Best Practices for Tuning Microsoft SQL Server on the HP ProLiant DL980

Technical white paper

Table of contents

Introduction .................................................................................................................................. 2
Prerequisite Reading and System Configuration ....................................................................... 2
Server Recommendations ......................................................................................................... 3
   BIOS Settings ......................................................................................... 3
   Hyper-threading Considerations ......................................................................................... 4
   Card Placement: Understanding Processor Enumeration and I/O Slots ....................... 4
Operating System Recommendations ...................................................................................... 5
   Windows (all versions) ............................................................................................... 5
   Windows Server 2008 SP2 and Windows Server 2008 R2 ........................................... 6
   Windows Server 2008 R2 ......................................................................................... 6
Storage Recommendations ...................................................................................................... 7
   Provide Sufficient I/O and Storage to Run SQL Server ............................................. 7
   Use the Recommended Storport Driver with Fibre Channel Host Bus Adapters ........ 8
   Verify Maximum Queue Depth is Greater Than or Equal to the Number of Spindles .... 9
   Verify Switch Port Speed is set to Maximum .................................................................. 9
Network Recommendations ..................................................................................................... 9
   Configure Receive Side Scaling (RSS) ............................................................................. 9
   Enable Options for Offload Processing ......................................................................... 11
SQL Server Tuning Recommendations .................................................................................... 11
   Use Startup Options to Lock Pages in Memory ......................................................... 11
   Use SQL Server Startup Flags ................................................................................... 12
   Ensure SQL Server Starts Up Immediately After System Boot .................................. 12
   Enable Write Caching on the Log Disk ......................................................................... 12
   Segregate Network Processing from SQL Processing .................................................. 13
   Use Data Compression Appropriately .......................................................................... 14
Application-dependent SQL Parameter Recommendations .................................................. 14
   For OLTP Workloads ............................................................................................... 14
   For Business Intelligence Workloads ......................................................................... 17
Conclusion ............................................................................................................................. 18
For More Information ............................................................................................................ 19
Documentation Feedback ...................................................................................................... 19
Introduction

The HP ProLiant DL980 G7 Server stands at the pinnacle of HP’s line of scale-up x86 servers. The DL980 not only supports the most recent Microsoft Windows 64-bit operating systems, but also incorporates all of the following technologies:

- The latest Intel Xeon extended 64-bit processors
- Quick Path Interconnect technology high speed links supporting a Non Uniform Memory Access architecture
- Industry standard I/O architecture based on PCI-e (and optionally, PCI-X) buses
- HP PREMA Architecture (for more information, see the technical overview at: http://h20195.www2.hp.com/V2/GetPDF.aspx/4AA3-0643ENW.pdf).

Because of this powerful synergy of proven technologies and engineering, the HP line of scale-up platforms is exceptionally well-suited for workloads requiring high-performance processing, such as business intelligence and other line-of-business applications.

Large databases at the core of these workloads quickly reach the 32-bit architectural limit of x86-based systems. But the recent generation of HP servers implementing Windows 64-bit operating systems and architecture offer much greater headroom and can therefore take advantage of scale-up x86 platforms. For example, the HP ProLiant DL980 G7 running Microsoft Windows Server 2008 R2 supports a maximum of 2 terabytes of main memory, 160 logical processors (with the new Intel Xeon E7 family), and 16 PCI cards.

The HP ProLiant DL980 is the ideal platform to capitalize on the advantages of Microsoft SQL Server 2008 (x64) and SQL Server 2008 R2. SQL Server 2008 R2 delivers increased security, scalability, and availability to enterprise data and analytical applications, while making them easier to build, deploy, and manage. Optimized for 64-bit addressing, SQL Server 2008 R2 takes advantage of advanced memory addressing capabilities for essential resources such as buffer pools, caches, and sort heaps, thereby reducing the need to perform multiple I/O operations to move data in and out of memory from disk. This greater processing capacity without the penalties of I/O latency means greater application scalability.

Although SQL Server 2008 R2 runs “out-of-the-box” on HP scale-up x64 servers, we recommend some tuning guidelines to maximize performance and take full advantage of the capabilities of this platform. This document describes configuration settings that represent current “best practices” in tuning an HP scale-up x64 system for Microsoft Windows Server 2008 R2 (or Windows Server 2008) and SQL Server 2008 R2 (or SQL Server 2008).

Prerequisite Reading and System Configuration

This document focuses on SQL Server tuning for the HP ProLiant DL980 G7. However, before doing any SQL Server tuning work, users must install and configure their HP ProLiant DL980 G7 server with either Windows Server 2008 SP2 or Windows Server 2008 R2 using the recommendations given in the HP whitepaper, Best Practices When Deploying Microsoft Windows Server 2008 R2 or Microsoft Windows Server 2008 SP2 on HP ProLiant DL980 G7 Servers. This document is available at: http://h20000.www2.hp.com/bc/docs/support/SupportManual/c02577023/c02577023.pdf.

Note

- Windows Server 2008/R2 is used throughout this document to describe features common to both Windows Server 2008 and Windows Server 2008 R2.
- SQL Server 2008/R2 is used to describe features common to SQL Server 2008 and SQL Server 2008 R2. When the discussion refers to a specific version of either, that version is explicitly stated.
It is particularly important that you install the following software on the system before attempting any SQL Server tuning:

- **ProLiant Support Pack (PSP)** – the HP drivers and software for the platform
- **Smart Update** – the HP-tested set of Microsoft updates and hotfixes for the OS

In addition, installation of the HP System Providers (version 8.7.1 – June, 2011 or later) is highly recommended in systems running Windows Server 2008 R2. In these systems, the logical processors may not be assigned in APIC order, but instead re-assigned based on system measurements of Non Uniform Memory Architecture (NUMA) distances during kernel group formation. Installation of the HP System Providers ensures that the Windows logical processors are set optimally during kernel group configuration.

