Best Practices

Optimizing your Microsoft® SQL Server Environment

Sean Barber, DBA

UNIVERSAL
SOFTWARE SOLUTIONS, INC.

Optimizing your Microsoft® SQL Server Environment

Partitioning

If the Windows Partition for the SQL Database was created within Windows using disk management, it is not aligned on a 1024 byte boundary and must be corrected. The correction usually involves destroying and recreating the partition using diskpart.exe and then it is formatted with the correct block size for SQL Server Databases. Hardware RAID and Hardware SAN solutions expect to be aligned on 1024 byte boundaries to avoid the hardware from making two I/O writes for each one I/O write that Windows requests. If you are not aligned properly, the double writes will cause a 15 to 20% speed penalty in disk I/O and it is noticeable. Nothing you can see or do from within Windows will reveal this penalty; you will need to look at the SAN or RAID solution's tools to see this misalignment and the resulting performance penalty. This is because Windows is "told" by the RAID controller or SAN that it is working with a "Basic Disk" or a "Simple Volume".

SQL Server Log Files

If the SQL Log Files are located on the same Windows disk as the SQL Database, they should be moved to a separate disk.

“TEMPDB” File

There should be one tempdb file per processor core in your tempdb file group. If one file is the default, this should be on a separate disk.

“TEMPDB” Log

Tempdb log should be on a different partition as your tempdb database file.

MDF and LDF Files

MDF files and LDF files should be on different partitions. Not the same partition as TempDB or Paging file or Operating system. Stop SQL Server and defragment the MDF and LDF files for all of your databases. Windows Defrag cannot and will not defrag a locked file. This requires a Microsoft command line tool.

Paging Files

Paging files should be set to be at least 1.5 times the amount of RAM the server has installed. If the server is 32 bit OS and the maximum paging file size is limited to 4095 but the recommended amount of paging space is higher, have you created multiple paging files on the same partition in order to meet the recommended paging file size? There is a Microsoft prescribed procedure for creating more than one 4095MB paging file per partition.

Paging files should not be on the same volume as your operating system or on the same volume as your database files or on the same volume as your log files.

Have you used a boot time defragmentation algorithm to defrag the paging files and all of the "system" files that show up as green areas in Windows Defrag? It requires a Microsoft command line tool. A paging file that is in more than one file fragment is especially ironic, considering the job it is supposed to be doing.

SQL Tips

Install and run the Microsoft tool named "SQL 2005 Best Practices Analyzer" (this, or the current version). Have you cured all performance and security issues presented to you?

Have you stopped SQL Services and did a Windows Defragmentation after completing the above 2 items? Then use the SQL Tools to do a file shrink and reorganize pages and release space task.

Have you granted the user rights for Instantaneous File Growth in Group Policies, and have you changed the growth from percentage to MB for each database?

Is your network connection from the clients to the SQL server at Gigabit speed the entire pathway? Are the Component Server and the SQL Server plugged into the same physical switch?

Set SQL compatibility level to SQL 2005:

http://msdn.microsoft.com/en-us/library/bb510680.aspx?ppud=4

Microsoft Updates

Apply each and every possible Microsoft Update to the SQL Server and the Component Server(s). If you have MS patches that you don't want to apply, ask yourself why. Are they irrelevant? If so, why is the service or app enabled so Windows offers the patch? Remove and/or disable the vulnerable service or app, or install the patch. There is no acceptable middle ground where MS offers any patch that you don't elect to install.

SQL Server Memory

Check SQL server memory usage pattern. In Performance Manager (perfmon), you can find explicit counters given to check the SQL Server memory usage. SQL Server: Memory Manager: Total Server Memory counter can tell you the amount of dynamic memory the server is presently consuming. You can also look into:

- SQL Server: Buffer Manager: Buffer cache hit ratio
- SQL Server: Buffer Manager: Page Life Expectancy
- Process: Working Set

SQL Server: Buffer Manager:

- **Buffer cache hit ratio** should be more than 90% for a SQL Server.
- **Page Life Expectancy** should be above 350. If it is lower or almost equal to 350-400 mark and you are expecting much growth of the database, then be ready to buy more memory modules for your server.
- **Process**: Working Set tells you how much memory SQL Server is consuming. If it is lower than what you have configured by MIN SERVER MEMORY, then you have allocated more memory than the server needs. If it exceeds MAX SERVER MEMORY, then look to buy additional memory.

If you are using SQL Server 2005/2008, then you can use Dynamic Management Views (DMV), which give more insight to the problem. I will talk more about DMVs in future articles.

Check the average free space and handling of memory in SQL Server.

Determining memory for a SQL Server box is not only checking how much memory the SQL Server requires; but, how much memory does the box require. A couple of perfmon counters that come to mind are:

- **Memory: Pages/sec** determines how many pages are read/written to the disk because of hard page faults. A hard page fault pertains to data fetching from hard disk, which adds latency to the throughput of the data. So, this counter should be as low as possible.
- **Memory: Available Bytes** simply indicates how many bytes are available. If the counter is high, then your server is working fine, but check the counter when the server is undergoing the normal load that it is expected to handle.

Maintenance of index in any database is critical (see introduction). Any database that grows rapidly needs index maintenance.

Growth of your HDMS Database

Database growth can be of three types:

1. Linear growth (grows at a constant amount over time).
2. Compound growth (grows at a constant rate for a specific period of time).
3. Geometric growth (grows periodic by some increment, which is in turn growing at a constant rate).

Below are the formulae for different type of growths:

- Linear: Current disk space + (growth * No of periods)
- Compound: Current disk space * (1 + Growth %) ^ No of periods
- Geometric: Current growth + [initial increment * {1-Incrementgrowth rate^ (No of period+1)}]/ (1 – growth rate)

To illustrate the fact, let us take one example. Say for instance, you have a database and it is of 100 GB. If the database grows at 10% per year, then after 2 years, it will be:

100 + (10 * 2) = 120 GB (linear growth)

If the database grows at 5% rate/month for 2 years, then it will be:

100 * (1 + 0.05) ^ 24 = 322.5 GB (compound growth)

Take another case. Say you already have a database of 100GB. Now the growth rate is so that it starts at 10 GB/month and the increment itself increments at 5%/month, then in 24 months, it will be 100 + {(10 * (1- (1.05 ^ (24 + 1))) / (1-1.05)} = 100 + 477 = 577 GB (geometric growth).

Above there is formulae to identify and calculate the growth of your data file. Estimate your memory growth according to the same formulas. Additionally, if your users are growing at a 10% rate, then the connections to the server will also increase in that way. Be prepared to handle this and be sure your server is also prepared.

Index Maintenance

- Do you have an index maintenance plan?
- Index maintenance should be performed on a weekly schedule. Rebuild & Reorganize.