[**Advanced SQL Server performance tuning**](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

By: [Paul Randal](http://www.sqlskills.com/blogs/paul)

Posted on: April 13, 2011 7:32 am

FROM: <http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/>

(Check out my Pluralsight online training course: [*SQL Server: Performance Troubleshooting Using Wait Statistics*](http://www.pluralsight.com/training/Courses/TableOfContents/sqlserver-waits)and my [comprehensive library of all wait types and latch classes](https://www.sqlskills.com/help/waits).)

It’s all very well having whizz-bang 3rd-party performance monitoring and troubleshooting tools, but sometimes you have to get deeper into what’s going on with SQL Server than any of these tools can go. Or you have to call Customer Support or Premier Support so \*they\* can dive in deeper.

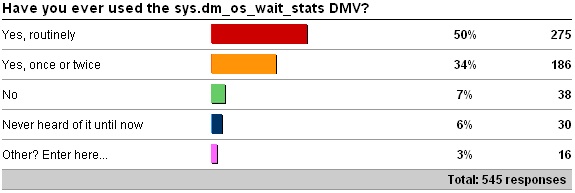
Typically you or they are going to make use of four DMVs that give increasingly advanced information about what’s going on for use in performance troubleshooting:

* [*sys.dm\_os\_wait\_stats*](http://msdn.microsoft.com/en-us/library/ms179984.aspx)
* [*sys.dm\_os\_waiting\_tasks*](http://msdn.microsoft.com/en-us/library/ms188743.aspx)
* [*sys.dm\_os\_latch\_stats*](http://msdn.microsoft.com/en-us/library/ms175066.aspx)
* *sys.dm\_os\_spinlock\_stats* (this one isn’t documented at all and is only mentioned in a few places online)

A few weeks ago I kicked off a survey to find out whether you’ve heard of or used these DMVs. In this post I’m going to present the survey results and explain a bit about these DMVs, focusing the most attention on latches and spinlocks. This started out as a small post but grew into a 10-page, 2500 word article :-)

Here are the results (in each of the Other values, a few people asked what DMVs are – see [Dynamic Management Views and Functions](http://msdn.microsoft.com/en-us/library/ms188754.aspx) in BOL).

**sys.dm\_os\_wait\_stats**



Other values:

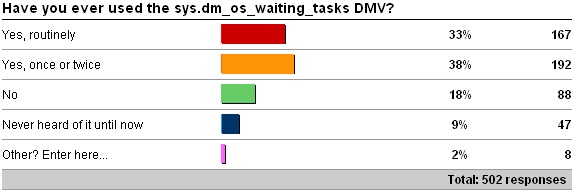
* 12 x Yes, more than once or twice but not routinely.
* 2 x Only because of your wait statistics post.

The survey results are not surprising, especially among readers of my blog.

Wait statistics are the bread-and-butter of performance tuning. SQL Server is keeping track of what resources threads need to wait for, and how long they need to wait. By analyzing which resources (and combinations of resource) are being waited for the most, you can get an idea of where to start digging in further. An example might be that if most of the waits are [*PAGEIOLATCH\_SH*](http://www.sqlskills.com/help/waits/pageiolatch_sh/) waits, and this wasn’t the case in your wait stats baseline, you might look at the I/O subsystem performance using the *sys.dm\_io\_virtual\_file\_stats* DMV (which I blogged about [here](http://www.sqlskills.com/blogs/paul/how-to-examine-io-subsystem-latencies-from-within-sql-server/)).

Last December I wrote a long blog post introducing wait statistics, showing how to use the *sys.dm\_os\_wait\_stats* DMV, giving links to resources, and explaining the most common ones that people see in the field based on data from more than 1800 SQL Servers – see [Wait statistics, or please tell me where it hurts](http://www.sqlskills.com/blogs/paul/wait-statistics-or-please-tell-me-where-it-hurts/).

**sys.dm\_os\_waiting\_tasks**



Other values:

* 6 x Yes, More than once or twice but not routinely.

I’m surprised that these results don’t tie in more closely with the results for *sys.dm\_os\_wait\_stats*, but they’re reasonably close.

The *sys.dm\_os\_waiting\_tasks* DMV shows you what is currently being waited on by everything running on the system.

