



SQL Server Performance Tuning



Who am I

Name: Scott Heffron, MCAD
Title: Database Engineer
Company: Access Development

Software:	DataBases:
•ASP.NET	•MS SQL Server, MCP 2000
•C++	•Sybase
•C# Compact Framework	•Oracle
•ColdFusion	•SQL Anywhere



Agenda

- What is Performance Tuning
- Areas of Performance Tuning
- Target Areas
- Case Studies
- Where to go from here
- Questions



What is Performance Tuning

The problem is a simple one: How do you get the most value from your SQL Server deployments? Faced with this problem, many of us ask: Am I getting the best efficiency? Will my application scale?

A scalable system is one in which the demands on the database server increase in a predictable and reasonable manner.



What are the Tuning Drivers

- Meet service level agreements (SLA)
- Improve Efficiency
- Ensure scalability



Tuning Areas

- | | |
|-----------------------------|---------------------------|
| ■ Baselining & Benchmarking | ■ Cursors |
| ■ Performance Counters | ■ Bottlenecks |
| ■ SQL Profiler | ■ Buffer Cache Management |
| ■ Managing Indexes | ■ Query Plans |
| | ■ sp_configure settings |



Baselining & Benchmarking

Means to an End



Baselining and benchmarking give us a picture of the resource consumption over a period of time.

- Observe the application in real time in a test area
- Play back a recording of the execution in real time

The best outcome is reached by observing the actual workload. Either in real time or simulation



Performance Counters

What counters should I monitor? In terms of managing SQL Server, there are two broad reasons for monitoring performance counters:

- Operational
- Bottlenecks

Although they have some overlap, these two reasons allow you to easily choose a number of data points to monitor.



Operational Monitoring

- Resources like CPU, Disk Space, or Memory
- Data files allowed to grow
- Trend Analysis

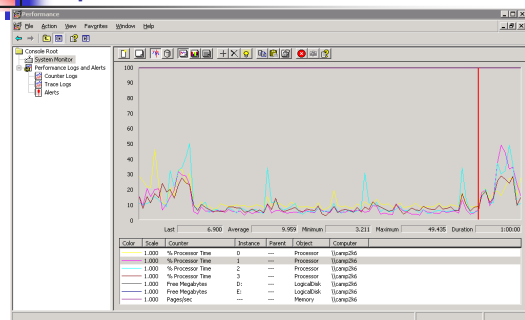


Operational Counters

Counter	Reason
Processor\% Processor Time	Monitor CPU consumption
LogicalDisk\Free Megabytes	Monitor the free space on the disk(s)
Databases\Data File(s) Size (KB)	Trend growth over time
Memory\Pages/sec	Check for paging, a good indication that memory resources might be short



Operational Performance





Bottleneck Monitoring

- Is there a CPU bottleneck
- Is there an I/O bottleneck
- Are the subsystems healthy
 - Buffer Cache
 - Procedure Cache
- Is there contention in the database

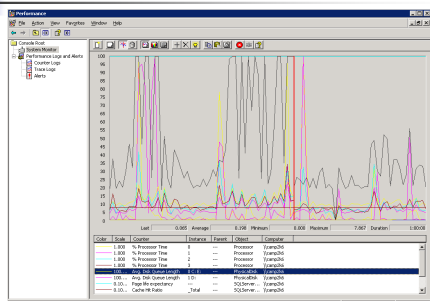


Bottleneck Counters

Counter	Reason
Processor\% Processor Time	Monitor CPU consumption allows us to check for a bottleneck on the server
PhysicalDisk\Avg. Disk Queue Length	Check for disk bottlenecks – if the value exceeds 21 then it is likely that a disk bottleneck exists.
Buffer Manager\Page Life Expectancy	Page Life Expectancy is the number of seconds a page stays in the buffer cache. A low number indicates that pages are being evicted without spending much time in the cache, thereby reducing the effectiveness of the cache.
SQL Server:Cache Manager\Cache Hit Ratio	A low Plan Cache hit ratio means that plans are not being reused.



