Symantec Consulting Services

SQL Configuration Best Practice

Status:	**First Draft**
Version:	**0.1**
Altiris Version:	**SMP 7**
Date:	**8th June 2010**
Author:	**Kath Clarges**

Copyright

SYMANTEC Software Corporation
350 Ellis Street
Mountain View, CA 94043
USA
Phone 650-527-8000
Fax 650-527-2908
www.symantec.com

Third party Legal Notice

Disclaimer

Revision History

Version	Date	Author	Changes
0.1	8th June 2010	Kath Clarges	Initial draft

Reviewer information

Version	Date	Reviewer	Position
0.1			

Referenced KB Articles

AKB	Title
40488	Creating a maintenance plan in SQL Server 2005 or 2008 to optimize database performance
49632	Optimizing SQL Server 2005 on Windows Server 2003 32-bit
28122	SQL Tuning - Best Practices for Altiris Notification Servers
52371	Notes on Configuring SQL Server for use with SMP 7
39381	What SQL Rights are needed for the Application Identity
51915	How to set the purge maintenance for the CMDB
45803	Altiris 7 – Planning and Implementation Guide
46349	SMP Platform Support matrix
1859	Items to help improve Console and Server performance
Cc966414	Physical database storage design

Contents

SQL Configuration

On Box – Off Box

A local database server configuration means the database server is located on the same machine as the Notification Server. This is the recommended configuration for small environments since there will be minimal contention of resources between the Notification Server and the database server.

A remote database server configuration means the database server is located on a different machine from the Notification Server. This is recommended for larger environments since the workload of the database server can be offloaded from the machine handling the Altiris Agent requests. However, the database server and Notification Server must have a very high speed network connection between them (1GB Ethernet is recommended).

Benefits of a remote database include:-

1) No contention for resources with the Symantec Management Platform (SMP) and associated applications
2) Can leverage performance improvements of x 64 SQL
3) Enables the use of VMware for the SMP (small to medium size environments)

SQL Versions

Supported versions of SQL server:

The SMP is supported on some versions (but **not** all versions) of:

- SQL Server 2005
- SQL Server 2005 Express
- SQL Server 2008
- SQL Server 2008 Express

This information is based on the March 2, 2010.

SQL on Server Operating System	7.0 SP4
Microsoft SQL Server 2005 SP2	✓
Microsoft SQL Server 2005 SP3	✓
Microsoft SQL Server 2005 Express SP2	✓
Microsoft SQL Server 2005 Express SP3	✓
Microsoft SQL Server 2008	✓
Microsoft SQL Server 2008 SP1	✓
Microsoft SQL Server 2008 Express	✓
Microsoft SQL Server 2008 Express SP1	✓

NOTE: The SMP installation is tested with Microsoft SQL Standard and Enterprise versions.

- Symantec Management Platform and Altiris Solutions Support Matrix (46349)

SQL Permissions

SQL Server, Databases and Authentication

The SMP 7 stores all of its information in one database on one SQL server, and accesses that database through one account. The default database name is Altiris_CMDB. The SMP 7 can authenticate to the SQL Server using either Windows authentication, or SQL Server authentication.
If the SMP 7 uses Windows Authentication to access the SQL server, then it can use the Altiris Application Identity or it can use a different account. The Application Identity is the Windows account that Altiris uses on its own server.
The name of the SQL server, the name of the database and the account credentials can be specified during installation, using Symantec Installation Manager (SIM).
These can also be modified using the Symantec Management Console (SMC).

Warning: changing these settings in the SMC does NOT move or copy the database, and may cause the loss of access to the Altiris database.

SQL Server Permissions

The SMP 7 uses only one Schema within its database.
Within the Altiris database, the SMP 7 needs to be able to:

- create, alter, and drop tables
- create, alter, and drop views
- create, alter, and drop stored procedures
- execute stored procedures
- view, insert and drop records in these tables

If the Altiris database is not created before installing the SMP 7, then the SIM will also need to create the Altiris database.

