<http://glennberrysqlperformance.spaces.live.com/blog/cns!45041418ECCAA960!1340.entry>

Updated SQL Server 2005 Diagnostic Queries

This is a set of queries that I often have clients run to give me a pretty decent high level overview of how their SQL Server system is configured and how it is performing.

The initial queries are instance or server specific, while the later ones are database specific (which means you have to switch to the database you are concerned with).  Most of the DMV queries require VIEW SERVER STATE permission on the database server.

These queries will work on SQL Server 2005 or 2008, but not on SQL Server 2000.

-- SQL Server 2005 Diagnostic Information Queries

-- Glenn Berry April 2009

-- http://glennberrysqlperformance.spaces.live.com/

-- SQL Version information for current instance

SELECT @@VERSION AS 'Version Info';

-- Hardware Information

SELECT cpu\_count AS 'Logical CPU Count', hyperthread\_ratio AS 'Hyperthread Ratio',

cpu\_count/hyperthread\_ratio As 'Physical CPU Count',

physical\_memory\_in\_bytes/1048576 AS 'Physical Memory (MB)'

FROM sys.dm\_os\_sys\_info;

-- get sp\_configure values for instance

EXEC sp\_configure 'Show Advanced Options', 1

GO

RECONFIGURE

GO

EXEC sp\_configure

-- File Names and Paths for all databases in instance

SELECT dbid, fileid, filename

FROM sys.sysaltfiles;

-- Recovery model for all databases on instance

SELECT [name], recovery\_model\_desc, log\_reuse\_wait\_desc, [compatibility\_level]

FROM sys.databases;

-- Clear Wait Stats

-- DBCC SQLPERF('sys.dm\_os\_wait\_stats', CLEAR);

-- Isolate top waits for server instance since last restart or statistics clear

WITH Waits AS

(SELECT wait\_type, wait\_time\_ms / 1000. AS wait\_time\_s,

100. \* wait\_time\_ms / SUM(wait\_time\_ms) OVER() AS pct,

ROW\_NUMBER() OVER(ORDER BY wait\_time\_ms DESC) AS rn

FROM sys.dm\_os\_wait\_stats

WHERE wait\_type NOT IN( 'SLEEP\_TASK', 'BROKER\_TASK\_STOP',

'SQLTRACE\_BUFFER\_FLUSH', 'CLR\_AUTO\_EVENT', 'CLR\_MANUAL\_EVENT',

'LAZYWRITER\_SLEEP')) -- filter out additional irrelevant waits

SELECT W1.wait\_type,

CAST(W1.wait\_time\_s AS DECIMAL(12, 2)) AS wait\_time\_s,

CAST(W1.pct AS DECIMAL(12, 2)) AS pct,

CAST(SUM(W2.pct) AS DECIMAL(12, 2)) AS running\_pct

FROM Waits AS W1

INNER JOIN Waits AS W2

ON W2.rn <= W1.rn

GROUP BY W1.rn, W1.wait\_type, W1.wait\_time\_s, W1.pct

HAVING SUM(W2.pct) - W1.pct < 95; -- percentage threshold

-- Signal Waits for instance

SELECT '%signal (cpu) waits' = CAST(100.0 \* SUM(signal\_wait\_time\_ms) / SUM (wait\_time\_ms) AS NUMERIC(20,2)),

'%resource waits'= CAST(100.0 \* SUM(wait\_time\_ms - signal\_wait\_time\_ms) / SUM (wait\_time\_ms) AS NUMERIC(20,2))

FROM sys.dm\_os\_wait\_stats;

-- Page Life Expectancy value

SELECT cntr\_value AS 'Page Life Expectancy'

FROM sys.dm\_os\_performance\_counters

WHERE object\_name = 'SQLServer:Buffer Manager'

AND counter\_name = 'Page life expectancy';

-- Buffer Pool Usage

SELECT TOP (10) [type], sum(single\_pages\_kb) AS [SPA Mem, Kb]

