# SQL Server Trace Flags - Complete list

**By** [**Konstantin Taranov**](http://www.sqlservercentral.com/Authors/Articles/Konstantin_Taranov/1532632/)**, 2019/02/14 (first published: 2017/02/23)**

## Trace Flags List

Summary: **593 trace flags**

Updated on Jan 24, 2019.

#### Trace Flag: -1

Function: Sets trace flags for all client connections, rather than for a single client connection. Because trace flags set using the -T command-line option automatically apply to all connections, this trace flag is used only when setting trace flags using [DBCC TRACEON](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-transact-sql) and [DBCC TRACEOFF](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceoff-transact-sql).  
Link: <http://www.sql-server-performance.com/2002/traceflags/>

#### Trace Flag: 101

Function: Verbose Merge Replication logging output for troubleshooting Merger repl performance  
Link: <https://support.microsoft.com/help/2892633>  
Scope: global only

#### Trace Flag: 102

Function: Verbose Merge Replication logging to msmerge\_history table for troubleshooting Merger repl performance  
Link: <https://support.microsoft.com/help/2892633>  
Scope: global only

#### Trace Flag: 105

**Undocumented trace flag**  
Function: Join more than 16 tables in SQL server 6.5  
Link: [SQL Server 6.5: Some Useful Trace Flag](http://www.databasejournal.com/features/mssql/article.php/1443351/SQL-Server-65-Some-Useful-Trace-Flags.htm)

#### Trace Flag: 106

Function: If you are using Web Synchronization, you can start Replmerg.exe and pass the -T 106 option to use trace flag 106. This enables you to see the messages that are sent to and from the Publisher. The agent writes the client's input messages to a file that is named ExchangeID(guid).IN.XML, and writes the output messages to a file that is named ExchangeID(guid).OUT.XML. (In these file names, guid is the GUID of the Exchange Server session.) These files are created in the directory from which Replmerg.exe was invoked. For security, you should delete these files after you are finished.  
Link: <http://technet.microsoft.com/en-us/library/ms151872(v=sql.105).aspx>

#### Trace Flag: 107

**Undocumented trace flag**  
Function: SQL 6.5/7/8 – Interprets numbers with a decimal point as float instead of decimal  
Link: None

#### Trace Flag: 110

**Undocumented trace flag**  
Function: SQL 6.5 – Turns off ANSI select characteristics  
Link: None

#### Trace Flag: 120

**Undocumented trace flag**  
Function: FIX: Error message when you schedule a Replication Merge Agent job to run after you install SQL Server 2000 Service Pack 4: "The process could not enumerate changes at the 'Subscriber'"  
Link: None

#### Trace Flag: 139

Function: Forces correct conversion semantics in the scope of DBCC check commands like [DBCC CHECKDB](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-checkdb-transact-sql), [DBCC CHECKTABLE](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-checktable-transact-sql) and [DBCC CHECKCONSTRAINTS](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-checkconstraints-transact-sql), when analyzing the improved precision and conversion logic introduced with compatibility level 130 for specific data types, on a database that has a lower compatibility level. **Note: This trace flag applies to SQL Server 2016 RTM CU3, SQL Server 2016 SP1 and higher builds.**  
**WARNING: Trace flag 139 is not meant to be enabled continuously in a production environment, and should be used for the sole purpose of performing database validation checks described in this Microsoft Support article. It should be immediately disabled after validation checks are completed.**  
Link: <https://support.microsoft.com/help/4010261>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 144

Function: Force server side bucketization. For legacy applications where change to client side code is not an option and when the application has queries that are improperly parameterized, this trace flag forces server side bucketization.  
Link: <http://blogs.msdn.microsoft.com/sqlprogrammability/2007/01/13/6-0-best-programming-practices>

#### Trace Flag: 146

**Undocumented trace flag**  
Function: Consider using when replaying against SQL 8.0, to avoid an attempt to set an encrypted connection.  
Link: None

#### Trace Flag: 166

**Undocumented trace flag**  
Function: Unclear. Observable effect was to change the identifier for act1008 to act1009 in a query plan.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 168

Function: Bugfix in ORDER BY. This hotfix introduces trace flag 168. After you apply this hotfix, you must enable trace flag 168. Trace flag 168 must be set before the database is migrated to SQL Server 2005. If trace flag 168 is set after the database is migrated, the query result will remain unsorted.  
Link: <https://support.microsoft.com/help/926292>

#### Trace Flag: 174

Function: Increases the SQL Server Database Engine plan cache bucket count from 40,009 to 160,001 on 64-bit systems.  
**Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: <https://support.microsoft.com/help/3026083/fix-sos-cachestore-spinlock-contention-on-ad-hoc-sql-server-plan-cache>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 176

Function: Enables a fix to address errors when rebuilding partitions online for tables that contain a computed partitioning column.  
Link: <https://support.microsoft.com/help/3213683/fix-unable-to-rebuild-the-partition-online-for-a-table-that-contains-a>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 204

Function: SQL 6.5 – Backward compatibility switch that enables non-ansi standard behavior. E.g. previously SQL server ignored trailing blanks in the like statement and allowed queries that contained aggregated functions to have items in the group by clause that were not in the select list.  
Link: <https://support.microsoft.com/help/153096/fix-sql-server-6.5-service-pack-1-fixlist>

#### Trace Flag: 205

Function: Reports to the error log when a statistics-dependent stored procedure is being recompiled as a result of auto-update statistics.  
Link: <https://support.microsoft.com/help/195565>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 206

**Undocumented trace flag**  
Function: SQL 6.5 – Provides backward compatibility for the set user statement. KB 160732  
Link: None

#### Trace Flag: 208

**Undocumented trace flag**  
Function: SET QUOTED IDENTIFIER ON  
Link: None

#### Trace Flag: 210

Function: SQL 9 – Error when you run a query against a view: "An error occurred while executing batch"  
Link: <https://support.microsoft.com/help/945892>

#### Trace Flag: 212

**Undocumented trace flag**  
Function: SQL 9 – Query may run much slower when compared to SQL 8 when you use a cursor to run the query  
Link: None

#### Trace Flag: 220

**Undocumented trace flag**  
Function: “FIX: Error Message: "Insufficient key column information for updating" Occurs in SQL Server 2000 SP3”  
Link: None

#### Trace Flag: 221

**Undocumented trace flag**  
Function: “FIX: The query runs slower than you expected when you try to parse a query in SQL Server 2000”  
Link: None

#### Trace Flag: 222

**Undocumented trace flag**  
Function: “FIX: Each query takes a long time to compile when you execute a single query or when you execute multiple concurrent queries in SQL Server 2000”  
Link: None

#### Trace Flag: 237

**Undocumented trace flag**  
Function: Tells SQL Server to use correlated sub-queries in Non-ANSI standard backward compatibility mode  
Link: None

#### Trace Flag: 242

**Undocumented trace flag**  
Function: Provides backward compatibility for correlated subqueries where non-ANSI-standard results are desired  
Link: None

#### Trace Flag: 243

**Undocumented trace flag**  
Function: Provides backward compatibility for nullability behavior. When set, SQL Server has the same nullability violation behavior as that of a ver 4.2: Processing of the entire batch is terminated if the nullability error (inserting NULL into a NOT NULL field) can be detected at compile time; Processing of offending row is skipped, but the command continues if the nullability violation is detected at run time.Behavior of SQL Server is now more consistent because nullability checks are made at run time and a nullability violation results in the command terminating and the batch or transaction process continuing.  
Link: None

#### Trace Flag: 244

**Undocumented trace flag**  
Function: Disables checking for allowed interim constraint violations. By default, SQL Server checks for and allows interim constraint violations. An interim constraint violation is caused by a change that removes the violation such that the constraint is met, all within a single statement and transaction. SQL Server checks for interim constraint violations for self-referencing DELETE statements, INSERT, and multi-row UPDATE statements. This checking requires more work tables. With this trace flag you can disallow interim constraint violations, thus requiring fewer work tables.  
Link: None

#### Trace Flag: 246

**Undocumented trace flag**  
Function: Derived or NULL columns must be explicitly named in a select…INTO or create view statement when not done they raise an error. This flag avoids that.  
Link: None

#### Trace Flag: 253

**Undocumented trace flag**  
Function: Prevents ad-hoc query plans to stay in cache  
Link: <http://www.sqlservercentral.com/Forums/Topic837613-146-1.aspx>

#### Trace Flag: 257

**Undocumented trace flag**  
Function: Will invoke a print algorithm on the XML output before returning it to make the XML result more readable  
Link: None

#### Trace Flag: 260

Function: Prints versioning information about extended stored procedure dynamic-link libraries (DLLs). When SQL Server is started with the trace flag -T260 or if a user with system administrator privileges runs DBCC TRACEON (260), and if the extended stored procedure DLL does not support \_\_GetXpVersion(), a warning message (Error 8131: Extended stored procedure DLL '%' does not export \_\_GetXpVersion().) is printed to the error log. (Note that \_\_GetXpVersion() begins with **two underscores**.)  
Link: <https://docs.microsoft.com/en-us/sql/relational-databases/extended-stored-procedures-programming/creating-extended-stored-procedures>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 262

**Undocumented trace flag**  
Function: SQL 7 – Trailing spaces are no longer truncated from literal strings in CASE statements  
Link: None

#### Trace Flag: 272

**Note: Recommended for SQL Server 2012**  
Function: Disabling the identity cache. It prevents identity gap after restarting SQL Server 2012 instance, critical for columns with identity and tinyint and smallint data types.  
Link: <http://www.big.info/2013/01/how-to-solve-sql-server-2012-identity.html>  
Link: [https://web.archive.org/web/20160822054721/https://connect.microsoft.com/SQLServer/feedback/details/739013/failover-or-restart-results-in-reseed-of-identity](https://web.archive.org/web/20160822054721/https:/connect.microsoft.com/SQLServer/feedback/details/739013/failover-or-restart-results-in-reseed-of-identity)  
Link: <https://dbafromthecold.com/2017/05/24/disabling-the-identity-cache-in-sql-server-2017/>  
Link: [Demo](https://github.com/ktaranov/sqlserver-kit/blob/master/Errors/Identity_gap_sql_server_2012.sql)  
Link: <https://stackoverflow.com/q/14146148/2298061>  
Scope: global only

#### Trace Flag: 274

**Undocumented trace flag**  
Function: “FIX: Error message when you insert a new row into a view in SQL Server 2005: Cannot insert explicit value for identity column in table when IDENTITY\_INSERT is set to OFF”  
Link: None

#### Trace Flag: 302

**Undocumented trace flag**  
Function: Output Index Selection info  
Link: [SQL Server 6.5: Some Useful Trace Flag](http://www.databasejournal.com/features/mssql/article.php/1443351/SQL-Server-65-Some-Useful-Trace-Flags.htm)

#### Trace Flag: 304

**Undocumented trace flag**  
Function: Changed the reported CachedPlanSize.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 310

**Undocumented trace flag**  
Function: Outputs info about actual join order  
Link: [SQL Server 6.5: Some Useful Trace Flag](http://www.databasejournal.com/features/mssql/article.php/1443351/SQL-Server-65-Some-Useful-Trace-Flags.htm)

#### Trace Flag: 320

**Undocumented trace flag**  
Function: Disables join-order heuristics used in ANSI joins. To see join-order heuristics use flag 310. SQL Server uses join-order heuristics to reduce of permutations when using the best join order.  
Link: None

#### Trace Flag: 323

Function: Outputs detailed info about updates  
Link: [SQL Server 6.5: Some Useful Trace Flag](http://www.databasejournal.com/features/mssql/article.php/1443351/SQL-Server-65-Some-Useful-Trace-Flags.htm)  
Link: <https://support.microsoft.com/help/153096/fix-sql-server-6.5-service-pack-1-fixlist>

#### Trace Flag: 325

**Undocumented trace flag**  
Function: Prints information about the cost of using a non-clustered index or a sort to process an ORDER BY clause  
Link: None

#### Trace Flag: 326

**Undocumented trace flag**  
Function: Prints information about estimated & actual costs of sorts. Instructs server to use arithmetic averaging when calculating density instead of a geometric weighted average when updating statistics. Useful for building better stats when an index has skew on the leading column. Use only for updating the stats of a table/index with known skewed data.  
Link: None

#### Trace Flag: 330

**Undocumented trace flag**  
Function: Enables full output when using the SET SHOWPLAN\_ALL option, which gives detailed information about joins  
Link: None

#### Trace Flag: 342

**Undocumented trace flag**  
Function: Disables the costing of pseudo-merge joins, thus significantly reducing time spent on the parse for certain types of large, multi-table joins. One can also use SET FORCEPLAN ON to disable the costing of pseudo-merge joins because the query is forced to use the order specified in the FROM clause.  
Link: None

#### Trace Flag: 345

**Undocumented trace flag**  
Function: Changes join order selection logic in SQL Server 6.5  
Link: [SQL Server 6.5: Some Useful Trace Flag](http://www.databasejournal.com/features/mssql/article.php/1443351/SQL-Server-65-Some-Useful-Trace-Flags.htm)

#### Trace Flag: 445

**Undocumented trace flag**  
Function: Prints ”compile issued” message in the errorlog for each compiled statement, when used together with 3605  
Link: None

#### Trace Flag: 460

Function: Replace error message [8152](https://docs.microsoft.com/en-us/sql/relational-databases/errors-events/database-engine-events-and-errors?view=sql-server-2017#errors-8000-to-8999) with [2628](https://docs.microsoft.com/en-us/sql/relational-databases/errors-events/database-engine-events-and-errors?view=sql-server-2017#errors-2000-to-2999) (String or binary data would be truncated. The statement has been terminated.). Description for [2628](https://docs.microsoft.com/en-us/sql/relational-databases/errors-events/database-engine-events-and-errors?view=sql-server-2017#errors-2000-to-2999) mesage has useful information - which column had the truncation and which row.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://www.procuresql.com/blog/2018/09/26/string-or-binary-data-get-truncated/>  
Link: <https://feedback.azure.com/forums/908035-sql-server/suggestions/32908417-binary-or-string-data-would-be-truncated-error>  
Link: <https://blogs.msdn.microsoft.com/sql_server_team/string-or-binary-data-would-be-truncated-replacing-the-infamous-error-8152/>  
Link: <https://support.microsoft.com/help/4468101>  
Scope: global or session  
SQL Server Version: 2019, >= 2017 CU12  
Demo: <https://github.com/ktaranov/sqlserver-kit/blob/master/Scripts/Trace_Flag/Trace_Flag_460.sql>

#### Trace Flag: 506

**Undocumented trace flag**  
Function: Enforces SQL-92 standards regarding null values for comparisons between variables and parameters. Any comparison of variables and parameters that contain a NULL always results in a NULL.  
Link: None

#### Trace Flag: 610

Function: Controls minimally logged inserts into indexed tables.  
Link: <http://msdn.microsoft.com/en-us/library/dd425070%28v=SQL.100%29.aspx>  
Link: <https://www.pythian.com/blog/minimally-logged-operations-data-loads/>  
Link: <https://msdn.microsoft.com/library/dd425070.aspx>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://orderbyselectnull.com/2017/07/10/trace-flag-610-and-sql-server-2016/>  
Scope: global or session

#### Trace Flag: 611

**Undocumented trace flag**  
Function: SQL 9 – When turned on, each lock escalation is recorded in the error log along with the SQL Server handle number. Aaron confirmed this still works in SQL 2014. Outputs info of the form: "Escalated locks - Reason: LOCK\_THRESHOLD, Mode: S, Granularity: TABLE, Table: 222623836, HoBt: 150:256, HoBt Lock Count: 6248, Escalated Lock Count: 6249, Line Number: 1, Start Offset: 0, SQL Statement: select count(\*) from dbo.BigTable"  
Link: None

