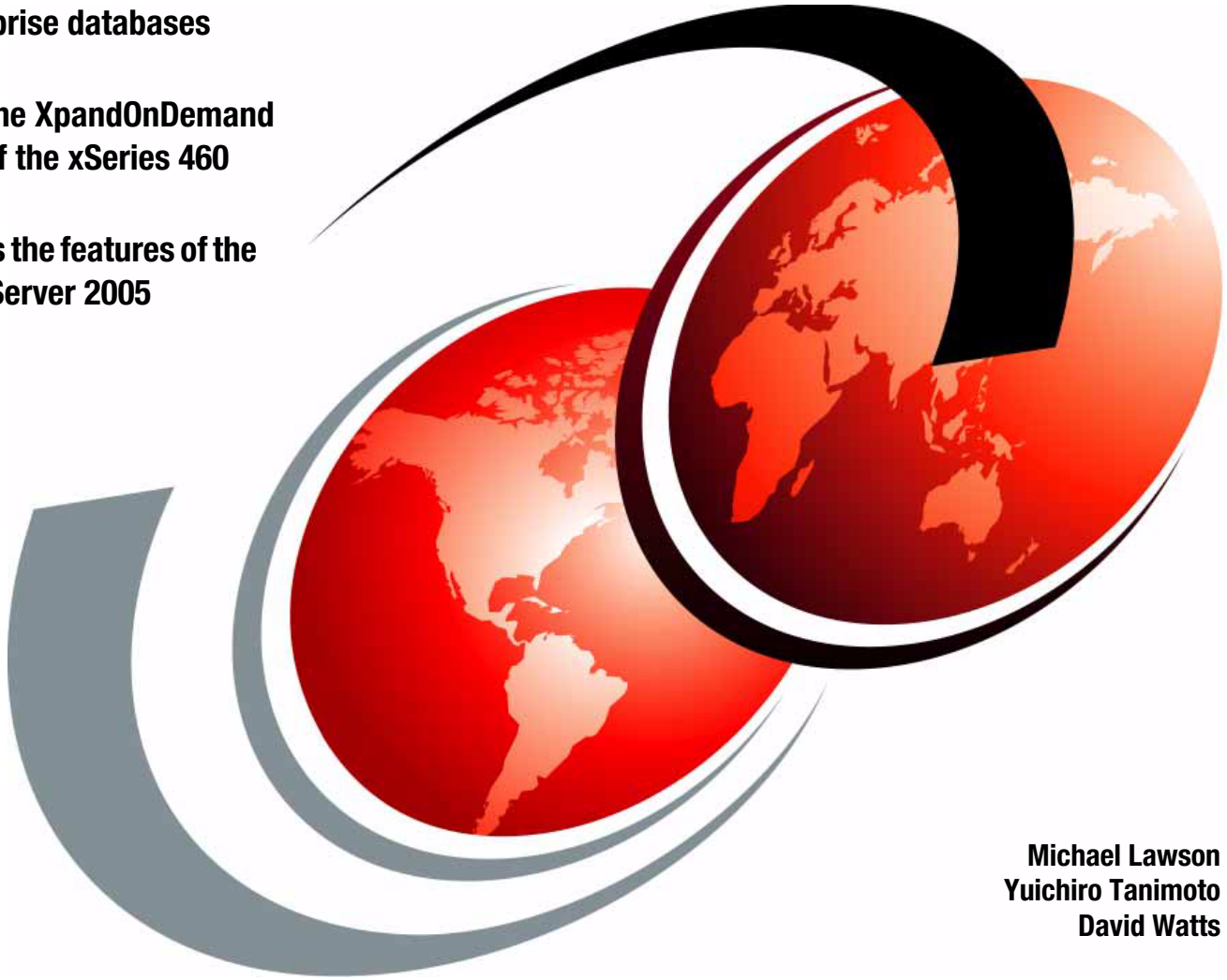
IBM @server

IBM

SQL Server 2005 on the IBM @server xSeries 460 Enterprise Server

- Describes how “scale-up” solutions suit enterprise databases
- Explains the XpandOnDemand features of the xSeries 460
- Introduces the features of the new SQL Server 2005



Michael Lawson
Yuichiro Tanimoto
David Watts

Redpaper

ibm.com/redbooks

International Technical Support Organization

SQL Server 2005 on the IBM @server xSeries 460 Enterprise Server

December 2005

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (December 2005)

This edition applies to Microsoft SQL Server 2005 running on the IBM @server xSeries 460.

Contents

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

This document created or updated on December 1, 2005.

Send us your comments in one of the following ways:

- Use the online **Contact us** review redbook form found at:
  ibm.com/redbooks
- Send your comments in an email to:
  redbook@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HZ8 Building 662
  P.O. Box 12195
  Research Triangle Park, NC 27709-2195 U.S.A.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

@server®	Chipkill™	ServeRAID™
@server®	IBM®	TotalStorage®
Redbooks (logo) ™	Netfinity®	X-Architecture™
iSeries™	Redbooks™	
xSeries®	ServerGuide™	

The following terms are trademarks of other companies:

Microsoft, Windows, Windows NT, Excel, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Preface

The IBM @server® xSeries® 460 is the number one x86 scalable server in the industry. The combination of the x460 with Microsoft® SQL Server 2005 creates a perfect match for addressing customer needs for more database performance. The x460 is an effectve four-way server, but provides optimal scalability for those customers who need room to grow.

One of the key markets for the x460 is database applications such as SQL Server 2005 (“Yukon”). SQL Server 2005 can effectively take advantage of the 32 processors and 512 GB of RAM available with a large x460 solution. This combination provides customers with the performance and scalability required to address their complex database workload needs.

The x460 is an example of *scale-up* technology. With the x460, when customers want to increase the capacity of their server infrastructure, they can simply add nodes to the x460 to increase the number of central processing units (CPUs) and the amount of installed memory. The advantage of scaling up (versus *scale-out* where customers extend their computing resources by purchasing additional and separate servers) is simpler management and control. The x460 starts at two processors and 2 GB of RAM and scales up to 32 processors and 512 GB of RAM.

This paper describes how the “scale-up” features of the xSeries 460 and SQL Server 2005 are an ideal fit. The paper discusses the key features of each, including how the multi-node scalability is designed to ensure a near-linear scalability from a single-node four-way server all the way to an 8-node 32-way complex. This paper also offers performance tuning advice directly from the IBM® performance labs.

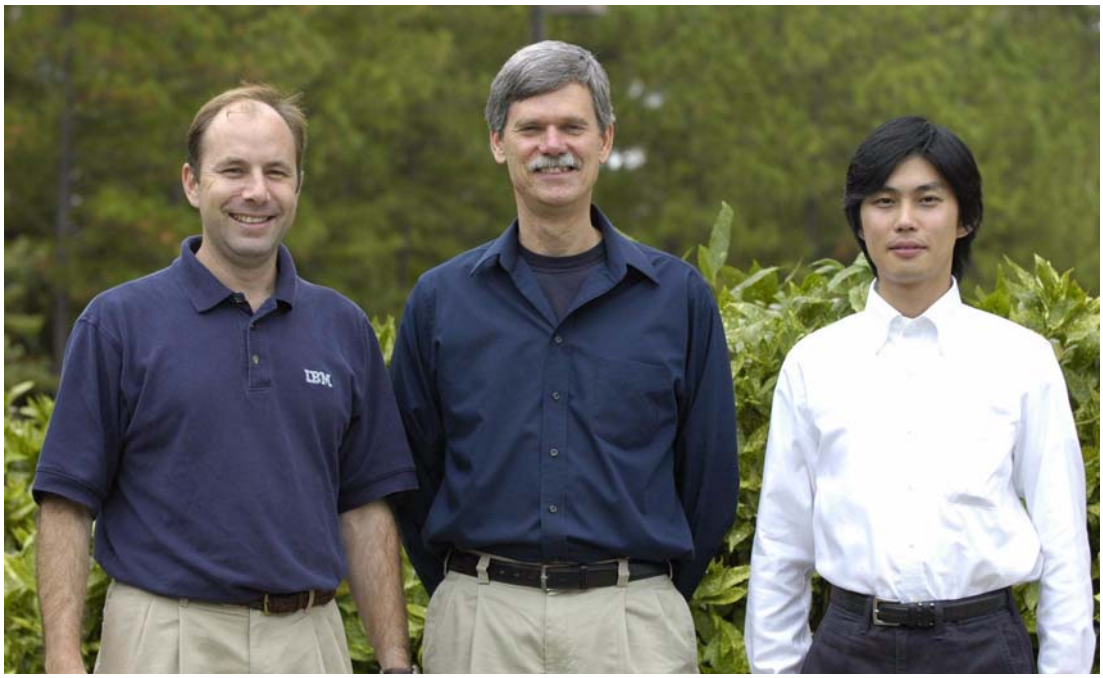
The team that wrote this Redpaper

This Redpaper was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Michael Lawson is a database and virtualization specialist at the IBM Center for Microsoft Technologies in Kirkland, WA. He has a B.A. in Information Sciences and Mathematics from the University of California at Santa Cruz. Michael has 25 years of experience as a database administrator, the last 10 years with Microsoft SQL Server. Since joining IBM in 1999, he has supported IBM customers running SQL Server on xSeries servers, in the Windows® Solution Lab. He has MCDBA (Microsoft Certified Database Administrator) certification.

Yuichiro Tanimoto is an IT Specialist at IBM Global Services in IBM Japan. He has been engaged with Intel® architecture-based servers and Microsoft products since he started his IBM career in 1997. His area of expertise is performance optimization of Microsoft SQL Server, Windows Server, SAN (TotalStorage® DS4000 series), and xSeries. He holds a Bachelor of Law degree from Gakushuin University in Japan. He has MCDBA (Microsoft Certified Database Administrator) certification.

David Watts is a Consulting IT Specialist at the IBM ITSO Center in Raleigh. He manages residencies and produces Redbooks™ on hardware and software topics related to xSeries systems and associated client platforms. He has authored over 30 redbooks and redpapers. He holds a Bachelor of Engineering degree from the University of Queensland (Australia) and has worked for IBM for over 15 years. He is an IBM @server Certified Specialist for xSeries and an IBM Certified IT Specialist.



The team (left to right): David Watts, Michael Lawson, Yuichiro Tanimoto

Thanks to the following people for their contributions to this project:

IBM Corporation:

Susan Goodwin, Senior Performance Engineer, Kirkland
Michael Lee, Senior Systems Engineer, Kirkland
Daniel Ghidali, Advisory Engineer, Enterprise Solutions, Kirkland
Ron Arndt, Software Engineer, Kirkland
Ralph Begun, Lead xSeries Engineer, Raleigh
Linda Robinson, ITSO Graphics Editor, Raleigh

Microsoft Corporation:

Kevin Cox
Slava Oks
The SQLOS development team

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this Redpaper or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  ibm.com/redbooks

- Send your comments in an e-mail to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HZ8 Building 662
  P.O. Box 12195
  Research Triangle Park, NC 27709-2195

1

The x460 enterprise server

Delivering an industry-leading, 64-bit framework for high-performance scalable computing, the IBM @server xSeries 460 is built on the power of IBM @server X3 Architecture, the third generation of IBM Enterprise X-Architecture™ technology. X3 Architecture drives the x460 to deliver the performance, availability, expandability, and manageability required for the next generation of industry-standard servers.

Four-socket performance, expandable to 32-way, and 64-bit memory addressability provide an optimized high-end database platform. At the crossroads of a major industry transition to mainstream 64-bit applications and dual-core processors, X3 Architecture delivers a formidable combination of 64-bit performance, availability, and investment protection that is not yet available in an industry-standard x86 server.

Extensive chipset development experience and industry-leading performance and availability breakthroughs have uniquely positioned IBM to propose a robust and powerful server, offering innovation that delivers real business and IT results.

This chapter covers the following topics:

1.1 Scale up versus scale out

There are two trends in the design of server solutions: *scale-out* and *scale-up*.

Scale-out refers to the idea of increasing processing capacity by adding additional servers to a solution. Adding servers to a Web farm to handle larger numbers of users is a good example of scale-out. A user can connect to any Web server, because there is no shared data and the Web applications are stateless.

Scale-out is not typically a database solution; however, SQL Server can do scale-out using distributed partitioned views, also called federated databases. This can work well when the data can be partitioned on a natural boundary, like geographical region or division of a company. However, managing a distributed partition configuration can require significant manual effort.

Scale-up refers to the idea of increasing processing capacity by adding additional processors (and memory and I/O bandwidth) to a single server, making it more powerful. This is precisely what the x460 is designed to do: to scale up by adding chassis to a hardware partition to form two-node, four-node, and eight-node configurations. SQL Server 2005 is also designed for scale-up.

Thus, when your database application needs to scale-up to handle increased demands, the x460 and SQL Server 2005 together can grow from a one-node or two-node server to an enterprise-class eight-node server with 32 processor sockets and 512 GB of physical memory, which the 64-bit SQL Server 2005 x64 can take full advantage of.

1.2 X3 Architecture

X3 Architecture is the culmination of many years of research and development that has resulted in what is currently the fastest processor and memory controller in the Intel processor marketplace. With support for up to 32 Xeon MP processors and over 20 Gbps of memory bandwidth per 64 GB of RAM up to a maximum of 512 GB, the xSeries servers that are based on the X3 Architecture offer maximum performance and broad scale-up capabilities.

1.2.1 The X3 Architecture servers

The three servers based on the X3 Architecture are the x460, x366, and x260. They have a common set of technical specifications and features, but there are key differences.

The **xSeries 460** is the flagship server with the following characteristics:

- Each chassis occupies 3U of rack space and supports four CPUs and 64 GB of RAM.
- It has six hot-swap drive bays and six 266 MHz PCI-X 2.0 hot-swap slots.
- It is targeted at eight-way and above configurations where effective scale-up options are essential.
- Up to eight systems can be connected together to form one single 32-way complex with up to 512 GB RAM.

The **xSeries 366** is a high-performance four-way server with the following characteristics:

- With the same mechanical design as the x460, it offers up to 64 GB of RAM and up to four Xeon MP processors.
- It is targeted at two-way and four-way high performance commercial computing such as database, e-mail, and e-commerce applications.

The **xSeries 260** is also a high-performance four-way server:

- Same central electronics as the x366 and x460
- Larger 7U chassis to house up to 12 hot-swap disk drives and a full-height internal tape drive
- Targeted at two-way and four-way high performance commercial computing applications where more internal disk storage is required

1.2.2 Key features

The x460, x366, and x260 have a number of common features:

- X3 Architecture that features the XA-64e third-generation chipset
- Models with single-core or dual-core processors and single-core models are upgradable
- Common system boards: the CPU/memory board, the I/O board, and the PCI-X board
- Up to four Intel Xeon™ MP processors that support 64-bit addressing with the Intel Extended Memory 64 Technology (EM64T) architecture
- Up to 64 GB of RAM, using high performance PC2-3200 ECC DDR2 dual inline memory modules (DIMMs)
- Active Memory with Memory ProteXion, memory mirroring, memory hot-swap and hot-add, and Chipkill™
- Six full-length 64-bit 266 MHz PCI-X 2.0 Active peripheral component interconnect (PCI) slots
- Integrated Adaptec AIC-9410 serial-attached SCSI (SAS) controller
- Support for internal RAID using an optional ServeRAID™-8i adapter and ServeRAID-6M also supported for external SCSI storage with the EXP400 enclosure
- Integrated dual-port Broadcom 5704 PCI-X Gigabit Ethernet
- Integrated Baseboard Management Controller; Remote Supervisor Adapter II SlimLine adapter standard (x460) or optional (x366 and x260)
- Support for the IBM Integrated xSeries Adapter for iSeries™ (IXA) for a direct high speed link to an iSeries server (x460 and x366 only)
- Hot-swap fans and power supply
- Light path diagnostics to identify any failed components

- Three-year warranty on site, nine hours per day, five days per week, with a next business day response

The key features that differ between the three servers are shown in Table 1-1.

Table 1-1 Key feature differences

	xSeries 460	xSeries 366	xSeries 260
Processors	Intel Xeon MP “Potomac” or Xeon 7020/7040 “Paxville” processors	Intel Xeon MP “Cranford” or Xeon 7020/7040 “Paxville” processors	Intel Xeon MP “Cranford” processors
Installed / max processors	2 /4	1 / 4	1 / 4
Memory standard / maximum	2 / 64 GB	2 / 64 GB	1 or 2 / 64 GB
Largest configuration	8 nodes (32-way)	1 node (4-way)	1 node (4-way)
Rack height	3U	3U	7U
Tower-to-rack conversion	No	No	Yes
Power supplies	2x 1300W supplies (650W at 110V), both standard	1300W supplies (650W at 110V), one standard, one optional	2x 775W supplies / 4 (220V or 110V), both standard
Remote Supervisor Adapter II SlimLine	Standard	Optional	Optional
Hot-swap disk drive bays	Six (2.5” bays, SAS)	Six (2.5” bays, SAS)	Six standard, additional six optional (3.5’ bays)
Optical media	8x DVD-ROM	8x DVD-ROM	40x CD-ROM
Diskette drive	Optional (external USB)	Optional (external USB)	Standard (internal)
Tape drive bay	No	No	Two half-high 5.25” bays, can be used as a single full-height bay

1.2.3 IBM XA-64e third-generation chipset

The x460s use the third generation IBM XA-64e chipset. The architecture consists of the following components:

- One to four Xeon MP processors
- One Hurricane Memory and I/O Controller (MIOC)
- Two Calgary PCI Bridges

Figure 1-1 on page 5 shows the block diagram of the X3 Architecture.

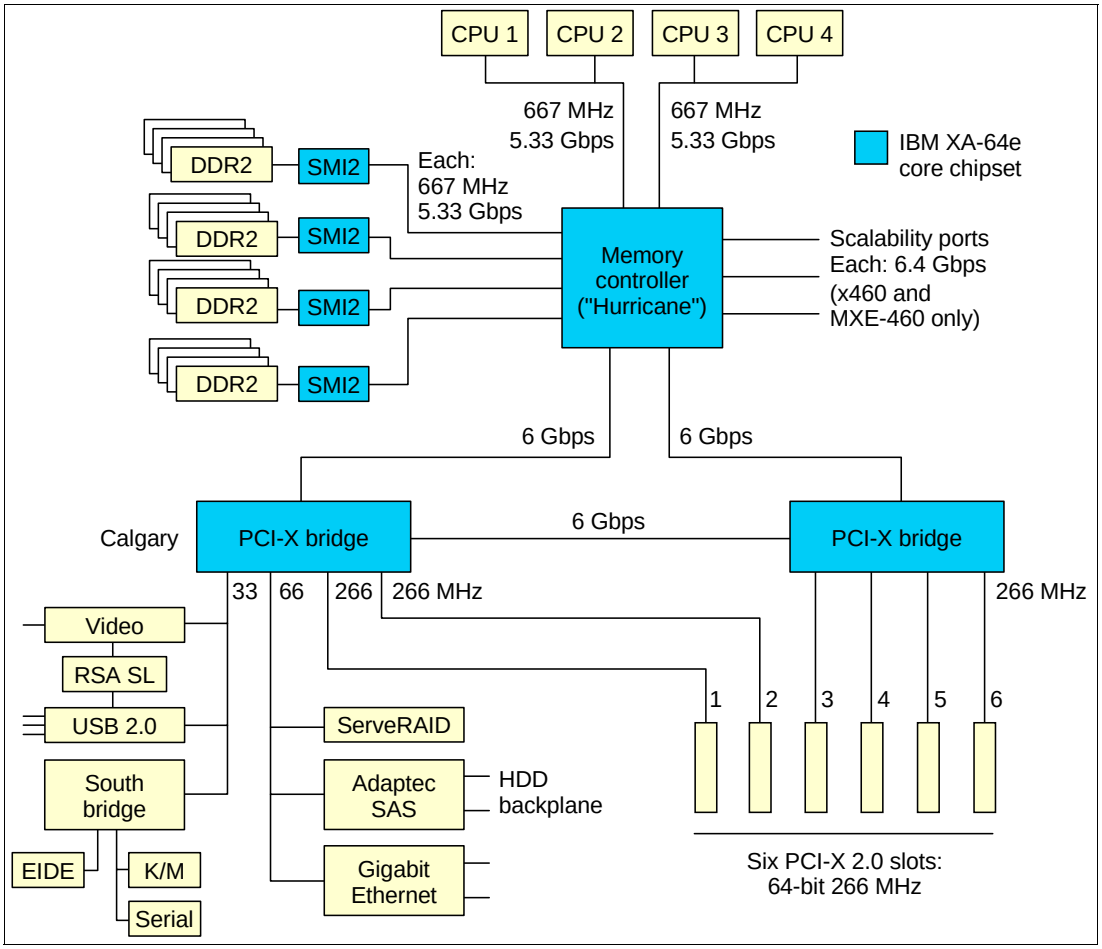


Figure 1-1 X3 Architecture system block diagram

Each memory port out of the memory controller has a peak throughput of 5.33 Gbps. DIMMs are installed in matched pairs with two-way interleaving to ensure that the memory port is fully utilized. Peak throughput for each PC2-3200 DDR2 DIMM is 2.67 Gbps.

There are four memory ports, and spreading installed DIMMs across all four memory ports can improve performance. The four independent memory ports, or memory cards, provide simultaneous access to memory. With four memory cards installed, and DIMMs in each card, peak memory bandwidth is 21.33 Gbps.

The memory controller routes all traffic from the four memory ports, two microprocessor ports, and the two PCI bridge ports. The memory controller also has embedded DRAM, which holds a snoop filter lookup table. This filter ensures that snoop requests for cache lines go to the appropriate microprocessor bus and not to both of them, therefore improving performance.

One PCI bridge supplies four of the six 64-bit 266 MHz PCI-X slots on four independent PCI-X buses. The other PCI bridge supplies the other two PCI-X slots (also 64-bit, 266 MHz) plus all the onboard PCI devices, including the optional ServeRAID-8i and Remote Supervisor Adapter II SlimLine daughter cards.

1.2.4 XceL4v cache

The XceL4v Dynamic Server Cache is a new technology developed as part of the IBM XA-64e third-generation chipset. It is used in two ways:

- For a single four-way server, the XceL4v and its embedded DRAM (eDRAM) is used as a snoop filter to reduce traffic on the front side bus. It stores a directory of all processor cache lines to minimize snoop traffic on the dual front side buses and minimize cache misses.
- When the x460 is configured as a multi-node server, this technology dynamically allocates 256 MB of main memory in each node for use as an L4 cache directory and scalability directory. In a 32-way configuration, the result is 2 GB of XceL4v cache.

Used in conjunction with the XceL4v Dynamic Server Cache is an eDRAM, which in single-node configurations contains the snoop filter lookup tables. However, in a multi-node configuration, this eDRAM contains the L4 cache directory and the scalability directory.

Note: The amount of memory that BIOS reports is the result of subtracting the XceL4v cache from the installed memory.

1.3 XpandOnDemand

XpandOnDemand is the term given to the ability of the x460 to scale-up as required by adding nodes to an x460 complex. This ability ensures that customers only pay for the computing resources they need at the present time without sacrificing the investment they have made in hardware and software should they choose to upgrade.

The investment protection features of the x460 include:

- Processors: Customers can upgrade from two-way to four-way, eight-way, 16-way, and ultimately 32-way processing should they require it.
- Memory: The standard 2 GB can be expanded to 512 GB
- PCI-X slots: The x460 has six PCI-X 2.0 slots standard, but an x460 complex can be expanded to 48 slots if required.
- Drives and USB devices: The number of internal disk drives and other internal resources is also increased as a result of such expansions.

These “pay-as-you-grow” design options are achieved by connecting x460 servers (or x460s with MXE-460 expansion units) together to form multi-chassis (or multi-node) complexes. Each node contains processors, memory and PCI-X slots, and other components in the x460. By joining the nodes together, the single operating system running in the entire complex has full access to all resources in all attached computing nodes.

In addition, after the nodes are connected together to form a larger complex, should the business need arise, you can divide the complex back into partitions. These partitions are formed on node boundaries. For example, an eight-node x460 complex could be divided into a four-node partition and two 2-node partitions, each running its own operating system. This flexibility ensures that the computing resources match the need of the business.

The x460 uses the Intel Xeon MP processors with EM64T extensions to support 64-bit operating systems. This allows you to grow more easily as the needs of your business change over time, transitioning to 64-bit applications or adding incremental performance capacity when you need it without the penalty of paying for costly up-front infrastructure.

The x460 also supports the next wave of processor technology with the Intel dual-core CPUs.

The main advantages of XpandOnDemand capability are:

- Investment protection: You pay only for the performance that is necessary today. You do not have to pay more then you need now. XpandOnDemand supports 32-bit and 64-bit applications on the same platform. You can migrate to 64-bit as needed or when the 64-bit versions of commercial applications become available.
- Pay-as-you-grow: Server performance can grow with your company. There is no need to buy a new, more powerful server; just expand the server you already have.
- Configuration and performance flexibility: Server configuration and performance can be simply modified to smooth peak loading or to perform certain periodic, more processor-intensive tasks such as weekly accounting and billing.
- Near linear performance increase: When you add nodes to xSeries 460 based on Enterprise X-Architecture, you naturally gain additional processors and power. You also acquire greater memory capabilities, more PCI-X slots, more internal storage, more front side buses, more memory controllers with additional memory buses, and more chipsets on the motherboard, all of which can achieve a near linear performance increase.

1.4 Memory subsystem

The x460 uses one to four memory cards to implement memory. Each card holds four DIMMs as shown in Figure 1-2 on page 8. The servers have one or two memory cards installed as standard (this depends on the model). To achieve maximum performance, it is best if you install and use as many memory cards as possible.

