



PS Series Best Practice Deploying Microsoft® SQL Server in an iSCSI SAN

Abstract

This Technical Report describes how to deploy SQL Server in an iSCSI SAN with PS Series storage arrays. It provides configuration and management recommendations for SQL and the PS Series storage.



Copyright © 2005 EqualLogic, Inc.

October 2005

EqualLogic is a registered trademark of EqualLogic, Inc.

All trademarks and registered trademarks mentioned herein are the property of their respective owners.

Possession, use, or copying of the documentation or the software described in this publication is authorized only under the license agreement specified herein.

EqualLogic, Inc. will not be held liable for technical or editorial errors or omissions contained herein. The information in this document is subject to change. Your performance can vary.

PS Series Firmware Version 2.2.3 or later.

Table of Contents

Revision Information	iv
Introduction to SQL and PS Series Storage	1
Benefits of Deploying SQL with PS Series Storage	2
Planning and Design Considerations	3
Optimizing Server Memory Resources	5
Planning the SQL Volume Configuration	6
SQL Volume Layout Guidelines	6
Sizing SQL Volumes	10
SQL Volume Layout	11
Distributing Data	13
Choosing the Group RAID Level for SQL	13
Deploying SQL with a PS Series SAN	13
Background Information	13
Basic Steps	14
Configuring SQL to Use PS Series Storage	14
Specifying Destination Folders During SQL Installation	15
Moving Log Files	16
Creating a User Application Database	16
Delaying Loading SQL Server	17
Optimizing SQL Server	17
Defragmenting NTFS Disks	17
Migrating SQL Databases from DAS to a PS Series SAN	18
Part 1: Moving User and System Databases	18
Part 2: Changing the Default Database Location	18
SQL Data Management	20
Defragmenting SQL File Tables	20
Expanding SAN Storage Online	21
Backing Up Data Using VSS	21
Summary	22
Documentation and Customer Support	22

Revision Information

The following table describes the release history of this Technical Report.

Technical Report	Date	Document Revision
1.0	10/10/2005	Initial Release

The following table shows the software versions used for the preparation of this Technical Report.

Vendor	Model	Software Revision
Microsoft®	Windows® Server™ 2003 Enterprise Edition	Service Pack 1
Microsoft	SQL Server 2000	Version 8.00.2039 Service Pack 4
Microsoft	iSCSI Software Initiator	Version 2.0
QLogic™	QLA4010 iSCSI HBA	BIOS: 1.11 Firmware: 03.00.00.04
QLogic	SANsurfer Manager	4.01.00
QLogic	SCSIport Miniport Driver	2.1.0.3
QLogic	STORport Miniport Driver	2.1.0.8
EqualLogic®	Auto-Snapshot Manager for Windows	Version 1.1
EqualLogic	PS Series Firmware	Version 2.2.3

Introduction to SQL and PS Series Storage

As workloads grow in the industry today, SQL Server databases continue to increase in size, requiring more and more storage capacity. For example, when transaction rates increase, the storage subsystem must be able to handle the associated growth in disk space usage and I/O throughput. In addition, as database sizes increase, it is increasingly difficult to complete tape backup operations in a timely manner. If an outage occurs, it can take a business many hours or even days to restore service and recover from the outage.

SQL Server provides the enterprise data management platform your organization needs to adapt quickly to a fast-changing environment. Benchmarked for scalability, speed, and performance, SQL Server delivers rapid return on your data management investment, with low implementation and maintenance costs and rapid development of enterprise-class business applications.

A PS Series group—an iSCSI SAN consisting of one or more intelligent storage arrays connected to an IP network—is an excellent storage choice for SQL Server environments. PS Series storage provides fast setup, automated management, easy scalability, multipath I/O support, SAN boot support, and remote volume replication. Not only does a PS Series SAN improve storage utilization efficiency and availability, it also delivers flexibility and ease-of-management, regardless of SAN scale.

In addition, by utilizing the Microsoft Windows Server 2003 Volume Shadow Copy Service (VSS), the SQL Server VSS writer, and the EqualLogic Auto-Snapshot Manager for Windows VSS provider, you can improve backup and restore operations through point-in-time copies of data called *shadow copies* or *snapshots*.

Thus, by deploying SQL Server with a PS Series SAN, businesses can combine the industry-leading database application with reliable, scalable, and high-performance disk storage to meet the ever-expanding needs of enterprise users.

To get the maximum benefits from SQL and a PS Series SAN, you should adhere to the best practices for SQL Server 2000, as outlined by Microsoft and industry experts. In addition, this Technical Report describes requirements and recommendations for deploying SQL Server with PS Series storage arrays, including best practices for performance, reliability, scalability, flexibility, and recoverability.

Key issues in this report include the following:

- Design considerations, including availability, performance, scalability, and management.
- How to set up a PS Series group and volumes.
- Benefits of booting servers from a SAN.
- How to set up an SQL server, including optimizing the server and connecting to volumes.
- Configuring SQL to use PS Series volumes.
- Expanding SAN capacity and file systems without affecting availability.
- Backing up SQL data.

Benefits of Deploying SQL with PS Series Storage

The benefits of deploying SQL server with a PS Series SAN are as follows:

- **Rapid deployment and configuration of storage** – In less than 20 minutes, a PS Series SAN can be operating and providing storage for SQL. A simple setup utility lets you quickly configure an array on the network and create a PS Series group. Automation of complex operations like RAID configuration, disk sparing, data provisioning, and load balancing means that even novices can effectively manage the SAN.
- **Redundant hardware and hot serviceable configuration** – PS Series storage arrays are fully redundant with dual controllers, power supplies, and fans—all of which can be serviced online and without disrupting applications. In addition, support for multipath I/O provides end-to-end redundancy for SQL server storage, ensuring maximum reliability and availability.
- **Data protection.** All data is protected with RAID and spare disks. Combined with “hot” service capabilities, online operation is assured.
- **Simple and immediate storage expansion** – Using modular PS Series storage arrays as the SQL Server storage solution, you can increase SAN storage capacity and performance online, without server or application disruption.
- **SAN boot capability** – Not only do iSCSI host bus adapters (HBAs) increase I/O performance, they also provide the ability to install and boot the Windows operating system from a PS Series volume, increasing disaster tolerance. When server hardware fails, the unit can be quickly removed and replaced with a similarly-configured spare hardware platform. This new platform can be directed to the SAN boot volume and, in minutes, resume providing application services. Other benefits of SAN boot include centralized storage management and reliable and highly available storage resources that eliminate the need for mirrored boot volumes.
- **Network path protection and load balancing** – Using multiple NICs or iSCSI HBAs with the Microsoft iSCSI Software Initiator Version 2.0, you can configure multipath I/O and increase the reliability and performance of SQL Server. Also known as MPIO, multipath I/O enables the dynamic load balancing of iSCSI SAN traffic across redundant paths between the SQL server and the PS Series SAN.
- **Advanced Management features.** PS Series storage comes standard with a comprehensive set of features including automatic load balancing, virtual volume management, snapshots for instant backup and restore, volume cloning for rapid server provisioning, auto-replication capability, multipath I/O support, and cluster support for comprehensive disaster recovery.
- **VSS-based backups** – EqualLogic provides a VSS provider, called Auto-Snapshot Manager for Windows, that interacts with the PS Series storage, Microsoft Windows Server 2003, the SQL Server VSS writer, and a VSS requestor backup application to dramatically improve backup operations by creating flexible, space-efficient point-in-time copies of data called snapshots or shadow copies.
- **Remote site volume replication** – With PS Series auto-replication capability, SQL Server data can be automatically transferred to remote data centers, protecting the data from serious failures, ranging from the destruction of the volume to a complete site disaster—with no impact on data availability or performance.