Finally, we also encourage users to read all of the relevant Microsoft tuning whitepapers. References and links to these papers are found at the end of this document.

**Server Recommendations**

The HP ProLiant DL980 scale-up x86 server has a processor-based architecture where each CPU socket is presented to the OS as a separate NUMA node. The cores on each socket appear to the OS as separate CPUs, and with Hyper-threading enabled, each core appears as two Logical Processors (LPs). Each CPU also has onboard local memory controllers that manage the memory attached to that processor. Since accessing local memory is always much faster than accessing remote memory, you must take some steps to maximize local access and minimize remote access in order to achieve the best performance from a NUMA server.

**BIOS Settings**

The ROM-Based Setup Utility (RBSU) is used to set certain configuration parameters at the BIOS or hardware level. Most of the default settings are fine and the system will run satisfactorily with them, but extensive testing has shown that changing some of the default settings yields higher performance with certain workloads.

You can access the RBSU Setup utility by pressing **F9** at the ProLiant splash screen during the boot process. Listed below are the BIOS settings that are critical for peak performance. These settings should be verified at the first available opportunity and changed as shown, if necessary:

- **System Options > Processor Options > Hyper-threading > Enable or Disable** (see section on Hyper-threading below)
- **Power Management Options > HP Power Profile > Custom**
- **Power Management Options > HP Power Regulator > OS Control**
- **Power Management Options > Advanced Power Management Options > Minimum Processor Idle Power State > C1E** (reduces power when possible, for performance + power savings) or **NO C-states** (when performance is highest priority; power saving functionality is ignored)
- **Advanced Options > Advanced Performance and Tuning Options > HW Prefetch > Enabled**
- **Advanced Options > Advanced Performance and Tuning Options > Adjacent Sector Prefetch > Enabled**
- **Advanced Options > Advanced System ROM Options > Address Mode 44-bit > Enabled** (this setting is for Windows Server 2008 R2 only, and mandatory if ≥ 1TB RAM. On Windows Server 2008 SP2, this setting should remain Disabled, since that OS uses a 40-bit address mode.)
**Note**
However these settings are configured, Microsoft Windows Server operating systems always recognize the system’s **logical processors**. These are the total number of **cores** (when Hyper-threading is OFF), or the total number of processor **threads** (when Hyper-threading is turned ON).

**Hyper-threading Considerations**

With Intel Xeon 65xx and 75xx processors, each NUMA node (processor socket) can contain up to 8 CPU cores, and with the new Intel E7 processor family, up to 10 cores. To extend processing capabilities even further, you can enable Intel Hyper-threading and each core will appear as two logical processors to the OS. Likewise, a single physical processor with 10 cores appears as 20 functional processors to both the OS and the DBMS.

HP ProLiant DL980 G7 Servers ship with Hyper-threading enabled by default. Depending on the workload, Hyper-threading can increase system performance by up to 40% (20% is typical). But in some cases it can increase contention and thereby cause a drop in performance. Or if the workload does not have enough parallelism, Hyper-threading can actually increase response time, since each Hyper-threaded CPU is effectively slower than a physical core. In short, you should always test your particular workload with and without Hyper-threading before committing to its use.

**Card Placement: Understanding Processor Enumeration and I/O Slots**

The following diagram illustrates the physical layout of I/O slot locations and other components, relative to the DL980 processor boards.

![Diagram of I/O slots vs. processor boards](image)

**Figure 1:** I/O slots vs. processor boards

Embedded adapters (like network card NC375i, video, and so on) are connected to processors 0 (zero) and 1.
The above figure, when used in combination with the following table, should help you determine the best location for your adapters according to each card’s characteristics (such as the number of lanes), with the ultimate goal of distributing the load equally across all processors.

Table 1: PCIe slot, type, and capability

<table>
<thead>
<tr>
<th>Adapter</th>
<th>PCIe Slot #</th>
<th>Physical Connector</th>
<th>IOH</th>
<th>With PCI-X PCIe expander (588139-B21)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCIe Gen 2 (x8 Electrical)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>x16</td>
<td>SublO</td>
<td>64 bit PCI-X 100 Mhz</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>x16</td>
<td>SublO</td>
<td>x16 Electrical (on PCI-X SublO only)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>x16</td>
<td>SublO</td>
<td>Slot does not exist for PCI-X Sub Assy</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>x16</td>
<td>SublO</td>
<td>x16 Electrical (on PCI-X SublO only)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>x16</td>
<td>Main IO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>x16</td>
<td>Main IO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>x16</td>
<td>LPIO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>x16</td>
<td>LPIO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>x16</td>
<td>LPIO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>x16</td>
<td>LPIO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCIe Gen 2 (x4 Electrical)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>x8</td>
<td>SublO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>x8</td>
<td>Main IO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>x8</td>
<td>Main IO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>x8</td>
<td>Main IO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>x8</td>
<td>LPIO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCIe Gen 1 (x4 Electrical)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>x8</td>
<td>SublO</td>
<td>64 bit PCI-X 100 Mhz</td>
<td></td>
</tr>
</tbody>
</table>

*Note:* This table describes slot characteristics using HP’s “DL580 G7 PCI Express Kit” (HP Ref: 588137-B2, called “SublO” in this document) and HP’s “DL980 LP PCIe I/O Expansion Module” (HP Ref: AM434A, called “LPIO” in this document). The last column only applies when using the optional “DL580 G7 PCI-X Combo Express Kit (HP Ref: 588139-B21) for the SublO (instead of 588137-B2).

