



Whitepaper

Scaling SharePoint 2007:
Storage Architecture

Russ Houberg

April 11, 2008

Table of Contents

OVERVIEW	3
PURPOSE	3
BACKGROUND	3
INTRODUCTION	3
OPTIMIZING TEMPDB DATABASE PERFORMANCE	4
<i>Recommendations</i>	4
OPTIMIZING CONTENT DATABASE PERFORMANCE	5
<i>Recommendations</i>	5
OPTIMIZING SEARCH DATABASE PERFORMANCE	6
<i>Recommendations</i>	6
OTHER SHAREPOINT DATABASES	7
<i>Recommendations</i>	7
OTHER I/O PERFORMANCE CONSIDERATIONS	8
<i>Physical Volume File Fragmentation</i>	8
<i>Database Index/Statistics Fragmentation</i>	8
<i>WSS Column Indexes vs Large Content Databases</i>	8
IMPLICATIONS OF A VERY LARGE CONTENT DATABASE	9
MONITORING SHAREPOINT I/O PERFORMANCE	10
SQL SERVER MANAGEMENT FOR SHAREPOINT ADMINISTRATORS	11
<i>Move tempdb</i>	11
<i>Add tempdb data files and set all data file sizes (in KB)</i>	11
<i>Pre-Create a Database (for Content or Search databases)</i>	12
<i>Move a database file from a source volume to a destination volume</i>	13
<i>Redistribute content from a single large database data file to multiple data files</i>	14
SUMMARY	15
<i>Disk I/O Impact on SharePoint Performance</i>	15
<i>The Bottom Line</i>	15
APPENDIX A: REFERENCES	16

Overview

Purpose

The purpose of this whitepaper is three fold. First, this document is intended to provide prescriptive guidance targeted at the storage subsystem for a highly scalable SharePoint implementation in order to facilitate architectural planning. Second, it is intended to provide techniques for monitoring scalability metrics for existing SharePoint implementations. Finally, it provides techniques for modifying storage architecture of SharePoint systems that are suffering from poor performance due to the fact that the I/O burden on the storage subsystem has exceeded its capability.

Background

Back in the 1990s, GM came up with a new slogan for their Pontiac sport sedan line of cars. The related commercial centered on a Hawaiian guy whose grandfather preached the “E Aho Laua” philosophy. This phrase translates to “Wider is Better”. His grandfather was talking about surfboards, Pontiac was talking about cars, and in this whitepaper, we’re talking about the Storage Subsystem for SharePoint.

Most experienced client server developers know that poor database architecture can absolutely wreck the performance of an application. A poor relational design, inefficient stored procedures, or even improperly indexed tables can have a dramatically negative effect on how SQL Server is able to serve data to your application. The same is true not only of logical design but also of physical storage implementation, particularly when large databases are present. Even with a properly architected database, the sheer storage requirements of large database data files cause SQL Server to spend significant additional time to seek the requested data.

While much of the specific detail in the prescriptive guidance of this whitepaper can be found in the [SharePoint Capacity Planning Guide](#), this document will strive to explain a little bit of the “how and why” that back up the recommendations. Note that the information contained in this whitepaper is based on implementing SQL Server 2005. However, most recommendations also apply to SQL Server 2008 as well.

The primary goal is to provide specific SharePoint storage architecture guidance, discuss the reasons behind the recommended limits for SharePoint databases, and discuss how we might extend those limits by implementing SharePoint databases in SQL Server in a more granular way.

In the end, when you’re weighing the cost of that new SAN with the performance requirements for your user base, just remember that when it comes to I/O performance... “Wider is Better”.

Introduction

It is important to review the databases that will most directly impact the functional performance characteristics of SharePoint. The sections immediately below attempt to provide prescriptive guidance for implementing databases in the order of impact to SharePoint performance.

The document will then continue by reflecting on the 100 GB content database size recommendation from Microsoft. Finally, a series of SQL scripts will be provided to help administrators implement or modify the database recommendations in this document.

Optimizing TempDB Database Performance

When tempdb disk I/O is limited, it is difficult to support large SharePoint environments. The tempdb database is used by SQL Server to store intermediate results during query and sort operations. So the tempdb can have a significant impact on any highly scaled client/server application.

