Availability Group transaction log backup *126*
Availability Groups
  SnapManager support for *17*

B

backing up
  adding run command *113*, *115*, *126*, *136*, *138*, *152*, *154*
  system resources *19*, *20*, *22*, *24*, *25*, *27*
Backup and Verification tab
  Invalid database label *97*, *103*, *105*, *109*, *111*
  performing a database verification *113*, *115*, *126*, *136*, *138*, *152*, *154*
  performing a full database backup *113*, *115*, *126*, *136*, *138*, *152*, *154*
  performing a transaction log backup *113*, *115*, *126*, *136*, *138*, *152*, *154*
  scheduling a job to run later *145*
backup data
  organization of SnapManager *99*
backup management groups
  about *113*, *115*, *126*, *136*, *138*, *152*, *154*
  assigning a new full database backup to a group *113*, *115*, *126*, *136*, *138*, *152*, *154*
  changing the group assignment for an existing full database backup *113*, *115*, *126*, *136*, *138*, *152*, *154*
  using with SnapManager operations
    database verification *113*, *115*, *126*, *136*, *138*, *154*
    explicit deletion of multiple backup Snapshot copies *113*, *115*, *126*, *136*, *138*, *154*
    full database backup *97*, *109*, *113*, *115*, *126*, *136*, *138*, *154*
backup method
  Snapshot-based *97*, *103*, *105*, *109*, *111*
  stream-based *97*, *103*, *105*, *109*, *111*
backup retention. See Snapshot copies *113*, *115*, *126*, *136*, *138*, *154*
backup sets
  organization of SnapManager *99*
backup sets, SnapManager
  data organization within *97*, *103*, *105*, *109*, *111*
  guidelines for restoring *160*, *161*, *163*, *166*, *167*, *178*
  how Snapshot copies are used *7*, *11*, *14*, *17*
  naming convention for *97*, *103*, *105*, *109*, *111*
Backup Wizard
  Invalid database label *97*, *103*, *105*, *109*, *111*
  performing a database verification *113*, *115*, *126*, *136*, *138*, *152*, *154*
  performing a full database backup *113*, *115*, *126*, *136*, *138*, *152*, *154*
  performing a transaction log backup *113*, *115*, *126*, *136*, *138*, *152*, *154*
  scheduling a job to run later *145*
backups
  protecting *85*
backups, full database
  default backup settings *303*
  default verification settings *303*

before you install or upgrade *20*, *22*, *24*, *27*
bulk-logged recovery model, SQL Server
  as supported by SnapManager *160*, *161*, *163*, *166*, *167*, *178*
  definition of *7*, *11*, *14*, *17*
busy Snapshot
  avoiding during a SnapManager operation
    database verification only *113*, *115*, *126*, *136*, *138*, *154*
    full database backup with verification *113*, *115*, *126*, *136*, *138*, *154*
    when leaving database attached after verification *290*, *292*, *294*, *298*, *301*, *313–315*
  deleting
    unable to delete using SnapManager *113*, *115*, *126*, *136*, *138*, *154*
    using Data ONTAP *113*, *115*, *126*, *136*, *138*, *154*

C

centralized transaction log backups
  setting up a network location *84*
  using SnapManager share *84*
CIFS protocol, as supported by SnapManager
  access to LUN objects *204*, *210*, *211*, *220*
  cannot be used to back up or restore databases *7*, *11*, *14*, *17*
clone replica
  creating *191*
  deleting *191*
cloning
  adding run command *180*, *188*
  purpose of *180*
cloning an AlwaysOn cluster *191*
cloning databases
  VMDK *196*
cluster. See Windows cluster *28*, *36*, *40*, *43*
clusters, Windows
  SnapManager installation in existing *28*, *33*, *35*, *36*, *40*, *43*
cmdlets
  new-backup *260*
color, database icon *113*, *115*, *126*, *136*, *138*, *154*
commands
  reseed-backup *270*
Configuration Wizard
  about *62*, *63*, *65*, *80*, *81*, *83*
  how it stores databases on volumes *62*, *63*, *65*, *80*, *81*, *83*
  migrating databases back to local disks *62*, *63*, *65*, *80*, *81*, *83*
  migrating databases from local disks to LUNs *62*, *63*, *65*, *80*, *81*, *83*
  moving multiple SnapInfo directories to a single SnapInfo directory *62*, *63*, *65*, *80*, *81*, *83*
  Reconfig database label *62*, *63*, *65*, *80*, *81*, *83*
  when to run the Configuration Wizard *62*, *63*, *65*, *80*, *81*, *83*
connecting to a node *60*
connecting to a server *60*
copy-based restore method *160*, *161*, *163*, *166*, *167*, *178*
creating reports directory *60*

