

SnapManager® for Microsoft® SQL Server 2000 Best Practices

Gerson Finlev | Network Appliance, Inc. | February, 2005 | TR 3323

TECHNICAL REPORT

Network Appliance, a pioneer and industry leader in data storage technology, helps organizations understand and meet complex technical challenges with advanced storage solutions and global data management strategies.

Table of Contents

List of Figures and Tables

1. Introduction

- 1.1. Factors That Affect SQL Server Scalability
- 1.2. Network Appliance Storage Devices, Scalability, and High Availability
- 1.3. SQL Server 2000 Configurations with FlexVol
- 1.4. Network Appliance SnapManager for SQL Server 2000
- 1.5. SnapManager for SQL Server 1.5 Enhancements

2. Database(s) Logical and Physical Layout

- 2.1. System Databases
- 2.2. Consolidation Planning Phase One: Current SQL Server Instances
- 2.3. Consolidation Planning Phase Two: Future SQL Server Instance(s)

3. Sizing

4. Preparing the Host and NetApp Storage Device

- 4.1. Filer Requirements
- 4.2. NetApp Storage Device Option Settings

5. Installing Network Appliance SnapDrive 3.x

- 5.1. SnapDrive 3.x Installation Tips

6. Creating and Managing LUNs with the SnapDrive GUI

7. Installing and Configuring Network Appliance SMSQL

- 7.1. General Info
- 7.2. SQL Server 2000 Resource Use and Concurrent Database Backup
- 7.3. Database Consistency Check
- 7.4. SnapManager for SQL Server 2000 in a Microsoft Cluster Environment

8. Snapshot

- 8.1. Snapshot Copies Created by SnapManager for SQL Server 2000
- 8.2. Snapshot Copies of LUNs for Write Access
- 8.3. Disk Space Consumed by Read/Write Snapshot Copies
- 8.4. Space Reservation

9. Backing Up and Restoring SQL Server Databases with SMSQL

- 9.1. Backup Recommendations
- 9.2. Archiving Backups

10. Using SMSQL with Network Appliance SnapMirror

- 10.1. Checklist for Configuring SnapManager and SnapMirror

11. Management Tools

- 11.1. Snapshot Reserve Usage Monitoring
- 11.2. Disk Space Usage Monitoring
- 11.3. Filer Event Monitoring
- 11.4. Performance Monitoring and Tuning—Set of Basic Counters
- 11.5. Terminal Server or Remote Desktop

12. Summary

13. Ethernet and FCP Technology Diagrams

Appendix A. Database's Logical and Physical Layout

- A.1. Common Database Layouts
- A.2. Large Databases with Sequential I/O Access
- A.3. Very Large Database with High I/O Performance
- A.4. Several Databases on One Windows Server

Appendix B. General Consolidation Principles

Appendix C. Checklist, Cross-Reference, Q&A

Appendix D. Additional Resources

List of Figures

Figure 1) Database setup and recommendations overview
 Figure 2) Sample current SQL Server information
 Figure 3) I/O throughput at peak load
 Figure 4) Reference platform (Windows 2000)
 Figure 5) Recommended database placement
 Figure 6) SnapDrive installation wizard
 Figure 7) SnapDrive service account
 Figure 8) Specifying a LUN path
 Figure 9) Concurrent backup screen
 Figure 10) Maximum number of work threads
 Figure 11) Bypassing database verification
 Figure 12) Snapshot name list (SnapDrive)
 Figure 13) Transaction rate during 24 hours
 Figure 14) SMSQL and SnapMirror
 Figure 15) SnapMirror status in SnapDrive
 Figure 16) SnapMirror status in FilerView
 Figure 17) LUN error due to space limitations
 Figure 18) Database's available space
 Figure 19) Available LUN space
 Figure 20) Syslog messages in FilerView
 Figure A1) Basic structure of a SQL Server database
 Figure A2) Database layout with two LUNs
 Figure A3) Database layout using two volumes
 Figure A4) Database layout with multiple filer groups
 Figure A5) Multiple databases
 Figure B1) Before consolidating
 Figure B2) Storage consolidation
 Figure B3) Complete consolidation

1. Introduction

This paper is intended to be a best-practice guide and is for experienced SQL Server 2000 administrators who have read the *Network Appliance™ SnapDrive™ v.3.x Installation, Administration Guide* and the *SnapManager for SQL Server 2000 v.1.5 (SMSQL) Installation and Administration Guide*. Readers of this best-practice guide should have a solid understanding of the SQL Server 2000 storage architecture and administration as well as SQL Server 2000 backup and restore concepts. The recommendations found in this document are best practices to assist with the design and configuration of a SnapManager for SQL Server 2000 environment.

Note: Both the SnapDrive and SnapManager for SQL Server 2000 Installation and Administration Guides can be found at <http://now.netapp.com> (username and password required).

Microsoft SQL Server often manages databases utilized by mission-critical applications. It can be disruptive and costly to businesses when employees are not able to process new orders or other database-dependent transactions. The physical storage has to be able to scale with the database's just-in-time storage requirements. The storage has to be highly available, including the availability to quickly restore to a previously known state.

This best practice guide will focus on how to integrate NetApp storage and the SQL Server's storage engine that manages the physical storage. All recommendations will be generic and not specific for a certain application. Applications that utilize SQL Server as a back end will often

have specific requirements determined by the application design, which cannot be included in this technical report.

This technical report will discuss best practices for a database's physical layout when different I/O access patterns are generated by different application workloads, for example, OLTP and DSS.

Important functional differences between SMSQL 1.5 and SMSQL 1.0 will be clearly marked throughout this technical report.

1.1. Factors That Affect SQL Server Scalability

SQL Server databases can grow rapidly in size while processing transactions, and the transaction rate can increase over time. As the databases grow larger, extra storage will be required, and as the transaction rate increases, the storage platform must be able to support the increases in I/O throughput. With increased database size, it becomes increasingly difficult to complete time-consuming tape backup operations in a reasonable amount of time. In the case of an outage it can take days to restore service, assuming all of the backup tapes are available and error-free.

In order to design and implement highly available and scalable SQL Server architectures, system managers must look well beyond the availability of their server hardware. SQL Server scalability is often impacted by the limited scalability of storage devices. Today's storage devices must be able to increase physical storage without interrupting any applications using the storage device, and the storage system must be able to load balance all I/O requests over available physical disk drives. SQL Server availability is often dependent on how well the storage device handles failed components such as failed disk drives, memory, or processors. Managing these failed storage components reliably is critical to today's storage devices.

1.2. Network Appliance Storage Devices, Scalability, and High Availability

NetApp storage devices are uniquely scalable and by design highly available. NetApp storage devices scale from the smallest 2TB device to the largest 64TB devices by easily adding disk drives or changing controllers; the saved file structures will not be influenced by the added resources. NetApp storage devices are designed to be highly available. Traditional single-parity RAID technology offers protection from a single failed disk drive, but modern disk drive capacity has increased significantly, and bit error correction rates have stayed the same. The traditional parity RAID technology protection is in most cases good enough, but configurations that require very high availability should use NetApp RAID Double Parity, or RAID-DP. RAID-DP will protect against secondary error events during RAID reconstructs caused by a failed disk drive. RAID-DP is discussed in TR3298: [NetApp Data Protection: Double-Parity RAID for Enhanced Data Protection with RAID-DP](#).

1.3. SQL Server 2000 Configurations with FlexVol

FlexVol, introduced with Data ONTAP 7G, is an excellent technology that makes it possible to optimize storage use by logically partitioning NetApp volumes (**aggregates**) into smaller logical volumes (**FlexVol volumes**). FlexVol greatly enhances the usability of NetApp storage devices in many SQL Server environments:

Many SQL Server environments support small databases with very high I/O rates, which require many more disk drives than the physical database size require. FlexVol makes it possible to create an aggregate with many disk drives and logically partition the aggregate into smaller FlexVol volumes. This makes it possible to create smaller FlexVol volumes that support a high I/O rate spread over many disk drives.

An aggregate supporting multiple FlexVol volumes makes it possible to let several smaller SQL Server environments running on dedicated servers share a single traditional NetApp volume.

SMSQL 1.5 supports six SQL Server instances (SMSQL 1.0 supports four instances). FlexVol makes it possible to let an aggregate support multiple SQL Server instances.

A Snapshot™ copy of a database in a FlexVol volume can easily be cloned for testing or development purposes.

More information about FlexVol can be found in:

TR3356: Introduction to Data ONTAP 7G Release

TR3347: A Thorough Introduction to FlexClone™ Volumes

1.4. Network Appliance SnapManager for SQL Server 2000

Network Appliance SnapManager for Microsoft SQL Server 2000 is a complete data management solution that enables Microsoft SQL Server to leverage the many powerful and specialized features inherent to Network Appliance storage technology.

Benefits at a glance:

Quickly back up and restore entire SQL Server database(s) using Network Appliance Snapshot™ technology.

The Network Appliance SnapDrive Microsoft Management Console (MMC) application provides an easy-to-use graphical interface to simplify the management of NetApp disk, storage, and Snapshot resources.

This technology allows great consolidation of SQL Server 2000 resources.

Network Appliance storage technology offers industry-leading performance.

SQL Server database(s) stored in Snapshot copies can be mirrored to one or more locations for archival or disaster recovery purposes.

SQL Servers are able to serve more data and are easier to scale and manage.

Flexible pools of storage can provide storage resources for multiple SQL Server instances, and disks can be added on-the-fly as storage needs grow.

The ability to quickly back up and restore online SQL Server database(s) in Snapshot copies is the primary purpose of SnapManager for Microsoft SQL Server 2000. SMSQL dramatically reduces the time it takes to back up and recover SQL Server database(s) from online Snapshot files.

Use FlexVol to quickly clone a database located in a FlexVol (See TR3347: [A Thorough Introduction to FlexClone Volumes](#)).

1.5. SnapManager for SQL Server 1.5 Enhancements

SMSQL 1.5 has enhancements requested by users of SMSQL 1.0; the main enhancements are:

Sharing two LUNs. SMSQL 1.0 only allowed sharing of one LUN. This has been expanded to include two LUNs (see Figure A2, Appendix A).

Restoring selected databases when sharing LUNs. SMSQL 1.0 would restore all databases located in the same LUN. SMSQL 1.5 will now only restore selected databases. For example, if all files belonging to 20 databases are located in one or two LUNs, then the DBA is now able to select specific databases to restore.

MSCS and verification server. SMSQL 1.0 required a remote non-MSCS-aware verification server. This requirement has been loosened. The virtual SQL Server instance can now check the consistency of its online database, and it is possible *but not recommended* for the DBA to skip verification of the backed-up database.

Backup management group is a new concept with SMSQL 1.5. It is now possible to create daily, weekly, or the standard backup management group, which makes it more flexible for a DBA to create specific backup plans for each database.

Run command after operation is a new function. This command can be setup to be executed after a backup or verification; common use of this command is to archive backup sets.

For a complete overview of SnapManager for Microsoft SQL Server 2000, see **SnapManager 1.5 for Microsoft SQL Server: Installation and System Administration Guide**, downloadable from the NOW™ site.

2. Database(s) Logical and Physical Layout

The best practices presented in this guide are general principles, not guidelines for specific environments. Applications using the SQL Server back end for query processing will often have special resource dependency requirements that are outside the discussion of this paper. This paper will discuss general requirements for OLTP (random access) and DSS (sequential access) types of applications.

Designing and implementing a SQL Server database is fairly intuitive, but can be critical for future growth and performance and important for optimal use of SMSQL. Understanding the relationship among database files, LUNs, and NetApp filer volumes is important for optimal use of SMSQL. This section will discuss database file objects and explore various layouts of SQL Server databases and ways to utilize LUNs that can be supported by SMSQL.

Both system and user databases deploy the same basic structure; therefore, the given description will cover system databases and user-defined databases. The focus is not the logical structure of database files but a database's physical files and how they are accessed. A database consists of two or three different physical file types with different content and use. The file types use the following default file extensions:

.ldf is a transaction log. A database has at least one transaction log that contains information necessary to recover all transactions in a database.

.mdf is the database's primary data file. Every database has one primary data file that keeps track of all other database files and internal database structure, in addition to storing data.

.ndf is the secondary data file. A database can have zero or more secondary data files that contain only data.

Data files can be logically grouped into **file groups**. A file group consists of one or more data files.

Migrating SQL Server databases involves copying database files and executing SQL statements, which are what SnapManager for SQL Server 2000 does when it migrates databases to NetApp storage; it works on a database's physical files. The following illustrates the difference between a primary data file and a secondary data file; given a 'test' database consisting of three files:

'E:\[folder]\test_data.mdf,' 'E:\[folder]\test_data_1.ndf,' and 'E:\[folder]\test_log.ldf.' The administrator of database 'test' wants to move the database to a new Windows® server. The storage device on the new server is using 'E,' which is the same drive letter as was on the old server. Since the logical locations of the physical files haven't really changed, then the database attach statement can just point to the primary file, which contains all necessary information to find all other files:

```
sp_attach_db 'test',
'E:\[folder]\test_data.MDF'
```

If the storage device on the new server is mapped to a different driver letter ('H'), then the above statement would fail; all database files have to be directly referenced in the attach database statement:

```
sp_attach_db 'test',
'H:\[folder]\test_data.MDF',
'H:\[folder]\test_data_1.NDF',
'H:\[folder]\test_log.LDF'
```

Figure 1 gives an overview of a recommended physical layout dependent on the database type and number of databases. Appendix A is recommended reading with a detailed discussion around databases' physical layout. The application vendor will often have specific application-dependent recommendations that have to be understood in the context of NetApp storage devices.

	Small OLTP ¹ Database	Many Small OLTP Databases	Very Large OLTP Database	Small Data Warehouse ²	Large Data Warehouse
Single LUN	X	X		X	
Two LUNs ³	X	X		X	
Three or more LUNs			X		X
Single Volume ^{4 5}	X	X		X	
Multiple ⁶ Volumes			X		X
The LUN Can Be Shared by Many Databases ⁷	X	X		X	
The LUNs			X		X

Cannot Be Shared by Any Other Database					
Multiple Data Files in File Group ⁸			X	X	X
Different File-Groups for Index and Table ⁹			X	X	X

Figure 1 Database setup and recommendations overview.

How to Use the Information in Figure 1

Placement of SQL Server files on LUNs and volumes depends on the number of databases, database types, and the sizes of the databases.

The physical placement of database files depends on several related factors, but there are some basic rules that have to be followed:

Several databases can **share a single LUN** when all database files are placed in the one LUN. For example, a LUN has been mounted on 'X,' and all files (log and data files) belonging to MyDB are placed in 'X,' in which case all files belonging to MyDB_1 can also be located on 'X' (see Appendix A.1).

Several databases are able to share **two LUNs** when all database files are placed in the two LUNs. For example, a LUN has been mounted on 'D,' and another LUN has been mounted on 'L.' MyDB's transaction log file is placed in 'L,' and MyDB's data files are placed in 'D,' in which case MyDB_1's transaction log can also be placed in 'L,' and its data files can be placed in 'D' (see Appendix A.1).

