### Description

This article consolidates best practices for SQL Server 2005 or later server implementation. It also presents a simplistic process that can be used to tune and troubleshoot performance.

The focus is on Operating System and SQL Server settings that should be taken into consideration. The information contained in this article is geared toward servers that are dedicated to SQL. Consider each setting's potential effect on the Operating System and any other applications and services that may be hosted along-side your SQL instance before making changes. Make one change at a time and test its impact before making other changes to help keep your testing simple, and to actually be able to draw accurate conclusions about the impact of the change.

**Implementation references**

* **Physical Database Storage Design:**<http://technet.microsoft.com/en-us/library/cc966414.aspx>
  + "RAID10 yields excellent read-write performance."
  + "RAID5 can havemuch lower write performance than any other configurationbecause it requires extra reading and writing activities for the parity blocks in addition to reading and writing the data."
  + "For optimized I/O parallelism, use [a] 64 KB or 256 KB stripe size", when creating a RAID.
  + If necessary to combat excessive Page Latch Waits on tempdb-based activity, the tempdb data file can be split up along the following guideline: (number of physical CPU cores allocated for use by SQL)/4, up to a maximum of 8 data files within its filegroup. It should be noted that splitting tempdb into multiple files on the same spindle can be beneficial, which could seem counterintuitive. Where possible, doing this across multiple spindles would likely further improve the performance. The default recommendation is to split tempdb into two files, e.g. tempdb.mdf and tempdb2.ndf. Further splitting should only be done when indicated by excessive Page Latch Waits on tempdb.
* **Disk Partition Alignment Best Practices for SQL Server:**[**http://msdn.microsoft.com/en-us/library/dd758814(v=sql.100).aspx**](http://technet.microsoft.com/en-us/library/dd758814(SQL.100).aspx)
  + Correctly implement disk partition alignment on each RAID:
    - "Disk alignment produced significant improvement compared to nonaligned disks. The measurements document enhancements in excess of 30% for disk latency and duration."
    - "The performance of six aligned disks was comparable to or better than eight nonaligned disks."
    - Check your RAID's partition alignment using the following at a command line: "wmic partition get BlockSize, StartingOffset, Name, Index"
* **SQL Server Best Practices:** <http://technet.microsoft.com/en-us/library/cc966412.aspx>
  + "When formatting the partition that will be used for SQL Server data files, it is recommended that you use a 64KB allocation unit size for data, logs, and tempdb."
* **File Placement:** Where possible, place the database files, transaction log files, and the tempdb files each on separate, physical I/O devices (spindles/arrays/LUNs). This improves performance by allowing multiple physical devices to concurrently service reads and writes to these files.

**Performance Troubleshooting process**

Use the steps and information below to identify and address specific performance issues.

**Before you begin**:

* **IMPORTANT**: Ensure that you are keeping the database files' physical and logical fragmentation to a minimum by:
  + Periodically analyzing the database drive file system fragmentation level using a tool such as Microsoft Disk Defragmenter.
  + View the Defragmenter report and defragment the drive if the database files are fragmented.
  + Rebuilding the database indexes periodically using the information in [HOWTO8589](http://www.symantec.com/docs/HOWTO8589): Creating a maintenance plan in SQL Server 2005 or later to optimize database performance. Either implement the plan shown in that article, or one that incorporates the subplans and settings outlined in this article, and verify that this plan executes successfully and often enough for your environment.
* It is generally recommended to update to the most recent SQL Server Service Pack that is currently supported by your Symantec application. Often, these Service Packs will include performance fixes.
* The above is generally also true for the Server Operating System and you should update to the most recent OS Service Pack supported.

**Steps**:

* Use Performance Monitor while the performance problem is occurring to identify a bottleneck
  + See [HOWTO9722](http://www.symantec.com/docs/HOWTO9722): Common Performance Monitor counter thresholds. If you identify a bottleneck on the SQL Server itself, continue, otherwise, look for bottlenecks on other servers or clients in the system.
* After you have identified the bottleneck, check for the bottleneck category, Memory, CPU, Disk, within the "Categories" section below to see if a related setting can be configured more optimally.
* Run the Database Health scan tool in SSE Tools and have the results reviewed by Symantec Support. See [HOWTO60787](http://symantec.com/docs/howto60787)
* If an appropriate change is found, make the change and then observe whether performance has improved.
* Once SQL Server and the OS have been optimally configured for the server's hardware, you are generally left with the following options:
  + Reduce the amount of load on SQL, where appropriate, by decreasing the frequency of:
    - Client policy updates
    - Client inventory reporting
    - Client task execution
    - Client task polling
    - Scheduled tasks
  + Identify long-running or blocking SQL statements and evaluate whether they can be optimized:
    - See [HOWTO9817](http://www.symantec.com/docs/HOWTO9817): How can I view current SQL process activity and identify blocks?
  + Upgrade the hardware associated with the bottleneck, e.g. adding more RAM, faster CPUs, more or faster Disks, etc.