#### Trace Flag: 617

Function: SQL 9 – When turned on, each lock escalation is recorded in the error log along with the SQL Server handle number. As long as there are no SCH\_M lock requests waiting in the ‘lock wait list’, the ‘lock wait list’ will be bypassed by statements issued in uncommitted read transaction isolation level. If there is a SCH\_M lock request in the ‘lock wait list’, a query in uncommitted read transaction isolation level will not bypass the ‘lock wait list’, but the SCH\_S lock request will go into the ‘lock wait list’. In order behind the SCH\_M lock waiting in the same list. As a result the grant of the SCH\_S request for such a query is dependent on the grant and release of the SCH\_M lock request entering the ‘lock wait list’ earlier.  
Link: <https://blogs.msdn.microsoft.com/saponsqlserver/2014/01/17/new-functionality-in-sql-server-2014-part-3-low-priority-wait/>

#### Trace Flag: 634

Function: Disables the background columnstore compression task. SQL Server periodically runs the Tuple Mover background task that compresses columnstore index rowgroups with uncompressed data, one such rowgroup at a time. Columnstore compression improves query performance but also consumes system resources. You can control the timing of columnstore compression manually, by disabling the background compression task with trace flag 634, and then explicitly invoking ALTER INDEX REORGANIZE or ALTER INDEX REBUILD at the time of your choice.  
Link: [Niko Neugebauer Columnstore Indexes – part 35](http://www.nikoport.com/2014/07/24/clustered-columnstore-indexes-part-35-trace-flags-query-optimiser-rules/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlservergeeks.com/trace-flag-634-disable-background-columnstore-compression/>  
Scope: global only

#### Trace Flag: 646

Function: Serves for getting detailed information on which Columnstore were eliminated by the Query Optimiser right into the error log.  
Link: [Niko Neugebauer Columnstore Indexes – part 35](http://www.nikoport.com/2014/07/24/clustered-columnstore-indexes-part-35-trace-flags-query-optimiser-rules/)  
Link: <http://www.sqlskills.com/blogs/joe/exploring-columnstore-index-metadata-segment-distribution-and-elimination-behaviors>

#### Trace Flag: 647

Function: Avoids a new-in-SQL 2012 data check (done when adding a column to a table) that can cause ALTER TABLE... ADD operations to take a very long time. The KB has a useful query for determining the row size for a table.  
Link: <https://support.microsoft.com/help/2986423/fix-it-takes-a-long-time-to-add-new-columns-to-a-table-when-the-row-size-exceeds-the-maximum-allowed-size>

#### Trace Flag: 652

Function: Disable page pre-fetching scans. If you turn on trace flag 652, SQL Server no longer brings database pages into the buffer pool before these database pages are consumed by the scans. If you turn on trace flag 652, queries that benefit from the page pre-fetching feature exhibit low performance.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 653

**Undocumented trace flag**  
Function: Disables read ahead for the current connection  
Link: None

#### Trace Flag: 661

Function: Disables the ghost record removal process. A ghost record is the result of a delete operation. When you delete a record, the deleted record is kept as a ghost record. Later, the deleted record is purged by the ghost record removal process. When you disable this process, the deleted record is not purged. Therefore, the space that the deleted record consumes is not freed. This behavior affects space consumption and the performance of scan operations.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 662

**Undocumented trace flag**  
Function: Prints detailed information about the work done by the ghost cleanup task when it runs next. Use TF [3605](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#3605) to see the output in the errorlog  
Link: <http://blogs.msdn.com/b/sqljourney/archive/2012/07/28/an-in-depth-look-at-ghost-records-in-sql-server.aspx>

#### Trace Flag: 669

Function: “...prevents user queries from queuing requests to the ghost cleanup process”. This flag is a workaround for stack dumps occurring right after SQL Server startup, where user queries (that queue pages for ghost cleanup) were running so quickly after SQL startup that they were queuing pages before the ghost cleanup process had actually initialized.  
Link: <https://support.microsoft.com/help/3027860/error-17066-or-17310-during-sql-server-startup>

#### Trace Flag: 683

**Undocumented trace flag**  
Function: According to the KB, used to workaround a bug in SQL 2000 SP3 by reverting to pre-SP3 parallel-scan behavior in parallel queries. Database-Wiki.com: “Disallow row counter and column mod counters to be partitioned”  
Link: None

#### Trace Flag: 692

Function: Disables fast inserts while bulk loading data into heap or clustered index. Starting SQL Server 2016, fast inserts is enabled by default leveraging minimal logging when database is in simple or bulk logged recovery model to optimize insert performance for records inserted into new pages. With fast inserts, each bulk load batch acquires new extent(s) bypassing the allocation lookup for existing extent with available free space to optimize insert performance. With fast inserts, bulk loads with small batch sizes can lead to increased unused space consumed by objects hence it is recommended to use large batch size for each batch to fill the extent completely. If increasing batch size is not feasible, this trace flag can help reduce unused space reserved at the expense of performance.  
**Note: This trace flag applies to SQL Server 2016 RTM and higher builds.** Link: <https://blogs.msdn.microsoft.com/sql_server_team/sql-server-2016-minimal-logging-and-impact-of-the-batchsize-in-bulk-load-operations/>  
Scope: global or session

#### Trace Flag: 698

**Undocumented trace flag**  
Function: SQL 9 – Performance of INSERT operations against a table with an identity column may be slow when compared to SQL 8  
Link: None

#### Trace Flag: 699

**Undocumented trace flag**  
Function: Turn off transaction logging for the entire SQL dataserver  
Link: None

#### Trace Flag: 670, 671

**Undocumented trace flag**  
Function: Disables deferred deallocation. But note Paul White’s comment on the post! The flag # may actuall by 671.  
Link: [Controlling SQL Server memory dumps](https://blogs.msdn.microsoft.com/psssql/2009/11/17/how-it-works-controlling-sql-server-memory-dumps)

#### Trace Flag: 715

Function: Enables table lock for bulk load operations into a heap with no non-clustered indexes. When this trace flag is enabled, bulk load operations acquire bulk update (BU) locks when bulk copying data into a table. Bulk update (BU) locks allow multiple threads to bulk load data concurrently into the same table, while preventing other processes that are not bulk loading data from accessing the table. The behavior is similar to when the user explicitly specifies TABLOCK hint while performing bulk load, or when the sp\_tableoption table lock on bulk load is enabled for a given table. However, when this trace flag is enabled, this behavior becomes default without any query or database changes.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 806

Function: enables DBCC audit checks to be performed on pages to test for logical consistency problems. These checks try to detect when a read operation from a disk does not experience any errors but the read operation returns data that is not valid. Pages will be audited every time that they are read from disk. Page auditing can affect performance and should only be used in systems where data stability is in question.  
Link: <http://technet.microsoft.com/en-au/library/cc917726.aspx>  
Link: <http://www.sqlskills.com/blogs/paul/how-to-tell-if-the-io-subsystem-is-causing-corruptions>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Link: <https://technet.microsoft.com/en-au/library/cc917726.aspx>  
Scope: ?

#### Trace Flag: 809

**Undocumented trace flag**  
Function: SQL 8 – Limits the amount of Lazy write activity  
Link: None

#### Trace Flag: 815

Function: SQL 8/9 – Enables latch enforcement. SQL Server 8 (with service pack 4) and SQL Server 9 can perform latch enforcement for data pages found in the buffer pool cache. Latch enforcement changes the virtual memory protection state while database page status changes from "clean" to "dirty" ("dirty" means modified through INSERT, UPDATE or DELETE operation). If an attempt is made to modify a data page while latch enforcement is set, it causes an exception and creates a mini-dump in SQL Server installation's LOG directory. Microsoft support can examine the contents of such mini-dump to determine the cause of the exception. In order to modify the data page the connection must first acquire a modification latch. Once the data modification latch is acquired the page protection is changed to read-write. Once the modification latch is released the page protection changes back to read-only.  
Link: <https://technet.microsoft.com/en-us/library/cc966500.aspx>  
Link: <https://blogs.msdn.microsoft.com/psssql/2012/11/12/how-can-reference-counting-be-a-leading-memory-scribbler-cause>

#### Trace Flag: 818

Function: Turn on ringbuffer to store info about IO write operations. Used to troubleshoot IO problems  
Link: <https://support.microsoft.com/help/826433/>  
Link: <https://technet.microsoft.com/en-us/library/cc966500.aspx>  
Link: <https://support.microsoft.com/help/828339/>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: ?

#### Trace Flag: 822

**Undocumented trace flag**  
Function: A workaround for SQL 2000 over-committing memory on the machine  
Link: None

#### Trace Flag: 825

**Undocumented trace flag**  
Function: In SQL 2000, enables Buffer Pool support for NUMA. TF 888 must be used.  
Link: None

#### Trace Flag: 828

Function: SQL 8 - When enabled checkpoint ignores the recovery interval target and keeps steady I/O otherwise it uses recovery interval setting as a target for the length of time that checkpoint will take  
Link: <https://support.microsoft.com/help/906121> Link: <https://blogs.msdn.microsoft.com/psssql/2008/04/11/how-it-works-sql-server-checkpoint-flushcache-outstanding-io-target/>

#### Trace Flag: 830

Function: SQL 9 – Disable the reporting of CPU Drift errors in the SQL Server errorlog like SQL Server has encountered 2 occurrence(s) of I/O requests taking longer than 15 seconds to complete  
Link: <https://support.microsoft.com/help/897284>  
Link: <https://technet.microsoft.com/en-us/library/aa175396(v=SQL.80).aspx>

#### Trace Flag: 831

**Undocumented trace flag**  
Function: Protect unchanged pages in the buffer pool to catch memory corruptions  
Link: None

#### Trace Flag: 834

Function: Uses Microsoft Windows large-page allocations for the buffer pool. Trace flag 834 causes SQL Server to use Microsoft Windows large-page allocations for the memory that is allocated for the buffer pool. The page size varies depending on the hardware platform, but the page size may be from 2 MB to 16 MB. Large pages are allocated at startup and are kept throughout the lifetime of the process. Trace flag 834 improves performance by increasing the efficiency of the translation look-aside buffer (TLB) in the CPU.  
**Note: If you are using the Columnstore Index feature of SQL Server 2012 to SQL Server 2016, we do not recommend turning on trace flag 834.**  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: <https://support.microsoft.com/help/3210239>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 835

**Undocumented trace flag**  
Function: SQL 9 / 10 – On 64 bit SQL Server it turns off Lock pages in memory  
Link: None  
Scope: ?

#### Trace Flag: 836

Function: Trace flag 836 causes SQL Server to size the buffer pool at startup based on the value of the max server memory option instead of based on the total physical memory. You can use trace flag 836 to reduce the number of buffer descriptors that are allocated at startup in 32-bit Address Windowing Extensions (AWE) mode. Trace flag 836 applies only to 32-bit versions of SQL Server that have the AWE allocation enabled. You can turn on trace flag 836 only at startup.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: <https://blogs.msdn.microsoft.com/psssql/2012/12/11/how-it-works-sql-server-32-bit-paeawe-on-sql-2005-2008-and-2008-r2-not-using-as-much-ram-as-expected/>  
Scope: global only

#### Trace Flag: 839

Function: (Apparently) forces SQL Server to treate all NUMA memory as “flat”, as if it was SMP.  
Link: <https://blogs.msdn.microsoft.com/psssql/2010/04/02/how-it-works-soft-numa-io-completion-thread-lazy-writer-workers-and-memory-nodes>

#### Trace Flag: 840

Function: SQL 9 – When trace turned on, SQL Server can perform larger I/O extent reads to populate the buffer pool when SQL Server starts this populates the buffer pool faster. Additionally, the larger I/O extent reads improve the initial query compilation and the response time when SQL Server starts.  
Link: <https://blogs.msdn.microsoft.com/ialonso/2011/12/09/the-read-ahead-that-doesnt-count-as-read-ahead>

#### Trace Flag: 842

**Undocumented trace flag**  
Function: Use sys.dm\_os\_memory\_node\_access\_stats to verify local vs. foreign memory under NUMA configurations after turning on this flag  
Link: None

#### Trace Flag: 845

Function: Enable Lock pages in Memory on Standard Edition  
Link: <https://support.microsoft.com/help/970070>  
Link: <https://support.microsoft.com/help/2708594/fix-locked-page-allocations-are-enabled-without-any-warning-after-you-upgrade-to-sql-server-2012>

**Undocumented trace flag**

#### Trace Flag: 851

Function: According to Bob Ward’s PASS 2014 talk on SQL Server IO, “disable[s] BPE even if enabled via ALTER SERVER”  
Link: None

#### Trace Flag: 861

**Undocumented trace flag**  
Function: According to the error log this disables buffer pool extension.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 862

**Undocumented trace flag**  
Function: According to the error log this enables buffer pool extension. This TF probably doesn’t do anything anymore.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 876

**Undocumented trace flag**  
Function: Turns 8k page allocations for Column Store segments into 2MB instead.  
Link: <https://twitter.com/slava_oks/status/1044257034361757696>  
Link: <https://github.com/ktaranov/sqlserver-kit/issues/151>  
Scope: ?

**Undocumented trace flag**

#### Trace Flag: 888

Function: Enables support for locked pages for SQL 2000  
Link: None

#### Trace Flag: 902

Function: Bypasses execution of database upgrade script when installing a Cumulative Update or Service Pack. If you encounter an error during script upgrade mode, it is recommended to contact Microsoft SQL Customer Service and Support (CSS) for further guidance.  
**Warning: This trace flag is meant for troubleshooting of failed updates during script upgrade mode, and it is not supported to run it continuously in a production environment. Database upgrade scripts needs to execute successfully for a complete install of Cumulative Updates and Service Packs. Not doing so can cause unexpected issues with your SQL Server instance.**  
Link: <https://support.microsoft.com/help/2163980>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://blogs.msdn.microsoft.com/luti/2017/05/17/sql-server-offline-after-applying-service-pack/>  
Scope: global only

#### Trace Flag: 916

**Undocumented trace flag**  
Function: The KB article references the flag in the context of seeing a Profiler dump  
Link: None

#### Trace Flag: 1106

Function: SQL 9 - Used space in tempdb increases continuously when you run a query that creates internal objects in tempdb  
Link: <https://support.microsoft.com/help/947204>  
Link: <https://blogs.msdn.microsoft.com/arvindsh/2014/02/24/tracking-tempdb-internal-object-space-usage-in-sql-2012>

#### Trace Flag: 1117

Function: When a file in the filegroup meets the autogrow threshold, all files in the filegroup grow.  
**Note: Beginning with SQL Server 2016 this behavior is controlled by the AUTOGROW\_SINGLE\_FILE and AUTOGROW\_ALL\_FILES option of ALTER DATABASE, and trace flag 1117 has no affect. For more information, see** [**ALTER DATABASE File and Filegroup Options (Transact-SQL)**](https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-database-transact-sql-file-and-filegroup-options)**.**  
Link: <https://www.littlekendra.com/2017/01/03/parallelism-and-tempdb-data-file-usage-in-sql-server/>  
Link: [SQL Server 2016 : Getting tempdb a little more right](https://blogs.sentryone.com/aaronbertrand/sql-server-2016-tempdb-fixes/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlskills.com/blogs/paul/tempdb-configuration-survey-results-and-advice>  
Link: <https://blogs.msdn.microsoft.com/ialonso/2011/12/01/attempt-to-grow-all-files-in-one-filegroup-and-not-just-the-one-next-in-the-autogrowth-chain-using-trace-flag-1117>  
Link: <http://sql-articles.com/articles/general/day-6trace-flag-1117-auto-grow-equally-in-all-data-file>  
Link: <http://www.ryanjadams.com/2017/05/trace-flag-1117-growth-contention/>  
Link: <https://www.sqlskills.com/blogs/paul/misconceptions-around-tf-1118/>  
Scope: global only

