



Windows® HPC Server 2008

Windows HPC Server 2008 Head Node Performance Tuning

Microsoft Corporation

Published: December 2008

Abstract

This whitepaper seeks to simplify the process of performance tuning a Windows Server® 2008 head node, providing you with the information that you need to plan and configure the Windows® HPC Server 2008 cluster deployments best suited to running your workload. This paper will focus on issues related to the Windows HPC Server 2008 components that run on the cluster's head node, enabling you to plan, monitor, and tune the performance of these components.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2008 Microsoft Corporation. All rights reserved.

Microsoft, Active Directory, Excel, SQL Server, Visual Studio, Windows, the Windows logo, and Windows Server are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owner

Contents

Introduction	5
How to use this Guide	5
Overview of Head Node Architecture and Resource Usage	6
Hardware Requirements Based on Workload	7
Sample Capacity Data	8
Disk Space Requirements.....	9
Data Growth Example	9
Database Configuration	10
SQL Server Express vs. SQL Server Standard.....	10
Choosing the Right Version of SQL for Your Head Node	10
Maintenance Plans.....	10
SQL Configuration Best Practices	11
Infrastructure Requirements for Running SOA Jobs.....	12
Factors that Affect SOA Capacity Planning	12
Testing the Maximum Message Throughput of a Broker Node.....	13
Best Practices for Running SOA Workloads	13
Basic Techniques for Measuring and Improving System Performance	14
CPU.....	14
Memory.....	14
Disk.....	14
Best Practices	16
Minimum spec for a large-cluster head node.....	16
Head nodes should not be configured with additional roles.....	16
Use the HPC Client Tools rather than logging into the head node directly	16
Disable performance counter collection and event forwarding on very large clusters	16
Disable unneeded head node services	17
Do not run NAT on the head node for larger clusters	17
Ensure a reasonable period of storage for completed Jobs	17
Summary	18
Appendix 1: Performing SQL Operations	19

Configuring SQL Memory Usage	19
Pre-Allocating SQL Databases	20
Appendix 2: Viewing Windows Performance Metrics	21
Using the Windows Resource Monitor	21

Introduction

Planning a cluster for running high-performance computing (HPC) applications is a complicated process, with a variety of factors contributing to the overall reliability and performance of the cluster. This whitepaper seeks to simplify this process, providing you with the information that you need to plan and configure the Windows HPC Server 2008 cluster deployments best suited to running your workload.

This paper will focus on issues related to the Windows HPC Server 2008 components, and *not* on the performance of applications running on the Windows HPC Server 2008 platform. For information on performance analysis and tuning of HPC applications, please see the whitepaper entitled [Performance Tuning a Windows HPC Cluster](#).

Every cluster is different, not just in terms of the hardware and software that it runs, but also in terms of the users and workloads that it serves. This document will help you to decide what is right for your cluster with regard to:

- The hardware and configuration required on your head node in order to support your workload.
- The database storage required to keep the HPC Server 2008 Job Scheduler running smoothly.
- The database configuration that best suits your needs.
- Configuring your cluster to run service-oriented architecture workloads in a reliable and performant manner.
- Measuring and improving system performance.
- Best practices to keep your cluster performing reliably and efficiently.

How to Use This Guide

The goal of this guide is to provide you with information about how your workload affects the way that the various Windows HPC Pack services make use of the head node hardware. To make best use of this guide, we recommend taking measurements *in your environment, with your workload* to learn how your system is behaving. Once you've gathered this information, this guide should enable you to determine what hardware or configuration changes you can make to improve the performance and reliability of your system.

Note that much of the advice in this guide is intended for very large or very high throughput compute clusters. We expect that most HPC Server users will find a standard, out-of-the-box configuration sufficient for their requirements.

Overview of Head Node Architecture and Resource Usage

The head node of an HPC Server 2008 cluster includes a number of different components which provide services essential to the smooth operation of your cluster. Of these, the following are the most important in determining the overall performance and reliability of the cluster:

- The **HPC Job Scheduler Service** is responsible for queuing jobs and tasks, allocating resources, dispatching the tasks to the compute nodes, and monitoring the status of the job, tasks, and nodes.
- The **HPC Management Service** controls all aspects of compute cluster operation and that manages the cluster database. This service on the head node provides overall cluster management of node discovery, as well as configuration management. Each node in the cluster also runs one instance of the HPC Management service, which communicates with the HPC Management service on the head node and is responsible for node discovery within the cluster. This service is backed by the **HPC SDM Store Service**.
- **SQL Server 2005** serves as the storage platform for all of the head node components.