The following recommendations are listed in order of least resource intensive (read lower cost) to most resource intensive. Implementing all recommendations will yield the highest performing configuration.

Recommendations

1. Pre-size your tempdb. Since autogrowth will cause this database to grow significantly anyway, this is essentially a free option. So there is no reason not to do this! Dynamic data file extensions are an expensive SQL Server operation. When you consider that every time SQL Server is restarted, the tempdb is reset to its original size, you can see how SQL Server can spend a lot of time extending the tempdb. The tempdb size total size should be preset to 25% of the predicted size of your largest database. "Autogrowth" should remain enabled. If the tempdb data file(s) autogrow beyond your predicted size, then they should be permanently resized again taking into account maximum perceived growth.
2. Add tempdb data files such that the number of data files equals the number of CPU cores present in the SQL Server machine. Each data file (primary .mdf and additional .ndf(s)) should be equal in size. The size of each data file can be calculated using this formula: $[\text{MAX DB SIZE (KB)}] \times [.25] / [\# \text{ CORES}] = \text{DATA FILE SIZE (KB)}$
3. Place the tempdb database file(s) on a RAID 10 logical unit if at all possible.
4. With a multi-core SQL Server, you should have multiple data files. If possible, separate each data file to separate logical units consisting of unique physical disk spindles. The tempdb log file should also be placed on a unique logical unit.
5. Do not allow other data files from other databases to be placed on any of the tempdb logical units.

Optimizing Content Database Performance

SharePoint Content databases contain the bulk of all content in the site collections that comprise the portal instance. In WSS v3.0, not only is all of the content metadata stored in the database, but the binary files themselves related to document library records are also stored in the database. That statement should be qualified with the knowledge that a hotfix (and SP1) is available that will allow the externalization of content. However the API is rudimentary and should be avoided in most circumstances.

The point, however, is that SharePoint content databases grow very rapidly when used in document imaging system or in file share replacement scenarios. With this in mind it is important to consider growth and overhead factors when determining how much content will eventually be stored in a given content database.

Given the wide range of file plan variables for any given organization the formulas for predicting content database size are a moving target and as such are outside the scope of this document. However, once raw file storage quantities are determined, the following formula may be used to estimate database overhead:

Database Overhead Formula:

Low: $1.2 * [\text{Raw Storage Size}] = \text{ContentDB Size}$

High: $1.5 * [\text{Raw Storage Size}] = \text{ContentDB Size}$

Once the storage requirements have been determined, once again attention should be paid to maximizing I/O performance of the content databases.

The following recommendations are listed in order of least resource intensive (read lower cost) to most resource intensive. Implementing all recommendations will yield the highest performing configuration.

Recommendations

1. Pre-construct and pre-size your content databases. Once content database sizes have been predicted, it is recommended that the content databases should pre-created using a script that generates the empty databases based on other recommendations in this document. "Autogrow" should be left on to prevent issues down the road. The content databases should be monitored so that the databases don't exceed predetermined limits.
2. Place the content database file(s) on a RAID 10 logical unit(s) if at all possible. If this is not possible, then RAID 5 is an acceptable alternative.
3. With a multi-core SQL Server, the primary file group for each content database should consist of a data file for each CPU core present in the SQL Server. If possible, separate each data file to exist on separate logical units consisting of unique physical disk spindles.
4. Do not allow data files from different content databases to share LUN space. If this is not possible then stripe the data files from multiple content databases across multiple logical units. For example, instead of putting 2 data files from contentdb_01 on E:\ and 2 data files from contentdb_02 on F:\, consider spreading the 2 data files from each database onto drives E:\ and F:\.

Optimizing Search Database Performance

The SharePoint search database is a very important and often ignored component of SharePoint scalability. Consider the fact that stored in the search database is every metadata value for every document, in every library, in every site, in every site collection, in every web application that is served by the related Shared Service Provider. Also stored in the search database are the history log, search log, crawl statistics tables, links (anchor) tables and statistics tables.

Often times a Shared Services Provider serves many site collections consisting of many content databases. If each of those content databases are 100GB or more in size, then it must be considered that the sum total of the metadata along with the related anchor and crawl information can grow substantially. As an example a single search database that services queries and full crawls of 50 million content items (Microsoft recommended limit for an Index Server) can be over 500GB in size!