FROM sys.dm\_os\_memory\_clerks

GROUP BY type

ORDER BY SUM(single\_pages\_kb) DESC;

-- Switch to user database

--USE YourDatabaseName;

--GO

-- Individual File Sizes and space available for current database

SELECT name AS 'File Name' , physical\_name AS 'Physical Name', size/128 AS 'Total Size in MB',

size/128.0 - CAST(FILEPROPERTY(name, 'SpaceUsed') AS int)/128.0 AS 'Available Space In MB' FROM sys.database\_files;

-- Cached SP's By Execution Count

SELECT TOP (25) qt.text AS 'SP Name', qs.execution\_count AS 'Execution Count',

qs.execution\_count/DATEDIFF(Second, qs.creation\_time, GetDate()) AS 'Calls/Second',

qs.total\_worker\_time/qs.execution\_count AS 'AvgWorkerTime',

qs.total\_worker\_time AS 'TotalWorkerTime',

qs.total\_elapsed\_time/qs.execution\_count AS 'AvgElapsedTime',

qs.max\_logical\_reads, qs.max\_logical\_writes, qs.total\_physical\_reads,

DATEDIFF(Minute, qs.creation\_time, GetDate()) AS 'Age in Cache'

FROM sys.dm\_exec\_query\_stats AS qs

CROSS APPLY sys.dm\_exec\_sql\_text(qs.sql\_handle) AS qt

WHERE qt.dbid = db\_id() -- Filter by current database

ORDER BY qs.execution\_count DESC;

-- Cached SP's By Worker Time

SELECT TOP (25) qt.text AS 'SP Name', qs.total\_worker\_time AS 'TotalWorkerTime',

qs.total\_worker\_time/qs.execution\_count AS 'AvgWorkerTime',

qs.execution\_count AS 'Execution Count',

ISNULL(qs.execution\_count/DATEDIFF(Second, qs.creation\_time, GetDate()), 0) AS 'Calls/Second',

ISNULL(qs.total\_elapsed\_time/qs.execution\_count, 0) AS 'AvgElapsedTime',

qs.max\_logical\_reads, qs.max\_logical\_writes,

DATEDIFF(Minute, qs.creation\_time, GetDate()) AS 'Age in Cache'

FROM sys.dm\_exec\_query\_stats AS qs

CROSS APPLY sys.dm\_exec\_sql\_text(qs.sql\_handle) AS qt

WHERE qt.dbid = db\_id() -- Filter by current database

ORDER BY qs.total\_worker\_time DESC;

-- Cached SP's By Logical Reads

SELECT TOP (25) qt.text AS 'SP Name', total\_logical\_reads,

qs.execution\_count AS 'Execution Count', total\_logical\_reads/qs.execution\_count AS 'AvgLogicalReads',

qs.execution\_count/DATEDIFF(Second, qs.creation\_time, GetDate()) AS 'Calls/Second',

qs.total\_worker\_time/qs.execution\_count AS 'AvgWorkerTime',

qs.total\_worker\_time AS 'TotalWorkerTime',

qs.total\_elapsed\_time/qs.execution\_count AS 'AvgElapsedTime',

qs.total\_logical\_writes,

qs.max\_logical\_reads, qs.max\_logical\_writes, qs.total\_physical\_reads,

DATEDIFF(Minute, qs.creation\_time, GetDate()) AS 'Age in Cache'

FROM sys.dm\_exec\_query\_stats AS qs

CROSS APPLY sys.dm\_exec\_sql\_text(qs.sql\_handle) AS qt

WHERE qt.dbid = db\_id() -- Filter by current database

ORDER BY total\_logical\_reads DESC;

-- Possible bad Indexes (writes > reads)

SELECT object\_name(s.object\_id) AS 'Table Name', i.name AS 'Index Name', i.index\_id,

user\_updates AS 'Total Writes', user\_seeks + user\_scans + user\_lookups AS 'Total Reads',

user\_updates - (user\_seeks + user\_scans + user\_lookups) AS 'Difference'

