

# Advanced Configuration Options in Microsoft SQL Server 2005

The ability to effectively manage application and database performance is critical for IT organizations. The Microsoft® SQL Server™ 2005 database platform includes an array of configuration settings that can help improve resource utilization and optimize system performance. This article discusses some of the advanced configuration options in SQL Server 2005 and their associated best practices and limitations.

BY ANANDA SANKARAN AND SHABANA M.

*Related Categories:*

*Characterization*

*Database*

*Microsoft SQL Server 2005*

*Microsoft Windows*

*Performance*

*Visit [www.dell.com/powersolutions](http://www.dell.com/powersolutions)  
for the complete category index.*

**M**icrosoft SQL Server 2005 provides advanced configuration options that can significantly affect system performance. These instance-wide options—which can be set through SQL Server Management Studio or the `sp_configure` stored procedure—control various aspects of database operation and performance. With the appropriate settings for these options, enterprise IT organizations can help ensure efficient resource utilization and high performance for their SQL Server 2005 environments.

### Setting the recovery interval parameter

When an instance of Microsoft SQL Server starts, all transactions at the time the instance was previously stopped need to be rolled back or forward based on their commit status. The recovery interval option specifies the maximum

time (in minutes) for performing recovery operations and controls the frequency of checkpoints issued in each database. During each checkpoint, SQL Server flushes all log information and modified data pages from the buffer cache to the disk. A checkpoint is issued whenever the number of data modifications made after the last checkpoint reaches a certain limit estimated by SQL Server (based on the ability of SQL Server to perform recovery operations within the specified recovery interval) or whenever the log becomes 70 percent full. The default value of this parameter is 0 (indicating automatic configuration), which usually means a recovery time of less than one minute and a checkpoint approximately every minute for active databases. This parameter does not take into account the time required for undoing long-running transactions that span several checkpoints.

<b>Server</b>	Dell™ PowerEdge™ 2800 server with two Intel® Xeon® processors at 3.6 GHz, 2 MB L2 cache, and 8 GB of RAM
<b>Storage</b>	Dell/EMC CX300 Fibre Channel storage system with one disk array enclosure
<b>RAID configuration</b>	Two 10-spindle RAID-10 volumes for database files and one 4-spindle RAID-10 volume for the log, with 15,000 rpm Fibre Channel disks
<b>Storage interconnect</b>	Brocade SilkWorm 4100 Fibre Channel switch and QLogic QLE2360 Fibre Channel host bus adapter
<b>Network interconnect</b>	Dell PowerConnect™ 5324 switch
<b>Client servers</b>	Eight Dell PowerEdge 750 servers
<b>OS</b>	Microsoft Windows Server 2003 Enterprise x64 Edition with Service Pack 1
<b>Application</b>	SQL Server 2005 Enterprise Edition (x64)

Figure 1. Test server configuration

In December 2005, a team of Dell engineers conducted stress tests to measure the impact of changing the recovery interval parameter on database performance. The server configuration used for the tests is described in Figure 1. A generic transaction-based workload from a set of client servers was applied on a SQL Server 2005 deployment at recovery interval values of 0, 5, 10, and 15. OS and database performance counters were collected during a steady period when the load was constant and the same for each test. Figures 2 and 3 show the variation of the significant counter values. In Figure 2, the checkpoint pages per second metric reflects the number of checkpoints issued by the server, and disk transfers per second reflects the total number of reads and writes issued to the logical disks. As the recovery interval was increased from 0 to 15, the number of checkpoints issued decreased, and hence the