Hardware Requirements Based on Workload

All of a computer's resources (memory, CPU, and disk) are vital to the performance of your HPC Server head node in different ways. Insufficient RAM, hard disk capacity, network capacity, and CPU power can all result in sub-optimal performance of the HPC Server head node.

The following table provides details on how each of these resources can impact the capacity of the head node:

Table 1: Impact of Hardware Resources on Head Node Services

Resource	Details
CPU	Scheduling passes will put load on the CPU, especially for clusters with many nodes/processors. CPU is also an important limiting factor for the scheduler's ability to service client requests. Note that CPU load may also have an impact on SQL Server performance, which can affect a number of head node services.
Memory	Memory is a major factor in the scheduler's ability to handle clients connected to the scheduler, particularly when users are viewing the job queue using the GUI. Note that memory may also have an impact on SQL Server performance, which can affect a number of head node services.
Disk Access	Because SQL Server is used for storage of all scheduler and cluster management states, disk access impacts many different cluster operations. In particular, disk access will be the primary bottleneck for clusters with large rates of job submission or cluster configuration changes.
Network	The head node uses network resources for both client communication and head node-compute node communication. This can include deployment and network services (NAT, DHCP, DNS).

The size of your workload will be the largest determining factor in how powerful your head node needs to be with regard to these different hardware components. Workload can generally be measured in terms of:

- **Job Throughput** - The number and size of jobs which will be submitted to the cluster in a given time period
- **Concurrent Users** - The number of users who will be simultaneously connected to the head node
- **Cluster Size** - The number of compute nodes (and processors) managed by the head node

As each of these metrics increases in size, your cluster's CPU, memory, disk and network capabilities may also need to increase.

Sample Capacity Data

To provide a frame of reference for your own environment, we include here some details on a cluster that we tested internally. This cluster was composed of 256 nodes (2048 processors), with 30 concurrent users who submitted a total of 150,000 jobs a day for a week. The cluster's head node has the following configuration:

- 2x Intel E5320 Quad-Core processors @ 1.86 GHz
- 8 GB RAM
- 3x 10000 RPM disks
- Gigabit Ethernet Management Network

The primary bottleneck for this cluster turned out to be disk write I/O, which limited the maximum job throughput that the scheduler was able to handle. We were able to improve the overall job throughput dramatically by moving the SQL Transaction Log to a separate hard disk.¹

¹ See the section on SQL Configuration Best Practices for more details on this

Disk Space Requirements

The amount of space taken up by the SQL database used by the Windows HPC Pack is based mostly on:

- The number of jobs stored in the system (each of which is stored in the SQL database)
- The size of jobs (number of tasks per job)
- The number of nodes being managed (for which performance counters, monitoring, and history data are stored in SQL Server)

One important factor in determining how large the database will grow is the clean-up interval of the job scheduler (set using the `TTLCompletedJobs` parameter, which can be set using the `cluscfg` tool). The job scheduler will delete all job data from the database after a specified amount of time. Users who set a high value for `TTLCompletedJobs` may see very significant data growth.

Data Growth Example

When testing with the aforementioned 256 node cluster, we reached a total of about 1,000,000 jobs stored in the database (about 100 jobs a minute over 5 days). Each job had around 10 tasks, with each task running on between 32-64 cores. The head node database eventually grew to a size of about 100 GB, with the largest contributing factors being:

- 966,000 jobs taking up approximately 1 GB of storage
- 9,000,000 tasks taking up ~65 GB (including 4 K of cached output per task)
- 20 GB of job allocation history data (historical data of where every job and task in the system ran)
- Approximately 20 million performance counter data points collected from the compute nodes, taking up about 0.6 GB

As this data indicates, the primary driver of data growth will be the number and size of tasks, especially when those tasks redirect output and error streams to the database instead of to an output file.