Planning and Design Considerations

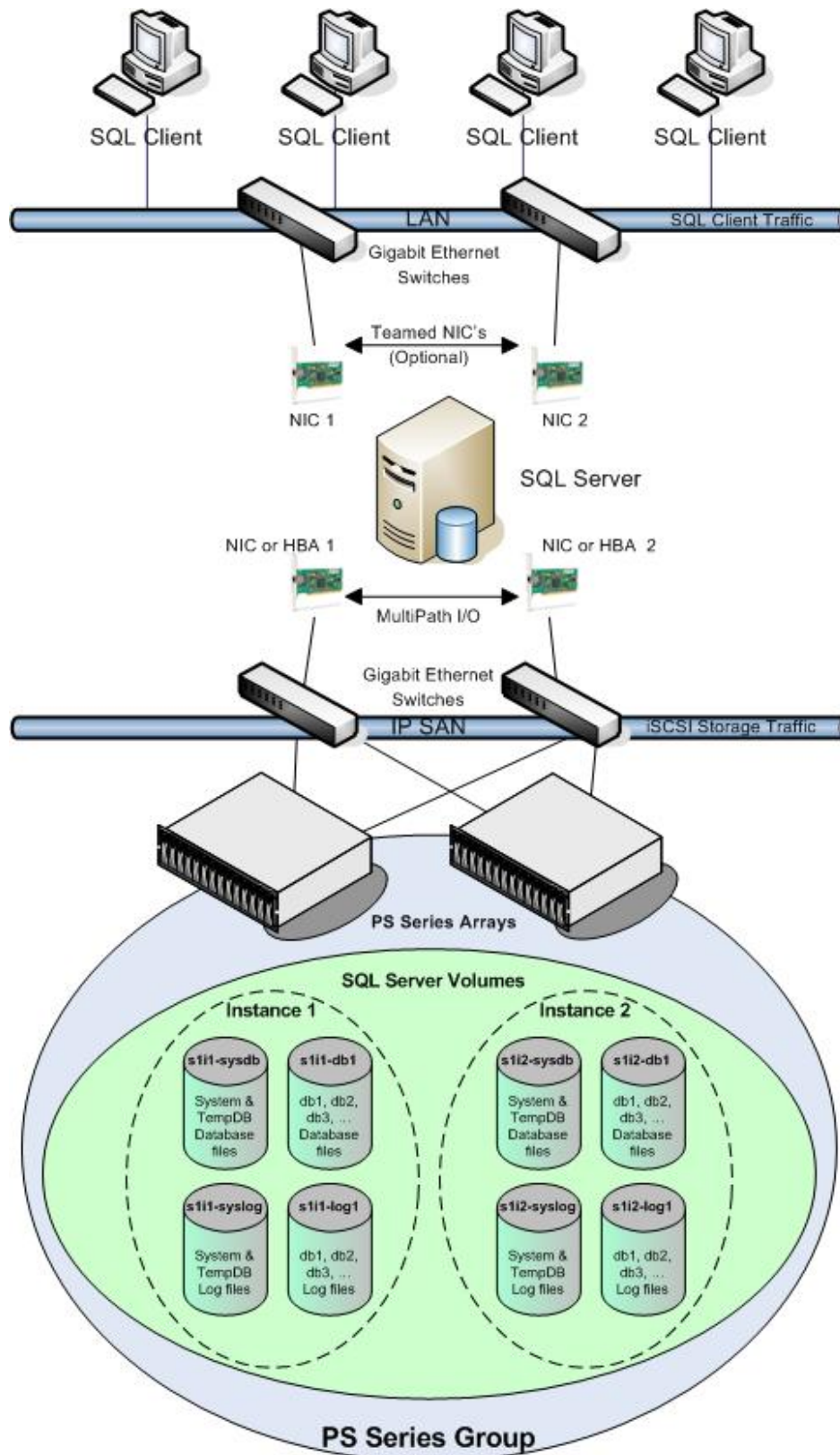
When designing an SQL Server environment, you should understand these challenges:

- **Reliability, availability, and serviceability** – The environment must be robust, resilient, and easily repaired by hot swapping components while the system remains available.
- **Performance** – The environment must optimize user productivity and response time.
- **Scalability** – As your workload grows and capacity needs increase, the environment must be able to accommodate changes without affecting users.
- **Ease of management** – Setup and configuration, backup and recovery, and day-to-day administration should be handled easily and have minimal impact on operations and users.

It is critical to focus on these challenges *before* you begin the initial SQL installation. The Technical Report, *Deploying Microsoft Windows Server 2003 in an iSCSI SAN*, located on the EqualLogic Customer Support website, describes how to design a robust server and network environment that uses PS Series storage to overcome the challenges IT managers face today.

Figure 1 shows a SQL server configured to operate in a high-availability environment for a medium size organization.

Figure 1 – SQL High Availability Design for a Medium Size Organization



Optimizing Server Memory Resources

It is important to properly configure your Windows 2003 server for optimal memory utilization. Memory configuration can affect the database I/O load and performance. An SQL server configured with insufficient memory will generate heavier I/O loads to its databases, while configuring sufficient memory will make your SQL installation more efficient in its use of storage resources. By making your server as tuned and efficient as possible, you can get the maximum benefit from using PS Series storage.

Microsoft's published minimum memory requirement for SQL Server 2000 Enterprise Edition is 64 MB, while 128 MB or more is recommended. If you are running either Standard Edition or Developer Edition, you must have at least 64 MB.

The memory manager component of SQL Server 2000 eliminates the need for manual management of the memory available to SQL Server. When SQL Server starts, it dynamically determines how much memory to allocate, based on how much memory the operating system and other applications are currently using. As the load on the computer and SQL Server changes, the allocated memory also changes.