I created a scenario with 200 clients creating and dropping small temp tables to create tempdb latch contention. Using the DMV, I can see what’s being waited on (I’ve removed the columns describing blocking from the output in this case to make it fit on screen):

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |  |
| --- | --- |
| 1  2 | SELECT \* FROM sys.dm\_os\_waiting\_tasks; GO |

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |
| --- |
| waiting\_task\_address session\_id exec\_context\_id wait\_duration\_ms     wait\_type          resource\_address   resource\_description  -------------------- ---------- --------------- -------------------- ------------------ ------------------ --------------------  0x000000000044E508   2          0               4091305              XE\_DISPATCHER\_WAIT NULL               NULL  0x000000000044E988   9          0               4121252              FSAGENT            NULL               NULL  0x000000000044EBC8   20         0               4121251              BROKER\_TRANSMITTER NULL               NULL  0x000000000044F4C8   63         0               53                   PAGELATCH\_EX       0x0000000088FFEED8 2:1:1139  0x000000000044EE08   64         0               8                    PAGELATCH\_UP       0x0000000080FE8AD8 2:1:1  0x000000000044F288   87         0               0                    PAGELATCH\_UP       0x0000000080FE8AD8 2:1:1  0x000000000044F048   91         0               53                   PAGELATCH\_EX       0x0000000088FFEED8 2:1:1139  0x000000000044F948   92         0               61                   PAGELATCH\_EX       0x0000000088FFEED8 2:1:1139  0x000000000044F708   101        0               10                   PAGELATCH\_EX       0x0000000080FEEC58 2:1:120  0x000000000044FDC8   103        0               37                   PAGELATCH\_UP       0x0000000080FE8AD8 2:1:1  0x000000008744E088   118        0               11                   PAGELATCH\_EX       0x0000000080FEEC58 2:1:120  0x000000008744E2C8   121        0               66                   PAGELATCH\_UP       0x0000000080FE8AD8 2:1:1  0x000000008744E508   122        0               33                   PAGELATCH\_EX       0x0000000080FEEC58 2:1:120  0x000000008744E748   155        0               32                   PAGELATCH\_EX       0x0000000080FEEC58 2:1:120  0x000000008744E988   158        0               27                   PAGELATCH\_EX       0x0000000080FEEC58 2:1:120  0x000000008744EBC8   163        0               34                   PAGELATCH\_EX       0x0000000080FEEC58 2:1:120  0x000000008744EE08   168        0               66                   PAGELATCH\_UP       0x0000000080FE8AD8 2:1:1  0x000000008744F048   179        0               26                   PAGELATCH\_UP       0x0000000080FE8AD8 2:1:1  .  . |

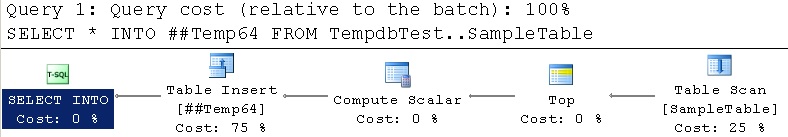
As you can see, the classic tempdb latch contention is showing – page ID (2:1:1) – the first PFS page in tempdb. (See [here for more on tempdb contention](http://www.sqlskills.com/blogs/paul/a-sql-server-dba-myth-a-day-1230-tempdb-should-always-have-one-data-file-per-processor-core/), and [here for more on PFS pages](http://www.sqlskills.com/blogs/paul/inside-the-storage-engine-gam-sgam-pfs-and-other-allocation-maps/).)

My colleague Joe Sack created a script that pulls in data from a bunch of other DMVs to make the *sys.dm\_os\_waiting\_tasks* output more useful, which I’ve modified into the following (note that ‘text’ on one line does not have delimiters because that messes up the code formatting plugin):

