

The SQL Server 2008 Data Collector: Part 3

14 Jun 2010 4:47 PM | [Q](#)

This is the third post in a series about the data collector in SQL Server 2008. In the [first post](#), we looked at how to set up the data collector; in the [second post](#), we looked at the standard counters that are captured by the data collector.

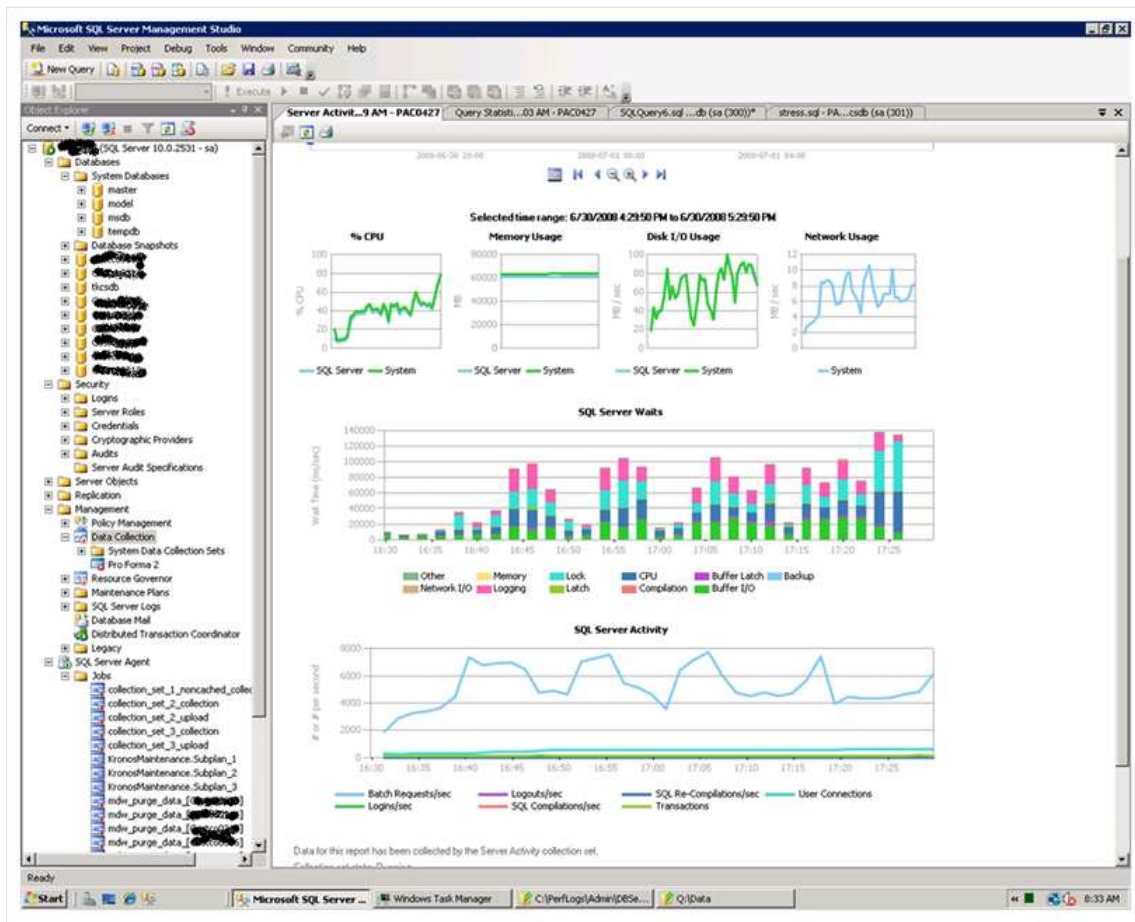
In this post, we will look at retrieving the data for charts. The data collector includes some built-in charts to get you started. We will also look at some simple techniques to do data mining for significant measures. We will conclude by looking at the ability to compare counters from different periods (or runs) on SQL Server.