Use the following formula to calculate how much disk space needed for the search database:

GB of disk space required = Total_Content_Size (in GB) x File_Size_Modifier x 4

where File_Size_Modifier is a number in the following range, based on the average size of the files in your corpus:

- 1.0 if your content consists of very small files (average file size = 1KB).
- 0.12 if your content consists of moderate files (average file size = 10KB).
- 0.05 if your content consists of large files (average file size 100KB or larger)

For these reasons, a robust I/O subsystem is absolutely crucial for the search database. The following recommendations are listed in order of least resource intensive (read lower cost) to most resource intensive. Implementing all recommendations will yield the highest performing configuration.

Recommendations

1. Pre-construct and pre-size your search databases. Once the search database size has been predicted, it is recommended that the search database should be pre-created using a script that appropriately generates the empty database based on other recommendations in this document. "Autogrow" should be left on to prevent issues down the road.
2. Place the content database file(s) on RAID 10 logical unit(s) if at all possible. This is practically a requirement for large scale systems. The search database is VERY read and write intensive. RAID 10 is STRONGLY recommended.
3. With a multi-core SQL Server, the primary file group for the search database should consist of a data file for each CPU core present in the SQL Server. If possible, separate each data file to exist on separate logical units consisting of unique physical disk spindles.
4. Do not place any other database data files on any logical unit where search database data files reside. If possible, even try to ensure that the RAID 10 logical units for the search database data files do not share their physical spindles with other databases.
5. Place the search database log file on its own logical unit.

Other SharePoint Databases

The remaining SharePoint “base” databases are not nearly as read or write intensive as the tempdb, content databases, or the search database. These databases include:

- SharePoint Configuration Database
- Shared Services Provider (SSP) Database (not the search database!)
- SharePoint Admin Content Database

Finally, resources can be shared! Following these recommendations will yield a good cost vs performance balance for these lesser taxed databases:

Recommendations

1. There is no need for multiple data files per database.
2. All database data files for these databases can share a logical unit on shared physical disk spindle resources.
3. The log files for these databases can share a logical unit on a shared physical disk spindle.

Other I/O Performance Considerations

Physical Volume File Fragmentation

Physical volume file fragmentation is often ignored and can play a key role in any observed database performance degradation. Once database sizes have been determined, the actual size of the logical units being created must be considered. Unfortunately, high speed/fault tolerant storage isn't cheap so this is an important decision.

Logical units should be large enough to contain the data that will be stored plus at least 25%. This is to allow for enough space for a defragmentation routine to optimize file structures. 50% may even be a safer recommendation as there are always storage requirements that aren't thought of in the beginning.

You can manually execute defragmentation operations as database performance begins to degrade or you can implement a server class defragmentation solution on a scheduled basis.

Database Index/Statistics Fragmentation

Out of the box, SharePoint implements a timer job called SPDatabaseStatisticsJobDefinition that is supposed to maintain the efficiency of the indexes and statistics in SharePoint databases. However, this job isn't always as effective as a good quality maintenance plan. Also, large or long running operations may necessitate the need for an index rebuild upon completion. For example, after a large content migration into SharePoint it would be a good idea to rebuild the indexes.

The standard maintenance plan wizard should be sufficient to guide the creation of a usable index/statistics refresh. However there is a KB article warning here. If SQL Server SP2 is not installed, the maintenance plan may break the search database. See this KB article for more information: <http://support.microsoft.com/default.aspx/kb/930887/en-us>. Also SEE APPENDIX.

WSS Column Indexes vs Large Content Databases

In SharePoint, it is possible to create column indexes in document libraries. These indexes can dramatically improve the load speed of a library view if that view contains an indexed column in the filter criteria. Also, any programmatic searches that use the Site Data Query engine to retrieve results will also benefit from having an indexed column in the CAML "where" clause.

These indexes aren't true SQL Server table indexes, but rather pseudo-indexes that are table driven. Because of this, there is an impact to the content database as indexes are created and deleted. A large content database consisting of several hundred GB can result in a table locking situation during the manipulation of the column indexes.