The standard installation of the x460 has two memory cards. Memory is two-way interleaved to ensure maximum data throughput. Standard installed RAM is model dependant and combinations of 512 MB, 1 GB, 2 GB, and 4 GB DIMMs are supported. With 4 GB DIMMs, a total amount of 64 GB RAM can be installed. In an x460 eight-node system, the maximum installable memory size is 512 GB (8 x 64 GB).

x460 Multi-node configurations: As discussed in 1.5, “Multi-node configurations” on page 9, with x460 multi-node configurations, 256 MB of memory in each node is allocated to the XceL4v cache. As a result, the memory as seen by the operating system is reduced by 256 MB for each node (2 GB for an eight-node complex, for example).

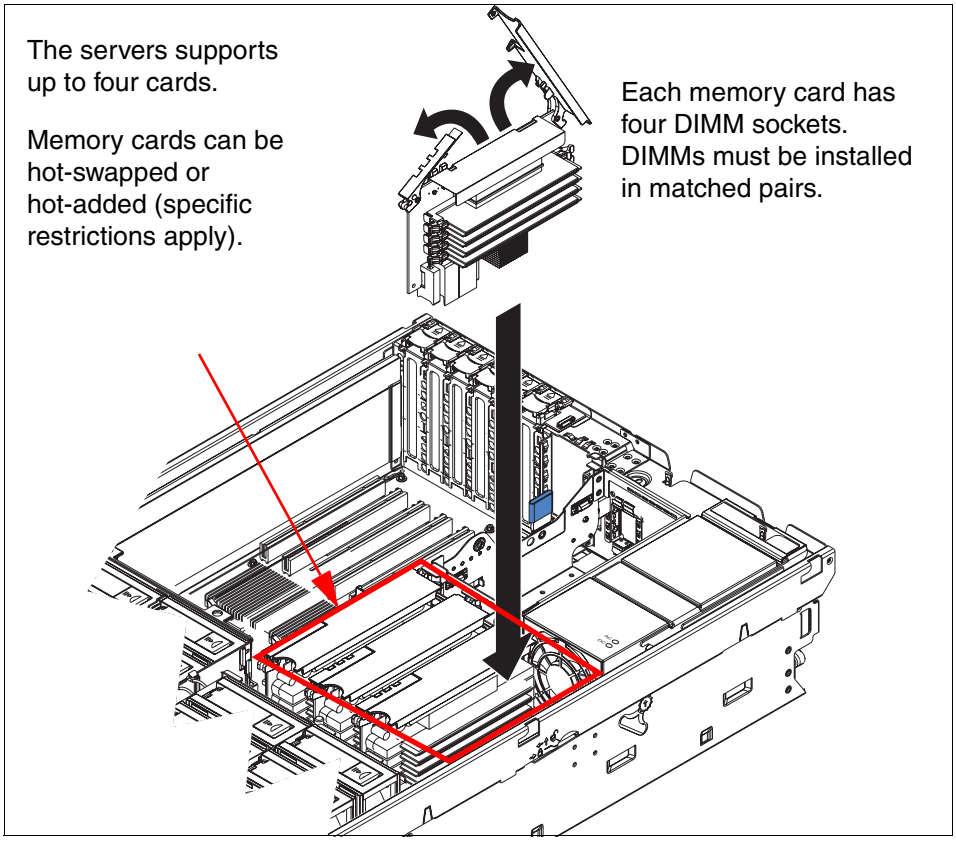


Figure 1-2 Memory card locations

The memory subsystem has the following features:

- Memory mirroring and hot-swap

  Memory mirroring is available on all X3 Architecture servers for increased fault tolerance. Memory mirroring is operating system independent, because all mirroring activities are handled by the hardware.

  When memory mirroring is enabled in BIOS, the system divides memory in half and configures one half to be a mirror copy of the other. In the event of a failure, the mirrored copy of the data is used until the memory is replaced (either while the server is running with the hot-swap feature or with the server shut down).

  The x460 supports hot-swap memory, which means that if a DIMM fails, you can replace it with a new DIMM without stopping the server. This advanced feature allows for maximum system availability. Hot-swap memory requires that memory mirroring be enabled.

  There is no performance degradation when implementing memory mirroring for the same amount of usable RAM. For example, if you have 2 GB of RAM in a non-memory mirrored configuration, you do not lose any performance by installing an additional 2 GB of RAM and enabling memory mirroring.

  Important: Because of memory mirroring, you only have half of the total amount of memory available. If 8 GB is installed, for example, then the operating system sees 4 GB once memory mirroring is enabled (it is disabled in the BIOS by default).

- Hot-add memory

  The x460 also supports the hot-add memory feature, which allows you to add DIMMs without stopping the server.

  Hot-add and hot-swap are mutually exclusive. You can only enable one of these features.

  Hot-add requires operating system support and currently only the Enterprise and Datacenter editions of Windows Server 2003 are supported. SQL Server 2005 also supports hot-add memory.

  There are restrictions as to what memory can be installed before hot-add is enabled. See the *x460 User's Guide* for more information.

- Memory ProteXion (redundant bit steering)

  Redundant bit steering (RBS) is the technical term for Memory ProteXion. RBS works in addition to ECC and provides an additional level of protection from memory failure.

  When a single bit in a memory DIMM fails, RBS automatically moves the affected bit to an unused bit in the memory array, which eliminates the need for ECC correction and returns the memory subsystem to peak performance.

  Although not recommended in a production environment, you can disable Memory ProteXion in BIOS by setting Memory Array to **High Performance Memory Array**. See 4.1.2, "Firmware and BIOS" on page 58 for details.

1.5 Multi-node configurations

XpandOnDemand scalability features provide the flexibility to expand the x460 server capacity in terms of number of CPUs, memory, and I/O slots as the demand grows.

Customers can expand the server as follows:

- CPUs from two-way up to 32-way
- Single-core processors can be replaced with dual-core processors
- Memory from 2 GB to 512 GB
- PCI-X slots from 6 to 48

This scalability is achieved by connecting multiple x460s or MXE-460s to the base x460. These nodes then form a single complex. The supported expansion steps are listed in Table 1-2.

An x460 server can be configured together with one, three, or seven MXE-460s to form a single eight-way, 16-way, or 32-way complex.

Table 1-2 x460 scalability options

Nodes	CPUs	Maximum RAM	PCI slots	Number of MXE-460s*
1	2-way	64 GB	6	None
1	4-way	64 GB	6	None
2	8-way	128	12	1
4	16-way	256 GB	24	3
8	32-way	512 GB	48	7
* Additional nodes can be either MXE-460 or x460 servers				

You can also form multi-node complexes using multiple x460s or combinations of x460s and MXE-460s. With these combinations, you can partition the complex as described in 1.5.3, “Partitioning” on page 13.

As shown in Figure 1-3, a scalable system consists of an x460 server and one, three, or seven MXE-460 systems.

A fully configured, eight-node, scalable system would have 32 processors, 512 GB of memory (using 4 GB DIMMs), 48 PCI-X 2.0 adapters, 3.5 TB of disk space (non-RAID), and 16 Gigabit Ethernet connections.

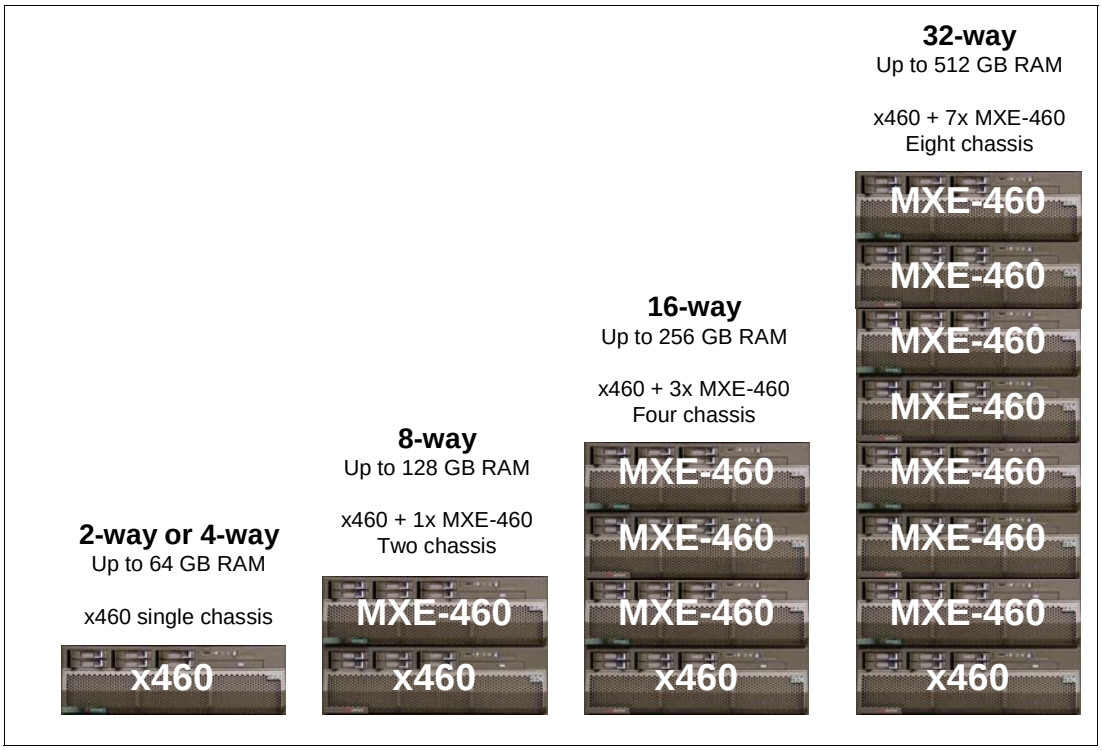


Figure 1-3 The four multi-node configurations supported

The multi-node configurations can have more than one x460. Only one is shown in Figure 1-3 for simplicity. In fact, if you wish to use partitioning as described in 1.5.3, “Partitioning” on page 13, you will need one x460 for each partition to be the primary node.

The first x460 is known as the primary node; all other nodes in a complex are called secondary nodes. The primary node must always be an x460 (MXE-460s cannot be primary nodes).

Only one, two, four, or eight nodes are supported. Other combinations cannot be selected in the configuration utility.

For performance reasons, you must have the same amount of memory in each node. A minimum of 2 GB of RAM is required in each node. The x460 and MXE-460 are technically capable of operating with less than 2 GB of RAM installed in a multi-node configuration, but this type of configuration requires Server Proven Opportunity Request for Evaluation (SPORE) approval before it can be supported.

In a multi-node configuration, 256 MB of RAM per node is allocated to the XceL4v cache. In an eight-node 32-way complex, this means that 2 GB of RAM is allocated to XceL4v and is not available to the operating system.

1.5.1 Positioning the x460 and the MXE-460

The MXE-460 is almost identical to the x460. The purpose of the MXE-460 is to serve as an expansion node for multi-node configurations. The MXE-460 is less expensive than the x460 and can be used as secondary nodes in partitions.

When building a multi-node configuration, you can use a combination of x460 and MXE-460 systems as the nodes. The number of x460s determines how you can partition the complex. For example, if you require an eight-node complex, you can do the following:

- If you configure one x460 and seven MXE-460s, then you can only create one partition—a 32-way partition.
- If you configure eight x460s and no MXE-460s, then you have the maximum amount of flexibility for making partitions—from a single 32-way partition, to eight 4-way partitions, and combinations in between.

Tip: For every partition you wish to create, the primary node in that partition must be an x460. Having an MXE-460 as a primary partition is not supported.

The technical differences between the x460 and the MXE-460 are as follows:

- Processors and memory are standard in the x460. The standard MXE-460 has neither. Customers must install matching CPUs and the appropriate amount of memory (you should also match the amount of memory installed).
- The x460 has a DVD-ROM as standard. The MXE-460 has no optical drive standard.
- Some components of the country kit are different. For example, the MXE-460 country kit does not include a ServerGuide™ CD.
- The MXE-460 is not supported as a primary node in any partition.

After you have cabled and configured a multi-node complex, you need to partition the complex. Most customers do not create more than one single partition in the complex. However, there are some advantages to creating multiple partitions:

- You can run different operating systems or versions on different partitions without the need for products such as VMware ESX Server.
- You can easily reconfigure the partitions if you must perform certain periodic processor-intensive or memory-intensive tasks, such as a weekly accounting or a business intelligence analysis task.

Figure 1-4 on page 12 shows the configuration alternatives for two-node and four-node complexes, and the partitioning options available to you as a result. Note that the eight-node configuration is not shown in the diagram, but the same rules apply.

If you want maximum partitioning flexibility, all your nodes must be x460s.

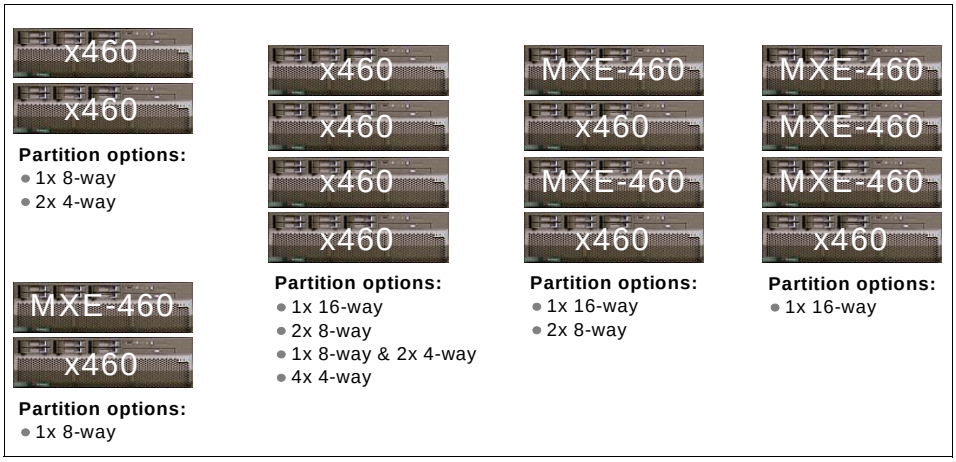


Figure 1-4 Supported partitioning for two-node and four-node complexes

The key rules for partitioning are as follows:

- Partitioning is always at node boundaries. You cannot form, for example, a partition using two processors in a node and two processors in another node.
- The primary node for every partition must be an x460. The use of MXE-460s limits the partitioning you can do.
- All nodes in a multi-node partition must have four processors installed.

1.5.2 Node connectivity

To create a multi-node complex, the servers and expansion modules are inter-connected using a high-speed data bus known as the *scalability bus*. The connection uses copper colored *scalability cables*.

Note: These cables are not compatible with the x440 and x445 equivalent cables.

There are three different cabling schemes, depending on the number of nodes used in the complex. These are shown in the following diagrams. In a two-node configuration, two scalability cables are used to connect both chassis. The second cable provides redundancy for the chassis interconnect as well as a slight performance benefit.

Figure 1-5 depicts the scalability cabling plan for a two-node/eight-way configuration:

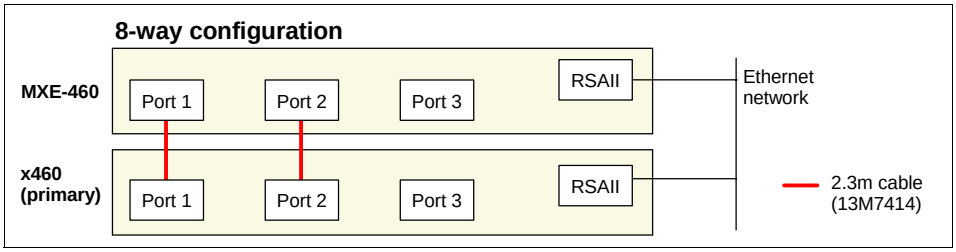


Figure 1-5 Cabling for a two-node configuration

Figure 1-6 on page 13 depicts the scalability cabling plan for a four-node, 16-way configuration.

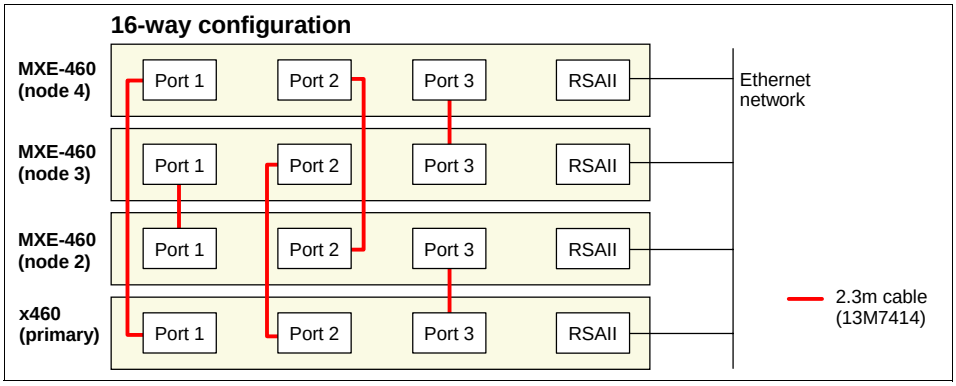


Figure 1-6 Cabling for a four-node configuration

Figure 1-7 depicts the scalability cabling plan for an eight-node, 32-way configuration.

In an eight-node configuration, there is one-hop and two-hop access to the memory in the destination nodes. For example, as seen in Figure 1-7, from node 1, node 4 can be reached through one scalability cable. This is one-hop access. However, node 8 cannot be reached from node 1 in 1 hop. Instead, two hops, with direction of node 1 → node 4 → node 8 or node 1 → node 5 → node 8, are necessary. This memory access is described in more detail in 3.1, “Scalable hardware implementation” on page 36.

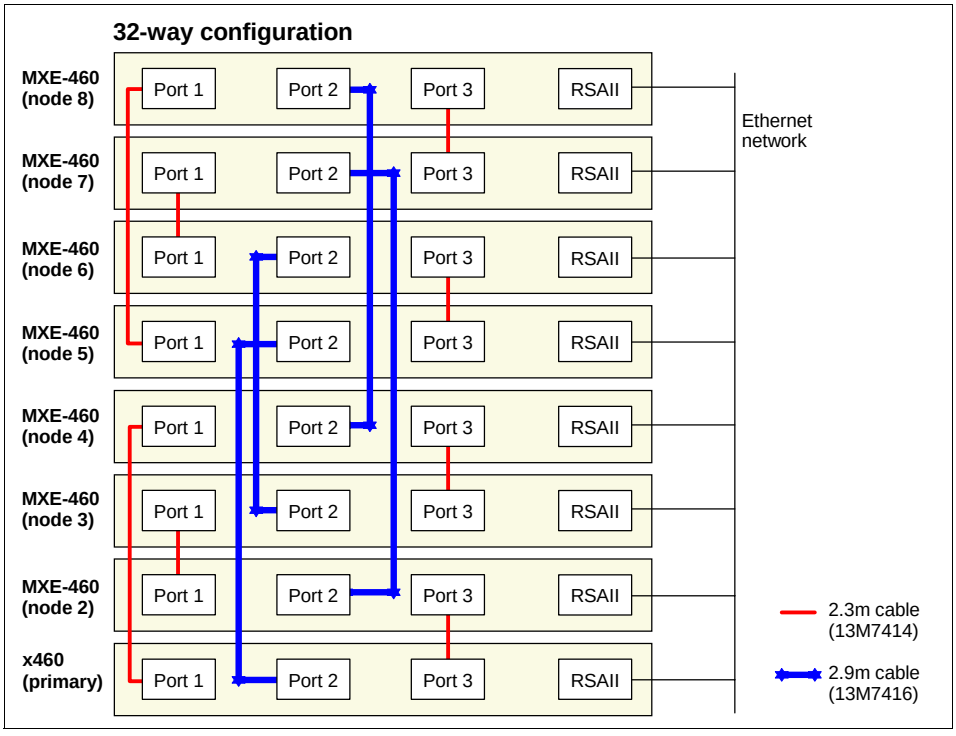


Figure 1-7 Cabling for an eight-node configuration

1.5.3 Partitioning

As discussed in 1.5, “Multi-node configurations” on page 9, the complex can be configured as one scalable partition with two, four, or eight nodes. Alternatively, you can divide this complex

into multiple independent partitions. For example, an eight-node configuration can be divided into two 4-node systems by changing the configuration without changes to the cabling.

The decision to partition must be made during the planning stage of a multi-node system, because the primary node in a multi-node complex must always be an x460. There is no support for configuring multiple partitions in a complex that consists of one x460 with one, three, or seven MXE 460s attached. You must have one x460 as the primary node for every partition you create (Figure 1-8).

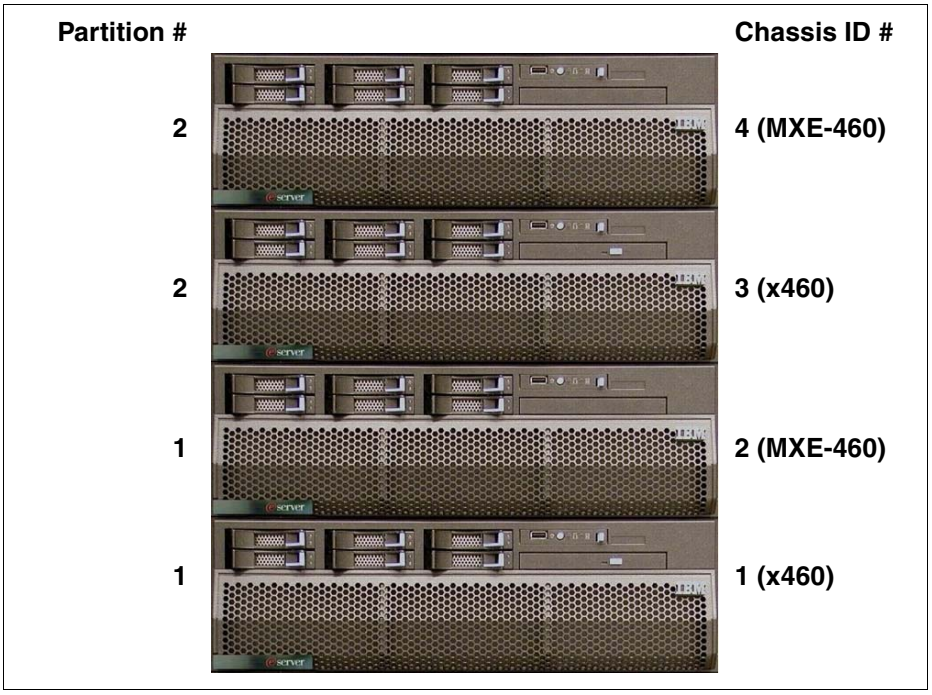


Figure 1-8 Four-node complex split into two partitions

1.6 64-bit with EM64T

Intel Extended Memory 64 Technology (EM64T) is an enhancement to the Intel IA32 architecture. With this enhancement, processors can run newly written 64-bit applications as well as existing 32-bit applications on Windows Server 2003, x64 edition at the same time. With Xeon processors with EM64T, applications can use a maximum of 16 TB virtual address space (VAS) on Microsoft Windows Server 2003, which far exceeds the 4 GB VAS limitation for 32-bit applications.

x460 based on the X3 Architecture with up to 32 Xeon MP processors and a maximum of 512 GB physical memory offers unparalleled performance and excellent scale-up capabilities.

1.6.1 What makes a 64-bit processor

On the subject of what makes a 64-bit processor, several factors are often discussed and sometimes confused:

- Operand size (integer or floating point)
- Register sizes
- Internal or external bus widths
- Physical or virtual address sizes

These factors are used at times to distinguish between processor size. For example, the 8086 and 8088 processors are almost identical, with the exception that the 8086 uses a 16-bit data bus while the 8088 uses an 8-bit data bus. The 8086 is considered a 16-bit computer, while the 8088 is considered an 8-bit computer.

The Xeon (Prestonia or Gallatin, not EM64T) is considered a 32-bit processor because it has 32-bit general purpose registers (GPRs) and a 32-bit VAS. Note, however, that the 32-bit Xeon also supports both 64-bit integer and floating-point operands, 128-bit internal data paths, 64-bit external data paths, and 36-bit physical addressing. Itanium® 2 also supports 64-bit GP registers, 82-bit floating-point registers, 64-bit virtual addresses, and 128-bit external data paths. So, why is it a 64-bit architecture? The operands, Prestonia, or Gallatin Xeon would not be considered 64-bit. The answer cannot be related to the data paths, because none of those is 64-bit. Physical addresses are 50 bits, not 64. The one thing of significance here is that 64 bits are the virtual addresses.

Definition of 64-bit: A 64-bit processor is a processor that is able to address 64 bits of VAS. A 64-bit processor can store data in 64-bit format and calculate arithmetic operations on 64-bit operands. In addition, a 64-bit processor has GPRs and arithmetic logical units (ALUs) that are 64 bits wide.

The x86-64/EM64T processors have 64-bit GP registers, and they support 64-bit integer and floating-point operations. So, from that perspective, they are true 64-bit. Data buses are all 64 bits wide or greater. So, from that standpoint, they too are true 64-bit. In addition, 64-bit virtual addresses are supported. (Actually, 64-bit addresses are allowed, but only the least significant 48 bits are used. The top 16 bits are required to be zero. It is a distinction that could be raised, but one that is unimportant for all practical purposes because 48-bit addresses are 256 TB.)

Instructions involving 64-bit addresses are given full support. The processor can operate in either 32-bit or 64-bit modes, and there are sufficient resources within the processor to support either mode. Both modes are supported for the sake of backwards compatibility, not because of any inherent restriction on 64-bit operation. EM64T is basically a 64-bit processor, but it is a processor that retains its compatibility with earlier generations of hardware.

This is one of the principle sources of confusion. The issue of *extensions*, a term that Intel uses, is about the instruction set and how compatible it is with existing software. The 32-bit instruction set is extended to provide greater flexibility while maintaining compatibility with existing software. The implementation of that instruction set is what makes the difference in the assignment of any sensible determination about what constitutes a true 64-bit architecture. The x86 cores were beefed up where necessary to ensure that there is full 64-bit support. So, in effect it is more accurate to describe them as 64-bit architectures that support backward compatibility modes.