It is far better, performance-wise, to install cards into slots best suited for their characteristics. If for example an adapter is a PCIe x8 (8 lanes), best throughput is obtained in the available x8 slots. When impossible to do that, keep in mind that the PCIe x4 slots (with 4 lanes) are generally preferred for lower performance adapters.

Operating System Recommendations

The HP ProLiant DL980 scale-up x86 server supports both Windows Server 2008 and Windows Server 2008 R2. Both of these operating systems are NUMA-aware and can fully leverage the resources of large x86 servers. While Windows Server 2008 can manage up to 64 logical processors, Windows Server 2008 R2 can go even further (up to 256 logical processors), providing unprecedented workload size, scalability, and across-the-board availability and reliability.

Windows (all versions)

Recommendations for all versions of Windows include the following:

1. Do not start Windows with the /debug boot option enabled. While useful for troubleshooting, the /debug option affects performance.
2. Disable all unnecessary services. Many of the services started by default are unnecessary. To disable these services, right-click on My Computer > Manage > Services and Applications > Services,
disable unused services such as Active Directory. If you are unsure of a service, err on the side of caution and leave it enabled. You may need to experiment to ensure optimum functionality.

**Note**
Windows Server 2008 supports a maximum of 64 logical processors. For this reason, on DL980s with 8 processors and 8 cores or more each, it is often better to disable Hyper-threading. In order to fully utilize all of the processors in these systems it is preferable to run Windows Server 2008 R2 instead. You can then determine whether your workload performs better with Hyper-threading enabled or disabled on Windows Server 2008 R2.

For more information about this issue, refer to the HP whitepaper, Best Practices When Deploying Microsoft Windows Server 2008 R2 or Microsoft Windows Server 2008 SP2 on HP Proliant DL980 G7 Servers, at the link given previously in this document.

---

**Windows Server 2008 SP2 and Windows Server 2008 R2**

Some of the Windows Server OS registry entries related to system behavior can affect performance. Refer to the Microsoft performance tuning guidelines for more information, using the links provided at the end of this document. We recommend that you test these registry modifications in your environment to see how much benefit they deliver, and then decide whether to implement them. These performance tuning guidelines are intended for advanced administrators only, so consider them carefully.

Possible registry modifications include the following:

- **Allocate PFN database sparsely and with small pages:**
  \HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\LargePageMinimum = REG_DWORD 0xffffffff

- **Verifier.sys - Disable random verification of drivers for debugging:**
  (normally disabled by default, but verify; performance is adversely affected if enabled)
  \HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\DontVerifyRandomDrivers = REG_DWORD 0x00000001

- **Disable PartMgr/FltMgr/VolMgr ioctl counters:**
  (implement only if no applications use these counters)
  \HKLM\SYSTEM\CurrentControlSet\Services\PartMgr\EnableCounterForIoctl = REG_DWORD 0x0

- **Disable NTFS creation of 8.3 filenames:**
  (make sure this is compatible with applications)
  \HKLM\SYSTEM\CurrentControlSet\Control\FileSystem\NtfsDisable8dot3NameCreation = REG_DWORD 0x00000001

---

**Windows Server 2008 R2**

As stated previously, Windows Server 2008 R2 is the recommended OS for the DL980 because it can support up to 160 logical processors when Hyper-threading is enabled on servers with 8 processors and 10 cores per processor. To complement the previously recommended global BIOS settings, there is an additional High Performance Power option that is available when using Windows Server 2008 R2 with SQL Server.

To select this High Performance power option in Windows Server 2008 R2, follow these steps:

1. Click **Start > Control Panel**.
2. Select **View > Large Icons**.
3. Click **Power Options**.
4. Select **High Performance**.
In addition, the following power configuration commands allow you to disable C-states in Windows (similar to choosing the “NO C-states” option at the BIOS level, as described earlier). These commands do not require a system reboot:

- `powercfg -setacvalueindex scheme_min sub_processor 5d76a2ca-e8c0-402f-a133-2158492d58ad 1`
- `powercfg -setacvalueindex scheme_max sub_processor 5d76a2ca-e8c0-402f-a133-2158492d58ad 1`
- `powercfg -setacvalueindex scheme_balanced sub_processor 5d76a2ca-e8c0-402f-a133-2158492d58ad 1`
- `powercfg -setactive scheme_current`

**Note**

To revert any power configuration command back to its previous state, simply rerun the command with a 0 (zero) at the end instead of a 1. For example:

```
powercfg -setacvalueindex scheme_min sub_processor 5d76a2ca-e8c0-402f-a133-2158492d58ad 0
```

---

**Storage Recommendations**

Storage is an important factor when considering SQL Server workloads, and appropriate sizing is required. But that in turn requires a good understanding of an application’s I/O characteristics, such as the frequency of reads and writes, and the amount of data typically moved in those operations. Specific guidelines for calculating the optimal storage size for your particular application are beyond the scope of this document. Instead, we want to provide you with some general, workload-dependent recommendations.

**Provide Sufficient I/O and Storage to Run SQL Server**

A common mistake in setting up a SQL server is to under-provision the I/O links to storage. For example, although a modern storage array can function with a single fibre link to the system (an extreme case) this clearly does not provide adequate performance. Therefore, it is important to add enough fibre channel or SAS links to handle the bandwidth needed by the I/O load.

Keeping in mind that different storage components have specific characteristics affecting their performance, you can use the following rules of thumb to size your I/O and storage configuration:

**For Physical Disks/Hard Disk Drives (HDDs)**

I/O rates and service times should be monitored periodically to ensure the storage farm is not being saturated. When setting up the system, run an I/O load tool to fully load the farm as expected in production. Make note of the service times, max read and write rates, and so on. This data comprises your Unloaded Storage Metric.