While the index modification routines are building or destroying indexes, other SharePoint users are effectively locked out of any browse operations of any sites that use the same content database where the indexes are being modified. If the content database is large enough or if the I/O for the content database is poorly optimized, it is possible that an index add/delete operation may even time out in the browser. If this happens, allow time for the index operation to complete, possibly as long as 15 to 30 minutes after the timeout occurs. Once browsing of the site returns to normal the operation will likely be finished.

This issue is one of the few functional limitations of having a very large content database. The other limitations are related to the ability to backup and restore a site collection through the UI.

Implications of a Very Large Content Database

Microsoft suggests that the optimal maximum content database size should be 50GB. They also recommend that for most SharePoint implementations, the solution should not include any content databases larger than 100GB. This is commonly referred to as the “100GB content database size limitation”.

In fact, this is not a true limitation but rather a recommendation. SQL Server databases have been scaling far beyond 100GB for years now. Practically speaking, the recommendation is based primarily on two significant factors:

1. Service Level Agreement (SLA) requirements for a given organization may dictate that backup operations for the SharePoint databases must be executable in a limited amount of time. The size of the content databases will have a direct impact on how long it takes to execute that backup.
2. The storage subsystem must be robust enough to handle the disk I/O requirements of the SharePoint solution that it serves.

As long as a given organization is able to mitigate these two considerations, then the content databases can be allowed to grow. Real world implementations have seen successful SharePoint deployments that have implemented database sizes of 100GB, 150GB, 200GB, 250GB, 300GB, 350GB and 400GB.

In order to mitigate the SLA requirements, a solution might employ a high speed disk-to-disk backup solution, database mirroring, or database log shipping to a disaster recovery site.

The combination of a strong storage subsystem and the recommendations in this whitepaper will go a long way to mitigate the I/O requirements of very large content databases (over 100GB) in a given SharePoint solution.

Monitoring SharePoint I/O Performance

Microsoft provides a handy tool called Performance Monitor to allow the monitoring of statistics that will help tune an application. When monitoring the disk I/O performance for SharePoint, we have a simple indicator.

To get a quick indication of whether or not the storage subsystem is keeping up with demand, open up Performance Monitor and watch the Average Disk Queue Length counter.

- Monitor the “total” value first.
 - o If ADQ stays in the decimal range, then I/O performance is excellent
 - o If ADQ stays in the low single digits, the I/O performance is acceptable
 - o If ADQ regularly averages in the low double digits, then I/O performance is degraded and users may be occasionally affected.
 - o If ADQ regularly averages in the triple digits or higher, then I/O performance is definitely suffering and significant user impact is likely.
- If the “total” value exceeds recommended limits, then add counters for individual storage volumes to determine the offending volume. If the recommendations of this whitepaper were followed, it should be possible to determine offending database(s). Once that is determined, it may be possible to manipulate the storage subsystem or SQL Server to relieve the identified I/O stress.

Other helpful I/O counters are:

- Logical Disk: Disk Transfers/sec
- Logical Disk: Disk Read Bytes/sec & Disk Write Bytes/sec
- Logical Disk: Average Disk sec/Read (Read Latency)
- Logical Disk: Average Disk sec/Write (Write Latency)
- Logical Disk: Average Disk Byte/Read
- Logical Disk: Average Disk Byte/Write
- Logical Disk: Current Disk Queue Length
- Logical Disk: Average Disk Reads/Sec
- Logical Disk: Average Disk Write/Sec

Microsoft also provides a tool called SQLIO which can be used to determine the I/O capacity of a given SQL storage configuration. The SQLIO tool can be found here:

<http://www.microsoft.com/downloads/details.aspx?familyid=9A8B005B-84E4-4F24-8D65-CB53442D9E19&displaylang=en>

SQL Server Management for SharePoint Administrators

Move tempdb

After executing this script and restarting SQL Server, the tempdb data and log files will be initialized in the new location.

```
USE master
GO
```

```
ALTER DATABASE tempdb MODIFY FILE (NAME = tempdev, FILENAME = '[New data folder path]tempdb.mdf')
GO
```

```
ALTER DATABASE tempdb MODIFY FILE (NAME = templog, FILENAME = '[New log folder path]templog.ldf')
GO
```

(Restart SQL Server)

Add tempdb data files and set all data file sizes (in KB)