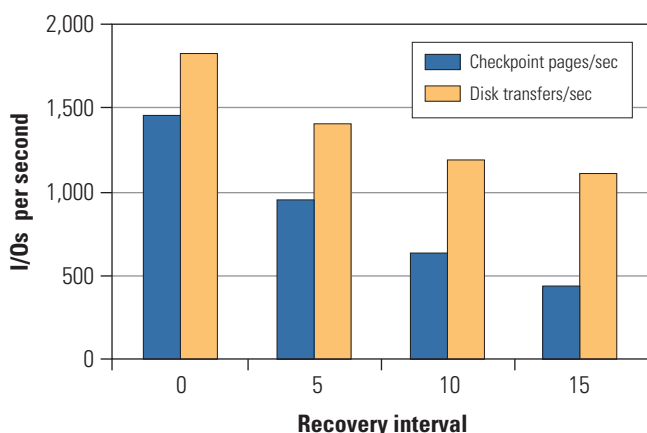


Figure 2. I/O variation with different recovery intervals

disk transfers decreased as well. Figure 3 shows a corresponding decrease in disk-transfer response time, suggesting an improvement in I/O performance.

Normally, the default value of 0 is the recommended recovery interval setting. However, in certain deployments system performance may be affected because checkpoints are issued too frequently. In such cases, administrators can increase the value in small increments, which reduces checkpoint frequency and can help improve system I/O performance. This improvement is, however, obtained at the expense of database recovery time during a restart, and the recovery interval should never be changed to a large value that increases recovery time to unacceptable levels. Administrators should thoroughly assess the trade-off between performance and recoverability before changing this parameter from the default value. In failover clustering configurations, the recovery interval should be left at the default value, because large values may lead to long failover times.

### Configuring thread scheduling

The SQL Server 2005 affinity mask options assign specific threads to processors, reducing both thread migration across processors and context switching. SQL Server 2005 includes two affinity mask options: affinity mask (also known as CPU affinity mask) and affinity I/O mask. In most cases, the OS default affinity provides the optimal performance. However, the affinity mask option can be useful for resource allocation when multiple instances of SQL Server are hosted on the same system.

Two other configuration settings related to thread scheduling are the max worker threads and lightweight pooling options. Each SQL Server instance maintains a pool of threads or fibers for its use. A fiber is a lightweight thread that requires fewer resources than a thread and can switch context in user mode. The max worker threads option controls the maximum size of a pool of threads, and

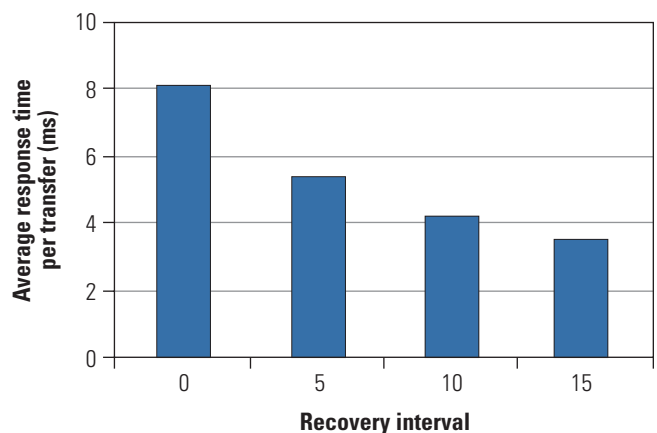


Figure 3. I/O response-time variation with different recovery intervals

the lightweight pooling option controls whether an instance of SQL Server uses threads or fibers.

### Enabling large-memory support

SQL Server 2005 can provide support for up to 1 TB of RAM with the appropriate 64-bit edition of both SQL Server 2005 and the Microsoft Windows Server® 2003 OS. On 32-bit Microsoft Windows® operating systems, SQL Server 2005 provides large-memory support using Address Windowing Extensions (AWE). AWE is a set of extensions to the Windows management functions that allows applications to acquire physical memory directly and map views of physical memory to their virtual address spaces, enabling physical memory usage beyond 4 GB and up to 64 GB. Each Windows process has its own virtual address space, the size of which depends on the architecture (32-bit or 64-bit). A 32-bit process can map only up to 4 GB; only 2 GB of this space (3 GB with the /3GB boot option) is made available to the application, and the rest is reserved for the OS.