Database Configuration

SQL Server Express vs. SQL Server Standard

Windows HPC Server 2008 ships with Microsoft SQL Server 2005 Express Edition, the freely available version of the Microsoft SQL Server database application. SQL Server Express is identical to the retail versions of SQL Server in many respects, but there are some key benefits provided by the full version of SQL Server. These include the following features (among others) which may be important to Windows HPC Server administrators:

- Unlimited database size (SQL Server Express databases are limited to 4 GB in size)²
- SQL Server Management Studio for configuring and maintaining your SQL database
- Support for high-availability configurations
- Multi-threading to take advantage of multiple processors on the head node
- Unlimited RAM usage for database caching

Choosing the Right Version of SQL for Your Head Node

Given the factors listed above, we highly recommend the use of SQL Server Standard (or a premium edition) for users who:

- Run clusters of more than 128 nodes.
- Run clusters with a very high rate of job throughput (greater than 10,000 jobs per day).
- Need the additional reliability, performance, and flexibility provided by the SQL Server Management Studio tools (including support for maintenance plans).
- Need to store job and task data in the scheduler database for an extended period of time and will exceed the 4 GB limit imposed by SQL Server Express.

Also note that SQL Server Express will not support Windows HPC Server installations running in a high-availability configuration.

Maintenance Plans

A typical SQL Server maintenance plan covers the following:

- Database backup
- Consistency checks
- Index defragmentation

² See the section entitled “Disk Space Requirements” for more details on the storage requirements of Windows HPC Server 2008

We generally recommend that you rebuild your indexes after 250,000 jobs or one month (whichever is shorter), if not more often. How often you do consistency checks and backups will depend on your business requirements.

We recommend running maintenance only when there is little to no user activity, preferably during a scheduled downtime (especially for larger clusters), as it can severely impact job throughput and user experience.

SQL Configuration Best Practices

We strongly recommend the following best practices when configuring the SQL Server database for your Windows HPC Server head node:

- Configure SQL Server to allocate at least 4 GB of memory for the management and scheduler services.³
- Pre-allocate the databases up front, ensuring very infrequent growths.⁴
- Monitor index fragmentation using the SQL Server Management Studio and defragment indexes when appropriate through a maintenance plan.
- If reporting is heavily used, consider moving the reporting database to another disk.
- Set up the system to separate the log and database onto different platters if heavy load is expected. Ideally, place the system partition, data, and logs on separate platters.

³ See Appendix 1: Performing SQL Operations for more details.

⁴ See Appendix 1: Performing SQL Operations for more details.

Infrastructure Requirements for Running SOA Jobs

SOA applications have some special capacity planning requirements because of the particular way that they interact with the cluster and the additional need for WCF Broker nodes in the cluster.

The key factors impacting the overall throughput of a SOA cluster are:

1. Brokers are the primary bottleneck for SOA workload throughput.
2. Brokers are generally CPU bound (not network bound) in terms of their ability to handle large throughputs.
3. Running multiple brokers on a single system may yield better overall throughput than running a single large broker on a system (i.e., two smaller SOA sessions may be better than one large SOA session).

Factors that Affect SOA Capacity Planning

In general, the SOA-specific capacity requirements of your cluster will be most strongly impacted by the following factors:

- The number of concurrent SOA sessions which will be running at any given time
- The rate at which requests are sent in each SOA session
- The size of the messages being sent
- Processing time per request (longer requests can scale out to more services without overloading the broker)

As the number of connections, request rate, and message size increase (and as the processing time per request decreases), you can increase capacity by:

- Scaling out to additional broker nodes.
- Upgrading to more powerful CPUs on your broker nodes.

As the below graph shows, increasing the number (and speed) of cores available to your brokers will significantly increase their throughput capability.