After executing this script and restarting SQL Server, the existing tempdb data file will be set to the new size and the new tempdb data files will be initialized in the specified locations and set to the specified size.

```
USE [master]
GO
```

```
ALTER DATABASE [tempdb] MODIFY FILE ( NAME = N'tempdev', SIZE = [Calculated]KB )
GO
```

```
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev_02', FILENAME = N'[Second data file path]tempdev_02.ndf' , SIZE = [Calculated]KB , FILEGROWTH = 10%)
GO
```

```
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev_03', FILENAME = N'[Third data file path]tempdev_03.ndf' , SIZE = [Calculated]KB , FILEGROWTH = 10%)
GO
```

```
.
.
.
```

```
ALTER DATABASE [tempdb] ADD FILE ( NAME = N'tempdev_0N', FILENAME = N'[Nth data file path]tempdev_0N.ndf' , SIZE = [Calculated]KB , FILEGROWTH = 10%)
GO
```

(Restart SQL Server)

Pre-Create a Database (for Content or Search databases)

The number of data files, size of the data files, and location of the data files for the database should be determined using information in this whitepaper. Once that is determined, plug those values into the script below. Once this script is executed, a properly distributed blank database will exist. Simply choose the database name of the (blank) database that this script created when configuring SharePoint through the SharePoint Central Administration or when using STSADM.

```
CREATE DATABASE [ContentDBName] ON PRIMARY
( NAME = N'ContentDBName', FILENAME = N'D:\MSSQL\SP Data\Content
DBs\contentdb_01\ContentDBName.mdf' , SIZE = 52428800KB , FILEGROWTH = 10%),
( NAME = N'ContentDBName_02', FILENAME = N'D:\MSSQL\SP Data\Content
DBs\contentdb_logs\ContentDBName_02.ndf' , SIZE = 52428800KB , FILEGROWTH = 10%)
```

```
LOG ON
```

```
( NAME = N'ContentDBName_log', FILENAME = N'D:\MSSQL\SP Data\Content
DBs\contentdb_02\ContentDBName_log.ldf' , SIZE = 1024KB , FILEGROWTH = 10%)
GO
```

```
EXEC dbo.sp_dbcmptlevel @dbname=N'ContentDBName', @new_cmptlevel=90
GO
```

```
IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))
begin
EXEC [ContentDBName].[dbo].[sp_fulltext_database] @action = 'disable'
end
GO
```

```
ALTER DATABASE [ContentDBName] SET ANSI_NULL_DEFAULT OFF
GO
ALTER DATABASE [ContentDBName] SET ANSI_NULLS OFF
GO
ALTER DATABASE [ContentDBName] SET ANSI_PADDING OFF
GO
ALTER DATABASE [ContentDBName] SET ANSI_WARNINGS OFF
GO
ALTER DATABASE [ContentDBName] SET ARITHABORT OFF
GO
ALTER DATABASE [ContentDBName] SET AUTO_CLOSE OFF
GO
ALTER DATABASE [ContentDBName] SET AUTO_CREATE_STATISTICS ON
GO
ALTER DATABASE [ContentDBName] SET AUTO_SHRINK OFF
GO
ALTER DATABASE [ContentDBName] SET AUTO_UPDATE_STATISTICS ON
GO
ALTER DATABASE [ContentDBName] SET CURSOR_CLOSE_ON_COMMIT OFF
GO
ALTER DATABASE [ContentDBName] SET CURSOR_DEFAULT GLOBAL
GO
ALTER DATABASE [ContentDBName] SET CONCAT_NULL_YIELDS_NULL OFF
GO
```

```
ALTER DATABASE [ContentDBName] SET NUMERIC_ROUNDABORT OFF
GO
ALTER DATABASE [ContentDBName] SET QUOTED_IDENTIFIER OFF
GO
ALTER DATABASE [ContentDBName] SET RECURSIVE_TRIGGERS OFF
GO
ALTER DATABASE [ContentDBName] SET AUTO_UPDATE_STATISTICS_ASYNC OFF
GO
ALTER DATABASE [ContentDBName] SET DATE_CORRELATION_OPTIMIZATION OFF
GO
ALTER DATABASE [ContentDBName] SET PARAMETERIZATION SIMPLE
GO
ALTER DATABASE [ContentDBName] SET READ_WRITE
GO
ALTER DATABASE [ContentDBName] SET RECOVERY FULL
GO
ALTER DATABASE [ContentDBName] SET MULTI_USER
GO
ALTER DATABASE [ContentDBName] SET PAGE_VERIFY CHECKSUM
GO
USE [ContentDBName]
GO
IF NOT EXISTS (SELECT name FROM sys.filegroups WHERE is_default=1 AND name =
N'PRIMARY') ALTER DATABASE [ContentDBName] MODIFY FILEGROUP [PRIMARY]
DEFAULT
GO
```