A key difference between EM64T and Itanium is that the Itanium Explicitly Parallel Instruction Computing (EPIC) architecture does not support both 32-bit and 64-bit modes. In fact, it is less flexible than EM64T. The key to a successful and high performing Itanium solution is ensuring that the application will be able to take advantage of the highly parallel architecture, which typically requires a complete rewrite of the application.

Remember: AMD64 and EM64T architectures are not compatible with IA-64 (Itanium 2). This compatibility also means that application code is different for other 64-bit platforms and must be completely changed (either rewritten, ported, or recompiled) before running on any x86-64 machines.

The benefits of 64-bit computing on an x86-64 platform are tied mainly to the integer and memory addressing components of an application or OS. Sixty-four-bit computing does not speed up floating point workloads, logical workloads, and integer math unless most of calculations in the application use greater numbers (for example, indexes in the large database or mail systems). In 64-bit processors, default integer calculation mode is 32 bit, not 64 (because, when adding 1+1, there is no need to enter 64 bit).

1.6.2 Intel Xeon with EM64T

The Xeon processor is the server version of the Pentium® 4 (P4) family. There are two versions of the Xeon that are available currently. The Xeon DP supports two-way symmetric multiprocessing (SMP) natively, and the Xeon MP supports four-way SMP natively. The term *natively* means without additional logic. The IBM XA-32 and XA-64e chipsets provide additional logic that supports up to 32-way Xeon MP configurations.

The single-core versions of the Xeon MP are Cranford and Potomac. Potomac supports a 40-bit physical address space and Cranford supports 36-bit. With 40-bit physical addressing, the processor is capable of addressing up to 1 TB of real memory and 36-bit corresponds to 64 GB.

The width of a memory address dictates how much memory the processor can address. As shown in Table 1-3, a 32-bit processor can address up to 2^{32} bytes or 4 GB. A 64-bit processor can theoretically address up to 2^{64} bytes or 16 EB (or 16777216 TB).

Table 1-3 Relationship between address space and number of address bits

Bits (Notation)	Address space
8 (2^8)	256 bytes
16 (2^{16})	64 KB
32 (2^{32})	4 GB
64 (2^{64})	18 Exabytes (EB)

Current implementation limits are related to memory technology and economics. As a result, physical addressing limits for processors are less, as shown in Table 1-4.

Table 1-4 Memory addressability by current processors

Processor	Physical addressing
Intel Xeon MP Gallatin (32-bit)	4 GB (32-bit)
Intel EM64T Nocona (64-bit)	64 GB (36-bit)
Intel EM64T Cranford (64-bit)	1 TB (40-bit)
Intel EM64T Potomac (64-bit)	1 TB (40-bit)
Intel Itanium 2 (64-bit)	1 Petabyte (50-bit)
AMD Opteron processor (64-bit)	1 TB (40-bit)

These values are the limits imposed by the processors. Memory addressing can be limited further by the chipset or supporting hardware in the server. For example, the x460 Potomac-based server addresses up to 512 GB of memory in a 32-way configuration when using 4 GB DIMMs—a technology and physical space limitation.

A memory address is a unique identifier for a memory location where a processor or other device can store a piece of data for later retrieval. Each address identifies a single byte of storage.

All applications use *virtual* addresses, not physical. The operating system maps any (virtual) memory requests from applications into physical locations in RAM. When the total amount of virtual memory used by all applications combined exceeds the physical RAM installed, the difference is stored in the page file also managed by the operating system.

1.7 Dual core processors

The X3 Architecture chipset enables the x460 to take full advantage of the new Intel dual core Xeon processors. These new processors increase performance and the effectiveness of price and performance.

1.7.1 Why dual core processors

Traditionally, processor designers have increased the performance of their designs by increasing the core frequency that the chip runs at. For example, the Pentium III Xeon had a 550 MHz frequency, while the current Xeon MP runs at 3.3 GHz. Performance has also increased as clock speed increased.

Recently, however, increasing frequency has become difficult. To increase processor performance by enhancing frequency while maintaining present power consumption, manufacturers need to use smaller transistors. But the transistor sizes have become so small that the increase in processor frequency leads to huge power consumption.

The current solution is to implement the concept of a two-way system, but integrate the two processors, or cores, into one silicon die. This brings the benefits of two-way SMP with less power consumption and faster data throughput between the two cores. To keep power consumption down, the resulting core frequency is lower, but the additional processing capacity means an overall gain in performance.

Figure 1-9 on page 18 compares the basic building blocks of the Xeon MP single-core processor (Potomac) and dual-core processor (Paxville).

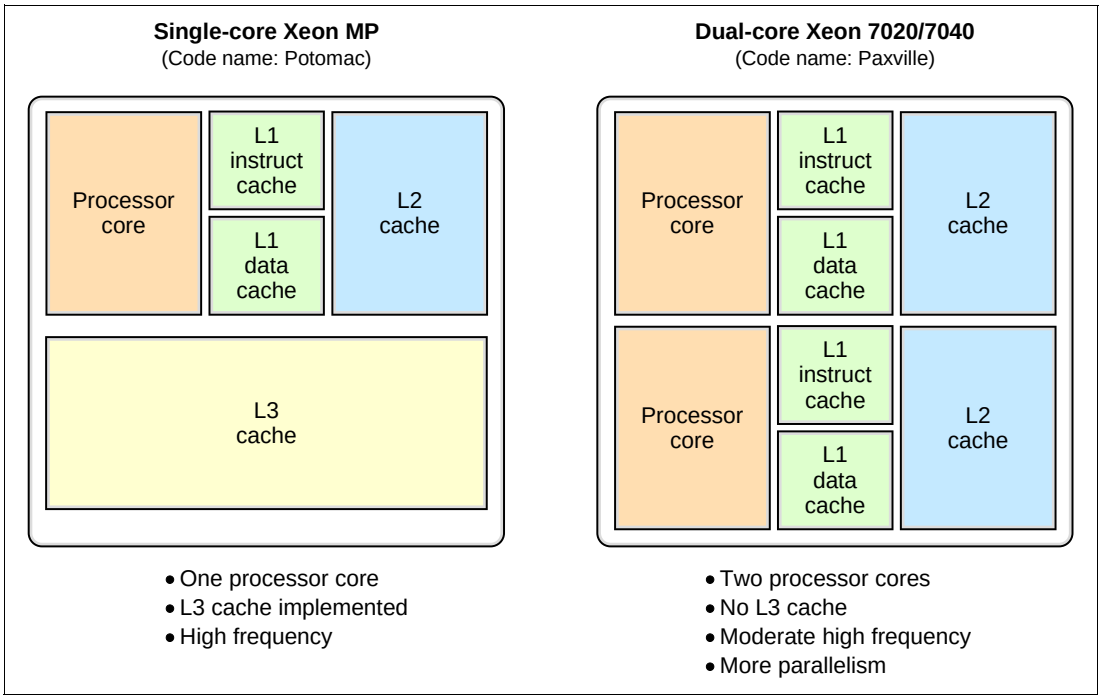


Figure 1-9 Feature of single core and dual core processors

In addition to the two cores, the dual-core processor has separate L1 instruction and data caches for each core and separate execution units (integer, floating point, and so on), registers, issue ports, and pipeline for each core. A dual core processor achieves more parallelism than Hyper-Threading Technology because these resources are not shared between the two cores. There is an estimated 1.2 to 1.5 times improvement when the dual core Xeon MP is compared to the current single-core Xeon MP.

With double the number of cores for the same number of sockets, it is even more important that the memory subsystem be able to meet the demand for data throughput. The 21 Gbps peak throughput of the memory subsystem in the X3 Architecture of the x460 means that the server is well suited to dual core processors.

1.7.2 Performance

Intel has compared their single core and dual core processors using a number of CPU benchmarks, as shown in Table 1-5. The data shows that the raw CPU performance improvement of dual core is in the order of 1.25 to 1.5 times the performance of the single-core processor.

Table 1-5 Performance improvement with dual core processor

Benchmark type	Single core	Dual core
SPECint_rate_base2000	100%	151%
Linpack	100%	139%
LS-Dyna	100%	134%
Star-CD	100%	131%
Fluent	100%	131%

Benchmark type	Single core	Dual core
Database	100%	131%
SPECfp_rate_base2000(est.)	100%	125%
Single core processor: Xeon processor 3.6 GHz with 2 MB L2 cache Dual core: Xeon processor 2.8 GHz with 4 MB		

In all the benchmarked types, dual core processor performance is superior to single core even though the processor frequency is slower. The reasons for this include:

- Dual core has twice as many execution resources (execution unit, register, and so on).
- The L2 cache of dual core is larger than that of single core.
- The benchmark application generates multiple threads.

For more details, see:

http://www.intel.com/products/processor/xeon/dcprodbrief.htm

1.7.3 Software licensing

Software licensing fees for dual core processors are often as important as performance. There are two ways to count the number of processors:

- Per socket: A four-way dual core server counts as 4.
- Per core: A four-way dual core server counts as 8.

Of the many vendors, Microsoft has stated they will charge their software license fees by socket. Visit the following Web site for details:

http://www.microsoft.com/licensing/highlights/multicore.mspx

1.7.4 Comparing with single core

Now that dual core processor models of the x460 are available, customers have more choices to make. When choosing between single core and dual core models, there are key points to consider.

The benefits of dual core over single core include:

- Hardware costs: For a given number of cores, fewer x460 nodes are needed. For example, a dual core processor-based system with 16 cores means two x460 servers, while a single core processor-based system with 16 cores means four x460 servers.
- Software costs: Because Microsoft licenses Windows and SQL Server 2005 by the socket, software license costs are lower for a dual core configuration.
- CPU performance per node: When comparing the CPU performance of a fully configured x460 node, dual core can offer up to a 50% performance improvement, depending on the application. For example, with a two-node x460, a dual core configuration with 16 cores will outperform a single core configuration with 8 cores in most applications.

  In addition, when comparing two configurations with the same number of cores, the dual-core configuration has half the number of x460 nodes, which means fewer scalability port connections and lower memory latency associated with those ports.

The benefits of single-core over dual-core include:

- Performance per core: For more server applications, two single core processors will outperform a single dual core processor. So, for a fixed number of cores, maximum performance will be higher with a single core system.
- Higher frequency: Single-core processors have a faster core frequency. So, for applications where this makes a difference in performance, single-core processors will be the better choice.
- Memory performance: Even though memory latency may be higher, the memory request queue will likely be shorter because the ratio of cores to memory controllers will be 4 to 1 instead of 8 of 1 with dual core. As described in 3.1, “Scalable hardware implementation” on page 36, the memory queue length is as important as memory latency when it comes to the performance of multi-node x460 configurations. Similarly, there is less front-side bus contention with fewer cores.

The key point to make, however, is that the customer workload is a very important part of the equation when determining whether a single core or comparable dual core system is better. Customers should carefully benchmark their applications to determine workload and choose a system.

2

SQL Server 2005

In this chapter, we discuss the enhancements and new features of the latest release of the Microsoft database management system, SQL Server 2005, and why the x460 is the perfect hardware platform for running it. Just as the x460 can run either 32-bit or 64-bit code, the SQL Server 2005 is available in both 32-bit and 64-bit editions. Likewise, as your requirements for increased memory and processors grow, the x460 and SQL Server 2005 together can take full advantage of these additional resources, with excellent scalability.

It has been five years since Microsoft has introduced a major release of SQL Server. This new release, SQL Server 2005, has enhancements in every dimension:

- A new user-mode operating system, called SQLOS (which we discuss in 3.4.1, “SQL Server Operating System” on page 41)
- Major enhancements to the Database Engine, Analysis Services, Integration Services (formerly Data Transformation Services) and Reporting Services
- A new client layer called SQL Client
- New tools for management and development
- New features for high availability such as Database Mirroring, Data Partitioning and support for hot-add memory

This chapter covers the following topics:

2.1 SQL Server 2005 editions

There are five editions of SQL Server 2005:

- SQL Server 2005 Enterprise Edition (32-bit and 64-bit)
- SQL Server 2005 Standard Edition (32-bit and 64-bit)
- SQL Server 2005 Workgroup Edition (32-bit only)
- SQL Server 2005 Express Edition (32-bit only)
- SQL Server 2005 Developer Edition (32-bit and 64-bit)

Greater use of server resources and product features is possible with these systems, starting from the Express Edition up through Workgroup, Standard, and the Enterprise Editions. The Developer Edition contains all the features of the Enterprise Edition, but is licensed for development and testing, not for production use.

The Enterprise Edition scales to the performance levels that are required to support the largest enterprise online transaction processing (OLTP), highly complex data analysis, data warehousing systems, and Web sites. Enterprise Edition has comprehensive business intelligence and analytical capabilities and high availability features such as failover clustering and database mirroring that allow it to handle the most mission-critical enterprise workloads.

Enterprise Edition is the most comprehensive edition of SQL Server 2005 and is ideal for the largest organizations and the most complex requirements. It is also available in a 120-day Evaluation Edition for the 32-bit or 64-bit platform.

For a detailed description of which features are supported in which edition, see *SQL Server 2005 Books Online* topic “Features Supported by the Editions of SQL Server 2005.” Official Microsoft SQL Server 2005 documentation is available from:

`http://www.microsoft.com/sql/2005`

2.2 New and enhanced features of SQL Server 2005

In this section, we provide an overview of each of the major enhancements in SQL Server 2005. We cover the high availability features in more detail in 2.5, “SQL Server 2005 high availability” on page 29.

2.2.1 Database Engine enhancements

The Database Engine introduces new programmability enhancements such as integration with the Microsoft .NET Framework (specifically, the Common Language Runtime component) and Transact-SQL enhancements, new XML functionality, and new data types. It also includes improvements to the scalability and availability of databases.

The new features provided by the Database Engine include:

- **Database mirroring**

  Database mirroring can be used to enhance the availability of SQL Server databases by providing fast failover and automatic client redirection to a secondary server. In contrast to failover clustering, database mirroring keeps two copies of the database, does not require specialized hardware, and is easier to set up and maintain.

  We cover this feature in more detail in 2.5.1, “Database mirroring” on page 29.

- **Data partitioning**

  Partitioning tables and indexes can provide the following benefits:

  – Large tables or indexes can be more manageable because of quick and efficient access to or management of data subsets, while maintaining the integrity of the overall collection.

  – Querying large tables or indexes is likely to be faster and more efficient on multiple CPU computers.

  We cover this new feature in more detail in 2.5.2, “Data partitioning” on page 30.

- **Hot-add memory**

  Additional physical memory can be installed in a running server, and SQL Server 2005 will recognize and use the additional memory immediately.

  We describe this new feature in more detail in 2.5.3, “Support for hot-add memory” on page 32.

- **Online restore**

  With SQL Server 2005, database administrators can perform a restore operation while the rest of the database remains online and available. Online restore improves the availability of SQL Server because only the data being restored is unavailable.

- **Online indexing operations**

  The online index option allows concurrent modifications (updates, deletes, and inserts) to the table or any associated indexes during index maintenance operations. For example, while a clustered index is being rebuilt, users can continue to make updates to the underlying data and perform queries against the data.

- **Fast recovery**

  A new faster recovery option improves the availability of SQL Server databases. Users can reconnect to a recovering database after the transaction log has been rolled forward. It is no longer necessary to wait for the rollback phase to complete.

- **Security enhancements**

  SQL Server 2005 includes security enhancements such as database encryption, secure default settings, password policy enforcement, fine grained permissions control, and an enhanced security model.

- **Dedicated administrator connection**

  SQL Server 2005 introduces a dedicated administrator connection (DAC) that administrators can use to access a running server even if the server is locked or otherwise unavailable. This capability allows administrators to troubleshoot problems on a server by executing diagnostic functions or Transact-SQL statements.

- **Snapshot isolation**

  SQL Server 2005 introduces a new “snapshot” isolation level that is intended to enhance concurrency for OLTP applications. In earlier versions of SQL Server, concurrency was based solely on locking, which can cause blocking and deadlocking problems for some applications. Snapshot isolation depends on enhancements to row versioning and is intended to improve performance by avoiding reader-writer blocking scenarios.

2.2.2 Analysis Services enhancements

Analysis Services introduces new management tools, an integrated development environment, and integration with the .NET Framework. Many new features extend the data mining and analysis capabilities of Analysis Services.

New or improved features provided by the Analysis Services include:

- New data mining algorithms
- Clustering support
- Key performance indicators (KPIs)
- Relational online analytical processing (ROLAP)
- Proactive caching
- Integration with Microsoft Office products

2.2.3 Additional enhancements and features

These enhancements expand on existing functionality, or in the case of Notification Services and Server Broker, provide new functionality.

- **Integration Services enhancements**

  Integration Services (formerly Data Transformation Services) introduces a new extensible architecture and a new designer that separates job flow from data flow and provides a rich set of control flow semantics. Integration Services also provides improvements to package management and deployment, along with many new packaged tasks and transformations. The new Maintenance Plan Wizard builds packages that you can customize with Integration Services.

- **Replication enhancements**

  Replication offers improvements in security, manageability, availability, programmability, mobility, scalability, and performance, such as:

  - A new Replication Monitor
  - The ability to make schema changes to published tables
  - Improved support for non-SQL Server Subscribers
  - Merge synchronization over the Web
  - Relaxed large data type restriction on Updatable Transactional Subscribers

- **Reporting Services enhancements**

  Reporting Services is a new server-based reporting platform that supports report authoring, distribution, management, and user access.

- **Tools and utilities enhancements**

  SQL Server 2005 introduces an integrated suite of management and development tools that improve the ease-of-use, manageability, and operations support for large scale SQL Server systems.

- **Data access interfaces enhancements**

  SQL Server 2005 supplies improvements in Microsoft Data Access Components (MDAC) and the .NET Frameworks SQL Client provider for greater ease-of-use, control, and productivity for developers of database applications.

- **Notification Services**

  Notification Services is a new platform for building highly-scaled applications that send and receive notifications. Notification Services can send timely, personalized messages to thousands or millions of subscribers using a wide variety of devices

- **Service Broker**

  Service Broker is a new technology for building database-intensive applications that are secure, reliable, and scalable. Service Broker provides message queues that the applications use to communicate requests and responses.

2.3 Windows, SQL Server, and 32-bit versus 64-bit

SQL Server 2005 is available in three versions: as a 32-bit version, EM64T 64-bit (x64), and Itanium 64-bit versions. However, the x460 only runs the 32-bit and x64 versions. Running a 64-bit version of SQL Server also requires a matching version of the operating system.

Table 2-1 shows the combinations of Windows Server 2003 and SQL Server that run on the x460.

Table 2-1 Valid combinations of Windows and SQL Server running on the x460

	WS 2003, 32-bit	WS2003, 64-bit Itanium	WS 2003, x64 (EM64T)
SQL 2000 32-bit	Supported	Not valid	Supported
SQL 2000 64-bit Itanium	Not valid	Not valid on the x460	Not valid
SQL 2005 32-bit	Supported	Not valid	Supported
SQL 2005 64-bit Itanium	Not valid	Not valid on the x460	Not valid
SQL 2005 x64 EM64T	Not valid	Not valid	Supported

In this section, we explore the three combinations of running 32-bit and EM64T 64-bit (x64) versions of both Windows Server 2003 and SQL Server 2005:

- Both 32-bit
- x64 Windows and 32-bit SQL Server
- x64 Windows and 64-bit SQL Server

2.3.1 Windows and SQL Server, both 32-bit

When you run 32-bit Windows Server 2003, you can run either 32-bit SQL Server 2000 or 32-bit SQL Server 2005. All processes run in 32-bit, so there is no 64-bit option for SQL Server in this case.

SQL Server 2000 runs as a user process, with the standard 4 GB of address space (see Figure 2-1 on page 26), normally divided into two:

- 2 GB for the kernel
- 2 GB for the user mode portion

With 4 GB of RAM, you can use the /3GB boot.ini switch to change the split to 1 GB for the kernel and 3 GB for the user portion of the VAS.

If you have more than 4 GB of physical memory installed, 32-bit SQL Server can use it as the database buffer pool. However, you must enable Physical Address Extension (PAE) in Windows (add the /PAE switch to boot.ini) and enable Address Windowing Extensions (AWE) in SQL Server (using **`sp_configure`**). All SQL Server memory objects, binary code, data buffers, database page headers (512 MB for 65 GB of database buffers), sort area, connections, stored procedure caches, open cursors (basically, everything but the unmapped cached database pages) must fit in the 2 GB user mode portion of the process address space. To access the database buffer pool pages above the 4 GB line, SQL Server must map them into the address space below the 4 GB line. Refer to Figure 2-1. This mapping incurs some performance overhead.

Note: For systems with more than 16 GB of RAM you cannot enable both /3GB and /PAE. See Section 9.11 of the IBM Redbook *Tuning IBM @server xSeries Servers for Performance*, SG24-5287.

With AWE on, SQL Server 2000 allocates its full “max server memory (MB)” amount and never releases memory until it is shut down.

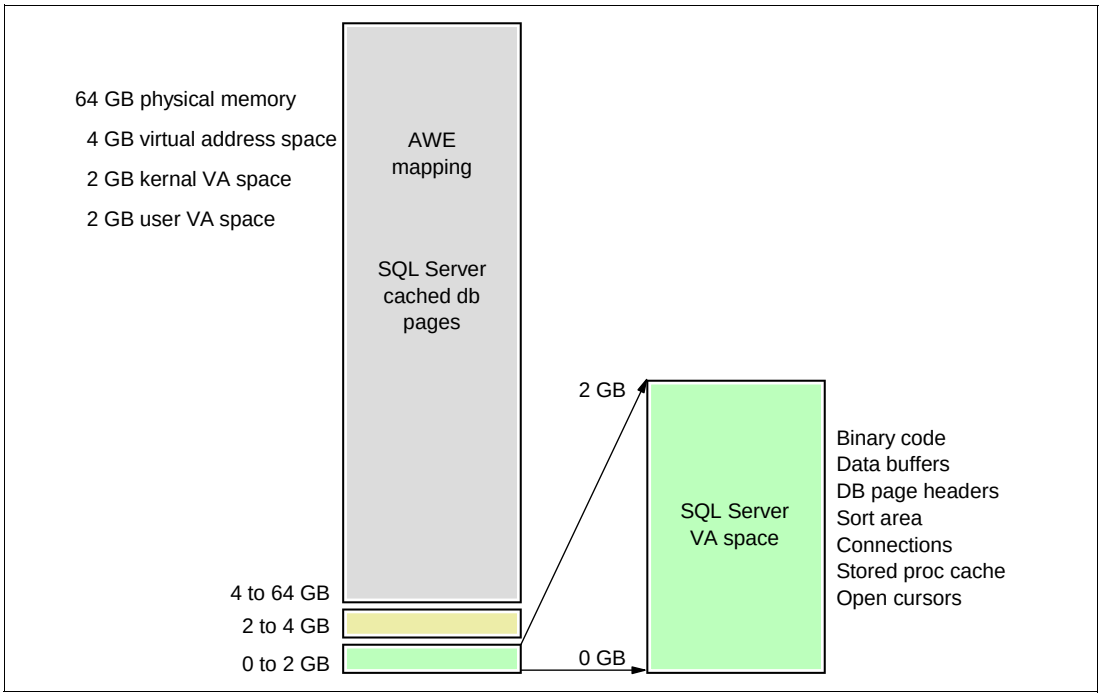


Figure 2-1 SQL Server 2000 uses AWE memory only for buffer pool pages

If your SQL Server workload is constrained by the 2 GB user mode limit for reasons other than requiring more database buffer pages, then it will not benefit from having more than 4 GB of physical memory. This is a hardware architectural bottleneck, which is relieved by using 64-bit hardware and software.

SQL Server 2005 32-bit experiences the same hardware architectural bottleneck. SQL Server 2005 manages all of its memory (including the AWE memory) dynamically, releasing and allocating memory in response to internal and external memory pressure.

2.3.2 Windows 64-bit and SQL Server 32-bit

When you have Windows Server 2003, x64 Edition installed, you can run either SQL Server 2000 with Service Pack 4 or 32-bit SQL Server 2005. (We discuss running SQL Server x64 in 2.3.3, “Windows and SQL Server 2005, both 64-bit” on page 27.)

SQL Server 2000 will run in WOW64 (Windows on Windows) in a user mode address space of 4 GB. Unlike running the same application on 32-bit Windows, SQL Server 2000 has a full 4 GB of user mode address space because the 64-bit kernel runs in its own address space. This provides some relief for SQL Server workloads that are constrained by the 2 GB user mode address space under a 32-bit operating system.

SQL Server 2000 can also be configured to use AWE, which allows it to access up to 64 GB of physical memory on EM64T processors. The AWE memory above 4 GB must still be mapped into the lower 4 GB user mode address space to be used.

SQL Server 2005 32-bit benefits in the same way that SQL Server 2000 does with a 4 GB user mode address space that is not shared with the operating system. With AWE enabled, SQL Server 2005 can access up to 64 GB of physical memory on the x460. This is because the maximum memory for a process running in the WOW64 on Intel EM64T is 64 GB. SQL Server 2005 will manage all of its memory (including the AWE memory) dynamically, releasing and allocating memory in response to internal and external memory pressure.

2.3.3 Windows and SQL Server 2005, both 64-bit

When you run Windows Server 2003 x64, you can (and should) install the x64 version of SQL Server 2005. There is no 64-bit version of SQL Server 2000 for use on EM64T-based servers.

SQL Server 2005 64-bit enjoys the same memory addressability as the 64-bit operating system. The 64-bit user mode address space is not limited to 4 GB, and it can use memory up to the operating system maximum for any purpose, not just for database buffers. AWE mapping is not required because the memory model is flat under 64-bit addressing. This provides the most efficient utilization of resources for SQL Server 2005.