When files belonging to a database are placed in **more than two LUN**, then no other database can use any of the LUNs. For example, **if three LUNs** are mounted on 'X,' 'Y,' and 'Z' and MyDB has its log file located in 'X,' some data file(s) in 'Y,' and other data files in 'Z,' then 'X,' 'Y,' and 'Z' cannot be used by any other database (see Appendixes A.2 and A.3).

Use Figure 1 to focus on possible database placements using SMSQL with NetApp storage devices. The table is easiest to use when the analyzing process follows these steps:

1. What is the number of databases managed by a Windows system?¹⁰ If the system has to manage more databases than available LUNs, then multiple databases have to share one or two **LUNs**.
2. Is the database type **OLTP** or **data warehouse**? The I/O access patterns that these two types need to support is very different; the I/O load on OLTP is mostly characterized by random access, and the I/O load on data warehouse consists of mostly sequential access. Hence, only mix the two different access patterns when the I/O load is fairly low. When the I/O load is high, then it is best to use two different volumes, one for OLTP databases and another volume for data warehouses.

3. Is one of the database types a **large OLTP database**? A system will usually only support one or a few large databases, in which case available drive letters are not an issue. If the database contains several large tables, then create a file group with four data files for each table. If the different tables have different access patterns—for instance, one table is mostly read intensive, and the other table is mostly write intensive—then use two different volumes.
4. Is the database type a **large data warehouse**? It is very important to place tables in a data warehouse in file groups with multiple data files. Multiple data files increase the potential I/O rate, because SQL Server uses the multiple files to create a table across all data files and to issue multiple I/O requests per data file in a file group when scanning a table or sections of a table.
5. Estimate I/O load per volume. After deciding on the physical layout, then each volume must be sized according to space requirements and estimated I/O rate, which will be discussed in Section 3.

The outcome of analyzing the SQL Server environment(s) results in data useful for the next two planning phases, but let us discuss the system databases first.

2.1. System Databases

Except for the **tempdb**, system databases are fairly small (less than 20MB) with low I/O rate. Tempdb is recreated every time SQL Server is restarted and should not be backed up. There is one tempdb per SQL Server instance. Since the size of tempdb can become very large and it may support high I/O rates, it is a good practice to locate the tempdb on a dedicated volume. The size and I/O load is a function of:

Number of databases managed by the SQL Server instance

Types of SQL statements executed

Number of users connected to the SQL Server instance

Before allocating a dedicated volume for tempdb, its size and load should be well understood, which is most easily done on a similar SQL Server environment in production.

2.2. Consolidation Planning Phase One: Current SQL Server Instances

All information necessary for designing the new setup is collected while current and possible new SQL Server environment(s) are analyzed.

Figure 2 illustrates the result of collecting information for phase one. Multiple SQL Server instances can execute on each Windows server, and each SQL Server instance can manage many databases. *Database Type* shows the I/O access pattern each database is supporting ("OLTP" is mostly random, "DSS" is mainly sequential, and "Test" is unknown). Access pattern is important when deciding on the relationship among database, LUN, and filer volume. The *High Availability Index* is a value that shows how important the database is for the company and is a deciding factor when deciding which databases should be managed by a clustered environment. If a disaster recovery site is being considered, then *High Availability Index* is also a useful value to determine which database should be located on a volume configured for mirroring. Information in Figure 2 is high-level information that is necessary for grouping databases and database placement on the storage device. Current database size in Figure 2 combined with throughput data in Figure 3 is necessary when sizing the storage devices for:

Initial size

Future need

Snapshot copies

Disk drives needed to service expected I/O rate

It can be difficult to get data describing I/O throughput during peak loads (holidays, end of quarter, or other peak throughput times), but the data is very important to size storage devices correctly for current load and future loads at peak times. Sizing will be discussed in more detail in Section 3.

Windows Server "DDZ" in Computer Lab 111

SQL Server Instance	Database	Database Size	Database Type	High Availability Index
SQL Server 1	DB 1	39GB	DSS	3
	DB 2	25GB	OLTP	1
SQL Server 2	DB 3	200GB	OLTP	2
	DB 4	75MB	Test	4
	DB 5	50MB	OLTP	2
2 SQL Server Instances	5 databases	265GB		

Figure 2 Sample current SQL Server information.

Database	Read per Second	Write per Second	Change Rate per Month	Growth Rate per Month
DB 1	5MB	1MB	15%	5%
DB 2	8MB	1.5MB	10%	8%
DB 3	20MB	6MB	12%	6%
DB 4	0.5MB	0.1MB	5%	40%
DB 5	1.5MB	0.2MB	20%	10%
Total	35MB	8MB	32GB	16GB

Figure 3 I/O throughput at peak load.

2.3. Consolidation Planning Phase Two: Future SQL Server Instance(s)

The classification (Figure 2) is used to create the new environment(s), which is done in two steps. The first step is to create a reference platform (Figure 4); the second step is to group databases according to:

High Availability Index

Database Type

Size

The reference platform in Figure 4 illustrates a clustered environment, including a NearStore® system for archiving backed-up databases that can be used in the case of a disaster. Remember, the number of LUNs a clustered system (one or four nodes) can mount is the same as a single nonclustered system, one LUN per available drive letter, about 23 LUNs.

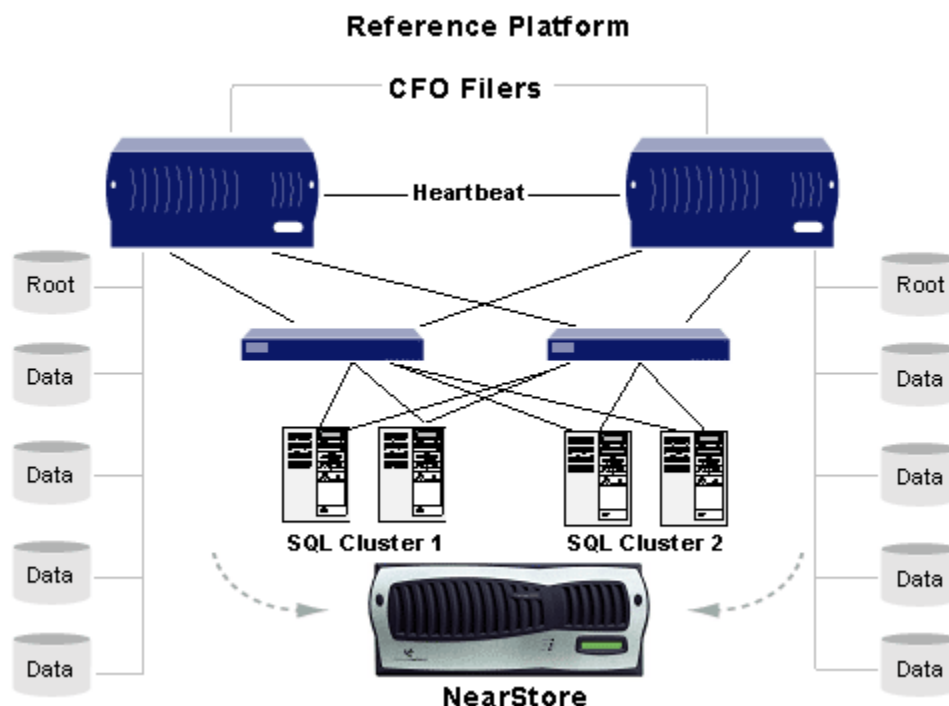


Figure 4 Reference platform (Windows 2000).

Figure 5 illustrates an example of a new SQL Server instance in a clustered configuration with database placements in six different volumes. The example database, 'DB 27,' is large (1TB) and is utilizing two volumes. It consists of two large tables (and multiple smaller tables); one large table supports mostly random access, and the other table supports mostly sequential access. Configuration for access pattern is mainly important when the storage system has to maintain high I/O throughput with good service times. Appendix A discusses database file placement on NetApp volumes. The advantages and disadvantages are discussed for each supported physical layout as shown in Figure 1.

Consolidation of multiple SQL Server instances can be very complicated and is, somewhat, discussed in Appendix B. The main issue when consolidating multiple SQL Server instances is to create relationships among SQL Server instance -> databases -> LUNs -> volumes -> disk drives. It is not recommended to let more than a single Windows server create and manage LUNs on a single volume.

SQL Cluster 1-FAS940C

SQL Server Instance	Database	Database Size	Database Type	High Availability Index	LUN Mount (Device)	Volume #
---------------------	----------	---------------	---------------	-------------------------	--------------------	----------

Network Appliance Inc.

Quorum Disk					Q	1
Cluster Server 1	System databases	3GB			Z	1
	DB 1	270GB	OLTP	1	E	2
	DB 2	8GB	OLTP	2	F	2
	DB 3	170MB	OLTP	1	G	2
	DB 4	2GB	OLTP	2	H	2
	DB 17	270MB	OLTP	2	I	2
	DB 19	43MB	OLTP	1	J	2
	DB 21	28MB	OLTP	2	O, P	2
	DB 22	320MB	OLTP	2	O, P	2
	DB 23	77MB	OLTP	2	J	2
SMSQL	SnapInfo				S	3
Cluster Server 2	System databases	3GB			X	4
	DB 10	100MB	DSS	2	K	5
	DB 27	1112GB	DSS	2	L, M, R	5, 6
	DB 38	2GB	DSS	1	N	5
SMSQL	SnapInfo				Y	4

Figure 5 Recommended database placement.

The example in Figure 5 shows system databases, including tempdb, are placed in the same volume as the quorum disk or SnapInfo. The I/O load on the quorum disk is low and will not influence tempdb. SnapInfo's I/O load is greatest when a transaction log is being backed up (sequential write), which in most cases will not create an I/O problem for tempdb, but make sure to monitor I/O performance during production (see Section 11).

3. Sizing

Estimating storage requirements can be very complicated, especially when a sizing project involves server consolidation. The basic concept consists of several related objects:

1. System databases
 - a. Focus on tempdb
2. User-defined databases
 - a. Data files
 - b. Transaction log files
3. SnapInfo
 - a. Metadata
 - b. Full backup
 - c. Backup of transaction log
 - d. Streaming-based backup of system databases
 - e. Streaming-based backup of user-defined databases

4. Snapshot copy creation

Sizing storage for SQL Server databases consists of two related sizing areas:

Sizing for space

Sizing for throughput

Sizing for throughput often requires more disk drives than the actual size requires.

Each of these sizing objects is thoroughly discussed in Technical Report 3363: [SQL Server 2000 Sizing and Capacity Planning Guidelines](#). The report includes a link to a sizing Excel spreadsheet that can be used as a template for sizing consolidated databases.

4. Preparing the Host and NetApp Storage Device

There are several things that need to be done to prepare a NetApp storage device and the SQL Server host to create a reliable system with optimal performance. Refer to the latest *SnapDrive* and *SnapManager for SQL Server 2000 Administration Guides* to ensure the proper licenses and options are enabled on the NetApp storage device.

4.1. Filer Requirements

Data ONTAP™ 6.4.4 6.5.1, or above is required for SnapDrive 3.x. For current requirements please visit <http://now.netapp.com>.

FCP license must be enabled if FCP LUNs will be used.

iSCSI license must be enabled if iSCSI LUNs will be used.

SnapRestore® license must be enabled for SnapManager to restore backups.

SnapMirror® licenses must be enabled if SnapMirror will be used.

SnapManager license must be enabled.

At least one Intel® gigabit card for connectivity (copper or fiber).

4.2. NetApp Storage Device Option Settings

SnapDrive will, for reliability and stable operation, set all required options on the NetApp storage device during its installation or while creating a LUN.

The volume-based option *minra* is one option the administrator can set according to the I/O workload. Set *minra* to *on* if the volume's access pattern is random and off if the access pattern is sequential.

Confirm that *wafn.nt_admin_priv_map_to_root* is on.

Users running SnapManager for Microsoft SQL Server 2000 must be logged on with an account that has server role sysadmin on the SQL Server instance.

5. Installing Network Appliance SnapDrive 3.x

SnapDrive interfaces directly with Network Appliance storage devices and the Microsoft Windows 2000 or 2003 volume manager to facilitate the management of LUNs provisioned on NetApp storage.

Network Appliance SnapDrive 3.x is composed of a Win32 service, a Microsoft Management Console (MMC) Windows management application, and the device drivers for iSCSI and FCP.

SnapDrive provides a layer of abstraction and a consistent, transparent interface between Network Appliance filers and Windows applications. The Fibre Channel (FCP) and iSCSI resources that are managed by SnapDrive appear to the Windows 2000 or 2003 host server and its applications as locally attached disks.

In addition to providing a management interface and presenting NetApp LUNs to the Windows 2000 or 2003 operating system as basic disks, SnapDrive is capable of managing all of the functions related to Snapshot. For an overview of SnapDrive, see TR3197, Technical Overview of SnapDrive, at the following location: [Network Appliance - Technical Overview of SnapDrive](#).

Prior to launching the installation of SnapDrive, use the checklist below to help eliminate the potential for errors or delays during or after the installation.

Resolve any hardware, cabling, or network issues or other errors.

Make sure all of the necessary software and printed or electronic documentation (found on <http://now.netapp.com>) is organized and nearby before beginning.

Configure DNS, hostname, and IP address-related services:

- Verify that all related systems, including filers, servers, and clients, are configured to use an appropriately configured DNS server (for more information, see TR 3124 at [Integrating NetApp Filers with the Microsoft Windows Active Directory Environments](#)).
- Manually add the filers' host names to DNS.
- Enable, configure, and test RSH access on the filer(s) for administrative access (for more information, see Chapter 2 of the Data ONTAP 6.x Storage Management Guide).

License all of the necessary protocols and software on the filer.

Configure the filer(s) to join the Windows NT® or Windows 2000 Active Directory domain by using the FilerView® administration tool or by running `filer> cifs setup` on the console (for more information, see Chapter 3 of the Data ONTAP 6.x File Access Management Guide).

Make sure the filers' date and time are synchronized with the Active Directory domain controllers. This is necessary for Kerberos authentication. A time difference greater than five minutes will result in failed authentication attempts.

Verify that all of the service packs and hotfixes are applied to the Microsoft Windows 2000 Server or Microsoft Windows 2003 Server and Microsoft SQL Server 2000.

5.1. SnapDrive 3.x Installation Tips

Prior to installing the SnapDrive application, the iSCSI or Fibre Channel HBA drivers must be installed. The SnapDrive 3.x installation wizard provides an option to install the drivers before proceeding with the installation.

Note:

Do not proceed with the SnapDrive 3.x installation until the iSCSI or HBA drivers are installed and the host rebooted.

When using iSCSI, it is **not** necessary to install the HBA drivers.

When using FCP, it is **not** necessary to install the iSCSI drivers.