SQL Server can directly acquire up to 64 GB of physical memory or the maximum supported by the OS as nonpaged memory using AWE (see Figure 4). AWE support was included with SQL Server 2000 on 32-bit platforms, but SQL Server 2005 enhances the AWE support by dynamically allocating and managing AWE-mapped memory to balance overall system memory needs when deployed on Windows Server 2003. The following should be taken into account when enabling AWE with SQL Server 2005:

- AWE support is provided only with SQL Server 2005 Enterprise Edition and Developer Edition.
- A maximum of 64 GB of physical memory is supported through AWE; if 3 GB of virtual address space is configured for SQL Server (using the /3GB switch), then the overall physical memory support is limited to 16 GB.
- Physical Address Extension (PAE) support for accessing physical memory greater than 4 GB is provided only on Windows 2000 Advanced Server and Datacenter Server and on Windows Server 2003 Enterprise Edition (32-bit) and Datacenter Edition (32-bit).
- Only database pages can reside in the physical memory allocated through AWE. Memory allocated through AWE cannot be used for supporting additional database users, threads, databases, queries, and other objects that permanently reside in the virtual address space.
- SQL Server 2005 Analysis Services cannot take advantage of AWE-mapped memory.

AWE requires the database engine to run under a Windows account with “Lock pages in memory” permission. Administrators can further control memory allocation using the min server memory and max server memory options, which enforce

upper and lower limits on SQL Server memory utilization. The SQL Server 2005 AWE mechanism differs when used with Windows 2000 Server as compared with Windows Server 2003. On Windows 2000 Server, AWE memory allocation is static—locked during startup and not released until shutdown. If memory allocation is not controlled using the max server memory setting, all available physical memory except 128 MB is used. On Windows Server 2003, AWE memory is allocated dynamically and returned to the system as required to balance the overall system needs. The max server memory option must be set to guarantee additional memory for other system applications or SQL Server instances. In failover clustering scenarios, all server nodes must have adequate physical memory to handle the memory requirements of all SQL Server instances for both before and after fail-over load conditions.

AWE is not necessary on 64-bit versions of SQL Server 2005 because virtual address space is not limited to 4 GB. However, the Windows account under which the database engine is running should still have “Lock pages in memory” permission. This setting enables SQL Server to hold on to the allocated memory and prevents the OS from paging the allocated memory, which can help provide performance improvements. The dynamic memory management mechanism releases locked memory appropriately if a system-wide resource shortage occurs. A 64-bit edition of SQL Server is better suited than a 32-bit edition for supporting heavy analytical processing workloads because of the large amount of

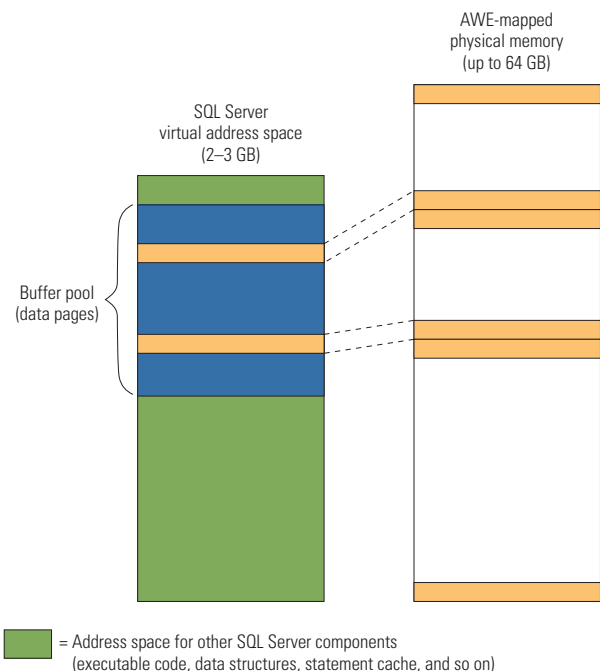


Figure 4. AWE mapping of physical memory into SQL Server virtual space

virtual address space available. On a 32-bit edition, only a limited amount of address space in the buffer pool can be used for operations like sorting, hashing, and grouping, which can have a significant impact on analytical processing workloads.