FROM sys.dm\_db\_index\_usage\_stats AS s WITH (NOLOCK)

INNER JOIN sys.indexes AS i WITH (NOLOCK)

ON s.object\_id = i.object\_id

AND i.index\_id = s.index\_id

WHERE objectproperty(s.object\_id,'IsUserTable') = 1

AND s.database\_id = db\_id()

AND user\_updates > (user\_seeks + user\_scans + user\_lookups)

AND i.index\_id > 1

ORDER BY 'Difference' DESC, 'Total Writes' DESC, 'Total Reads' ASC;

-- Missing Indexes for entire instance by Index Advantage

SELECT user\_seeks \* avg\_total\_user\_cost \* (avg\_user\_impact \* 0.01) AS index\_advantage, migs.last\_user\_seek,

mid.statement AS [Database.Schema.Table],

mid.equality\_columns, mid.inequality\_columns, mid.included\_columns,

migs.unique\_compiles, migs.user\_seeks, migs.avg\_total\_user\_cost, migs.avg\_user\_impact

FROM sys.dm\_db\_missing\_index\_group\_stats AS migs WITH (NOLOCK)

INNER JOIN sys.dm\_db\_missing\_index\_groups AS mig WITH (NOLOCK)

ON migs.group\_handle = mig.index\_group\_handle

INNER JOIN sys.dm\_db\_missing\_index\_details AS mid WITH (NOLOCK)

ON mig.index\_handle = mid.index\_handle

ORDER BY index\_advantage DESC;

-- Breaks down buffers used by current database by object (table, index) in the buffer cache

SELECT OBJECT\_NAME(p.object\_id) AS 'ObjectName', p.object\_id,

p.index\_id, COUNT(\*)/128 AS 'buffer size(MB)', COUNT(\*) AS 'buffer\_count'

FROM sys.allocation\_units AS a

INNER JOIN sys.dm\_os\_buffer\_descriptors AS b

ON a.allocation\_unit\_id = b.allocation\_unit\_id

INNER JOIN sys.partitions AS p

ON a.container\_id = p.hobt\_id

WHERE b.database\_id = db\_id()

GROUP BY p.object\_id, p.index\_id

ORDER BY buffer\_count DESC;

-- Detect blocking

SELECT t1.resource\_type AS 'lock type',db\_name(resource\_database\_id) AS 'database',

t1.resource\_associated\_entity\_id AS 'blk object',t1.request\_mode AS 'lock req', --- lock requested

t1.request\_session\_id AS 'waiter sid', t2.wait\_duration\_ms AS 'wait time', -- spid of waiter

(SELECT [text] FROM sys.dm\_exec\_requests AS r -- get sql for waiter

CROSS APPLY sys.dm\_exec\_sql\_text(r.sql\_handle)

WHERE r.session\_id = t1.request\_session\_id) AS 'waiter\_batch',

(SELECT substring(qt.text,r.statement\_start\_offset/2,

(CASE WHEN r.statement\_end\_offset = -1

THEN LEN(CONVERT(nvarchar(max), qt.text)) \* 2

ELSE r.statement\_end\_offset END - r.statement\_start\_offset)/2)

FROM sys.dm\_exec\_requests AS r

CROSS APPLY sys.dm\_exec\_sql\_text(r.sql\_handle) AS qt

WHERE r.session\_id = t1.request\_session\_id) AS 'waiter\_stmt', -- statement blocked

t2.blocking\_session\_id AS 'blocker sid', -- spid of blocker

(SELECT [text] FROM sys.sysprocesses AS p -- get sql for blocker

CROSS APPLY sys.dm\_exec\_sql\_text(p.sql\_handle)

WHERE p.spid = t2.blocking\_session\_id) AS 'blocker\_stmt'

FROM sys.dm\_tran\_locks AS t1

INNER JOIN sys.dm\_os\_waiting\_tasks AS t2

ON t1.lock\_owner\_address = t2.resource\_address;