If you run the Microsoft SQL Server Best Practices Analyzer Tool, it will detect if one or more settings need to be corrected. For more information, see the following Microsoft TechNet article:

<http://www.microsoft.com/downloads/details.aspx?displayla%20ng=en&familyid=B352EB1F-D3CA-44EE-893E-9E07339C1F22&displaylang=en>

If your SQL server has more than 2 GB of memory, and you are running SQL Server 2000 Enterprise Edition, you should consider tuning the server according to the Microsoft guidelines *How to configure memory for more than 2 GB in SQL Server*:

<http://search.support.microsoft.com/?scid=kb:en-us:274750>

For SQL Server 2000 memory optimization details, see the Microsoft Knowledge Base article *How to determine proper SQL Server configuration settings*:

<http://support.microsoft.com/default.aspx?scid=kb:en-us:319942>

In addition, you will need to tune the operating system. This procedure is detailed in the Knowledge Base article *Large memory support is available in Windows Server 2003 and in Windows 2000*:

<http://search.support.microsoft.com/kb/283037/>

Planning the SQL Volume Configuration

SQL Server has a number of different files with different functions that make up the storage installation. A SQL Server instance has multiple databases. For example, each SQL instance includes the following:

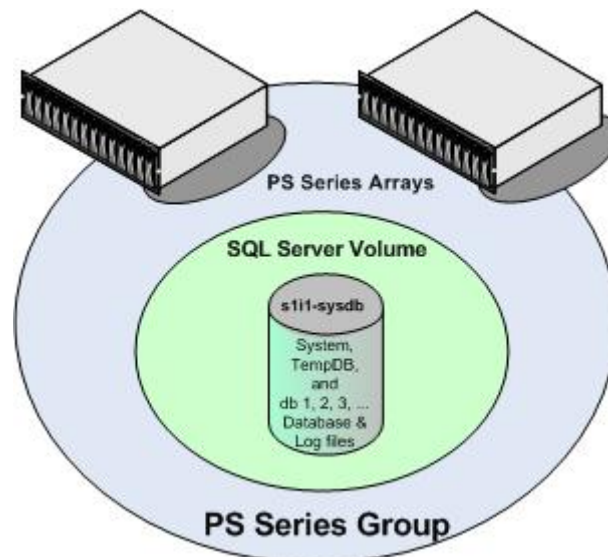
- System files (`master.mdf`, `master.ldf`, `model.mdf`, `model.ldf`, `msdbdata.mdf`, `msdblog.ldf`, `tempdb.mdf`, and `tempdb.ldf`) hold the system information and temporary database and logs.
- Sample database files (`pubs.mdf`, `pubs_log.ldf`, `northwnd.mdf`, and `northwnd.ldf`) are created if the option to create sample databases is chosen at installation.
- Database files (`*.mdf`) are created by the user for specific applications and hold all the committed information.
- SQL transaction logs (`*.ldf`) are created when the user creates a database for a specific application. They record all changes to an SQL database prior to the changes actually being committed.

SQL Volume Layout Guidelines

The goal when setting up SQL volumes in a PS Series group is to optimize manageability for the size of your environment. You should consider how you will monitor disk usage, in addition to how you will perform backup operations. Some guidelines for creating and setting up SQL volumes are as follows:

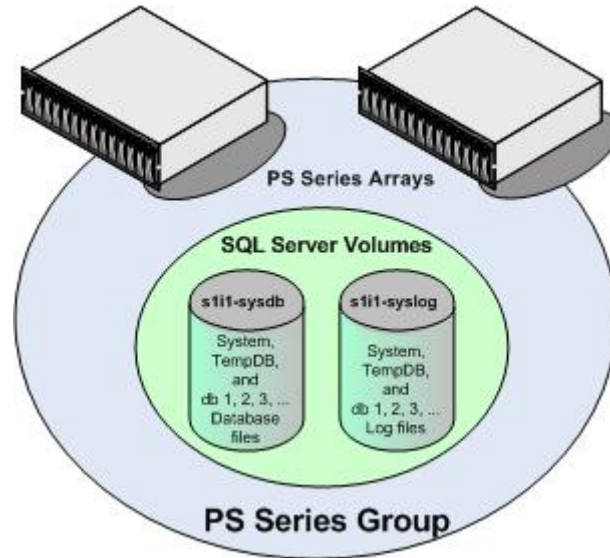
- **Place databases and logs on their own volumes.** When designing the volume layout scheme for SQL Server, there are many options. However, some general guidelines can be identified, based on the size of your business. By default, the SQL installation places the system and user databases and logs on the same volume. This may be suitable for some organizations and is shown in Figure 2.

Figure 2 – Default SQL Volume Layout – Single-Volume Example



Alternately, a small business that needs a simple SQL deployment could put database files (both system and user) on one volume and the logs on another volume, as shown in the following figure. This increases performance and simplifies administration.

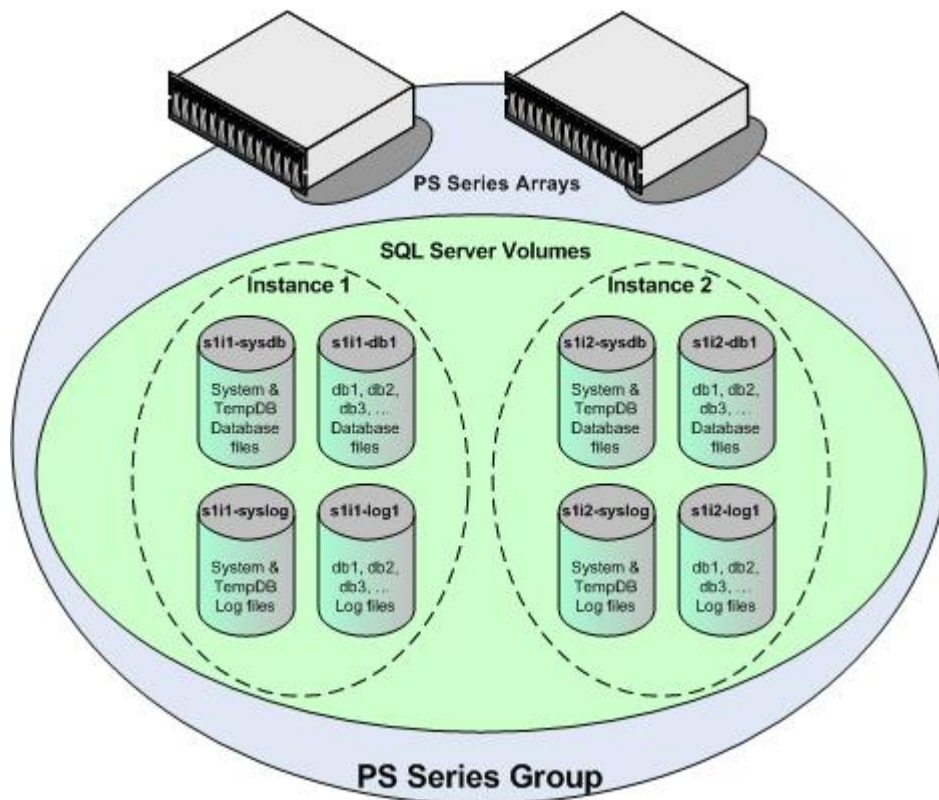
Figure 3 – SQL Volume Layout for a Small Organization – Two-Volume Example



A medium-size business that supports multiple databases inside a single SQL instance could create a volume for system databases, a volume for user databases, and a volume for logs. This provides the flexibility to add more user databases and grow your SQL environment easily.

If an SQL Server deployment requires several SQL Server instances, you could create a separate set of volumes for each instance. This configuration will enable you to easily manage multiple instances and their databases and provides a good recovery model.

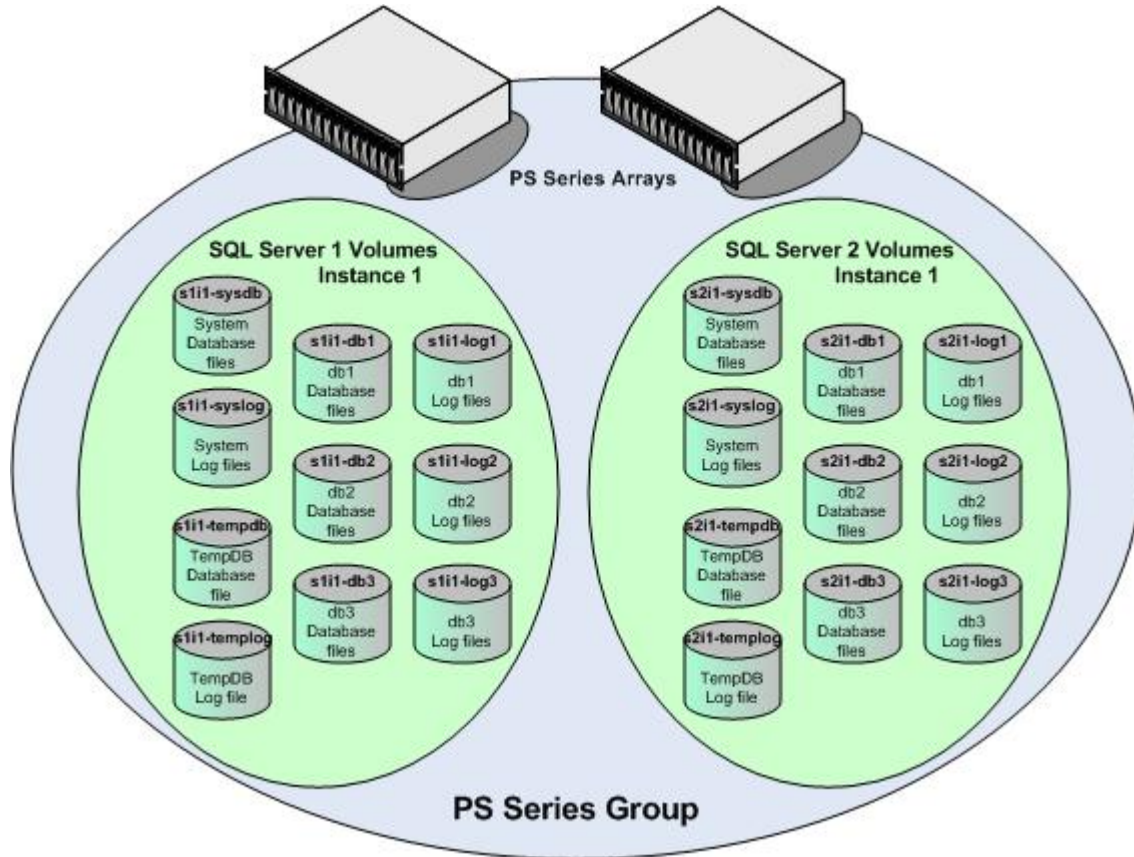
Figure 4 – SQL Volume Layout for a Medium-Size Organization – Multiple Volumes/Multiple Instances Example



A large business running an Enterprise level SQL server instance will need a more complex volume layout. For example, you need a volume for near-static system databases, a volume for the temporary database (TempDB), multiple volumes for user databases, and one or more volumes for logs. It is important to isolate TempDB because of its dynamic nature. This temporary database, which gets recreated every time SQL is started, grows as user database activity increases. Impromptu queries and similar activities will also cause TempDB to grow. With enough activity, TempDB could increase to a size that will consume your entire volume. Isolating this database in its own volume gives you more flexibility and simplifies management.

Note that, if a second database instance is required, it could be located on the same SQL server or on a different server, as described in Figure 5.

Figure 5 – SQL Volume Layout for a Large Organization – Advanced Configuration Example



- **Create the database and log files as close to their estimated final size.** When creating a database, enabling the “Automatically grow file” feature (the default) is recommended to prevent the database or transaction log from becoming full unexpectedly. One of the biggest benefits of PS Series storage is that volumes can be increased dynamically without incurring server downtime, although you must monitor space usage and manually resize the volume.

However, you should try to obtain an accurate estimate of the final size of the database and log files for the following reasons:

- Every time the “Automatically grow file” feature is activated, it consumes CPU and I/O resources.
- The more times that SQL Server has to automatically grow the size of a transaction log, the more transaction log virtual files have to be created and maintained by SQL Server. This increases recovery time if your transaction logs need to be restored.
- If you have fragmentation at the operating system (or file) level that can be fixed with a third-party product like Disk Keeper™, you must stop SQL Server in order for it to work. A *.mdf file will be fragmented only if you did not have enough contiguous space on the volume when it was created or if it was resized at a later date and there was not enough contiguous space. Therefore, you should periodically defragment the file system, especially after you expand an SQL volume.

- **Defragment SQL file tables.** Fragmentation can occur at the file table level. DBCC SHOWCONTIG allows you to determine the amount of index fragmentation. If you have clustered indexes, you can use DBCC DBREINDEX or DBCC INDEXDEFRAG to correct the fragmentation..
- **Increase the automatic growth percentage.** Every time an SQL Server file automatically grows, it must find unallocated space on the volume in the PS Series group. Usually, the space that is allocated will not be contiguous with the current volume space. In fact, the space may come from multiple locations in the PS Series group. The higher the growth percentage, the less often the database will automatically grow, thus limiting the degree of fragmentation.
- **Disable the automatic shrinking of databases.** By default, the auto shrink option is disabled. If you allow SQL Server to automatically shrink its size, the space it deallocates will leave fragments of unallocated disk space in the PS Series group. Automatic shrinking of databases should be disabled to reduce fragmentation and overhead. You can use the SQL Server Enterprise Manager to disable the database from shrinking automatically.

Sizing SQL Volumes

To size volumes for SQL Server databases, consider the space needed and the desired throughput. Determining the initial size of the database and log volumes depends on many factors, including how you plan to use SQL Server.

Considerations for sizing SQL Server volumes include the following:

- The number of transactions that will be performed and written each day. The more transactions you perform, the more transaction log space you need.
- The number of queries that will be performed each day. The more large queries (with sorting) you perform, the more TempDB system database space you need.
- Whether your applications will create additional tables inside the databases.
- Whether the database will only be populated with a lot of data and support queries, or whether there will be many writes to the database.
- Will you be adding more databases in the future and at what rate?

The best way to roughly determine the space required for a database volume is to base it on previous experience with similar databases or consult with the database administrator or application developer. You can also obtain a theoretical size that is based on the number of bytes in the fields and the number of rows.

The ability to increase volume size dynamically as your business grows and database usage increases is an important feature of a PS Series SAN. If you underestimate the size of a PS Series volume, it is quick and easy to increase its size. However, with proper planning, using this feature should be necessary only if unexpected changes to the environment occur (for example, a merger with another company causes the database to grow substantially).

Sizing for the optimal throughput may require you to deploy more disk drives than what is needed to meet the database space requirements. In other words, a 1 TB database will fit on three 400 GB disk drives. However, in general, you will have better performance if you spread the database across four or more 400 GB disk drives, because more drives mean more requests can be handled in parallel. Since a PS Series group dynamically load balances volumes by automatically spreading volume data across all the disks and arrays in the group, the more arrays in a group, the more disk drives are available for use.

