DEPLOYING MICROSOFT SQL SERVER 2008 ON DELL POWEREDGE SERVERS

By Eric Hale
Corey Bunch
Laura White

Tuning database servers can be key to simplifying operations and enhancing performance. Best practices based on real-world experience can help administrators successfully deploy and optimize the 64-bit version of the Microsoft® SQL Server® 2008 database platform on Dell™ PowerEdge™ servers running the Microsoft Windows Server® 2008 Enterprise x64 Edition OS.

Related Categories:

Dell PowerEdge servers
Microsoft SQL Server
Microsoft Windows Server 2008

Visit DELL.COM/PowerSolutions for the complete category index.

The 64-bit versions of Microsoft SQL Server 2008 and Microsoft Windows Server 2008 introduce a wide variety of powerful, flexible features and management tools. By following best practices from Dell and Principled Technologies, administrators can create high-performance, highly available databases using the 64-bit version of SQL Server 2008 on Dell PowerEdge servers running Windows Server 2008 Enterprise x64 Edition. The practices described in this article are based on hands-on testing and research performed by Principled Technologies as well as real-world experience, and can help systems administrators and database administrators simplify operations, enhance performance and reliability, and take advantage of features introduced in the 64-bit version of SQL Server 2008.[1]

INSTALLING AND SETTING UP SQL SERVER 2008

The process of installing and setting up the 64-bit versions of SQL Server 2008 and Windows Server 2008 on a Dell PowerEdge server is typically straightforward. When carrying out this process, administrators should be sure they are using the latest tested and validated software, firmware, and driver versions for network interface cards, storage arrays, and other components.[2] They should also deploy the software in a Microsoft Active Directory® domain and use Microsoft Windows® authentication, which allows them to take advantage of the integrated security and centralized management features of Active Directory.

Installing Windows Server 2008

The first step is to install Windows Server 2008, a process that usually takes at least one hour. In general, administrators should configure all database servers with static IP addresses, which helps increase stability and ensure that SQL Server 2008 remains available even after a Dynamic Host Configuration Protocol (DHCP) server failure. In addition, they should typically use canonical name (CNAME) records to assign aliases to database servers, enabling them to isolate users and applications from changes to the underlying server infrastructure, which helps simplify deployments and migrations.

After performing the initial installation, administrators should next configure the remaining drives on the server, formatting all SQL Server volumes—including those for data, logs, and temporary database (tempdb) files—as NT File System (NTFS) volumes. Microsoft recommends a 64 KB allocation unit size for these volumes. Administrators should avoid values

[1] For the full version of this deployment guide, including step-by-step instructions on installing and setting up SQL Server 2008 on a Dell PowerEdge 2950 server, see "SQL Server 2008 x64 on Windows Server 2008 Enterprise x64 on Dell PowerEdge 2950," by Principled Technologies, March 2008, DELL.COM/Downloads/Global/Solutions/Public/White_Papers/sql2008_ws2008.pdf.

[2] For more detailed information on SQL Server 2008, visit DELL.COM/SQL2008.

less than 8 KB, which can increase the risk of torn pages whose contents are split across disk allocation units.

Because Windows Server 2008 Enterprise x64 Edition does not support compressing drives when the allocation unit size is 64 KB, and the 64-bit version of SQL Server only creates read-only databases on compressed drives, compressed drives should not be used. SQL Server 2008 does include built-in compression features that administrators can use when appropriate to their performance and capacity requirements.

Best practices recommend separating tempdb and transaction log files onto their own disks on separate disk groups when possible; doing so can help increase I/O performance by helping ensure that these files do not share physical disks. Likewise, administrators should create full-text catalogs on their own physical disks. Administrators should also group files with similar I/O characteristics, such as log files; because heterogeneous workloads may have different and possibly competing I/O characteristics, combining them can reduce overall performance.

To help achieve optimal performance that scales with heavy workloads, Microsoft recommends having from 0.25 to 1 data file (per file group) for each server processor core. If these files share spindles, however, the server can experience contention when multiple database processes access them simultaneously. In this case, administrators should consider breaking the data files into a number of files equal to half the number of cores. With tempdb files, they should use one data file per core.

Installing SQL Server 2008

Administrators should typically allow at least 30 minutes for installing SQL Server 2008. In general, they should not deploy it on an Active Directory controller because Active Directory processes add overhead that could reduce SQL Server performance. They should, however, deploy it on a member server in an Active Directory domain. They should also grant the SQL Server service account only necessary rights on the local server, and should not make SQL Server service accounts members of the Domain Administrators group.[3]

After completing the installation, administrators should give the SQL Server service account the right to prevent the OS from paging its memory to disk by enabling the "lock pages in memory" setting (see Figure 1). SQL Server can dynamically allocate and deallocate memory to help relieve memory pressure and swapping; however, another process can request a substantial amount of memory and cause the OS to swap SQL Server memory to disk before SQL Server can react. The "lock pages in memory" setting helps prevent this problem.