See *Introducing Windows Server x64 on IBM @server xSeries Servers*, REDP-3982 for more information about 32-bit and 64-bit memory addressing with Windows Server 2003.

2.4 Windows Server 2003 editions

In this section, we compare the editions of Windows Server 2003 SP1, discuss the Datacenter edition, and the processor and memory limits.

2.4.1 Comparing Windows Server 2003 editions

Microsoft offers four editions of Windows Server 2003: Web, Standard, Enterprise, and Datacenter. Table 2-2 provides a comparison of the Standard, Enterprise, and Datacenter editions.

Table 2-2 Feature comparison of the Windows Server 2003 SP1 Editions

Features	32-bit editions			x64 (64-bit) editions		
	Standard	Enterprise	Datacenter	Standard	Enterprise	Datacenter
Maximum memory (GB)	4	64	128	32	1024	1024
Maximum CPU sockets	4	8	32	4	8	64
Maximum server cluster nodes	None	8	8	None	8	8
IBM Datacenter program	No	No	Yes	No	No	Yes
Hot-add memory	No	Yes	Yes	No	Yes	Yes
NUMA support	No	Yes	Yes	No	Yes	Yes

Windows Server 2003, Datacenter Edition can provide server clustering and scaling with respect to memory and processors that is beyond any other current or previous version of

Windows Server. The Datacenter Edition is also part of a program designed to provide the highest level of availability and support.

Although Enterprise Edition appears to be similar to Datacenter Edition in Table 2-2 on page 27, there is a distinct difference between the two in terms of the ability to scale processors. In addition, Datacenter has a complete High Availability Program that was designed around it.

2.4.2 Windows Datacenter models

IBM offers Microsoft Windows Server 2003, Datacenter Edition on the x460, either the 32-bit version or the 64-bit version.

There are two x460 Datacenter offerings:

- IBM Datacenter High Availability Program offering

  This end-to-end offering is a fully configured, certified, pre-installed system for customers who want to maintain a tightly controlled environment for maximum availability.

  The Datacenter High Availability Program delivers a complete system configuration that has been certified down to the I/O subsystem and device drivers. The certification process is a rigorous load test cycle that ensures that every major hardware and device driver has been tested together to provide maximum availability in the customer's environment. To maintain this high availability, the solution must be maintained as a certified configuration. This offering leverages the industry solution integration skills of IBM and Microsoft. It is the ideal solution for a customer who wants a solution environment based on best practices but does not have the IT staffing to perform the work.

- IBM Datacenter Scalable offering

  Customers who already have a well-managed IT infrastructure and simply want a Microsoft Windows operating system that scales greater than eight-way can choose this offering. With this solution, customers have more freedom to leverage their existing IT infrastructure.

  The IBM Datacenter Scalability offering delivers a certified server that has been through the same rigorous certification process as the IBM Datacenter High Availability Program offering, but gives the customer more freedom in choosing I/O and other system components. The IBM Datacenter Scalability offering provides a scalable Windows solution, so that customers can implement a solution that leverages their own IT staff, processes, and procedures.

2.4.3 Processor and memory limits

Windows Server 2003 x64 Edition supports a significant amount of physical memory. The numbers of processors that are supported has also increased in Windows Server 2003, Datacenter Edition. The amount of physical memory that is supported for the Enterprise and Datacenter editions is greater than the amount of memory that can be installed in servers today.

The maximum configuration of the x460 is 32-way (eight chassis with 32 processor sockets) and 512 GB of physical memory (using 4 GB DIMMs).

Windows and the x460 also support dual-core processors that have two physical processor cores on a single chip. The Microsoft licensing policy is based on the number of processors (that is, the number of sockets) and not the number of cores.

Using Intel Hyper-Threading Technology, a single physical processor core can execute multiple threads (instruction streams), making it appear to the operating system as two (logical) processors for each physical processor installed. Microsoft licensing is based on physical processors (sockets) and not the number of logical processors.

Therefore, if an x460 is fitted with four dual core Xeon processors and Hyper-Threading Technology is enabled, a license for Windows Server 2003, Standard Edition is sufficient even though the system contains eight processor cores and the Windows operating system sees 16 logical processors.

2.5 SQL Server 2005 high availability

Three new features, database mirroring, data partitioning and hot-add memory, support the objective of highly available SQL Server 2005 databases running on the x460.

2.5.1 Database mirroring

Database mirroring is a high availability feature in which two databases are paired and observed by a *witness*. One database, the *principal*, is active and processing user transactions. The other database, the *mirror*, is passive, receiving log updates from the principal, which the mirror applies to keep it in sync. If the principal were to fail for some reason, then, if the mirror and the witness are able to communicate over the network, they can agree to start the mirror as the active database, and the mirror becomes the principal.

Database mirroring is a new feature with SQL Server 2005. The benefits include the following:

- It is a software solution that does not require specialized hardware or two sets of identical hardware. Only standard servers connected over a network are required.
- The database that is being mirrored is not installed on shared storage. There are actually two copies of the database, so the storage is not a single point of failure.
- There is no distance limitation, because the mirroring is done over an Ethernet network.
- The failover is very quick, just a few seconds.
- It is much easier to set up and manage than failover clustering.
- Database mirroring can be used by itself, or in combination with failover clustering or log shipping.

Note: Database mirroring is disabled by default. At the time of writing, Microsoft stated that the feature is for evaluation purposes only, is not supported, and should not be used in production. See the Microsoft support Web site for the latest information about support for database mirroring.

The new SQL Client layer, which the applications use to communicate with SQL Server, participates in making the switch from principal to mirror invisible to the application. When the application first connects to the principal, SQL Client caches both the principal and the mirror names. This way, if the principal goes offline and the application tries to reconnect, the SQL Client layer will automatically and transparently try the mirror name instead. The end user still sees a drop and reconnect, like they would for failover clustering.

In database mirroring, the mirrored database is unavailable. However, using another new feature, called database snapshot, it is possible to create a point-in-time snapshot of the mirror database, and that snapshot database can then be used for reporting. Thus, an added

benefit to achieving high availability is that the reporting workload can be moved off of the principal onto the mirror.

Figure 2-2 on page 30 shows a sample database mirroring setup. Server A is acting as the principal for the mirrored database. Clients are connected to the SQL Server instance running on Server A, and the principal database is being updated.

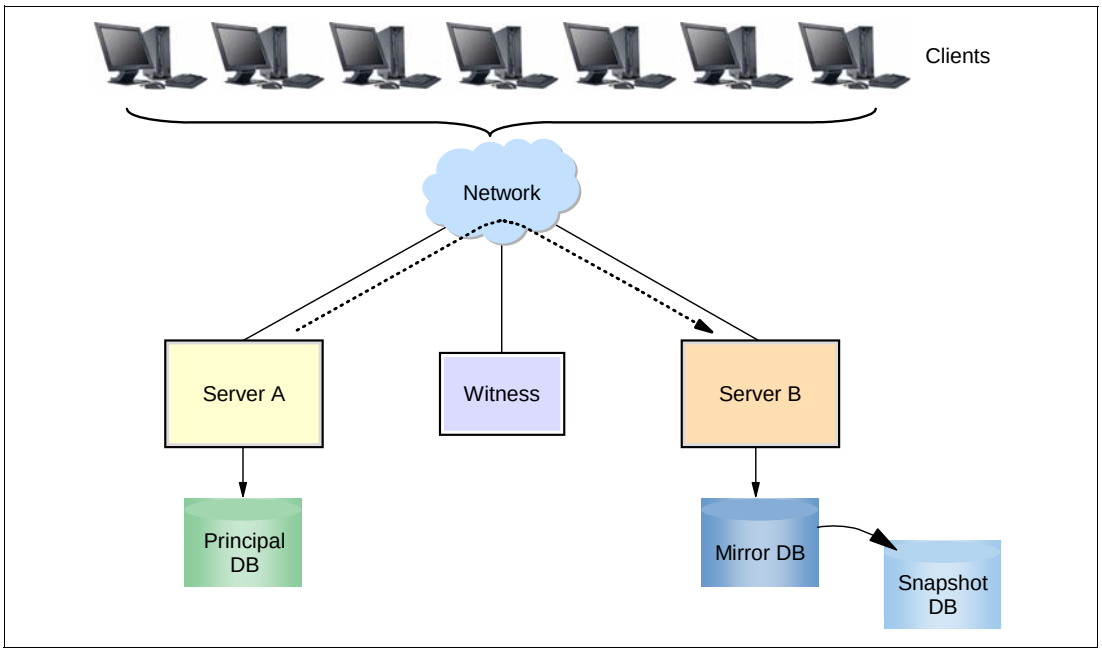


Figure 2-2 SQL Server 2005 database mirroring

Database mirroring is sending the transaction log changes (the dashed line in the figure) through the network to Server B, which is acting as the mirror. Server B applies those log changes to the mirror database to keep it in sync with the principal database. Clients have cached the name of the SQL Server instance acting as the mirror, Server B.

If a failure occurs on Server A, the Witness server and Server B, the mirror, contact each other, and agree to switch the role of Server B to begin acting as the principal. Clients meanwhile attempt to connect to the cached name of Server B. When they succeed, they are connected to Server B, which is now using the database copy that it has kept in sync through the mirroring process.

In addition, this example shows an embellishment of straight database mirroring, in that a snapshot has been taken of the mirror database. This allows clients to connect to Server B and access the snapshot database for purposes of reporting. The advantage of this is that the processing load for reporting has been transferred to Server B and does not affect the normal transactional processing on Server A.

Be aware, however, that it is not possible to back up a snapshot database. In the event of a media failure, it might not be possible to recreate the snapshot at the same point-in-time for reporting purposes.

2.5.2 Data partitioning

Data partitioning is a new feature with SQL Server 2005. It should not be confused with dynamic partitioned views, which is a different feature that was introduced in a previous

release of SQL Server. The query optimizer can access plans that skip partitions, if they are not needed, and that process partitions in parallel, which the x460 can exploit.

Data partitioning is designed to significantly increase the availability, to the user, of a very large table that exhibits the *sliding window* phenomena (for example, a table for which only the most recent 12 months are of interest). In this scenario, at the end of each month, the oldest month is removed from the table and the most recent month is added.

Without data partitioning, the process of *sliding the window*, for a very large table with millions of rows, is likely to cause an extended period of reduced availability for that table. This is because deleting rows for the oldest month and inserting the rows for the newest month is very disk intensive. In addition, any indexes on the table must also be updated and possibly rebuilt. Otherwise, the indexes might be unacceptably fragmented and lead to poorly performing queries.

With data partitioning, however, it is possible to build the large table in sections, called partitions, which permits old ones to be swapped out and new ones swapped in easily (that is, as meta data operations). This is possible if the indexes on the partitions have been *aligned* with the partitions.

Figure 2-3 shows the four conceptual steps required to remove the Jan 2005 data from the OrderYr table and insert the Jan 2006 data.

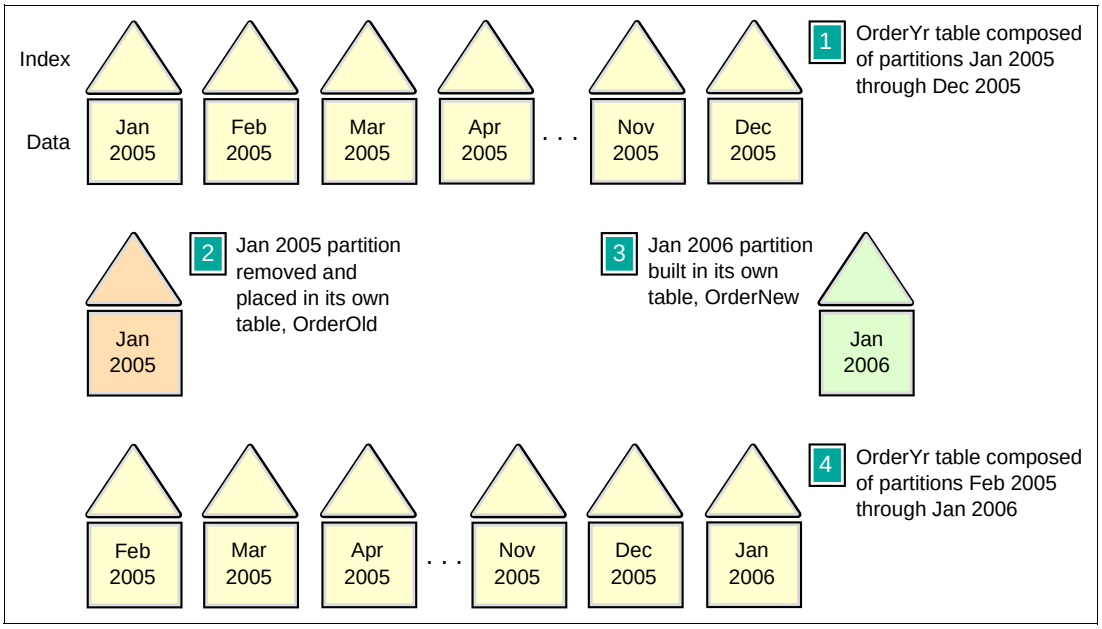


Figure 2-3 Data partitioning for sliding window data

The key points to observe are when actual data modification is taking place and when only metadata is changing. Figure 2-3 shows four steps:

1. In step 1, we define the OrderYr table to consist of the 12 partitions (Jan 2005 through Dec 2005), and the indexes are aligned with the partitions. This is our starting point.
2. In step 2, we modify only the meta data for the OrderYr table to exclude the Jan 2005 partition and place it in its own table: OrderOld. This happens instantly and has practically no impact on availability.
3. In step 3, we build the Jan 2006 partition in a staging table called OrderNew. This actual data modification can take a long time and impact the disks storing the OrderNew table, but it does not affect the OrderYr table. So, again availability is not affected.

4. In step 4, we modify the metadata for the OrderYr table to include the Jan 2006 partition. Again, this happens instantly and has practically no impact on availability.

So, the entire operation of sliding the window does not require any downtime for users. They can access the OrderYr table, with normal availability, even while they are removing and adding millions of rows.

2.5.3 Support for hot-add memory

Releases of SQL Server before SQL Server 2005 supported dynamic memory, which allowed SQL Server to automatically adjust memory usage when there was spare memory on the system. However, SQL Server was limited by the amount of memory available at startup.

With SQL Server 2005, this limit on startup memory availability is removed. SQL Server now supports hot-add memory in Windows Server 2003, which allows users to add physical memory without restarting the server, if the server hardware supports it (the x460 does).

Hot-add memory is only available for 64-bit SQL Server, and for 32-bit SQL Server when AWE is enabled. Hot-add memory is not available for 32-bit SQL Server when AWE is not enabled. Hot-add memory is only available for Windows Server 2003, Enterprise and Datacenter Editions. It also requires special hardware that is provided by the x460.

For example, suppose you are running SQL Server 2005 and Windows Server 2003, Enterprise Edition 32-bit on a computer with 16 GB of physical memory. The 32-bit operating system is configured to limit applications to 2 GB of virtual memory address space; AWE has been activated on SQL Server and the -h switch enabled during startup. To increase server performance, you add another 16 GB of memory. SQL Server recognizes the additional memory immediately, and begins to use it as necessary, without a restart of the server.

Note: Removing physical memory from the system still requires restarting the server.

2.6 Customer proof of concept

IBM, Microsoft, and a mutual customer recently conducted a large data warehouse scalability proof of concept (POC) with a large x460 complex and SQL Server 2005. The POC used the customer's workload and showed excellent scaling results.

2.6.1 Hardware configuration

The x460 was cabled as an eight-node system and we used the Remote Supervisor Adapter (RSA) II Web Interface to configure the complex as a single-node (four-way), two-node (eight-way), four-node (16-way) and eight-node (32-way) server. Each node had four 3.3 GHz Xeon MP processors with 32 GB of physical memory for a total of 32 processors and 256 GB of memory.

We installed four Fibre Channel (FC) host bus adapters (HBAs), two in the first node and two in the second node. The x460 was connected by two Silkworm Brocade FC switches to an IBM DS4500 Storage Array containing 8 TB of storage. We created 16 RAID-5 arrays of seven drives each, using a total of 112 drives. There were four 4-way 32-bit clients.

Figure 2-4 on page 33 illustrates this hardware configuration.

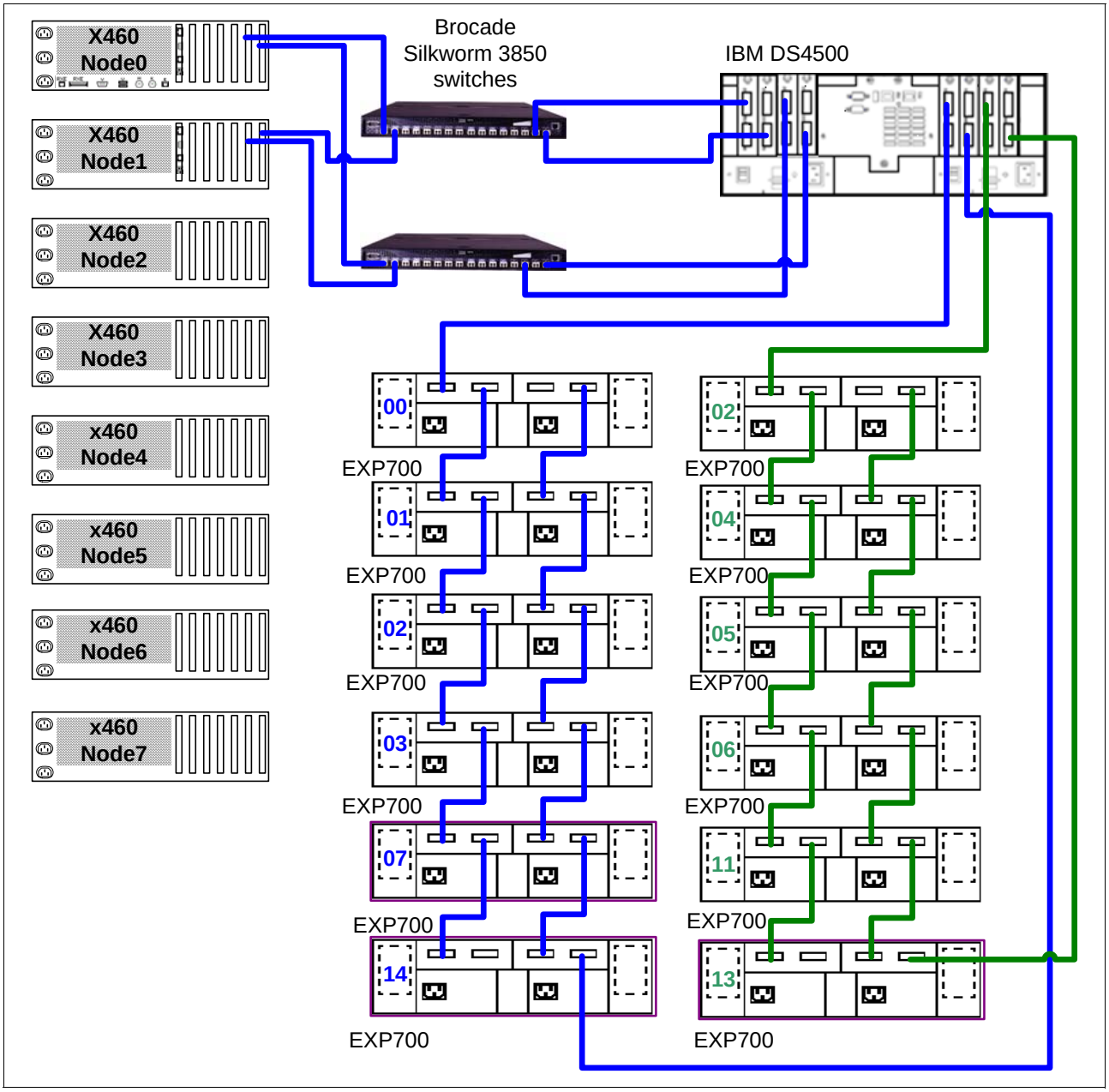


Figure 2-4 Large data warehouse configuration

2.6.2 Software and storage configuration

We installed Windows Server 2003 Datacenter x64 Edition and a pre-GA build of SQL Server 2005 on the x460. We used default settings for Windows and SQL Server, except for tempdb. Tempdb was moved from the default local storage location, placed on the SAN, and split into multiple files.

The 16 arrays were divided into two sets of eight arrays each. Disk volumes were created with software striping that used one or the other of the array sets. A special disk volume was created for tempdb, which was a stripe using all 16 arrays.

Figure 2-5 on page 34 shows two user databases and Tempdb. Database 1 is composed of Data_1 and Log_1. Database 2 is composed of Data_2 and Log_2.

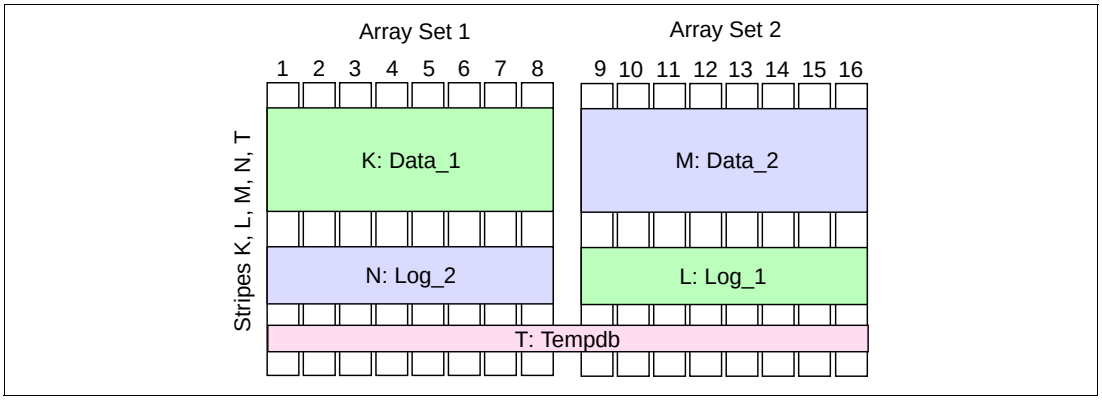


Figure 2-5 POC storage layout

When Database 1 is in use, it has access to all of the disk spindles. Likewise for Database 2. When Tempdb is needed, it uses all the disk spindles. When Database 1 is being loaded, the data and log files are on separate sets of spindles, so that there is no disk contention. There is some contention, however, when both databases are being updated, and Tempdb always contends with either database. This is a satisfactory trade off for this customer. An attractive feature of this layout is that it has a pattern that is repeatable, regardless of the number of databases or size of the storage system, and still provides excellent performance.

2.6.3 Results

The scalability tests tested the customer's DSS (complex query) workload on the configuration from four-way up to 32-way. This workload fit completely in memory, so it was extremely CPU intensive and the I/O rate was low. Figure 2-6 shows the results of the tests.

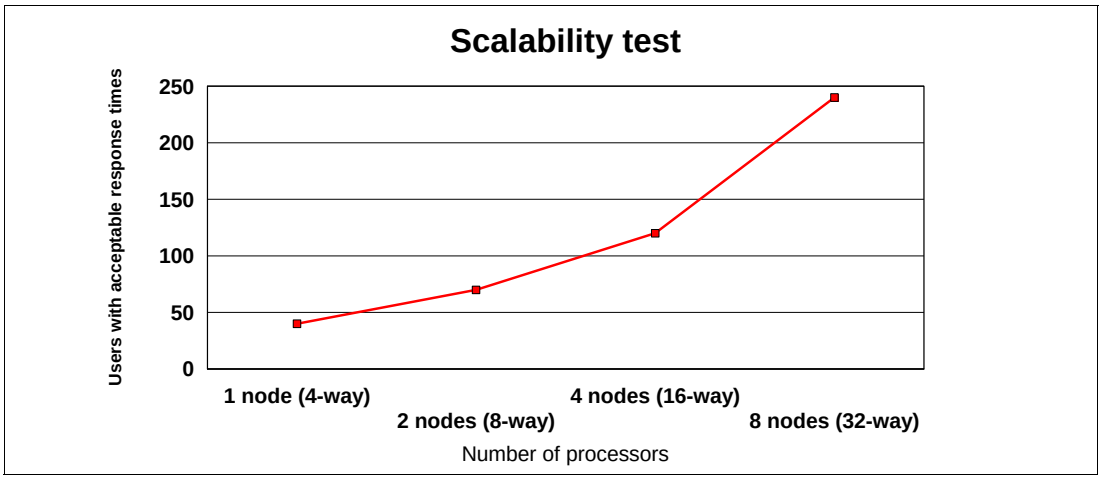


Figure 2-6 POC scalability tests

Scaling from 4 to 8 CPUs was 1.8, from 8 to 16 CPUs was 1.7, and from 16 to 32 CPUs was a 2.0. When you double CPUs and memory, as we did in these tests, the best possible scaling is 2.0. Normally this does not happen; 1.7 is considered excellent for this type of test. A possible explanation for the 2.0 scaling was a change in the behavior of SQL Server 2005 when running on larger configurations. At greater than 16 CPUs, SQL Server 2005 enables *superlatches*, which provides greater efficiency.

This POC demonstrated the excellent out-of-the-box scalability of the x460 and SQL Server 2005 without using NUMA tuning parameters.

3

Scalability and affinity

Both the x460 and SQL Server 2005 scale very well to 32 processors and 512 GB of RAM. This type of scalability, where all system resources across the multi-node complex (processors, memory, and PCI resources) are seen as one unified hardware platform for the operating system and applications, is known as scale-up.

The key to effective scalability is *affinity*, that is, linking system resources that are in the same node to maximize performance. For example, for the best performance, any memory operations that a processor executes should involve RAM that is installed in the same node as the processor. Fetching memory from another node can impact performance.