Bottleneck Performance





Performance Counters Review

- Memory
 - Available Mbytes
 - Pages/sec
- Paging File - % usage
- Physical Disk
 - % Disk Time
 - Avg. Disk Queue Length
 - Avg. Disk sec/Read
 - Avg. Disk sec/Write
 - Disk Reads/sec
 - Disk Writes/sec
- Processor - % Processor Time
- SQL Server:Buffer Manager
 - Buffer cache hit ratio
 - Page Life Expectancy
- SQL Server:General Statistics
 - User Connections
- System – Processor Queue Length



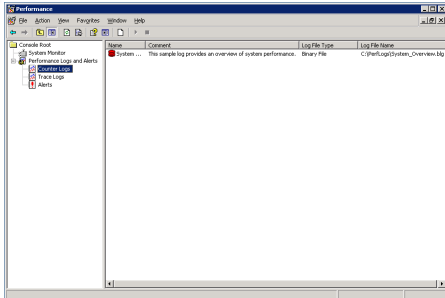
Create Performance Counter File



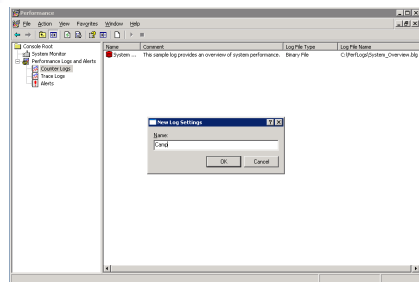
Monitoring Workstation

- Always on
- Not your workstation
- SQL Express 2005
- SQL Tools
 - SSMS
 - Profiler
 - Performance Monitor