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

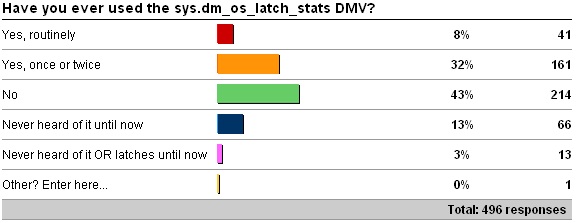
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | SELECT      [owt].[session\_id],      [owt].[exec\_context\_id],      [ot].[scheduler\_id],      [owt].[wait\_duration\_ms],      [owt].[wait\_type],      [owt].[blocking\_session\_id],      [owt].[resource\_description],      CASE [owt].[wait\_type]          WHEN N'CXPACKET' THEN              RIGHT ([owt].[resource\_description],                  CHARINDEX (N'=', REVERSE ([owt].[resource\_description])) - 1)          ELSE NULL      END AS [Node ID],      --[es].[program\_name],     [est].text,      [er].[database\_id],      [eqp].[query\_plan],      [er].[cpu\_time] FROM sys.dm\_os\_waiting\_tasks [owt] INNER JOIN sys.dm\_os\_tasks [ot] ON      [owt].[waiting\_task\_address] = [ot].[task\_address] INNER JOIN sys.dm\_exec\_sessions [es] ON      [owt].[session\_id] = [es].[session\_id] INNER JOIN sys.dm\_exec\_requests [er] ON      [es].[session\_id] = [er].[session\_id] OUTER APPLY sys.dm\_exec\_sql\_text ([er].[sql\_handle]) [est] OUTER APPLY sys.dm\_exec\_query\_plan ([er].[plan\_handle]) [eqp] WHERE      [es].[is\_user\_process] = 1 ORDER BY      [owt].[session\_id],      [owt].[exec\_context\_id]; GO |

There’s too much information in the output to usefully show in this post, but I can see the actual T-SQL statements being run (in this case a lot of *DROP TABLE* and *SELECT \* INTO* of global temp tables) and the XML query plans. Clicking on one of them in SSMS gives me the actual plan – very cool:



This means I can see from the *sys.dm\_os\_wait\_stats* DMV what the prevalent resource waits are, then use the *sys.dm\_os\_waiting\_tasks* DMV to see which queries are waiting for those resources – and then dive in deeper to see why.

**sys.dm\_os\_latch\_stats**



As I suspected most readers have heard of latches, but 75% of respondents haven’t used the DMV or have only used it once or twice.

A latch is a lightweight synchronization mechanism that protects access to read and change in-memory structures – for instance, 8KB page buffers in the buffer pool (latch class = [*BUFFER*](http://www.sqlskills.com/help/latches/buffer/)), or the data structure that represents a database’s data and log files (latch class = [*FGCB\_ADD\_REMOVE*](http://www.sqlskills.com/help/latches/fgcb_add_remove/)). A latch is only held for the duration of the operation, unlike a lock which may be held until a transaction commits. One example of locks and latches – imagine a table where an update query has caused lock escalation so that a table X lock is held on the table. As the query continues updating more records in the table, it won’t acquire any more locks, but any data and index pages that are updated in memory must be EX (exclusive) latched before the update can occur. The latch acts as the synchronization mechanism to prevent two threads updating the page at the same time, or a thread reading the page while another is in the middle of updating it. Another example is if you run a select query using *NOLOCK* – although the query will not acquire SH (share) locks at any level in the table, the threads must acquire SH latches on pages before they can be read – to synchronize with possible concurrent updaters.

If a thread requires a latch it will be moved from *RUNNING* to *SUSPENDED* and put on the waiter list to await notification that the latch has been acquired in the requested mode.

Latch waits correspond to [*LATCH\_XX*](http://www.sqlskills.com/help/waits/latch_ex/) waits in the output from the *sys.dm\_os\_wait\_stats* DMV, so digging into to which latches are accounting for most waits can show where a bottleneck is on the system.

You can reset latch wait statistics just like regular wait statistics using:

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |  |
| --- | --- |
| 1  2 | DBCC SQLPERF('sys.dm\_os\_latch\_stats', CLEAR); GO |

An example set of output from the DMV is:

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |  |
| --- | --- |
| 1  2  3  4 | SELECT \* FROM sys.dm\_os\_latch\_stats WHERE [waiting\_requests\_count] > 0 ORDER BY [wait\_time\_ms] DESC; GO |

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |
| --- |
| latch\_class                       waiting\_requests\_count wait\_time\_ms         max\_wait\_time\_ms  --------------------------------- ---------------------- -------------------- --------------------  BUFFER                            113181121              466697044            1233  ACCESS\_METHODS\_HOBT\_COUNT         66676                  331193               577  ACCESS\_METHODS\_HOBT\_VIRTUAL\_ROOT  15018                  68865                125  LOG\_MANAGER                       130                    5610                 234  FGCB\_ADD\_REMOVE                   299                    5073                 32  TRACE\_CONTROLLER                  1                      0                    0  VERSIONING\_TRANSACTION\_LIST       1                      0                    0  ACCESS\_METHODS\_HOBT\_FACTORY       64                     0                    0 |

You can also aggregate them in the same way as I described in my [big wait stats blog post](http://www.sqlskills.com/blogs/paul/wait-statistics-or-please-tell-me-where-it-hurts/), using code below:

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | WITH [Latches] AS      (SELECT          [latch\_class],          [wait\_time\_ms] / 1000.0 AS [WaitS],          [waiting\_requests\_count] AS [WaitCount],          100.0 \* [wait\_time\_ms] / SUM ([wait\_time\_ms]) OVER() AS [Percentage],          ROW\_NUMBER() OVER(ORDER BY [wait\_time\_ms] DESC) AS [RowNum]      FROM sys.dm\_os\_latch\_stats      WHERE [latch\_class] NOT IN (          N'BUFFER')      --AND [wait\_time\_ms] > 0     ) SELECT      MAX ([W1].[latch\_class]) AS [LatchClass],      CAST (MAX ([W1].[WaitS]) AS DECIMAL(14, 2)) AS [Wait\_S],      MAX ([W1].[WaitCount]) AS [WaitCount],      CAST (MAX ([W1].[Percentage]) AS DECIMAL(14, 2)) AS [Percentage],      CAST ((MAX ([W1].[WaitS]) / MAX ([W1].[WaitCount])) AS DECIMAL (14, 4)) AS [AvgWait\_S] FROM [Latches] AS [W1] INNER JOIN [Latches] AS [W2]      ON [W2].[RowNum] <= [W1].[RowNum] GROUP BY [W1].[RowNum] HAVING SUM ([W2].[Percentage]) - MAX ([W1].[Percentage]) < 95; -- percentage threshold  GO |

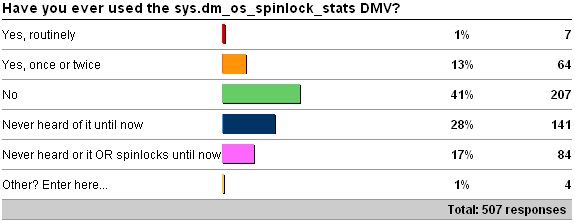
Here’s an example after clearing the latch stats and running the tempdb contention test (I described above) for 30 seconds:

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |
| --- |
| LatchClass                        Wait\_S  WaitCount  Percentage  AvgWait\_S  --------------------------------- ------- ---------- ----------- ----------  ACCESS\_METHODS\_HOBT\_COUNT         7.92    1471       75.41       0.0054  ACCESS\_METHODS\_HOBT\_VIRTUAL\_ROOT  1.38    393        13.15       0.0035  LOG\_MANAGER                       1.20    12         11.44       0.1001 |

Most of the latch classes are undocumented, but I’ll be shedding light on them as I blog more about latch stats.

**sys.dm\_os\_spinlock\_stats**



Other values:

* 1 x Just learned of this at SQLskills training!
* 1 x Learned about it, after you tweeted about it on 3/23 to a co-worker.

This is really cool that more than 40% of respondents have never heard of this DMV or spinlocks – education time!

A spinlock is another lightweight synchronization mechanism used to control access to certain data structures in the engine – used when the time that the spinlock will be held is very short. They are different from latches because a thread waiting for a latch will yield the scheduler and go onto the waiter list whereas a thread waiting to acquire a spinlock will burn some CPU “spinning” to see if it can get the CPU before giving up and backing off (yielding the scheduler) before trying again. This may allow another thread to execute that is holding the spinlock and eventually release it, allowing the system to proceed (yes, a thread can yield the scheduler and move to the waiter list while holding a spinlock!) because another thread can then acquire the spinlock.

It is perfectly normal for spinlock collisions and spins to occur on a busy system, but sometimes a bottleneck can occur on systems with larger numbers of CPUs where collisions are more likely – this can drain CPU resources while many threads are spinning trying to acquire the spinlock.

Running the DMV shows you the list of all spinlocks on the system (all of which are undocumented – but I’ll be working on that going forward) – here is some partial output:

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |  |
| --- | --- |
| 1  2  3 | SELECT \* FROM sys.dm\_os\_spinlock\_stats ORDER BY [spins] DESC; GO |