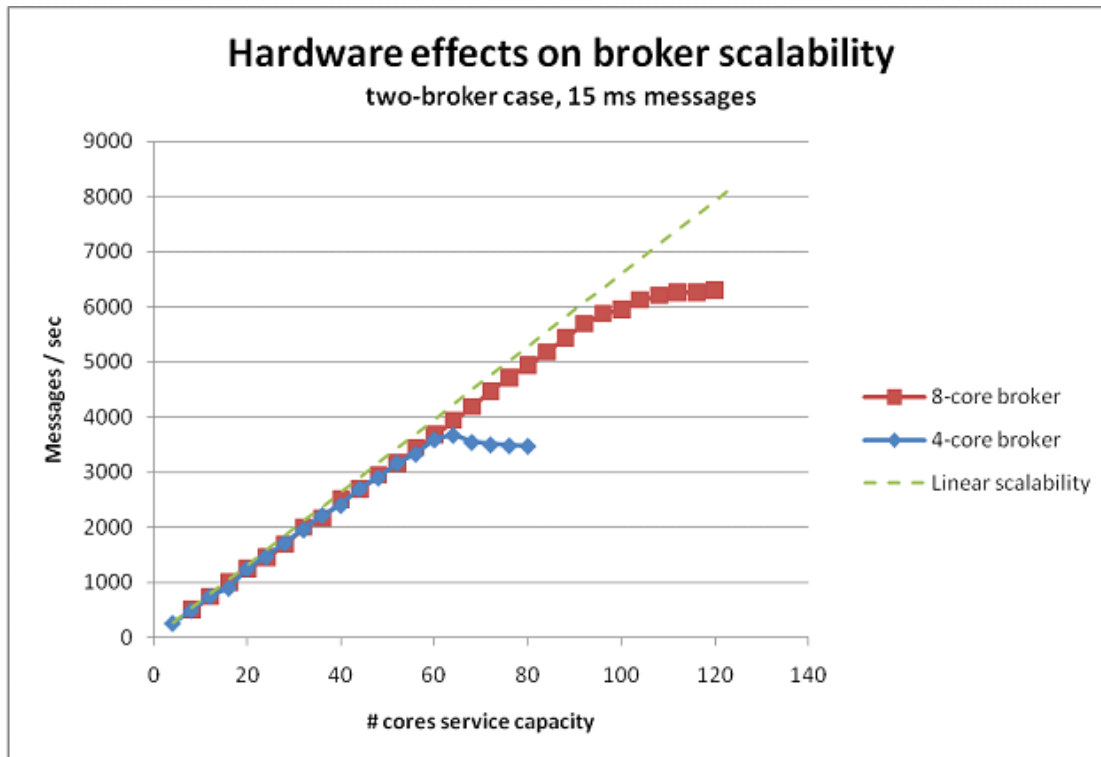


Figure 1: Messages per second on an 8-core node vs. a 4-core node

Testing the Maximum Message Throughput of a Broker Node

You can perform a single experiment to determine the maximum message throughput (message/second) that a broker is capable of handling on your hardware with your workload. To do so, use the Windows Performance Monitor counters for CPU and HPC Windows Communication Foundation (WCF) broker calls per second to track the broker's usage of a node. You can then gradually increase the number of service nodes (or number of clients) available on the back end until the number of messages the broker can handle stops increasing.

Best Practices for Running SOA Workloads

Avoid sending large messages through the WCF broker

Large messages (greater than 64 KB) put a higher stress on the broker due to the cost of serialization. Instead of sending large messages through the broker, consider putting data on in a file share or database and only send essential, per-request information in the actual WCF request.

Avoid using Transport-level security if it is not needed.

Transport-level security imposes a significant overhead on WCF messaging. If message security is not required, it is recommended that you disable transport-level security to increase performance.

Basic Techniques for Measuring and Improving System Performance

Measuring system performance on Windows is actually quite easy using the Reliability and Performance Monitor.⁵ By examining certain core metrics, you should be able to determine if a Windows HPC Server head node or broker node is hitting a hardware bottleneck and what that bottleneck is.

CPU

If CPU utilization is consistently 90% or higher on a head node or broker node, you may want to consider the following changes:

- In the case of very high CPU utilization on a broker node, scale out to additional broker nodes.
- Make sure that users submit from remote clients rather than logging into the head node to submit.
- Reduce load by disabling unnecessary services or features.
- Upgrade to more (or more powerful) processors.

Memory

Ideally, there should be zero Hard Faults / Second when things are working. If there are a significant number of hard faults per second, you may want to:

- Reduce the size of the job database by lowering the TTLCompletedJobs property (see the Best Practices section for more details).
- Reduce memory pressure by disabling unnecessary services or features.
- Increase the amount of RAM in the system.

Disk

Check the performance counters for Average Disk Seconds (Avg. Disk Sec) per Write and Average Disk Seconds per Read. If these are high (i.e., more than 100 ms), you may want to consider optimizing the way that your database is stored on disk (see the section on [configuring SQL](#) for additional details on this).

Disk Queue Length is another good indicator of your system's disk utilization; a Disk Queue Length of greater than 1 per physical platter is a warning sign that your system is frequently waiting for Disk I/O.

⁵ See Appendix 2: Viewing Windows Performance metrics

From an Windows HPC Pack perspective, write I/O bottlenecks will probably be a result of SQL log writes, and may be improved by moving the log to a separate platter (or by adding a write cache controller). Read bottlenecks will most often be caused by the Job Scheduler accessing the job queue, and performance may be improved by:

- Making sure the scheduler's SQL database is not stored on the system partition.
- Moving the SQL database onto a RAID array or faster disks to improve read-speed.

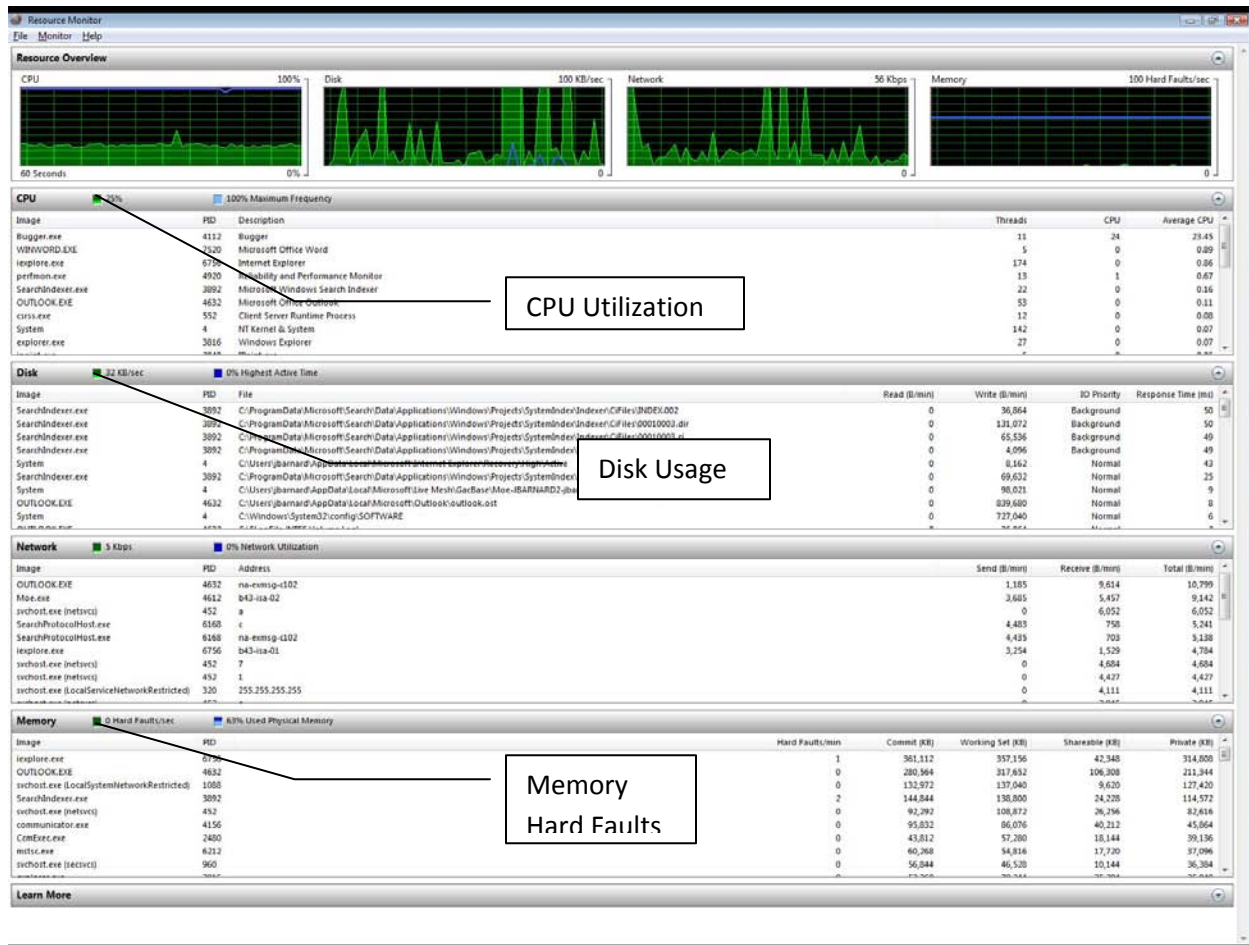


Figure 2: The Windows Resource Monitor

Best Practices

There are a number of best practices which we recommend for large clusters, and most will also improve the behavior of smaller clusters. The only exceptions to these guidelines are test clusters on which performance and reliability of the head node services may not be critical, and smaller clusters where the head node services will not be highly stressed.

Minimum specification for a large-cluster head node

As a general guideline, we recommend that the head node of a large cluster meet at least the following minimum specification:

- 4 CPU Cores
- 2 disks
- 8 GB of RAM

Head nodes should not be configured with additional roles

For virtually all clusters, we recommended that head nodes *not* be configured with any additional role (compute node or WCF broker node). Having the head node serve two purposes will prevent it from completely focusing on either. You can change the roles performed by your head node from the Node Management page of the HPC Cluster Manager by right-clicking on the head node and selecting *Change Role*.

Use the HPC Client Tools rather than logging into the head node directly

When operating under a heavy load, head node performance can be negatively impacted by having many users connected with remote desktop connections. Rather than having users connect to the head node using terminal services, users and administrators should install the HPC Pack Client Utilities on their workstations and access the cluster using these remote tools.

Disable performance counter collection and event forwarding on very large clusters

On large clusters, performance counter and event collection can put a large burden on the HPC Management Service and SQL Server. For these clusters, it may be desirable to disable these capabilities using the `-CollectCounters` and `-ForwardEvents` flags with the `Hpc-SetClusterProperty` Windows PowerShell™ cmdlet.

Disable unneeded head node services

In order to ensure a minimal hardware footprint from the operating system, any unnecessary operating system services should be disabled. We especially encourage disabling any desktop-oriented features that may have been enabled.

Do not run NAT on the head node for larger clusters

Although the HPC Pack allows quick configuration of the Routing and Remote Access Service (RRAS) running on the head node to provide Network Address Translation (NAT) and allow compute nodes to reach the enterprise network, this may turn the head node into a significant bottleneck for network bandwidth and may also put a significant workload on the head node. For larger clusters or clusters with a lot of communication between compute nodes and the public network, we recommend that you either provide direct public network connections to each compute node, or provide a dedicated NAT router (such as a separate Windows Server® dual-homed on the two networks and running RRAS).

Ensure a reasonable period of storage for completed Jobs

The TTLCompletedJobs property of cluscfg allows control over how long completed jobs will remain in the system. While setting this value high ensures that jobs are maintained in the system for a long time (which may be desirable), having a lot of jobs in the system will increase the storage and memory requirements of the system, since the database (and queries against it) will generally be larger.

Summary

As clusters grow in size and throughput, the stress put on the scheduling and management components increases. In general, the key elements necessary to ensure your head node is up to the challenge are:

- Enough RAM to handle the number of simultaneous users
- Enough disk space to store job queue and management data over time
- A performant, well-maintained SQL Server database backing the cluster services

With these components in place, your cluster should be up to even the most demanding workloads.

Appendix 1: Performing SQL Operations

Configuring SQL Memory Usage

You can set the maximum amount of memory that SQL will use to ensure that other services on the head node can get the memory they need. To do so in SQL Server Management Studio:

1. Right-click on the server instance and choose Properties.
2. Select the Memory tab.
3. Configure the maximum server memory. We recommend setting this to 4 GB less than the total memory present on the machine.

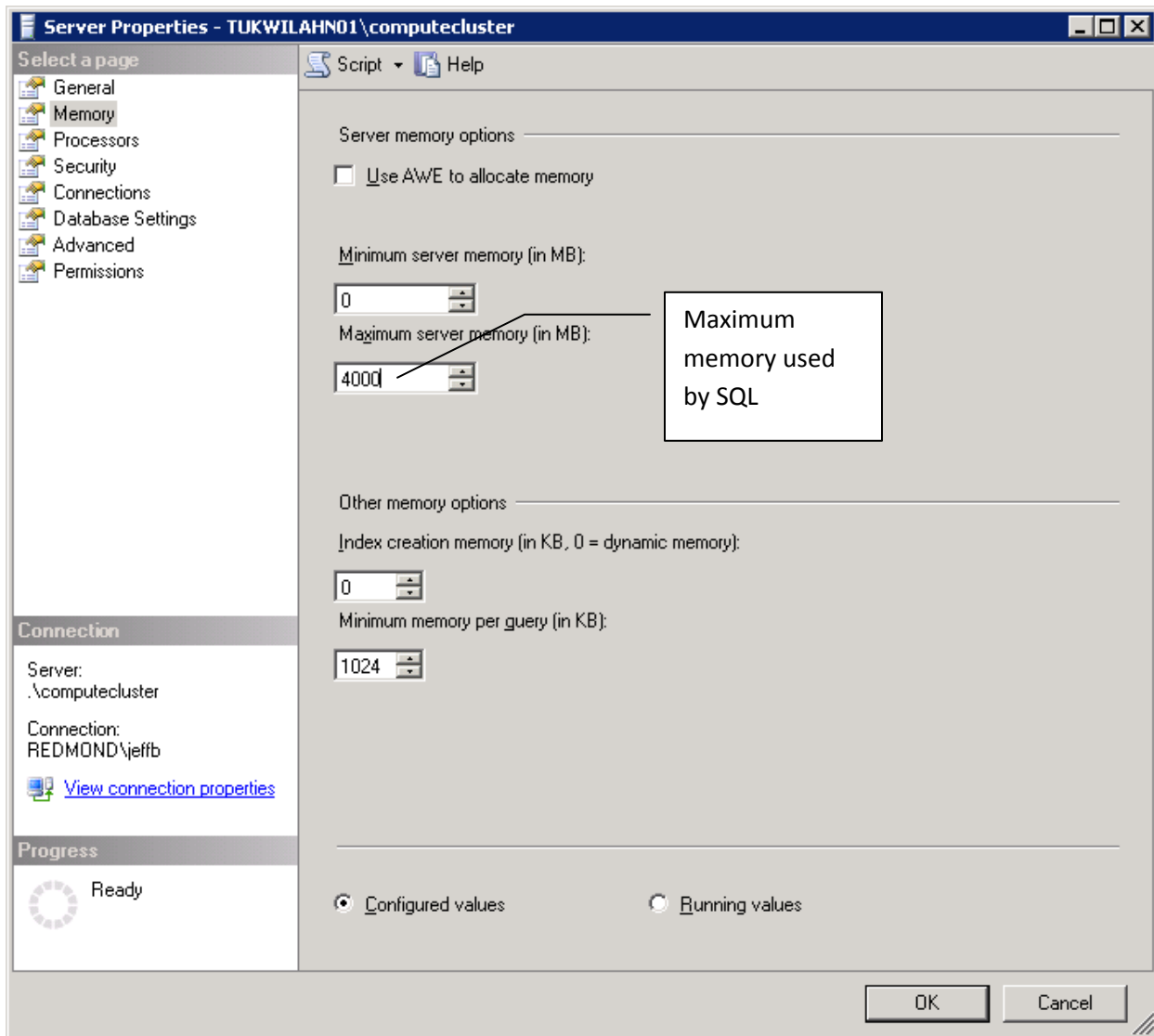


Figure 3: Configuring SQL's Memory Usage

Pre-Allocating SQL Databases

Pre-Allocating SQL databases provides a large initial size for the database. This reduces the need for later database file growth, which can be expensive to perform.

To set an initial database size in SQL Server Management Studio:

1. Right-click the CCPClusterService database in the left navigation tree and choose Properties.
2. Click the File tab.
3. Set the initial size of the database. A good rule of thumb is to allocate at least 100 KB per job that will be stored in the system at any one time. If you submit very large or small jobs (in terms of the number of tasks), adjust this accordingly. For more details on this, see the Disk Space Requirements section.

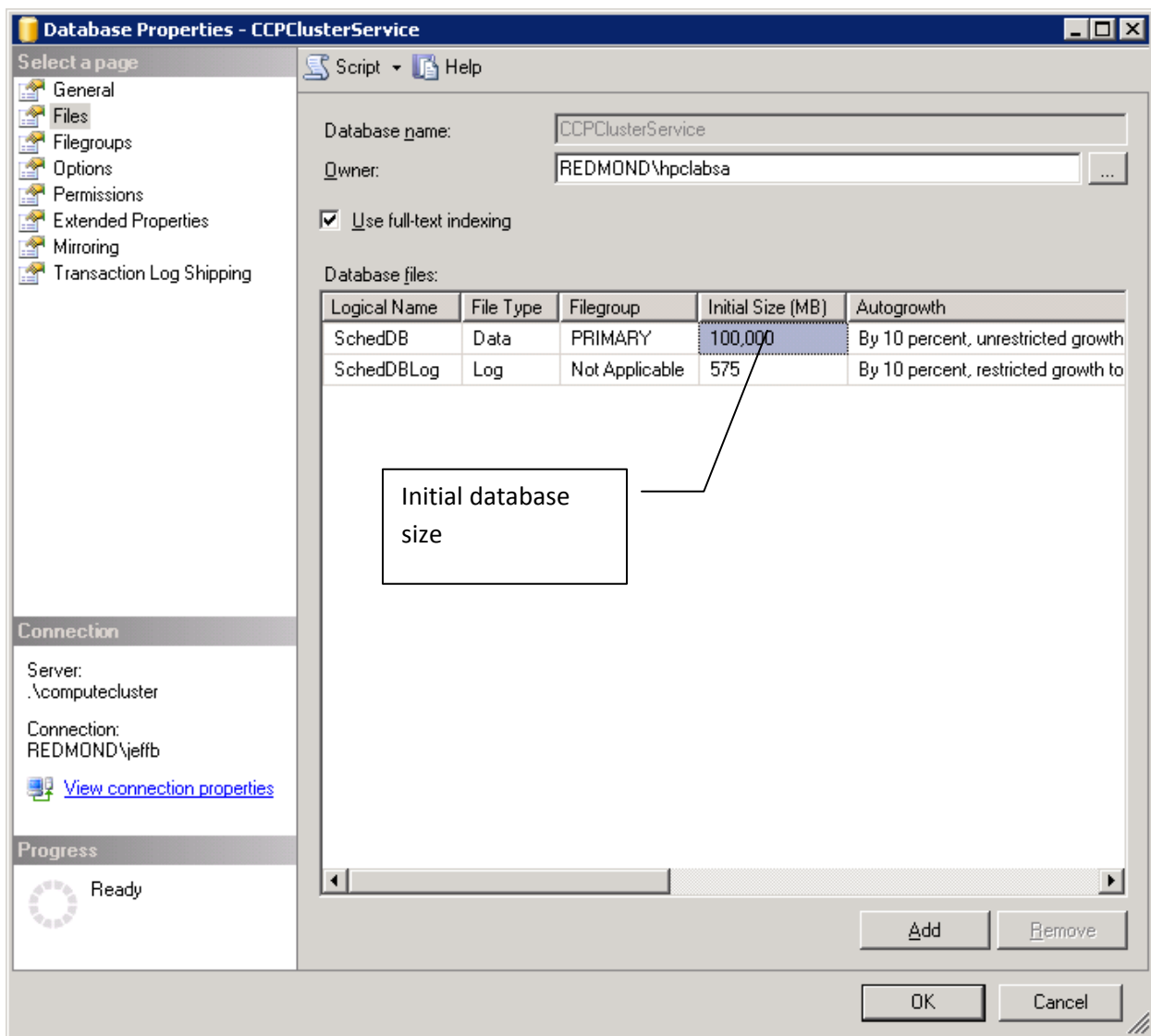


Figure 4: Setting the Initial SQL Database Size

Appendix 2: Viewing Windows Performance Metrics

Using the Windows Resource Monitor

To view Windows performance metrics using the Windows Resource Monitor:

1. Right-click on the Windows taskbar and select Task Manager.
2. Click the Performance tab.
3. Click the Resource Monitor button.