SQL Volume Layout

When you perform the initial installation of SQL Server 2000, if you choose the default setting, four system databases will be installed on the volume you specify at the installation prompt. The databases have their own set of files and logs and do not share those files with other databases.

The system databases are shown in the following table. By default, they will all be installed in the same volume. Depending on the size of your organization, the table suggests volume locations to which you can move those databases and logs after the installation completes.

Note: For the purposes of this Technical Report and the examples shown next, a sample volume naming convention is used for the first SQL instance and its associated databases. The convention has been chosen so that you can scale up, as needed, with minimal changes to existing configurations. For example, in the volume name `sl11-sysdb`, the “s1” represents the server (for example, server 1), and the “i1” represents the database instance (for example, instance 1). This convention is used only as a guideline and can be easily changed to match your environment.

System Database Files

System Database Files	Physical File Name	Default Size, Typical Setup	PS Series Volume / Windows Server Disk Name			
			Default	Small Org	Medium Org	Large Org
master data	master.mdf	11.0 MB	sl11-sysdb	sl11-sysdb	sl11-sysdb	sl11-sysdb
master log	mastlog.ldf	1.25 MB	sl11-sysdb	sl11-syslog	sl11-syslog	sl11-syslog
model data	model.mdf	0.75 MB	sl11-sysdb	sl11-sysdb	sl11-sysdb	sl11-sysdb
model log	modellog.ldf	0.75 MB	sl11-sysdb	sl11-syslog	sl11-syslog	sl11-syslog
msdb data	msdbdata.mdf	12.0 MB	sl11-sysdb	sl11-sysdb	sl11-sysdb	sl11-sysdb
msdb log	msdblog.ldf	2.25 MB	sl11-sysdb	sl11-syslog	sl11-syslog	sl11-syslog
tempdb data	tempdb.mdf	8.0 MB	sl11-sysdb	sl11-sysdb	sl11-sysdb	sl11-tempdb
tempdb log	templog.ldf	0.5 MB	sl11-sysdb	sl11-syslog	sl11-syslog	sl11-templog

In addition to the four system databases that are installed by default, two sample user databases are installed, as shown in the following table.

Sample User Database Files

Sample User Database Files	Physical File Name	Default Size, Typical Setup	PS Series Volume / Windows Server Disk Name			
			Default	Small Org	Medium Org	Large Org
pubs data	pubs.mdf	1.0 MB	sl1l-sysdb	sl1l-sysdb	sl1l-sysdb	sl1l-sysdb
pubs log	pubs_log.ldf	0.75 MB	sl1l-sysdb	sl1l-syslog	sl1l-syslog	sl1l-syslog
northwind data	northwnd.mdf	3.0 MB	sl1l-sysdb	sl1l-sysdb	sl1l-sysdb	sl1l-sysdb
northwind log	northwnd.ldf	1.0 MB	sl1l-sysdb	sl1l-syslog	sl1l-syslog	sl1l-syslog

Note: The default sizes of the system database files and the sample user database files vary slightly in different editions of SQL Server 2000. The default sizes are small and should *not* be used as a measure for volume sizing.

Once SQL is installed, you can create additional volumes for the database and logs for the SQL applications you will use. The following table shows an example.

User Application Database Files

User Application Database Files	Physical File Name	Default Size, Typical Setup	PS Series Volume / Windows Server Disk Name			
			Default	Small Org	Medium Org	Large Org
sl1l-db1 data	sl1l-db1_data.mdf	1.0 MB	sl1l-sysdb	sl1l-sysdb	sl1l-db1	sl1l-db1
sl1l-db1 log	sl1l-db1.ldf	1.0 MB	sl1l-sysdb	sl1l-syslog	sl1l-log1	sl1l-log1
sl1l-db2 data	sl1l-db2.mdf	1.0 MB	sl1l-sysdb	sl1l-sysdb	sl1l-db1	sl1l-db2
sl1l-db2 log	sl1l-db2.ldf	1.0 MB	sl1l-sysdb	sl1l-syslog	sl1l-log1	sl1l-log2
sl1l-db3 data	sl1l-db3.mdf	1.0 MB	sl1l-sysdb	sl1l-sysdb	sl1l-db1	sl1l-db3
sl1l-db3 log	sl1l-db3.ldf	1.0 MB	sl1l-sysdb	sl1l-syslog	sl1l-log1	sl1l-log3

Notes: The default sizes of the user databases are inadequate and should not be used as a guideline for volume sizing.

It is recommended that you create your user databases and logs as close to their ultimate size, as possible.

If you want to create a separate volume for the temporary database (TempDB), in general, the size of should be approximately 20 percent of the size of your largest SQL Server database.

Distributing Data

The load on a database file consists of random reads and random writes. The load on a log file consists of sequential writes during normal operation and sequential reads during the restoration of a database. When using direct attached storage (DAS), the write profile of these files can have a significant impact on performance because of the potential for I/O bottlenecks. However, the distinction in the write profile is less important when using a PS Series group because I/O is automatically distributed across disks and arrays, dynamically balancing the load.

Also, although distributing databases and logs across spindles in a DAS configuration provides more flexibility when backing up data, in the case of a PS Series group, if VSS is used for backup operations, data location has little relevance, because snapshots of volumes can be created simultaneously in a coordinated manner.

The more arrays you have in the PS Series group, the more disks are available for the group to distribute data across.

Choosing the Group RAID Level for SQL

Before creating a PS Series group, you should determine which RAID level, either RAID 10 or RAID 50, to configure on the group members (storage arrays). For large SQL implementations that want optimal performance, RAID 10 is a good choice for the group RAID level. However, for many SQL implementations, RAID 50 can be used to provide maximum storage capacity, in addition to good performance.

For more information on RAID levels in a PS Series group, see the *Understanding Group RAID Levels* Technical Report on the EqualLogic Customer Support website.

Deploying SQL with a PS Series SAN

The following sections describe SQL Server requirements and recommendations and the basic tasks for deploying SQL with a PS Series SAN.

Background Information

It is recommended that you are familiar with the following documents:

- Microsoft SQL Server 2000 – Deployment:
<http://www.microsoft.com/technet/prodtechnol/sql/2000/deployment.mspx>
- Technical Report *Deploying Microsoft Windows Server 2003 in an iSCSI SAN*, located on the EqualLogic Customer Support website.

Follow these steps to deploy SQL Server with PS Series storage:

1. **Set up a PS Series group and create the volumes required for the SQL environment.** See *Planning the SQL Volume Configuration* for volume layout and sizing guidelines. Be sure to create access control records for each volume to allow the appropriate servers access to the volume. Also, reserve snapshot space for each volume if you will be creating snapshots or using Auto-Snapshot Manager for VSS backups.

Optionally, you can also create volumes for booting SQL servers from the SAN, a volume for a quorum disk if you are using Microsoft Cluster Server, or a VSS control volume if using Auto-Snapshot Manager.

See the PS Series *QuickStart* and *Group Administration* manuals and the EqualLogic Best Practice Technical Report *Deploying Microsoft Windows Server 2003 in an iSCSI SAN*, located on the EqualLogic Customer Support website, for more information.