**Cheat Sheet - Categories**

**Memory - SQL Server**

* Enable the **Optimize for Ad hoc Workloads**setting in the SQL Server instance settings, Advanced page. This allows more memory to be used for the buffer pool cache, rather than for procedure caching of single-use batches.
* Set a value in the **Maximum server memory** option of the 'Server memory options' page, that leaves enough memory for the Operating System itself and tools such as SQL Management Studio to run effectively, e.g. around 4-6 GB less than the amount of RAM installed.
* If AWE support is enabled in SQL, ensure that the account used for the SQL Server service has the **Lock Pages in Memory** right.2
  + Both AWE and **Lock Pages in Memory** can benefit 64-bit SQL Servers as well as 32-bit.3
  + **Lock Pages in Memory** is found in the "User Rights Assignment" of the Local Security Policy.

**Settings - SQL Server Instance**

* If your SQL server has more than 4 cores, lowering the **Max Degree of Parallelism** (rather than leaving at the default of 0 (unlimited)) will provide better performance. To set:
  + In Server Properties, on the Advanced page, set the **Max Degree of Parallelism** to half the number of physical processor cores on your system, up to a maximum value of 8.4
    - For a typical SMP server, the suggested value is 2.
    - For a reporting server, or heavily used Asset server, the suggested value is 4.
    - For servers that are queried by IT Analytics, the suggested value is 4.
    - Monitor CXPACKET waits on the SQL instance. The goal is that they be less than 5% of relevant waits. 6 If CXPACKET waits remain higher than 5%, consider lowering the **Max Degree of Parallelism** by half again.
* Set the **Cost Threshold for Parallelism** to 20
* Set **Arithmetic Abort Enabled**to true in the Database Properties on the Options page, Miscellaneous section.

**Disk - SQL Server System Databases**

* If tempdb is on a single spindle, or I/O channel:
  + Ensure that tempdb has two data files:
  + Right-click on tempdb and click Properties
  + Go to the Files page and click the Add button
  + Set the **Logical Name** of the new file to tempdev2 and the **File Name** to tempdb2.ndf
  + Set the **Initial Size** of both Data files to 100MB and the **Autogrowth**to grow by 10MB

**Disk - SQL Server User Databases**

* Ensure that the account used for the SQL Server Service has the **Perform volume maintenance tasks** right. Without this, data files cannot be expanded using **Instant file initialization** which improves performance of data file growth.  **Perform volume maintenance tasks** is found in the "User Rights Assignment" of the Local Security Policy.
* Plan for the expected maximum sizes of your your data and transaction log files and set their initial sizes to that value plus 10-20%. One way to estimate this is to monitor the database files' growth in a pre-production environment and extrapolate.
  + In SQL Management Studio, right-click your database and click Properties.
  + On the Files page, in the Database Files table, set the Initial Size fields to the values you have estimated.
* Set a healthy Autogrowth increment on your data and transaction log files as a contingency in the event that the data files need to grow during a non-maintenance period. You should not rely on Autogrowth to manage your database file sizes, but should monitor the files and re-size them according to your projected needs during maintenance periods.
  + In SQL Management Studio, right-click your database and click Properties
  + On the Files page, in the Database Files table, click the ellipsis button in the **Autogrowth** column of a file row
  + Enable **Autogrowth**
  + Set the **File Growth** to a value such as 500MB or 10%
  + Set the **Maximum File Size** to a value that will allow for some Autogrowth, but also ensures that the files do not grow out of control.
  + Repeat for each database file

**Database Settings - User Databases**

* Reporting server, or parent server in a hierarchy:  
  **NOTE**: These settings are enabled by default on 7.5 SP1 and above systems but may need to be checked on systems that were upgraded.  
  + Enable **Read Committed Snapshot** mode (replace database name with appropriate name) using:  
      
    alter database Symantec\_CMDB  
    set read\_committed\_snapshot on  
    with rollback after 30;
  + Enable the **Auto Update Statistics Asynchronously** setting in the Database Properties, Options page.
* Disable **Autoshrink**. Shrinking the database automatically is not recommended for multiple reasons:
  + **Autoshrink**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.5

**References**

1. Configuring Hosts and Host Instances  
   <http://msdn.microsoft.com/en-us/library/ee308915(BTS.10).aspx>
2. How to: Enable the Lock Pages in Memory Option (Windows)  
   <http://technet.microsoft.com/en-us/library/ms190730(SQL.90).aspx>
3. Using AWE, locked pages in memory, on 64 bit  
   <http://blogs.msdn.com/slavao/archive/2005/04/29/413425.aspx>
4. General guidelines to use to configure the MAXDOP option  
   <http://support.microsoft.com/kb/329204>
5. Considerations for the "autogrow" and "autoshrink" settings in SQL Server  
   <http://support.microsoft.com/kb/315512>
6. CXPACKET Wait Stats & 'max degree of parallelism' Option: Introduction to Using Wait Stats to Identify & Remediate Query Parallelism Bottlenecks  
   <http://blogs.msdn.com/b/jimmymay/archive/2008/11/28/case-study-part-1-cxpacket-wait-stats-max-degree-of-parallelism-option-introduction-to-using-wait-stats-to-identify-remediate-query-parallelism-bottlenecks.aspx>