D

data
  configuration plan, creating *44*, *46*, *54*
  management, supported by SnapManager *7*, *11*, *14*, *17*
  organization of SnapManager backup *99*
data protection
  overview *85*
database cloning
  purpose *180*
database consistency checker. See DBCC *14*
database icon color *113*, *115*, *126*, *136*, *138*, *154*
database label
  Invalid *97*, *103*, *105*, *109*, *111*
  Reconfig *62*, *63*, *65*, *80*, *81*, *83*
database verification
  avoiding busy Snapshot *290*, *292*, *294*, *298*, *301*, *303*, *313–315*
  information you need to specify *113*, *115*, *126*, *136*, *138*, *152*, *154*
  scheduling the job to run later *145*
  using the Backup and Verification tab *113*, *115*, *126*, *136*, *138*, *152*, *154*
  using the Backup Wizard *113*, *115*, *126*, *136*, *138*, *152*, *154*
databases
  backing up on VMDKs *193*
  preparation for SnapVault (clustered Data ONTAP) *89*
  preparing to replicate volumes *88*
databases, SQL Server
  backing up after renaming *160*, *161*, *163*, *166*, *167*, *178*
  backing up before installing SnapManager *19*, *20*, *22*, *24*, *25*, *27*

maximum per LUN *44*, *46*, *54*
maximum per SQL Server computer *44*, *46*, *54*
maximum per SQL Server instance *44*, *46*, *54*
restoring from archive *204*, *210*, *211*, *220*
DataFabric Manager server requirements *23*
datasets
what they are *90*
DBCC
as used by SnapManager Backup *97*, *103*, *105*, *109*, *113*, *115*, *126*, *136*, *138*, *152*, *154*
as used by SnapManager Restore *160*, *161*, *163*, *166*, *167*, *178*
as used by the Configuration Wizard *62*, *63*, *65*, *80*, *81*, *83*
definition of *7*, *11*, *14*, *17*
drive letters required for verifying a backup *97*, *103*, *105*, *109*, *111*
specifying settings *113*, *115*, *126*, *136*, *138*, *152*, *154*, *290*, *292*, *294*, *298*, *301*, *303*, *313–315*
disaster recovery
general guidelines *204*, *210*, *211*, *220*
restoring system databases *204*, *210*, *211*, *220*
using NDMP or dump archives
general procedure *210*, *220*
guidelines *210*, *220*
using other SQL Server backup sets *204*, *210*, *211*, *220*
using SnapMirror replication
general procedure *210*, *220*
guidelines *210*, *220*
using SnapVault archives
guidelines *210*, *220*
distribution database, definition of *14*
drive letters required for DBCC *97*, *109*
dump command, storage system
archiving LUNs that contain SnapManager backup sets *7*, *11*, *14*, *17*

E

enabling SnapManager
allow databases on any LUN or VMDK configuration *294*
Enterprise Manager.
See SQL Server Enterprise Manager *97*, *105*, *109*, *111*