Additionally, outside of the Altiris database, SMP 7 needs to:

- view information from the msdb database
- execute some system stored procedures

Minimum SQL Security Requirements

We recommend that the user account used when installing Notification Server is the DBO for the Notification Database.

The Notification Server user account will need “dbcreator” permissions in order for the Notification Server installation process to create the Notification Database. If the SQL administrator creates the Notification Database prior to Notification Server installation, then this permission is not necessary.

Just point the installation process to the already present Notification Database.

On the Notification Database itself, the user will need the following permissions:

- Public - General access to the database.
- db_ddladmin - Adds, modifies, or drops objects in the database.
- db_datareader - Sees all data from all user tables in the database.
- db_datawriter - Adds, changes, or deletes data from all user tables in the database.

PLEASE NOTE: The above information is the Minimum requirements. The Best Practice is to make the Application Identity user the DBO of the database.

To verify that the Application Identity is DBO on the database, run the following SQL command as the Application Identity:

- use Altiris_CMDB --or replace with the name of the Notification Server database
- select user

If that returns anything other than "dbo" then the user is not the DBO of the database.

Disk Configuration

Example Disk Configuration

Logical Drive	Purpose	Disk Configuration
C:\	Operating System	Raid 1
D:\	SQL Binaries & Paging File	Raid 1
E:\	SQL Data	Raid 5
F:\	SQL Log Files	Raid 10
G:\	Tempdb	Raid 10
H:\	Tempdb Log files	Raid 10

File Placement

Where possible, place the database files, transaction log files, and the tempdb files each on separate, physical I/O devices (disks/arrays/LUNs). This improves performance by allowing multiple physical devices to concurrently service reads and writes to these files.

Hardware vs. Software RAID

There are two ways RAID is implemented: through hardware, and through software. The hardware implementation refers to the traditional method implemented with a collection of

disk drives working together for fault tolerance through striping or mirroring of data, as described previously in this paper. The software version refers to RAID implementation provided by the operating system.

Software RAID is usually the cheaper of the two solutions. However, there are a few drawbacks to using software RAID. The most significant drawback is that, because the RAID engine runs in kernel mode, it shares CPU time quanta with other kernel mode components and overlying applications. In contrast, the hardware RAID solution does not face this problem because the RAID firmware executes on a separate dedicated chip. This also allows for asymmetric multiprocessing between the system processor and the RAID controller.

RAID

RAID stands for Redundant Array of Inexpensive (or Independent) Disks. It is a collection of disk drives working together to optimize fault tolerance and performance. There are various RAID levels, but only the RAID levels significant to SQL Server are described here.

RAID0 (simple striping): Simplest configuration of disks that stripe the data. RAID0 does not provide any redundancy or fault tolerance. Data striping refers to sequentially writing data in a round-robin style up to a certain stripe size, a multiple of a disk sector (usually 512 bytes). Data striping yields good performance because multiple disks are concurrently servicing the I/O requests. The positive points for RAID0 are the cost, performance, and storage efficiency. The negative impact of no redundancy can outweigh its positive points.

RAID1 (simple mirroring): This configuration creates an exact copy or mirror of all the data on two or more disks. This RAID level gives good redundancy and fault tolerance, but poor storage efficiency. To fully take advantage RAID1 redundancy, it is recommended to use independent disk controllers (referred to as duplexing or splitting). Duplexing or splitting removes single-point failures and allows multiple redundant paths.

RAID5 (striping with parity): RAID5 uses block-level striping where parity is distributed among the disks. RAID5 is the most popular RAID level used in the industry. This RAID level gives fault tolerance and storage efficiency. However, RAID5 gives a larger negative impact for all write operations, especially sequential writes.

RAID10 (stripe of mirrors): RAID10 is essentially many sets of RAID1 or mirrored drives in a RAID0 configuration. This configuration combines the best attributes of striping and mirroring: high performance and good fault tolerance. For these reasons, we recommend using this RAID level. However, the high performance and reliability level is the trade-off for storage capacity.

Note that in all levels of RAID configuration, the storage efficiency is always limited to a multiple of the smallest drive size.