#### Trace Flag: 1118

Function: Removes most single page allocations on the server, reducing contention on the SGAM page. When a new object is created, by default, the first eight pages are allocated from different extents (mixed extents). Afterwards, when more pages are needed, those are allocated from that same extent (uniform extent). The SGAM page is used to track these mixed extents, so can quickly become a bottleneck when numerous mixed page allocations are occurring. This trace flag allocates all eight pages from the same extent when creating new objects, minimizing the need to scan the SGAM page.  
**Note: Beginning with SQL Server 2016 this behavior is controlled by the SET MIXED\_PAGE\_ALLOCATION option of ALTER DATABASE, and trace flag 1118 has no affect. For more information, see ALTER DATABASE SET Options (Transact-SQL).**  
Link: <http://blogs.msdn.com/b/psssql/archive/2008/12/17/sql-server-2005-and-2008-trace-flag-1118-t1118-usage.aspx>  
Link: <http://www.sqlskills.com/blogs/paul/misconceptions-around-tf-1118/>  
Link: <https://support.microsoft.com/help/328551>  
Link: [SQL Server 2016 : Getting tempdb a little more right](https://blogs.sentryone.com/aaronbertrand/sql-server-2016-tempdb-fixes/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://chrisadkin.org/2015/04/14/well-known-and-not-so-well-known-sql-server-tuning-knobs-and-switches>  
Scope: global only

#### Trace Flag: 1119

Function: Turns off mixed extent allocation (Similar to 1118?)  
Link: [TECHNET List Of SQL Server Trace Flags](http://social.technet.microsoft.com/wiki/contents/articles/13105.trace-flags-in-sql-server.aspx)

#### Trace Flag: 1124

**Undocumented trace flag**  
Function: Unknown. Has been reportedly found turned on in some SQL Server instances running Dynamics AX. Also rumored to be invalid in public builds of SQL Server  
Link: None

#### Trace Flag: 1140

**Undocumented trace flag**  
Function: A workaround for a bug in SQL 2005 SP2, SP3, and SQL 2008, where mixed page allocations climb continually, due to a change in the way that mixed-page allocations are done.  
Link: None

#### Trace Flag: 1165

**Undocumented trace flag**  
Function: This [presentation](http://www.youtube.com/watch?v=SvseGMobe2w&feature=youtu.be) by Bob Ward says that this TF outputs the recalculated #’s (every 8192 allocations) for the proportional fill algorithm in database allocation when multiple files are present..  
Link: None

#### Trace Flag: 1180

**Undocumented trace flag**  
Function: SQL 7 - Forces allocation to use free pages for text or image data and maintain efficiency of storage. Helpful in case when DBCC SHRINKFILE and SHRINKDATABASE commands may not work because of sparsely populated text, ntext, or image columns.  
Link: None

#### Trace Flag: 1197

**Undocumented trace flag**  
Function: Applies only in the case of SQL 7 – SP3, similar with trace flag 1180  
Link: None

#### Trace Flag: 1200

Function: Prints detailed lock information as every request for a lock is made (the process ID and type of lock requested)  
Link: [TECHNET List Of SQL Server Trace Flags](http://social.technet.microsoft.com/wiki/contents/articles/13105.trace-flags-in-sql-server.aspx)  
Link: <https://blogs.msdn.microsoft.com/sqlserverstorageengine/2008/03/30/tempdb-table-variable-vs-local-temporary-table>  
Link: [KB169960](https://web.archive.org/web/20150111103047/http:/support.microsoft.com:80/kb/169960)  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: ?

#### Trace Flag: 1202

**Undocumented trace flag**  
Function: Insert blocked lock requests into syslocks  
Link: None

#### Trace Flag: 1204

Function: Returns the resources and types of locks participating in a deadlock and also the current command affected. Writes information about deadlocks to the ERRORLOG in a "text format"  
Link: <https://support.microsoft.com/help/832524>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: global only

#### Trace Flag: 1205

Function: More detailed information about the command being executed at the time of a deadlock. Documented in SQL 7 BOL.  
Link: <https://support.microsoft.com/help/832524/sql-server-technical-bulletin---how-to-resolve-a-deadlock>

#### Trace Flag: 1206

Function: Used to complement flag 1204 by displaying other locks held by deadlock parties  
Link: [KB169960](https://web.archive.org/web/20150111103047/http:/support.microsoft.com:80/kb/169960)

#### Trace Flag: 1208

Function: KB: “Prints the host name and program name supplied by the client. This can help identify a client involved in a deadlock, assuming the client specifies a unique value for each connection.”  
Link: [KB169960](https://web.archive.org/web/20150111103047/http:/support.microsoft.com:80/kb/169960)

#### Trace Flag: 1211

Function: Disables lock escalation based on memory pressure, or based on number of locks. The SQL Server Database Engine will not escalate row or page locks to table locks. Using this trace flag can generate excessive numbers of locks. This can slow the performance of the Database Engine, or cause 1204 errors (unable to allocate lock resource) because of insufficient memory. If both trace flag 1211 and 1224 are set, 1211 takes precedence over 1224. However, because trace flag 1211 prevents escalation in every case, even under memory pressure, we recommend that you use 1224. This helps avoid "out-of-locks" errors when many locks are being used.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlskills.com/blogs/paul/a-sql-server-dba-myth-a-day-2330-lock-escalation>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: global or session

#### Trace Flag: 1216

**Undocumented trace flag**  
Function: SQL 7 - Disables Health reporting. Lock monitor when detects a (worker thread) resource level blocking scenario. If a SPID that owns a lock is currently queued to the scheduler, because all the assigned worker threads have been created and all the assigned worker threads are in an un-resolvable wait state, the following error message is written to the SQL Server error log: Error 1223: Process ID %d:%d cannot acquire lock "%s" on resource %s because a potential deadlock exists on Scheduler %d for the resource. Process ID %d:% d holds a lock "%h" on this resource.  
Link: None

#### Trace Flag: 1217

**Undocumented trace flag**  
Function: Disables (for 7.0) the “UMS Health” reporting messages described in the KB article.  
Link: None

#### Trace Flag: 1222

Function: Returns the resources and types of locks that are participating in a deadlock and also the current command affected, in an XML format that does not comply with any XSD schema.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://blog.sqlauthority.com/2017/01/09/sql-server-get-historical-deadlock-information-system-health-extended-events>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: global only

#### Trace Flag: 1224

Function: Disables lock escalation based on the number of locks. However, memory pressure can still activate lock escalation. The Database Engine escalates row or page locks to table (or partition) locks if the amount of memory used by lock objects exceeds one of the following conditions:

* Forty percent of the memory that is used by Database Engine. This is applicable only when the locks parameter of sp\_configure is set to 0.
* Forty percent of the lock memory that is configured by using the locks parameter of sp\_configure. For more information, see [Server Configuration Options (SQL Server)](https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/server-configuration-options-sql-server). If both trace flag 1211 and 1224 are set, 1211 takes precedence over 1224. However, because trace flag 1211 prevents escalation in every case, even under memory pressure, we recommend that you use 1224. This helps avoid "out-of-locks" errors when many locks are being used.  
  **Note: Lock escalation to the table- or HoBT-level granularity can also be controlled by using the LOCK\_ESCALATION option of the ALTER TABLE statement.**  
  Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
  Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
  Scope: global or session

#### Trace Flag: 1228

Function: Enable lock partitioning. By default, lock partitioning is enabled when a server has 16 or more CPUs. Otherwise, lock partitioning is disabled. Trace flag 1228 enables lock partitioning for 2 or more CPUs. Trace flag 1229 disables lock partitioning. Trace flag 1229 overrides trace flag 1228 if trace flag 1228 is also set. Lock partitioning is useful on multiple-CPU servers where some tables have very high lock rates. You can turn on trace flag 1228 and trace flag 1229 only at startup.  
Link: [Trace Flag 1228 and 1229](http://www.sqlservercentral.com/Forums/Topic741825-146-1.aspx)  
Link: [Microsoft SQL Server 2005 TPC-C Trace Flags](http://webcache.googleusercontent.com/search?q=cache:Nttlt2Dp8egJ:blogs.msmvps.com/gladchenko/2009/08/21/sql_trace_flags_tpc-c/+&cd=6&hl=en&ct=clnk&gl=ru)

#### Trace Flag: 1229

Function: Enable lock partitioning. By default, lock partitioning is enabled when a server has 16 or more CPUs. Otherwise, lock partitioning is disabled. Trace flag 1228 enables lock partitioning for 2 or more CPUs. Trace flag 1229 disables lock partitioning. Trace flag 1229 overrides trace flag 1228 if trace flag 1228 is also set. Lock partitioning is useful on multiple-CPU servers where some tables have very high lock rates. You can turn on trace flag 1228 and trace flag 1229 only at startup.  
Link: [Trace Flag 1228 and 1229](http://www.sqlservercentral.com/Forums/Topic741825-146-1.aspx)  
Link: [Microsoft SQL Server 2005 TPC-C Trace Flags](http://webcache.googleusercontent.com/search?q=cache:Nttlt2Dp8egJ:blogs.msmvps.com/gladchenko/2009/08/21/sql_trace_flags_tpc-c/+&cd=6&hl=en&ct=clnk&gl=ru)

#### Trace Flag: 1236

Function: Enables database lock partitioning. Fixes performance problem in scenarios with high lock activity in SQL 2012 and SQL 2014.  
**Note: Beginning with SQL Server 2012 SP3 and SQL Server 2014 SP1 this behavior is controlled by the engine and trace flag 1236 has no effect.**  
Link: <https://support.microsoft.com/help/2926217>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 1237

Function: Allows the ALTER PARTITION FUNCTION statement to honor the current user-defined session deadlock priority instead of being the likely deadlock victim by default.  
**Note: Starting with SQL Server 2017 and database** [**compatibility level**](https://docs.microsoft.com/sql/t-sql/statements/alter-database-transact-sql-compatibility-level) **140 this is the default behavior and trace flag 1237 has no effect.**  
Link: <https://support.microsoft.com/help/4025261>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session or query

#### Trace Flag: 1260

Function: Disabled mini-dump for non-yield condition. Disables mini-dump generation for "any of the 17883, 17884, 17887, or 17888 errors. The trace flag can be used in conjunction with trace flag –T1262. For example, you could enable –T1262 to get 10- and a 60-second interval reporting and also enable – T1260 to avoid getting mini-dumps."  
Link: [A Topical Collection of SQL Server Flags v6](https://sqlcrossjoin.files.wordpress.com/2016/04/sqlcrossjoin_traceflagrepository_v6.pdf)  
Link: [How To Diagnose and Correct Errors 17883, 17884, 17887, and 17888](https://msdn.microsoft.com/library/cc917684.aspx)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 1261

**Undocumented trace flag**  
Function: SQL 8 - Disables Health reporting. Lock monitor when detects a (worker thread) resource level blocking scenario. If a SPID that owns a lock is currently queued to the scheduler, because all the assigned worker threads have been created and all the assigned worker threads are in an un-resolvable wait state, the following error message is written to the SQL Server error log: Error 1229: Process ID %d:%d owns resources that are blocking processes on scheduler %d.  
Link: None

#### Trace Flag: 1262

Function: The default behavior (for 1788\* errors) is for SQL to generate a mini-dump on the first occurrence, but never after. 1262 changes the behavior: “When –T1262 is enabled, a mini-dump is generated when the non-yielding condition is declared (15 seconds) and at subsequent 60-second intervals for the same non-yield occurrence. A new nonDiagCorrect17883etc; yielding occurrence causes dump captures to occur again.” In SQL 2000 this was a startup-only flag; in 2005+ it can be enabled via TRACEON. Note that the flag is also covered in Khen2005, p400, but with no new information.  
Link: [A Topical Collection of SQL Server Flags v6](https://sqlcrossjoin.files.wordpress.com/2016/04/sqlcrossjoin_traceflagrepository_v6.pdf)  
Link: [How To Diagnose and Correct Errors 17883, 17884, 17887, and 17888](https://msdn.microsoft.com/library/cc917684.aspx)

#### Trace Flag: 1264

Function: Collect process names in non-yielding scenario memory dumps  
Link: [A Topical Collection of SQL Server Flags v6](https://sqlcrossjoin.files.wordpress.com/2016/04/sqlcrossjoin_traceflagrepository_v6.pdf)  
Link: <https://support.microsoft.com/help/2630458/>

#### Trace Flag: 1400

Function: SQL 9 RTM – Enables creation of database mirroring endpoint, which is required for setting up and using database mirroring  
Link: None

#### Trace Flag: 1439

Function: Trace database restart and failover messages to SQL Errorlog for mirrored databases  
Link: [Trace flags in sql server from trace flag 902 to trace flag 1462](http://www.sqlserverf1.com/tag/sql-server-trace-flag-1448/)

#### Trace Flag: 1448

Function: Enables the replication log reader to move forward even if the async secondaries have not acknowledged the reception of a change. Even with this trace flag enabled the log reader always waits for the sync secondaries. The log reader will not go beyond the min ack of the sync secondaries. This trace flag applies to the instance of SQL Server, not just an availability group, an availability database, or a log reader instance. Takes effect immediately without a restart. This trace flag can be activated ahead of time or when an async secondary fails.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 1449

Function: When you use SNAC to connect to an instance of a principal server in a database mirroring session: "The connection attempted to fail over to a server that does not have a failover partner".  
Link: <https://support.microsoft.com/help/936179>

#### Trace Flag: 1462

Function: Disables log stream compression for asynchronous availability groups. This feature is enabled by default on asynchronous availability groups in order to optimize network bandwidth.  
Link: [Tune compression for availability group](https://docs.microsoft.com/sql/database-engine/availability-groups/windows/tune-compression-for-availability-group)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlskills.com/blogs/paul/sql-server-2008-performance-boost-for-database-mirroring>  
Link: <http://sqlblog.com/blogs/joe_chang/archive/2014/03/13/hekaton-and-benchmarks.aspx>  
Scope: global only

#### Trace Flag: 1482

Function: Prints information to the Error Log (3605 is not necessary) for a variety of transaction log operations are done, including when the MinLSN value is reset, when a VLF is formatted, etc.  
Link: None

#### Trace Flag: 1504

Function: Dynamic memory grant expansion can also help with parallel index build plans where the distribution of rows across threads is uneven. The amount of memory that can be consumed this way is not unlimited, however. SQL Server checks each time an expansion is needed to see if the request is reasonable given the resources available at that time. Some insight to this process can be obtained by enabling undocumented trace flag 1504, together with 3604 (for message output to the console) or 3605 (output to the SQL Server error log). If the index build plan is parallel, only 3605 is effective because parallel workers cannot send trace messages cross-thread to the console.  
Link: [Internals of the Seven SQL Server Sorts – Part 1](https://sqlperformance.com/2015/04/sql-plan/internals-of-the-seven-sql-server-sorts-part-1)

#### Trace Flag: 1603

Function: Use standard disk I/O (i.e. turn off asynchronous I/O)  
Link: None

#### Trace Flag: 1604

Function: Once enabled at start up makes SQL Server output information regarding memory allocation requests  
Link: None

#### Trace Flag: 1609

Function: Turns on the unpacking and checking of RPC information in Open Data Services. Used only when applications depend on the old behavior.  
Link: None

#### Trace Flag: 1610

Function: Boot the SQL dataserver with TCP\_NODELAY enabled  
Link: None

#### Trace Flag: 1611

Function: If possible, pin shared memory -- check errorlog for success/failure  
Link: None

#### Trace Flag: 1613

Function: Set affinity of the SQL data server engine's onto particular CPUs -- usually pins engine 0 to processor 0, engine 1 to processor 1...  
Link: None

#### Trace Flag: 1615

Function: Khen2005, page 385 (paraphrased): directs SQL to use threads instead of fiber even if the “lightweight pooling” config option is on. (Apparently, sometimes SQL wouldn’t start successfully when using lightweight pooling, and so this lets you get SQL up and running, so that you can turn the config option off)  
Link: None

#### Trace Flag: 1704

Function: Prints information when a temporary table is created or dropped  
Link: None

#### Trace Flag: 1717

Function: MSShipped bit will be set automatically at Create time when creating stored procedures  
Link: None

#### Trace Flag: 1800

Function: Enables SQL Server optimization when disks of different sector sizes are used for primary and secondary replica log files, in SQL Server AG and Log Shipping environments.  
Link: <https://support.microsoft.com/help/3009974>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 1802

Function: SQL 9 - After detaching a database that resides on network-attached storage, you cannot reattach the SQL Server database  
Link: <https://support.microsoft.com/help/922804>

#### Trace Flag: 1806

Function: Disable Instant File Initialization. Used to guarantee the physical data file space acquisition during data file creation or expansion, on a thin provisioned subsystem  
Link: <http://technet.microsoft.com/en-au/library/cc917726.aspx>  
Link: <https://blogs.msdn.microsoft.com/sql_pfe_blog/2009/12/22/how-and-why-to-enable-instant-file-initialization>  
Link: <http://www.sqlskills.com/blogs/paul/a-sql-server-dba-myth-a-day-330-instant-file-initialization-can-be-controlled-from-within-sql-server>  
Link: <https://support.microsoft.com/help/2574695/file-initialization-takes-a-long-time-for-sql-server-database-related-operations>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: ?

#### Trace Flag: 1807

Function: Enable option to have database files on SMB share for SQL Server 2008 and 2008R2  
Link: <http://blogs.msdn.com/b/varund/archive/2010/09/02/create-a-sql-server-database-on-a-network-shared-drive.aspx>  
Link: <https://support.microsoft.com/help/304261/description-of-support-for-network-database-files-in-sql-server>

#### Trace Flag: 1808

Function: Directs SQL Server to ignore auto-closing databases even if the Auto-close property is set to ON. Must be set globally. Present in Yukon forward  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/04/11/want-your-sql-server-to-simply-ignore-the-auto_close-setting-for-all-open-databases-for-which-it-has-been-enabled>

#### Trace Flag: 1810

Function: Prints the file create/open/close timings  
Link: None

#### Trace Flag: 1816

Function: Bob Ward briefly references this flag in his PASS 2014 SQL Server IO talk, saying that it “could provide more details around errors” that occur with IO done to SQL data files in Azure Storage.  
Link: None

#### Trace Flag: 1851

Function: Anecdotally, from a JustDave’s notes on an Amanda Ford talk at SQL Relay Reading 2014: “...disables the automerge functionality for in-memory oltp”  
Link: <https://justdaveinfo.wordpress.com/2014/10/16/october-13-microsoft-sql-relay-reading>

#### Trace Flag: 1903

Function: SQL 8 - When you capture a SQL Profiler trace in a file and then you try to import the trace files into tables by using the fn\_trace\_gettable function no rows may be returned  
Link: Note

#### Trace Flag: 1905

Function: Unknown  
Link: [Upgrading an expired SQL Server 2016 Evaluation Edition](https://www.codykonior.com/2017/11/30/upgrading-an-expired-sql-server-2016-evaluation-edition/)

#### Trace Flag: 2301

Function: Trace flag 2301 enables advanced optimizations that are specific to decision support queries. This option applies to decision support processing of large data sets.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.queryprocessor.com/ce_join_base_containment_assumption>  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/772232/make-optimizer-estimations-more-accurate-by-using-metadata>  
Scope: global or session or query

#### Trace Flag: 2309

Function: In SQL 2014, enables output from a 3rd parameter for [DBCC SHOW\_STATISTICS](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-show-statistics-transact-sql) such that the partial statistics histogram (for just one partition) is shown.  
Link: <https://sqlperformance.com/2015/05/sql-statistics/incremental-statistics-are-not-used-by-the-query-optimizer>  
Link: <http://blog.dbi-services.com/sql-server-2014-new-incremental-statistics>

#### Trace Flag: 2312

Function: Enables you to set the query optimizer cardinality estimation model to the SQL Server 2014 through SQL Server 2016 versions, dependent of the compatibility level of the database.  
Link: [KB2801413](https://support.microsoft.com/help/2801413)  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlservergeeks.com/sql-server-2014-trace-flags-2312/>  
Scope: global or session or query

#### Trace Flag: 2315

Function: Aaron: I stumbled onto this one. Seems to output memory allocations taken during the compilation process (and maybe the plan as well? “PROCHDR”), as well as memory broker states & values at the beginning and end of compilation.  
Link: None

#### Trace Flag: 2318

Function: Aaron: stumbled onto this one as well. I’ve only seen one type of output so far: “Optimization Stage: HEURISTICJOINREORDER”. Maybe useful in combo with other compilation trace flags to see the timing of join reordering?  
Link: None

#### Trace Flag: 2324

Function: Disables Implied Predicates  
Link: <https://answers.sqlperformance.com/questions/2299/why-not-seek-predicate.html?utm_content=buffer9bed5&utm_medium=social&utm_source=twitter.com&utm_campaign=buffer>

#### Trace Flag: 2328

Function: SQL 9+ - Makes cardinality estimates upon resulting selectivity. The reasoning for this is that one or more of the constants may be statement parameters, which would change from one execution of the statement to the next.  
Link: <https://blogs.msdn.microsoft.com/ianjo/2006/03/28/disabling-constant-constant-comparison-estimation>  
Link: <http://www.queryprocessor.ru/isnumeric_ce_bug_eng>

#### Trace Flag: 2329

Function: Disables “Few Outer Rows” optimization  
Link: <http://www.queryprocessor.com/few-outer-rows-optimization>

#### Trace Flag: 2330

Function: Query performance decreases when sys.dm\_db\_index\_usage\_stats has large number of rows  
Link: <http://www.brentozar.com/archive/2015/11/trace-flag-2330-who-needs-missing-index-requests/>  
Link: <https://chrisadkin.org/2015/04/14/well-known-and-not-so-well-known-sql-server-tuning-knobs-and-switches/>

#### Trace Flag: 2332

Function: PWhite: “Force DML Request Sort (CUpdUtil::FDemandRowsSortedForPerformance)”  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/01/26/optimizing-t-sql-queries-that-change-data.aspx>

#### Trace Flag: 2335

Function: Causes SQL Server to assume a fixed amount of memory is available during query optimization. It does not limit the memory SQL Server grants to execute the query. The memory configured for SQL Server will still be used by data cache, query execution and other consumers.  
**Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: <https://support.microsoft.com/help/2413549>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://dba.stackexchange.com/questions/53726/difference-in-execution-plans-on-uat-and-prod-server>  
Scope: global or session or query

#### Trace Flag: 2336

Function: Aaron: Another one that I stumbled onto. Appears to tie memory info and cached page likelihoods with costing  
Link: None

#### Trace Flag: 2340

Function: Causes SQL Server not to use a sort operation (batch sort) for optimized nested loop joins when generating a plan. Beginning with SQL Server 2016 SP1, to accomplish this at the query level, add the USE HINT query hint instead of using this trace flag.  
**Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://blogs.msdn.microsoft.com/psssql/2010/01/11/high-cpu-after-upgrading-to-sql-server-2005-from-2000-due-to-batch-sort>  
Link: <http://www.queryprocessor.com/batch-sort-and-nested-loops>  
Scope: global or session or query

#### Trace Flag: 2341

Function: Enables the use of a hash join for joins to column store indexes even when the join clause would normally be removed “during query normalization”.  
Link: <https://support.microsoft.com/help/3146123/query-plan-generation-improvement-for-some-columnstore-queries-in-sql-server-2014-or-2016>

#### Trace Flag: 2363

Function: TF Selectivity  
Link: [Cardinality Estimation Framework 2014 First Look](http://www.somewheresomehow.ru/cardinality-estimation-framework-2014-first-look/)  
Link: <http://www.queryprocessor.com/ce-process>  
Link: <https://sqlperformance.com/2014/01/sql-plan/cardinality-estimation-for-multiple-predicates>

#### Trace Flag: 2368

**Undocumented trace flag**  
Function: For one query, this resulted in a parallel plan significantly more expensive than the naturally occurring serial plan. Could be related to trace flag 3651.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 2371

Function: Changes the fixed auto update statistics threshold to dynamic auto update statistics threshold.  
**Note: Beginning with SQL Server 2016 this behavior is controlled by the engine and trace flag 2371 has no effect.**  
Link: <https://support.microsoft.com/help/2754171>  
Link: <http://blogs.msdn.com/b/saponsqlserver/archive/2011/09/07/changes-to-automatic-update-statistics-in-sql-server-traceflag-2371.aspx>  
Link: <https://blogs.msdn.microsoft.com/axinthefield/sql-server-trace-flag-2371-for-dynamics-ax/>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 2372

Function: Displays memory utilization during the optimization process. Memory for Phases Memory before and after deriving properties and rules (verbose)  
Link: [More Undocumented Query Optimizer Trace Flags](http://www.benjaminnevarez.com/2012/04/more-undocumented-query-optimizer-trace-flags/)  
Link: [Cardinality Estimation Framework 2014 First Look](http://www.somewheresomehow.ru/cardinality-estimation-framework-2014-first-look/)  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)

#### Trace Flag: 2373

Function: Displays memory utilization during the optimization process. Memory for Deriving Properties.  
Link: [More Undocumented Query Optimizer Trace Flags](http://www.benjaminnevarez.com/2012/04/more-undocumented-query-optimizer-trace-flags/)  
Link: [Cardinality Estimation Framework 2014 First Look](http://www.somewheresomehow.ru/cardinality-estimation-framework-2014-first-look/)

#### Trace Flag: 2374

**Undocumented trace flag**  
Function: Removes QueryHash and QueryPlanHash information from estimated query plans.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 2382

Function: SSC: “SQL 8 -Statistics collected for system tables.”  
Link: None

#### Trace Flag: 2387

**Undocumented trace flag**  
Function: There was a small change in CPU and IO costs for some operators. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 2388

Function: Changes the output of [DBCC SHOW\_STATISTICS](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-show-statistics-transact-sql). Instead of the normal Header/Vector/Histogram output, instead we get a single row that gives information related to whether the lead column of the stat object is considered to be ascending or not. This TF is primarily helpful in watching the state of a stat object change from “Unknown”, to “Ascending” (and potentially to “Stationary”). Also In SQL Server, if you want to see the information of last four statistics update on a statistics object then you can use trace flag 2388. In simple words, we can say that this trace flag provide us the historical information about statistics update.  
Link: [SQL Server - estimates outside of the histogram - half-baked draft](http://sql-sasquatch.blogspot.ru/2017/09/sql-server-estimates-outside-of.html)  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-2388/>  
Scope: session only

#### Trace Flag: 2389

Function: Enable automatically generated quick statistics for ascending keys (histogram amendment). If trace flag 2389 is set, and a leading statistics column is marked as ascending, then the histogram used to estimate cardinality will be adjusted at query compile time.  
Link: [KB2801413](https://support.microsoft.com/help/2801413)  
Link: <http://blogs.msdn.com/b/ianjo/archive/2006/04/24/582227.aspx>  
Link: <http://www.sqlmag.com/article/tsql3/making-the-most-of-automatic-statistics-updating--96767>  
Link: <http://sqlperformance.com/2016/07/sql-statistics/trace-flag-2389-new-cardinality-estimator>  
Link: <https://www.sswug.org/sswugresearch/community/trace-flag-2389-and-the-new-cardinality-estimator/>  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [SQL Server - estimates outside of the histogram - half-baked draft](http://sql-sasquatch.blogspot.ru/2017/09/sql-server-estimates-outside-of.html)  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-2389/>  
Scope: global or session or query

#### Trace Flag: 2390

Function: Enable automatically generated quick statistics for ascending or unknown keys (histogram amendment). If trace flag 2390 is set, and a leading statistics column is marked as ascending or unknown, then the histogram used to estimate cardinality will be adjusted at query compile time  
Link: <http://blogs.msdn.com/b/ianjo/archive/2006/04/24/582227.aspx>  
Link: [KB2801413](https://support.microsoft.com/help/2801413)  
Link: <http://www.sqlmag.com/article/tsql3/making-the-most-of-automatic-statistics-updating--96767>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://blogs.msdn.microsoft.com/ianjo/2006/04/24/ascending-keys-and-auto-quick-corrected-statistics>  
Link: [SQL Server - estimates outside of the histogram - half-baked draft](http://sql-sasquatch.blogspot.ru/2017/09/sql-server-estimates-outside-of.html)  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-2390/> Scope: global or session or query

#### Trace Flag: 2392

**Undocumented trace flag**  
Function: Trace Flag 2392 can be used to turn the missing index feature off completely (as a workaround to the issue that is corrected by the hotfix). This trace flag has been in the product since SQL Server 2005. The problem is, it will not disable/enable missing index stats collection unless it is enabled at startup. If you set it as a startup TF and restart SQL Server, then no missing index stats are collected. If you then subsequently disable TF 2392 while SQL Server is running, it still won’t collect any missing index stats (despite what you may expect).  
Link: <https://www.sqlskills.com/blogs/glenn/sql-server-missing-indexes-feature-and-trace-flag-2392/>  
Link: <https://support.microsoft.com/help/4042232/fix-access-violation-when-you-cancel-a-pending-query-if-the-missing-in>  
Scope: global only

#### Trace Flag: 2398

Function: Another one I stumbled upon myself...outputs info about “Smart Seek costing”: e.g.: “Smart seek costing (75.2) :: 1.34078e+154 , 1”  
Link: None

#### Trace Flag: 2399

**Undocumented trace flag**  
Function: Small changes in operator costs were observed for some queries. These were typically less than 0.01 units.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 2418

**Undocumented trace flag**  
Function: Disables serial Batch mode processing.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 2422

Function: Enables the SQL Server Database Engine to abort a request when the maximum time set by Resource Governor REQUEST\_MAX\_CPU\_TIME\_SEC configuration is exceeded.  
**Note: This trace flag applies to SQL Server 2017 CU3 and higher builds.**  
Link: <https://support.microsoft.com/help/4038419>  
Scope: global only

#### Trace Flag: 2430

Function: Fixes performance problem when using large numbers of locks  
Link: <https://support.microsoft.com/help/2754301>  
Link: <https://support.microsoft.com/help/2746341/fix-high-cpu-usage-when-you-execute-an-update-statement-that-includes-a-where-current-of-cursor-clause-in-sql-server-2008>

#### Trace Flag: 2440

Function: SQL 10 - Parallel query execution strategy on partitioned tables. SQL 9 used single thread per partition parallel query execution strategy. In SQL 10, multiple threads can be allocated to a single partition by turning on this flag.  
Link: <https://social.msdn.microsoft.com/Forums/sqlserver/en-US/dc010af9-afa0-4c87-937c-4343b4e1119a/trace-flag-2440>

#### Trace Flag: 2453

Function: Allow a table variable to trigger recompile when enough number of rows are changed with may allow the query optimizer to choose a more efficient plan.  
**Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: <http://sqlperformance.com/2014/06/t-sql-queries/table-variable-perf-fix>  
Link: <https://support.microsoft.com/help/2952444>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://www.brentozar.com/archive/2017/02/using-trace-flag-2453-improve-table-variable-performance>  
Link: <https://www.brentozar.com/archive/2018/03/table-valued-parameters-unexpected-parameter-sniffing>  
Link: [TEMPDB – Files and Trace Flags and Updates](https://blogs.msdn.microsoft.com/sql_server_team/tempdb-files-and-trace-flags-and-updates-oh-my/)  
Link: <http://www.sqlservergeeks.com/the-correct-cardinality-estimation-for-table-variable-using-trace-flag-2543/>  
Scope: global or session or query

#### Trace Flag: 2456

Function: Relieves RESOURCE\_SEMAPHORE\_MUTEX contention, which may be primarily due to a bug in SQL 2005.  
Link: None

#### Trace Flag: 2466

Function: When SQL Server is determining the runtime DOP for a parallel plan, this flag directs it to use logic found in “older versions” (the post doesn’t say which versions) to determine which NUMA node to place the parallel plan on. This older logic relies on a polling mechanism (roughly every 1 second), and can result in race conditions where 2 parallel plans end up on the same node. The newer logic “significantly reduces” the likelihood of this happening.  
Link: <https://blogs.msdn.microsoft.com/psssql/2013/09/27/how-it-works-maximizing-max-degree-of-parallelism-maxdop>

#### Trace Flag: 2467

Function: “If target MAXDOP target is less than a single node can provide and if trace flag 2467 is enabled attempt to locate least loaded node”  
Link: <https://blogs.msdn.microsoft.com/psssql/2013/09/27/how-it-works-maximizing-max-degree-of-parallelism-maxdop>  
Link: [SQL Server Parallel Query Placement Decision Logic](https://blogs.msdn.microsoft.com/psssql/2016/03/04/sql-server-parallel-query-placement-decision-logic/)

#### Trace Flag: 2468

**Undocumented trace flag**  
Function: “Find the next node that can service the DOP request. Unlike full mode, the global, resource manager keeps track of the last node used. Starting from the last position, and moving to the next node, SQL Server checks for query placement opportunities. If a node can’t support the request SQL Server continues advancing nodes and searching.”  
Link: [SQL Server Parallel Query Placement Decision Logic](https://blogs.msdn.microsoft.com/psssql/2016/03/04/sql-server-parallel-query-placement-decision-logic/)

#### Trace Flag: 2470

**Undocumented trace flag**  
Function: Fixes performance problem when using AFTER triggers on partitioned tables  
Link: <https://support.microsoft.com/help/2606883>

#### Trace Flag: 2479

**Undocumented trace flag**  
Function: When SQL Server is determining the runtime DOP for a parallel plan, this flag directs it to limit the NUMA Node placement for the query to the node that the connection is associated with.  
Link: <https://blogs.msdn.microsoft.com/psssql/2013/09/27/how-it-works-maximizing-max-degree-of-parallelism-maxdop>  
Link: [SQL Server Parallel Query Placement Decision Logic](https://blogs.msdn.microsoft.com/psssql/2016/03/04/sql-server-parallel-query-placement-decision-logic/)

#### Trace Flag: 2486

**Undocumented trace flag**  
Function: In SQL 2016 (CTP 3.0 at least), enables output for the “query\_trace\_column\_values” Extended Event, allowing the value of output columns from individual plan iterators to be traced.  
Link: <http://www.queryprocessor.com/query-trace-column-values>

**Undocumented trace flag**

#### Trace Flag: 2505

Function: SQL 7 - Prevents DBCC TRACEON 208, SPID 10 errors from appearing in the error log (Note: DBCC TRACEON(208) just means “SET QUOTED IDENTIFIER ON”)  
Link: None

#### Trace Flag: 2508

**Undocumented trace flag**  
Function: Disables parallel non-clustered index checking for DBCC CHECKTABLE  
Link: None

#### Trace Flag: 2509

**Undocumented trace flag**  
Function: Used with DBCC CHECKTABLE to see the total count of forward records in a table  
Link: None

#### Trace Flag: 2514

**Undocumented trace flag**  
Function: Verbose Merge Replication logging to msmerge\_history table for troubleshooting Merger repl performance  
Link: <http://sqlblog.com/blogs/argenis_fernandez/archive/2012/05/29/ghost-records-backups-and-database-compression-with-a-pinch-of-security-considerations.aspx>

#### Trace Flag: 2520

**Undocumented trace flag**  
Function: For SQL Server prior 2005. Forces DBCC HELP to return syntax of undocumented DBCC statements. If 2520/2588 is not turned on, DBCC HELP will refuse to give you the syntax stating: "No help available for DBCC statement 'undocumented statement'". Also affects dbcc help ('?')  
Link: <http://www.sqlskills.com/blogs/paul/dbcc-writepage/>  
Scope: session only

#### Trace Flag: 2521

**Undocumented trace flag**  
Function: SQL 7 SP2 - Facilitates capturing a Sqlservr.exe user-mode crash dump for postmortem analysis  
Link: None

#### Trace Flag: 2469

Function: Enables alternate exchange for INSERT INTO ... SELECT into a partitioned columnstore index.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/3204769/>  
Scope: global or session or query

#### Trace Flag: 2528

Function: Disables parallel checking of objects by DBCC CHECKDB, DBCC CHECKFILEGROUP, and DBCC CHECKTABLE. By default, the degree of parallelism is automatically determined by the query processor. The maximum degree of parallelism is configured just like that of parallel queries. For more information, see [Configure the max degree of parallelism Server Configuration Option](https://msdn.microsoft.com/en-us/library/ms189094.aspx). Parallel DBCC should typically be left enabled. For DBCC CHECKDB, the query processor reevaluates and automatically adjusts parallelism with each table or batch of tables checked. Sometimes, checking may start when the server is almost idle. An administrator who knows that the load will increase before checking is complete may want to manually decrease or disable parallelism. Disabling parallel checking of DBCC can cause DBCC to take much longer to complete and if DBCC is run with the TABLOCK feature enabled and parallelism set off, tables may be locked for longer periods of time.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://technet.microsoft.com/en-us/library/ms189094.aspx>  
Link: <http://www.sqlskills.com/blogs/paul/checkdb-from-every-angle-how-long-will-checkdb-take-to-run>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: global or session

#### Trace Flag: 2529

**Undocumented trace flag**  
Function: Displays memory usage for DBCC commands when used with TF 3604  
Link: None

#### Trace Flag: 2536

**Undocumented trace flag**  
Function: Allows you to see inactive records in transaction log using fn\_dblog. Similar to trace flag 2537 for older version than SQL Server 2008.  
Link: <http://www.sqlsoldier.com/wp/sqlserver/day19of31daysofdisasterrecoveryhowmuchlogcanabackuplog>

#### Trace Flag: 2537

**Undocumented trace flag**  
Function: Allows you to see inactive records in transaction log using fn\_dblog  
Link: <http://www.sqlsoldier.com/wp/sqlserver/day19of31daysofdisasterrecoveryhowmuchlogcanabackuplog>  
Link: <http://www.sqlskills.com/blogs/paul/finding-out-who-dropped-a-table-using-the-transaction-log>  
Link: <http://sqlserverandme.blogspot.ru/2014/03/how-to-view-transaction-log.html>  
Scope: session only

#### Trace Flag: 2540

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2541

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2542

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)  
Link: [Controlling SQL Server memory dumps](https://blogs.msdn.microsoft.com/psssql/2009/11/17/how-it-works-controlling-sql-server-memory-dumps)

#### Trace Flag: 2543

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2544

Function: Produces a full memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)  
Link: <https://blogs.msdn.microsoft.com/askjay/2010/02/05/how-can-i-create-a-dump-of-sql-server>  
Link: <https://social.msdn.microsoft.com/Forums/sqlserver/en-US/13ce4292-b8a7-41fa-a173-645693957d70/sqldumper?forum=sqldisasterrecovery&forum=sqldisasterrecovery>

#### Trace Flag: 2545

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2546

Function: Dumps all threads for SQL Server in the dump file  
Link: [KB917825](https://support.microsoft.com/help/917825/)  
Link: <https://blogs.msdn.microsoft.com/askjay/2010/02/05/how-can-i-create-a-dump-of-sql-server>  
Link: <https://social.msdn.microsoft.com/Forums/sqlserver/en-US/13ce4292-b8a7-41fa-a173-645693957d70/sqldumper?forum=sqldisasterrecovery&forum=sqldisasterrecovery>  
Link: <https://blogs.msdn.microsoft.com/psssql/2008/09/12/sql-server-2000-2005-2008-recoveryrollback-taking-longer-than-expected>

#### Trace Flag: 2547

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2548

Function: Shrink will run faster with this trace flag if there are LOB pages that need conversion and/or compaction, because that actions will be skipped.  
Link: <http://blogs.msdn.com/b/psssql/archive/2008/03/28/how-it-works-sql-server-2005-dbcc-shrink-may-take-longer-than-sql-server-2000.aspx>

Thanks to: Andrzej Kukula

#### Trace Flag: 2549

Function: Runs the DBCC CHECKDB command assuming each database file is on a unique disk drive. DBCC CHECKDB command builds an internal list of pages to read per unique disk drive across all database files. This logic determines unique disk drives based on the drive letter of the physical file name of each file.  
**Note: Do not use this trace flag unless you know that each file is based on a unique physical disk. Although this trace flag improve the performance of the DBCC CHECKDB commands which target usage of the PHYSICAL\_ONLY option, some users may not see any improvement in performance. While this trace flag improves disk I/O resources usage, the underlying performance of disk resources may limit the overall performance of the DBCC CHECKDB command.**  
Link: <http://blogs.msdn.com/b/saponsqlserver/archive/2011/12/22/faster-dbcc-checkdb-released-in-sql-2008-r2-sp1-traceflag-2562-amp-2549.aspx>  
Link: <https://support.microsoft.com/help/2634571>  
Link: <https://support.microsoft.com/help/2732669>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 2550

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2551

Function: Produces a filtered memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/477863/sql-server-is-terminating-because-of-fatal-exception-c0150014>

#### Trace Flag: 2552

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2553

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2554

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2555

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2556

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2557

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2558

Function: Disables integration between CHECKDB and Watson  
Link: [KB917825](https://support.microsoft.com/help/917825/)  
Link: [Controlling SQL Server memory dumps](https://blogs.msdn.microsoft.com/psssql/2009/11/17/how-it-works-controlling-sql-server-memory-dumps)

#### Trace Flag: 2559

Function: Unknown, but related to controlling the contents of a memory dump  
Link: [KB917825](https://support.microsoft.com/help/917825/)

#### Trace Flag: 2562

Function: Runs the DBCC CHECKDB command in a single "batch" regardless of the number of indexes in the database. By default, the DBCC CHECKDB command tries to minimize tempdb resources by limiting the number of indexes or "facts" that it generates by using a "batches" concept. This trace flag forces all processing into one batch. One effect of using this trace flag is that the space requirements for tempdb may increase. Tempdb may grow to as much as 5% or more of the user database that is being processed by the DBCC CHECKDB command.  
**Note: Although this trace flag improve the performance of the DBCC CHECKDB commands which target usage of the PHYSICAL\_ONLY option, some users may not see any improvement in performance. While this trace flag improves disk I/O resources usage, the underlying performance of disk resources may limit the overall performance of the DBCC CHECKDB command.**  
Link: <http://blogs.msdn.com/b/saponsqlserver/archive/2011/12/22/faster-dbcc-checkdb-released-in-sql-2008-r2-sp1-traceflag-2562-amp-2549.aspx>  
Link: <https://support.microsoft.com/help/2634571>  
Link: <https://support.microsoft.com/help/2732669>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 2566

Function: Runs the DBCC CHECKDB command without data purity check unless DATA\_PURITY option is specified.  
**Note: Column-value integrity checks are enabled by default and do not require the DATA\_PURITY option. For databases upgraded from earlier versions of SQL Server, column-value checks are not enabled by default until DBCC CHECKDB WITH DATA\_PURITY has been run error free on the database at least once. After this, DBCC CHECKDB checks column-value integrity by default.**  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://sqlperformance.com/2012/11/io-subsystem/minimize-impact-of-checkdb>  
Link: <https://support.microsoft.com/help/2888996/fix-data-purity-corruption-in-sys.sysbinobjs-table-in-master-database-when-you-log-on-to-sql-server-by-using-the-sa-account-and-then-run-dbcc-checkdb>  
Scope: global only

#### Trace Flag: 2588

Function: For SQL Server since 2005. Forces DBCC HELP to return syntax of undocumented DBCC statements. If 2520/2588 is not turned on, DBCC HELP will refuse to give you the syntax stating: "No help available for DBCC statement 'undocumented statement'". Also affects dbcc help ('?')  
Link: <http://www.sqlskills.com/blogs/paul/dbcc-writepage/>  
Scope: session only

#### Trace Flag: 2701

Function: SQL 6.5 - Sets the @@ERROR system function to 50000 for RAISERROR messages with severity levels of 10 or less. When disabled, sets the @@ERROR system function to 0 for RAISERROR messages with severity levels of 10 or less  
Link: None

#### Trace Flag: 2861

Function: Keep zero cost plans in cache. Tip: Avoid Using Trace Flag 2861 to Cache Zero-Cost Query Plan  
Link: None

#### Trace Flag: 2880, 2881

Function: Both 2880 and 2881 are related to a SQL 2000 hotfix introduced to solve problems where ad-hoc queries would cause the procedure cache to get too big  
Link: None

#### Trace Flag: 3001

Function: Stops sending backup entries into MSDB  
Link: <https://bytes.com/topic/sql-server/answers/162385-how-do-i-prevent-sql-2000-posting-message-event-viewer-application-log>

#### Trace Flag: 3004

Function: Returns more info about Instant File Initialization. Shows information about backups and file creations use with [3605](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#3605) to direct to error log. Can be used to ensure that SQL Server has been configured to take advantage of IFI correctly.  
Link: <https://blogs.msdn.microsoft.com/psssql/2008/01/23/how-it-works-what-is-restorebackup-doing/>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Link: <https://blogs.msdn.microsoft.com/sql_pfe_blog/2009/12/22/how-and-why-to-enable-instant-file-initialization/>  
Link: [Undocumented Trace Flags: Inside the Restore Process](https://blog.rdx.com/undocumented-trace-flags-inside-the-restore-process/)  
Scope: session only

#### Trace Flag: 3014

Function: Returns more info about backups to the errorlog: Backup activity, Restore activity , File creation.  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Link: <https://blogs.msdn.microsoft.com/psssql/2008/02/06/how-it-works-how-does-sql-server-backup-and-restore-select-transfer-sizes>  
Link: [Undocumented Trace Flags: Inside the Restore Process](https://blog.rdx.com/undocumented-trace-flags-inside-the-restore-process/)  
Link: [What’s CHECKDB doing in my database restore?](http://www.mikefal.net/2018/04/10/whats-checkdb-doing-in-my-database-restore/)  
Scope: session only

#### Trace Flag: 3023

Function: Enables CHECKSUM option as default for BACKUP command  
**Note: Beginning with SQL Server 2014 this behavior is controlled by setting the backup checksum default configuration option. For more information, see** [**Server Configuration Options (SQL Server)**](https://msdn.microsoft.com/en-us/library/ms189631.aspx).  
Link: <https://support.microsoft.com/help/2656988>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 3028

Function: Enables a hotfix for a problem encountered when backing up to tape with specific backup options  
Link: None

#### Trace Flag: 3031

Function: SQL 9 - Will turn the NO\_LOG and TRUNCATE\_ONLY options into checkpoints in all recovery modes  
Link: <http://www.sqlskills.com/blogs/paul/backup-log-with-no_log-use-abuse-and-undocumented-trace-flags-to-stop-it>

#### Trace Flag: 3034

Function: Overrides the server default, and thus always forces backup compression unless the backup command had the no\_compression clause explicitly present.  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/02/24/vdi-backups-and-backup-compression-default>

#### Trace Flag: 3035

Function: Overrides the server default to always avoid compression, unless the backup command explicitly uses the compression clause. If both 3034 and 3035 are enabled, 3035 takes precedence  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/02/24/vdi-backups-and-backup-compression-default>

#### Trace Flag: 3039

Function: As long as the SQL edition supports backup compression, this will allow VDI backups to be affected by the default compression setting just as non-VDI BACKUP commands are affected.  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/02/24/vdi-backups-and-backup-compression-default>

#### Trace Flag: 3042

Function: Bypasses the default backup compression pre-allocation algorithm to allow the backup file to grow only as needed to reach its final size. This trace flag is useful if you need to save on space by allocating only the actual size required for the compressed backup. Using this trace flag might cause a slight performance penalty (a possible increase in the duration of the backup operation). For more information about the pre-allocation algorithm, see [Backup Compression (SQL Server)](https://msdn.microsoft.com/en-us/library/bb964719.aspx).  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/2001026/inf-space-requirements-for-backup-devices-in-sql-server>  
Link: <https://blogs.msdn.microsoft.com/psssql/2011/08/11/how-compressed-is-your-backup/>  
Link: <https://docs.microsoft.com/en-us/sql/relational-databases/backup-restore/backup-compression-sql-server>  
Link: <https://sqlstudies.com/2017/03/16/compressed-backup-errors-and-tf-3042/>  
Scope: global only

#### Trace Flag: 3051

Function: Enables SQL Server Backup to URL logging to a specific error log file.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://msdn.microsoft.com/en-us/library/jj919149.aspx>  
Scope: global only

#### Trace Flag: 3057

Function: Enables functionality after a hotfix that allows a log backup that was taken on a t-logfile hosted on a drive with “Bytes per physical sector”=512 to be restored onto a log file/drive that has “Bytes per physical sector”=4096  
Link: <https://support.microsoft.com/help/2987585/restore-log-with-standby-mode-on-an-advanced-format-disk-may-cause-a-9004-error-in-sql-server-2008-r2-or-sql-server-2012>

#### Trace Flag: 3101

Function: Fix performance problems when restoring database with CDC  
Link: <https://support.microsoft.com/help/2567366/>

#### Trace Flag: 3104

Function: Causes SQL Server to bypass checking for free space  
Link: <http://sqlblogcasts.com/blogs/martinbell/archive/2011/07/06/Mount-point-Permission-Issues.aspx>  
Link: <http://www.databasejournal.com/features/mssql/article.php/1547551/Troubleshooting-SQL-Server-BackupRestore-Problems.htm>

#### Trace Flag: 3106

Function: Required to move sys databases  
Link: None

#### Trace Flag: 3111

Function: “FIX: Backup or Restore Using Large Transaction Logs May Return Error 3241” Causes LogMgr::ValidateBackedupBlock to be skipped during backup and restore operations, allowing backups of very large T-logs to succeed.  
Link: None

#### Trace Flag: 3117

Function: QL 9 - SQL Server 2005 tries to restore the log files and the data files in a single step which some third-party snapshot backup utilities do not support. Turing on 3117 does things the SQL 8 way multiple-step restore process.  
Link: None

#### Trace Flag: 3205

Function: Disable HW compression for backup to tape drives  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 3207

Function: Fixes SQL 6.5 so that tape drives work correctly with DUMP and LOAD statements  
Link: None

#### Trace Flag: 3210

Function: According to Bob Ward’s PASS 2014 talk on SQL Server IO, prints information about “collisions and wait times” that occur between the various “Asynchronous Disk Pool” threads during BACKUP (what about RESTORE?) operations.  
Link: None

#### Trace Flag: 3212

Function: Prints “Backup stats” to the SQL log  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/10/24/why-does-restoring-a-database-needs-tempdb>

#### Trace Flag: 3213

Function: Output buffer info for backups to ERRORLOG  
Link: <https://blogs.msdn.microsoft.com/psssql/2008/02/06/how-it-works-how-does-sql-server-backup-and-restore-select-transfer-sizes>  
Link: <https://blogs.msdn.microsoft.com/psssql/2008/01/28/how-it-works-sql-server-backup-buffer-exchange-a-vdi-focus/>  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-3213/>  
Scope: global or session

#### Trace Flag: 3216

Function: Prints quite a lot of info about RESTORE internals. Only seems to print to the error log (TF 3605 is required).  
Link: <http://jamessql.blogspot.ru/2013/07/trace-flag-for-backup-and-restore.html>

#### Trace Flag: 3222

Function: Disables the read ahead that is used by the recovery operation during roll forward operations  
Link: [TECHNET List Of SQL Server Trace Flags](http://social.technet.microsoft.com/wiki/contents/articles/13105.trace-flags-in-sql-server.aspx)

#### Trace Flag: 3226

Function: By default, every successful backup operation adds an entry in the SQL Server error log and in the system event log. If you create very frequent log backups, these success messages accumulate quickly, resulting in huge error logs in which finding other messages is problematic. With this trace flag, you can suppress these log entries. This is useful if you are running frequent log backups and if none of your scripts depend on those entries.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlskills.com/blogs/paul/fed-up-with-backup-success-messages-bloating-your-error-logs>  
Link: <https://blogs.msdn.microsoft.com/sqlserverstorageengine/2007/10/30/when-is-too-much-success-a-bad-thing>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: global only

#### Trace Flag: 3231

Function: SQL 8/9 - Will turn the NO\_LOG and TRUNCATE\_ONLY options into no-ops in FULL/BULK\_LOGGED recovery mode, and will clear the log in SIMPLE recovery mode. When set, BACKUP LOG with TRUNCATE\_ONLY and BACKUP LOG with NO\_LOG do not allow a log backup to run if the database's recovery model is FULL or BULK\_LOGGED.  
Link: <http://www.sqlskills.com/blogs/paul/backup-log-with-no_log-use-abuse-and-undocumented-trace-flags-to-stop-it>  
Link: <http://www.sqlskills.com/blogs/kimberly/understanding-backups-and-log-related-trace-flags-in-sql-server-20002005-and-2008>  
Scope: ?

#### Trace Flag: 3282

**Undocumented trace flag**  
Function: SQL 6.5 - Used after backup restoration fails  
Scope: ?

#### Trace Flag: 3400

Function: Prints the recovery timings  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/392158/recovery-portion-of-sql-2008-restore-takes-much-longer-than-normal-when-restoring-from-sql-2005-backup>

#### Trace Flag: 3408

Function: This forces SQL Server startup to use a single thread when recovering all DBs at SQL Server startup, instead of running through its algorithm for determining how many threads to allocate to DB recovery  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/10/08/how-much-is-crash-recovery-parallelized-in-which-order-are-databases-recovered/>

#### Trace Flag: 3412

**Undocumented trace flag**  
Function: The KB article refers to SQL 6.5, but it is possible that the TF still prints out info to the SQL error log, so leaving it here for now. KB: “...reports when each transaction is rolled forward or back [examine the error log for progress]. However, you will not see any progress if SQL Server is rolling a large transaction forward or back. Additionally, this trace flag duplicates the sp\_configure setting Recovery flags..."  
Link: None

#### Trace Flag: 3422

Function: Cause auditing of transaction log records as they're read (during transaction rollback or log recovery). This is useful because there is no equivalent to page checksums for transaction log records and so no way to detect whether log records are being corrupted e careful with these trace flags - I don't recommend using them unless you are experiencing corruptions that you can't diagnose. Turning them on will cause a big CPU hit because of the extra auditing that's happening.  
Link: <https://support.microsoft.com/help/215458>  
Link: <http://www.sqlskills.com/blogs/paul/how-to-tell-if-the-io-subsystem-is-causing-corruptions>  
Link: <http://technet.microsoft.com/en-au/library/cc917726.aspx>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: ?

#### Trace Flag: 3427

Function: Enables fix for issue when many consecutive transactions inserting data into temp table in SQL Server 2016 consume more CPU than in SQL Server 2014. Another change in SQL Server 2016 behavior that could impact tempdb-heavy workloads has to do with Common Criteria Compliance (CCC), also known as C2 auditing. We introduced functionality to allow for transaction-level auditing in CCC which can cause some additional overhead, particularly in workloads that do heavy inserts and updates in temp tables. Unfortunately, this overhead is incurred whether you have CCC enabled or not. In SQL Server 2016 you can enable trace flag 3427 to bypass this overhead starting with SP1 CU2. Starting in SQL Server 2017 CU4, we automatically bypass this code if CCC is disabled.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/3216543>  
Link: [TEMPDB – Files and Trace Flags and Updates](https://blogs.msdn.microsoft.com/sql_server_team/tempdb-files-and-trace-flags-and-updates-oh-my/)  
Scope: global only

#### Trace Flag: 3448

Function: Introduced in the KB to fix a race condition leading to a hung database in mirroring failover situations. “ This trace flag forces new connections to keep checking for database state every two seconds instead of waiting for a lock for infinite time. It helps ending the connection tasks faster as the mirroring reac hes the start of the recovery phase and releasing more worker threads to be used by database mirroring.”  
Link: <https://support.microsoft.com/help/2970421/>

#### Trace Flag: 3449

Function: If you enable global TF 3449 (and you are on SQL Server 2012 SP3 CU3 or later or SQL Server 2014 SP1 CU7 or later), you will get much better performance by avoiding a FlushCache call in a number of different common scenarios, such as backup database, backup transaction log, create database, add a file to a database, restore a transaction log, recover a database, shrink a database file, and a SQL Server “graceful” shutdown. Link: <https://support.microsoft.com/help/3158396/>  
Link: <https://blogs.msdn.microsoft.com/psssql/2017/06/29/sql-server-large-ram-and-db-checkpointing/>  
Link: [Hidden Performance & Manageability Improvements in SQL Server 2012 / 2014](https://sqlperformance.com/2018/01/sql-performance/hidden-performance-manageability-improvements-sql-server-2012-2014)  
Scope: global only

#### Trace Flag: 3459

Function: Disables parallel redo. Assume that you use an Always On availability group (AG) that contains heap tables. Starting in SQL Server 2016, parallel thread for redo operations is used in secondary replicas. In this case, heap tables redo operation may generate a runtime assert dump or the SQL Server may crash with an access violation error in some cases.  
**Note: This trace flag applies to SQL Server 2016 (13.x) and SQL Server 2017 (14.x).**  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/3200975/>  
Link: <https://support.microsoft.com/help/4101554/>  
Link: <https://support.microsoft.com/help/4339858/>  
Scope: global only

#### Trace Flag: 3468

Function: Disables [indirect checkpoints](https://docs.microsoft.com/en-us/sql/relational-databases/logs/database-checkpoints-sql-server?view=sql-server-2017#IndirectChkpt) on tempdb.  
**Note: This trace flag applies to SQL Server 2016 (13.x) SP1 CU5, SQL Server 2017 (14.x) CU1 and higher builds.**  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 3499

**Undocumented trace flag**  
Function: Provides a workaround for doing a rolling upgrade from SQL 2005 to SQL 2008 with a DB that has a full-text index  
Link: None

#### Trace Flag: 3502

Function: Writes info about checkpoints to error log.  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Link: <https://blogs.msdn.microsoft.com/joaol/2008/11/20/sql-server-checkpoint-problems/>  
Link: <http://www.sqlskills.com/blogs/paul/a-sql-server-dba-myth-a-day-1530-checkpoint-only-writes-pages-from-committed-transactions/>  
Scope: session only

#### Trace Flag: 3503

Function: Indicates whether the checkpoint at the end of automatic recovery was skipped for a database (this applies only to read-only databases)  
Link: <http://www.sql-server-performance.com/2002/traceflags/>

#### Trace Flag: 3504

Function: For internal testing. Will raise a bogus log-out-of-space condition from checkpoint  
Link: <https://blogs.msdn.microsoft.com/joaol/2008/11/20/sql-server-checkpoint-problems/>  
Link: <http://www.sqlskills.com/blogs/paul/a-sql-server-dba-myth-a-day-1530-checkpoint-only-writes-pages-from-committed-transactions/>

#### Trace Flag: 3505

Function: Disables automatic checkpoints. Setting trace flag 3505 may increase recovery time and can prevent log space reuse until the next checkpoint is issued. Make sure to issue manual checkpoints on all read/write databases at appropriate time intervals. "For high availability systems, such as clusters, Microsoft recommends that you do not change the recovery interval because it may affect data safety and availability."  
Link: <http://www.sqlskills.com/blogs/paul/benchmarking-1-tb-table-population-part-2-optimizing-log-block-io-size-and-how-log-io-works/>  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Scope: ?

#### Trace Flag: 3601

Function: Appears to disable CPU instruction prefetching. The Blog post to the right uses it, in concert with 3603, to enable SQL 2000 to run on a machine with a # of processors that is not a power of 2  
Link: <https://blogs.msdn.microsoft.com/sqlserverfaq/2009/05/27/info-sql-2000-msde-installation-will-fail-if-you-have-number-of-cpus-on-a-box-which-is-not-in-power-of-2>

#### Trace Flag: 3602

Function: Records all error and warning messages sent to the client  
Link: <https://support.microsoft.com/help/199037/>

#### Trace Flag: 3603

Function: Disables “Simultaneous Multithreading Processor check”. Used in concern with TF 3601 in the blog post to the right to enable SQL 2000 to run on a machine with a # of processors that is not a power of 2  
Link: <https://blogs.msdn.microsoft.com/sqlserverfaq/2009/05/27/info-sql-2000-msde-installation-will-fail-if-you-have-number-of-cpus-on-a-box-which-is-not-in-power-of-2>

#### Trace Flag: 3604

Function: Enables the output from a large number of trace flags and DBCC commands to be sent back to the client. The Connect issue notes that problems can occur when using 3604 with a query that executes across a linked server. [This CSS page](https://blogs.msdn.microsoft.com/psssql/2009/05/11/how-do-i-determine-which-dump-triggers-are-enabled/) points out that 3604 is necessary for DBCC DumpTrigger(‘display’)  
Link: <http://blogs.msdn.com/b/askjay/archive/2011/01/21/why-do-we-need-trace-flag-3604-for-dbcc-statements.aspx>  
Link: [Internals of the Seven SQL Server Sorts – Part 1](https://sqlperformance.com/2015/04/sql-plan/internals-of-the-seven-sql-server-sorts-part-1)  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/306380/trace-flag-issue-7300-3604>  
Link: [How to Find the Statistics Used to Compile an Execution Plan](http://sqlblog.com/blogs/paul_white/archive/2011/09/21/how-to-find-the-statistics-used-to-compile-an-paul_white)  
Link: [A Row Goal Riddle](https://orderbyselectnull.com/2018/03/30/a-row-goal-riddle/)  
Link: [Undocumented Trace Flags: Inside the Restore Process](https://blog.rdx.com/undocumented-trace-flags-inside-the-restore-process/)  
Link: [What’s CHECKDB doing in my database restore?](http://www.mikefal.net/2018/04/10/whats-checkdb-doing-in-my-database-restore/)  
Scope: session only

#### Trace Flag: 3605

Function: Sends a variety of types of information to the SQL Server error log instead of to the user console. Often referenced in KB and blog articles in the context of other trace flags (e.g. 3604).  
Link: <https://blogs.msdn.microsoft.com/askjay/2011/01/21/why-do-we-need-trace-flag-3604-for-dbcc-statements/>  
Link: [Undocumented Trace Flags: Inside the Restore Process](https://blog.rdx.com/undocumented-trace-flags-inside-the-restore-process/)  
Link: [What’s CHECKDB doing in my database restore?](http://www.mikefal.net/2018/04/10/whats-checkdb-doing-in-my-database-restore/)  
Scope: session only

#### Trace Flag: 3607

Function: Skip recovery on startup  
Link: <http://sqlkbs.blogspot.se/2008/01/trace-flag.html>  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/10/24/why-does-restoring-a-database-needs-tempdb/>

#### Trace Flag: 3608

Function: Prevents SQL Server from automatically starting and recovering any database except the master database. If activities that require tempdb are initiated, then model is recovered and tempdb is created. Other databases will be started and recovered when accessed. Some features, such as snapshot isolation and read committed snapshot, might not work. Use for Move System Databases and Move User Databases.  
**Note: Do not use during normal operation.**  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [Importance of Performing DBCC CHECKDB on all SQL Server Databases](https://www.mssqltips.com/sqlservertip/4581/importance-of-performing-dbcc-checkdb-on-all-sql-server-databases/)  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/10/24/why-does-restoring-a-database-needs-tempdb>  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-3608/>  
Scope: global only

#### Trace Flag: 3609

Function: Recovering all databases, but not clearing tempdb  
Link: <http://basitaalishan.com/2012/02/20/essential-trace-flags-for-recovery-debugging/>  
Link: [Importance of Performing DBCC CHECKDB on all SQL Server Databases](https://www.mssqltips.com/sqlservertip/4581/importance-of-performing-dbcc-checkdb-on-all-sql-server-databases/)  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/10/24/why-does-restoring-a-database-needs-tempdb>  
Scope: global only

#### Trace Flag: 3610

Function: SQL 9 - Divide by zero to result in NULL instead of error  
Link: None

#### Trace Flag: 3614

Function: Modifies the order of startup operations so that SQL Server can successfully start up even if many user connections are being attempted during SQL startup  
Link: None

#### Trace Flag: 3625

Function: Limits the amount of information returned to users who are not members of the sysadmin fixed server role, by masking the parameters of some error messages using '\*\*\*\*\*\*'. This can help prevent disclosure of sensitive information.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 3626

Function: Turns on tracking of the CPU data for the sysprocesses table.  
Link: None

#### Trace Flag: 3628

Function: CSS’s mysterious description: “Includes ‘other errors’ in the dump based on a severity.”  
Link: [Controlling SQL Server memory dumps](https://blogs.msdn.microsoft.com/psssql/2009/11/17/how-it-works-controlling-sql-server-memory-dumps)

#### Trace Flag: 3629

Function: CSS: A memory dump will “include messages marked to include with this trace flag enabled.”  
Link: [Controlling SQL Server memory dumps](https://blogs.msdn.microsoft.com/psssql/2009/11/17/how-it-works-controlling-sql-server-memory-dumps)

#### Trace Flag: 3635

Function: Print diagnostic information. Trace Flag 3635 Diagnostics are written to the console that started it. There are not written to the errorlog, even if 3605 is turned on.  
Link: None

#### Trace Flag: 3640

Function: Eliminates sending DONE\_IN\_PROC messages to client for each statement in stored procedure. This is similar to the session setting of SET NOCOUNT ON, but when set as a trace flag, every client session is handled this way.  
Link: <https://blogs.msdn.microsoft.com/selvar/2010/07/13/delete-operation-from-a-reporting-service-2005-report-manager-fails-with-internalcatalogexception-and-throws-watson-mini-dump>

#### Trace Flag: 3651

**Undocumented trace flag**  
Function: Can cause stack dumps. For one query, this resulted in a parallel plan significantly more expensive than the naturally occurring serial plan.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 3654

Function: Apparently increases info found in the sys.dm\_os\_memory\_allocations DMV (which appears to have replaced the DBCC MEMOBJLIST command) Bob Ward also discusses it in his PASS 2013 session, saying that it turns on tracing for all memory allocations done by “Memory Objects” (a specific SQLOS memory term). This flag will have a significant impact on system performance.  
Link: <https://blogs.msdn.microsoft.com/psssql/2012/11/12/how-can-reference-counting-be-a-leading-memory-scribbler-cause>  
Link: <https://blogs.msdn.microsoft.com/slavao/2005/08/30/talking-points-around-memory-manager-in-sql-server-2005>  
Link: <https://support.microsoft.com/help/2888658/>

#### Trace Flag: 3656

Function: Enables resolve of all call stacks in extended events  
Link: <http://sqlcat.com/sqlcat/b/msdnmirror/archive/2010/05/11/resolving-dtc-related-waits-and-tuning-scalability-of-dtc.aspx>  
Link: [Controlling SQL Server memory dumps](https://blogs.msdn.microsoft.com/psssql/2009/11/17/how-it-works-controlling-sql-server-memory-dumps)  
Link: <http://www.sqlskills.com/blogs/paul/determine-causes-particular-wait-type>

#### Trace Flag: 3659

Function: Enables logging all errors to error log during server startup  
Link: <http://spaghettidba.com/2011/05/20/trace-flag-3659/>  
Link: [Change SQL Server Collation – Back to Basics](http://jasonbrimhall.info/2018/04/12/change-sql-server-collation/)  
Scope: global only

#### Trace Flag: 3660

Function: W/o this flag, for DBs that have Auto\_Close=true and for DBs on Express Edition, DB recovery is normally deferred until first user access when SQL starts up. This TF forces DB recovery to always run (well, only for DBs that actually need recovery done) at SQL Server startup.  
Link: <https://blogs.msdn.microsoft.com/ialonso/2012/10/08/how-much-is-crash-recovery-parallelized-in-which-order-are-databases-recovered>

#### Trace Flag: 3663

Function: CSS: “By default [SQL Server] allows system cache involvement [with writing to the SQL Error log] to avoid some of the performance issues you might be suspecting, but you can force it to use FILE\_FLAG\_WRITE\_THROUGH” with TF 3663  
Link: <http://blogs.msdn.com/b/psssql/archive/2011/01/07/discussion-about-sql-server-i-o.aspx>

#### Trace Flag: 3688

Function: Removes messages to error log about traces started and stopped  
Link: <https://support.microsoft.com/help/922578>

#### Trace Flag: 3689

Function: Logs extended errors to errorlog when network disconnect occurs, turned off by default. Will dump out the socket error code this can sometimes give you a clue as to the root cause.  
Link: <https://support.microsoft.com/help/922578>

#### Trace Flag: 3701

Function: Unknown  
Link: [Upgrading an expired SQL Server 2016 Evaluation Edition](https://www.codykonior.com/2017/11/30/upgrading-an-expired-sql-server-2016-evaluation-edition/)

#### Trace Flag: 3801

Function: Prohibits use of USE DB statement  
Link: None

#### Trace Flag: 3861

Function: This flag allows the SQL Server DB startup code to move system tables to the primary filegroup. Introduced due to behavior in the SQL 2014 upgrade process, where system tables could be created in a secondary filegroup (if that FG was the default).  
Link: <https://support.microsoft.com/help/3003760/>

#### Trace Flag: 3913

Function: SQL 7/8 - SQL Server does not update the rowcnt column of the sysindexes system table until the transaction is committed. When turned on the optimizer gets row count information from in-memory metadata that is saved to sysindexes system table when the transaction commits.  
Link: None

#### Trace Flag: 3917

Function: According to Bob Ward’s PASS 2014 SQL Server IO talk, enables trace output (3605 is required) for the Eager Write functionality that is used with bulk logged operations (such as SELECT INTO)  
Link: None

#### Trace Flag: 3923

Function: Let SQL Server throw an exception to the application when the 3303 warning message is raised  
Link: <https://support.microsoft.com/help/3014867>

#### Trace Flag: 3924

Function: Enables a fix where “XA” transactions started within a JDBC-connected Java app are not closed properly and stay open indefinitely.  
Link: <https://support.microsoft.com/help/3145492/>

#### Trace Flag: 3940

Function: According to Bob Ward’s PASS 2014 SQL Server IO talk, forces the Eager Write functionality to throttle at 1024 outstanding eager writes.  
Link: None

#### Trace Flag: 4001

Function: Very verbose logging of each login attempt to the error log. Includes tons of information  
Link: None

#### Trace Flag: 4010

Function: Allows only shared memory connections to the SQL Server. Meaning, you will only be able to connect from the server machine itself. Client connections over TCP/IP or named pipes will not happen.  
Link: <https://blogs.msdn.microsoft.com/sqlserverfaq/2011/05/11/inf-hey-my-sql-server-service-is-not-starting-what-do-i-do>  
Link: <https://blogs.msdn.microsoft.com/psssql/2008/09/05/sql-server-2005-setup-fails-in-wow-x86-on-computer-with-more-than-32-cpus>  
Link: [Upgrading an expired SQL Server 2016 Evaluation Edition](https://www.codykonior.com/2017/11/30/upgrading-an-expired-sql-server-2016-evaluation-edition/)

#### Trace Flag: 4013

Function: Trace flag 4013 write entries in error log whenever a new connection established. These entries contain login name and SPID also.  
Link: <http://sqlkbs.blogspot.se/2008/01/trace-flag.html>  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-4013/>  
Scope: global or session

#### Trace Flag: 4020

Function: Boot without recover  
Link: None

#### Trace Flag: 4022

Function: Directs the SQL instance to ignore stored procedures that have been configured as “auto-start” procedures. Their auto-start configuration is not affected, so the next time the instance is started w/o this flag they will return to their normal behavior.  
Link: <https://blogs.msdn.microsoft.com/sqlserverfaq/2011/05/11/inf-hey-my-sql-server-service-is-not-starting-what-do-i-do/>  
Link: [Upgrading an expired SQL Server 2016 Evaluation Edition](https://www.codykonior.com/2017/11/30/upgrading-an-expired-sql-server-2016-evaluation-edition/)  
Link: [Change SQL Server Collation – Back to Basics](http://jasonbrimhall.info/2018/04/12/change-sql-server-collation/)  
Scope: global only

#### Trace Flag: 4029

Function: Logs extended errors to errorlog when network disconnect occurs, turned off by default. Will dump out the socket error code this can sometimes give you a clue as to the root cause.  
Link: <https://blogs.msdn.microsoft.com/sql_protocols/2005/12/19/vss-sql-server-does-not-exist-or-access-denied>

#### Trace Flag: 4030

Function: Prints both a byte and ASCII representation of the receive buffer. Used when you want to see what queries a client is sending to SQL Server. You can use this trace flag if you experience a protection violation and want to determine which statement caused it. Typically, you can set this flag globally or use SQL Server Enterprise Manager. You can also use [DBCC INPUTBUFFER](https://docs.microsoft.com/en-us/sql/t-sql/database-console-commands/dbcc-inputbuffer-transact-sql).  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-4030/>  
Scope: global only

#### Trace Flag: 4031

Function: Prints both a byte and ASCII representation of the send buffers (what SQL Server sends back to the client). You can also use [DBCC OUTPUTBUFFER](https://docs.microsoft.com/en-us/sql/t-sql/database-console-commands/dbcc-outputbuffer-transact-sql).  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-4031/>  
Scope: global only

#### Trace Flag: 4032

Function: Traces the SQL commands coming in from the client. When enabled with 3605 it will direct those all to the error log.  
Link: <https://support.microsoft.com/help/199037/>  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-4032/>  
Scope: global only

#### Trace Flag: 4044

Function: SA account can be unlocked by rebooting server with trace flag. If sa (or sso\_role) password is lost, add this to your RUN\_serverfile. This will generate new password when server started.  
Link: None

#### Trace Flag: 4052

Function: SQL 9+ Prints TDS packets sent to the client (output) to console. Startup only.  
Link: None

#### Trace Flag: 4055

Function: SQL 9+ Prints TDS packets received from the client to console. Startup only.  
Link: None

#### Trace Flag: 4101

Function: “FIX: Reorder outer joins with filter criteria before non-selective joins and outer joins” Enabling this flag may increase the chance that selective filter criteria on an OUTER JOIN will influence the OJ earlier in the plan, rather than the more typical behavior of INNER JOINs being prioritized before OJs. Note that 4101 is also required to enable KB942444.  
Link: <https://support.microsoft.com/help/318530/>

#### Trace Flag: 4102

Function: SQL 9 - Query performance is slow if the execution plan of the query contains semi join operators Typically, semi join operators are generated when the query contains the IN keyword or the EXISTS keyword. Enable flag 4102 and 4118 to overcome this.  
Link: <https://support.microsoft.com/help/946020/>

#### Trace Flag: 4103

Function: “FIX: A query may take longer to run in SQL Server 2005 SP1 than it takes to run in the original release version of SQL Server 2005 or in SQL Server 2000” Applies particularly to queries that contain subqueries with “many column joins”.  
Link: None

#### Trace Flag: 4104

Function: SQL 9 - Overestimating cardinality of JOIN operator. When additional join predicates are involved, this problem may increase the estimated cost of the JOIN operator to the point where the query optimizer chooses a different join order. When the query optimizer chooses a different join order, SQL 9 system performance may be slow.  
Link: None

#### Trace Flag: 4105

Function: “FIX: The SQL Server 2005 query optimizer may incorrectly estimate the cardinality for a query that has a predicate that contains an index union alternative”  
Link: None

#### Trace Flag: 4106

Function: “FIX: A query may take a long time to compile when the query contains several JOIN clauses against a SQL Server 2005 database”  
Link: None

#### Trace Flag: 4107

Function: SQL 9 - When you run a query that references a partitioned table, query performance may decrease  
Link: None

#### Trace Flag: 4108

Function: “FIX: The query performance is very slow when you use a fast forward-only cursor to run a query in SQL Server 2005”  
Link: None

#### Trace Flag: 4109

Function: “FIX: Error message when you run a query that uses a fast forward-only cursor in SQL Server 2005: "Query processor could not produce a query plan because of the hints defined in this query”  
Link: <https://support.microsoft.com/help/926773/>

#### Trace Flag: 4110

Function: “FIX: The query performance is slow when you run a query that uses a user-defined scalar function against an instance of SQL Server 2005”  
Link: None

#### Trace Flag: 4111

Function: Fixes a cardinality estimate issue with an unnamed builtin function. The KB article title shows that the issue was initially hit due to timeouts with the Replication Merge agent, but the problem was ultimately a poor query plan.  
Link: None

#### Trace Flag: 4112

Function: Enables a fix for a problem that occurs when a linked server from 2005 or 2008 targets SQL 2000: “This problem occurs because SQL Server 2005 generates an execution plan that has a remote query. SQL Server 2005 must execute the remote query against SQL Server 2000 to retrieve the required data. SQL Server 2000 cannot handle the remote query. Therefore, error 107 occurs in SQL Server 2000. Then, error 107 is propagated back to SQL Server 2005. Therefore, error 107 occurs in SQL Server 2005, and error 8180 occurs in SQL Server 2005.”  
Link: <https://support.microsoft.com/help/936223/>

#### Trace Flag: 4115

Function: “FIX: A query that you run by using a FORWARD\_ONLY cursor takes alonger time to run in Microsoft SQL Server 2005 than in SQL Server 2000 ”The fix apparently increases the likelihood that a certain type of cursor will use an index seek (as it did in SQL 2000) rather than regressing to a scan for each Fetch. The notes also contain some interesting info about sp\_cursoropen  
Link: None