2. **Optimize the SAN network for performance.** See the EqualLogic Technical Report *Network Connection and Performance Guidelines* for more information.
3. **Optionally, configure the environment so that SQL servers can boot from the SAN.** For details, see the EqualLogic Technical Report *QLogic QLA4010: Booting Windows 2003 From a PS Series Group SAN*
4. **Prepare and optimize the SQL servers and connect them to volumes.** For details, see the EqualLogic Technical Report *Deploying Microsoft Windows Server 2003 in an iSCSI SAN*.
5. **Install SQL on the server and configure it to use the PS Series storage.** See *Configuring SQL to Use PS Series Storage* for information about specifying the iSCSI disks for the database and log files.

SQL Server 2000 installation and deployment requirements are well documented in the *SQL Server 2000 Deployment Guide*, which can be found at:

<http://www.microsoft.com/technet/prodtechnol/sql/2000/deployment.msp>

After deployment, you can expand iSCSI disks online, expand PS Series group capacity, and backup volumes, as described in *SQL Data Management*.

Configuring SQL to Use PS Series Storage

The following sections take you through the procedure for using SQL Server with PS Series storage. We will assume a medium-size organization, with one user-created SQL database in one SQL instance on a single server.

Follow the recommendations in the EqualLogic Best Practice Technical Report *Deploying Microsoft Windows Server 2003 in an iSCSI SAN*, located on the EqualLogic Customer Support website to set up your server and volumes.

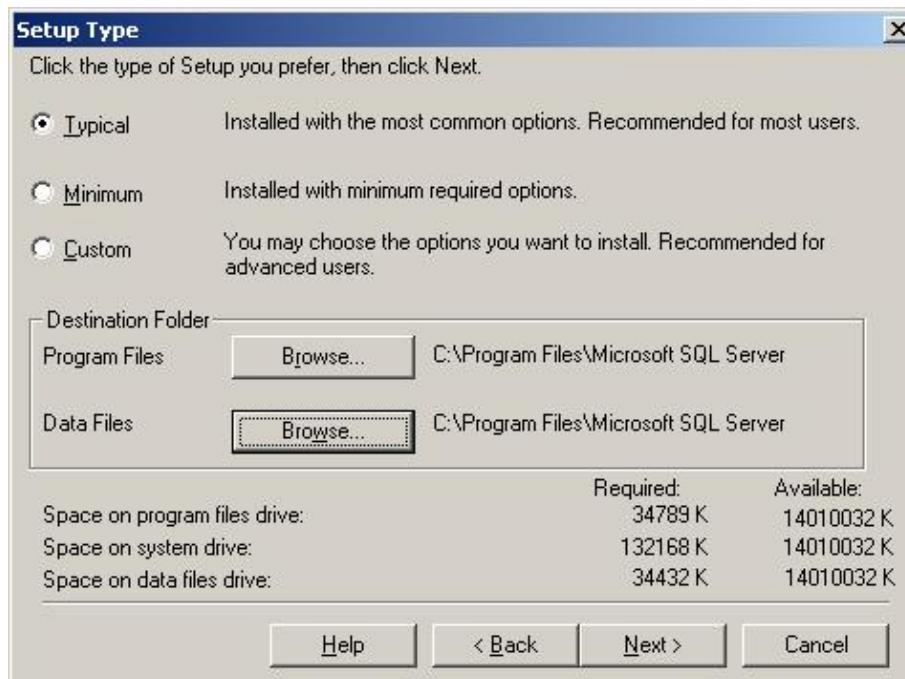
Using the recommendations and guidelines outlined in *Sizing SQL Volumes*, the following volumes should already be created in the PS Series group, and the server should already be connected to the volumes using the specified drive letters:

Drive	Volume Name	Description
E:\	s1i1-sysdb	System and TempDB databases
F:\	s1i1-syslog	System and TempDB logs
S:\	s1i1-dbl	User database data
T:\	s1i1-log1	User database transaction logs

Specifying Destination Folders During SQL Installation

At this point if you have not already done so, start the installation of SQL Server on the server. During the SQL Server installation, you will be presented with the Setup Type dialog box, which enables you to specify the destination folders in the **Program Files** and **Data Files** fields.

Figure 6 – SQL Setup Type – Destination Folders



The SQL setup program does not allow you to place individual SQL files in specific locations.

In the Setup Type dialog box, the **Program Files** field can remain the default system drive, or you can specify another location, as desired.

In **Data Files** field, specify an iSCSI disk (that is the drive letter associated with a connected PS Series volume) and directory. Using the example configuration, this should be the volume used for system & TempDB databases, you would specify the following in the **Data Files** field:

E:\MSSQL\Data

This will place the system database and log files, the TempDB database and log files, and the sample database and log files in this directory.

Moving Log Files

Once the SQL installation is complete, it is recommended that you move the log files for the system, TempDB, and sample database from the data files location to the iSCSI disk you want to use for system and TempDB logs. Using the example configuration, this would be the following:

F:\Logs

For information on moving log files, see the Microsoft Knowledge Base article *Moving SQL Server databases to a new location with Detach/Attach*:

<http://support.microsoft.com/default.aspx?scid=kb:en-us:224071>

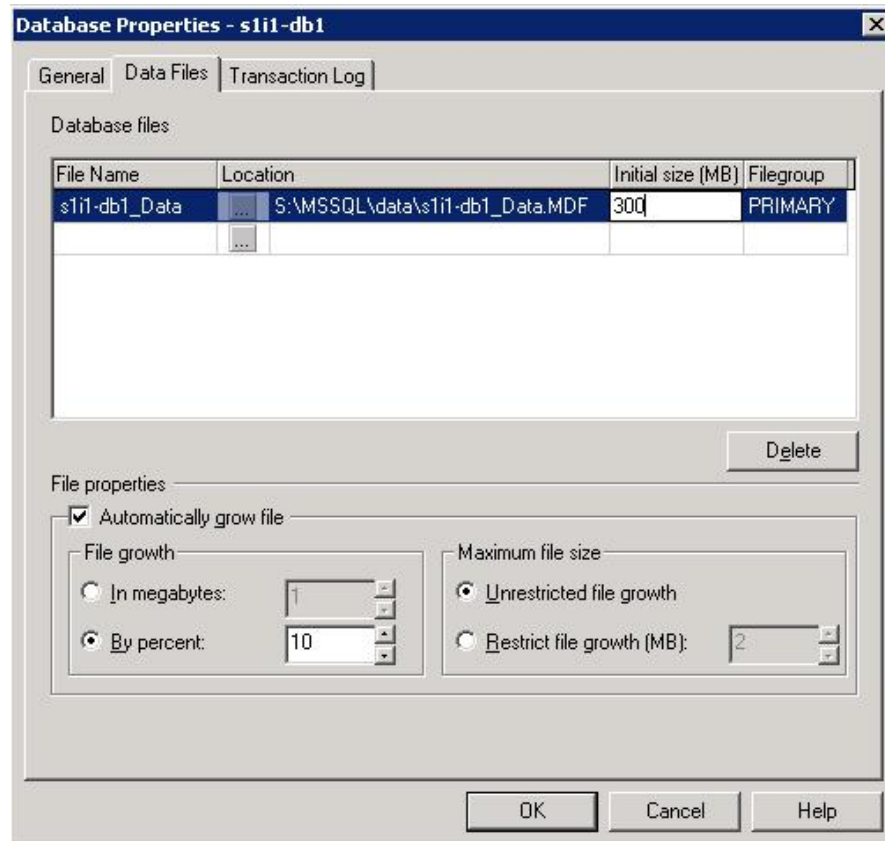
Moving log files requires that you have exclusive access to all user and system databases.