This chapter describes scalability and affinity in more detail. It also explains how scalability and performance can become complicated in a server environment with hundreds, if not thousands, of supported users. In a high-end server, such as the x460, many workloads occur simultaneously on the server. Thus, in this environment, it is important to distribute server resources optimally. When affinity is not configured properly, contention of resources might occur, which often leads to serious performance degradation.

There are a number of parameters in SQL Server and Windows Server 2003 that are related to scalability and affinity. When these Microsoft products are combined with the x460, the solution demonstrates high scalability.

This chapter covers the following topics:

3.1 Scalable hardware implementation

In the standard single chassis four-way configuration, the x460 acts as an industry standard SMP system. Each processor has equal access to all system resources.

In an SMP environment, the concentration of memory access to the memory controller is an obstacle for scalability. The memory controller is a core component that manages I/O to and from memory. With SMP configurations, when the processors are added, the number of memory controllers does not change. As the number of transactions increases, so do requests to the memory controller, which causes a significant bottleneck. As a result, SMP is less efficient the more processors you have.

However, with multi-node x460 configurations, a NUMA-like architecture (non-uniform memory architecture) is implemented by connecting the scalability ports of each node together (see Figure 1-1 on page 5). These ports are directly connected to the memory controller in each node and allow high speed communication between processors located in different nodes. The ports act as though they were hardware extensions to the CPU local buses. They direct read and write requests to the appropriate memory or I/O resources, and they also maintain cache coherency between the processors.

The term *NUMA* is not completely correct because not only can memory be accessed in a non-uniform manner, but also I/O resources. PCI-X and USB devices might be associated with nodes. The exception to this are I/O devices such as diskette and CD-ROM drives that are disabled because the classic PC architecture precludes multiple copies of these items.

The key to effective scalability is to add memory controllers as you add nodes. With the eight-way x460, there are two memory controllers. The 16-way has four memory controllers, and the 32-way has eight memory controllers.

In multi-node x460 configurations, the physical memory in each node is combined to form a single coherent physical address space. The result is a system where, for any given region of physical memory, some processors are closer to it than other processors. Conversely, for any processor, some memory is considered local and other memory is remote.

Memory can be described in one of three ways, depending on the relation to a given processor:

- *Local memory*: Memory is in the same node as the processor.
- *Remote memory*: Memory is installed in another node that is directly connected by scalability cables.

  Figure 3-1 on page 37 shows an example of a local memory access (CPU 1) and a remote memory access (CPU 6).

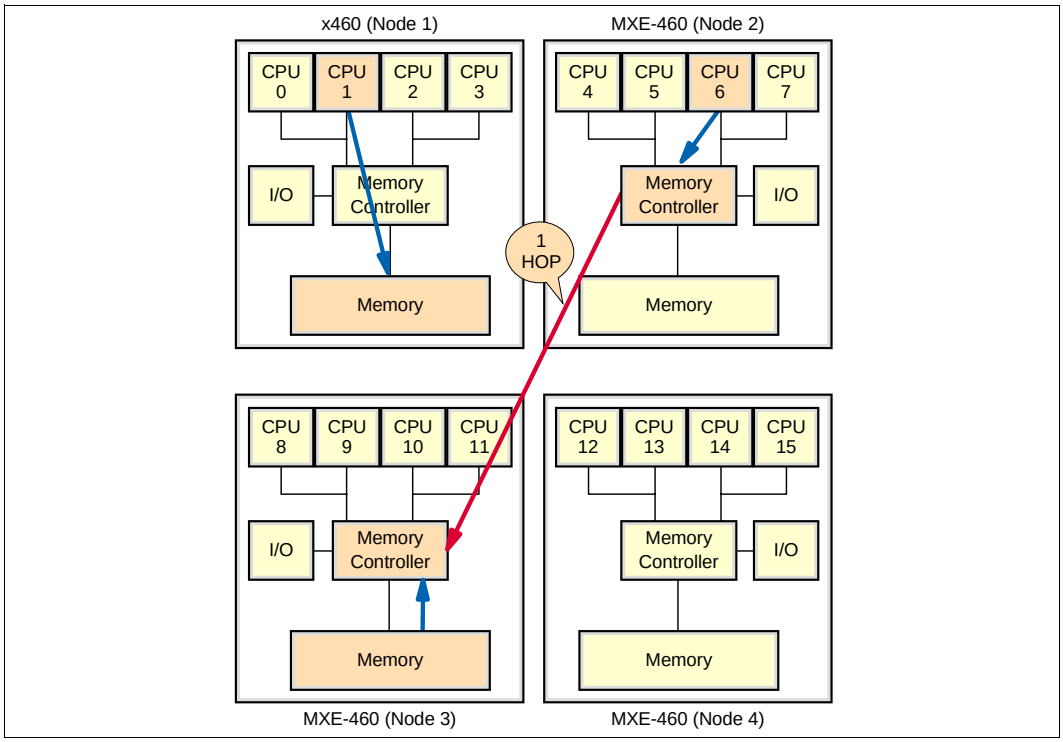


Figure 3-1 Local memory access and remote memory access

- *Far memory*: In eight-node (32-way) configurations, some nodes are not directly connected together (CPU 0 as shown in Figure 3-2, for example). In such situations, memory that is two “hops” (two interconnect cables) away from a given processor is “far.”

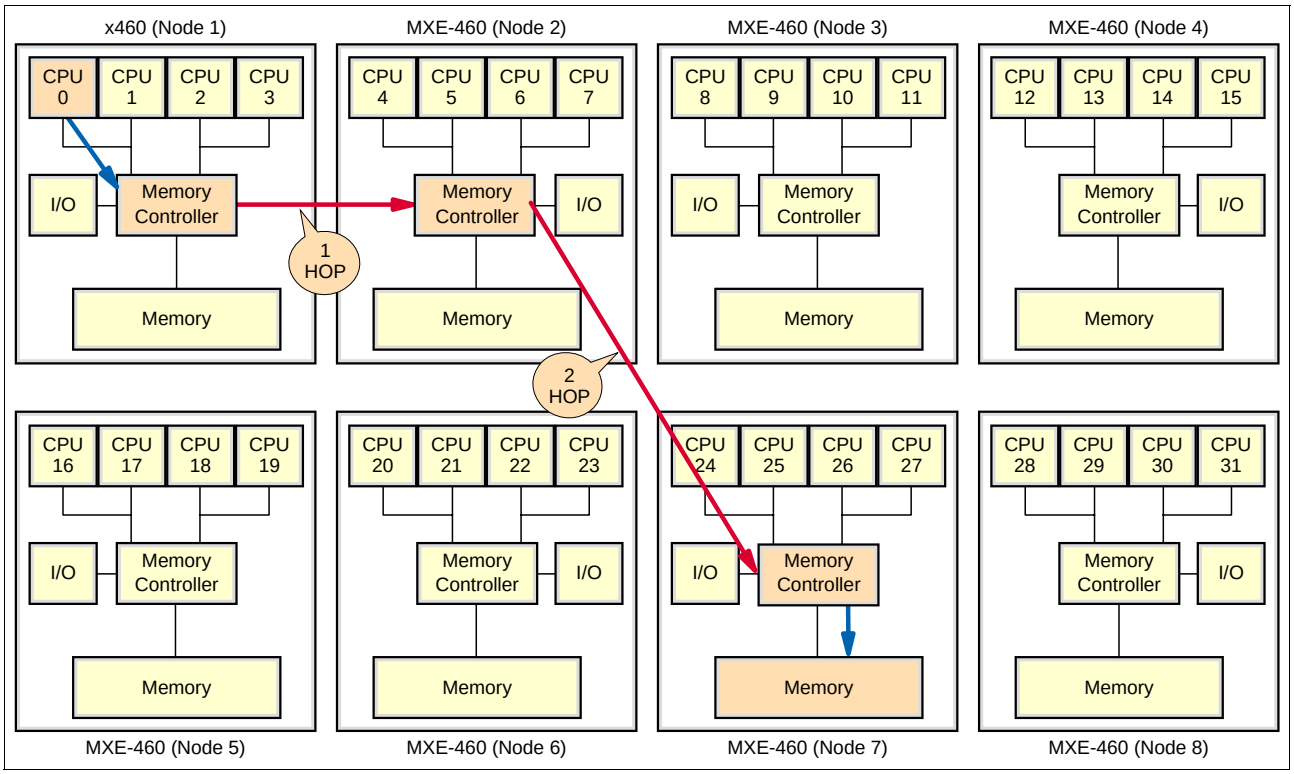


Figure 3-2 Far memory access

There are three important aspects that are associated with memory subsystem performance in this type of configuration: *memory latency*, *queuing time*, and *snooping*.

- Memory latency

  Remote and far memory access takes more time than local memory access of the travel time through the interconnects (one interconnect for a remote memory access, two for a far access). The delay incurred as a result of this long path is called *latency*. Remote and far memory access can decrease performance when compared with local memory access. So, to increase the performance of servers in a NUMA environment, the operating system and application must strive to limit the use of remote and far memory as much as possible. This concept is known as *affinity*.

  Affinity in the x460 means that local resources (CPU, memory, PCI devices) are grouped together and that the operating system recognizes these groups and attempts to limit the resources used by a thread to just those in a group.

  However, while affinity is important, this focus on latency misses the actual problem that NUMA is attempting to solve: shorten memory request queues.

- Memory queuing time

  Latency is only part of the picture, but is often the aspect that receives the most focus. As with a large SMP-based design, the more processors that access the same memory controller and memory, the greater the potential for poor performance. The bottleneck then becomes the number of access operations in the queue to be actioned.

  Another way to think about this is to consider the analogy of shopping at a local supermarket. Directly in front of you is a check-out lane with 10 customers standing in line but 20 meters to your left is another check-out lane with only two customers standing in line. Which would you go to? The check-out lane closest to your position has the lowest latency because you do not have far to travel. But the check-out lane 20 meters away has much greater latency because you have to walk that far.

  Clearly most people would walk the 20 meters, incurring a delay, to arrive at a check-out lane with only two customers instead of 10. We think this way because our experience tells us that the time waiting to check-out with 10 people ahead of us (the request queue) is far longer than the time needed to walk to the “remote” check-out lane (latency) and wait for only two people ahead.

  This analogy clearly communicates the performance effects of queuing time versus latency. In a computer server, with many concurrent outstanding memory requests, we would gladly incur some additional latency (walking) to spread memory transactions (check-out process) across multiple memory controllers (check-out lanes) because this greatly improves performance by reducing the queuing time.

- Snooping

  In a traditional SMP design, before a piece of data is retrieved from memory, all caches that are local to every processor must be queried to determine whether that data is stored in a cache and whether it has been modified by another process running in that processor. For a small SMP, this *snooping* procedure (called the MESI protocol, for the four states that data can be in: modified, exclusive, shared, invalid) is sufficiently efficient. However, as you add more processors, the overhead becomes overwhelming.

  The solution implemented in the x460 is a special fourth level of cache, the XceL4v cache, that maintains a table of all contents and eliminates the need to query every processor. In addition, for multi-node x460 configurations, the XceL4v acts as a cache to reduce latency across the scalability cables.

It is important to note that the traffic across the scalability cables is not only memory access. As you can see in Figure 3-1 on page 37, each node also has local PCI-X slots that can have

devices such as FC adapters, Ethernet controllers, and the like. The operating system and application must maximize overall performance for these devices also.

3.2 Static Resource Affinity Table

The Static Resource Affinity Table (SRAT) contains topology information for all the processors and memory in a system. The x460 implements the SRAT table in firmware. The topology information includes the number of nodes in the system and which memory is local to each processor. By using this function, the NUMA topology of the x460 is recognized by the operating system. The SRAT table also includes hot-add memory information. Hot-add memory is the memory that can be hot-added while the system is running, without requiring a reboot.

The Advanced Configuration and Power Interface (ACPI) 2.0 specification introduces the concept of *proximity domains* in a system. Resources, including processors, memory, and PCI adapters in a system, are tightly coupled, and the operating system can use this information to determine the best resource allocation and the scheduling of threads throughout the system. The SRAT table is based on this ACPI specification.

You can find more about the SRAT table at:

http://www.microsoft.com/whdc/system/CEC/SRAT.mspx

3.3 Affinity in Windows Server 2003

This section discusses how process, scheduling, and memory management are optimized for the x460 scalable architecture and Windows Server 2003. It also describes the relevant parameters in Windows Server 2003.

3.3.1 NUMA optimization for Windows Server 2003

Most editions of Windows Server 2003 are optimized for NUMA as listed in Table 3-1. In general, when we discuss Windows Server 2003 in this section, we mean Windows Server 2003 editions that are optimized for NUMA.

Windows Server 2003 obtains the NUMA information from the SRAT table in the system BIOS while booting. That is, NUMA architecture servers must have the SRAT table and the x460 to use this function. Windows Server 2003 cannot recognize system topology without the SRAT table.

Table 3-1 Versions of Windows Server optimized for NUMA

	x86 (32-bit)	x64 (64-bit)	IA64 (64-bit)
Windows 2003 Web Edition	No	Not applicable	Not applicable
Windows Server 2003 Standard Edition	No	No	Not applicable
Windows Server 2003 Enterprise Edition	Yes	Yes	Yes
Windows Server 2003 Datacenter Edition	Yes	Yes	Yes

3.3.2 Process and thread scheduling

Process scheduling is optimized in the NUMA environment. When a new process is created, Windows Server 2003 uses a round-robin algorithm to assign it to the next NUMA node in the system. Each process has a local node and default processor that belong to it.

The thread scheduler for Windows Server 2003 is also optimized for NUMA. The operating system schedules the ideal processor based on priority, or, if that processor is not available, it schedules the thread to another processor in the local node. If all processors in the local node are unavailable, the operating system schedules the thread to processors in another node. This is sometimes referred to as *soft processor affinity* to contrast with the hard processor affinity you can configure with Windows System Resource Manager (WSRM) or within SQL Server 2005. Thus, the sequence of scheduling is as follows:

1. The ideal processor for the thread, which is selected when the thread is created
2. Any real processor that is in the same node as the ideal processor for the thread
3. Any logical processor that is in the same node as the ideal processor for the thread

If this prioritization fails, then Windows Server 2003 only considers other processor nodes if the average processor utilization of the other nodes is lower than the average utilization of the ideal node. Note, however, that Windows Server 2003 does not differentiate between remote and far resources (as defined in 3.1, “Scalable hardware implementation” on page 36).

You can prevent Windows Server 2003 from scheduling certain processors (for example, non-local processors) by using *hard processor affinity*. Hard affinity is the function of binding specific CPUs to a process and is available with the following tools:

- Affinity option in applications such as SQL Server 2005

  In SQL Server 2005, you can choose which processors can be used by the product by changing the affinity mask. By default, SQL Server 2005 can use all processors in the system.

  See 3.4, “Affinity in SQL Server 2005” on page 41 for more detail.

- WSRM

  WSRM is a tool that provides resource management of processors and memory resources. With WSRM, you can specify which CPUs you want each process to run on.

  You can install WSRM from the Windows Server 2003 CD-ROM. WSRM is included with Windows Server 2003 Enterprise Edition and Datacenter Edition and is covered by the license for these products. Alternatively, you can download it from:

  `http://www.microsoft.com/windowsserver2003/downloads/wsrm.mspx`

  In both cases, the editions of Windows Server 2003 that can install WSRM are listed in Table 3-2.

Table 3-2 Windows Server editions that include WSRM

	x86 (32-bit)	x64 (64-bit)	IA64 (64-bit)
Windows 2003 Web Edition	No	No	No
Windows Server 2003 Standard Edition	No	No	No
Windows Server 2003 Enterprise Edition	Yes	Yes	Yes
Windows Server 2003 Datacenter Edition	Yes	Yes	Yes
Windows 2000 Family	No	No	No

Normally you should not use WSRM on a system that is running SQL Server 2005 in a production environment because the two schedulers (WSRM and the CPU Affinity Mask in SQL Server) might cause conflicts and system degradation. However, there are situations where you might want to use both. For example, if you wish to share the x460 between SQL Server and Internet Information Server (IIS), then you could use WSRM to restrict IIS to only processors 0 and 1, for example, and configure SQL Server to be restricted to just processors 2 through 7. It is important to ensure that no processors are available to both schedulers because this causes contention problems.

3.4 Affinity in SQL Server 2005

This section describes the affinity options with SQL Server 2005 and NUMA optimization for SQL Server 2005.

3.4.1 SQL Server Operating System

New to SQL Server 2005 is a layer between SQL Server and the operating system called SQL Server Operating System (SQLOS). SQLOS is a user-level, highly configurable operating system with a powerful API that enables automatic locality and advanced parallelism.

SQLOS attempts to hide the complexity of the underlying hardware from high-level programmers. It also provides a comprehensive set of features to programmers who are willing to take advantage of the hardware underneath the system. SQLOS services include non-preemptive scheduling, memory management, deadlock detection, exception handling, hosting for external components such as CLR, and other services.

SQLOS has a hierarchical architecture, which means that SQLOS changes its structure based on the hardware platform that SQL Server 2005 is running on. Figure 3-3 illustrates SQLOS on an SMP-based server.

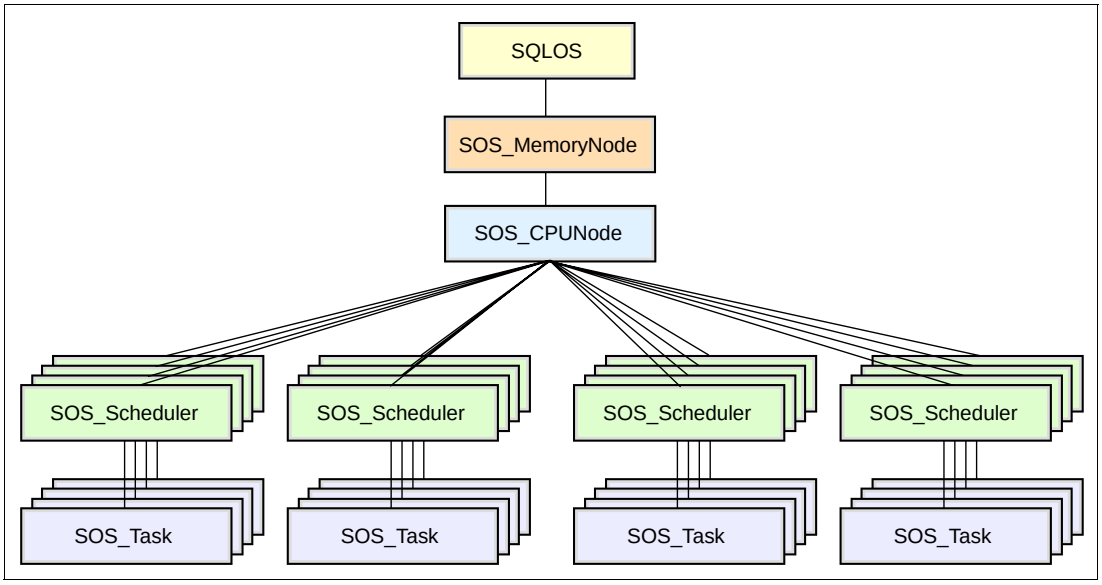


Figure 3-3 SQLOS for an SMP server (eight-way)

Figure 3-4 on page 42 illustrates SQLOS on a two-node NUMA server such as a two-node (8-way) x460.

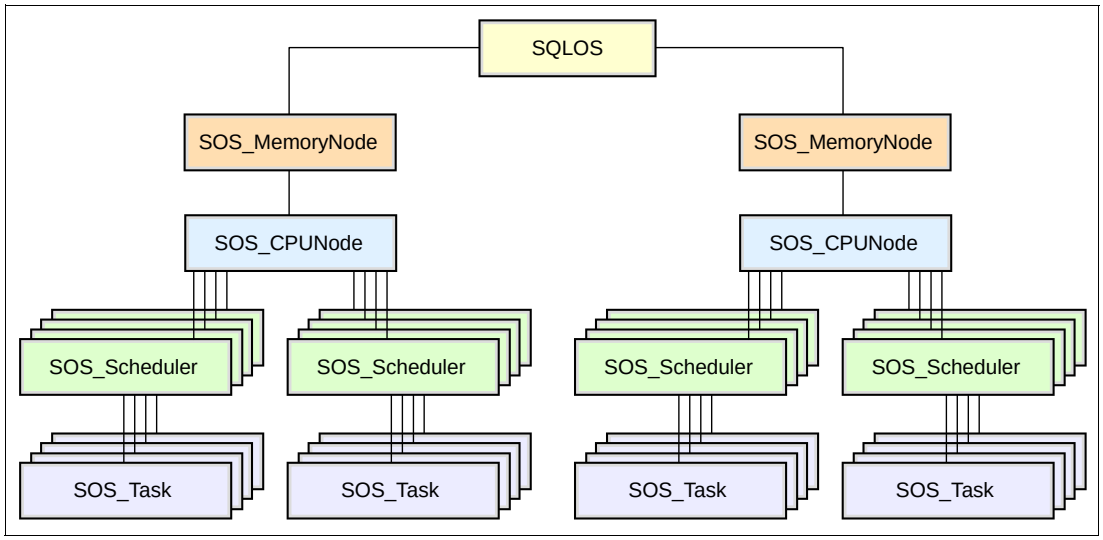


Figure 3-4 SQLOS on a NUMA server (eight-way x460 - two nodes)

Notice that with SQLOS on an SMP-based server, SQLOS only has one memory node, while SQLOS on the two-node x460 has two memory nodes. SQLOS recognizes the two nodes and the information of locality regarding processors and memory in the server.

3.4.2 Processor and I/O affinity

By default, no specific processor affinity is set in SQL Server 2005, and all processors can be scheduled to perform all tasks. SQL Server 2005 still attempts to localize the use of resources to take advantage of the NUMA design of the x460. If you wish to precisely define how system resources are used, you can enable hard processor affinity in SQL Server 2005. You can configure the affinity setting per database instance.

There are two ways to define an affinity mask in SQL Server 2005:

- Use SQL Server Management Studio graphical interface.
- Use the **`sp_configure`** stored procedure.

To change processor affinity, do the following:

1. Select **Start** → **Programs** → **Microsoft SQL Server 2005** → **SQL Server Management Studio**. The Server Properties window opens.
2. Connect to the SQL Server instance.
3. Right-click the instance icon in **Object Explore**, and click **Properties**.
4. Click **Processors** in Select a page in the left pane. See Figure 3-5 on page 43.
5. Select or clear the check marks in the Processor Affinity column to specify which processors you want this particular database instance to use.

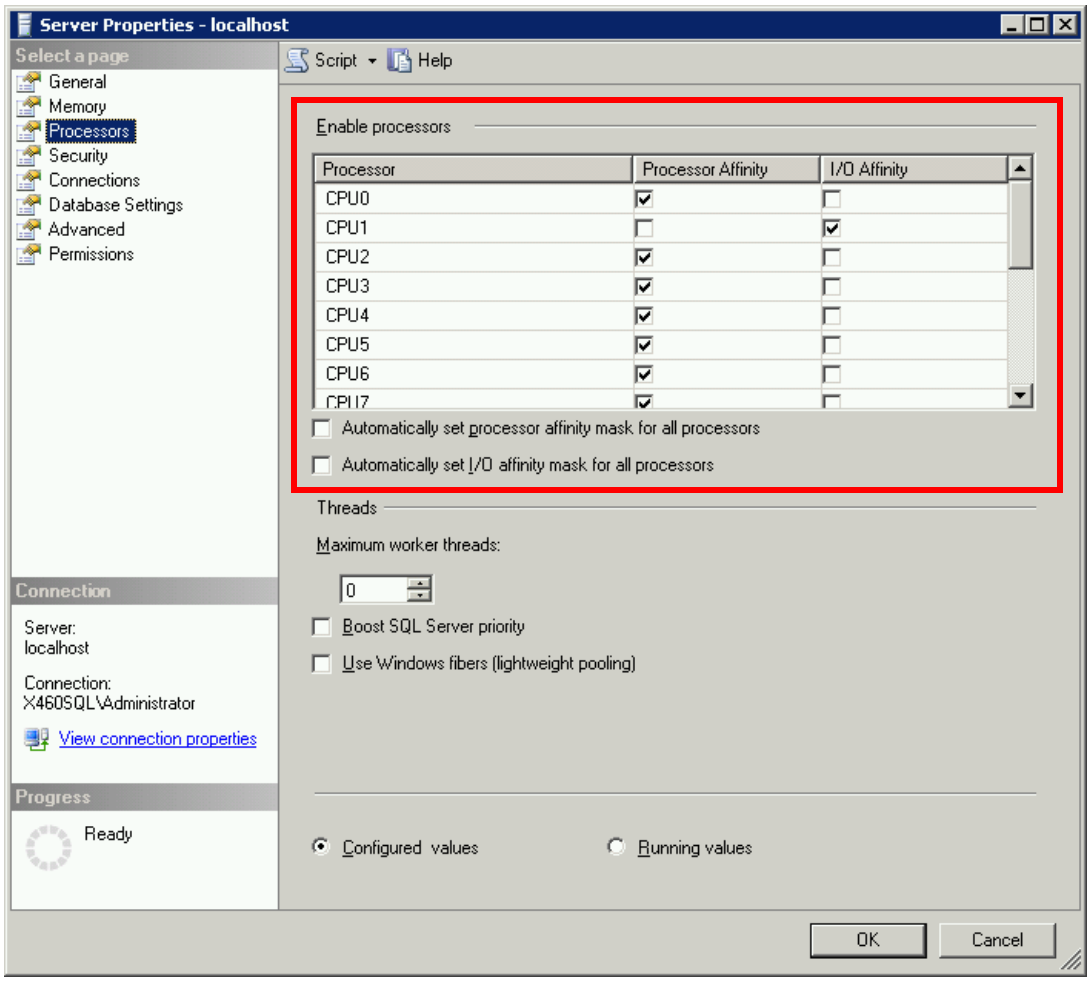


Figure 3-5 Processor affinity and I/O affinity in SQL Server 2005

You can also set affinity to disk I/O by using I/O affinity. I/O affinity can associate SQL Server disk I/O to a specified subset of processors so that specified processors handle all disk I/O.