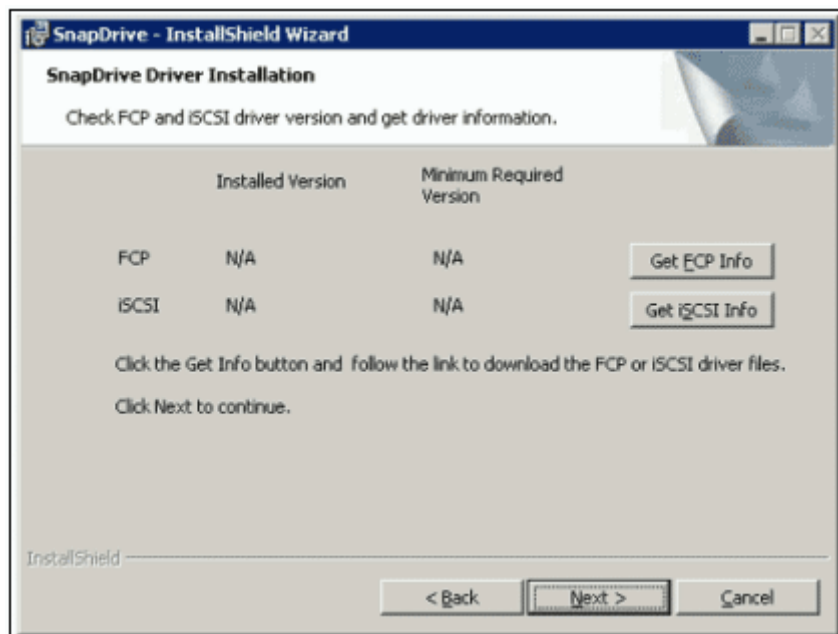


Figure 6 SnapDrive installation wizard.

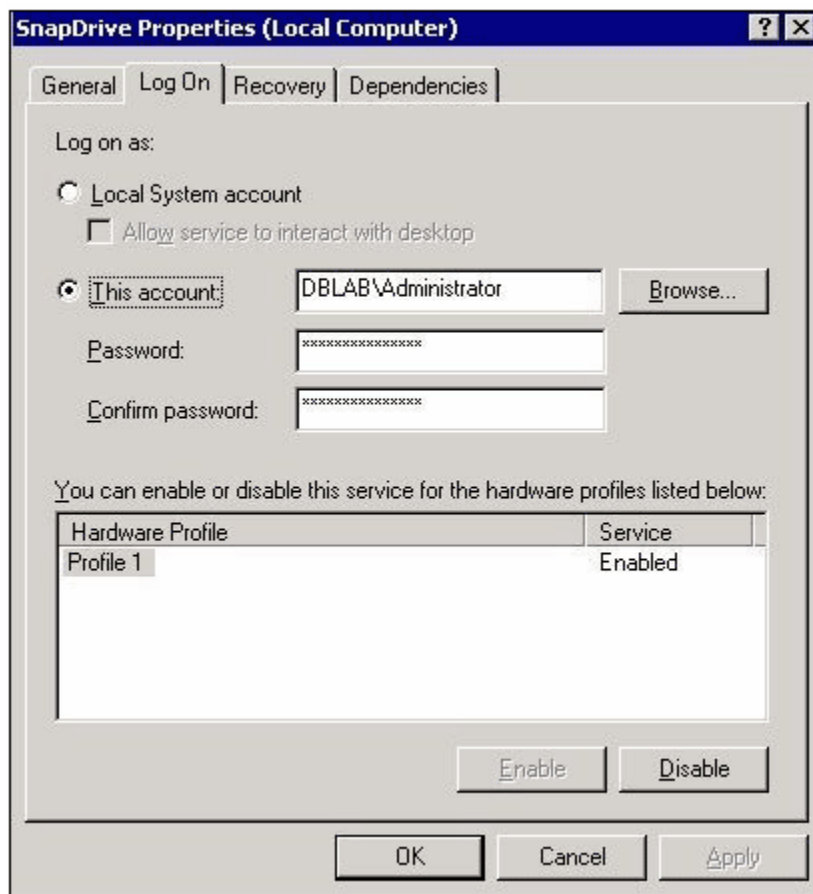


Figure 7 SnapDrive service account.

Once the device driver is installed, select the same account used by the Microsoft SQL Server services when selecting the service account for the SnapDrive and SnapManager for Microsoft SQL Server services.

When creating or configuring the properties for the domain service account, select the *Password never expires* checkbox. Doing so protects the account and service from failing due to user password policies.

Reminder: It's important to make certain the service account has the following permissions:

Read and write or full control access to the locations on the filer in which LUNs will be created (likely if it's already a member of the administrator's group).

RSH access to the filer(s). For more information on configuring RSH, see the *Data ONTAP Administration Guide*.

6. Creating and Managing LUNs with the SnapDrive GUI

Once installed, SnapDrive can be used to create LUNs on Network Appliance filers for use by Windows 2000 and Windows 2003 hosts.

The process of creating a LUN is virtually the same, and there is no distinction at the host application level. LUNs are LUNs on a NetApp storage device that are accessed over either TCP/IP (for iSCSI) or Fibre Channel (for FCP) disk access protocols.

General Tips for Creating LUNs:

When specifying a UNC path to a share of a volume to create a LUN, use IP addresses instead of host names. This is particularly important with iSCSI, as host-to-IP name resolution issues can interfere with the locating and mounting of iSCSI LUNs during the boot process.

Always use SnapDrive to create LUNs for use with Windows to avoid the complexities inherent in Fibre Channel.

Calculate disk space requirements to accommodate for data growth, Snapshot copies, and space reservations.

Leave automatic Snapshot scheduling off as configured by SnapDrive. To turn off automatic Snapshot activities, use FilerView or the following example command:

```
filer> vol options voldata1 nosnap on
```

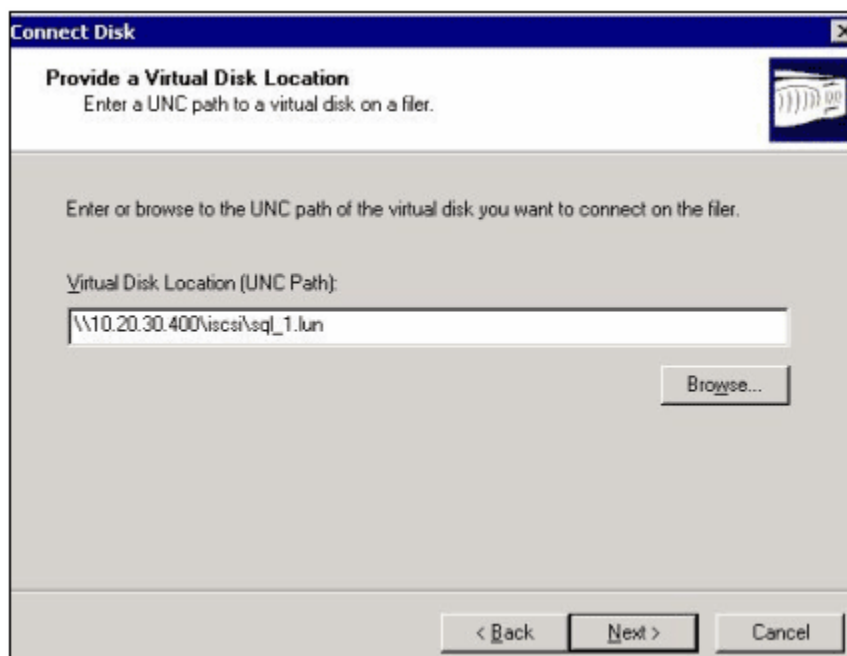


Figure 8 Specifying a LUN path.

Important Tips for Using Fibre Channel:

Verify that all of the hardware and software meets the supported system requirements. The latest hardware and software requirements for filer platforms, host platforms, Fibre Channel switches, and third-party software are available on the NOW Web site. Log on to the NOW Web site at <http://now.netapp.com> and view the SAN support matrix for the latest information and updates.

For more information on configuring Fibre Channel between Windows hosts and filers, please see the *Host Bus Adapter Installation and Setup Guide for Fibre Channel Protocol on Windows*.

7. Installing and Configuring Network Appliance SMSQL

7.1. General Info

After following the steps outlined in the SnapDrive 3.x Installation Guide and as listed in the previous sections of this paper, proceed with the installation of SnapManager for SQL Server 2000. The SnapManager for SQL Server application is a powerful tool that allows for migrating system and user databases or some of a database's files as well as controlling backup schedules and managing Snapshot. It is important to protect access to SMSQL, since the altering its settings could produce undesirable results.

7.2. SQL Server 2000 Resource Use and Concurrent Database Backup

When SMSQL backs up one or more databases managed by SQL Server, then SQL Server requires internal resources (work threads) to manage each database that is being backed up. If not enough resources are available, the backup operation will fail. Two related SMSQL settings control resources used by SQL Server: '*maximum number of databases to backup concurrently*' (Figure 9) and '*maximum number of work threads*' (Figure 10).

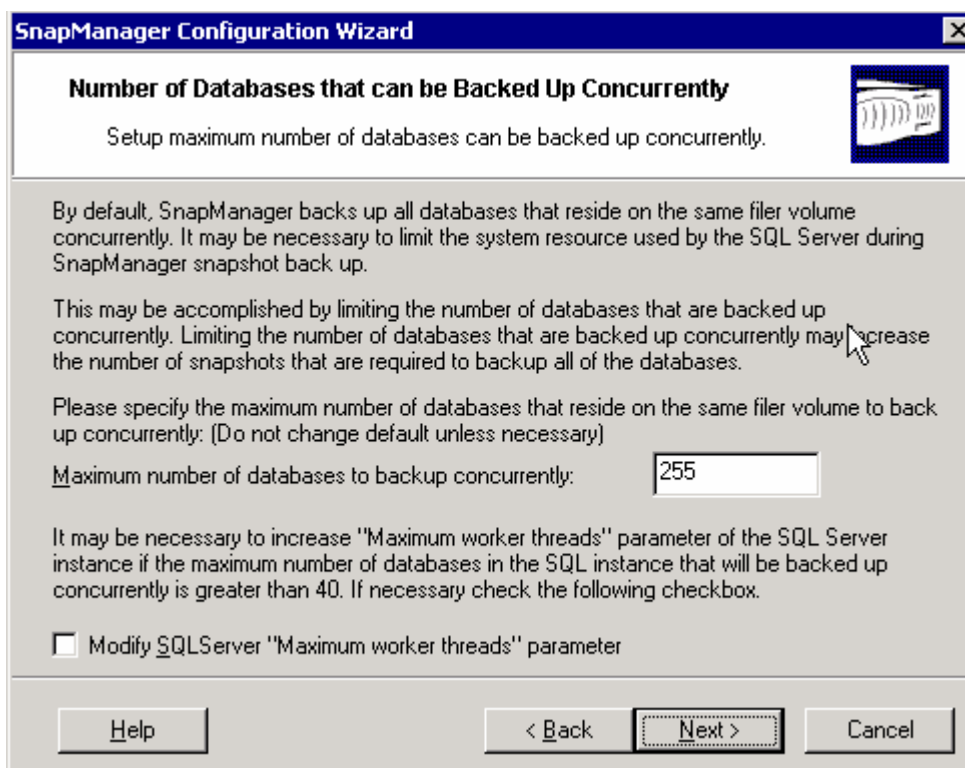


Figure 9 Concurrent backup screen.

It is important to understand the effect of changing the two settings:

The effect of changing '*maximum number of databases to backup concurrently*' is most easily explained by an example. The DBA configures the system to back up 98 databases concurrently, which ought to be possible, because the maximum value is 255, but the operation fails because SQL Server doesn't have enough resources to manage 98 databases concurrently. The DBA decides to test with a different setting, and when the maximum value has been decreased to 50 the backup succeeds with no errors. Because the maximum has been changed to 50, SMSQL will create two Snapshot copies. The first Snapshot copy will contain the full backups of the first 50 databases, and the second Snapshot copy will contain the full backups of the last 48 databases.

SQL Server uses work threads for its internal processing, including managing Snapshot backup of each database; it uses about two or more work threads per database. Therefore, the DBA might decide to change '*maximum number of work threads*' instead of '*maximum number of databases*.' Work threads used by SQL Server for managing databases being backed up by SMSQL are on top of other work threads managing user workloads (user connections, query processing, checkpoints, replication, and many other operations). Each work thread uses memory resource. Therefore, if the server has enough memory resources, then the DBA could have increased the current maximum number of work threads with another 200 work threads, and concurrent backup of the 98 databases would probably have succeeded. SQL Server has to be restarted after the value for 'work threads' is changed for the new value to take effect.

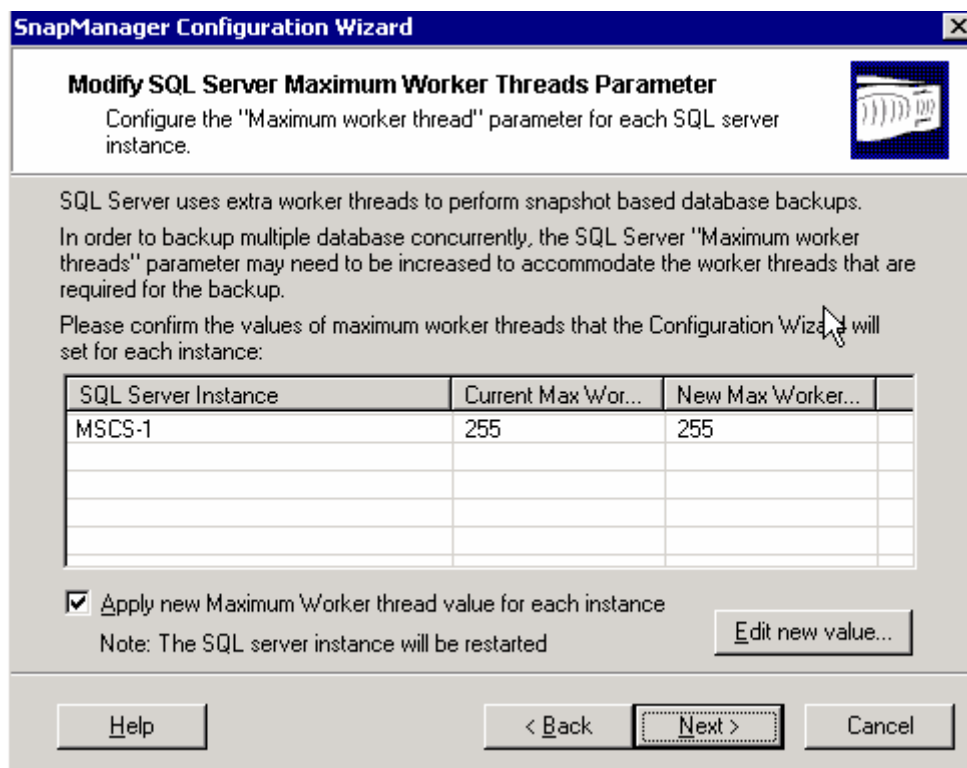


Figure 10 Maximum number of work threads.

7.3. Database Consistency Check

It is strongly recommended to verify all databases' consistency before and after migration, before and after a database has been backed up. It is also recommended to execute the verification process during off-peak time with little load on the production system or to offload the verification process to another SQL Server instance running on another server that is mostly dedicated to this purpose.

Databases can be very large, so large that the verification process takes so much time that it is not practical to check the consistency every time the database is backed up, in which case it is recommended to check the consistency of the database once a week or more and to check the consistency of every backed-up transaction log.

Many database administrators have experienced how problematic and time-consuming it is when an inconsistent database has been restored. After much checking the DBA realizes that the database was inconsistent before it was backed up. Hence, it is not always convenient to check the database's consistency frequently, but it is strongly recommended. SMSQL will not let a DBA restore a backed-up database that hasn't been checked for inconsistencies by default, but SMSQL does have an option that let a database be restored even though it has not been consistency checked. The DBA has to acknowledge, during the restore process, that a safety net is being bypassed, as shown in Figure 11.

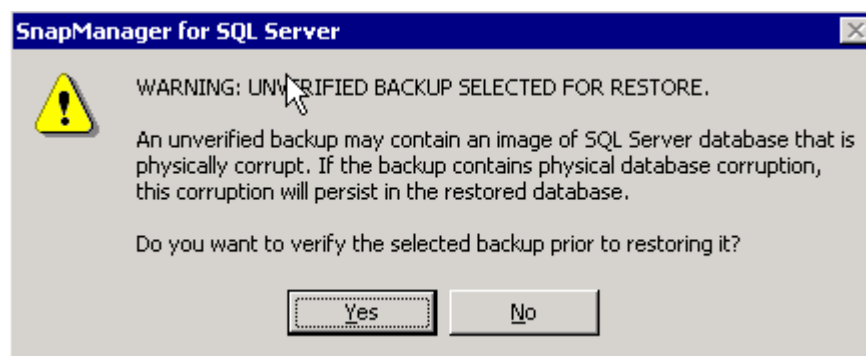


Figure 11 Bypassing database verification.

7.4. SnapManager for SQL Server 2000 in a Microsoft Cluster Environment

When configuring a Microsoft cluster environment, begin the setup by following the instructions in *Network Appliance SnapDrive v.3.x Installation and Administration Guide*. Make certain that the cluster functions correctly by using the Cluster Manager to move cluster groups from node to node. After verifying the move group works correctly, proceed to install SQL Server followed by SMSQL. The result will be one cluster group per virtual SQL Server instance. Each cluster group will contain at least three LUNs: one LUN for system databases, one LUN for user-created databases, and one LUN for SnapInfo.

For each virtual SQL Server:

1. From SnapDrive, create a cluster-aware LUN that is located in the cluster group that will contain SQL Server—for example, **SQL Group 1** (a new cluster group can be created from SnapDrive or from the Cluster Manager).
2. Create a second cluster-aware LUN for user/application-created database(s) in cluster group **SQL Group 1**.
3. Create a third cluster-aware LUN for SnapInfo in cluster group **SQL Group 1**.
4. Install the virtual server in **SQL Group 1** using the LUN for system databases.
5. Make the LUN for user database(s) a resource for the virtual server. From Cluster Manager:
 - a. *Open SQL Group 1.*
 - b. *Right click the virtual SQL Server.*
 - c. *Properties.*
 - d. *Dependency.*
 - e. *Make the LUN appear in the dependency list.*

- f. Make sure that the LUN is available by creating a dummy database from SQL Server's Enterprise Manager (delete the database).

6. Install SMSQL as a local application.

If more virtual SQL Servers will be installed, this can be done from each node or from a single node, and then each cluster group moved to the correct node after the virtual SQL Server has been installed and configured as illustrated above.

Verification of backed up databases must be done by a nonclustered SQL Server instance either installed on the local Windows server or on another Windows server that can be a dedicated verification server. The reason is simply because a virtual SQL Server cannot be used for verifying a database's consistency. The LUN in the Snapshot copy will be restored during the verification process, and the temporary restored LUN cannot be a cluster resource and therefore cannot be included in SQL Server's dependency list. A virtual SQL Server instance can be used for verifying the online database.

8. Snapshot

It is beyond the scope of this paper to explore Snapshot in great detail. However it is necessary to understand the fundamentals of the unique Network Appliance Snapshot functionality and how it relates to Microsoft SQL Server 2000.

Network Appliance Snapshot backups occur in a matter of seconds, and each consumes only the amount of disk space that has changed since the last Snapshot copy was created. Thus they consume minimal disk space while providing up to 255 online point-in-time images using Data ONTAP 6.4 or later. The amount of disk space consumed by an individual Snapshot copy is determined by two factors:

The rate data changes within file systems (in MB/sec or MB/hour, for example)

The amount of time that elapses between creation of Snapshot copies

The measure of change (in megabytes, gigabytes, etc.) that occurs in the file system between creation of Snapshot copies is called the delta. The total amount of disk space required by a given Snapshot copy equals the delta changes in the file system and a small amount of Snapshot metadata.¹

8.1. Snapshot Copies Created by SnapManager for SQL Server 2000

When SnapManager for SQL Server 2000 creates Snapshot backups, it names them according to the server name, date, and time. The one exception is the most recent Snapshot copy, which ends with "recent" instead of a date and time. When another Snapshot backup occurs, it assumes the "recent" name, and the former one is renamed to reflect its original date and time. Use SnapDrive to see all Snapshot copies created for a volume. Metadata is copied to SnapInfo when a Snapshot copy is created by SMSQL.

When SnapManager for SQL Server 2000 creates Snapshot backups, it names them according to the, type (SnapInfo or sqlsnap), server name, date, and time. The one exception is the most recent Snapshot copy, which ends with "recent" instead of a date and time. When another Snapshot backup occurs, it assumes the "recent" name, and the former one is renamed to reflect its original date and time (Figure 12).

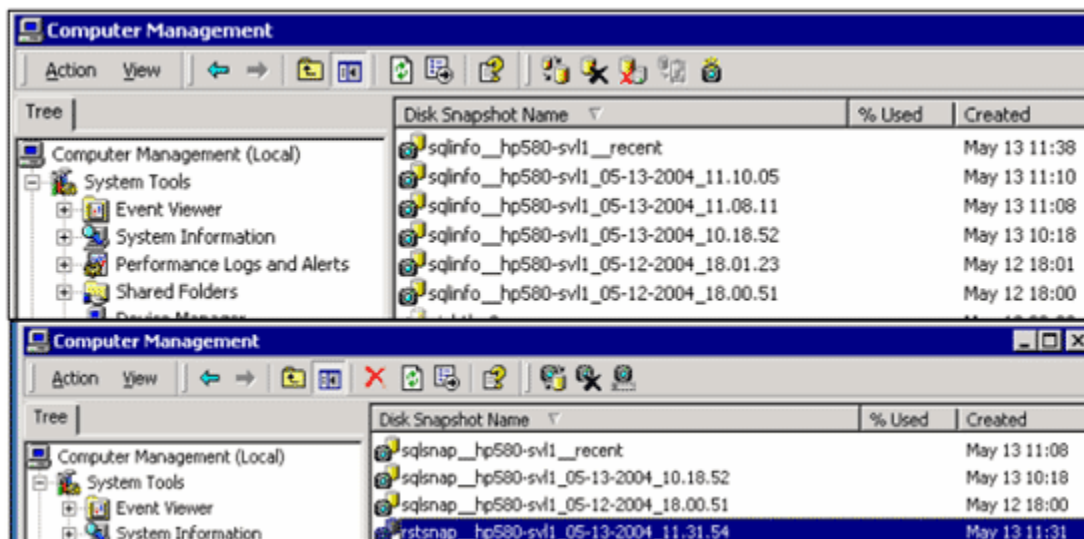


Figure 12 Snapshot name list (SnapDrive).

8.2. Snapshot Copies of LUNs for Write Access

Snapshot backups of SQL Server databases are read-only point-in-time images. This protects the data and guarantees the integrity of the Snapshot backups. SMSQL will, in certain situations, restore a LUN in a Snapshot copy for temporary read/write access (restore to alternative location using `_Writeable_snapshot`). Writable Snapshot is used during a verification process and when a DBA restores a database to an alternative location for recovering from operational mistakes. A restored writable Snapshot copy is for short-term use only.

Note: It is not recommended to create a Snapshot copy of a volume that contains an active Snapshot copy.

8.3. Disk Space Consumed by Read/Write Snapshot Copies

Even though read/write Snapshot files are initially linked to the existing blocks in the Snapshot copy, it is necessary for the filer to allocate blocks equal to the entire size of the file should it be completely overwritten and a copy of it created. There must be enough available disk space to accommodate the entire size of the original LUN while the read/write Snapshot copy is mounted. The space consumed by the read/write Snapshot copy is marked as free disk space when it is disconnected using the Disconnect Disk command within SnapDrive.

Note: Though technically possible, it is not recommended to create duplicates of read/write Snapshot copies (.rws files) unless absolutely necessary.

8.4. Space Reservation

Space reservation is designed to ensure that protected files (or files that have space reservation turned on) always have the free space they expect so that a Snapshot copy of the LUN can complete even if 100% of its blocks have changed.

SQL Server preallocates file space for its preallocated database page. This is the database engine's way to make certain there is enough space and it is able to write all database pages. When SQL Server updates database pages, it expects writes to complete (preallocated), so if WAFL cannot find enough free blocks to write all changed blocks, then SQL Server will fail.

Data ONTAP 6.4 or newer provides for space reservation at the qtree and file levels. Any file may be designated as protected, but it is turned on by default for LUNs.

When creating a LUN, the NetApp storage device verifies there is enough available disk space to accommodate the specified size. By default, WAFL reserves blocks equal to two times the specified size of the LUN (when Snapshot is used) so that all future writes can be completed. If space reservation was not enabled, it would be possible to oversubscribe the available storage. If this occurs, SQL Server will, because its space is preallocated, interpret it to mean the storage device is bad, and it will mark affected databases as suspect.

Note: SnapDrive requires that all LUNs have space reservation enabled. Disabling space reservation for a LUN is not supported with SnapDrive. See *Network Appliance SnapDrive v.3.x Installation and Administration Guide* for more detailed information about space reservation.

9. Backing Up and Restoring SQL Server Databases with SMSQL

How to back up and restore SQL Server databases is thoroughly discussed in *SnapManager for SQL Server 2000 v.1.5 Installation and Administration Guide*. Only areas that have to be well understood will be discussed in this report.

SMSQL will back up:

Snapshot copies created by SnapDrive are volume-based Snapshot copies. Therefore, all databases placed in any LUN using the volume will be backed up at the same time (see Section 7).

Snapshot copies will be created of all volumes used by the database(s) that is being backed up.

System databases will be backed up using streaming-based backup.

User databases placed in a LUN that also contains system databases will be backed up using streaming-based backup.

Backup of transaction logs will always use streaming-based backup.

SMSQL will restore database(s) by:

Requesting SnapDrive to use Single_File_SnapRestore to restore all LUNs used by the database(s)

Restoring selected databases out of many sharing one or two LUNs

Requesting SQL Server to restore the databases

Requesting SQL Server to apply selected backed-up transaction logs

The most efficient configuration is one user database per LUN.

9.1. Backup Recommendations

SMSQL's full backup using Snapshot is quick and doesn't influence the performance compared to traditional backup with Enterprise Manager. Transaction log backup is functionally the same as the transaction log backup done by Enterprise Manager. How much time it takes to back up a transaction log depends mainly on the active section of the transaction log that has to be extracted and copied to the dump directory (SnapInfo). When planning the backup and restore process, the methodology or considerations are different from considerations when backup is done by Enterprise Manager. Backup of the transaction log usually takes longer than a full backup when SMSQL is used, which is the opposite when Enterprise Manager is used for backup. Therefore, the main consideration when planning backup and restore processes is, *what is the longest restore window allowed by the application?*

Transactions such as insert, delete, or update will add records to a database's transaction log. Therefore, a high transaction rate adds more records to a transaction log compared to a lower transaction rate.

All backup and recovery processes are created according to each environment's unique requirements, but a *general recommendation is to back up the database and transaction log every two hours during production time and less often during off-peak hours*. It is also recommended to offload the verification process to another dedicated server that will not influence online users' performance or to schedule verification processes to off-peak hours. If restore requirements are strict, then it is recommended to analyze the transaction load and to do some testing. An example:

A customer's restore requirement is to be able to restore a corrupted database in 30 minutes or less. Testing has shown that backup every 30 minutes during peak time makes it possible to restore in 20 minutes (without the verification process). Backup every 30 minutes during 24 hours will create 48 Snapshot copies of the volume with the database and 48 Snapshot copies of the volume containing SnapInfo. Since another requirement is to have backups for a whole week online, then to back up every 30 minutes is not an acceptable solution. The testing was done during peak database load, and the customer knows that the transaction rate varies over a 24-hour period. After analyzing the transaction load over a couple of weeks, the customer was able to create a table showing the transaction rate during 24 hours and to make the rate relative to the peak load.

In Figure 13, the customer peak transaction rate is from 11 a.m. until 3 p.m.; during that period backup has to be created every 30 minutes. The transaction rate drops to 50% between 3 p.m. and 8 p.m.; therefore, backup every hour will satisfy the customer's restore requirement. Transaction rate is very low between 10 p.m. and 8 a.m. during. The result is 19 backups during 24 hours. The database will be backed up every three hours during periods with low transaction rates, from 10 p.m. to 8 a.m. Every full backup will create a Snapshot copy of the volume containing the database and the volume containing SnapInfo (see TR3363, Section 4: [SQL Server Sizing and Capacity Planning Guidelines](#)); in this case 19 Snapshot copies will be created for each volume (38 Snapshot copies if just one volume is used). Every backup of a transaction log will also, by default, create a Snapshot copy of the volume that contains SnapInfo, but creation of the Snapshot copy can be turned off! *Note: If SnapMirror is used, then the default behavior must not be changed!*

Times	Trans Rate	Backup Rate (Min)	Snapshot Copies
-------	------------	-------------------	-----------------

1:00–6:00 a.m.	1%	180	1
6:00–8:00 a.m.	10%	180	1
8:00–10:00 a.m.	50%	60	1
10:00–11:00 a.m.	75%	45	1
11:00–3:00 p.m.	100%	30	8
3:00–8:00 p.m.	50%	60	5
8:00–10:00 p.m.	30%	90	1
10:00–1:00 a.m.	2%	180	1

Figure 13 Transaction rate during 24 hours.

9.2. Archiving Backups

It is strongly recommended to archive backup sets as soon as possible. Disasters happen, and if a disaster happens with the storage device containing the database(s) and backed-up images, then it is not possible to restore the database(s). Archive media can be tape, another filer, or a NearStore device. The archive process is described in *SnapManager for SQL Server 2000 v.1.5 (SMSQL) Installation and Administration Guide* and will not be repeated in this document.

10. Using SMSQL with Network Appliance SnapMirror

Network Appliance SnapMirror technology mirrors data to one or more Network Appliance filers over a local area network (LAN) or wide area network (WAN). Once source and destination relationships are established, a SnapMirror baseline transfer initializes the mirror to create a replica of the source on the destination. SnapMirror maintains the synchronization of the replica through efficient, incremental updates. Thus SnapMirror is highly effective in its use of valuable bandwidth, as it does not repeatedly send the entire LUN.

The frequency of SnapMirror update events is determined by the frequency of SMSQL backups. SnapDrive triggers SnapMirror updates upon completion of a SnapManager backup procedure. The SnapMirror maximum transfer rate can also be adjusted in kilobytes per second.

Key SnapMirror Concepts and Tips:

SnapMirror relationships are based on sources and destinations.

A destination updates the mirrored copy or replica of its source by "pulling" data across a LAN or WAN when an update event is triggered by the source.

Consistent with the pulling nature of SnapMirror, relationships are defined by specifying sources from which the destination filers synchronize their replicas.

Destination filers are identified on source filers by assigning destination filers privileges via the "snapmirror.access" protocol access option or by inclusion in the snapmirror.allow file.