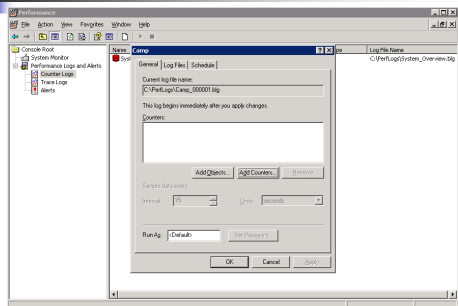
Create a Performance Counter File



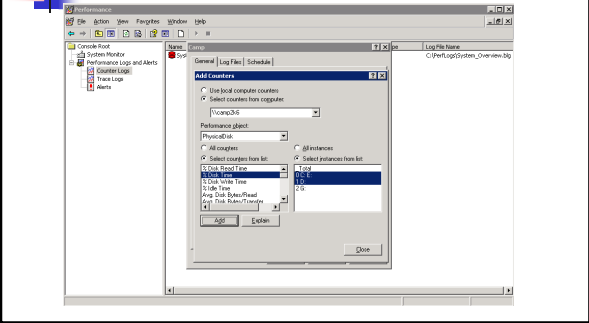
Step #1: Give it a name



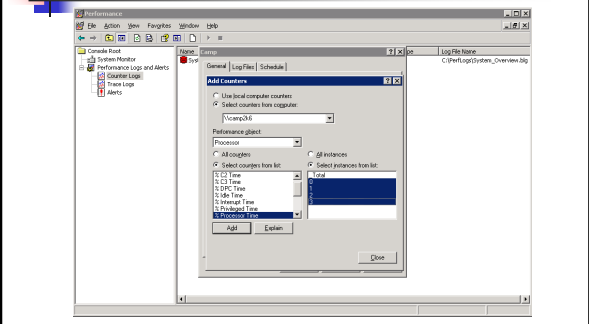
Step #2: Add the Counters



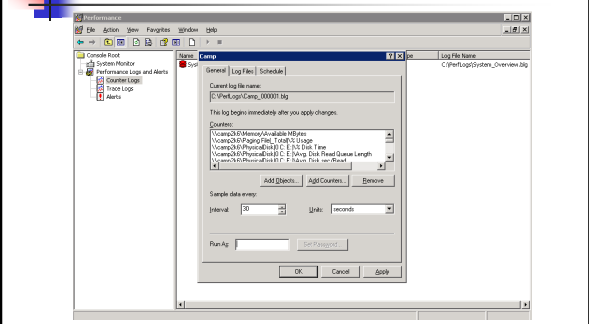
Step #3: Select certain Counters



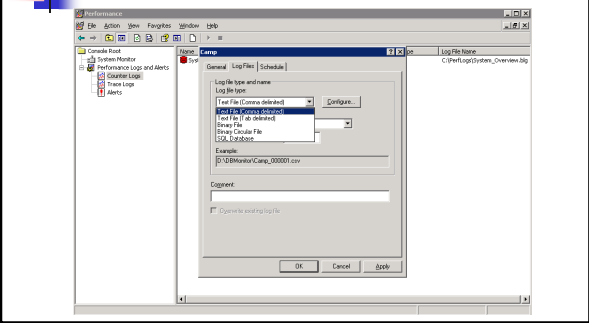
Check on Disks



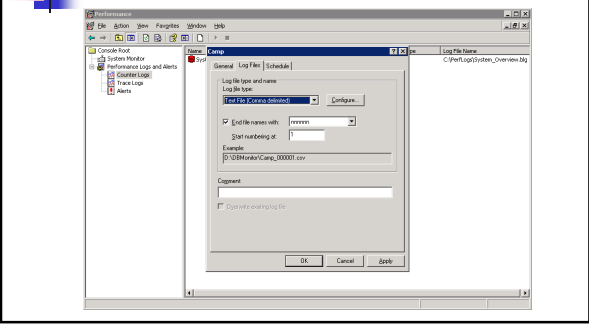
Step #4: Review Counters



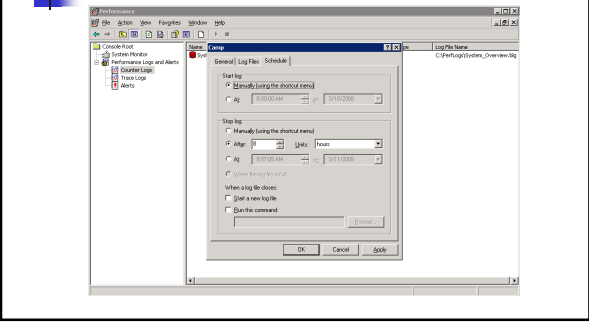
Step #5: Configure Destination



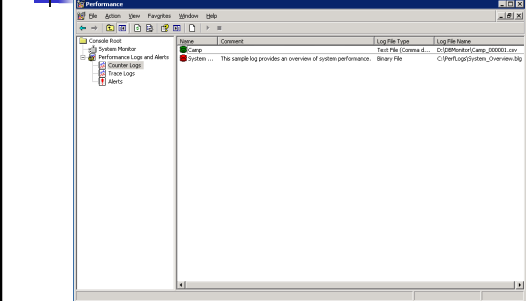
Step #5: Configure Destination - Continued



Step #6: Schedule Time



Step #7: Start



Review File Example

Profiler

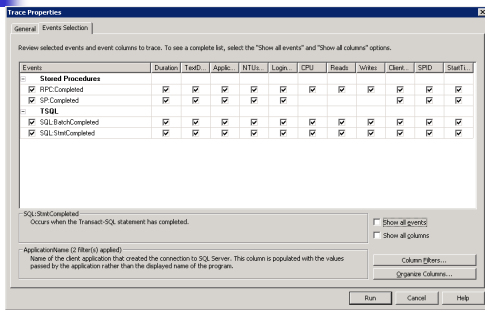
SQL Profiler

I use this tool to capture queries or stored procedures that take longer to run than my SLA allow.