This option can work effectively in the OLTP environment where high load is generated, especially in cases of a server with 16 or more processors. I/O affinity can enhance the performance of SQL Server threads that are issuing I/O. For example, you could assign CPU 1 for I/O affinity and all the other CPUs for processor affinity. However, I/O affinity does not always improve performance, so you must be careful before using it.

Note that this function does not support hardware affinity for individual disks or disk controllers.

We can configure I/O affinity in either SQL Server Management Studio (as shown in Figure 3-5) or the **`sp_configure`** stored procedure.

3.4.3 Network affinity

Network affinity is a new feature in SQL Server 2005. This feature provides clients with the ability to connect to specific nodes.

Tip: In the SQL Server Books Online, this network affinity is called *NUMA affinity.*

For example, consider an eight-way x460 and two network cards in the server, one in each node. In this case, we can configure the IP address that is associated with network card 1 to use the processors on node 1 (x460) and the IP address that is associated with network card 2 to use the processors on node 2 (MXE-460). We can configure these settings in the SQL Server Configuration Manager GUI or the Windows registry.

When this setting is used, the workload of IP address 1 can only be processed by processors 0 to 3 in node 1. In the same way, the workload of IP address 2 is only processed by processors 4 to 7 in node 2. Thus, each workload takes full advantage of local memory access because the requested data is located in local memory whenever possible. If the memory in the local node is insufficient, then memory in another node would be allocated.

To configure network affinity in SQL Server 2005, use a binary mask that represents nodes. The mask has a bit for each node as 76543210 with the first node as zero. See Figure 3-6. Set each node bit to 1 to select a node or 0 to not select a node.

For example:

- To specify just node 0, use mask 00000001 or hex 0x1.
- To specify just node 1, use mask 00000010 or hex 0x2.
- To specify nodes 0, 2, and 4, use mask 00010101 or hex 0x15.

To specify all nodes, use a mask of -1 or leave the field blank.

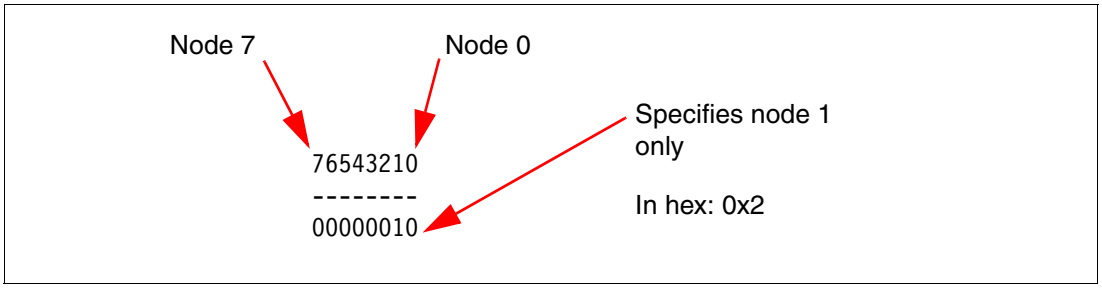


Figure 3-6 Determining the affinity mask to use for network affinity

Tip: You do not have to use different port numbers.

In our example, we configured the affinity of a two-node x460 (8-way) by mapping node 1 to 9.42.171.184, port 1444 and node 2 to 9.42.171.160, port 1500. You can do this as follows:

1. Select **Start** → **Programs** → **Microsoft SQL Server 2005** → **Configuration Tools** → **SQL Server Configuration Manager**.
2. In SQL Server Configuration Manager, expand **SQL Server 2005 Network Configuration** and expand **Protocols for <*instance name*>**, where <instance name> is the database instance you wish to configure.
3. In the details pane, right-click the IP address that you want to configure, and click **Properties**.
4. To associate a combination of IP addresses and port numbers to specific nodes, set Listen All to **No** in the Protocol tab, as shown in Figure 3-7 on page 45. The default is Yes.

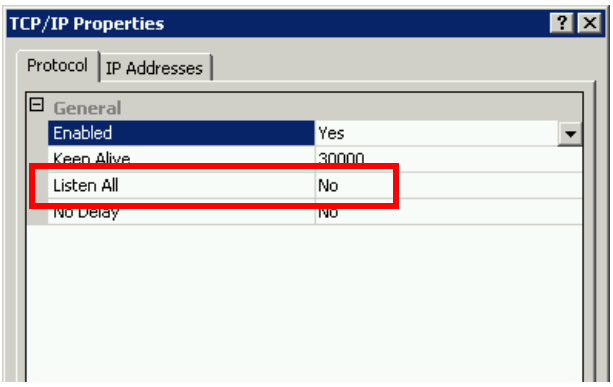


Figure 3-7 Protocol setting for affinity in SQL Server Configuration Manager

5. Click the IP Addresses tab (Figure 3-8). For each network adapter, enter the port number and node affinity mask in the form:

   `port[mask]`

   So, for network adapter 1, we want to instruct SQL Server to listen on port 1444 and have the processors in node 1 handle all requests. The value in the TCP Port field is:

   `1444[0x1]`

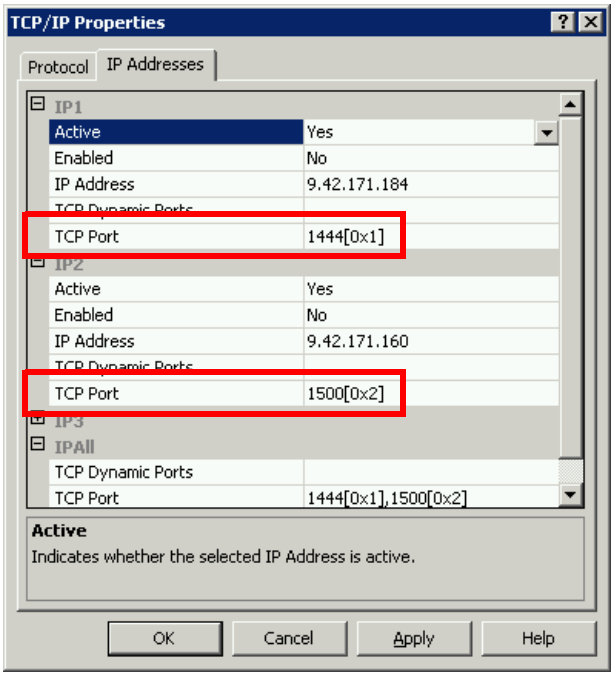


Figure 3-8 IP Address setting for NUMA affinity in SQL Server Configuration Manager

> **Note:** If you leave Listen All as **Yes** (the default) in Figure 3-7, the values in the corresponding fields in IPALL in the IP Addresses tab is applied to all IP addresses that are listed in the IP Addresses tab, as shown in Figure 3-8.

6. Restart SQL Server 2005 service so that the changes take affect.

3.4.4 Soft NUMA

Soft NUMA is another new feature in SQL Server 2005. You can use this feature to partition one physical node into multiple logical NUMA nodes. For example, you can divide an eight-way x460 that has two physical nodes into three logical nodes as follows:

- Logical NUMA node 0 to comprise of processors 0 to 3 in node 1
- Logical NUMA node 1 to comprise of processors 4 and 5 in node 2
- Logical NUMA node 2 to comprise of processors 6 and 7 in node 2

To specify the logical node arrangement, use a mask similar to that described in 3.4.3, “Network affinity” on page 43. Table 3-3 shows the masks for each of these logical nodes.

Table 3-3 Node identifier masks

Node #	Binary	Hexadecimal
Logical NUMA node 0	00001111	0xF
Logical NUMA node 1	00110000	0x30
Logical NUMA node 2	11000000	0xC0

Configuring the logical NUMA nodes requires that you directly edit the registry as follows:

1. Start **regedit**.
2. Create the registry keys and DWORD values as follows:
   - [HKLM\SOFTWARE\Microsoft\Microsoft SQL Server\90\NodeConfiguration\Node0] "CPUMask"=DWORD: 0000000F
   - [HKLM\SOFTWARE\Microsoft\Microsoft SQL Server\90\NodeConfiguration\Node1] "CPUMask"=DWORD: 00000030
   - [HKLM\Microsoft\Microsoft SQL Server\90\NodeConfiguration\Node2] "CPUMask"=DWORD: 000000C0

   These updates are shown in Figure 3-9 on page 47.
3. Map the logical nodes to the IP addresses and port numbers as described in 3.4.3, “Network affinity” on page 43.
4. Restart SQL Server service.

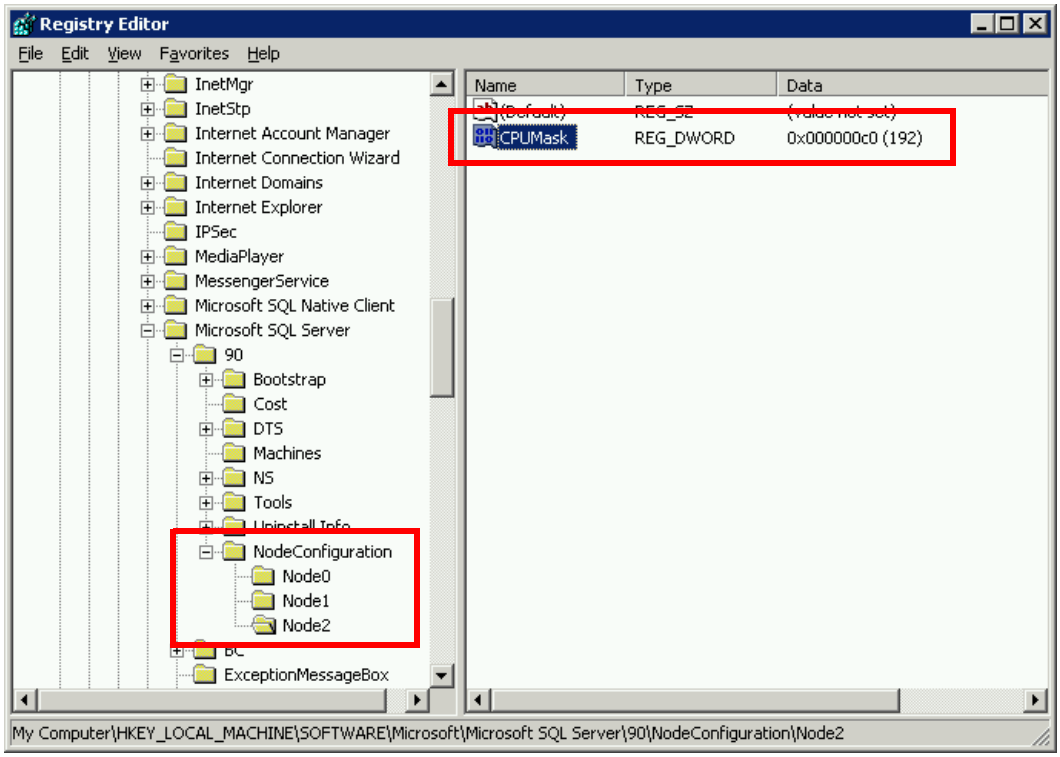


Figure 3-9 Registry settings to configure Soft NUMA nodes

You cannot create a logical node that spans multiple physical nodes. For example, you cannot configure a logical node with processor 3 in physical node 1 and processor 4 in physical node 2. You can, however, use processors in multiple physical nodes for one workload.

Consider an eight-way x460 with two physical nodes. We want to use six processors for OLAP workload and the remaining processors for data loading.

First, make three logical nodes as previously described. Then specify masks as follows (also illustrated in Figure 3-10 on page 48):

- To assign the OLAP workload to network card 1, specify node mask of logical nodes 1 and 2 (mask 00000011 or 0x3) in the TCP Port field for IP1.
- To assign the data loading workload to network card 2, specify logical node 3 for IP2 with mask 00000100 or 0x4.

When you perform this configuration, the OLAP workload to IP1 uses six processors (0 to 5) and the data loading workload to IP2 use two processors (6 and 7).

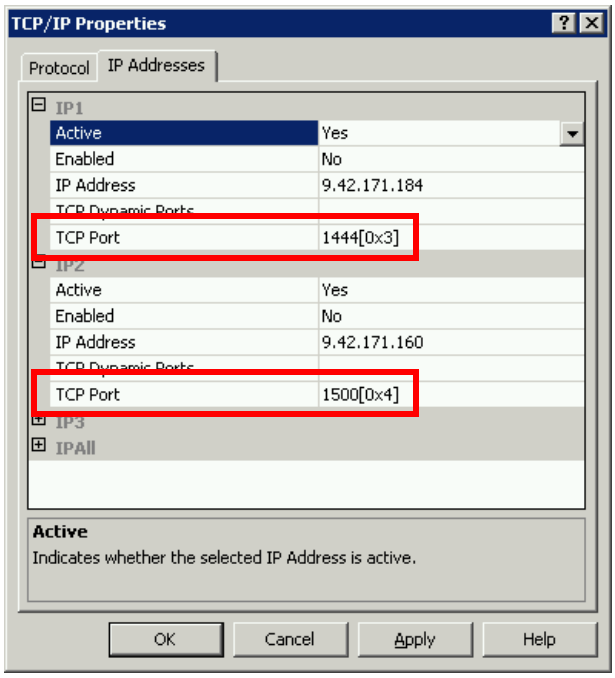


Figure 3-10 IP Address setting for soft NUMA in SQL Server Configuration Manager

3.4.5 Memory

SQL Server can allocate memory effectively in multi-node configurations such as the x460. As previously described, SQL Server 2005 has SQLOS, and it attempts to localize the use of resources to take advantage of the NUMA design of the x460. That is, SQL Server 2005 tries to allocate data in the same memory as that of the requesting processor to leverage local memory access.

SQL Server 2000 with SP4 is also NUMA-aware, although the implementation is not as efficient as SQL Server 2005. Prior to SP4, SQL Server 2000 was not NUMA-aware.

With SQL Server 2005, you can view how much memory is allocated on each node by using the DBCC MEMORYSTATUS command. When you try this command with SQL Server 2000, memory usage is not broken down by node.

In addition, you can monitor memory usage by using the Windows performance counter:

`SQLServer:BufferNode`

Tip: If you wish to see the affect of the NUMA awareness in SQL Server 2005, you can disable the NUMA optimizations with trace flag 8015. You can then confirm memory usage with the DBCC MEMORYSTATUS command. See SQL Server Books Online to learn how to set trace flags.

To configure memory per instance, use the SQL Server Management Studio or the **`sp_configure`** stored procedure. To change the memory setting with SQL Server Management Studio, follow these steps:

1. Select **Start** → **Programs** → **Microsoft SQL Server 2005** → **SQL Server Management Studio**.
2. Connect to SQL Server instance.
3. Right-click the instance icon in Object Explore, and select **Properties**.

4. Select **Memory** in Select a page in the left pane.
5. Change Minimum server memory and Maximum server memory, indicated in red in Figure 3-11.

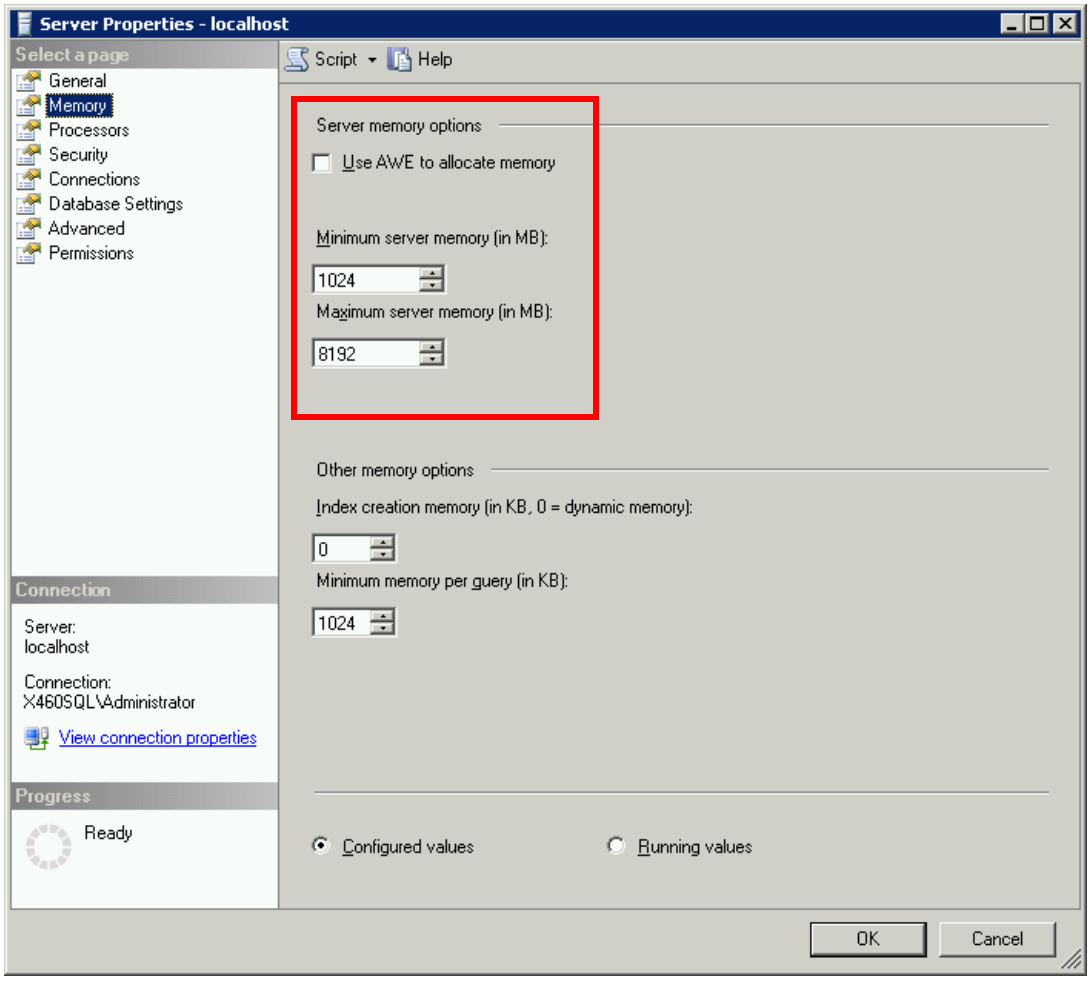


Figure 3-11 Memory setting in SQL Server Management Studio

3.5 Multiple instances

SQL Server 2005 can have multiple instances, and processors and memory can be configured per instance. If you have two or more databases, the use of multiple instances can offer manageability and cost advantages. However, because one copy of process sqlservr.exe runs for every instance, it is important that these processes do not cause contention of resources (processors, memory, PCI devices). In this section, we explain how to maximize performance in a multiple instances environment and how to decide whether to use a single instance or multiple instances when you have two or more databases.

3.5.1 Resource contention

When you use multiple instances, be careful how you configure the allocation of processors and memory. It is important that a processor not be shared between instances because doing so can cause performance degradation. For example, if you have two instances of SQL Server 2005 running on an 8-way server, you can incorrectly configure hard affinity to

processors 0 to 3 on both. The result is that both instances must share processors 0 to 3 while processors 4 to 7 are idle.

Note: By default, each instance has affinity to all installed processors. If you are running multiple instances, you must reconfigure affinity so that instances do not share processors.

Memory configuration is also important because sharing memory among instances can, in the worst case, cause system failure. For example, if you configure a minimum allocated memory for each instance but the total exceeds the amount of installed RAM, then this might cause an instance to fail.

3.5.2 Clustering issues

Mistakes in configuring multiple instances often cause serious problems in the environments in which SQL Server 2005 is running in an active/active cluster with Microsoft Clustering Service (MSCS). MSCS is high-availability solution implemented in Windows Server 2003 Enterprise Edition and Datacenter Edition. (It is also implemented in a part of Windows 2000 Server.) Two or more servers can be configured in a cluster and can switch the server in case of serious problems. For example, if an application stops because of a critical problem, the other server takes over the application after a short switching time. Thus, the application can continue to run normally.

If you have all nodes in a cluster configured to run applications and, during a failure, the surviving nodes take on the additional load from the failed node, then this is called *active-active* clustering. If instead one of the nodes is idle and its purpose is only to assume the load of a server in the event that server fails, then this is called an *active-passive* cluster.

When you run SQL Server in an active-active cluster with each instance using all processors and memory in each server, if you have not configured SQL Server properly, you might encounter serious problems when the takeover happens. If one server fails, both SQL Server instances try to run on the other server with the same configurations as before the failure, and both instances attempt to use all processors and all memory. With both instances using all processors, performance is degraded. Regarding memory, however, if the total of the minimum allocated memory for each instance exceeds the amount of installed RAM, then the failed-over instance might not restart.

Tip: It is now possible to reconfigure the minimum/maximum memory usage parameters using **`sp_configure`** without restarting SQL Server.

You can avoid resource contention and performance degradation by taking the following precautions:

- Use active-passive clustering.

  In active-passive clustering, one server is reserved for standby. Thus, there are no resource contentions if takeover occurs.

- Configure processors and memory of each instance appropriately on Active/Active MSCS.

  For example, consider having two 8-way 32 GB x460 servers. You can avoid resource contention after takeover by configuring four processors and 16 GB of RAM to both instances (that is, the sum total of both instances should be set to eight processors and 32 GB of RAM).

  Like the active-passive option, half of both servers will be idle. In addition, you should carefully select the four processors so that cluster node A, for example, assigns affinity to processors 0 to 3 and cluster node B assigns affinity to processors 4 to 7.

- Use active-active clustering and use **sp_configure** to reallocate memory usage.

  In this scenario, you have SQL Server issue the **sp_configure** procedure when it restarts to dynamically reallocate memory to all instances so that the total amount of memory allocated does not exceed the amount of memory installed.

  If you plan to implement this method of failure recovery, carefully test the scenario to ensure that all instances do start properly with the correct amount of memory allocated.

3.5.3 Performance

To gain good performance when you have configured multiple instances, you should map instances to a particular processor (or processors) in each node. For example, consider an eight-way x460 (Figure 3-12):

- Instance 1 is mapped to processors 0-3 in node 1.
- Instance 2 is mapped to processors 4-5 in node 2.
- Instance 3 is mapped to processors 6 and 7 in node 2.

Because the SQLOS on the x460 is NUMA-aware, this configuration maximizes the use of local memory access.

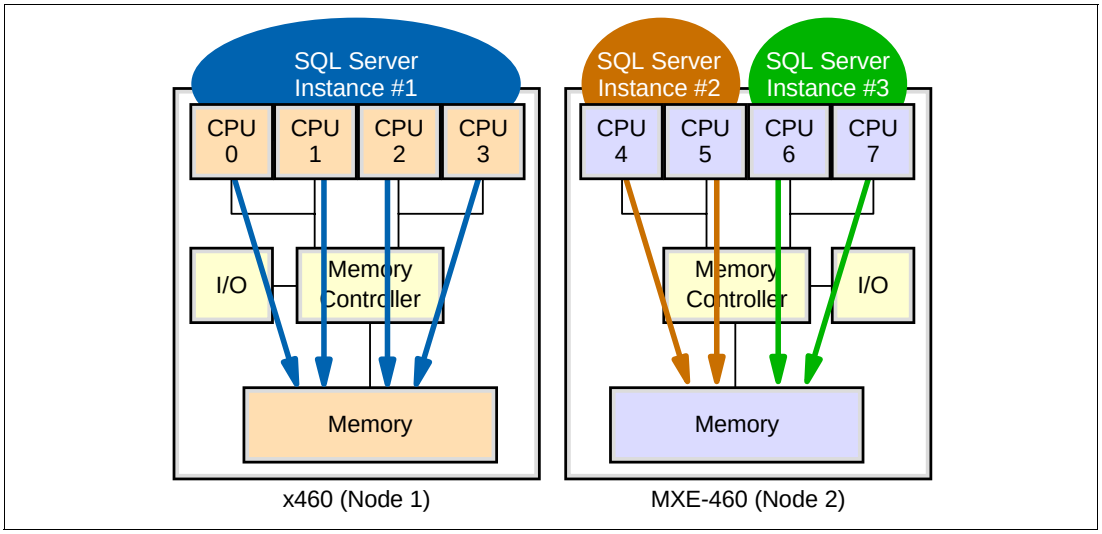


Figure 3-12 Memory access from multiple instances configured NUMA-conscious

You should avoid configuring an instance to use processors from multiple nodes, because this requires remote memory access (across the scalability connections), and performance can suffer. For example, do not configure processors 3 and 4 in the same instance. Of course, if you wish to create an instance with more than four processors, then this is unavoidable.

3.5.4 Single or multiple

There are two choices when running multiple databases on a server:

- Running multiple databases on a single instance

  Multiple databases on a single instance can be a simple way to host multiple databases because it is easier to implement than combining all source databases into a single target database. Multiple databases can be easier to manage, provide better recovery time, and spread logging activity over multiple logs. However, multiple databases can have a performance drawback in the following situation. When a local transaction spans two databases, SQL Server uses an internal two-phase commit. This is more expensive than a

commit that involves only one database. Another drawback is that the buffer pool is shared across the multiple databases, which can result in one database “starving” another for resources.

You also need a two-phase commit to implement transactions across multiple databases in multiple instances

- Running databases on multiple instances

  The multiple instances option offers a division of administrative responsibility and a simpler consolidation model than combining all source instances into a single target instance. Multiple instances that communicate with each other can do so more quickly when consolidated on a single server. When Hyper-Threading is enabled, always affinitize both logical processors on a physical processor to the same SQL Server instance.