Creating a User Application Database

To create a new user database for your specific application, launch the SQL Server Enterprise Manager, navigate to the desired SQL Server, left-click on **Databases**, and select **New Database**.

Enter a name for the new database (s1i1-db1 using the example configuration) and then select the **Data Files** tab. Change the location of the database file to the iSCSI disk for this purpose (using the example configuration, the location would be iSCSI disk s:) and specify the desired initial size of the database. Again, consult with your database administrator or application developer to determine the appropriate size.

Figure 7 – SQL Database Properties – Data Files



Next, click the **Transaction Log** tab and specify the location for the Log File and specify the initial size of the log, as desired. Using the example configuration, the location would be iSCSI disk T:)

When the SQL installation and setup is complete, using the example configuration, you should have the following database and log file paths configured:

Path	Description
C:\Program Files\Microsoft SQL Server\MSSQL	Program files
E:\MSSQL\Data	System and TempDB databases
F:\Logs	System and TempDB logs
S:\MSSQL\Data\s1i1-db1_Data.MDF	User database data
T:\Logs\s1i1-db1_Log.LDF	User database transaction logs

Delaying Loading SQL Server

When using PS Series volumes connected through the Microsoft iSCSI Software Initiator in an SQL environment, you must prevent the SQL server from attempting to start before the iSCSI volumes are online. Therefore, you must add a dependency to the Registry information to delay the startup of SQL services until after the iSCSI service has started.

For more information on this topic, refer to Microsoft Knowledge Base article 193888, *How to Delay Loading of Specific Services*.

<http://support.microsoft.com/default.aspx?scid=kb:en-us:193888>

Optimizing SQL Server

Run the Microsoft SQL Server Best Practices Analyzer Tool to detect if one or more settings need to be corrected. For more information, see the following Microsoft TechNet article:

<http://www.microsoft.com/downloads/details.aspx?displayla%20ng=en&familyid=B352EB1F-D3CA-44EE-893E-9E07339C1F22&displaylang=en>

In addition, follow the memory recommendations found in *Optimizing Server Memory Resources*, as well as the server optimizing recommendations in the EqualLogic Best Practice Technical Report *Deploying Microsoft Windows Server 2003 in an iSCSI SAN*, located on the EqualLogic Customer Support website.

Defragmenting NTFS Disks

Before putting a new Windows Server 2003 server into production, and periodically after, you should use third-party disk defragmenter tools for Windows Server 2003 to defragment the NTFS disks (PS Series volumes). A list of Microsoft Certified disk defragmenter tools can be found at:

<http://www.microsoft.com/windows/catalog/server/default.aspx?subID=22&xslt=globalsearch&qu=defrag&scope=0>

Migrating SQL Databases from DAS to a PS Series SAN

If you are currently using a local system disk, DAS, or other storage media for your current SQL configuration, you can easily migrate databases and logs to volumes in a PS Series group and gain the performance and scalability benefits of an iSCSI SAN.

The example below will move the SQL databases and logs from the local `D:` drive to the `S:` and `T:` iSCSI disks associated with two PS Series volumes in the SAN.

Note: As with any mission critical application that will have its configuration modified, it is suggested that you perform a full backup of all databases and logs before migrating. Also, when performing the migration, you should have exclusive access to the databases being moved.

This procedure has two parts. The first requires moving the existing user and system databases to volumes in the PS Series group. The second part is configuring SQL so that any new user databases are created on volumes in the PS Series group.

Part 1: Moving User and System Databases

Configure the PS Series group with volumes that will hold the SQL data and logs, as described in *Planning the SQL Volume Configuration*. Configure your server to connect to these volumes.

For example, map `S:` to the user database data volume, and map `T:` to the user database transaction log volume. See the EqualLogic Best Practice Technical Report *Deploying Microsoft Windows Server 2003 in an iSCSI SAN* for more information.

To move your existing user and system databases, refer to the Microsoft Knowledge Base article *Moving SQL Server databases to a new location with Detach/Attach*:

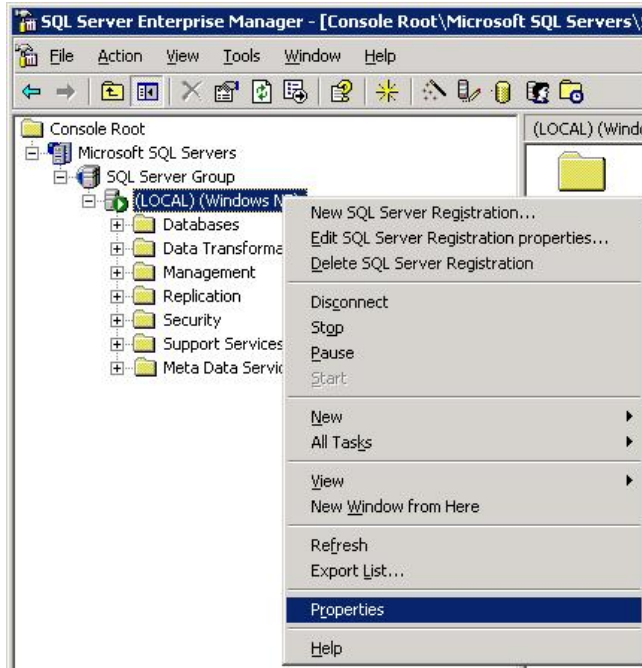
<http://support.microsoft.com/default.aspx?scid=kb;en-us;224071>

This procedure requires that you have exclusive access to all user and system databases.