Built-in Reports

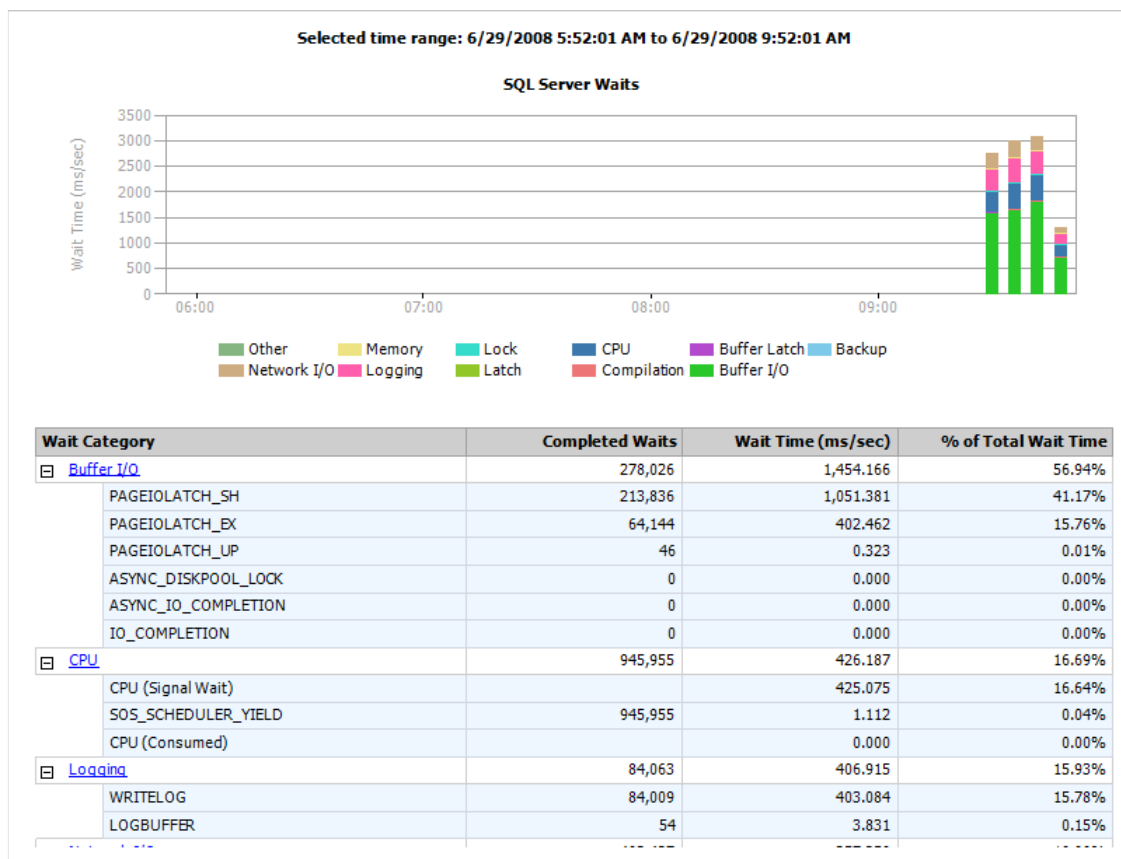
To find the built-in reports for the data collector, follow these steps:

1. Start SQL Server Management Studio, and go to one of the data collector databases.
2. Right-click the database, and then click **Reports**.
3. Click **Management Data Warehouse**.
4. Click **Management Data Warehouse Overview**.

A chart that resembles the following chart is displayed.



If you click items, you can drill down into more detail. For example:



Beyond the Built-in Reports

The built-in charts may not provide the data in the format or presentation that you want. Fortunately, the data is saved in SQL Server tables, so you can use Microsoft Excel, Microsoft Access, or other tools to quickly produce specific charts or data series for analysis.

Identifying what to look at.