Summarized I/O Activity of RAID Levels and Their Recommended Use

RAID Levels	RAID0	RAID1	RAID5	RAID10
Reliability	Lowest.	Very good.	Good.	Excellent.

	Lack of fault tolerance results in data loss.	Even better with duplexing.	Can tolerate single machine fault.	
Storage Efficiency	100%	50%	>50%, <100% (#drives - 1/#drives)	50%
Random Read	Excellent	Fair Worst of the RAID levels but better than a single drive.	Excellent.	Excellent.
Random Write	Excellent.	Fair. Worse than a single drive but better than some RAID levels.	Fair. Generally better with larger stripe sizes.	Very good.
Sequential Read	Excellent.	Fair. Comparable to a single drive.	Very good. Generally, better with smaller stripe sizes	Excellent.
Sequential Write	Excellent.	Good. Better than other RAID levels.	Fair.	Very good.
Cost	Lowest.	Moderate. Relatively high cost due to redundant drives; however, no expensive controller required	Moderate.	High.
Recommended use	Good for non-critical data or stagnantly updated data that gets backed up regularly or any data requiring fast write performance at very low cost. Great for testing.	Good for data that requires high fault tolerance at relatively low hardware cost (redundancy using parity requires more expensive hardware). Best for log files.	Very good for Read only data.	Data requiring high performance for both read and write and excellent reliability while trading off storage efficiency and cost.

SQL Configuration

The following are optimizations recommended by Microsoft for increased performance of SQL in general.

Optimizing Transaction Log Performance

General recommendations for creating transaction log files include:

a. Create the transaction log on a physically separate disk or RAID (redundant array of independent disks) device. The transaction log file is written serially. Therefore, using a separate, dedicated disk allows the disk heads to stay in place for the next write operation.

b. Set the original size of the transaction log file to a reasonable size to prevent the file from automatically expanding as more transaction log space is needed. As the transaction log expands, a new virtual log file is created. Write operations to the transaction log wait while the transaction log is expanded. If the transaction log expands too frequently, performance can be affected.
c. Set the file growth increment percentage to a reasonable size to prevent the file from growing by too small a value. If the file growth is too small compared to the number of log records being written to the transaction log, then the transaction log may need to expand constantly, affecting performance.

Optimizing tempdb Performance

General recommendations for the physical placement and database options set for the tempdb database include:

a. Allow the tempdb database to automatically expand as needed. This ensures that queries that generate larger than expected intermediate result sets stored in the tempdb database are not terminated before execution is complete.
b. Set the original size of the tempdb database files to a reasonable size to avoid the files from automatically expanding as more space is needed. If the tempdb database expands too frequently, performance can be affected.
c. Set the file growth increment percentage to a reasonable size to avoid the tempdb database files from growing by too small a value. If the file growth is too small compared to the amount of data being written to the tempdb database, then tempdb may need to expand constantly, thereby affecting performance.
d. Place the tempdb database on a fast I/O subsystem to ensure good performance. Stripe the tempdb database across multiple disks for better performance. Use file groups to place the tempdb database on disks different from those used by user databases.

Memory Configuration

In MS SQL Server 2005 (onwards) the memory allocation is set to DYNAMIC, that means the available memory on the server will be allocated to SQL Server and released whenever a windows service requests for such a memory.

It is recommended to leave the default dynamic allocation where the SQL server is remote and dedicated.

Static memory configuration may improve performance where:-

1) SQL is installed on a 32 bit OS with AWE enabled
2) Database is sharing resources with other SQL databases (named instances)

See AKB 3977 - Altiris© NS Performance and Scalability—SQL and Large Memory Configuration, for specific configurations.

Disable Auto Shrink

It is NOT recommended to shrink the data file, as this will slow SQL down by causing fragmentation when the data file is resized during the next automatic growth.

- Auto-shrink runs periodically in the background, consuming CPU and I/O cycles which can cause unexpected performance hits.
- Continually shrinking and regrowing the data files can lead to physical fragmentation of the database file, which hurts both sequential transfers and random accesses.