F

FCP
LUN access protocol *19*, *20*, *22*, *24*, *25*, *27*
federated backup *14*
filer-side license.
See per-storage system license *19*, *20*, *22*, *24*, *25*, *27*
filters
selecting backups *58*
selecting servers *58*
fractional space reservation
viewing status *318*
fractional space reservation policies
configuring *321*
full database backup
information you need to specify *113*, *115*, *126*, *136*, *138*, *152*, *154*
scheduling the job to run later *145*
selecting databases at the instance level
about *113*, *115*, *126*, *136*, *138*, *154*
using the Backup and Verification tab *113*, *115*, *126*, *136*, *138*, *154*
using the Backup Wizard *113*, *115*, *126*, *136*, *138*, *154*
Snapshot copy-based backup method *97*, *103*, *105*, *109*, *111*
stream-based backup files *97*, *103*, *105*, *109*, *111*
stream-based backup method *97*, *103*, *105*, *109*, *111*
using the Backup and Verification tab *113*, *115*, *126*, *136*, *138*, *152*, *154*
using the Backup Wizard *113*, *115*, *126*, *136*, *138*, *152*, *154*
volume-wide backups *97*, *103*, *105*, *109*, *111*
what to do if the backup fails *113*, *115*, *126*, *136*, *138*, *152*, *154*
full recovery model, SQL Server
as supported by SnapManager *160*, *161*, *163*, *166*, *167*, *178*
definition of *7*, *11*, *14*, *17*

G

group Managed Service Account requirements *23*
guidelines
for disaster recovery *204*, *210*, *211*, *220*
for disaster recovery using archives *204*, *210*, *211*, *220*
for performing a SnapManager Restore operation
choosing the type of restore to perform *160*, *161*, *163*, *166*, *167*, *178*
for restoring from a SnapManager backup set *160*, *161*, *163*, *166*, *167*, *178*
for volume sizing *44*, *46*, *54*

H

host systems
  requirements for Windows *20*
how you use SnapManager *14*

I

installation
  requirements for group Managed Service Accounts *23*
  requirements for SnapManager service account *23*
  requirements for SQL Server service account *23*
installation prerequisites *20*, *22*, *24*, *27*
installation process
  in existing cluster *28*, *33*, *35*, *36*, *40*, *43*
  preinstallation *19*, *20*, *22*, *24*, *25*, *27*
installing SnapManager
  components installed by SnapManager *28*
  on Windows Failover Cluster *28*
installing SnapManager on a standalone system
  in unattended mode *28*, *33*, *35*, *36*, *40*, *43*
installing SnapManager on a Windows cluster
  disk requirements for *28*, *33*, *35*, *36*, *40*, *43*
  system configuration requirements for *28*, *33*, *35*, *36*, *40*, *43*
interactive mode
  uninstalling SnapManager *28*, *33*, *35*, *36*, *40*, *43*
  upgrading SnapManager *28*, *33*, *35*, *36*, *40*, *43*
Invalid database label *97*, *109*
iSCSI
  LUN access protocol *19*, *20*, *22*, *24*, *25*, *27*

L

label, database
  Invalid *97*, *103*, *105*, *109*, *111*
  Reconfig *62*, *63*, *65*, *80*, *81*, *83*
licenses
  Windows host system requirements *19*, *20*, *22*, *24*, *25*, *27*
  Windows host system requirements for SnapManager *20*
log shipped databases
  database restore of *97*, *103*, *105*, *109*, *111*, *160*, *161*, *163*, *166*, *167*, *178*, *290*, *292*, *294*, *298*, *301*, *303*, *313–315*
  transaction log backup of *97*, *103*, *105*, *109*, *111*
  up-to-the-minute restore of *160*, *161*, *163*, *166*, *167*, *178*
LUN access protocol *20*, *22*, *24*, *27*
LUN size calculations *44*, *46*