My experience is that a flat line is not interesting to look at, especially when there may be a thousand measurements. One solution is to use drop-down combo boxes that list items that have changed a specific percentage over the period that you are examining. You can populate these combo boxes by using two Transact-SQL statements to identify the measures that have a significant change. In the following examples, I use 5 percent or 1.05 as the threshold for a significant change.

```
Select [path] from [snapshots].[performance_counters]
group by [path]
having max(formatted_value) > 1.05 * (min(formatted_value))
order by [path]
```

and

```
Select [wait_type]+'.[Waiting_Tasks_Count]'
from [snapshots].[OS_Wait_stats]
group by [wait_type]
having Max(Waiting_Tasks_Count) > 1.05 * (Min(Waiting_Tasks_Count)+1)
UNION
Select [wait_type]+'.[Wait_Time_ms]'
from [snapshots].[OS_Wait_stats]
group by [wait_type]
having Max(Wait_Time_ms) > 1.05 * (1+Min(Wait_Time_ms))
UNION
Select [wait_type]+'.[Signal_Wait_time_ms]'
from [snapshots].[OS_Wait_stats]
group by [wait_type]
having Max(Signal_Wait_time_ms) > 1.05 * (1+Min(Signal_Wait_time_ms))
```

If you increase the significance threshold from 5 percent (1.05) to 50 percent (1.50), you reduce the number of items that are returned, as shown in the following table.

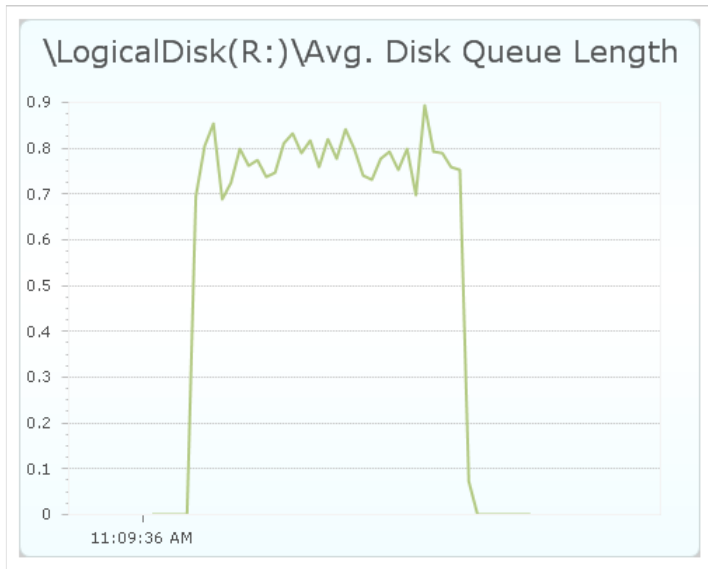
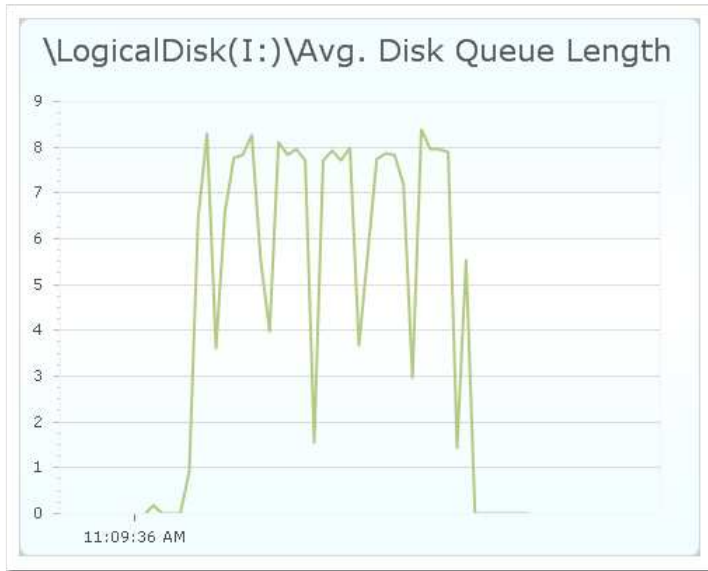
Measure	All	At 5%	At 50% (+1)
Performance Monitor	757	463	289