SnapDrive 3.x works with volume SnapMirror (VSM) only. Qtree SnapMirror (QSM) is not supported.

As discussed earlier, SnapManager is integrated with SnapDrive, which interfaces directly with Network Appliance filers using the iSCSI or FCP disk-access protocols. SnapManager relies upon

Network Appliance Inc.

SnapDrive for disk management, controlling Snapshot, and SnapMirror events. Figure 14 illustrates the integration among SMSQL, SQL Server 2000, SnapDrive, and NetApp storage devices.

SnapManager for SQL Server 2000 and SnapMirror		
SnapManager for SQL Server 2000		
SnapDrive 3.x		SQL Server 2000
Online Storage	SnapMirror	

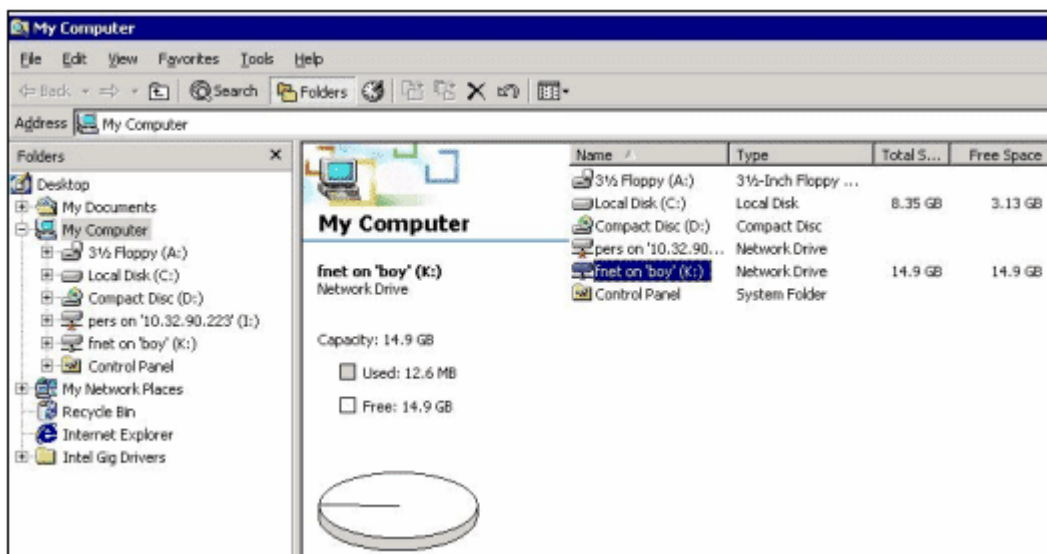


Figure 14 SMSQL and SnapMirror.

10.1. Checklist for Configuring SnapManager and SnapMirror

Install (via CIFS setup) both filers into the same Windows domain as SQL Server.

Configure the SQL Server, SnapDrive, and SnapManager Win32 services to use the same logon service account.

Make sure the SnapDrive service account has the "Log on as a service" right in the Windows 2000 operating system (normally occurs during installation).

Verify that RSH commands work between the SQL Server(s) and both storage devices using the specified service account.

License and enable FCP and/or iSCSI on both storage devices.

License and enable SnapMirror on both storage devices.

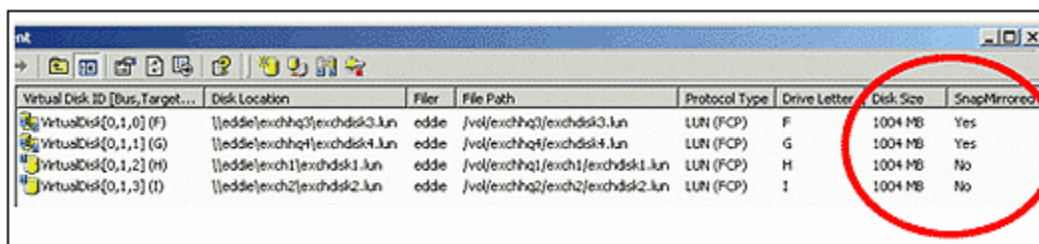
Establish SnapMirror relationships.

Make sure the storage device's SnapMirror schedule is turned OFF by assigning the "- - -" value (four dashes separated by spaces) in the "custom schedule" field.

Initialize the mirror configurations.

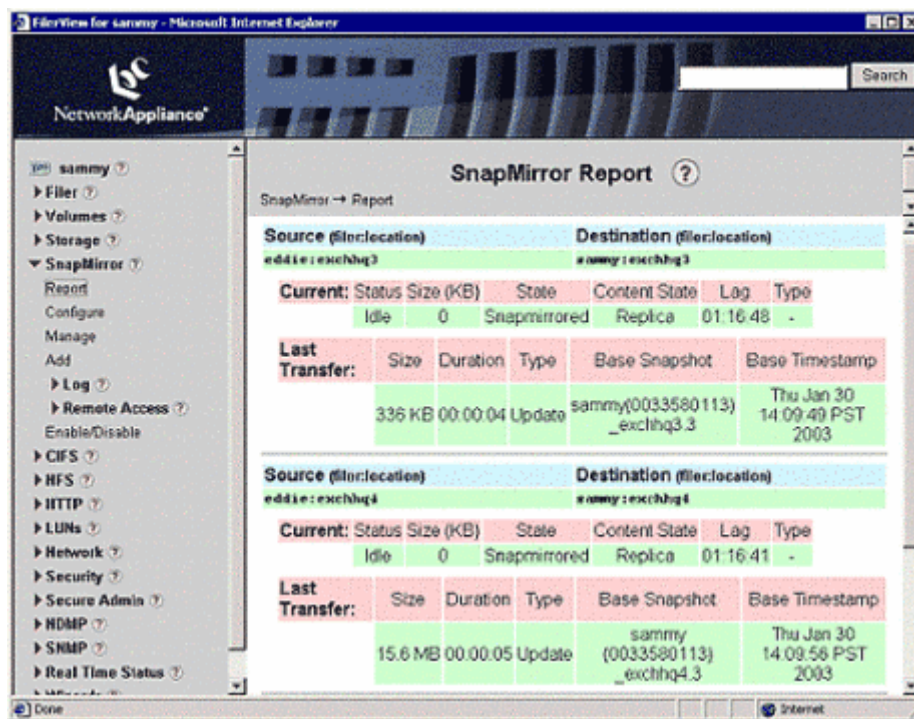
SnapMirror updates to the specified destinations will occur after the completion of every SMSQL backup.

SnapDrive and FilerView can be used to verify the successful completion and state of the configured mirrors.



Virtual Disk ID (Bus, Target...)	Disk Location	File	File Path	Protocol Type	Drive Letter	Disk Size	SnapMirrored
VirtualDisk{0,1,0} (F)	\\eddie\exchh3\exchdsk3.lun	eddie	/vol/exchh3/exchdsk3.lun	LUN (FCP)	F	1004 MB	Yes
VirtualDisk{0,1,1} (G)	\\eddie\exchh4\exchdsk4.lun	eddie	/vol/exchh4/exchdsk4.lun	LUN (FCP)	G	1004 MB	Yes
VirtualDisk{0,1,2} (H)	\\eddie\exch1\exchdsk1.lun	eddie	/vol/exchh1/exch1/exchdsk1.lun	LUN (FCP)	H	1004 MB	No
VirtualDisk{0,1,3} (I)	\\eddie\exch2\exchdsk2.lun	eddie	/vol/exchh2/exch2/exchdsk2.lun	LUN (FCP)	I	1004 MB	No

Figure 15 SnapMirror status in SnapDrive.



SnapMirror Report						
Source (filer/location)		Destination (filer/location)				
eddie : exchh3		sammy : exchh3				
Current:	Status	Size (KB)	State	Content State	Lag	Type
	Idle	0	Snapmirrored	Replica	01:16:48	-
Last Transfer:	Size	Duration	Type	Base Snapshot	Base Timestamp	
	336 KB	00:00:04	Update	sammy(0033580113)_exchh3.3	Thu Jan 30 14:09:49 PST 2003	
Source (filer/location)		Destination (filer/location)				
eddie : exchh4		sammy : exchh4				
Current:	Status	Size (KB)	State	Content State	Lag	Type
	Idle	0	Snapmirrored	Replica	01:16:41	-
Last Transfer:	Size	Duration	Type	Base Snapshot	Base Timestamp	
	15.6 MB	00:00:05	Update	sammy(0033580113)_exchh4.3	Thu Jan 30 14:09:56 PST 2003	

Figure 16 SnapMirror status in FilerView.

11. Management Tools

11.1. Snapshot Reserve Usage Monitoring

The task of monitoring the Snapshot reserve is automatically configured at the time of LUN creation. Simply monitor the application event log for warning events generated in the SnapDrive source and Snapshot Monitor event categories. Figure 23 demonstrates that due to Snapshot consumption, the reserve must be expanded to 63%, and there is not enough disk space to do so. In order to rectify this situation, either add more disks to the volume or remove some of the oldest SnapManager for Microsoft SQL Server 2000 Snapshot copies.

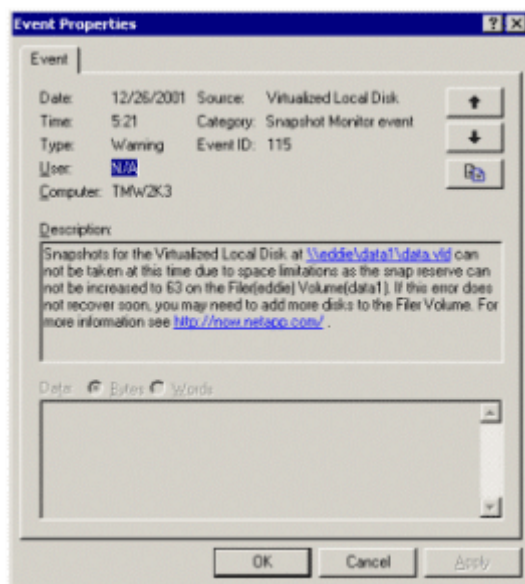


Figure 17 LUN error due to space limitations.

11.2. Disk Space Usage Monitoring

The amount of disk space used by each database managed by SQL Server 2000 should be monitored to ensure that the logical drives (LUNs) do not run out of space. SQL Server 2000 will autoextend its preallocated file space when there is not enough free file space to extend a table (if the table has been defined with autoextend on). SQL Server will stop processing transactions when it cannot extend its file space.

Monitoring free space has to be done in two levels. The first level is to free preallocated space; this space is available for table extents. If Perfmon or a similar monitoring tool is already used for monitoring the system performance, then it is easy to add counters to monitor database space (see Figure 9 and Figure 10). If the database size is only increasing slowly, then it is maybe enough for the DBA to check a database's available space once a quarter. A database's properties will show a database's current size and how much space is free. Figure 24 is an example with master properties; it shows a database size of 12.19MB with 1.27MB available for new table extends. When free space gets below or close to 64kB, then the preallocated file space will be extended. The second level to monitor is free LUN space, which can be done with Windows Explore; Figure 25.

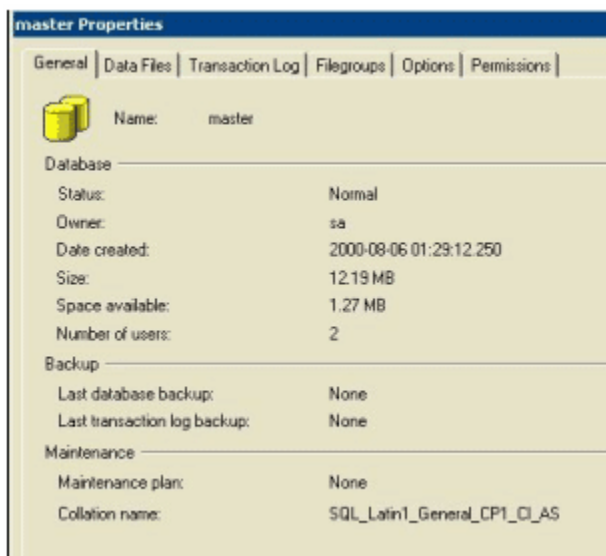


Figure 18 Database's available space.

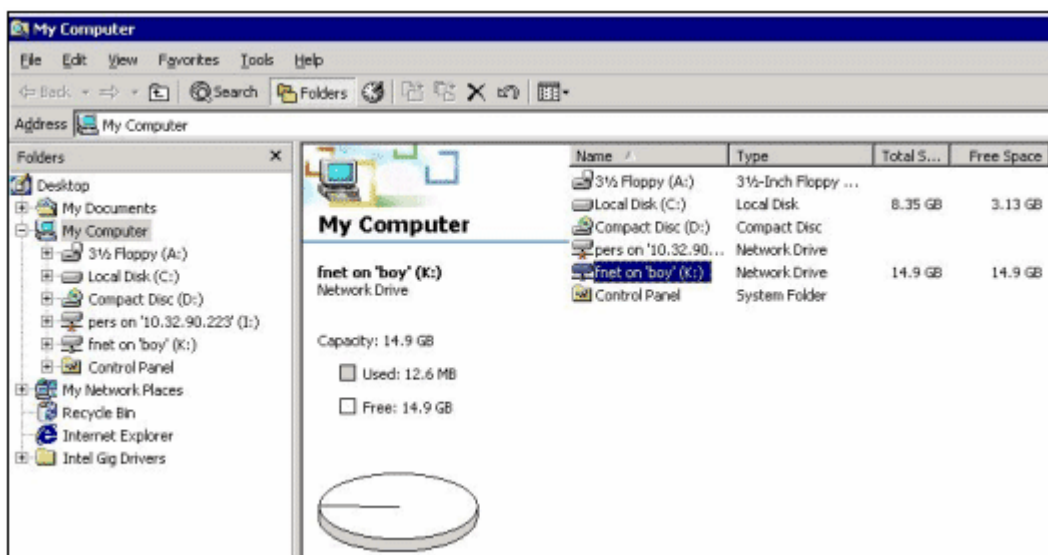


Figure 19 Available LUN space.

11.3. Filer Event Monitoring

Monitor the filer event logs to ensure proper operation of the filer. Issues may be discovered in the event logs that require administrative action. One such action may be to replace a disk in a volume if spare disks are not available.

This task can be completed by using the FilerView utility built into Data ONTAP. This utility can be started by pointing a Web browser to http://filername/na_admin. Next, click the FilerView link. FilerView will start and will ask for the credentials of a filer administrator. Once logged on to FilerView, click the Filer menu option on the left side of the screen. Then choose the Syslog Messages menu option. Review the log on the right side of the screen for any issues that may require administrative action. For more information on FilerView, refer to the Data ONTAP System Administrator's Guide.

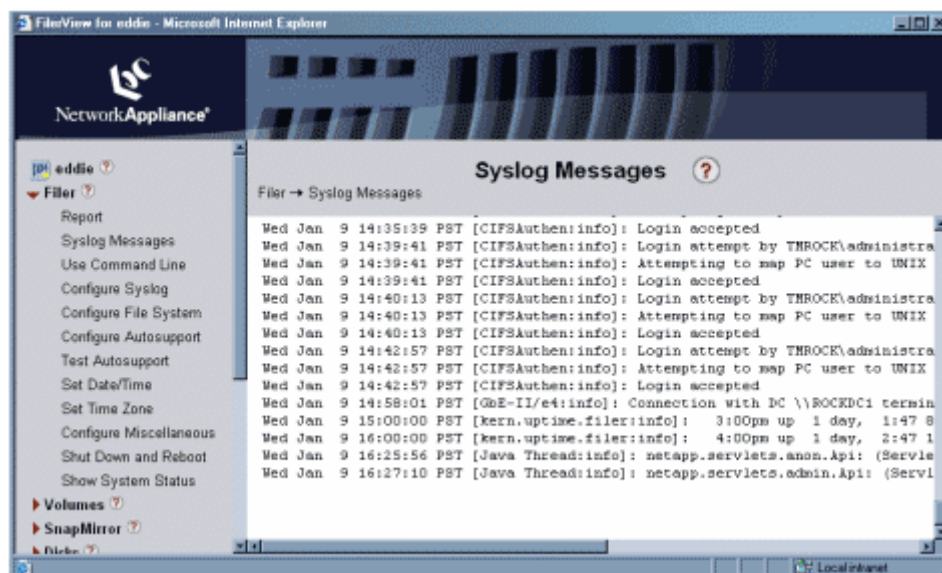


Figure 20 Syslog messages in FilerView.

11.4. Performance Monitoring and Tuning—Set of Basic Counters

Performance monitoring is something every administrator has to do and should do; performance tuning is something everybody has to do periodically and is a central requirement for implementing capacity planning guidelines correctly. Capacity planning, including recommended counters to collect, is discussed in TR3363, Section 5: [SQL Server Sizing and Capacity Planning Guidelines](#).

11.5. Terminal Server or Remote Desktop

Microsoft Terminal Server is not recommended as a way of administrating SnapDrive or SnapManager for SQL Server. Creating LUNs from a terminal server session, the drives can appear as if they were not created or have been disconnected when in fact they are operating without errors. The only way to fully resolve problems when using Terminal Server with SnapDrive is to log out of the session (do not disconnect). NetApp recommends avoiding Terminal Server for server management when possible. SMSQL will, at times, temporarily connect to LUNs in a Snapshot copy when backed-up database(s) have to be verified; this operation will fail if the action was activated through Terminal Server!

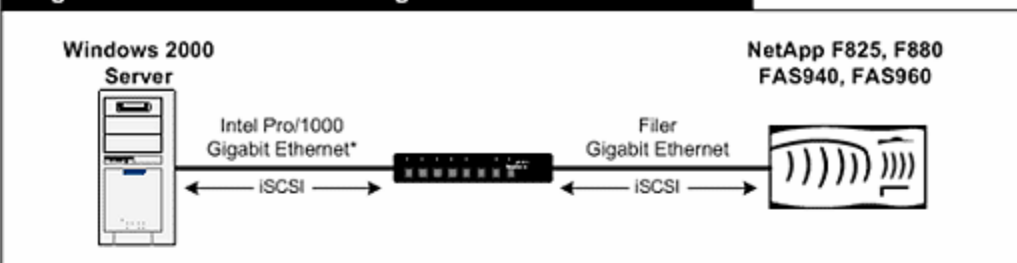
12. Summary

Network Appliance SnapManager for Microsoft SQL Server 2000 is a complete data management solution that provides backup and restore features using Snapshot technology. By reducing backup and restore times, minimizing application outages, and consolidating database storage, SMSQL delivers a cost-effective solution for managing critical SQL Server databases.

In conclusion, the recommendations made in this paper are intended to be an overview of best practices for *most environments*. This paper should be used as a set of guidelines when designing, deploying, or administering SnapManager for SQL Server 2000. To ensure a supported and stable environment, familiarize yourself with the resources provided in this paper and involve an Exchange specialist if necessary.

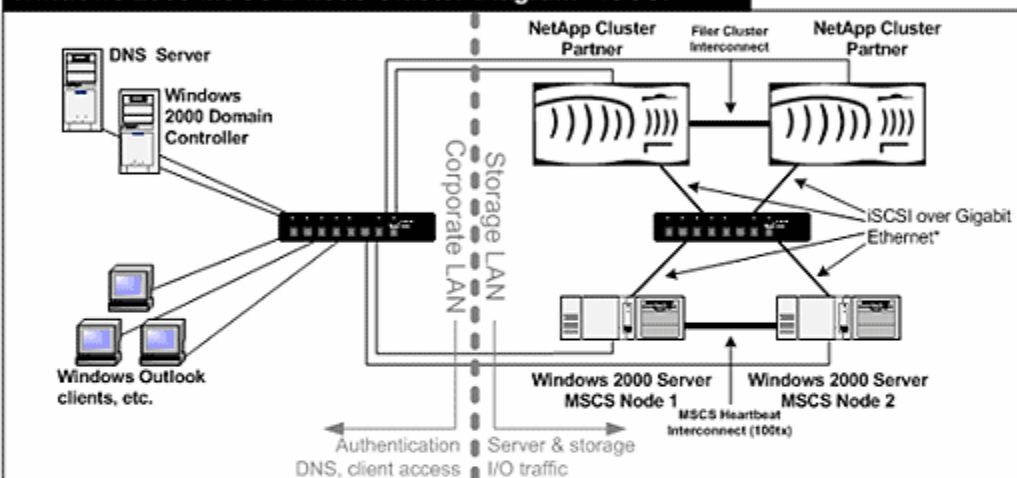
13. Ethernet and FCP Technology Diagrams

Single-node Windows 2000 diagram - iSCSI



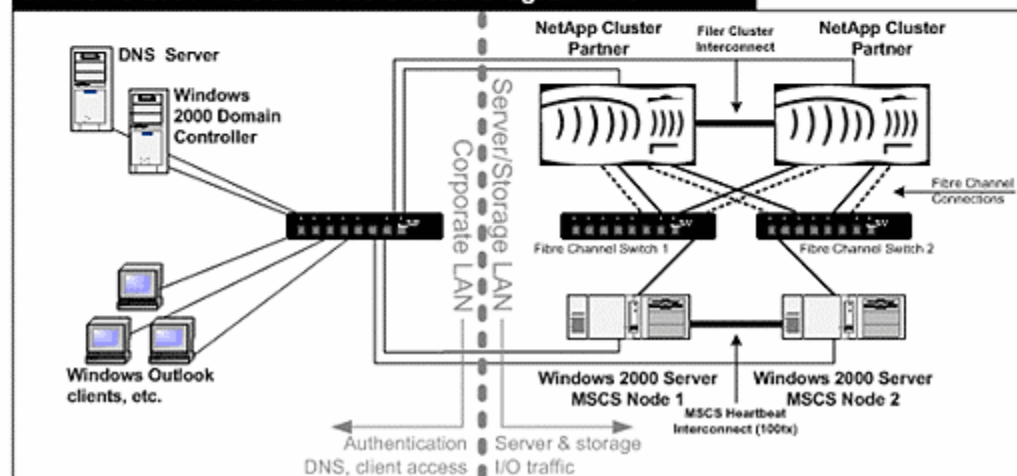
*Note: The Intel Pro/1000 Gigabit Ethernet card is required and must be installed in the Windows 2000 server for use with iSCSI

Windows 2000 MSCS 2-node Cluster Diagram - iSCSI



*Note: Intel Pro/1000 Gigabit Ethernet Cards are required and must be installed in both Windows 2000 servers for use with iSCSI

Windows 2000 MSCS 2-node Cluster Diagram - FCP



Appendix A. Database Logical and Physical Layouts

Designing and implementing SQL Server databases is in most aspects intuitive, but it can also be critical for future growth and performance and important for optimal use of SMSQL.

Network Appliance Inc.

Understanding the relationships among database files, LUNs, and NetApp filer volumes is important for optimal use of SMSQL. This appendix will discuss database file objects and explore various layouts of SQL Server databases and ways to utilize LUNs that can be supported by SMSQL.

Both system and user databases deploy the same basic structure; therefore, the given description will cover system databases¹¹, demonstration databases¹², and user-defined databases. The focus is not the logical structure of database files but a database's physical files and how they are accessed.

Figures A1 to A5 illustrate different layouts of SQL Server databases, followed by a description, including advantages as well as disadvantages.

A.1. Common Database Layouts

The most common layout of a database is with two files: one primary data file and one transaction log file, as illustrated in Figure A1. Both the data file and the log file use the same LUN: disk "M."

Advantages of the layout in Figure A1 are:

Only one drive letter is used for the database¹³.

It is easy to administrate and to understand the relationship between files and the database.

The LUN can be shared by several databases when all files belonging to the databases are located in the same LUN¹⁴.

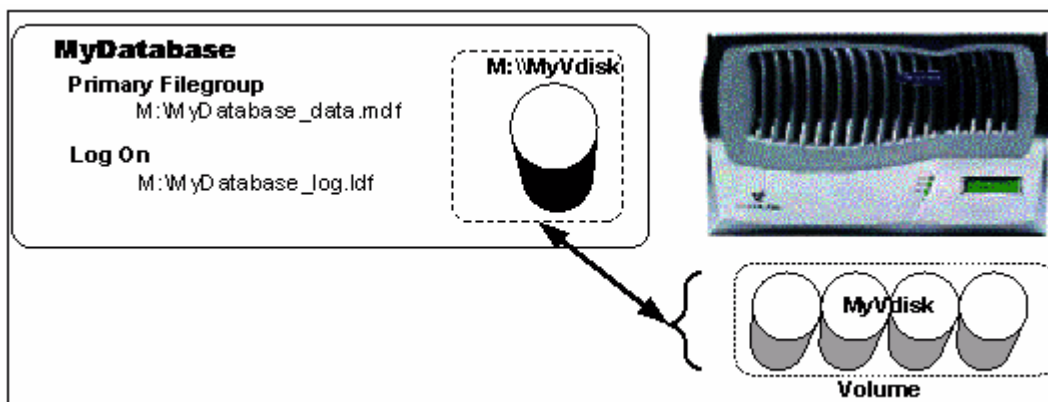
Making SnapMirror copies is simple, since only one filer volume is used by the database.

Disadvantages are:

If the volume fails, the database has to be recovered from an archive of a full backup and backup transaction logs. A volume almost never fails, but it can happen. Archiving backed-up databases is important for disaster recovery.

Synchronous access to the transaction log and random access to the data file are mixed in the same volume, but that is only a potential issue with environments that require high levels of I/O access. Database files can always be migrated with SMSQL.

It is not practical with very large databases (700GB and up).



Network Appliance Inc.

Figure A1) Basic structure of a SQL Server database.

Another common layout of the database's files is illustrated in Figure A2. It is very similar to the illustration in Figure A1, except that the transaction log file and the data file are each located in separate LUNs utilizing a single volume. The transaction log is located in the X device, and the data file is located in the M device.

SnapManager handling of two drive letters has changed with SMSQL 1.5. SMSQL 1.5 allows multiple databases to share two devices, which is prevented by SMSQL 1.0 (see Figure A2).

Advantages of the layout in Figure A2 are:

- SnapMirror transactions are simple when only one filer volume is used by the database.

- I/O is distributed over two separate virtual devices, so the network traffic can be divided between two distinct transport layers and volumes.

- The potential I/O throughput is greater¹⁵.

Disadvantages are:

- It uses two drive letters.

- If the single volume fails, the database has to be recovered from archive and backed-up transaction logs.

- Not practical with very large databases (700GB and up).

- The LUNs cannot be shared by any other database.

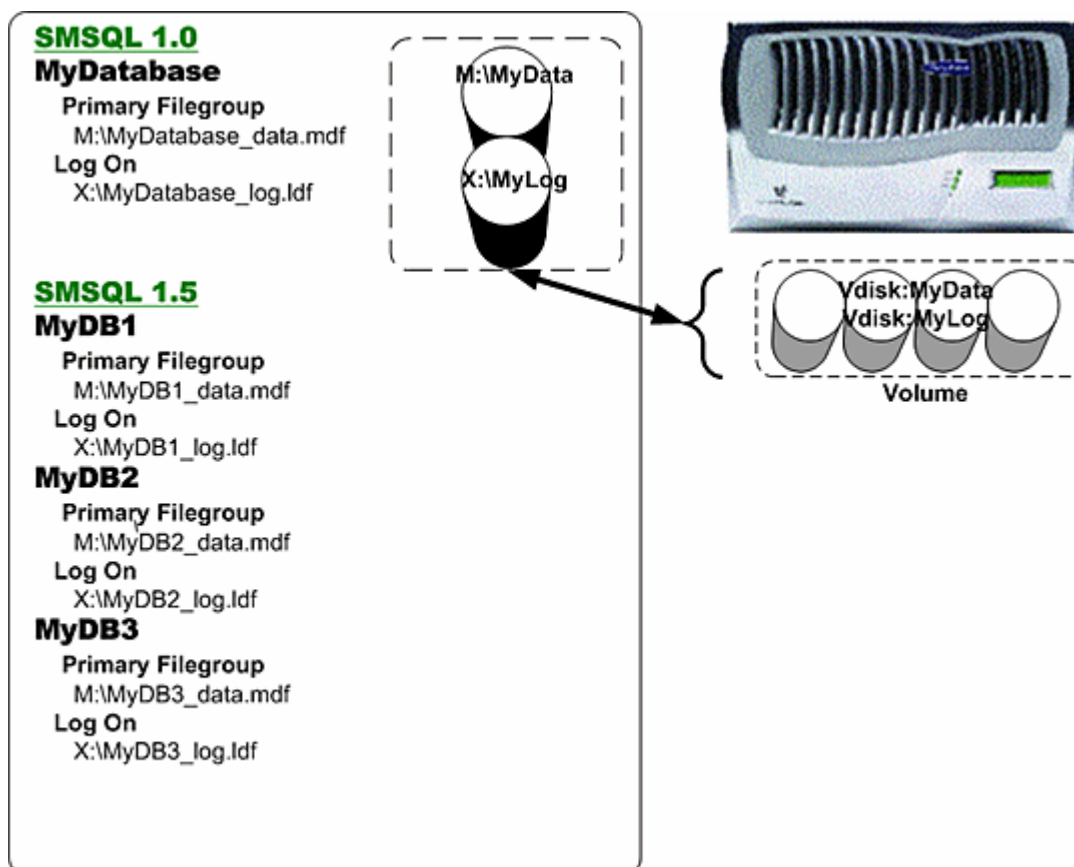


Figure A2) Database layout with two LUNs.

A.2. Large Databases with Sequential I/O Access

Large databases with sequential access can potentially increase the I/O performance when database objects (such as tables or indexes) are created in file groups consisting of several data files. When SQL Server realizes that a table or index has been created in a file group with several files, then SQL Server will extend the table space round robin over available files. In practice, this means that SQL Server will allocate 64kBrround robin over all files in the file group¹⁷.

The database in Figure A3 has three data files in *MyFilegroup* and one transaction log file. The files in file group *MyFilegroup* are all located in the M device and stored in one volume. The transaction log file¹⁸ is stored in device X, which is located in a second volume.

Advantages of the layout in Figure A3 are:

You may achieve higher concurrent I/O access when a table is sequentially scanned¹⁹.

I/O is distributed over two separate volumes, so the sequential access to the transaction log is to a different volume than the access to the data files. Access to the transaction log is always sequential, but access to data files is potentially random. Access to OLTP databases is generally random, and access to data warehouses is mostly sequential.

If the database becomes too large for one volume, then it is fairly easy to move one of the data files to another volume or to expand the file group with a new file located in another volume.

Disadvantages are:

It uses two drive letters.

SnapMirror processes are more complicated.

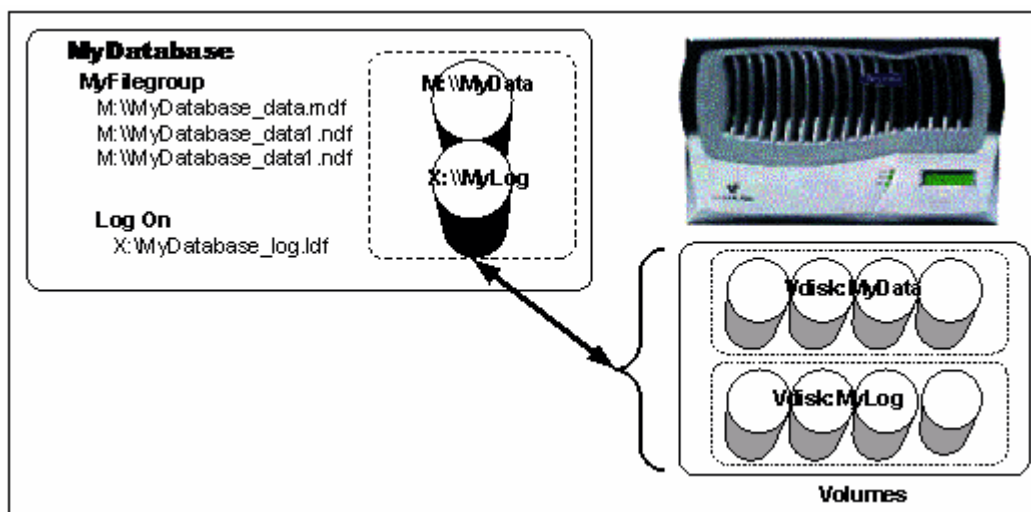


Figure A3) Database layout using two volumes.

A.3. Very Large Database with High I/O Performance

Large databases with high I/O performance requirements can create large tables in different file groups with multiple data files. Multiple data files can potentially increase the physical I/O rate²⁰ by making it possible for SQL Server to issue concurrent I/O requests to each file in a file group²¹. Data files in a file group can be stored in one single device, Filegroup 1, or in several devices, Filegroup 2 (Figure A4). Reasons for using several file groups are:

The database has several large tables²².

A table and its indexes can be created in different file groups.

The database contains very large tables that are too large for a single volume. The data files in the file group can easily be spread over multiple volumes, which can potentially also increase the I/O performance, but that depends on the application's access pattern.

Advantages of the layout in Figure A4 are:

You will be able to achieve more concurrent I/O access when tables are sequentially scanned.

A table and its indexes can be created in different file groups.

I/O is distributed over two separate volumes; therefore, access to the transaction log is to a different volume than the volume that contains the data files. Access to the transaction

log is always sequential, but access to data files is potentially random. Access to OLTP databases is generally random, and access to data warehouses is mostly sequential.

If the database becomes too large for one volume, it is fairly easy to move some of the data files to another volume or to expand the file group with new files located in another volume.

Disadvantages are:

Creating several file groups uses many drive letters.

The LUNs cannot be shared by any other database.

Backing up the files with SnapMirror is complicated.

The layout is complicated and rarely needed; a NetApp filer is great at load balancing I/O over available disks in a volume.

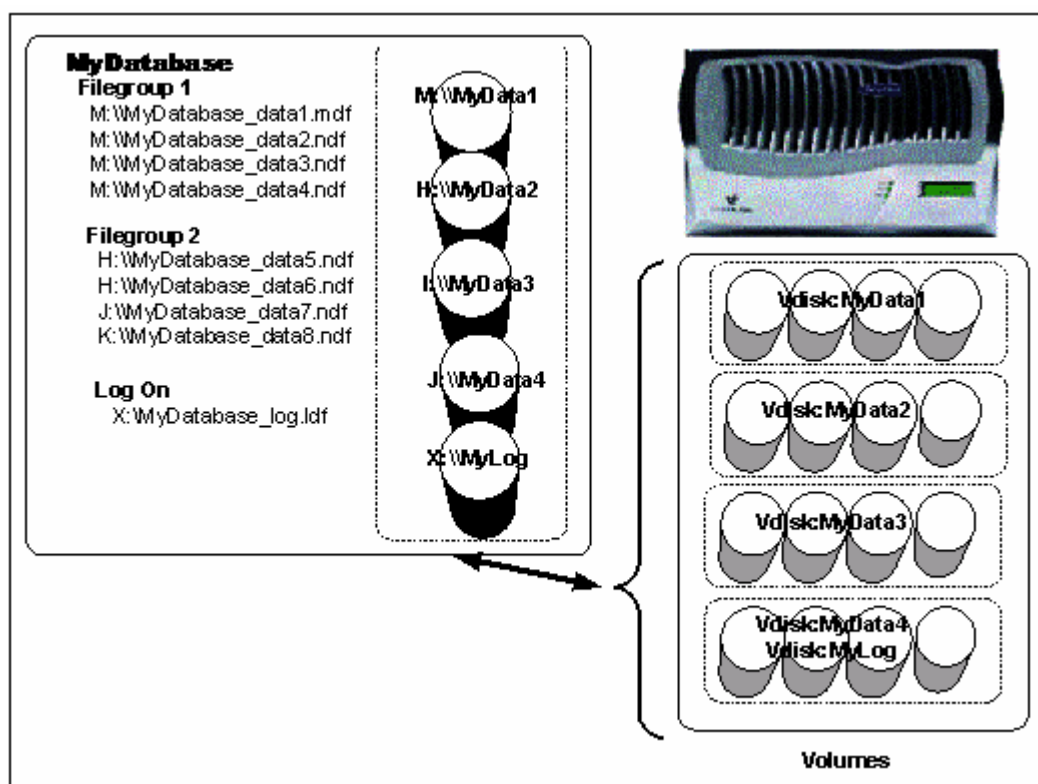


Figure A4) Database layout with multiple filer groups.

A.4. Several Databases on One Windows Server

We have until now illustrated possible layouts of a single database. In this section we will discuss how to deploy several databases controlled by one or more SQL Server instances running on a single Windows platform. Each Windows²³ platform has a fixed number of available drive letters—around 20 that all databases filers have to be allocated across. If there are more databases than available drive letters, then the databases have to be grouped together according to some criteria and stored in one LUN. Deciding on the physical layout for multiple databases spread over

several LUNs can make the design complicated.

Figure A5 illustrates two basic concepts when several databases are controlled from a single server.

The first concept is illustrated by databases MyDatabase and MyDB; all files belonging to those two databases are located in one single device, M. One or two devices can be shared by many databases as long as all files belonging to the databases are located in the one or two LUNs (see Figure A2). The second concept is illustrated by database MyLargeDB. When database files are spread over several (more than two) LUNs, then those LUNs cannot be used by any other database. This is because of the use of Snapshot and SFSR²⁴; if MyDatabase was located in the J drive, then SMSQL could not restore MyDatabase without MyLargeDB becoming corrupted (Figure A5).

When several databases share a single LUN, then SMSQL can quickly back up all databases in the volume²⁵ and can quickly restore all databases in the LUN; Section 8 describes how to restore a single database out of many databases in a LUN. It is often quicker to restore all databases in a LUN than to restore the corrupted database, but that depends greatly on the size of the databases. Restoring one of the databases on a LUN takes about the same time as using a SQL Server-level restore, but a full backup with SQL Server takes much longer than when SMSQL is used for a full backup.

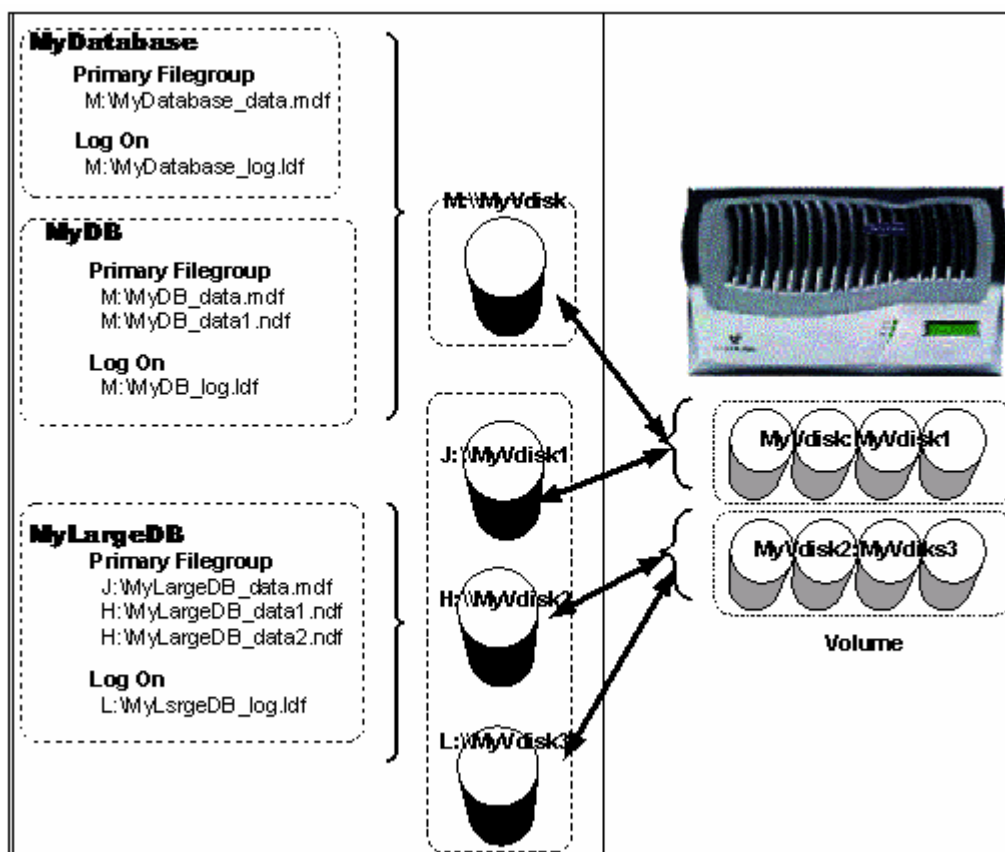


Figure A5) Multiple databases.

When a server has to handle more databases than there are available drive letters²⁶, you should sort all databases according to size, how critical each database is, and acceptable downtime.

Size. The larger a database, the greater the advantage of using SMSQL for backup and restore. Grouping the largest database with the smallest databases makes it possible to restore all databases if the large database has to be restored or to restore a single smaller database if that has to be restored.

Acceptable downtime. Knowing the acceptable downtime makes it possible for the DBA to decide the order in which databases have to be restored.

Criticality. Databases that are critical can be located in their own LUNs. This might require that more of the remaining databases be located together in the other LUNs, but this is a tradeoff that has to be made.

Appendix B. General Consolidation Principles

It is quite common for companies to add new Windows servers whenever new SQL Server applications are required for a business purpose. It is fairly easy to add new hardware to network infrastructure, to install the application, and to make the new environment available to end users. This works well for some companies but can create administrative and maintenance issues for others. Many companies begin to think about consolidation when they discover that it is hard to maintain environments with multiple servers and locally attached storage, which can become costly and complicated. This appendix will describe important considerations when consolidating SQL Server environments.

Figure B1 shows a common infrastructure with a number of servers, with each server supporting one SQL Server instance and application. Each server and its resources are independent²⁷ from each other and have to be administrated separately. The administration of each server includes:

- Performance monitoring and tuning

- Capacity planning

- Maintenance

- System-level backup

- Database backup

Not only is it complicated and time-consuming to administer many servers, but also available resources on one server cannot be utilized by another server, such as:

- Idle processors

- Memory

- Free files and disk space

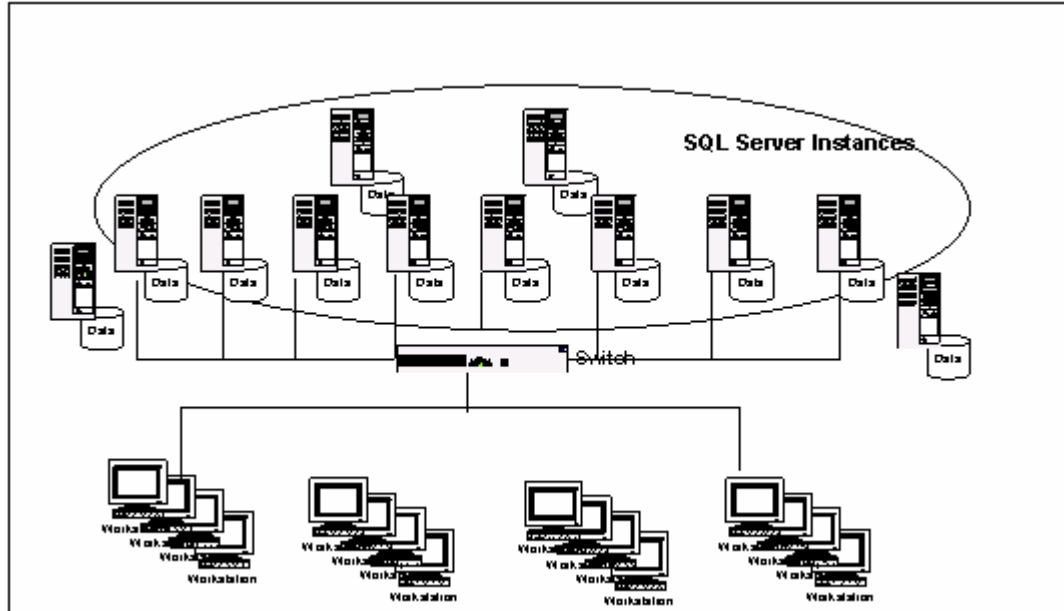


Figure B1) Before consolidating.

Consolidating the environment, as illustrated in Figure B1, makes it easier to administrate and to share resources among applications. Consolidation projects involve different but related projects, storage consolidation (Figure B2), and server consolidation (Figure B3).

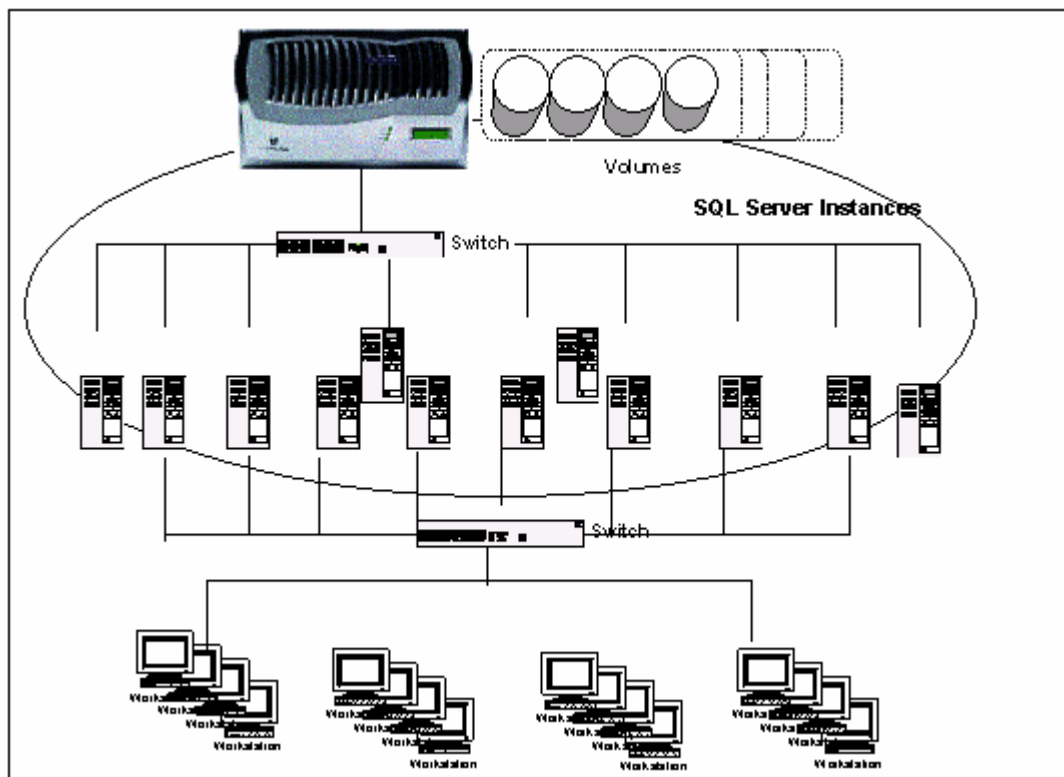


Figure B2) Storage consolidation.

Figure B2 illustrates a situation where the disk storage has been consolidated, making it much easier to share storage resources and do capacity planning among SQL Server instances. Consolidating server storage works well with limited numbers of servers, such as when there is a one-to-one relationship between a server and a NetApp volume. Even though it is possible for servers to share a NetApp volume, it is absolutely not recommended with SMSQL. Backups are volume-level Snapshot copies, which makes it complicated when a single volume is shared among servers; *one volume per SQL Server platform is recommended with SMSQL.*

It is advisable when the environment consists of many servers to consolidate them so that it is possible to create a simple relationship between one server and one volume.

Figure B3 illustrates the final consolidation, which consists of two SQL Server platforms and a cluster of filers.

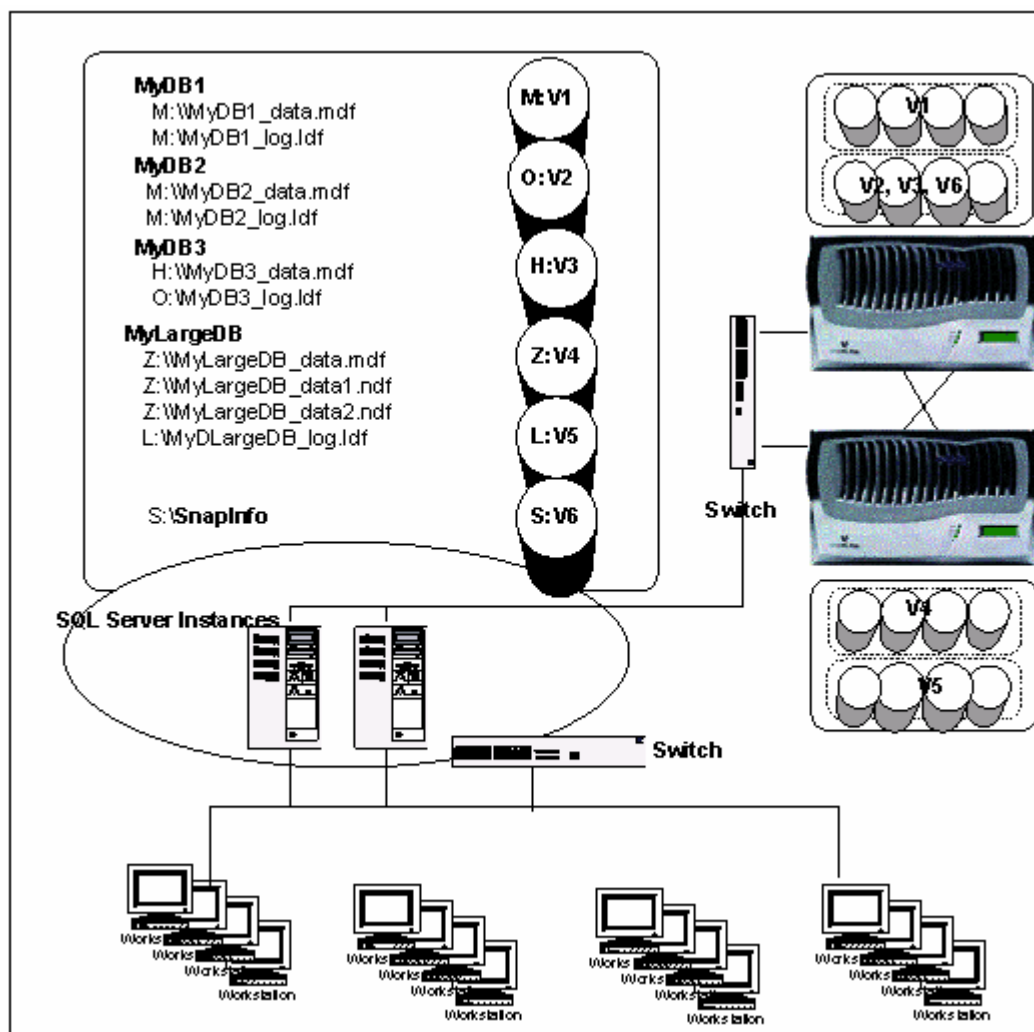


Figure B3) Complete consolidation.

Figure B3 exemplifies the total consolidation of the environment in Figure B1. All locally attached disk storage has been consolidated into a cluster of filers, and the many smaller servers have been consolidated into two larger servers.

Figure B3 also illustrates important points discussed in Appendix A:

A LUN can be shared when all files belonging to the databases are located in one LUN (MyDB1 and MyDB2).

When a database's files are located in different LUNs, the LUNs cannot be shared with other databases' files (MyDB3 and MYLargeDB).

SnapInfo²⁸ cannot be located in a LUN with any transaction log file. We recommend that SnapInfo be located in its own LUN.

There is a mapping between a LUN and a drive letter; a Windows server has a maximum number of usable drive letters, usually about 20²⁹.

The advantages of the final consolidation are:

- Much easier to administer
- Much easier to monitor performance
- Much easier to share resources
- Much lower administration work

Appendix C. Checklist, Cross-Reference, Q&A

This technical report is about best practices guidelines for how to implement SQL Server environments using SnapManager for SQL Server 2000 and NetApp storage. Many recommendations have been discussed and illustrated throughout this report; more recommendations are discussed in TR 3363: [SQL Server Sizing and Capacity Planning Guidelines](#). Also, many important points are discussed in *SnapManager for SQL Server 2000 Installation and Administration Guides*, which can be found at <http://now.netapp.com> (username and password required). This section will summarize common questions and recommendations from all three sources.

[s: p]x: points to a section or page in current technical report.

IAG ([s: p] x): points to a **section** or **page** in *SnapManager for SQL Server 2000 Installation and Administration Guides* (NOW access required).

SCPG ([s: p] x): points to a **section** or page in TR 3363: [SQL Server 2000 Sizing and Capacity Planning Guidelines](#).

1. Number of SQL Server instances per server? IAG(s 5)
2. Number of databases per SQL Server instance? IAG(s 5)
 - a. If there are more databases than available drive letters: s2, sA.4, sB, p19

- b. How to group databases when storing in one or two LUNs: s2, p19
 - c. One large database: s2, sA.4
 - d. Many small databases: see sA.4
- 3. Space requirements: SCPG(s 4)
 - a. Initial size per database: SCPG(s 4.1)
 - b. Growth rate per database: SCPG(s 4.1.2)
 - c. Change rate per database: SCPG(s 4.1.2)
 - d. Physical I/O rate per database: SCPG(pp. 20-21)
 - e. Space for transaction log file: SCPG(s 4.1.5)
 - f. Performance requirements: SCPG(s 4)
 - g. Space for system databases: SCPG(s 4.1.1)
 - h. SnapInfo space requirements: SCPG(s 4.1.6)
- 4. Number of Windows servers connected to one NetApp storage device? A relationship of one volume per server is recommended: p4, sB.2
- 5. MSCS: s7, p19, SCPG(s 4.3)
 - a. Number of drive letters per MSCS node: p19
 - b. How to verify a database in an MSCS environment: p5
- 6. SnapMirror: IAG(s 10)
 - a. We recommend that you locate all database files in one single volume.
 - b. Use rolling Snapshot backups to update the mirror more frequently than when the database is backed up: IAG(p233)
 - c. Disaster recovery: IAG(s13)
- 7. Backup and restore: (s 9)
 - d. Why is stream-based full backup used? IAG(s7, p112)
 - e. Drive letters have to be available for verifying a backed-up database on the verification server.
 - f. Truncating transaction log: SCPG(s 4.1.2)
 - g. Resources for concurrent database backup: s7
 - h. Restore one out of many databases in shared LUN: s5
 - i. Bypass database verification: p5
 - j. How do I manage the number of backup sets? IAG(p122)

Appendix D. Additional Resources

<http://now.netapp.com>

[Network Appliance SnapDrive 3.x Installation and Administration Guide](#)

[Network Appliance SnapManager for SQL Server 2000 Installation and Administration Guide](#)

[Network Appliance—Integrating NetApp Filers with the Microsoft Windows Active Directory Environments](#)

[NetApp Data Protection: Double-Parity RAID for Enhanced Data Protection with RAID-DP](#)

[Network Appliance—Guide to Integrating SQL Server with SnapDrive V. 2.0.1 and NetApp Filers](#)

[Network Appliance—SQL Server Sizing and Capacity Planning Guidelines Network Appliance—High-Availability Storage Appliance](#)

[Network Appliance—Assuring Reliable Enterprise Data Availability](#)

[Network Appliance—Best Practice Guidelines for Volume and RAID Group Configuration on NearStore R200](#)

[Network Appliance—Storage Networking by Network Appliance and Cisco Systems: High Availability for Network-Attached Storage](#)

[Microsoft TechNet: Microsoft Server 2000 Resource Kit](#)

[Microsoft TechNet: Planning for Consolidation with SQL Server 2000](#)

[254321-INF: Clustered SQL Server Do's, Don'ts, and Basic Warnings](#)

[MSDN Library: Creating a Failover Cluster](#)

[MSDN Library: SQL Server 2000 Consolidation Projects](#)

[SQL Server Central: Data Migration: Step by Step](#)

[SQL Server Central: Clustering SQL Server 2000 from 500 Feet](#)

¹ OLTP is mostly random access and mostly read access.

² Data warehouse has sequential access of mostly read but heavy write during load operations, often by batch jobs.

³ Allowing multiple databases sharing two LUNs is new with SMSQL 1.5.

⁴ SnapMirror® is simplest to implement when a database utilizes only one volume, including SnapInfo.

⁵ Sequential access and random access are mixed in the same volume, but that is only a potential issue with environments that require high levels of I/O access.

⁶ A database with multiple large tables with different access patterns can be located on different volumes and thereby allocate different access patterns to different volumes.

⁷ All databases will be backed up together. ⁸ Potentially higher concurrent I/O access when a table is sequentially scanned.

⁹ Potentially higher concurrent I/O to both data and index files.

¹⁰ A system in this discussion is none clustered Windows platform or an MSCS platform with one to four nodes. The main issue is the number of LUNs (LUNs) that can be mounted on a system. A LUN is mounted on a drive letter, and a system has 23 usable drive letters.

¹¹ There are several databases installed during the installation of SQL Server. (1) The master database is the main database. (2) The model database is a template for creating new databases. (3) tempdb is for temporary storage and is recreated every time SQL Server is started; there is no need to back up. (4) msdb contains information about backup, alerts, and jobs scheduled by the SQL Server Agent. (5) The distribution database is only visible if replication is active.

¹² Northwind and pubs demonstration databases are, by default, installed with SQL Server.

¹³ LUNs are created by SnapDrive and mapped to drive letters. Windows has a fixed number of drive letters; about 20 are available for SnapDrive to create LUNs on.

¹⁴ When a database's files utilize three or more LUNs, then those LUNs cannot be shared by any other database's files.

¹⁵ The potential I/O is higher, but the actual I/O rate depends on the application's access rate and pattern.

¹⁶ Round-robin allocation is relative to file sizes.

¹⁷ A SQL Server page is 8kb, and a table extent is eight pages.

¹⁸ A database can have multiple transaction log files, but all log files are viewed by SQL Server as one continuous logical file, and only one physical file will be used at a time. When all free space in one transaction log has been used, then the next file will be used. This will continue until the transaction log has been truncated or all available space in all files has been used. At this point, SQL Server will try to enlarge a transaction log file. If this is not possible, SQL Server will stop processing transactions to the database.

¹⁹ When the database engine realizes that it is sequentially scanning a table that is located in a file group with several data files, it will use that knowledge to request concurrent reads from each file in the file group.

²⁰ Physical I/O rate depends greatly on the number of available disk drives over which the I/O can be spread.

²¹ How well an application can drive I/O depends on access pattern and locality. Random access creates longer latency and lower throughput, relative to sequential access.

²² A table is allocated in a file group.

²³ When MSCS is used, each drive letter has to be available for both nodes, which means that one stand-alone server and two MSCS nodes have the same number of available drive letters.

²⁴ Single File SnapRestore.

²⁵ Snapshot copies are controlled by SnapDrive and are volume based.

²⁶ Remember that when SMSQL verifies a backed-up database, it connects the Snapshot copy, and a drive letter has to be available.

²⁷ The network infrastructure is usually a shared resource.

²⁸ SnapInfo is described in TR3363: SQL Server Sizing and Capacity Planning Guidelines

²⁹ SMSQL restores databases by restoring LUNs.

Network Appliance, Inc.

© 2005 Network Appliance, Inc. All rights reserved. Specifications subject to change without notice. NetApp, the Network Appliance logo, DataFabric, FAServer, FilerView, NearStore, NetCache, SecureShare, SnapManager, SnapMirror, SnapRestore, and WAFL are registered trademarks and Network Appliance, ApplianceWatch, BareMetal, Camera-to-Viewer, Center-to-Edge, ContentDirector, ContentFabric, Data ONTAP, EdgeFiler, HyperSAN, InfoFabric, MultiStore, NetApp Availability Assurance, NetApp ProTech Expert, NOW, NOW NetApp on the Web, RoboCache, RoboFiler, SecureAdmin, Serving Data by Design, Smart SAN, SnapCache, SnapCopy, SnapDirector, SnapDrive, SnapFilter, SnapMigrator, Snapshot, SnapSuite, SnapVault, SohoCache, SohoFiler, The evolution of storage, Vfiler, VFM, Virtual File Manager, and Web Filer are trademarks of Network Appliance, Inc. in the U.S. and other countries. All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such.



Network Appliance, Inc.
495 East Java Drive
Sunnyvale, CA 94089
www.netapp.com