When possible, administrators should leave the minimum and maximum server memory at their default settings of 0 and 2,147,483,647, respectively, enabling SQL Server to use as much memory as the system makes available. If changing these settings is necessary, they should ensure that the sum of the maximum memory settings across all processes is less than the amount of physical RAM available.

Finally, administrators should also enable instant file initialization. The default behavior is to initialize the storage with zeros when SQL Server creates a data file or allocates an extent to grow a data file, which can be time-consuming. Instant file initialization stops the system from initializing storage it allocates; instead, the storage remains un-initialized until SQL Server writes data to it. Microsoft testing shows a significant performance improvement when using instant file initialization.

ADMINISTERING AND MONITORING SQL SERVER 2008

Optimizing database server performance depends on more than a successful installation: the run parameters and database layout, for example, require careful thought and advance planning. When administering and monitoring the 64-bit version of SQL Server 2008, following best practices for configuring key settings and tracking performance data can help create an optimized, reliable deployment.

Administering SQL Server 2008

Key settings such as autogrowth, autoclose, tempdb data file creation, and processor and memory parameters can help administrators optimize SQL Server 2008 performance and avoid problems related to inappropriate settings.

Autogrowth. Administrators should be sure to size data, log, and tempdb files appropriately. Because autogrowth can create file fragmentation and reduce performance, they should plan file growth and manually expand files when the server is relatively idle rather than relying on autogrowth to size files for them.

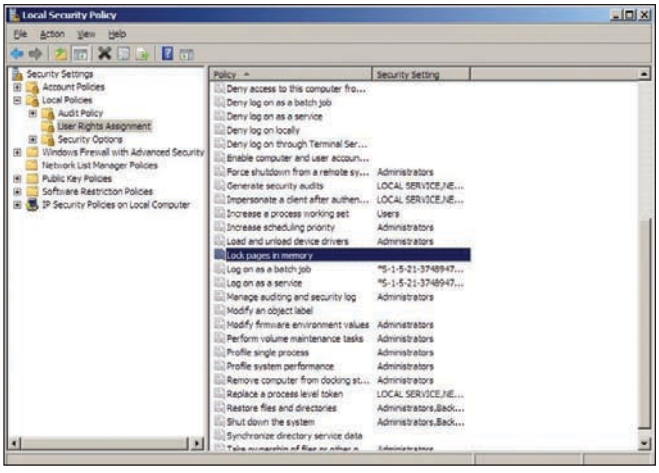

Figure 1. *"Lock pages in memory" setting in the 64-bit version of Microsoft SQL Server 2008*

[3] For more information on domain accounts for SQL Server service accounts, see "Setting Up Windows Service Accounts," by Microsoft Corporation, SQL Server 2008 Books Online, May 2008, msdn.microsoft.com/en-us/library/ms143504(SQL.100).aspx#Review_NT_rights.

Although instant file initialization can mitigate this performance impact, it does not eliminate the problem and does not help reduce file fragmentation. Administrators should typically use autogrowth only as a precaution to help prevent data files from filling up and forcing the database to a read-only state.

The autogrowth increment should be small enough to limit the performance impact of unplanned file growth and large enough to help prevent excessive file fragmentation. In general, best practices recommend setting this increment using fixed sizes—in megabytes or gigabytes—rather than percentages, which can lead to uncontrolled file growth. In addition, administrators must take into account the virtual log files within each physical log file. Because these virtual log files cannot span file extents, the autogrowth increment also limits the maximum size of virtual logs, which can make certain large transactions impossible to complete. Even when administrators have selected an appropriate increment, however, manually sizing files during periods of low activity is still preferred.

Autoclose. Administrators should typically leave the autoclose option at its default value of false. For a frequently used database, autoclose can cause many unnecessary opens and closes, which can reduce system performance.

Tempdb data file creation. The SQL Server installation process creates a single tempdb primary data file. Using a single tempdb data file, however, can create unacceptable latch contention and I/O performance bottlenecks. To mitigate these problems, administrators should allocate one tempdb data file per processor core.