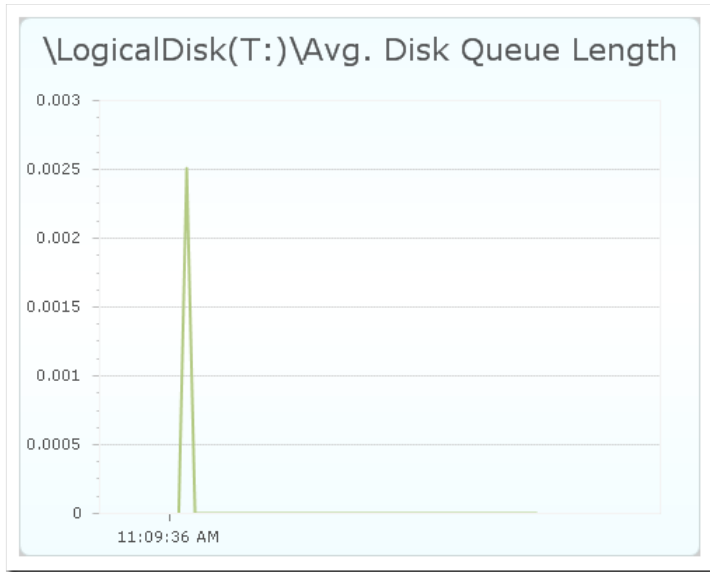
Wait type	378	3	0
-----------	-----	---	---

Of course, you are free to implement other statistical strategies to eliminate insignificant data.

The following charts illustrate what you extract from the two key tables, **[snapshots].[performance_counters]** and **[snapshots].[OS_Wait_stats]**. These tables have a column that is called **[collection_time]**, which indicates when the data was captured. (Every minute is the default.)

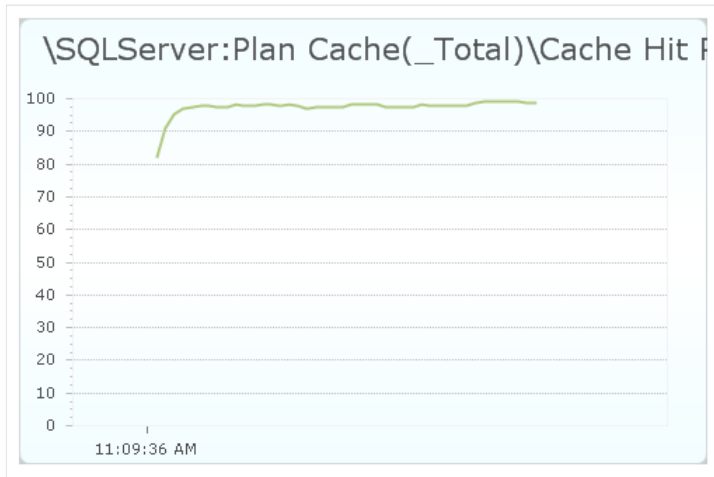
These charts give us a simple example. The first chart shows that disk I is struggling when it performs I/O (as seen by the high queue length). The second chart shows that disk R is able to satisfy I/O reasonably well but may not be able to handle a significant increase in load. The third chart shows that disk T appears to have very little I/O compared to the performance characteristics of the disk.