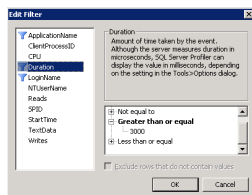
I run this tool and generally check on:

- RPC: Completed
- SP:Completed
- SQL:BatchCompleted
- SQL:StmtdCompleted

Stored Procedure & TSQL Events



Column Filters Section





Indexes usage looked at differently

- SQL Server 2000
- SQL Server 2005



SQL Server 2000 - Indexes



SQL

```
SELECT 'DBCC SHOWCONTIG('
      + CAST(id AS NVARCHAR(20)) + ') '
      + CHAR(10) + 'PRINT ' ' ' + CHAR(10)
FROM sysindexes
WHERE indid = 1
      OR indid = 0
ORDER BY rows DESC
```



Result of a ShowContig

```

DBCC SHOWCONTIG scanning 'EmailSendQueue2k5' table...
Table: 'EmailSendQueue2k5' (1479884539); index ID: 1, database ID: 7
TABLE level scan performed.
- Pages Scanned.....: 592720
- Extents Scanned.....: 74177
- Extent Switches.....: 127240
- Avg. Pages per Extent.....: 8.0
- Scan Density [Best Count:Actual Count].....: 58.23% [74090:127241]
- Logical Scan Fragmentation.....: 10.29%
- Extent Scan Fragmentation.....: 24.90%
- Avg. Bytes Free per Page.....: 944.7
- Avg. Page Density (full).....: 88.33%

```

Is a percentage. It is the ratio **Best Count** to **Actual Count**. This value is 100 if everything is contiguous; if this value is less than 100, some fragmentation exists.



SQL Server 2005 - Indexes



SQL

```

SELECT object_name(object_id) as name
, user_seeks
, user_scans
, user_lookups
, last_user_seek
, last_user_scan
, last_user_lookup
, user_updates
FROM sys.dm_db_index_usage_stats
WHERE object_name(object_id) NOT LIKE 'sys%'
ORDER BY user_seeks desc

```

Example

```
Use AdventureWorks
Go

SELECT City
       , StateProvinceID
       , PostalCode
FROM Person.Address
WHERE StateProvinceID = 9
GO
```

Results

	table_name	column_name	column_usage
1	[AdventureWorks].[Person].[Address]	City	INCLUDE
2	[AdventureWorks].[Person].[Address]	StateProvinceID	EQUALITY
3	[AdventureWorks].[Person].[Address]	PostalCode	INCLUDE

Index SQL:
CREATE NONCLUSTERED INDEX IX_Address_StateProvinceID ON Person.Address
(
StateProvinceID ASC
)

SQL:
SELECT City, StateProvinceID, PostalCode
FROM Person.Address
WHERE StateProvinceID = 9

Column Usage

Value	Description
Equality	Column contributes to a predicate that expresses equality, of the form: table.column = constant_value
Inequality	Column contributes to a predicate that expresses inequality, for example, a predicate of the form: table.column > constant_value
Include	Column is not used to evaluate a predicate, but is used for another reason, for example, to cover a query.

Most Expensive I/O Queries

SQL

```

SELECT TOP 20
SUBSTRING(qt.text, (qs.statement_start_offset/2)+1
, ((CASE qs.statement_end_offset
WHEN -1 THEN DATALENGTH(qt.text)
ELSE qs.statement_end_offset
END - qs.statement_start_offset)/2)+1)
, qs.execution_count
, qs.total_logical_reads
, qs.last_logical_reads
, qs.min_logical_reads
, qs.max_logical_reads
, qs.total_elapsed_time
, qs.last_elapsed_time
, qs.min_elapsed_time
, qs.max_elapsed_time
, qs.last_execution_time
, qp.query_plan
FROM sys.dm_exec_query_stats qs
CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) qt
CROSS APPLY sys.dm_exec_query_plan(qs.plan_handle) qp
WHERE qt.encrypted=0
ORDER BY qs.total_logical_reads DESC
    
```