You should consider the following when choosing which to implement:

- Operation and maintenance of the database
- Version of SQL Server 2005 (32-bit or 64-bit)
- Performance

Operation and maintenance of the database

You can start and stop SQL Server 2005 per instance. So, if one database needs continuous availability and the other database does not require this level of availability, you might consider multiple instances to reduce the operation and maintenance workload. For example, you can apply the Service Pack to SQL Server 2005 per instance. However, you need to stop the corresponding SQL Server service in Windows Server 2003. If you have multiple instances, you can apply the Service Pack separately. That is, you can apply Service Pack to the database that does not need continuous availability, and you can leave the continuously available database as it is.

Version of SQL Server 2005 (32-bit or 64-bit)

Thirty-two-bit applications can use only 2 GB of VAS because of the user space on 32-bit Windows. That is, 32-bit SQL Server 2005 can use a maximum of 2 GB of VAS per instance, or 3 GB if you are using 4 GB tuning in Windows Server 2003.

When you run 32-bit SQL Server 2005 on Windows Server 2003 x64 using WOW 64 (Windows on Windows), SQL Server 2005 can use a maximum of 4 GB as user space, as shown in Table 3-4. Then, if you use a single instance, you can use 2 GB (or 4 GB on x64 Windows) of VAS for multiple databases.

Table 3-4 Address space in 32-bit and 64-bit

Windows Server 2003	SQL Server 2005	Virtual memory limits	Physical memory limits
32-bit	32-bit	2 GB (3 GB with boot.ini flag)	64 GB
64-bit (x64)	32-bit	4 GB	64 GB
64-bit (x64)	64-bit (x64)	8 TB	1 TB

Alternatively, if you use multiple instances, you can use 2 GB (or 4 GB) x *n* (where *n* is the number of instances) as a VAS for multiple databases. In 64-bit Windows Server 2003 and SQL Server 2005, VAS is available to a maximum of 16 TB (8 TB for the kernel and 8 TB for the user).

Performance

If you use a single instance on a multiple node x460, you have some remote memory access or far memory access in a workload, as shown in Figure 3-13 on page 53. Although memory allocation of SQL Server 2005 is NUMA optimized, some remote and far memory access can still happen. For example, in Figure 3-13 on page 53, CPU 4 has to obtain the data from remote memory if CPU 0 has previously located the data in its local memory.

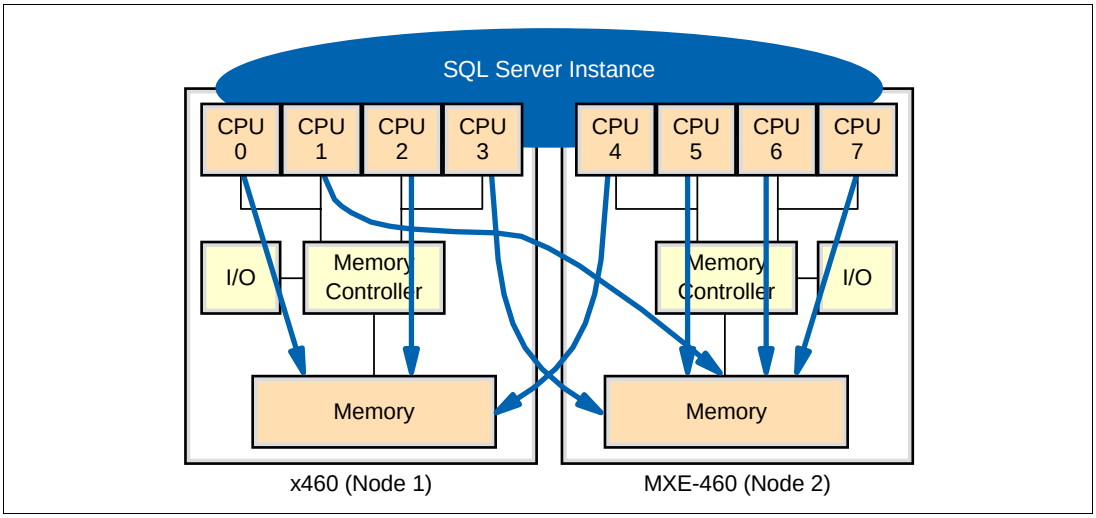


Figure 3-13 Memory access from single instance

In multiple instances, if the instance has affinity to the node set correctly, there is little remote and far memory access, as shown in Figure 3-12 on page 51. With 64-bit SQL Server 2005, the limitations of VAS are eliminated. However, when you are reducing remote and far memory access, you should examine multiple instances for multiple databases with SQL Server 2005 64-bit. In addition, if Hyper-Threading is enabled, you should always affinitize both logical processors on a physical processor to the same SQL Server instance.

A drawback of having multiple databases in a single instance is that the buffer pool is shared throughout the databases, so there is the potential that one database can “starve” the others of buffer pool resources. The drawbacks of having multiple instances are that you must do a two-phase commit and that you need Distributed Transaction Coordinator (DTC) to implement transactions across multiple databases.

3.6 Server consolidation

This section describes server consolidation. It introduces the model of general server consolidation and specifically discusses database server consolidation. This section also explains vertical consolidation and horizontal consolidation, both of which are used in database server consolidation.

3.6.1 General notion of consolidation

The general types of consolidation are:

- Centralization
- Physical consolidation
- Data integration
- Application integration

The simplest form of consolidation is *centralization*, where servers are moved to a common location. The move can be physical or virtual. Centralization provides very few advantages to server consolidation. However, it is a first step toward future consolidation efforts because it makes future growth easier and more measurable. It also makes future high availability a feasible solution.

Physical consolidation is the process of replacing a number of smaller servers with larger servers with the same processor architecture. In many situations, centralization is a prerequisite for physical consolidation. The number of servers that can be consolidated to a single server depends on the following:

- Capability of the old servers
- Usage of old servers
- Capability of the new servers

Data integration is the process of taking information from several disparate sources and merging it into a single repository and a common format. Each data element is made to follow the same business logic or, by deploying the shared storage subsystem, to manage disk requirements in a heterogeneous environment.

Application integration is the combining of multiple, similar applications, such as Web servers, onto one consolidated server. It is also the process of migrating an application to a larger system to support data integration (for example, migrating four SQL Server 2000 servers, each of which supports 100 users to a single SQL Server 2005 server that supports 500 users).

3.6.2 Database server consolidation

This section describes the types of database server consolidations, as listed in Table 3-5.

Table 3-5 Types of database server consolidation

Type	Description
Vertical consolidation	Databases with the same schema are merged into one database.
Horizontal consolidation	Databases with different schema are unified to one database server but in potentially one or more database instances.
Redesign consolidation	Databases with different schema are redesigned and then merged into one database.
Combination	The three consolidation types are arbitrarily combined.

Vertical consolidation is the case where data is managed separately by organizations throughout a company in each area and this company is doing nationwide consolidation and deployment. Because the schema of the databases is typically the same, data can be gathered to a single server for consolidation. It is possible to reduce both the number of databases and the number of server machines in vertical consolidation. We discuss this further in 3.6.3, “Vertical consolidation” on page 55.

Horizontal consolidation is used in the case where a personnel database, a finance database, and a sales database are managed by their respective departments. The database schema are perceived as different because the data for each business section is different and, therefore, the schema of the databases are not arranged. The databases are only moved horizontally, lined up side-by-side in the instance in a new server for consolidation. Thus, although the number of the databases does not change, you can reduce the number of server machines. We discuss this further in 3.6.4, “Horizontal consolidation” on page 56.

Redesign consolidation is usually a result of business process reengineering (BPR), mergers and acquisitions (M&A), and so forth. It is normal in the situations to redesign and reconstruct data so that the databases can be merged. However, once the reconstruction has been identified, this type of consolidation can be another form of vertical consolidation.

The last type of consolidation, *combination*, is a mixture of vertical consolidation, horizontal consolidation, and redesign consolidation.

3.6.3 Vertical consolidation

The purpose of vertical consolidation is to consolidate servers that contain the same types of database schema to one server. For example, consider four Netfinity® 8500R servers, each with the same database schema and the same workload as shown in Figure 3-14. The only difference between the four Netfinity 8500R servers is the data alone. Thus, if the four databases are unified by one database, we can also consolidate four Netfinity 8500R servers to one scalable x460 server.

If we simply move each database to the x460 without merging them into one database, then a local transaction that spans two databases uses an internal two-phase commit. This is more expensive than a commit that only involves one database.

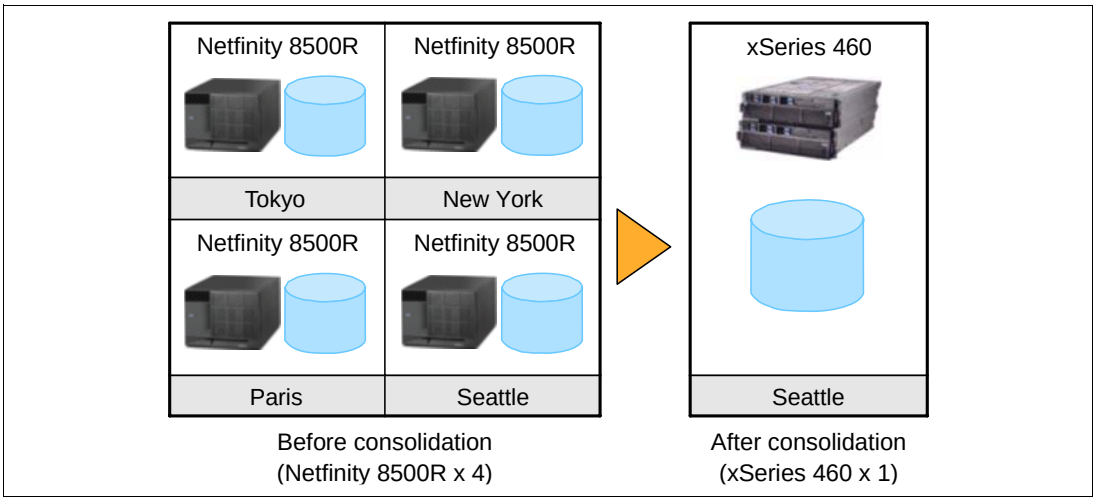


Figure 3-14 An example of vertical consolidation

In this example, we can use a typical extract, transform, load (ETL) tool that is suitable for data consolidation of SQL Server, including:

- SQL Server Integration Services in SQL Server 2005
- Data Transformation Services (DTS) in SQL Server 2000
- Bulk Copy Program (BCP) in both SQL Server 2000 and 2005
- Bulk Insert in both SQL Server 2000 and 2005
- Other third-party tools

SQL Server Integration Services offers powerful and flexible data transformation and is implemented in SQL Server 2005 by default. SQL Server Integration Services has a graphical interface with high operability. By contrast, both BCP and Bulk Insert only have a command-line interface. SQL Server Integration Services offers better performance than DTS. As a result, SQL Server Integration Services is recommended for your data consolidation.

3.6.4 Horizontal consolidation

The objective of horizontal consolidation is to consolidate servers that contain different types of databases (or database schema) to one server.

For example, suppose you have 12 Netfinity 5000 servers, two of each run either finance, HR, sales, marketing, IT, or executive databases (see Figure 3-15).

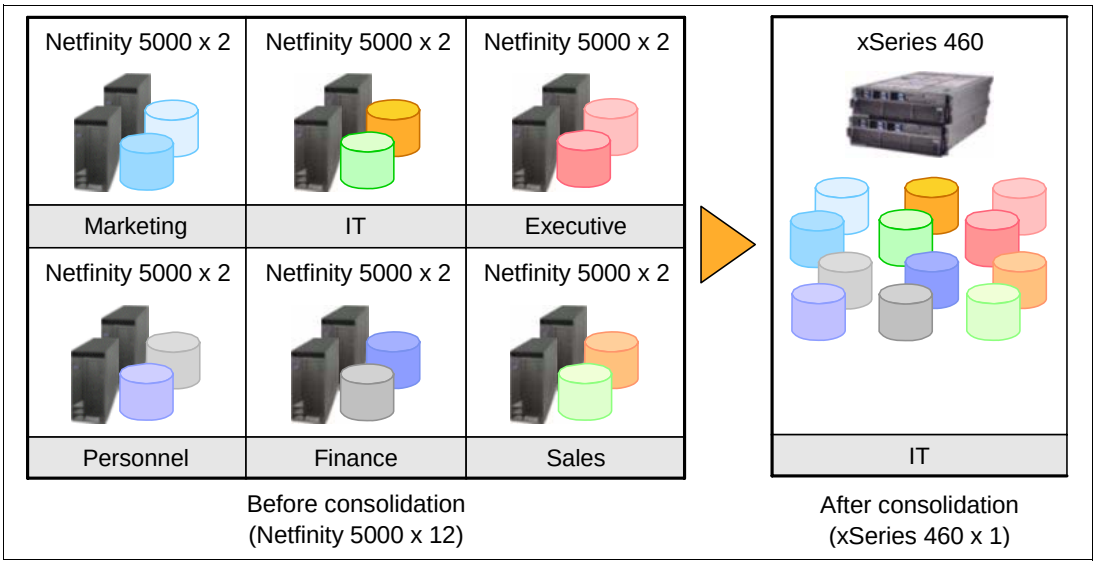


Figure 3-15 An example of horizontal consolidation

The schema and role of these databases are different. So, when the databases are consolidated, we move these databases to separate SQL Server instances on the x460. Then, the x460 has multiple databases that are running multiple SQL Server instances. In this case, the databases might be divided into six instances that are based on a business affairs viewpoint. However, we can also consider the number of instances from an operation and maintenance viewpoint or a performance viewpoint.

The multiple instances option offers a division of administrative responsibility and a simpler consolidation model than combining all source instances into a single target instance. Multiple instances that communicate with each other can do so more quickly when they are consolidated on a single server.

In the horizontal consolidation, each instance should have affinity to a subset of processors in a node as long as the number of processors needed does not exceed four. Also, you should configure the memory setting to avoid resource contention between the instances.

4

Configuration

In this chapter, we discuss setting up and configuring the x460 to run SQL Server 2005 x64 with an emphasis on performance tuning. We cover these topics in the approximate order that you encounter them when you are setting up the complete environment; first the hardware, then the operating system, then SQL Server, then the storage, and, finally, the database server in operation.

The topics covered are:

4.1 xSeries 460 configuration

In this section, we first consider the physical hardware setup and then the BIOS settings.

4.1.1 Setup

When you set up the physical hardware, the best performance is achieved by provisioning each of the x460 nodes uniformly. This complements the NUMA architecture by providing local resources on each node.

During the process of connecting the scalability cables for servers of up to four nodes (16-way), it is possible to connect every node to every other node. Cabling that introduces two hops can penalize performance (see 3.1, “Scalable hardware implementation” on page 36). This is unavoidable in the eight-node configuration. When eight nodes are cabled together and partitioned into one or two 16-ways, the 16-way does not perform as well; it acts as though it had been cabled separately as a 16-way for this reason.

In each x460 node, memory is installed in four memory cards, each of which has four DIMM slots, for a total of 16 DIMMs. In order of importance, there are three aspects to the memory configuration that affect performance:

1. Install and use all four memory cards in every node
2. Balance the amount of RAM installed in every node
3. Populate every DIMM socket

Optimal memory configuration uses every slot on every card on every node, with the same size DIMMs, so that the amount of memory is balanced across all the nodes. Other memory configurations that do not use all the memory cards or all the DIMM slots on the cards or that are unbalanced across the nodes penalize performance to some degree.

Eight DIMMs on four cards usually provides similar performance, giving a maximum difference of 3% to 5%. Four DIMMs on four cards will be slower than eight DIMMs by up to 10% for some applications. Four DIMMs on a single memory card will be about 50% slower than four DIMMs on two memory cards.

In x460 multi-node configurations, performance can be impacted by the installation of PCI cards such as network adapters, FC HBAs, and so on. To distribute the load equally, the optimal solution is to spread the adapters across all the nodes. For example, in a four node configuration with four FC cards, put one card in each node. Also, you should connect at least one Ethernet connection on each node. Each x460 has two integrated Gigabit Ethernet controllers.

4.1.2 Firmware and BIOS

After the hardware is physically set up, flash the firmware on all the nodes to the same supported level. Stability and performance issues can arise if the firmware is not the same on all nodes, or if it is not at a supported level.

The following list identifies the firmware that must be flashed and the recommended order. Your configuration might include other devices that must also be updated.

1. BIOS
2. Diagnostics
3. BMC firmware
4. Complex Programmable Logic Devices (CPLD) firmware
5. Remote Supervisor Adapter II SlimLine (RSA II SL) firmware

6. Serial Attached Small Computer System Interface (SAS) controller
7. SAS hard drives
8. ServeRAID firmware

Note: If you are running Windows Server 2003, Datacenter Edition, be sure that the firmware is certified for Datacenter.

When you make BIOS settings, keep them identical on all nodes. Begin by loading the default BIOS settings. The following parameters should be examined and tested with your workload to determine what maximizes performance.

- **Processor hardware prefetcher**

  By default, hardware prefetching, which is what processors use to prefetch extra cache lines for every memory request, is enabled. Recent tests in the performance lab have shown that disabling this feature provides the best performance for many commercial application types with a random workload such as online transaction processing (OLTP). The performance gain can be as much as 20%, depending on the application.

  To disable prefetch, go to BIOS Setup (press F1 when prompted at boot) and select **Advanced Settings** → **CPU** and set **HW Prefetch** to **Disabled**. This setting affects all processors in the chassis. Future releases of BIOS that ship to enable dual core will have HW Prefetch disabled by default.

  For the Decision Support System (DSS) workload that is mentioned in 2.6, “Customer proof of concept” on page 32, we realized an 18% improvement with this setting enabled compared to when it was disabled.

- **Processor adjacent sector prefetch**

  Also in **Advanced Settings** → **CPU**, when this setting is enabled, the processor retrieves both sectors of a cache line when it requires data that is not currently in its cache. When it is disabled, the processor only fetches the sector of the cache line that contains the data requested. The default is enabled.

  This setting might affect performance, depending on the application running on the server and memory bandwidth usage. Typically, it affects certain benchmarks by a few percent, although in most real applications, the effect is negligible. This control is provided for benchmark users that wish to fine tune configurations and settings.

- **Hyper-Threading**

  The Xeon MP processors in the x460 include a technology called Hyper-Threading, which makes a single CPU appear to the operating system as two logical processors, so that they can receive and process two data/instruction streams simultaneously.

  Hyper-Threading is enabled by default. For OLTP, enabled works best. However, test the effect of both by trying both. To change it, go to **Advanced Settings** → **CPU.**

  In x460 multi-node configurations, all nodes in a partition should have the same Hyper-Threading setting.

- **Memory array settings**

  This setting is in **Advanced Settings** → **Memory**. The choices are:

  – **Redundant Bit Steering** (the default) provides a good level of fault tolerance and performance.

  – Use **Hot Add Memory**, if you intend to add memory while the server is running, at the expense of reduced performance.

- Use **Full Array Memory Mirroring**, if you want the highest level of memory redundancy, at the expense of reducing available memory by half and reducing performance.
- Use **High Performance Memory Array**, if you wish to trade fault tolerance for the highest possible performance.

For additional information about x460 settings, see *Planning and Installing the IBM @server X3 Architecture Servers*, SG24-6797.

4.2 Windows Server 2003 x64 configuration

In this section, we cover those aspects of the Windows operating system that are relevant to a server that is running SQL Server 2005. In general, no special settings are required. You are likely to find that leaving the defaults offers the best results.

4.2.1 Windows installation, service pack, updates, and drivers

After installing Windows Server 2003 x64, apply the most recent service pack, security patches, and current supported drivers. Unsupported or back-level software can cause system instability, security vulnerabilities, and performance problems.

4.2.2 Windows settings

SQL Server Setup automatically sets Windows Server 2003 optimization settings in the **File and Printer Sharing for Microsoft Networks** properties to **Maximize data throughput for network applications** so that the server can accommodate more connections. See SQL Server Books Online topic “Maximizing Data Throughput.”

SQL Server Setup automatically sets server tasking to **none** in Windows Server 2003 (the SQL Server default), which gives foreground and background programs equal processing time. See SQL Server Books Online topic “Configuring Server Tasking.”

The Windows page file size is not important for performance. On a server dedicated to running SQL Server, there should be no paging. A modest-sized page file for taking mini-dumps, located on local storage, is sufficient. Complete system dumps of physical memory configurations of 64 GB and larger are impractical for debugging.

4.2.3 Anti-virus software

There are steps you can take to prevent anti-virus software from impacting database operations and performance.

If you decide to install anti-virus software on the server running SQL Server, configure it so that updating the virus signatures and scanning the system occur during periods of low levels of activity. Exclude the database file extensions MDF, LDF, and NDF from the virus scan, or exclude the folders that they are stored in. This prevents the SQL Server from trying to open SQL Server files that the anti-virus software has already opened, which can result in the database being marked as suspect.

Visit these Web sites for more information:

http://support.microsoft.com/default.aspx?scid=kb;en-us;309422
http://www.microsoft.com/sql/techinfo/tips/administration/virusscanning.asp

4.3 SQL Server 2005 x64 configuration

This section covers installing and configuring SQL Server 2005 x64.

4.3.1 SQL Server 2005 installation, service packs, and hot fixes

Before installation, create one or more domain user accounts to be used as SQL Server service logins, if you want SQL Server 2005 to communicate with other clients and servers.

Install SQL Server 2005, and install any applicable service packs and hot fixes.

4.3.2 SQL Server 2005 settings

After SQL Server is installed, consider adjusting the following configuration settings. SQL Server does excellent self-tuning. In general, the default settings work best.

Note: These settings can be made using the **`sp_configure`** command. Some of these options can be set only after configuring **show advanced options** as **1**.

affinity mask

This configuration option restricts a particular SQL Server instance so that it runs on a subset of the processors. If you are running only one instance of SQL Server 2005 and nothing else, then allowing SQL Server to use all processors creates the best performance. Multiple instances should always be configured to partition the hardware resources.

Use **affinity mask** and **max server memory** to place each instance on its own node, or group of nodes, for best performance. When you have Hyper-Threading enabled, if you affinitize CPUs, always affinitize both logical processors on a physical processor to the same SQL Server instance. Otherwise, performance will suffer. Also see 3.5, “Multiple instances” on page 49.

awe

It is not necessary to configure AWE for SQL Server 2005 x64. Sixty-four-bit applications do not require AWE because access to memory is not limited to 4 GB. However, for the highest level of performance on x64, you can enable the Lock Pages in Memory privilege and SQL Server 2005 automatically uses the AWE mechanism to allocate memory. Allocated memory is mapped to the SQL Server VAS immediately. After it is mapped, physical memory does not have to be unmapped.

Using this feature is advisable only when a value is specified for Max Server memory.

max degree of parallelism

Setting this option to **1** prevents the query optimizer from choosing parallel query plans. Parallel query plans reduce the time that it takes to complete a query at the expense of increased resource utilization and reduced total throughput. Using multiple processors to run a query is usually not desirable in an OLTP workload, although it is desirable in a DSS workload. Note that it is possible to assign query hints to individual queries to control the degree of parallelism. This is useful for a mixed workload with both OLTP and DSS.

max server memory

When max server memory is kept at the default setting, SQL Server acquires and frees memory in response to internal pressure (to expand because of work requests) and external pressure (to shrink from the operating system). On a dedicated server that is running a heavy

workload, SQL Server continues to acquire memory until the maximum amount of physical memory is nearly reached. In this case, you might find that guaranteeing memory for the operating system, by setting **max server memory** to 2 GB to 4 GB less than the maximum physical memory, gives better performance. Multiple instances should always be configured with **max server memory** set so that the instances do not compete for memory.

priority boost

Setting this option to **1** changes the base priority of SQL Server from 7 to 13. On a dedicated server, this might improve performance, although it can also cause priority imbalances between SQL Server functions and operating system functions, leading to networking or other errors. **Priority boost** should not be used when you are implementing failover clustering.

recovery interval

The default is 0, indicating automatic configuration. In practice, this generally means a recovery time of less than one minute and a checkpoint approximately every minute for active databases. Keep **recovery interval** set at 0 (self-configuring) unless the checkpoints impair performance because they occur too frequently. If so, try increasing the value in small amounts. See SQL Server Books Online topic “Understanding Recovery Performance.”

4.4 Storage configuration

In this section, we describe how you can configure the disk storage for SQL Server if you are connecting the x460 to a FC-attached SAN. A SAN offers great configuration flexibility. Similarly, SQL Server 2005 offers great flexibility because you can put database objects in multiple files and group files into multiple file groups. Each of the various database workloads (OLTP, DSS, and the maintenance operations), impose a characteristic I/O profile on the storage. Over time, new workloads can be introduced that might alter the I/O profile.

One of the most important and difficult design decisions that affects performance in a database implementation is how to make the best use of the flexibility offered by the SAN and SQL Server to maximize performance for the unique I/O profile exhibited initially and in the future. The problem has a chicken and egg quality, because the configuration of the storage is based on the I/O profile, but the I/O profile can best be determined by measuring it on a running system.

For starters, try to model your proposed storage configuration on paper, using a spreadsheet or a software tool designed to do this type of modeling. You can also attempt to extrapolate from the performance characteristics of an earlier implementation of the proposed environment. If this is too complicated or you do not know the characteristics, then you can use the rules that are described in the sections that follow.

4.4.1 Storage setup

The steps that are required to configure and present the Logical Unit Numbers (LUNs) to the operating system as physical disks are:

- Host to storage connection

  This addresses the paths between the x460 and the storage server. The components of these paths are the FC HBAs in the x460, the cables, the FC switches, and the storage server controllers. The guiding principle from a performance perspective is to spread the peak I/O traffic evenly through these components. From a redundancy perspective, no single component should cause failure, which implies a minimum of two HBAs, two switches, two controllers, and at least two paths to each LUN.