When the service times grow two to three times greater than your Unloaded Storage Metric, that is a clear indication your storage farm is being overloaded and the system I/O performance is starting to degrade, probably quickly. For example, if the storage farm consists of 2.5-inch 15KRPM SAS drives, and a 60/40 R/W mix is applied to the drives when no other work is applied, you might see Read service times of 4ms and Write times of 7ms. When these numbers grow from 8 to 12ms for Read, and from 14 to 21ms for Write, the system is nearing saturation and the storage farm should be reconfigured in order to reclaim some performance. This can be accomplished by adding spindles and spreading out the data base, or changing the storage to a flash technology, like SSDs.

Note that the numbers given here are examples only. Actual numbers seen may be higher or lower in the unloaded state.
For Solid State Drives (SSDs)

SSDs, HP I/O Accelerators, and Violin Memory Storage arrays can provide superb random I/O throughput by eliminating the rotational latency of spinning media. Standard 15K RPM disks can provide 100 to 200 IOPS per disk. SSDs can provide 50K IOPS per disk. HP I/O Accelerators can provide 100K IOPS per card. And Violin Memory Arrays can provide 220K IOPS per array.

You should consider implementing a ‘tier-0’ of one or more of these non-spinning storage technologies to enhance your performance for the highest volume tables or files. But all of these media types can be over-subscribed, with response times increasing exponentially as a result. So it is important to choose the correct technology for each tier of storage, balancing performance and cost.

Fibre Channel transfer speeds

Fibre Channel transfer speeds are as follows:

- 2 Gbps fibre channel transfer rate = ~180 MB/sec
- 4Gbps transfer rate = 350MB/sec
- 8Gbps transfer rate = 680MB/sec

Also be aware that for 8KB I/O rates, many fibre channel host bus adapters (HBAs) have a limit on throughput below the fibre channel bandwidth.

SCSI transfer speeds

SCSI transfer speeds are as follows:

- SAS 3Gbs transfer rate = ~300MB/sec
- SAS 6Gbs transfer rate = ~600MB/sec

PCIe slot transfer speeds

PCIe slot transfer speeds are as follows: (x# indicates the number of lanes)

- Gen1 → x4 Slot #1 = 800 MB/sec
- Gen2 → x4 Slots (#4, 7, 8, 10, 14) = 1.6 GB/sec
- Gen2 → x8 Slots (#2, 3, 5, 6, 9, 11, 12, 13, 15, 16) = 3.2 GB/sec

Having adequate storage (HDDs or SSDs) will definitely help sustain the high I/O required by a demanding SQL application workload.

You must also consider the characteristics of the workload. Online Transaction Processing (OLTP) workloads typically perform small, random I/O operations, while Decision Support (DS) workloads (large queries) perform fewer but larger I/O operations. With OLTP, you are more concerned with the I/O rate than the bandwidth; however the opposite is true for DS. Obviously, every application is different, and the I/O loads imposed on the system by those applications are unique.

The Windows Performance Monitor utility (perfmon.exe) provides basic data about I/O rates and throughput. Use this utility to monitor running applications and obtain the information necessary to design your I/O configuration.

In addition to the I/O, you must also configure the storage system. Configuration of the storage system is beyond the scope of this document. But by keeping the preceding rules of thumb in mind, you can configure the I/O to achieve optimum system performance and gain valuable information about your storage requirements.

Use the Recommended Storport Driver with Fibre Channel Host Bus Adapters

Although this may seem obvious, you must use the driver recommended for your storage environment to obtain the best performance with fibre channel HBAs. Vendors generally qualify an optimized set of compatible versions of
firmware and driver components. Depending on the system layout, it is often appropriate to use switch zoning or other methods of segmentation. Multiple paths may improve data availability and eliminate single points of failure in SAN components, but they also require multi-path software components running on Windows. And finally, storage vendors often develop their own Device Specific Modules (DSMs). These DSMs should be used whenever possible because they are optimized for your storage platform.

Verify Maximum Queue Depth is Greater Than or Equal to the Number of Spindles

With Emulex HBAs, use the One Command utility to set the queue depth per target or per Logical Unit Number (LUN). The default maximum queue depth (QueueDepth) is 32(dec), or 0x20h. Use the One Command utility to change this value to a number greater than or equal to the number of spindles seen by that HBA.

With Qlogic HBAs, use the SanSurfer utility to change the Qlogic firmware BIOS setting for execution throttle in NVRAM to be equal to or greater than the number of physical drives seen by that HBA (default = 16(dec), or 0x10h).

Be aware that these same guidelines can apply to fibre channel RAID controller as well. Many RAID controllers have a configuration option to return a busy status when a queue depth limit is exceeded. You should verify that these options are appropriately configured, based upon the number of disks in the LUN.

Note

While increasing the queue depth often provides benefits for a given server, it can also have detrimental effects on other servers utilizing the same storage array. You should refer to the HP SAN Design Guide and consider the potential impact to other servers if you increase the queue depth beyond the values recommended in the HP SAN Design Guide.

Verify Switch Port Speed is set to Maximum

Use the diagnostic LEDs to verify that the switch is operating at its maximum speed. To set the port speed on a Brocade switch, access the switch using its IP address via telnet or web browser. Autosense for HBAs should set the speed to its required value (or explicit speed value). On Qlogic switches, use the SanSurfer Switch Manager utility to set the port speed (Autosense is the default).

To manually configure the speed of a port, use the `portcfgspeed` command as follows:

```
Switch:admin> portcfgspeed port [speed]
```

The `speed` parameter is defined as: 0 = AN (Auto Negotiate). Use the interface Help tool for more information about other possible values for your switch.

Network Recommendations

SQL network traffic packets are typically small, so the maximum achievable bandwidth of a network link often cannot be fully utilized. We recommend that you not exceed 15,000 packets/second on a Gigabit link. Network Interface Controller (NIC) teaming is an option, but does require additional overhead that could affect performance.

With 10Gb links, the limiting factor is usually the CPU power available for handling interrupts. For this reason, there is a need to enable some more advanced features using Receive Side Scaling, or RSS.

Configure Receive Side Scaling (RSS)

RSS is a technique for distributing network traffic received from a single NIC over multiple CPUs while at the same time preserving sequential delivery to TCP connections. It allows for better scalability by varying the number and range of processors allotted to each network interface.
Setting Up RSS

Receive-side scaling is enabled by default in Windows Server 2008 and Windows Server 2008 R2. In order to take advantage of RSS, network drivers must be written with RSS capabilities. RSS cannot make use of more than 64 processors with Windows 2008 R2, even with SP1, at the time of SP1’s release in February, 2011. Use the `netsh` command to enable or disable RSS. If for some reason it is currently disabled on your server, enter the following command to turn it back on:

```
netsh interface tcp set global rss=enabled
```

Modern network drivers are configured to use RSS through settings found in the Windows Device Manager, in the Advanced Properties of the network interface, by following these steps:

1. Open the Device Manager and expand **Network Adapters**.
2. Right-click the adapter you want to configure, and select **Properties**.
3. On the Advanced tab, locate the **Receive-side Scaling** property and verify that the value is **Enabled**. If not, enable it.
4. Click **OK** and exit the Device Manager.

Other advanced properties can be enabled too, depending upon driver implementation. These properties correspond to the registry entries described in the following table:

**Table 2: Advanced properties with corresponding registry entries**

<table>
<thead>
<tr>
<th>Network Adapter Advanced Property</th>
<th>Corresponding Registry Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of CPUs allotted:</td>
<td><code>HKLM\SYSTEM\CurrentControlSet\Control\Class\{XXXXX72-XXX}\&lt;network adapter number&gt;\MaxRSSProcessors</code></td>
</tr>
<tr>
<td>Base CPU number:</td>
<td><code>HKLM\SYSTEM\CurrentControlSet\Control\Class\{XXXXX72-XXX}\&lt;network adapter number&gt;\RssBaseProcNumber</code></td>
</tr>
<tr>
<td>NUMA node affinitization:</td>
<td><code>HKLM\SYSTEM\CurrentControlSet\Control\Class\{XXXXX72-XXX}\&lt;network adapter number&gt;\NumaNodeID</code></td>
</tr>
<tr>
<td>Processor Group:</td>
<td><code>HKLM\SYSTEM\CurrentControlSet\Control\Class\{XXXXX72-XXX}\&lt;network adapter number&gt;\RssBaseProcGroup</code></td>
</tr>
</tbody>
</table>

**Note**

These settings apply only to R2 and NDIS 6.2-compliant drivers.


The values for these properties can vary depending on the cards and their configuration. In most cases these properties are configured correctly by default. But it is important to verify those default assignments on servers with several network interfaces, and override them if necessary, to better handle receive-intensive workloads. Before making changes to these settings however, HP also advises the engagement of HP support or consulting services to provide configuration-specific advice.
Enable Options for Offload Processing

Checksum Offloads, TCP Segmentation (Large Send), and Interrupt Coalescing are all advanced adapter properties that serve to minimize CPU utilization and improve network throughput.

To verify that the network adapter’s advanced property for Checksum Offload is enabled for Transmit/Receive TCP/IP, follow these steps:

1. Open the Device Manager and expand Network Adapters.
2. Right-click the adapter you want to configure, and select Properties.
3. On the Advanced tab, locate the Checksum Offload property and verify that the value is set to Tx/Rx TCP/IP Checksum.
4. Click OK and exit the Device Manager.
5. If you change the setting, you must reboot the server.

On the Gigabit Ethernet NIC driver, Coalescing is enabled by default. Coalesce buffers are used to copy fragments of a transmit packet before assigning them a transmit descriptor. This reduces the number of transmit descriptors required for each packet transmission.

For Intel NIC drivers, set the Interrupt Moderation Rate to High or Extreme. For other NICs, set the Interrupt Moderation Rate to minimize CPU utilization at the expense of higher latencies, or lower latency for higher CPU utilization (and more interrupts). In general, the former is recommended unless the application requires extremely low latencies. If in doubt, a good compromise is to set the interrupt moderation to Adaptive.

SQL Server Tuning Recommendations

While there are potentially many issues to consider when discussing SQL Server tuning, this whitepaper focuses mostly on how to run the SQL Server engine optimally on large NUMA platforms such as the HP ProLiant DL980 G7 server. In the sections above we already covered some of those relevant parameters found at the BIOS, Operating System, and Network levels. Now we need to look at SQL Server itself.

Use Startup Options to Lock Pages in Memory

A set of default startup options is written to the registry when SQL Server is installed, but these options can be overridden. For example, in SQL Server 2008, locking pages for the buffers is supported by default in both SQL Server Enterprise Edition and Developer Edition, whereby the Windows operating system no longer pages out the buffer pool memory within the SQL Server process. However, the Windows operating system can still page out the non-buffer pool memory within the SQL Server process. For more information about this topic, please refer to Microsoft Knowledge Base Article 97070, available at: [http://support.microsoft.com/kb/970070](http://support.microsoft.com/kb/970070)

To enable the Lock Pages in Memory option, follow these steps:

1. Click Start > Run.
2. In the Open box, type gedit.msc.
4. Select the Users Rights Assignment checkbox. The policies are displayed in the details pane.
5. In the details pane, double-click Lock Pages in Memory.
7. In the Select Users or Groups dialog box, add an account with privileges to run sqlservr.exe.

**Note**

To use this capability, the user running the instance of SQL Server (typically Administrators) must have the Lock Pages in Memory capability enabled.
When Lock Pages is used for SQL Server, an entry is logged in the SQL Server error log. You should set a "max server memory" value for the SQL Server instance(s) to ensure the operating system keeps a portion of the RAM for its own operation. A performance monitor helps to determine the best value here.

You also need to enable Trace Flag 845 with SQL Server 2008 R2 Standard Edition, in order to use locked pages for the buffer pool, along with granting the SQL Server service account the "Lock Pages in Memory" security privilege.

Use SQL Server Startup Flags

There are several SQL Server Trace flags you can use to increase SQL performance. These have proven useful in large systems like the DL980.

**T834 – Use Large Pages**
When combined with Lock Pages in Memory option described above, this flag forces SQL to use 2MB pages for the buffer cache instead of the default 4KB pages. This reduces Translation Lookaside Buffer (TLB) size and contention, as well as simplifies memory management by reducing the number of pages to be tracked and manipulated. This flag is particularly recommended on systems where SQL server is the main (or only) application running.

Depending upon the workload, enabling these settings may also increase system performance:

**T652 – Disable Page Prefetching Scans**
Use only for OLTP workloads, and only if the system is more than 80 percent busy.

**T661 – Disable Ghost Removal Processes**
Eliminates the overhead of deleted records being cleaned up by a ghost removal process. If the system is quite active, these deleted records are re-used quickly anyway, so this flag eliminates that overhead.

**T8744 – Disable Prefetch for Ranges**
Disables prefetching for nested loops.

Ensure SQL Server Starts Up Immediately After System Boot

SQL Server should be started as soon as possible after the system boots, especially when using the Use Large Page trace flag described above. This maximizes memory organization for SQL server and ensures that large pages can be easily acquired from the available system memory.

Enable Write Caching on the Log Disk

The bigger the system, the greater the number of transactions and need for a cached log file for the database. The amount of data in the cached log file is proportional to the database. For example, on a Superdome, if the storage array is too slow to handle the array stream, you should enable write caching on the log drive.

As a rule of thumb, on a server with eight or more processors you need significantly more storage for the cached log file. Use an appropriate disk configuration tool to configure and enable the hardware cache. To enable Write Caching on the log disk, follow these steps:

1. Right-click **My Computer** and select **Manage**.
2. Click **Device Manager** and expand **Disk Drives**.
3. Locate the log disk(s) and right-click each device entry. If you are using Secure Path or some other multi-path disk software, the correct disk may be a virtual device.
4. For each entry, right-click, select **Properties**, and click the Policies tab.
5. Verify the **Optimize for performance** radio button is selected.
6. Below that, verify that the **Enable write caching on the disk** and **Enable advanced performance** boxes are checked (see Figure 2).
If the cache is enabled at the hardware level, the first checkbox is usually selected but the second is not. The second checkbox is critical for maximum write performance on the log. These checkboxes can be selected while the system is running. The performance change is immediate and does not require a reboot.

7. Click **OK**, select the device(s) again, and verify that the boxes are still checked.

---

**Note**
Any change in the log disk hardware or cache configuration can cause the operating system to deselect these checkboxes.

Also be aware that many if not most fibre channel RAID controllers ignore whatever Write Back cache option is selected via the Windows interface, so you must use the RAID controller’s configuration utility to enable this option.

---

**Segregate Network Processing from SQL Processing**

Use the `alter server` command in SQL Server 2008 R2 to associate threads with specific processors and to specify which processors SQL Server can use. This command supersedes the Affinity Mask and Affinity64 Mask options used previously to perform these functions. You should also exclude SQL Server activity from any processors assigned a NIC workload by the OS.

---

**Note**
Soft NUMA (discussed below) can also accomplish this task.
Before changing the affinity settings, keep in mind that the OS assigns deferred procedure call (DPC) activity associated with NICs to the highest numbered processor in the system. In systems with more than one active NIC, each additional card’s activity is assigned to the next highest numbered processor. For example, an 8-processor system with two NICs has DPCs for the NICs assigned to processor 7 and processor 6.

Syntax for the new `alter server` command can be found at Microsoft’s SQL Server Books Online (http://msdn.microsoft.com/en-us/library/ms130214.aspx). The following example sets the affinity of SQL on a fully-loaded DL980, leaving it OFF of CPUs 2, 4, and 28, thereby allowing the NIC interrupts to reside there:

```
alter server configuration set process affinity cpu=0,1,3,5 to 27, 29 to 127
```

Configuration changes like this take place immediately, and can be done with the workload active. Any threads on the excluded processors will continue to run on those processors to completion, but no new threads will be assigned to the excluded LPs.

Use Data Compression Appropriately

Data compression is one of the new features introduced in SQL Server 2008. There are three different modes for compression:

- **Backup Compression**: Turned OFF by default. Can be turned ON for a given database. Reduces storage media needs but uses a great deal of processing power when compressing, and somewhat less when decompressing.

- **Row-level Compression**: Best used to compress data within a row. Uses fewer columns to store variable length data, such as strings that vary in length from row to row.

- **Page-level Compression**: Best used when the value of one or more columns is the same in multiple rows. The redundant data is stored once in the page and then reconstructed when read.

In all of these schemes there is a tradeoff between processing and storage performance. Use care in balancing the needs for normal processing with the compression and decompression of the data. Prototyping the specific application of compression within SQL is recommended.

Application-dependent SQL Parameter Recommendations

Use the `sp_configure` stored procedure to optimize resources. To modify the advanced configuration options, we recommend that you first set the Show Advanced Options property, then run `RECONFIGURE` and restart the SQL Server instance, by entering the following:

```
1. sp_configure Show Advanced options
2. GO
3. RECONFIGURE
```

For OLTP Workloads

Set Max Degree of Parallelism (MaxDOP) to 1. This option limits the number of processors used in parallel plan execution. If you use the `sp_configure` stored procedure to change the setting, you must first set Show Advanced Options to 1. The setting change takes effect immediately without a SQL instance stop and restart.

**Note**

Higher MaxDOP values can be used, of course, but OLTP performance degrades as this value rises to higher levels. As always, you must experiment with your workload to find the optimal setting.
You must also check the transaction log latency and write size. Since every transaction committed in SQL must be written and committed to the log, the SQL log can easily become a bottleneck and limit system performance. A quick check with the Windows Performance Monitor utility can easily verify this.

Log Write Service times should be very low (about 1ms). If they are not, the cache could be disabled, either within the array or in Windows. Average Log Write Sizes of greater than 30K can also indicate a bottleneck, possibly caused by a disabled cache. Or it can indicate that more log disks or a larger cache size is required.

If a software RAID0 stripe across multiple arrays is used for the log, it may sometimes appear that no log bottleneck exists, when in fact one does. If the sum of the average Log QueueDepth across all of the Log’s RAID arrays multiplied by the average log write size approaches (or is greater than) 64KB, you might have a log bottleneck. Measuring this requires that the log be on separate LUNs from the rest of the database.

Remember to spread all tables over multiple files and over multiple disks (the more spindles the better). Do not software-stripe LUNs that are already hardware-stripped either. While this makes them easier to manage, it degrades performance considerably.

**Drive Considerations**

Use the following guidelines regarding database storage:

- As noted above, if database "storage" latency is greater than 2 to 3 times what is was unloaded, you should add more storage.
- Keep random IOPS to <120/sec/spindle for 15KRPM drives and 100/sec/spindle for 10KRPM drives.
- For Gigabit NICs, keep packets below 30K/sec. Use the `sp_configure` stored procedure to set Lightweight Pooling to 1.
- When Lightweight Pooling is set to 1, SQL Server switches to Fibre Mode scheduling. In the event of excessive context switching, Lightweight Pooling provides better throughput by performing context switching inline, which reduces user/kernel mode context transitions. Note that Fibre Mode is generally not effective with a CPU utilization below 80 percent on a few or all CPUs.
- By default, SQL Server uses one thread per active SPID or user process. These threads work in a pooled configuration to keep the number of threads manageable. The advanced Lightweight Pooling configuration option (sometimes referred to as Fibre mode) uses Windows "fibre" support to handle several execution contexts with a single thread.

**Lightweight Pooling Restrictions**

- **SQL Server Agent (Microsoft KB303287):** When SQL Server runs in Lightweight Pooling mode (or Fibre mode) and the DTC service is started, unexpected behavior may occur. SQL Server Agent might not execute any jobs.
- **DTC (Microsoft KB303287):** If DTC operations are required on the server, the SQL Server instance should always run in Thread mode. In other words, Lightweight Pooling should be set to 0 (zero). Microsoft strongly recommends that you run the SQL Server instance in Thread mode when DTC is needed. If you use Lightweight Pooling/Fibre mode on a system that does not specifically need it, performance often degrades.
- **sp_xml_and *sp_OA* functionality (Microsoft KB322884):** Microsoft does not support the use of Microsoft Common Language Runtime (CLR) extended stored procedures or OLE Automation with any libraries loaded to run in the SQL Server memory space. CLR only uses thread-based scheduling and does not support fibre-mode scheduling. In later versions of SQL, you cannot load CLR by using extended stored procedures or sp_OA stored procedures.

For more information about the lightweight pooling option, refer to:


**Other Considerations**

Use NUMA support to reduce remote memory access. This can improve performance up to 60%, especially when combined with Connection Affinity. In general, segregate connections and thus data locality in a given node (for example, by region or department, or any other logical division that makes sense for the application). When NUMA support is enabled, SQL attempts to create a thread’s data structures in the same NUMA node, thereby reducing remote memory accesses.
In SQL Server 2008/R2, NUMA is enabled by default. Both versions of SQL Server also provide a Soft NUMA feature, again enabled by default, which enables more precise control as described below.

Use Connection Affinity to take further advantage of SQL Server NUMA features. With Connection Affinity, a SQL connection from the client is assigned affinity to a specific NUMA node. This assigns data structures to that NUMA node, further enhancing the NUMA capabilities of SQL. When Connection Affinity is enabled for all network connections, the requirement to use VIA for affinity no longer applies. Moreover, multiple connection ports may be used on a single hardware network adapter, further increasing the flexibility.

Soft NUMA allows database administrators to configure pseudo-NUMA nodes that SQL Server treats like hardware nodes. You can configure Soft NUMA nodes down to 1 processor, allowing fine control of connection affinity and workload distribution. In addition, smaller x86 servers without hardware NUMA capabilities can still run SQL employing Soft NUMA. These machines do not employ the NUMA concept of local and remote memory access, but they do allow the SQL administrator to balance the workload down to the level of a single processor.

If SQL Server 2008/R2 are run with no Soft NUMA nodes configured, then the hardware NUMA configuration is used. The NUMA configuration, hard or soft, is written at startup to the SQL log.

To use these features, follow these steps:

1. Use regedit to configure the Soft NUMA nodes and port listen strings.
2. Restart SQL.
3. Set clients to use ports configured above.

It is also recommended that you install the HP System Providers v8.7.1 on the system and use the Optimize Logical Processor Configuration button in the HP System Management Homepage to preset your logical processor and kernel groups (for nodes/group optimization).

A Configuration Example: How to Create 4 Soft NUMA Nodes with 2 CPUs Each

This example is for systems running SQL Server 2008 R2, and specifies the Group parameter, which applies only to systems with >64 LPs. This example should be ignored for SQL Server 2008, or if the executing system is equipped with ≤64 LPs.

First, run regedit and add the following entries to the system registry:

```
HKLM\SOFTWARE\Microsoft\Microsoft SQL Server\100\NodeConfiguration

[HKLM\SOFTWARE\Microsoft\Microsoft SQL Server\100\NodeConfiguration \Node0] "CPUMask"=dword:00000003 "Group"=dword:00000000

[HKLM\SOFTWARE\Microsoft\Microsoft SQL Server\100\NodeConfiguration \Node1] "CPUMask"=dword:0000000C "Group"=dword:00000000

[HKLM\SOFTWARE\Microsoft\Microsoft SQL Server\100\NodeConfiguration \Node2] "CPUMask"=dword:00000030 "Group"=dword:00000001

[HKLM\SOFTWARE\Microsoft\Microsoft SQL Server\100\NodeConfiguration \Node3] "CPUMask"=dword:000000C0 "Group"=dword:00000001
```

The CPUMask value is a bitmask of CPUs relative to the system. So Node0 described above includes CPUs 0 and 1, Node1 includes CPUs 2 and 3, and so on. There is no actual limit of 4 NUMA nodes; this is just an example.

**Note**

Although SQL allows configuration of Soft NUMA nodes that cross hardware NUMA node boundaries, this is not recommended because it results in excessive remote memory accessing.
Second, create the port listening strings using SQL Server Configuration Manager. Under SQL Server 2008/R2, select **Network Configuration > Protocols for MSSQLSERVER > TCP/IP > Properties > IPAddresses > IPAll** > TCP Port. You can also use regedit to modify the following key in the system registry:

```
[HKLM\SOFTWARE\Microsoft\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQLServer\SuperSocketNetLib\Tcp\IPA1]
```

The listening string format is:

```
PortNo1[SoftNumaNodeMask],PortNo2[SoftNumaNodeMask]... PortNo N [SoftNumaNodeMask N]
```

So the four ports for the Soft NUMA machines in this example would be:

```
1436[0x1],1437[0x2],1438[0x4],1439[0x8]
```

**Note**

Unlike SQL2000 which uses CPU masks directly in the listening string, SQL Server 2008/R2 specifies Soft NUMA masks. In other words, SQL Server 2008/R2 employ a two-level definition, while SQL2000 is only one-level.

Therefore, the full regedit key is:

```
[HKLM\SOFTWARE\Microsoft\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQLServer\SuperSocketNetLib\Tcp\IPA1]
"TcpPort"="1436[0x1],1437[0x2],1438[0x4],1439[0x8]" "TcpDynamicPorts"=""
"DisplayName"="Any IP Address"
```

The port number is the point of connection between the client and a specific Soft NUMA node. So if a client application wanted to connect to Soft NUMA Node 0, it would specify:

```
SQLCMD –E -Sservername,1436
```

Note that the default port, 1433, can also be used. This results in establishing connection affinity with all CPUs in the system. However the heavy load imposed by these connections will degrade the maximum potential of the system.

Another alternative to SQL destination port-based client affinity selection is to use multiple NICs and, for each one, set its own IP addresses for SQL Server to use.

**For Business Intelligence Workloads**

You should set MaxDOP (Maximum Degree Of Parallelism) to the ideal number. Use the number of CPUs per NUMA node, or a multiple of it, as a first order approximation for complex queries. For simple queries, use the MaxDOP hint. Finding the ideal MaxDOP setting takes some experimentation. For example, you could start with the number of CPUs per NUMA node (4, 6, 8 or 10), and then use the MaxDOP hint. Also, the larger the system and the more processors available, the larger MaxDOP can be without affecting other applications.

It is advisable to use Tuning Advisor to optimize indices. Defining proper indices can have the most impact on large queries. Creation, tuning, and rebuilding of indices must be done carefully for maximum affect.

When it comes to partition tables, especially large fact tables, it is preferable to break them down into manageable-sized chunks so they are spread across different disks. This improves query response and seek times, data loading, purging, and other system management operations. These improvements can also be accomplished using indexes.
A guideline for setting the Max Worker Threads value is to make it equal to the total number of DB connections + the number of CPUs + 8. Depending on your system configuration, setting Max Worker Threads to a value smaller than the default (255) can sometimes improve performance. If you use the `sp_configure` stored procedure to change the setting, you must first set Show Advanced Options to 1. The setting change takes effect immediately without a SQL instance stop and restart.

Also, if an application uses the TempDB database, you should put TempDB on fast disks with write cache enabled. These disks must be large enough to support the traffic (for example, avoid the AUTOGROW option). Spreading TempDB to multiple files also helps improve performance. A general rule to follow is one TempDB file per CPU.

**Conclusion**

To reiterate, just as Windows Server 2008 can only support 64 logical processors, SQL Server 2008 has the same limitation. For this reason you should only run SQL Server 2008 on the HP DL980 G7 when Hyper-threading is disabled. To best utilize the full computing power of the DL980, you should always run Windows Server 2008 R2 with SQL Server 2008 R2 as the Database Management System (DBMS).

HP ProLiant DL980 G7 Servers provide powerful computing and memory resources. To enable SQL Server to take full advantage of these resources, we recommend hardware and software tuning. By applying some or all of the guidelines in this document, you will achieve the highest performance from your server.
For More Information

For more information about HP ProLiant DL980 G7 servers:
http://www.hp.com/servers/DL980

Performance Tuning Guidelines for Windows Server 2008:

Performance Tuning Guidelines for Windows Server 2008 R2:

Documentation Feedback

HP welcomes your feedback. To make comments and suggestions about product documentation, send a message to:
docsfeedback@hp.com. Include the document title, part number, and filename found at the end of the URL string
(for example: c02861709.pdf). All submissions become the property of HP.