M

management groups.
  See backup management groups *113*, *115*, *126*, *136*, *138*, *152*, *154*
Management Studio.
  See SQL Server Management Studio *97*, *103*, *105*, *109*, *111*
managing transaction log backups *126*
master database, definition of *14*
MDAC (Microsoft Data Access Components) version *20*, *22*, *24*, *27*, *113*, *115*, *126*, *136*, *138*, *154*
mirrored volumes
  recovering SQL Server databases from. See SnapMirror replication, restoring from *204*, *210*, *211*, *220*
model database, definition of *14*
mount point
  limitations *44*, *46*, *54*
  limitations in a clustered environment *44*, *46*, *54*
Mounted volume naming conventions *44*, *46*
mounted volumes in SnapManager *44*, *46*
msdb database, definition of *14*
multiple-instance cluster *28*, *36*, *40*, *43*

N

NDMP-based backup utility
  archiving LUNs that contain SnapManager backup sets *7*, *11*, *14*, *17*
new-backup cmdlet
  syntax and description *260*
node
  connecting to *60*
NTBackup *14*
NTBackup, using to back up system resources *20*, *22*, *24*, *27*
NTFS *14*
NVRAM *14*

O

online Snapshot. See snapshot-based *14*

P

per-SQL Server license *20*, *22*, *24*, *27*
per-storage system license *20*, *22*, *24*, *27*
point-in-time restore *160*, *161*, *163*, *166*, *167*, *178*
preinstallation process *20*, *22*, *24*, *27*
preupgrade process *20*, *22*, *24*, *27*
processes
  installation in existing cluster *28*, *33*, *35*, *36*, *40*, *43*
  preinstallation *19*, *20*, *22*, *24*, *25*, *27*
protocol
  CIFS *7*, *11*, *14*, *17*, *210*, *211*, *220*
  FCP *19*, *20*, *22*, *24*, *25*, *27*
  iSCSI *19*, *20*, *22*, *24*, *25*, *27*
  NDMP *7*, *11*, *14*, *17*

R

rebuildm.exe (rebuild master) *210*, *220*
Reconfig database label *62*, *63*, *65*, *80*, *81*, *83*
recovery models, Microsoft SQL Server
  definition of *7*, *11*, *14*, *17*
remote administration server
  other requirements for *19*, *20*, *22*, *24*, *25*, *27*
remote verification server
  drive letters required for DBCC *97*, *103*, *105*, *109*, *111*
Report directory, SnapManager
  changing the location *290*, *292*, *294*, *298*, *301*, *303*, *313–315*
  default location *290*, *292*, *294*, *298*, *301*, *303*, *313–315*
  option to remove during uninstallation *28*, *33*, *35*, *36*, *40*, *43*
reports directory
  creating *60*
reports, SnapManager operational
  types of *200*, *201*
requirements
  SnapMirror with SnapManager *86*, *147*
  transaction log space *44*, *46*, *54*
  Windows host system *20*
reseed-backup command
  syntax and description *270*
restore method
  copy-based *160*, *161*, *163*, *166*, *167*, *178*
  including run command *160*, *161*, *163*, *166*, *167*, *178*
  snapshot-based *160*, *161*, *163*, *166*, *167*, *178*
  stream-based *160*, *161*, *163*, *166*, *167*, *178*
restore operations
  organization of SnapManager backup data for *99*
restore Snapshot copies
  about *160*, *161*, *163*, *166*, *167*, *178*
  deleting *160*, *161*, *163*, *166*, *167*, *178*
restoring databases
  after renaming a database *160*, *161*, *163*, *166*, *167*, *178*
  using a point-in-time restore *160*, *161*, *163*, *166*, *167*, *178*
  using an up-to-the-minute restore *160*, *161*, *163*, *166*, *167*, *178*
  with log-shipping implemented *97*, *103*, *105*, *109*, *111*, *160*, *161*, *163*, *166*, *167*, *178*, *290*, *292*, *294*, *298*, *301*, *303*, *313–315*
retention of backup Snapshot copies. See Snapshot copies *113*, *115*, *126*, *136*, *138*, *154*
rolling snapshots, SnapDrive
  advantages over increased SnapManager backups *86*, *147*
  to supplement automatic replication *86*, *147*
run command
  using with back up *113*, *115*, *126*, *136*, *138*, *152*, *154*
  using with clone *180*, *188*
  using with restore *160*, *161*, *163*, *166*, *167*, *178*
Run Command After Operation feature
  and generic backup naming (__recent suffix) *97*, *103*, *105*, *109*, *111*
  running a script from a UNC path *145*

S

SAN boot LUN *44*, *46*
scheduling
  running a backup or verification job for later *145*
selecting databases with filters *58*
selecting servers with filters *58*
server
  connecting to *60*
server-side license. See per-SQL Server license *20*, *22*, *24*, *27*
service account, SnapManager
  in workgroup mode *28*, *33*, *35*, *36*, *40*, *43*
service accounts
  SnapManager requirements *23*
  SQL Server requirements *23*
sets
  organization of SnapManager backup *99*
setup.exe utility *210*, *220*
simple recovery model, SQL Server

as supported by SnapManager *160*, *161*, *163*, *166*, *167*, *178*
definition of *7*, *11*, *14*, *17*
SnapDrive
overview
when to use as opposed to SnapManager *14*
rolling Snapshots *86*, *147*
storage requirements *44*, *46*, *54*
SnapInfo directory, SnapManager
moving multiple SnapInfo directories to a single SnapInfo directory *62*, *63*, *65*, *80*, *81*, *83*
naming conventions for *97*, *103*, *105*, *109*, *111*
rules for storing *62*, *63*, *65*, *80*, *81*, *83*
SnapManager Backup *97*, *103*, *105*, *109*, *111*
system databases stopped by SnapManager *62*, *63*, *65*, *80*, *81*, *83*
transaction log backup file names *97*, *103*, *105*, *109*, *111*
user databases detached by SnapManager *62*, *63*, *65*, *80*, *81*, *83*
SnapManager
application overview
command- line interface *14*
how it stores databases on volumes *62*, *63*, *65*, *80*, *81*, *83*
how it uses Snapshot copies as backups *14*
how it uses Snapshots as a restore precaution *160*, *161*, *163*, *166*, *167*, *178*
how it works with other backup methods *14*
how it works with SnapDrive *14*
maximum configurations supported by *44*, *46*
new functionality with this version *14*
relationship with other Data ONTAP-based components *14*
what it does *14*
what it does not do *14*
when to use it *14*
when to use SnapDrive instead *14*
where you install and run it *14*
application settings
configurable from the Configuration Wizard *62*, *63*, *65*, *80*, *81*, *83*
configurable outside of the Configuration Wizard *290*, *292*, *294*, *298*, *301*, *313–315*
backup sets
data organization within *97*, *109*
guidelines for restoring from *160*, *161*, *163*, *166*, *167*, *178*
naming convention for *97*, *109*
data management functions *7*, *11*, *14*, *17*
functions supported *7*, *11*, *14*, *17*
how you use *7*, *11*, *14*, *17*
installation in existing cluster *28*, *33*, *35*, *36*, *40*, *43*
installing
reinstalling *28*, *36*, *40*, *43*
installing on a standalone system *33*
installing or upgrading
installing on a standalone system *28*, *36*, *40*, *43*
option to remove Report directory during uninstall *28*, *36*, *40*, *43*
reinstalling *28*, *36*, *40*, *43*
uninstalling *28*, *36*, *40*, *43*
upgrading *28*, *36*, *40*, *43*
reinstalling *33*
terminology *7*, *11*, *14*, *17*
uninstalling *33*
upgrading *33*
user interface
command-line interface (CLI)
clone-backup *223*
SnapManager Backup
deleting oldest Snapshot copies with *97*, *103*, *105*, *109*, *111*
limitations *97*, *103*, *105*, *109*, *111*
managing the number of Snapshot copies and backup sets *97*, *103*, *105*, *109*, *111*
performing a database verification
about *97*, *109*
default verification settings *290*, *292*, *294*, *298*, *301*, *313–315*
information you need to specify *113*, *115*, *126*, *136*, *138*, *154*
using the Backup and Verification tab *113*, *115*, *126*, *136*, *138*, *154*
using the Backup Wizard *113*, *115*, *126*, *136*, *138*, *154*
performing a full database backup
about *97*, *109*
default backup settings *290*, *292*, *294*, *298*, *301*, *313–315*
default verification settings *290*, *292*, *294*, *298*, *301*, *313–315*
information you need to specify *113*, *115*, *126*, *136*, *138*, *154*
using the Backup and Verification tab *113*, *115*, *126*, *136*, *138*, *154*
using the Backup Wizard *113*, *115*, *126*, *136*, *138*, *154*
performing a transaction log backup
about *97*, *109*

default backup settings *290*, *292*, *294*, *298*, *301*, *313–315*
information you need to specify *113*, *115*, *126*, *136*, *138*, *154*
using the Backup and Verification tab *113*, *115*, *126*, *136*, *138*, *154*
using the Backup Wizard *113*, *115*, *126*, *136*, *138*, *154*
requirements *97*, *103*, *105*, *109*, *111*
starting or scheduling jobs
database verification *113*, *115*, *126*, *136*, *138*, *154*
full database backup *113*, *115*, *126*, *136*, *138*, *154*
transaction log backup *113*, *115*, *126*, *136*, *138*, *154*
using backup management groups *113*, *115*, *126*, *136*, *138*, *152*, *154*
what it does *97*, *103*, *105*, *109*, *111*
when to back up your databases *97*, *103*, *105*, *109*, *111*
SnapManager backup data
organization of *99*
SnapManager Reports
about *200*, *201*
Report directory
changing the location *290*, *292*, *294*, *298*, *301*, *313–315*
default location *290*, *292*, *294*, *298*, *301*, *313–315*
option to remove during uninstall *28*, *36*, *40*, *43*
SnapManager Restore
about *160*, *161*, *163*, *166*, *167*, *178*
cluster failure during *160*, *161*, *163*, *166*, *167*, *178*
cluster group state during *160*, *161*, *163*, *166*, *167*, *178*
Snapshot copies created as a precaution *160*, *161*, *163*, *166*, *167*, *178*
SnapManager support *44*, *46*
SnapMirror
overview *85*
requirements with SnapManager *86*, *147*
SnapMirror replication of SnapManager backups
how it works *86*, *147*
preparing to replicate volumes *88*
scheduling considerations *86*, *147*
supplementing with rolling Snapshots *86*, *147*
SnapMirror replication, restoring from
disaster recovery guidelines *204*, *210*, *211*, *220*
recovering SQL Server databases *204*, *210*, *211*, *220*
Snapshot copies
about *7*, *11*, *14*, *17*
backup Snapshot copies
about *14*
automatically deleting the oldest *97*, *109*
explicitly deleting multiple *113*, *115*, *126*, *136*, *138*, *154*
naming conventions *97*, *109*
creation methods, when to use *7*, *11*, *14*, *17*
how SnapManager uses *7*, *11*, *14*, *17*
maximum allowed per volume *97*, *103*, *105*, *109*, *111*
maximum number *7*, *11*, *14*, *17*
restore Snapshot copies
about *160*, *161*, *163*, *166*, *167*, *178*
explicitly deleting *160*, *161*, *163*, *166*, *167*, *178*
naming convention *160*, *161*, *163*, *166*, *167*, *178*
Snapshot copy-based
backup method *97*, *103*, *105*, *109*, *111*
restore method *44*, *46*, *54*, *160*, *161*, *163*, *166*, *167*, *178*
technology provided by Data ONTAP *7*, *11*, *14*, *17*
SnapVault
integrating with SnapManager (Data ONTAP operating in 7-Mode) *92*
limitations *90*
overview *85*
preparing to archive backups (clustered Data ONTAP) *89*
requirements *90*
understanding support for *90*
space requirements for transaction logs *44*, *46*
SQL Server
*.bak files *97*, *103*, *105*, *109*, *111*
*.trn files *97*, *103*, *105*, *109*, *111*
calculating database size *44*, *46*, *54*
recovery models
definition of *14*
requirements for service account *23*
See also SQL Server database *14*, *17*
See also SQL Server database[SQL Server *7*, *11*
See also SQL Server Enterprise Manager[SQL Server *7*, *11*, *14*
See also SQL Server Management Studio[SQL Server *7*, *11*
SnapManager rules for storing databases *62*, *63*, *65*, *80*, *81*, *83*
system databases
definition of *14*

stopped by SnapManager *44*, *46*, *62*, *63*, *65*, *80*, *81*, *83*
user databases
definition of *14*
detached by SnapManager *44*, *46*, *62*, *63*, *65*, *80*, *81*, *83*
SQL Server 2000
verifying an SQL Server 2005 database *111*
SQL Server 2005
MDAC version *19*, *20*, *22*, *24*, *25*, *27*
no user databases on root LUN *97*, *103*, *105*, *109*, *111*
setup.exe utility *204*, *210*, *211*, *220*
verifying an SQL Server 2000 database *111*
SQL Server 2012
enhancements provided by AlwaysOn feature *17*
SQL Server authentication method
limitations when scheduling a remote verification server *145*
SQL Server database
Auto Shrink option *113*, *115*, *126*, *136*, *138*, *152*, *154*
SQL Server Enterprise Manager
backing up transaction logs in a SnapManager environment *7*, *11*, *14*
detecting SnapManager transaction log backups *97*, *105*, *109*, *111*
viewing SnapManager full database backup files *97*, *105*, *109*, *111*
SQL Server instance
as a remote administration server
other requirements for *20*, *22*, *24*, *27*
as a remote verification server
drive letters required for DBCC *97*, *109*
maximum databases per *44*, *46*, *54*
maximum per SQL Server computer *44*, *46*, *54*
selecting databases at the instance level
full database backup *113*, *115*, *126*, *136*, *138*, *154*
transaction log backup *113*, *115*, *126*, *136*, *138*, *154*
SQL Server Management Studio
backing up transaction logs in a SnapManager environment *7*, *11*, *14*, *17*
detecting SnapManager transaction log backups *97*, *103*, *105*, *109*, *111*
viewing SnapManager full database backup files *97*, *103*, *105*, *109*, *111*
SQL__* SnapInfo subdirectory names *97*, *109*
sqlsnap__* snapshot names *97*, *109*
status, fractional space reservation
viewing *318*
stream-based operations
backup file names *97*, *103*, *105*, *109*, *111*
backup method *97*, *103*, *105*, *109*, *111*
restore method *44*, *46*, *54*, *160*, *161*, *163*, *166*, *167*, *178*
system databases, Microsoft SQL Server
backing up *97*, *103*, *105*, *109*, *111*
definition of *7*, *11*, *14*, *17*
distribution database, definition of *7*, *11*, *14*, *17*
master database, definition of *7*, *11*, *14*, *17*
migrating to LUNs *62*, *63*, *65*, *80*, *81*, *83*
model database, definition of *7*, *11*, *14*, *17*
msdb database, definition of *7*, *11*, *14*, *17*
restoring *204*, *210*, *211*, *220*
stopped by SnapManager *62*, *63*, *65*, *80*, *81*, *83*
tempdb database, definition of *7*, *11*, *14*, *17*
system resources, backing up *20*, *22*, *24*, *27*
systems
requirements for Windows host *20*