- Create arrays

  An array is composed of a set of disk spindles (usually in the range of 10 +/- 5 spindles), a RAID level and a stripe unit size. In general, a greater number of spindles provides greater throughput (I/O per second and Mbps). If you plan to use significantly less than the total disk capacity. RAID-10 usually performs better than RAID-5 (especially for random write workloads) while providing approximately half of the capacity and more resiliency to disk failures.

  A large stripe unit size benefits a sequential workload, and a small stripe size benefits a random workload. If you have no other information, a stripe unit size of 64 KB is advisable as a starting point from which you can tune. For a more in depth discussion of choosing these parameters, see the white paper, “Disk Subsystem Performance Analysis for Windows:

  http://www.microsoft.com/whdc/device/storage/subsys_perf.mspx

- LUNs

  LUNs are created from arrays. Using an entire array to create a single LUN is the simplest and most advisable design, because multiple LUNs created from a single array can interfere with the performance of the other LUNs, which share spindles. The likely result is erratic performance. One LUN per array also simplifies performance monitoring.

- Configure controller cache

  In an enterprise-level SAN, significant cache is available, which can enhance performance greatly. In general, take the database option, when one is offered. This includes settings such as:

  – Read caching ON
  – Cache read-ahead multiplier OFF
  – Write caching ON (be sure the cache is battery backed to avoid data corruption or power failures)

- Creating Windows logical disks

  When you create partitions in Windows, be sure they are aligned. You can use the **diskpart** program with the **/align** option, which became available with Windows 2003 Service Pack 1. You can test the alignment with the SQLIO utility mentioned in the next bullet point, using a heavy workload of small concurrent requests. Use an NTFS allocation size of 64 KB, prefer MBR over GPT, and basic volumes over dynamic volumes. Use the storport driver versus the scsiport driver.

  Note: SQL Server 2005 supports mount points. Mount points reduce the number of drive letters required, which can be an issue when building multi-node clusters.

- Stress and performance test

  To validate the storage that you have configured for reliable use with SQL Server, you can run SQLIOStress for several days, to detect rare, intermittent failures. You can obtain SQLIOStress from:

  http://support.microsoft.com/default.aspx?scid=kb;en-us;231619

  To verify the performance characteristics of the volumes that you have created, you can use the SQLIO utility, available from:

  http://www.microsoft.com/downloads/details.aspx?FamilyId=9A8B005B-84E4-4F24-8D65-CB53442D9E19&displaylang=en

Using SQLIO, you can put a load on the storage system, based on the I/O profile that you anticipate. You should keep a baseline set of statistics for each LUN (for example, 8 KB, 64 KB, 1 MB, random/sequential, read/write, and various queue depths) that you can use to compare against actual SAN performance when you run the database application.

Note: These utilities mentioned here do not use SQL Server. They are standalone programs that generate a synthetic load on the storage system. Also, it is not advisable to use them to stress the OS boot drives (the default for SQLIO), only the FC-attached storage.

4.4.2 Storage file placement

This section addresses placing the various database files on the logical disks that appear in the operating system. There are several conflicting goals to meet: availability, performance, and cost. Designing the LUNs, the database files, and their placement for a large SAN and a large complex database is a difficult task. In fact, there are so many details, complex interactions, and possibilities, that automation of the design or implementation can be very effective.

Automation is even more important when the workload is dynamic, because the storage configuration can be modified frequently while the database application is running. The following general guidelines supplement whatever level of automation is available:

- Availability

  To maximize availability, the log and backup files should be placed on arrays that are separate from the data files. This is because you can recover from an outage caused by media failure of the data files by using the log and backup files to restore the database. To reduce the chance of media failure, use RAID-10 over RAID-5. To reduce backup time and recovery time, which contributes to availability, use multiple data files and multiple backup devices to allow parallel data transfer to all devices simultaneously. To reduce recovery time, use RAID-10 and put the source and target files for these operations in separate arrays.

- Performance

  To maximize performance, all files should be put in RAID-10, especially data files that are subject to random workloads. To reduce disk contention, data and log files for the same database should be put in separate arrays. A large component of disk access time is moving the disk arm, so isolating a log file in its own arrays allows sequential writes to proceed with minimal disk arm movement, improving performance.

  Dividing a database into multiple file groups and putting them in separate arrays can improve performance by distributing the I/O load over more spindles. One approach is to separate base tables and their non-clustered indexes into separate file groups and place them on separate arrays. The query in Figure 4-5 on page 69 provides I/O statistics for balancing I/O in this way.

  Tempdb, which is a global resource for each SQL Server instance, can be a scalability bottleneck. Concurrent access to tempdb can be improved by splitting the tempdb data into multiple files. It is best if you create no more than one file per CPU, unless there are more than eight CPUs, in which case, the maximum number of files should be ¼ to ½ the number of CPUs. Tempdb should be in its own RAID-10 and separate from the user databases, for best performance.

  Sizing data, log, and tempdb files so that they do not have to grow, improves performance. Arrays that are well below their maximum space capacity perform better than ones that are nearly full.

- Cost

  When it is not economical to put each file in its own array, consolidating files with similar workloads is another option. For example, combine read-only data files in the same array, randomly accessed data files in the same array, and log files in the same array. Avoid putting files with disparate workloads in the same set of spindles. Combine files that are accessed at different times of the day in the same array. When it is not economical to use RAID-10, RAID-5 for the data files and backups might be acceptable.

  Try to match the performance requirements of the various files with the performance characteristics of the arrays, based on the measurements you made with SQLIO. The goal is to avoid wasting performance capability on files that do not need it.

4.4.3 Standardized storage configuration

The POC described in 2.6, “Customer proof of concept” on page 32 offers a different approach to storage configuration. In the proof of concept, standardized arrays, using the same number of spindles each, configured as RAID-5, were created using all of the available storage. Then software striping in the Windows operating system was used to create logical disks that combined various groupings of these standard building blocks.

Rather than try to design a custom configuration that is based on the expected performance characteristics of the workload, this approach tries to distribute multiple workloads across a very large number of individual disk spindles. The assumption is that any contention or hot spots are automatically spread out over enough disks that performance is acceptable. This approach has reasonable cost (for example, RAID-5), adequate performance, and no significant availability issues for a read-only (for example, DSS) workload. An additional advantage is that it does not require complex analysis to design and maintain the storage configuration, which translates into reduced support costs.

4.5 Database workloads

This section identifies typical database workloads and their I/O, CPU, and memory characteristics.

4.5.1 Maintenance operations

Maintenance operations include create, load, back up database, restore, and index rebuild. These operations can be characterized as heavy sequential read and write that use large block sizes. Usually they are scheduled during off-peak periods and the primary issue is how quickly they can complete.

With the exception of index rebuild, which has a CPU-intensive and memory-intensive middle phase, these operations are limited by the performance of the disk arrays involved and require little CPU or memory resources. It is possible to shorten the elapsed time of these operations by implementing multiple file groups on separate devices and performing the operation in parallel for each file group. See, for example, SQL Server Books Online topic “Optimizing Backup and Restore Performance.”

4.5.2 Application workloads

There are two main types of database application workloads: a DSS workload and an OLTP workload. Some environments operate a mixed workload with both types.

OLTP

An OLTP workload is characterized by random read/writes against the data files and sequential writes against the log file. Usually there is no tempdb activity. Because of the checkpoint process, the write activity to the data files occurs in spikes, usually once a minute for an active database.

To support the database recovery function, which depends on rolling forward the log files, the log must be periodically copied to another location. This is a sequential read activity that must occur simultaneously with the continuous sequential write activity to the log.

DSS

A DSS workload is characterized by heavy sequential reads that use large block sizes from the data files and little or no activity against the application database log file. There is usually heavy sequential read/write use of tempdb when large intermediate datasets cannot be stored in memory and must be sorted. There might be phases of high CPU usage, when the working data is able to fit in memory.

Mixed OLTP and DSS

When both OLTP and DSS workloads are run concurrently, there can be physical and logical contention. For example, a DSS query might be reading the same data that an OLTP transaction is updating. This is precisely the scenario that the new **Snapshot isolation** feature was created for. By keeping row versions of the data in tempdb, SQL Server 2005 can avoid reader-writer blocking. The impact on the hardware is then a heavier use of tempdb.

Contention for CPU resources can arise within a single instance. In this case, it can be resolved by using the Soft NUMA feature to partition CPUs by workload in a single SQL Server instance as described in 3.4.4, “Soft NUMA” on page 46.

4.6 Performance analysis process

In this section, we describe the process of analyzing performance. This analysis process includes the practice of keeping a performance baseline, how to monitor the hardware resources on a server that is running SQL Server, and the other SQL Server specific tools that are available.

Important: Although this paper focuses on hardware, we have observed that, although it is possible to make improvements in the 10% to 50% range by tuning hardware, it is also possible to make orders of magnitude improvement by performing tuning activities at the database and application levels.

4.6.1 Performance baseline

Keeping a performance baseline is a basic best practice. SQL Server database administrators (DBAs) often create a separate database that contains baseline information about how an application runs hour to hour, day to day, week to week, and month to month. Then, they can query this database to find averages and trends to create a profile of their applications when they are running well.

You can learn the profile of your application by determining:

- Typical transaction rate
- Typical I/Os per second; read and write to database, log, and tempdb files; and the random/sequential pattern

- Typical CPU utilization, user and kernel
- Typical memory utilization

The concept of keeping a *baseline* is very general. It includes keeping performance logs and Excel spreadsheets (in reality, any repository of historical measurements) on any component, including the operating system, the database management system, the application, and the hardware. The value of keeping this historical information is that when something changes unexpectedly in performance, you can refer to this baseline data and find clues in what has changed from then and now. This can lead more quickly to a resolution of the problem, than if you did not have this baseline data.

One procedure for collecting a performance baseline is:

1. Create a Counter Log, called Baseline_A, using the Performance Monitor GUI with a few key counters that monitor CPU, memory, disk, and network.
2. Run the command in Figure 4-1 from the command line, all on one line.

```
logman update counter Baseline_A -v mmddhhmm -b 1/1/2015 0:00:00 -e 1/1/2015
23:59:59 -r -cnf 4:00:00 -si 00:15 -f tsv -o "C:\Perflogs\Baseline_A" -u
Administrator *
```

Figure 4-1 Settings for capturing baseline performance data

After you have run the command and when you start the Counter Log, your statistics are collected every 15 seconds and they are saved in a log file named with the current date time stamp in the C:\Perflogs directory. It closes the current log file every four hours and creates a new log file. It runs until you stop it. You can create another Counter Log, called SQL_A, and include some of the SQL Server counters such as **Transactions/sec** for your databases. SQL_A might include more counters than Baseline_A. This way, you have a relatively small counter log for quick diagnosis and a larger one for more complete analysis.

4.6.2 Hardware resources

In this section we discuss monitoring the four hardware resources (CPU, memory, disk, and network) from the perspective of SQL Server 2005 running its typical workloads.

CPU

If the total processing percentage (**% Processor Time**) is higher than 80% for sustained periods, then you might have a CPU bottleneck. Normally the kernel mode time (**% Privileged Time**) is low for SQL Server. If it is high, there might be a problem with a disk driver or the disk subsystem.

To see if SQL Server has work queued that could run, but is waiting for a busy CPU, you must check inside SQL Server. This is because SQL Server does not queue its work to the Windows operating system. SQL Server has a user-mode scheduler in which it queues tasks that are waiting.

This query results in an instantaneous view of the number of tasks queued in SQL Server that can be run:

```
select sum(runnable_tasks_count)
  from sys.dm_os_schedulers
 where scheduler_id < 255
```

Figure 4-2 Tasks that can be run in SQL Server

If SQL Server frequently has a non-zero number of runnable tasks, then adding additional processors will likely result in more throughput.

Memory

The best way to monitor memory usage is to use this Windows performance counter: **SQLServer:BufferNode**.

For a server that is running only SQL Server, there should be no paging. That is, **Memory\Pages/sec** should be zero. Most of the memory that SQL Server uses is allocated for the Buffer Pool, which consists of 8 KB pages. **Buffer Manager\Database pages** gives the number of buffer pages that contain database content.

Take the number of buffer pages, multiply it by 8 KB, and compare it to the total amount of physical memory in the server. This gives you an indication of how much of your physical memory is actually being used. Use `DBCC MemoryStatus` to obtain a snapshot of how memory is allocated on each of the NUMA nodes and how it is consumed by the pools and caches in SQL Server.

Memory usage in SQL Server 2005 is very complex. However, there is a simple procedure for determining whether a workload will benefit from additional memory. It is based on the observation that most applications respond to increased memory according to the general graph shown in Figure 4-3.

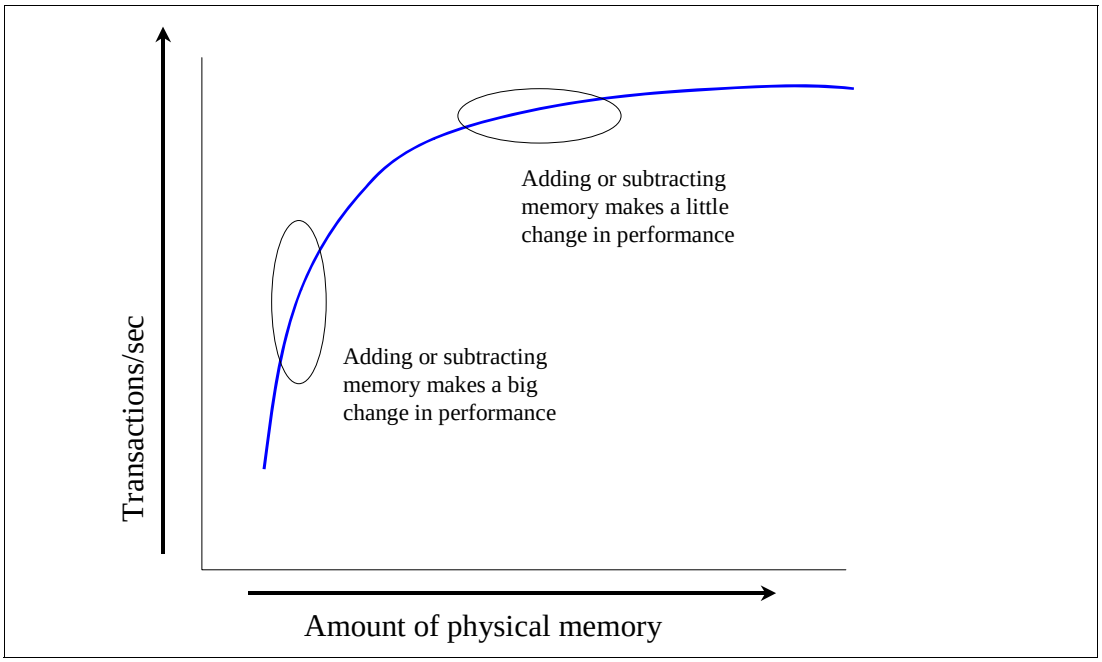


Figure 4-3 How more memory affects performance

The procedure is to measure the application performance with two amounts of memory, your current amount of memory, and a small amount less. If the difference in performance is not

great, then that means you are at the top part of the curve and adding more memory does not improve performance much. On the other hand, if the difference in performance is great, then you are at the bottom part of the curve and adding memory is likely to improve performance, up to the point where the curve begins to flatten out.

Disk

If CPU utilization is low, but response time or run time is not okay, you might have a disk bottleneck. The **Physical Disk** Perfmon counters might provide the insight needed for tuning most of the sequential I/O workloads. However, in the OLTP workload, the disk load of the log and checkpoint processes are adjusted by SQL Server and so are invisible to Perfmon. When writing to the log, the disk queue length is kept at 1. Likewise, the checkpoint is controlled so that it does not swamp the disk and severely reduce throughput. In this case, use the query in Figure 4-4 to capture reads, writes, and I/O stalls by database file.

```
select m.name, v.*, m.physical_name
  from sys.dm_io_virtual_file_stats (null, null) v
      ,sys.master_files m
 where v.database_id = m.database_id
   and v.file_id     = m.file_id
```

Figure 4-4 SQL Server I/O statistics by file

Using this information, you can compute reads/sec, writes/sec, read and write block sizes, and average milliseconds of wait time (I/O stalls) for read and write. If these wait times are large, putting the files in faster disk arrays can improve database performance.

Using one of the new dynamic management views (DMV), it is now possible to obtain I/O statistics by individual table and index, which can be used to make decisions about dividing up the tables into file groups and for application level tuning. See the query in Figure 4-5. See also “Dynamic management views” on page 70.

```
select o.name, i.
  from sys.dm_db_index_operational_stats(db_id, null, null, null) i
      ,sys.objects o
 where i.object_id = o.object_id
   and i.object_id > 100
```

Figure 4-5 SQL Server I/O statistics by table/index

Network

Network performance is rarely an issue with SQL Server. There is a new Perfmon SQL Server counter object, **Wait Statistics**, that includes the counter **Network IO waits**, which can prove useful. However, be aware that network waits can occur because the client is not consuming the packets fast enough, which would not indicate a bottleneck in the server hardware.

4.6.3 Other diagnostic and performance tools

The tools discussed in this section are useful for troubleshooting and tuning performance.

SQL Trace and SQL Profiler

SQL Trace and SQL Profiler are related tools. SQL Trace is the name for the tracing facility that is configured by using a set of system stored procedures. This facility does not drop trace

events when the server is under stress (unlike SQL Profiler). SQL Profiler is a GUI application front end for SQL Trace.

SQL Trace/Profiler is a good tool to use to find poorly performing queries, to determine the cause of deadlocks, or to collect a sample workload to replay for stress testing. A trace template can be used to collect workloads in the format needed by the Database Engine Tuning Advisor.

Database Engine Tuning Advisor

Database Engine Tuning Advisor (DTA) replaces the Index Tuning Wizard from SQL Server 2000. DTA recommends changes in indexes and partitioning for an existing database based upon a typical workload (a SQL trace) that is collected from the production system. You can use DTA to identify indexes that are not being used by your application. These indexes can be dropped because their maintenance creates unnecessary overhead. You can use DTA to identify new indexes that currently do not exist. These indexes can be added, potentially reaping large performance benefits.

Dynamic management views

One of the design goals of SQL Server 2005 is for users to be able to effectively troubleshoot and tune their databases. DMVs provide this visibility to the internal workings of SQL Server 2005.

The server wide DMVs are:

- **sys.dm_db_**

  This DMV is for databases that provide space and usage information by file, index, partition, task, and session. Note that queries that return index fragmentation statistics can cause intensive I/O on that index.

- **sys.dm_exec_**

  This DMV is for execution related information, including query plans, cursors and sessions.

- **sys.dm_io_**

  This DMV provides I/O information for data and log files, pending I/O information, and tape status.

- **sys.dm_os_**

  This DMV is for SQLOS related information, including memory, scheduling, tasks and wait statistics.

- **sys.dm_tran_**

  This DMV is for transaction-related information including active transactions, locking, snapshots, and the version store. Note that selecting row version information could return many rows and be resource intensive.

These are the component specific DMVs:

- **sys.dm_clr_** for the Common Language Runtime feature
- **sys.dm_db_mirroring_** for the Database Mirroring feature
- **sys.dm_fts_** for the Full Text Search feature
- **sys.dm_qn_** for the Query Notifications feature
- **sys.dm_repl_** for the Replication feature
- **sys.dm_broker_** for the Service Broker feature

Catalog views expose static metadata that describes all the user viewable objects in a SQL Server instance. For example, `sys.master_files` gives all the database file names that are known to the SQL Server instance. By combining catalog views with dynamic management views, you can create queries that interpret the internal data for troubleshooting and performance analysis, as we did in Figure 4-4 on page 69.

SQLdiag

The SQLdiag utility, which has its roots in the Microsoft Product Support Services group (actually first written by Ken Henderson), is a general purpose diagnostic tool. It collects performance logs, event logs, SQL traces, SQL blocking data, and SQL configuration data.

An especially useful mode for using this tool is to start it, reproduce an issue, and shut it down. The captured information that results from this mode can be analyzed for troubleshooting. You can extensively customize the data SQLdiag collects.

Query hints and plan guides

Query hints are not new to SQL Server 2005. They can be used to force the query optimizer to choose a specific query plan. They are useful when the optimizer occasionally does not choose the most efficient plan. Plan guides are an extension of this.

Using a plan guide, it is possible to modify a query, providing query hints, without changing the text of the query. This is useful when you have a third party application and do not want to or are unable to modify the code.

Abbreviations and acronyms

ACPI	advanced control and power interface
AMD	Advanced Micro Devices
API	application programming interface
AWE	Address Windowing Extensions
BCP	Bulk Copy Program
BIOS	basic input output system
BMC	baseboard management controller
BPR	Business Process Re-engineering
CD	compact disk
CD-ROM	compact disk read only memory
CLR	Common Language Runtime
CPU	central processing unit
DAC	dual address cycle
DB	database
DBA	database administrator
DBCC	database consistency checker
DIMM	dual inline memory module
DMV	dynamic management views
DP	dual processor
DRAM	dynamic random access memory
DSS	decision support system
DTA	Database Tuning Advisor
DTC	Distributed Transaction Coordinator
DTS	Data Transformation Services
DWORD	double word
ECC	error checking and correcting
EPIC	Explicitly Parallel Instruction Computing
ETL	extract, transform, and load
FC	Fibre Channel
FSB	front-side bus
GB	gigabyte
GP	general purpose
GUI	graphical user interface
HBA	host bus adapter
HPC	high performance computing
HR	human resources
HW	hardware
I/O	input/output
IBM	International Business Machines Corporation
IBM	International Business Machines Corporation
IIS	Internet Information Services
IP	Internet Protocol
IT	information technology
ITSO	International Technical Support Organization
IXA	Integrated xSeries Adapter
KB	kilobyte
MB	megabyte
MCDBA	Microsoft Certified Database Administrator
MDAC	Microsoft Data Access Components
MESI	modified exclusive shared invalid
MIOC	Memory and I/O Controller
MP	multiprocessor
MSCS	Microsoft Cluster Server
MXE	modular expansion enclosure
NUMA	Non-Uniform Memory Access
OLAP	online analytical processing
OLTP	online transaction processing
OS	operating system
PAE	Physical Address Extension
PC	personal computer
PCI	peripheral component interconnect
POC	proof of concept
RAID	redundant array of independent disks
RAM	random access memory
RBS	redundant bit steering
ROLAP	Relational Online Analytical Processing
RSA	Remote Supervisor Adapter
SAN	storage area network
SAS	Serial Attached SCSI
SCSI	small computer system interface
SL	SlimLine
SMP	symmetric multiprocessing

SPORE	Server Proven Opportunity Request for Evaluation
SQL	structured query language
SQLOS	SQL Operating System
SRAT	Static Resource Allocation Table
TB	terabyte
TCP	Transmission Control Protocol
URL	Uniform Resource Locator
USB	universal serial bus
VAS	virtual address space
WSRM	Windows System Resource Manager
XML	Extensible Markup Language

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this Redpaper.

IBM Redbooks

For information about ordering these publications, see “How to get IBM Redbooks” on page 76. Note that some of the documents referenced here may be available in soft copy only.

- *Windows Server 2003, Datacenter Edition on the IBM @server xSeries 445*, REDP-3700
- *VMware ESX Server: Scale Up or Scale Out?*, REDP-3953
- *Understanding IBM @server xSeries Benchmarks*, REDP-3957
- *Introducing Windows Server x64 on IBM @server xSeries Servers*, REDP-3982
- *Tuning IBM @server xSeries Servers for Performance*, SG24-5287
- *Planning and Installing the IBM @server X3 Architecture Servers*, SG24-6797

Other publications

These publications are also relevant as further information sources:

- SQL Server 2005 Books Online (available when you install SQL Server 2005)
- SQL Server Magazine, http://www.windowsitpro.com/SQLServer

Online resources

These Web sites and URLs are also relevant as further information sources:

- Microsoft SQL Server 2005 product overview

  http://www.microsoft.com/sql/2005/productinfo/overview.mspx
- Microsoft SQL Server 2005 home page

  http://www.microsoft.com/sql/2005/
- Slava Oks' WebLog

  http://blogs.msdn.com/slavao/
- Ken Henderson's WebLog

  http://blogs.msdn.com/khen1234/
- Professional Association for SQL Server (PASS)

  http://www.sqlpass.org
- Information and user community on SQL Server

  http://www.sqlservercentral.com
- SQL Server Worldwide Users Group

http://www.sswug.org

- Microsoft paid- and self-support options

  http://www.microsoft.com/sql/support

How to get IBM Redbooks

You can search for, view, or download IBM Redbooks, IBM Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads:

ibm.com/support

IBM Global Services

ibm.com/services

SQL Server 2005 on the IBM @server xSeries 460 Enterprise Server

Redpaper

Describes how "scale-up" solutions suit enterprise databases

Explains the XpandOnDemand features of the xSeries 460

Introduces the features of the new SQL Server 2005

The xSeries 460 is the number one x86 scalable server in the industry. The combination of the x460 and Microsoft SQL Server 2005 creates a perfect match for addressing needs for more database performance. The x460 is an effective four-way server, but provides optimal scalability for those customers who need room to grow.

One of the key markets for the x460 is database applications such as SQL Server 2005 ("Yukon") SQL Server can effectively take advantage of the 32 processors and 512 GB of RAM available with a large x460 solution, providing customers with the performance and scalability needed to address their complex database workload needs.

The x460 is an example of *scale-up* technology. With the x460, when customers want to increase the capacity of their server infrastructure, they can simply add nodes to the x460 to increase the number of CPUs and the amount of installed memory. The advantage of scaling up is simpler management and control.

This Redpaper describes how the "scale-up" features of the xSeries 460 and SQL Server 2005 are an ideal fit. The paper discusses the key features of each, including how the multi-node scalability is designed to ensure a near-linear scalability from a single-node four-way server all the way to an eight-node, 32-way complex. It also offers performance tuning advice directly from the IBM performance labs.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks