One of the prime purpose of the Perfmon is to discover bottlenecks. Thus, we must first understand what is a bottleneck?

A bottleneck occurs when simultaneous access BY APPLICATION OR USERS to shared resources such as (CPU, memory, disk, network or other resources) causes an overload for the resources. This demand on shared resources can cause poor response time and must be identified and tuned by the DBA. The tuning can be either hardware upgrade, application tuning or both. Monitoring SQL Server performance is a complex task, as performance depends on many parameters, both hardware and software

What causes bottleneck:

* Insufficient hardware resources which may require upgrade or replacement
* Resources are not distributed evenly; an example being one disk is being monopolized
* Incorrectly configured resources such as application or ill designed databases

Some of the key areas to monitor to identify bottlenecks are as follows:

* CPU utilization
* Memory usage
* Disk input/output (I/O)
* User connections
* Blocking locks

So now that we know what resources to monitor, how should we begin?

Before going out and buying a faster server or upgrading the existing server with faster CPUS, more memory and faster disks, obtain a baseline of metrics. And always rule out software issues first such as poorly designed databases and insufficient indexes

Start with obtaining a performance baseline.

You monitor the server over time so that you can determine Server average performance, identify peak usage, determine the time required for backup and restore activities, and so on. This gives you a baseline from which you can judge the server against to determine if you have a performance bottleneck. This may be a weekly or monthly set of metrics that you obtain. Each depends upon you environment.

Hardware bottlenecks

Most hardware bottlenecks can be attributed to disk throughput, memory usage, processor usage, or network bandwidth.

One problem can often mask another. You might suspect hard disk performance as the cause of degraded query performance.

You can sometimes relieve bottlenecks by rescheduling resource-intensive SQL Server activities so that there’s less of a load on the server.

One area you should check is the performance of queries.

If you have queries that use excessive server resources, overall performance suffers.

SQL Profiler and the Database Engine Tuning Advisor can both help you gather information about queries.

CPU utilization

* A chronically high CPU utilization rate may indicate that Transact-SQL queries need to be tuned or that a CPU upgrade is needed. CPU usage greater than 80% consistently (plateauing)

Memory usage

* Insufficient memory allocated or available to Microsoft SQL Server degrades performance. SQL Server is a hog for memory and the more memory the better. If in sufficient memory is not allocated to the SQL Server, then Data must be read from the disk rather than directly from the data cache. This can cause performance issues.

Disk input/output (I/O)

* Transact-SQL queries can be tuned to reduce unnecessary I/O; for example, by employing indexes.

User connections

* Too many users may be accessing the server simultaneously causing performance degradation.

Blocking locks

* Incorrectly designed applications can cause locks and hamper concurrency, thus causing longer response times and lower transaction throughput rates.

Monitoring processor use

High processor utilization can be an indication that:

1. You need a processor upgrade (faster, better processor).
2. You need an additional processor.
3. You have a poorly designed application.

By monitoring processor activity over time, you can determine if you have a processor bottleneck and often identify the activities placing an excessive load on the processor.

Counters that you should monitor include those in the following table.

|  |  |  |
| --- | --- | --- |
| Object | Counter | Description |
| Processor | % Processor Time | Value should be consistently less than 90%. A consistent value of 90% or more indicates a processor bottleneck. Monitor this for each processor in a multiprocessor system and for total processor time (all processors). |
| System | Processor Queue Length | The number of threads waiting for processor time. A consistent value of 2 or more can indicate a processor bottleneck. |

In most cases, a processor bottleneck indicates you need to either upgrade to a faster processor or to multiple processors. In some cases, you can reduce the processor load by fine-tuning your application

Monitor SQL Server memory usage

* Memory: Available MBytes
* Memory: Pages/sec
* Memory: Page Faults/sec

Available MBytes is the amount of physical memory, in Megabytes, immediately available for allocation to a process or for system use. The higher the value the better

The Pages/sec counter indicates the number of pages that were retrieved from disk. A high value can indicate lack of memory.

Monitor the Memory: Page Faults/sec counter to make sure that the disk activity is not caused by paging.

By default, SQL Server changes its memory requirements dynamically so there is no need to alter the memory parameters; but if you need to alter the memory, you can change that via GUI or TSQL

To monitor the amount of memory that SQL Server specifically uses monitor

* SQL Server: Buffer Manager: Buffer Cache Hit Ratio
* SQL Server: Memory Manager: Total Server Memory (KB)
* SQL Server: Buffer Manager: Page Life Expectancy

The Buffer Cache Hit Ratio counter is specific to SQL Server; Percentage of pages that were found in the buffer pool without having to incur a read from disk. a rate of 90 percent or higher is desirable. A value greater than 90 percent indicates that more than 90 percent of all requests for data were satisfied from the data cache rather than the disk

If the Total Server Memory (KB) counter is consistently high compared to the amount of physical memory in the computer, it may indicate that more memory is required.

Page Life Expectancy is the number of seconds a page will stay in the buffer pool without references. The recommended value of the Page Life Expectancy counter is approximately 300 seconds. Simply put, the longer the page stays in the buffer pool the better performance you will get rather than getting the data from the disk

Some common counters used for analyzing disk performance are:

* Avg. Disk sec/Read - the average time, in seconds, of a read of data from the disk.
* Avg. Disk sec/Write - is the average time, in seconds, of a write of data to the disk.

These metrics help how long the disks took to service an I/O request no matter what kind of hardware you using, weather its physical or virtual hardware. If your system had many disks, then it’s important to measure each disk for discovering bottlenecks

The average should be under 15ms with maximums up to 30ms. The less time it takes to read or write data the faster your system will be

* Disk Transfers/sec - is the rate of read and write operations on the disk.
* Disk Reads/sec - is the rate of read operations on the disk.
* Disk Writes/sec - is the rate of write operations on the disk.
* Avg. Disk Queue Length - is the average number of both read and write requests that were queued for the selected disk during the sample interval.
* Current Disk Queue Length - is the number of requests outstanding on the disk at the time the performance data is collected.