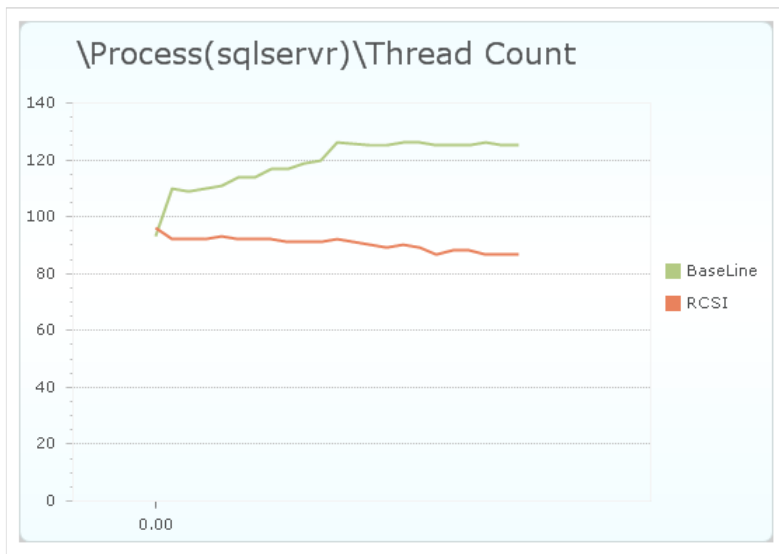
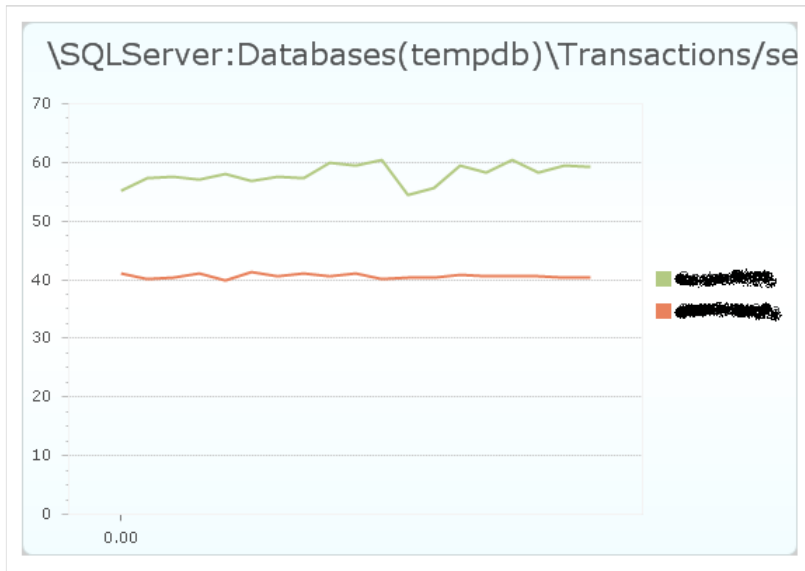
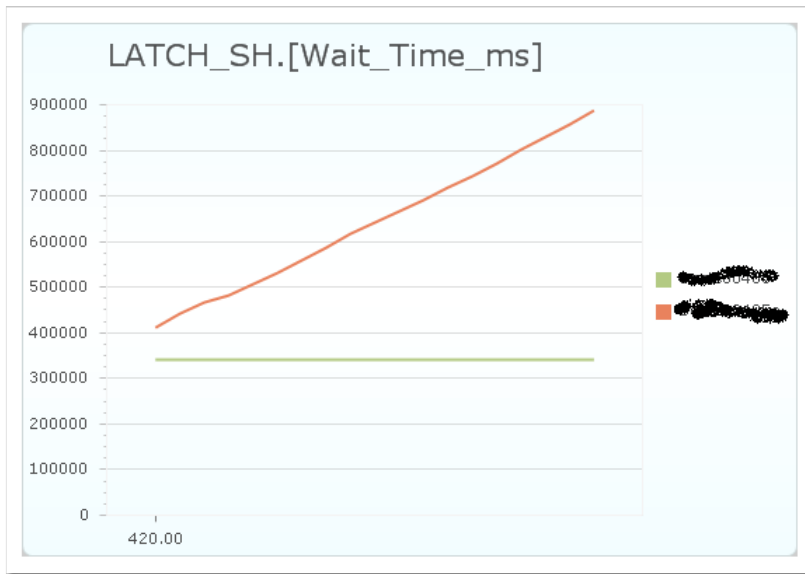
The high queue length on disk I suggests that you could improve performance by partitioning or moving tables to other drives.

Another chart shows that the cache hit ratio is ramping up and stabilizing where it should.