Auto-shrink is the worst offender as it starts every 30 minutes in the background and tries to shrink databases where the auto-shrink database option is set to true.
It is a somewhat unpredictable process in that it only shrinks databases with more than 25 percent free space. Auto-shrink uses lots of resources and causes performance-dropping fragmentation and so is not a good plan under any circumstances. You should always switch off auto-shrink with:

- `ALTER DATABASE MyDatabase SET AUTO_SHRINK OFF;`

Maintenance Plan

Additionally, the Altiris database needs to be appropriate maintained. This includes (but is not limited to):-

- database backup and restore
- table re-indexing
- file defragmentation

Maintaining the database can be accomplished using the built-in capabilities of SQL Server.
In some situations some 3rd party applications may be helpful.
Please note, that the SMP does not include any utilities to accomplish these maintenance tasks.

A SQL maintenance plan for rebuilding indexes and setting the index free space percentage to 10% within the Symantec/Altiris databases should be scheduled to run at least monthly – preferably weekly. This maintenance plan should also be configured to update column statistics (index statistics are updated during the index rebuild process). It is recommended that this

plan be scheduled to run at a time when database utilization by the Symantec applications is at its lowest. This could be on a Sunday during the day or a during a scheduled maintenance cycle, etc.

To build the SQL Maintenance plan for SQL 2005/2008:

1. Make sure that the SQL Server Agent service is running.
2. Open SQL Server Management Studio.
3. Expand the Management folder.
4. Right-click on Maintenance Plans and select Maintenance Plan Wizard.
5. When the SQL Server Maintenance Plan Wizard info page opens up; click Next.
6. Give the maintenance plan a name such as "Rebuild Indexes for Altiris Databases".
7. Leave the default option set to Single schedule for the entire plan…
8. Click the Change button to put in the schedule for this plan.
9. Enter in the chosen weekly time.
10. Click OK.
11. Click Next.
12. Check the options to Rebuild Indexes and Update Statistics.
13. Click Next.
14. Make sure that "Rebuild Index" task is at the top.
15. Click Next.
16. Click the Database drop-down.
17. In the These databases section, select all the databases you are trying to optimize, such as Altiris, Altiris_Incidents, eXpress, AeXRSdatabase, Symantec_CMDB and Symantec_CMDB_IntelAMT.
18. Click OK.
19. Select Change free space per page percentage to and set its value to 10% (20% if only rebuilding indexes monthly).
20. "Sort results in tempdb" should generally not be used; however, if SQL memory resources are low, then this will help, but it does cause rebuilding to take a lot longer.
21. Make sure that Keep index online while reindexing is unchecked. Altiris databases uses ntext fields which prevent clustered indexes from being rebuilt while online for those tables that have an ntext column.
22. Click Next.
23. Chose the same databases as before.
24. In the Update section, select Column Statistics Only.
25. In the Scan Type section, select Full scan
26. Click Next.
27. Check the option to Write a report to a text file and allow it to write to the default location.
28. Click Next.
29. Click Finish.

Once the wizard is finished with creation, you can click Close. To execute the maintenance plan, right-click it in the left hand Object Browser pane and click Execute.

Purge Maintenance

How to purge the Configuration Management Database of unwanted/outdated data

1. In the Symantec Management Console menu, click Settings > Notification Server > Purging Maintenance.
2. On the Purging Maintenance page, specify the Configuration Management Database purging settings that you want.
3. The Purging Maintenance tab lets you purge report and computer data.
4. The Resource Event Data Purge Settings tab lets you configure the purging of resource event data.
5. To override the purging schedule and purge the CMDB immediately, click Purge Now.
6. Click Apply.

Disk fragmentation

Make certain that the database files are not physically fragmented. Periodically check the fragmentation level on the volume where the database files are hosted and defragment when needed.

Named Instances

If a SQL Cluster is proposed for a shared database infrastructure, it is very important to properly evaluate the size of the cluster, number of nodes and the availability options. It is also critical that the individual databases for each Notification Server exist on a separate instance. This is recommended to avoid Tempdb contention.
In addition, this approach offers some flexibility around min and max memory allocation where additional resources can be configured for use by the hardest working or most critical database.