Move a database file from a source volume to a destination volume

Often times SharePoint solutions are implemented without properly planning for storage requirements resulting in “out of disk” errors. When this occurs, it may be necessary to create a new, larger volume and move the database to that new volume. The procedure for this is as follows:

1. Ensure a validated backup of the database has been recently executed.
2. Detach the database from SQL Server. You may need to check the “drop existing connections” checkbox.
3. Move the database content files and/or the log file from the source volume to the destination volume.
4. Re-attach the database to SQL Server. If using the SQL Management Studio, select the database “mdf” file from its current/new location. If any of the related database files show a message status of “Not Found”, simply locate that file in its new location.
5. Verify that the database is accessible.

Redistribute content from a single large database data file to multiple data files

Often times SharePoint solutions are implemented without properly planning for storage requirements resulting in disk I/O issues. In order to move from the out of the box configuration to a more highly scalable configuration, it may be necessary to redistribute content from a single content database data file to multiple data files. The procedure for this is as follows:

1. Ensure a validated backup of the database has been recently executed.
2. Add additional data files to the primary file group of the database using the recommendations in this whitepaper.
3. Execute this script:

```
USE [Content_DB_Name]
```

```
GO
```

```
DBCC SHRINKFILE (N'Content_DB_Primary_File_Logical_Name' , EMPTYFILE)
```

```
GO
```

This procedure will cause ALL of the content from the original (mdf) file to be equally distributed to the other (new ndf) files in the filegroup. This script may take SEVERAL HOURS to complete if the source database file is extremely large.

Summary

Disk I/O Impact on SharePoint Performance

SharePoint scalability and performance is an exercise in resource balancing. By following the recommendations of the SharePoint Capacity Planning tool provided by Microsoft, the topology necessary to support specified user requirements are clear. However, this tool explicitly states that the storage subsystem is not considered in the results.

So while the server infrastructure may be in place for a properly architected farm, it is easy for the storage subsystem to become overwhelmed by that those servers resulting in unexpected poor performance

A strong and expandable storage subsystem is the key to current and future performance of large scale Microsoft SharePoint implementations. The ability to spread content across as many disk spindles as possible is paramount.

The Bottom Line

The scalability of a software product is determined by that software's ability to consume additional hardware and, in turn, produce improved performance relative to the hardware increase.

The focal point of this whitepaper is that SharePoint is highly scalable, largely due to its ability to use additional storage capacity and servers and, in return, produce the ability to store more content and serve more users relative to the resources added.

Using these techniques, a SharePoint farm has been scaled and tested to at least 50 million content items which is the recommended guideline that a farm Index Server should support.

Even further, with the implementation FAST technology to handle Index/Search services, it is likely that a single SharePoint farm that implements these techniques will scale far beyond 50 million items.

Just remember that when it comes to any SharePoint database storage solution, try not to spare expense. "E Aho Laula"...Wider is better.

Appendix A: References

Capacity Planning for tempdb

<http://msdn2.microsoft.com/en-us/library/ms345368.aspx>

Estimate Performance and Capacity Requirements for Search Environments

<http://technet.microsoft.com/en-us/library/cc262574.aspx>

Performance Recommendations for Storage Planning and Monitoring

<http://go.microsoft.com/fwlink/?LinkID=105623&clcid=0x409>

“Office SharePoint Server 2007 – Administrator’s Companion”

Bill English with the Microsoft SharePoint Community Experts, Microsoft Press

How to Defragment Windows SharePoint Services 3.0 and MOSS Databases

<http://support.microsoft.com/default.aspx/kb/943345/>

Physical Storage Recommendations (Office SharePoint Server)

<http://technet.microsoft.com/en-us/library/cc298801.aspx>