T

tempdb database, definition of *14*
transaction log backup
information you need to specify *113*, *115*, *126*, *136*, *138*, *152*, *154*
of a log-shipped database *97*, *103*, *109*, *111*
scheduling the job to run later *145*
selecting databases at the instance level
about *113*, *115*, *126*, *136*, *138*, *154*
using the Backup and Verification tab *113*, *115*, *126*, *136*, *138*, *154*
using the Backup Wizard *113*, *115*, *126*, *136*, *138*, *154*
SnapManager backup data *97*, *103*, *105*, *109*, *111*
using the Backup and Verification tab *113*, *115*, *126*, *136*, *138*, *152*, *154*
using the Backup Wizard *113*, *115*, *126*, *136*, *138*, *152*, *154*
what to do if the backup fails *113*, *115*, *126*, *136*, *138*, *152*, *154*
transaction log backups
centralizing *84*
managing *126*
using SnapManager *84*
transaction logs
rules for storing *62*, *63*, *65*, *80*, *81*, *83*
space estimation *44*, *46*, *54*

volume requirements for *44*, *46*, *54*

U

unattended mode
installing SnapManager *28*, *33*, *35*, *36*, *40*, *43*
uninstalling SnapManager *28*, *33*, *35*, *36*, *40*, *43*
upgrading SnapManager *28*, *33*, *35*, *36*, *40*, *43*
uninstalling SnapManager
before you uninstall *28*, *33*, *35*, *36*, *40*, *43*
in interactive mode *28*, *33*, *35*, *36*, *40*, *43*
in unattended mode *28*, *33*, *35*, *36*, *40*, *43*
option to remove Report directory *28*, *33*, *35*, *36*, *40*, *43*
up-to-the-minute restore *160*, *161*, *163*, *166*, *167*, *178*
upgrading SnapManager
converting VLD-type virtual disks to LUNs *28*, *33*, *35*, *36*, *40*, *43*
Data ONTAP requirement *28*, *33*, *35*, *36*, *40*, *43*
in interactive mode *28*, *33*, *35*, *36*, *40*, *43*
in unattended mode *28*, *33*, *35*, *36*, *40*, *43*
Microsoft SQL Server requirement *28*, *33*, *35*, *36*, *40*, *43*
user databases, Microsoft SQL Server
backing up *97*, *103*, *105*, *109*, *111*
definition of *7*, *11*, *14*, *17*
detached by SnapManager *62*, *63*, *65*, *80*, *81*, *83*
migrating to LUNs *62*, *63*, *65*, *80*, *81*, *83*
using SnapManager *14*

V

VDisk__* SnapInfo subdirectory names *97*, *109*
verification settings
changing *151*
configuring *151*
per-instance *151*
VLD-type virtual disks *28*, *36*, *40*, *43*
VMDK
cloning databases *196*
setting up *192*
VMDK disks
creating *192*
VMDKs
backing up databases on *193*
volume size
assessing *44*, *46*, *54*
guidelines *44*, *46*, *54*
requirements for database files *44*, *46*, *54*
requirements for transaction logs *44*, *46*, *54*
transaction log sizing *44*, *46*, *54*
volume-wide backups *97*, *109*
volumes
preparing for replication *88*
volumes, storage system
maximum per single database *44*, *46*, *54*
maximum per SQL Server computer *44*, *46*, *54*
Vservers
See SVMs

W

Windows authentication
requirements for SnapManager service account *23*
Windows cluster
cluster failure during a restore *160*, *161*, *163*, *166*, *167*, *178*
cluster group state during a restore *160*, *161*, *163*, *166*, *167*, *178*
disk requirements for *28*, *33*, *35*, *36*, *40*, *43*
maximum size supported by SnapManager *28*, *33*, *35*, *36*, *40*, *43*
multiple-instance *28*, *33*, *35*, *36*, *40*, *43*
system configuration requirements *28*, *33*, *35*, *36*, *40*, *43*
Windows Failover Cluster
installing SnapManager on *28*
Windows host system requirements
drive letters required for DBCC *97*, *103*, *105*, *109*, *111*
SnapManager in workgroup mode *28*, *33*, *35*, *36*, *40*, *43*
SnapManager licenses *19*, *20*, *22*, *24*, *25*, *27*
Windows host systems
requirements for *20*
workgroup mode
SnapManager service account requirements *23*
Windows *33*
workgroup mode, Windows *28*, *36*, *40*, *43*