Autogrowth is enabled for tempdb files by default. As with data files, however, expanding tempdb files too frequently can reduce performance. Therefore, administrators should allocate enough initial space to the tempdb files to accommodate the expected workload. As a precaution, they should also set the file growth increment large enough to minimize tempdb expansions. By default, the file growth increment for tempdb files is 10 percent, with an unrestricted growth increment of 10 percent. Although a fixed increment for data files has several advantages, Microsoft recommends the 10 percent increment; the most suitable approach depends on the specific environment.

Administrators should also make all tempdb files equal in size. Because SQL Server uses a proportional fill algorithm, differently sized files do not distribute the I/O load evenly.

Processor and memory parameters. In general, administrators should leave the SQL Server processor and memory parameters at their default settings unless they need to accommodate special requirements. These settings run SQL Server at a standard priority, which lets it use all processors in the server and makes as much RAM available as SQL Server needs. Similarly, administrators should generally leave the network packet size at the default value of 4,096 bytes unless increasing it can enhance efficiency—which may be the case, for example, when processing bulk copy operations or using large image files.

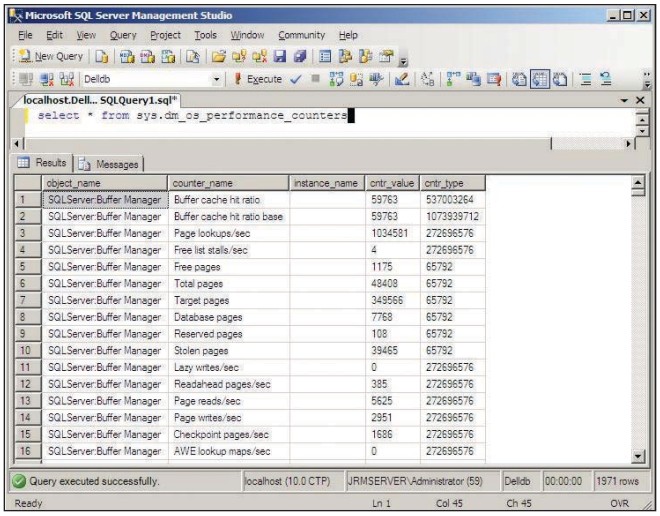

Figure 2. *Example results from the sys.dm_os_performance_counters statistic*

Best practices also recommend running SQL Server 2008 on dedicated servers when possible. Other applications running on the same server can reduce performance by competing with SQL Server for resources, and each additional application makes it increasingly difficult to tune the server for optimal database performance or to troubleshoot problems.

Monitoring SQL Server 2008

Once the server is running, administrators should monitor its performance to help them tune it for their specific workload. The 64-bit versions of Windows Server 2008 and SQL Server 2008 provide many monitoring tools and statistics, including the Windows Server 2008 Reliability and Performance Monitor, SQL Server 2008 dynamic management views and functions (DMVs/DMFs), and the SQL Server 2008 Performance Data Collector.

Reliability and Performance Monitor. The Windows Server 2008 Reliability and Performance Monitor includes multiple performance statistics. For example, the PhysicalDisk/% Idle Time counter tracks the percentage of idle time, which can indicate I/O performance; the Processor/% Processor Time counter tracks the percentage of processor time; and the SQLServer:BufferManager/Buffer Cache Hit Ratio counter tracks the buffer cache hit ratio, which indicates how often SQL Server is using physical memory to retrieve data. In general, best practices recommend that the PhysicalDisk/% Idle Time counter be greater than 20 percent, the Processor/% Processor Time counter be less than 70 percent, and the SQLServer:BufferManager/Buffer Cache Hit Ratio counter be greater than 95 percent.

Dynamic management views and functions. SQL Server 2008 offers over 100 DMVs/DMFs, which expose the current state of a SQL Server system. Administrators can use statistics collected from a DMV/DMF query to analyze server health and diagnose potential problems. Some of these statistics apply to the entire server, while others apply to specific databases. The sys.dm_os_performance_counters statistic, for example, returns the current

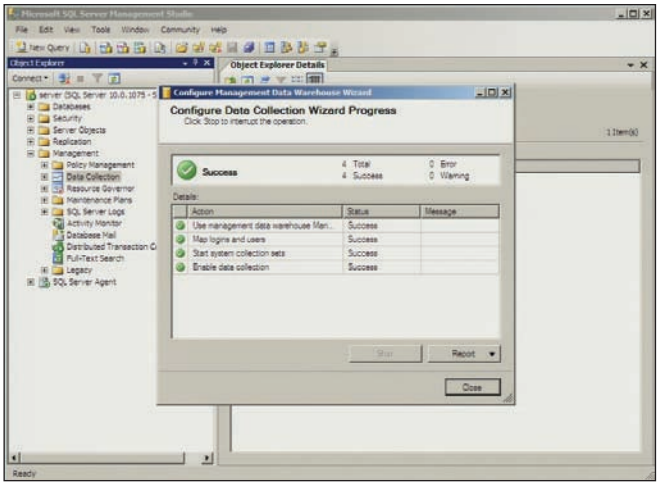

Figure 3. *Completed Configure Management Data Warehouse wizard in the 64-bit version of Microsoft SQL Server 2008*