Earlier releases of SQL Server had a limited amount of memory available at startup. SQL Server 2005 introduces support for hot-add memory on Windows Server 2003 Enterprise Edition and Datacenter Edition, enabling usage of physical memory added to the system without restarting. Hot-add memory is supported on all 64-bit versions of SQL Server 2005; it is supported on 32-bit versions only if AWE is enabled. The hot-add memory feature requires specific hardware support.

### Configuring the temporary database: tempdb

The tempdb database is a globally available system database for holding temporary objects within a SQL Server instance. This database is re-created every time SQL Server is started and does not persist across restarts. Only one tempdb database is created per SQL Server instance. Temporary objects that can be held in tempdb include the following:

- **User objects:** Temporary tables, stored procedures, variables, and cursors
- **Internal database engine objects:** Temporary or intermediate results during query processing, such as sorting, spooling, and hashing
- **Row version information:** Snapshot isolation or online index operations


The size of the tempdb database can be crucial for system performance—if the database is not sized appropriately to meet workload needs, the system can become too occupied trying to auto-grow the database. SQL Server 2005 makes extensive use of tempdb and requires more disk space than earlier SQL Server versions. This change is because of the tempdb space requirements of certain features introduced in SQL Server 2005 and enhancements to existing features. SQL Server 2005 is designed to improve tempdb performance by caching temporary tables and table

SQL Server 2005 introduces support for hot-add memory on Windows Server 2003 Enterprise Edition and Datacenter Edition, enabling usage of physical memory added to the system without restarting.

variables where applicable and minimally logging tempdb operations. Page allocation mechanisms have also been enhanced, which can help improve performance.

Administrators should consider application workload and which SQL Server features are being used when determining an appropriate tempdb size. The physical placement of the tempdb database files is also important for performance: best practices recommend that they be placed on logical volumes separate from those that host other databases or applications in the system. Other best practices for optimizing tempdb performance include using RAID striping and multiple data volumes, configuring the Simple Recovery model<sup>1</sup> for the database, allocating an appropriate initial size to accommodate typical workloads, and setting suitable auto-growth values for the database files for certain unplanned peak conditions.

### Configuring efficient, high-performance SQL Server 2005 environments

Microsoft SQL Server 2005 includes configuration options that can help manage system utilization effectively. Administrators should thoroughly understand and assess these options before applying them to particular environments. The default settings for these options should satisfy the requirements of most deployments, but in scenarios where significant control of SQL Server resource usage is necessary, administrators can modify these options to suit their deployment needs. 

**Ananda Sankaran** is a systems engineer consultant in the High-Availability Cluster Development Group at Dell. His current interests related to high-availability clustering include storage systems, application performance, business continuity, and cluster management. Ananda has a master's degree in Computer Science from Texas A&M University.

**Shabana M.** is an engineering analyst in the High-Availability Cluster Development Group at Dell. Her current interests include development of SCSI and Fibre Channel clustering solutions and application performance. She has a B.E. in Computer Science and Engineering from Cochin University of Science and Technology in India.

#### FOR MORE INFORMATION

**Dell and SQL Server 2005:**

[www.dell.com/sql](http://www.dell.com/sql)

**SQL Server 2005 documentation:**

[msdn2.microsoft.com/en-us/library/ms203721.aspx](http://msdn2.microsoft.com/en-us/library/ms203721.aspx)

<sup>1</sup> For more information about the Simple Recovery model, visit [msdn.microsoft.com/library/default.asp?url=/library/en-us/adminsql/ad\\_bkprst\\_60s9.asp](http://msdn.microsoft.com/library/default.asp?url=/library/en-us/adminsql/ad_bkprst_60s9.asp).