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |
| --- |
| name               collisions           spins                spins\_per\_collision sleep\_time           backoffs  ------------------ -------------------- -------------------- ------------------- -------------------- -----------  LOCK\_HASH          3629624              4402099957           1212.825            561                  817819  SOS\_CACHESTORE     11992297             3352117666           279.5226            6093                 71948  OPT\_IDX\_MISS\_KEY   63329610             2036811058           32.16207            15830                180845  SOS\_TLIST          9769744              574740437            58.82861            320                  3619  SOS\_SCHEDULER      2137875              107392996            50.23352            557                  7753  MUTEX              676406               83493780             123.4374            340                  3300  LOGCACHE\_ACCESS    2210697              83204315             37.63714            0                    252366  SOS\_RW             264489               70122059             265.1228            14                   799  XDESMGR            346005               61031449             176.3889            216                  3788  SOS\_SUSPEND\_QUEUE  3397384              53752545             15.82174            120                  2384  OPT\_IDX\_STATS      129814               19800952             152.5332            27                   356  BACKUP\_CTX         29730                16784471             564.5635            192                  1645  LOCK\_RESOURCE\_ID   17558                4363116              248.4973            20                   375  SOS\_TASK           206597               1898063              9.187273            16                   171  XVB\_LIST           266112               882691               3.316991            1                    63  .  . |

On 2005 you’ll need to use *DBCC SQLPERF (‘spinlockstats’)* and use *INSERT/EXEC* to get the results into a table. Eric Humphrey ([blog](http://www.erichumphrey.com/" \t "_blank)|[twitter](http://twitter.com/lotsahelp)) put the code together:

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | IF OBJECT\_ID (N'tempdb..#TempSpinlockStats1') IS NOT NULL      DROP TABLE [#TempSpinlockStats1]; GO CREATE TABLE [#TempSpinlockStats1] (      [name]                NVARCHAR(30) NOT NULL,      [collisions]          BIGINT NOT NULL,      [spins]               BIGINT NOT NULL,      [spins\_per\_collision] FLOAT NOT NULL,      [sleep\_time]          BIGINT NOT NULL,      [backoffs]            BIGINT NOT NULL ); INSERT INTO [#TempSpinlockStats1] EXEC ('DBCC SQLPERF(''spinlockstats'')'); GO |

The *LOCK\_HASH* spinlock, for instance, is used by the lock manager to look at one of the hash buckets holding lock resource hashes to tell whether lock can be granted or not.

The sleep\_time is an aggregate of how much time is spent sleeping between spin cycles when a backoff occurs.

I’ve put together some code that will allow you to see what spinlock activity occurs between two times. The code captures the output from the DMV into two temp tables, with whatever time period you want in between (to allow you to run a command), and then shows the difference between the two sets of data. I’ll show an example of running *DBCC CHECKDB*.

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59 | IF EXISTS (SELECT \* FROM [tempdb].[sys].[objects]      WHERE [name] = N'##TempSpinlockStats1')      DROP TABLE [##TempSpinlockStats1];   IF EXISTS (SELECT \* FROM [tempdb].[sys].[objects]      WHERE [name] = N'##TempSpinlockStats2')      DROP TABLE [##TempSpinlockStats2]; GO   -- Baseline  SELECT \* INTO [##TempSpinlockStats1] FROM sys.dm\_os\_spinlock\_stats WHERE [collisions] > 0 ORDER BY [name]; GO   -- Now do something  DBCC CHECKDB (N'SalesDB') WITH NO\_INFOMSGS; GO   -- Capture updated stats  SELECT \* INTO [##TempSpinlockStats2] FROM sys.dm\_os\_spinlock\_stats WHERE [collisions] > 0 ORDER BY [name]; GO   -- Diff them  SELECT      '\*\*\*' AS [New],      [ts2].[name] AS [Spinlock],      [ts2].[collisions] AS [DiffCollisions],      [ts2].[spins] AS [DiffSpins],      [ts2].[spins\_per\_collision] AS [SpinsPerCollision],      [ts2].[sleep\_time] AS [DiffSleepTime],      [ts2].[backoffs] AS [DiffBackoffs] FROM [##TempSpinlockStats2] [ts2] LEFT OUTER JOIN [##TempSpinlockStats1] [ts1]      ON [ts2].[name] = [ts1].[name] WHERE [ts1].[name] IS NULL UNION SELECT      '' AS [New],      [ts2].[name] AS [Spinlock],      [ts2].[collisions] - [ts1].[collisions] AS [DiffCollisions],      [ts2].[spins] - [ts1].[spins] AS [DiffSpins],      CASE ([ts2].[spins] - [ts1].[spins]) WHEN 0 THEN 0          ELSE ([ts2].[spins] - [ts1].[spins]) /              ([ts2].[collisions] - [ts1].[collisions]) END              AS [SpinsPerCollision],      [ts2].[sleep\_time] - [ts1].[sleep\_time] AS [DiffSleepTime],      [ts2].[backoffs] - [ts1].[backoffs] AS [DiffBackoffs] FROM [##TempSpinlockStats2] [ts2] LEFT OUTER JOIN [##TempSpinlockStats1] [ts1]      ON [ts2].[name] = [ts1].[name] WHERE [ts1].[name] IS NOT NULL      AND [ts2].[collisions] - [ts1].[collisions] > 0 ORDER BY [New] DESC, [Spinlock] ASC; GO |

And the output is as follows:

[?](http://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/)

|  |
| --- |
| New  Spinlock           DiffCollisions       DiffSpins            SpinsPerCollision DiffSleepTime        DiffBackoffs  ---- ------------------ -------------------- -------------------- ----------------- -------------------- ------------  \*\*\*  DBCC\_CHECK         4                    24                   6                 0                    0  \*\*\*  DIAG\_MANAGER       1                    0                    0                 0                    0  \*\*\*  FCB\_REPLICA\_SYNC   10                   16147                1614.7            0                    0  \*\*\*  LSID               9                    0                    0                 0                    0  \*\*\*  QUERYEXEC          5                    0                    0                 0                    0  \*\*\*  X\_PACKET\_LIST      2                    0                    0                 0                    0  \*\*\*  X\_PORT             8                    227                  28.375            0                    0  \*\*\*  XACT\_WORKSPACE     11                   0                    0                 0                    0  \*\*\*  XID\_ARRAY          7                    0                    0                 0                    0       BUF\_FREE\_LIST      2                    0                    0                 0                    0       HOBT\_HASH          2                    1                    0                 0                    0       LOCK\_HASH          3                    1818                 606               0                    0       SOS\_RW             2                    500                  250               0                    0       SOS\_SCHEDULER      5                    6                    1                 0                    0       SOS\_SUSPEND\_QUEUE  11                   39                   3                 0                    0       SOS\_TASK           1                    0                    0                 0                    0       SOS\_TLIST          1                    0                    0                 0                    0 |

You can see here which spinlocks were acquired to run the *DBCC CHECKDB* commands – those marked with \*\*\* did not appear in the ‘before’ set of spinlock stats. More on all of these in future posts.

You can also investigate spinlocks using extended events – again, more on that in future.

**Summary**

It’s possible to dive really deeply into what’s happening inside SQL Server using these four DMVs. Spinlocks in particular – what each means, what each controls and what contention on each them means (plus what you can do about it) – involve a lot of knowledge of what’s going on inside the engine, and I’m planning to spread some of that knowledge going forward – there’s an enormous amount of information to be published about latches and spinlocks.

Hope you’ll join me to learn about these – let me know if you find this stuff interesting and useful.

Thanks!

**Related Posts**

* [New whitepapers on latches and spinlocks published](https://www.sqlskills.com/blogs/paul/new-whitepapers-on-latches-and-spinlocks-published/)
* [Most common latch classes and what they mean](https://www.sqlskills.com/blogs/paul/most-common-latch-classes-and-what-they-mean/)
* [Survey: have you ever used these DMVs?](https://www.sqlskills.com/blogs/paul/survey-have-you-ever-used-these-dmvs/)
* [Survey: most prevalent latch waits (code to run)](https://www.sqlskills.com/blogs/paul/survey-most-prevalent-latch-waits-code-to-run/)
* [Announcing the comprehensive SQL Server Wait Types and Latch Classes Library](https://www.sqlskills.com/blogs/paul/announcing-the-comprehensive-sql-server-wait-types-and-latch-classes-library/)

Posted in: [Example Scripts](https://www.sqlskills.com/blogs/paul/category/example-scripts/), [Latches](https://www.sqlskills.com/blogs/paul/category/latches/), [Performance Tuning](https://www.sqlskills.com/blogs/paul/category/performance-tuning/), [Spinlocks](https://www.sqlskills.com/blogs/paul/category/spinlocks/), [Surveys](https://www.sqlskills.com/blogs/paul/category/surveys/), [Wait Stats](https://www.sqlskills.com/blogs/paul/category/wait-stats/)

[17 Comments](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comments)

**17 Responses to *Advanced SQL Server performance tuning***

1. http://1.gravatar.com/avatar/?s=44&d=mm&r=g*Thiago Dantas* says:

[April 13, 2011 at 12:40 pm](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1752)

Waiting for your book on the subject! :)

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1752#respond)

1. http://0.gravatar.com/avatar/?s=44&d=mm&r=g[*Amit Banerjee*](http://troubleshootingsql.com/) says:

[April 13, 2011 at 11:10 am](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1751)

SQL Nexus provides a Spinlock Stats Summary report which can be quite useful when troubleshooting performance degradation issues caused by high memory consumption for TokenPerm cache store.

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1751#respond)

1. http://0.gravatar.com/avatar/?s=44&d=mm&r=g*Aditya* says:

[April 13, 2011 at 9:33 pm](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1753)

Paul,

Really didint heard of Spinlocks until…Thanks for shedding light on Spinlocks and latches. Though Im still confused on Spinlocks part eagerly waiting for the future posts on the same….

Thank you very much

Aditya

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1753#respond)

1. http://1.gravatar.com/avatar/?s=44&d=mm&r=g*Michael Barrett* says:

[April 14, 2011 at 12:42 am](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1754)

Hi Paul

Good stuff… I’ve been looking for more documentation on the spinlocks in particular for quite some time. I am very much looking forward to your next posts on this!

/Michael

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1754#respond)

1. http://0.gravatar.com/avatar/?s=44&d=mm&r=g*amosquera* says:

[April 14, 2011 at 3:37 am](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1755)

Hi Paul,  
I’m found this very useful and interesting and "waiting" for more …

Thanks

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1755#respond)

1. http://2.gravatar.com/avatar/?s=44&d=mm&r=g*Jeroen Mostert* says:

[April 14, 2011 at 7:12 am](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1756)

The problem I see with latch and spinlock waits is that I have no idea how to see if 1) they are a problem for server performance (sure, there may be a lot, but then we have servers to do work after all) and 2) if they are, how you are supposed to reduce them without the obvious answer of performing less work ("don’t do that, then"). Performance indicators that aren’t actionable are of no use.

I/O problems are easy to identify and likewise easy to remedy, if you know your stuff about query execution. Latches and spinlocks, not so much — you probably need to know \*even more\* about query execution. So I, too, am eagerly looking forward to future posts on the subject.

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1756#respond)

1. http://0.gravatar.com/avatar/?s=44&d=mm&r=g*Dan Holmes* says:

[April 14, 2011 at 11:12 am](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1757)

I changed

SELECT  
wait\_type,  
waiting\_tasks\_count,  
wait\_time\_ms,  
wait\_time\_ms/ISNULL(NULLIF(waiting\_tasks\_count,0),1) AS Avg\_wait\_time,  
wait\_time\_ms/(SELECT CAST(SUM(wait\_time\_ms) AS numeric) FROM sys.dm\_os\_wait\_stats) AS percent\_wait\_of\_total,  
max\_wait\_time\_ms  
FROM sys.dm\_os\_wait\_stats  
ORDER BY Avg\_wait\_time DESC

to

SELECT  
wait\_type,  
waiting\_tasks\_count,  
wait\_time\_ms,  
wait\_time\_ms/CAST(ISNULL(NULLIF(waiting\_tasks\_count,0),1) AS NUMERIC(18,6)) AS Avg\_wait\_time,  
wait\_time\_ms/(SELECT CAST(SUM(wait\_time\_ms) AS numeric) FROM sys.dm\_os\_wait\_stats) AS percent\_wait\_of\_total,  
max\_wait\_time\_ms  
FROM sys.dm\_os\_wait\_stats  
ORDER BY Avg\_wait\_time DESC

so the avg wait time could be less than zero

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1757#respond)

1. [*SOS\_SCHEDULER\_YIELD waits and the LOCK\_HASH spinlock - Paul S. Randal*](http://www.sqlskills.com/blogs/paul/sos_scheduler_yield-waits-and-the-lock_hash-spinlock/) says:

[January 8, 2013 at 12:41 pm](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-4461)

[…] Advanced performance troubleshooting: waits, latches, spinlocks […]

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=4461#respond)

1. [*Benchmarking: Multiple data files on SSDs (plus the latest Fusion-io driver) - Paul S. Randal*](http://www.sqlskills.com/blogs/paul/benchmarking-multiple-data-files-on-ssds-plus-the-latest-fusion-io-driver/) says:

[January 15, 2013 at 1:37 pm](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-4980)

[…] went back and ran a single file test and used the sys.dm\_os\_waiting\_tasks DMV (see this blog post) to see what the various threads are waiting for. Here's some example […]

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=4980#respond)

1. http://1.gravatar.com/avatar/4d364328e398836b164a62bb78248ad8?s=44&d=mm&r=g[*Daniel Adeniji*](http://danieladeniji.wordpress.com) says:

[September 16, 2013 at 1:34 am](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-13411)

Paul Randal & Joe Sacks:

I know this post in a bit dated, but here I am trying to steal the SQL pertaining to “sys.dm\_os\_waiting\_tasks”.

And, as always, when one is stealing there is always an anchor that tells you ‘re stealing, and you got to own up to it.

The bait that I choked on this time is the line that reads “[est].1”. The sure-tell sign that something is amiss is the error returned (Incorrect syntax near ‘.1’.)

I have commented out that line for now, but I might be missing something that is just as useful as the other columns referenced.

Once again thanks for gracefully planting so much.

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=13411#respond)

* + http://0.gravatar.com/avatar/630de809a112f9ee05f86eeb8d7848b7?s=44&d=mm&r=g[*Paul Randal*](http://www.sqlskills.com/blogs/paul) says:

[September 16, 2013 at 5:46 am](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-13414)

Yeah – it was the code formatting plugin, where is a command to it. I removed the delimiters and it’s fine now.

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=13414#respond)

1. http://1.gravatar.com/avatar/4d364328e398836b164a62bb78248ad8?s=44&d=mm&r=g[*Daniel Adeniji*](http://danieladeniji.wordpress.com) says:

[September 16, 2013 at 1:41 am](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-13413)

I just went back and looked at Joe’s original post @ <http://www.digitalconcourse.com/dropzone/MSCOMM/PASSMN/PASSEVT20090616/Joe%20Sack%20Performance%20Troubleshooting%20with%20Wait%20Stats.pdf>.

And, I am a bit convinced that Joe’s erstwhile line read:

t.text

And, so [est].1 should probably read [est].text

Thanks,  
Daniel

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=13413#respond)

1. [*SQL Server Wait Statistics: Tell me where it hurts - Paul Randal*](http://www.sqlskills.com/blogs/paul/wait-statistics-or-please-tell-me-where-it-hurts/) says:

[March 19, 2014 at 2:45 pm](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-19134)

[…] Advanced SQL Server performance tuning […]

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=19134#respond)

1. http://0.gravatar.com/avatar/32142f79a7f63a3ec718047648ed7eae?s=44&d=mm&r=g*Mike* says:

[July 4, 2014 at 1:13 pm](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-65299)

I blog a lot on LinkedIn. I’m always amazed that people aren’t using pluralsight.com. Paul has several courses on here that are stellar. His wait stats course is the only one like out there. He walks you through latches and spinlocks. You get to see the code and listen to insights. So, head on over to PS and watch his course and the many others by the sqlskills group. I don’t know Paul, never met him, not being paid for this… etc… but you won’t find more advanced tutorials anywhere.

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=65299#respond)

1. [*Advanced SQL Server Performance Tuning | DiggerSite*](http://www.digger.dp.ua/advanced-sql-server-performance-tuning/) says:

[October 19, 2015 at 10:20 pm](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1013345)

[…] Источник: Advanced SQL Server Performance Tuning […]

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1013345#respond)

1. [*Identifying queries with SOS\_SCHEDULER\_YIELD waits - Paul S. Randal*](http://www.sqlskills.com/blogs/paul/identifying-queries-with-sos_scheduler_yield-waits/) says:

[May 12, 2016 at 8:16 am](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1033688)

[…] problem is that they’re not a real wait type, so you can’t use my script to look at sys.dm\_os\_waiting\_tasks and get the query plans of threads incurring that wait type, […]

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1033688#respond)

1. [*Survey: most prevalent latch waits (code to run) - Paul S. Randal*](http://www.sqlskills.com/blogs/paul/survey-most-prevalent-latch-waits-code-to-run/) says:

[May 23, 2016 at 5:02 pm](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/#comment-1035162)

[…] started blogging about latches and some of the deeper parts of SQL Server internals last year (see Advanced performance troubleshooting: waits, latches, spinlocks) and now I’d like to pick up that thread (no scheduling pun intended :-)) and blog some more […]

[Reply](https://www.sqlskills.com/blogs/paul/advanced-performance-troubleshooting-waits-latches-spinlocks/?replytocom=1035162#respond)