Part 2: Changing the Default Database Location

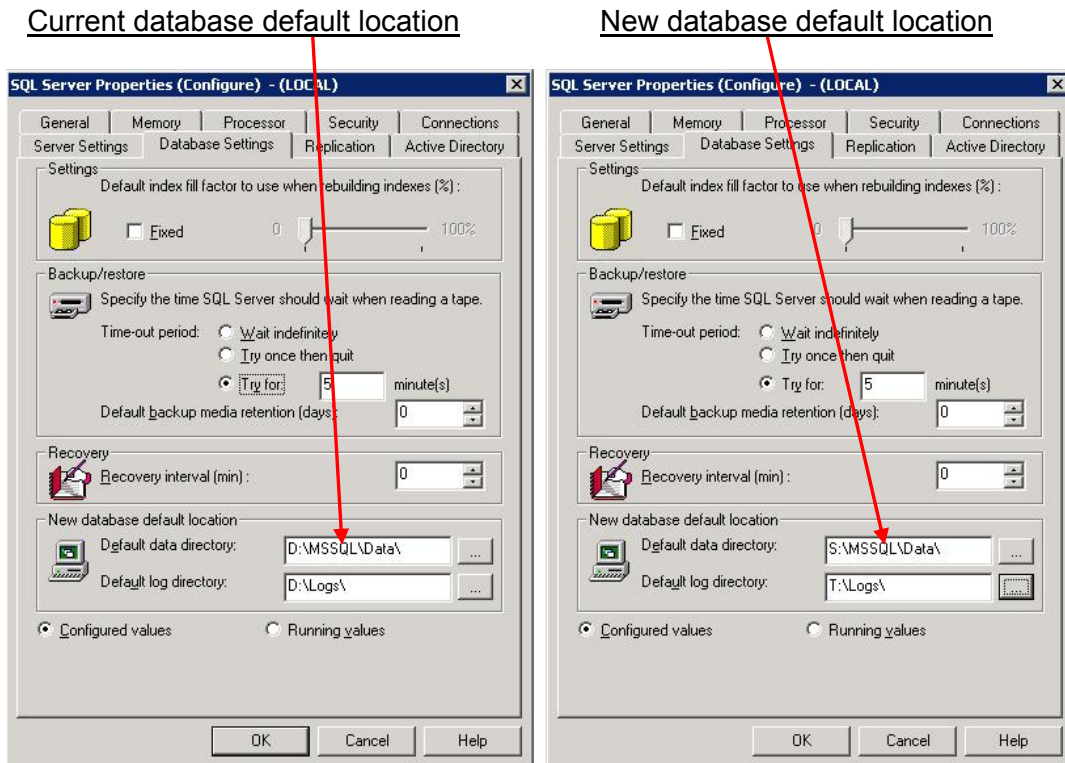
1. Open the SQL Enterprise Manager, right-click the SQL instance whose location you want to change, and click **Properties**. If you are performing this operation on the server that is running SQL, the object will be labeled LOCAL.

Figure 8 – SQL Server Enterprise Manager – Selecting Properties



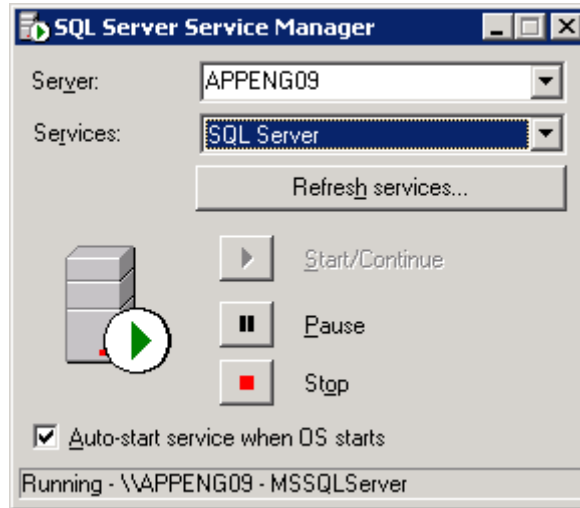
2. Click the **Database Settings** tab in the SQL Server Properties dialog box.
3. In the **New database default location** field, specify the new data and log directory locations.

Figure 11 – SQL Server Properties – Database Settings



4. Once you have specified the new default locations, click **OK**.
5. Immediately stop the SQL Server database by using the SQL Server Service Manager. Select any other services that are running with dependencies on SQL in the **Services** drop-down list and stop them also.

Figure 12 – SQL Server Service Manager – Stopping Services



6. Start your SQL Server database by using the SQL Server Service Manager. Select and start any other services that are running with dependencies on SQL in the **Services** drop-down list.
7. Test and verify that your SQL instance started correctly.

SQL Data Management

Backing up your data as well as monitoring for disk space usage are two important disk management tasks. You should monitor your SQL volumes so they do not fill up unexpectedly. When a volume starts to get full, you may want to consider increasing the volume size, as described below.

Defragmenting SQL File Tables

Not only will the file system fragment, but, the SQL file tables can also fragment over time which can adversely affect workload performance.

For more information, see *Microsoft SQL Server 2000 Index Defragmentation Best Practices*

<http://www.microsoft.com/technet/prodtechnol/sql/2000/maintain/ss2kidbp.mspx>

Expanding SAN Storage Online

As the storage requirements grow for each application, PS Series storage can be easily expanded, online and with no disruption to users.

For example, you can increase the size of a PS Series volume using the Group Manager GUI or the command line interface (CLI). You must then enable the operating system to recognize the size increase. The additional space will be immediately available for use. This procedure is described in the EqualLogic Technical Report *Microsoft Windows: Expanding Basic Disk Volumes*.

If the PS Series group does not have sufficient free space to increase a volume size or add new volumes, you will need to expand the group. To do this, simply add another array (member) to the group. See the PS Series *QuickStart* or the *Group Administration* manual for more information.

Backing Up Data Using VSS

You can back up SQL data residing on a PS Series storage array using any backup application that supports backing up SQL. However, to leverage the capabilities of Volume Shadow Copy Service (VSS) available in Windows Server 2003, SQL Server 2000, and EqualLogic Auto-Snapshot Manager for Windows, you must choose a backup application that supports VSS, in addition to the backup features you desire. Note that backup and restore products are at various stages of VSS requestor implementation.

Backup applications that support VSS and can be used as a VSS requestor include the following:

- VERITAS™ Backup Exec™
- CommVault® Galaxy™ Backup & Recovery
- CA BrightStor® ARCserve® Backup
- Bakbone® NetVault
- Legato® NetWorker

See the backup application documentation for information about using the application as a VSS requestor. See the Auto-Snapshot Manager for Windows Installation and Administration manual for information about using that VSS provider.

For more detailed backup and recovery procedures for specific backup and recovery vendors, see the Technical Reports on the EqualLogic Customer Support website.

Summary

An iSCSI SAN comprised of EqualLogic PS Series storage arrays provides an ideal storage infrastructure for an SQL Server installation. A PS Series SAN brings all the reliability and performance needed for a successful deployment. As the SQL installation grows and the workload increases, the SAN can scale easily, while maintaining availability.

An important part of a successful SQL installation is to follow Microsoft's recommendations for Windows system and SQL server configurations. In addition, you should follow the configuration best practices described in this Technical Report to ensure a robust installation that will meet your needs now and in the future.

Documentation and Customer Support

Visit the EqualLogic Customer Support website, where you can download the latest documentation and firmware. You can also view FAQs, the Knowledge Base, and Technical Reports and submit a service request.

EqualLogic PS Series storage array documentation includes the following:

- *Release Notes*. Provides the latest information about PS Series storage arrays and groups.
- *QuickStart*. Describes how to set up the hardware and start using a PS Series storage array.
- *Group Administration*. Describes how to use the Group Manager GUI to manage a PS Series group. This manual provides comprehensive information about product concepts and procedures.
- *CLI Reference*. Describes how to use the Group Manager command line interface to manage a group and individual arrays.
- *Hardware Maintenance*. Provides information on maintaining the PS Series storage array hardware.

To access the Customer Support website, from the EqualLogic website (www.equallogic.com), click **Support** and log in to a support account. If you do not have an account, create one by clicking the link under the login prompt.

To contact customer support, send e-mail to support@equallogic.com. If the issue is urgent, call 1-877-887-7337 to speak with a member of the customer support team.