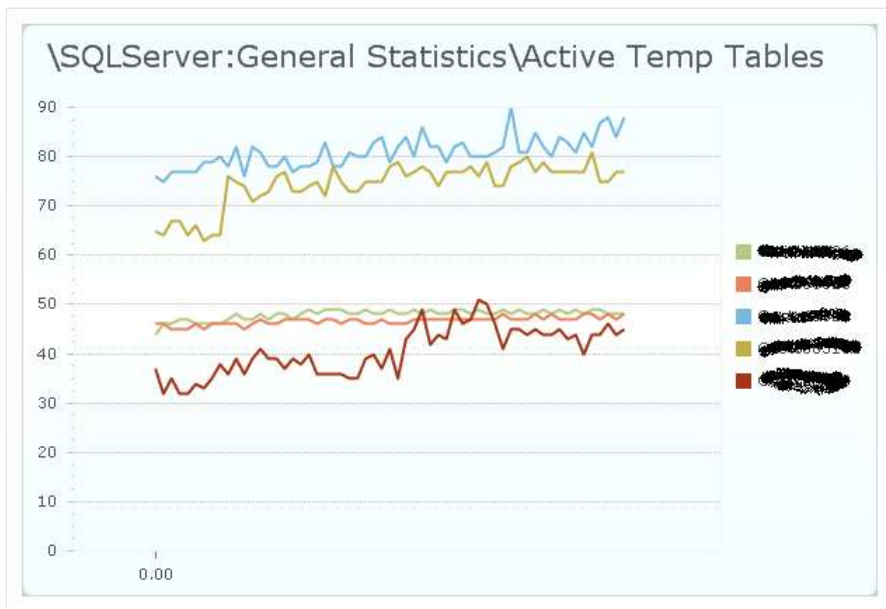
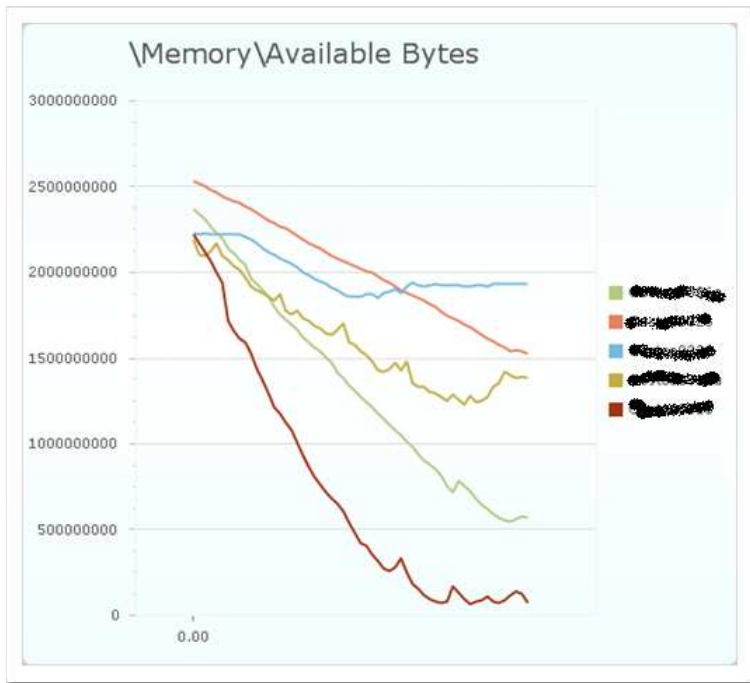
Comparing runs.

You can pull data into Excel and other products directly from the data collector database tables. Additionally, you can perform joins of data from different runs and periods and compare them. The following examples are the same charts from two different runs: one with RCSI enabled and the other with RCSI disabled.



Comparing multiple runs (or days of the week).

These charts illustrate comparing two runs with a change of a single factor. You can also compare different days of the week or the same days from different weeks to see changes over time. A few examples produced from the data collector are as follows.



Production Monitoring

I recommend that you keep the data collector on continuously to monitor what is going on:

- The load on the server is very light because the data collector captures information only once each minute.
- The data is retained for just one week, so you will not have a storage issue.
- If a problem occurs, trying to reproduce it is very time consuming (and often impossible). If the data collector is running, you have a much better chance of identifying the problem and preventing it for occurring again.

I suggest that you set aside one week of data each quarter so that you can do a quarter-by-quarter comparison of the changes in load and the SQL Server response to these changes. In a few hours and with lots of charts, you can track down and understand the cause of problems.

The data collector provides you with a rich set of performance and behavior data at a very low cost. It should be part of any standard practices.

Ken Lassenen is part of the original team that created Dr. GUI of MSDN and specializes in new and resurrected commercial product architecture. He developed architecture for several Microsoft websites, including the original [MSDN](#) site and the current [Microsoft Partner Network](#) site. He's equally at

home with SQL Server, XHTML, Section 508 accessibility standards, globalization, Security Content Automation Protocol (SCAP) security, C#, and ASP.NET server controls. When he is not having fun with technology, he enjoys taking lunch-break hikes in the North Cascades.