SQL Results

SQL column name	execution_count	total_logical_reads	last_logical_reads	min_logical_reads	max_logical_reads	total_elapsed_time
1. insert into Reprocessor exec sys.sp_send_column @c...	232285	42906691	2320	35	18783	1548304575
2. select PROCEDURE_QUALIFIER = _s_sco...	229237	39861534	42	30	18006	1886360877
3. [select distinct Country, State, StateLong from Properties...]	142	4503037	2512	2459	2401	20201024
4. update EnrolmentQueue3 set SerCount = (@SerCou...	26403	4934863	5	5	1938	209456458
5. SELECT FROM [EnrolmentQueue3] WHERE [Process...	8966	9891989	1099	1098	1106	5629102
6. SELECT @source = COUNT(*) FROM dbo.FailedLo...	2	781608	2909391	2909394	2909391	30081549
7. SELECT FROM [EnrolmentQueue3] WHERE [Inscr...	78972	5442072	69	69	121	31369759
8. UPDATE [EnrolmentQueue3] set ProcessingTime = ...	3566	3918075	1099	1099	1099	133916673
9. UPDATE [EnrolmentQueue3] set ProcessingTime = ...	3566	3918075	1099	1099	1099	136520002
10. INSERT INTO FailedEnrolmentQueue SELECT ...	1	3919545	3919545	3919545	3919545	15703071
11. SELECT @source = COUNT(*) FROM dbo.FailedLo...	1	3998724	3998724	3998724	3998724	14896916
12. IF EXISTS (SELECT 1 FROM dbo.process WITH ...	789888	2398008	3	3	3	23623839
13. SELECT @source = FailedGroup FROM dbo.group...	790295	2370795	3	3	3	14701521
14. SELECT [Introuper][@id][@id] FROM [conpout...	349543	2097366	6	6	84	74644382
15. select * from EnrolmentQueue3 where ProcessingTime...	1795	1361395	1099	1098	1099	13007983
16. select * from EnrolmentQueue3 where [State] = 'In Job...	1791	1958290	1099	1098	1099	13369169
17. SELECT [Inscr][@id][@id][@id] FROM ...	27221	18951029	68	68	68	26294969
18. IF EXISTS (select distinct FailedGroup from process where...	473664	1420992	3	3	3	13190769
19. select @@COSTM + SQL @@COSTM + SPY...	473664	1420992	3	3	3	10746666
20. [select distinct @Name = Home @@COSTM = ...	438207	13717671	3	3	3	8089378



Case Studies

DB CSI Database Crime Scene Investigation



Case Study #1



Case Overview

- Performance Area(s):**
- Baselining & Benchmarking
 - Overcoming Bottlenecks
 - Horrors of Cursors
 - Readability

There is a SQL Server job that gathers together data and then produces files to send to a business partner. This job normally takes from 6 – 8 minutes to complete. It is ran at 10am each morning.

On February 29th it started as normal. Except that is ran for 69+ hours. The next time it ran was March 3rd, but then it ran for 18 hours 30 minutes. I canceled because of the duration was to long.



Email

Email Sent out on March 3rd.

Some of you might already know, but we are having major issues with MAD propagation. This is preventing several steps in our merchant boarding process and maintenance:

MAD is not sending all daily MID files to GRS

IT is working on the issue and it's a high priority for them. I just wanted to pass along any updates. If any of you have any further updates, please let us know. The MR team really relies on all these processes in order to our job.




Investigation Steps

- Look at the log file (DTS)
- Found stored procedure causing issues
- Tore apart the stored procedure




Investigation Found

- Went to a certain record and froze
- Looked at the record and found that the parent and child had the same number
- Data Entry fixed their error.




Case Study #2



Case Overview

Performance Area(s):
Baselining & Benchmarking
Overcoming Bottlenecks
Improve Efficiency
Scale Stability

We have an application that is now timing out when a user queries for location information.



Investigation Steps

- Turn Profiler on and watch the system
- Have users hit that area
- Gather object information



Investigation Findings

- Many tables in join
- Tables had gotten to big



Case Outcome

- No changes to application needed
- System returning data faster
- Major table not being used for search



Case Solution

- Create Searching Table
- Refresh on weekly schedule
- Insert, Update and Delete using triggers



Questions & Answers



Contact Information

Email: Scott.Heffron@AccessDevelopment.com



Thank You