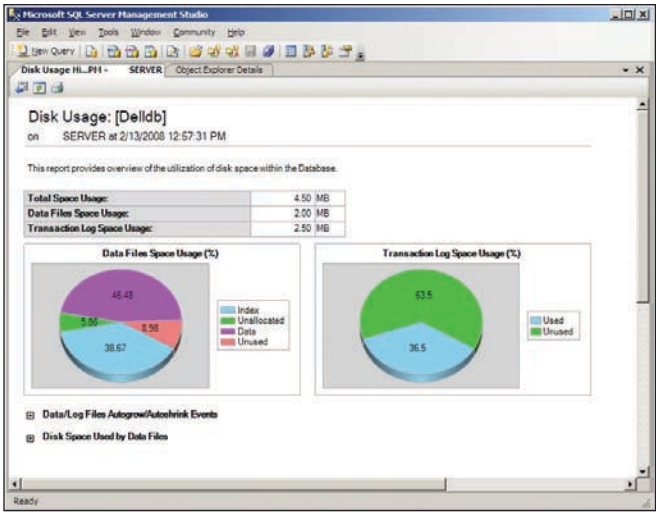

Figure 4. *Example report for the 64-bit version of Microsoft SQL Server 2008*

value of certain SQL Server performance counters (see Figure 2). Best practices for performance monitoring recommend capturing baseline data and setting up a regular monitoring plan.[4]

Performance Data Collector. The 64-bit version of SQL Server 2008 introduces the Performance Data Collector, which builds on the 64-bit SQL Server 2008 Data Collector architecture[5] and provides a framework for collecting diagnostic and performance data to help simplify database tuning. The Performance Data Collector is part of the broad set of capabilities in the Microsoft Performance Studio, which enables administrators to collect performance data from multiple databases and store it in a central repository so they can compare past and current SQL Server performance.

The Performance Data Collector stores its data in a special Management Data Warehouse database and uses three system data collection sets: Disk Usage, Server Activity, and Query Statistics. The Disk Usage set collects data about disk and log usage for databases on a server, the Server Activity set collects resource usage statistics and performance data from the server and SQL Server, and the Query Statistics set collects query statistics, query text, query plans, and specific queries.[6] Administrators can configure data collection in the SQL Server Management Studio using the Configure Management Data Warehouse wizard (see Figure 3), or can configure collection sets individually.

Once the Performance Data Collector is running, administrators can wait for the collection schedule to cycle completely or update the data immediately. They can then view a historical report or create a custom report using the 20 fields available for each collection (see Figure 4).

BEST PRACTICES FOR SQL SERVER 2008

The 64-bit versions of Microsoft SQL Server 2008 and Windows Server 2008 introduce multiple features and enhancements for enterprise IT environments. Advance planning and following the best practices described in this article can help administrators successfully deploy and optimize SQL Server 2008 on Dell PowerEdge servers running Windows Server 2008 Enterprise x64 Edition.

Eric Hale is a senior analyst with Principled Technologies.

Corey Bunch is an analyst with Principled Technologies.

Laura White is a senior technical writer and editor with Principled Technologies.

MORE ONLINE
DELL.COM/PowerSolutions

QUICK LINKS

Full deployment guide:
DELL.COM/Downloads/Global/Solutions/Public/White_Papers/sql2008_ws2008.pdf

Dell database solutions:
DELL.COM/SQL2008

Microsoft SQL Server 2008:
www.microsoft.com/sql/2008

[4] For more information on DMVs/DMFs, including query options and required parameters, see "Dynamic Management Views and Functions (Transact-SQL)," by Microsoft Corporation, SQL Server 2008 Books Online, May 2008, msdn2.microsoft.com/en-us/library/ms188754(SQL.100).aspx.

[5] For more information on this architecture, see "Data Collector Architecture and Processing," by Microsoft Corporation, SQL Server 2008 Books Online, May 2008, msdn2.microsoft.com/en-us/library/bb677355(SQL.100).aspx.

[6] For more information on data collection sets, see "System Data Collection Sets," by Microsoft Corporation, SQL Server 2008 Books Online, May 2008, technet.microsoft.com/en-us/library/bb964725(SQL.100).aspx.