#### Trace Flag: 4116

Function: SQL 9 - Query runs slowly when using joins between a local and a remote table  
Link: None

#### Trace Flag: 4117

Function: “FIX: A blocking issue occurs when you update rows in a table in SQL Server 2005...[A] problem occurs because the positioned update in a transaction performs a table scan on all involved tables. This behavior causes many update locks to be generated on many rows in the table. Additionally, SQL Server tries to add an update lock on the ow that has already been granted an exclusive lock by another transaction. Therefore, a blocking issue occurs.”  
Link: None

#### Trace Flag: 4119

Function: “FIX: The query performance is slower when you run the query in SQL Server 2005 than when you run the query in SQL Server 2000” The example given in the KB article indicates that selective LIKE predicates may not be considered fully when less-selective “comparison” (e.g. =, >, etc) predicates are done on the same parameter (or variable?) value as the LIKE predicate.  
Link: None

#### Trace Flag: 4120

Function: “FIX: Error message when you perform an update operation by using a cursor in SQL Server 2005: Transaction (Process ID ) was deadlocked on lock resources with another process and has been chosen as the deadlock victim” This issue is apparently due to deadlock issues related to upgrading a U lock to an X lock.  
Link: <https://support.microsoft.com/help/953948/>

#### Trace Flag: 4121

Function: SQL 9 - Query that involves an outer join operation runs very slowly. However, if you use the FORCE ORDER query hint in the query, the query runs much faster. Additionally, the execution plan of the query contains the following text in the Warnings column: NO JOIN PREDICATE.  
Link: None

#### Trace Flag: 4123

Function: Query that has many outer joins takes a long time to compile in SQL Server 2005  
Link: None

#### Trace Flag: 4124

Function: “FIX: A query performance issue occurs when you run a query against a column of the bigint data type in SQL Server 2005... If the All Density column [of SHOW\_STATISTICS] displays incorrect values of 1, you are encountering this problem”  
Link: None

#### Trace Flag: 4125

Function: SQL 9 - Query may take more time to finish if using an inner join to join a derived table that uses DISTINCT keyword  
Link: None

#### Trace Flag: 4126

Function: “FIX: The synchronization process is slow, and the CPU usage is high on the computer that is configured as the Distributor in SQL Server 2005” The problem manifested as a replication performance issue, but the following phrase found in the KB article indicates that it is a query processor issue: “the query that performs poorly shows that a join predicate is not pushed down to a Clustered Index Seek operator.”  
Link: <https://support.microsoft.com/help/959013/>

#### Trace Flag: 4127

Function: SQL 9 - Compilation time of some queries is very long in an x64-based version. Basically its more than execution time because more memory allocations are necessary in the compilation process.  
Link: Note

#### Trace Flag: 4128

Function: “FIX: When you update rows by using a cursor in SQL Server 2005, the update may take a long time to finish”  
Link: <https://support.microsoft.com/help/957872/>

#### Trace Flag: 4129

Function: “FIX: The values of the datetime column are not same for the rows that are copied when you copy data to a table by using the GETDATE() function in Microsoft SQL Server 2005”  
Link: None

#### Trace Flag: 4130

Function: XML performance fix  
Link: None

#### Trace Flag: 4131

Function: “FIX: When you run a query that contains a JOIN operation in SQL Server 2005 or SQL Server 2008, and the ON clause of the JOIN operator contains a LIKE predicate, the query runs slower than in SQL Server 2000”  
Link: None

#### Trace Flag: 4133

Function: “FIX: The size of the SQL Server 2005 error log file or of the SQL Server 2008 error log file grows very quickly when query notifications are created and destroyed in a high ratio” & “FIX: The restore operation takes a long time when you restore a database that has query notification enabled in SQL Server 2005 or in SQL Server 2008”  
Link: <https://support.microsoft.com/help/958006/>

#### Trace Flag: 4134

Function: Bugfix for error: parallel query returning different results every time The trace flag disables an optimization in the query optimizer. The optimization caused the issue described in the KB article when you try to insert into a table by selecting from the table itself. As turning on the trace flag could result in a perf degradation, you only should use it if you run into the issue described in the KB article.  
Link: <https://support.microsoft.com/help/2546901>  
Link: <http://sql-sasquatch.blogspot.se/2014/04/whaddayaknow-bout-sqlserver-trace-flag.html>  
Link: <https://social.msdn.microsoft.com/Forums/sqlserver/en-US/9ea718c2-e0e0-40cf-b12b-3269130448b7/trace-flag-4135-sql-server-2008?forum=sqldatabaseengine>

#### Trace Flag: 4135

Function: Bugfix for error inserting to temp table  
Link: <https://support.microsoft.com/help/960770>  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/541352/tempdb-errors-during-statistics-auto-update>

#### Trace Flag: 4136

Function: Disables parameter sniffing unless OPTION(RECOMPILE), WITH RECOMPILE or OPTIMIZE FOR value is used. To accomplish this at the database level, see [ALTER DATABASE SCOPED CONFIGURATION (Transact-SQL)](https://docs.microsoft.com/sql/t-sql/statements/alter-database-scoped-configuration-transact-sql). To accomplish this at the query level, add the OPTIMIZE FOR UNKNOWN query hint. Beginning with SQL Server 2016 SP1, to accomplish this at the query level, add the USE HINT query hint instead of using this trace flag. **Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: <http://blogs.msdn.com/b/axinthefield/archive/2010/11/04/sql-server-trace-flags-for-dynamics-ax.aspx>  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://kejser.org/trace-flag-4136-2/>  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-4136/>  
Link: <http://www.sqlservergeeks.com/sql-server-did-you-know-about-trace-flag-4136/>  
Scope: global or session or query

#### Trace Flag: 4137

Function: Causes SQL Server to generate a plan using minimum selectivity when estimating AND predicates for filters to account for correlation, under the query optimizer cardinality estimation model of SQL Server 2012 and earlier versions  
Beginning with SQL Server 2016 SP1, to accomplish this at the query level, add the USE HINT query hint instead of using this trace flag. **Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: <https://support.microsoft.com/help/2658214>  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session or query

#### Trace Flag: 4138

Function: Causes SQL Server to generate a plan that does not use row goal adjustments with queries that contain TOP, OPTION (FAST N), IN, or EXISTS keywords  
Beginning with SQL Server 2016 SP1, to accomplish this at the query level, add the USE HINT query hint instead of using this trace flag. **Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: <https://support.microsoft.com/help/2667211>  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://answers.sqlperformance.com/questions/1609/trying-to-figure-out-how-to-resolve-the-data-skew.html>  
Link: <http://dba.stackexchange.com/questions/55198/huge-slowdown-to-sql-server-query-on-adding-wildcard-or-top>  
Scope: global or session or query

#### Trace Flag: 4139

Function: Enable automatically generated quick statistics (histogram amendment) regardless of key column status. If trace flag 4139 is set, regardless of the leading statistics column status (ascending, descending, or stationary), the histogram used to estimate cardinality will be adjusted at query compile time  
Beginning with SQL Server 2016 SP1, to accomplish this at the query level, add the USE HINT query hint instead of using this trace flag. **Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: <https://support.microsoft.com/help/2952101>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [SQL Server - estimates outside of the histogram - half-baked draft](http://sql-sasquatch.blogspot.ru/2017/09/sql-server-estimates-outside-of.html)  
Link: [Parallelism in Hekaton (In-Memory OLTP)](http://www.nikoport.com/2018/01/20/parallelism-in-hekaton-in-memory-oltp/)  
Link: [Important Trace Flags That Every DBA Should Know](http://victorisakov.files.wordpress.com/2011/10/sql_pass_summit_2011-important_trace_flags_that_every_dba_should_know-victor_isakov.pdf)  
Link: <https://support.microsoft.com/help/974006>  
Scope: global or session or query

#### Trace Flag: 4199

Function: Enables query optimizer (QO) changes released in SQL Server Cumulative Updates and Service Packs. QO changes that are made to previous releases of SQL Server are enabled by default under the latest database compatibility level in a given product release, without trace flag 4199 enabled.  
The following table summarizes the behavior when using specific database compatibility levels and trace flag 4199:

| **Database compatibility level** | **TF 4199** | **QO changes from previous database compatibility levels** | **QO changes for current version post-RTM** |
| --- | --- | --- | --- |
| 100 to 120 | Off On | Disabled Enabled | Disabled Enabled |
| 130 | Off On | Enabled Enabled | Disabled Enabled |
| 140 | Off On | Enabled Enabled | Disabled Enabled |

To enable this at the database level, see [ALTER DATABASE SCOPED CONFIGURATION (Transact-SQL)](https://docs.microsoft.com/sql/t-sql/statements/alter-database-scoped-configuration-transact-sql).  
**Note: Starting with SQL Server 2016 SP1, to accomplish this at the query level, add the USE HINT** [**query hint**](https://docs.microsoft.com/en-us/sql/t-sql/queries/hints-transact-sql-query) **instead of using this trace flag.**  
Link: <https://support.microsoft.com/help/974006>  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/974006/>  
Link: <https://sqlworkbooks.com/2017/04/selectively-enabletrace-flag-4199-and-query_optimizer_hotfixes-in-sql-server-2016/>  
Link: <https://sqlworkbooks.com/2017/04/trace-flag-4199-no-per-session-override-if-you-enable-it-globally/>  
Link: <http://www.sqlservergeeks.com/sql-server-2016-database-scoped-configuration-and-trace-flag-4199/>  
Scope: global or session or query

#### Trace Flag: 4606

Function: Over comes SA password by startup. Disables password policy check during server startup.  
Link: <https://blogs.msdn.microsoft.com/sqlserverfaq/2011/05/11/inf-hey-my-sql-server-service-is-not-starting-what-do-i-do>  
Link: <https://blogs.msdn.microsoft.com/sqlserverfaq/2008/07/31/upgrade-of-sql-server-2000-instance-to-sql-server-2005-fails-with-error-similar-to-enforce-password-policy>

#### Trace Flag: 4610

Function: When you use trace flag 4618 together with trace flag 4610, the number of entries in the cache store is limited to 8,192. When the limit is reached, SQL 2005 removes some entries from the TokenAndPermUserStore cache store.  
Link: <https://support.microsoft.com/help/959823>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://blogs.msdn.microsoft.com/psssql/2008/06/16/query-performance-issues-associated-with-a-large-sized-security-cache/>  
Scope: global only

#### Trace Flag: 4612

Function: Disable the ring buffer logging - no new entries will be made into the ring buffer  
Link: <http://blogs.msdn.com/b/lcris/archive/2007/02/19/sql-server-2005-some-new-security-features-in-sp2.aspx>

#### Trace Flag: 4613

Function: Generate a minidump file whenever an entry is logged into the ring buffer  
Link: <http://blogs.msdn.com/b/lcris/archive/2007/02/19/sql-server-2005-some-new-security-features-in-sp2.aspx>

#### Trace Flag: 4614

Function: Enables SQL Server authenticated logins that use Windows domain password policy enforcement to log on to the instance even though the SQL Server service account is locked out or disabled on the Windows domain controller.  
Link: <https://support.microsoft.com/help/925744>

#### Trace Flag: 4616

Function: Makes server-level metadata visible to application roles. In SQL Server, an application role cannot access metadata outside its own database because application roles are not associated with a server-level principal. This is a change of behavior from earlier versions of SQL Server. Setting this global flag disables the new restrictions, and allows for application roles to access server-level metadata.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/906549/>  
Scope: global only

#### Trace Flag: 4618

Function: Limits number of entries per user cache store to 1024. It may incur a small CPU overhead as when removing old cache entries when new entries are inserted. It performs this action to limit the size of the cache store growth. However, the CPU overhead is spread over time. When used together with trace flag 4610 increases the number of entries in the TokenAndPermUserStore cache store to 8192  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/933564>  
Link: <https://support.microsoft.com/help/959823/>  
Scope: global only

#### Trace Flag: 4620

Function: According to the Connect item, causes permission checking to be done on a global cache instead of the per-user caches that were introduced in SQL 2008. The thread includes some interesting information on the cache stores, especially as they relate to TokenPermAndUserStore.  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/467661/sql-server-2008-has-incorrect-cache-names-in-sys-dm-os-memory-cache-counters>

#### Trace Flag: 4621

Function: SQL 9 – After 4610 & 4618 you can still customize the quota for TokenAndPermUserStore cache store that is based on the current workload  
Link: <https://support.microsoft.com/help/959823>

#### Trace Flag: 5004

Function: Pauses TDE encryption scan and causes encryption scan worker to exit without doing any work. The database will continue to be in encrypting state (encryption in progress). To resume re-encryption scan, disable trace flag 5004 and run ALTER DATABASE SET ENCRYPTION ON.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 5101

Function: Forces all I/O requests to go through engine 0. This removes the contention between processors but could create a bottleneck if engine 0 becomes busy with non-I/O tasks.  
Link: <http://dba.fyicenter.com/Interview-Questions/SYBASE/What_is_Trace_Flag_Definitions_in_Sybase.html#1.3.4#1.3.4>

#### Trace Flag: 5102

Function: Prevents engine 0 from running any non-affinitied tasks.  
Link: <http://dba.fyicenter.com/Interview-Questions/SYBASE/What_is_Trace_Flag_Definitions_in_Sybase.html#1.3.4#1.3.4>

#### Trace Flag: 5302

Function: Alters default behavior of select…INTO (and other processes) that lock system tables for the duration of the transaction. This trace flag disables such locking during an implicit transaction.  
Link: <https://support.microsoft.com/help/153096/>

#### Trace Flag: 6498

Function: Enables more than one large query compilation to gain access to the big gateway when there is sufficient memory available. It is based on the 80 percentage of SQL Server Target Memory, and it allows for one large query compilation per 25 gigabytes (GB) of memory.  
**Note: Beginning with SQL Server 2014 SP2 and SQL Server 2016 this behavior is controlled by the engine and trace flag 6498 has no effect.**  
Link: <https://support.microsoft.com/help/3024815>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://blogs.msdn.com/b/sql_server_team/archive/2015/10/09/query-compile-big-gateway-policy-changes-in-sql-server.aspx>  
Scope: global only

#### Trace Flag: 6527

Function: Disables generation of a memory dump on the first occurrence of an out-of-memory exception in CLR integration. By default, SQL Server generates a small memory dump on the first occurrence of an out-of-memory exception in the CLR. The behaviour of the trace flag is as follows: If this is used as a startup trace flag, a memory dump is never generated. However, a memory dump may be generated if other trace flags are used. If this trace flag is enabled on a running server, a memory dump will not be automatically generated from that point on. However, if a memory dump has already been generated due to an out-of-memory exception in the CLR, this trace flag will have no effect.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 6530

Function: Enables a hotfix for “ FIX: Slow performance in SQL Server 2012 or SQL Server 2014 when you build an index on a spatial data type of a large table”  
Link: <https://blogs.msdn.microsoft.com/psssql/2013/11/19/spatial-indexing-from-4-days-to-4-hours>  
Link: <https://support.microsoft.com/help/2896720/>

#### Trace Flag: 6531

Function: Enables adjustment in the SQLOS scheduling layer to handle queries that issue many short-duration calls to spatial data (which is implemented via CLR functions): “ This fix introduces the trace flag 6531 to indicate to the SQLOS hosting layer that the spatial data type should avoid preemptive protections. This can reduce the CPU consumption and improve the overall performance for spatial activities. Only use this trace flag if the individual, spatial method invocations (per row and column) take less than ~4ms. Longer invocations without preemptive protection could lead to scheduler concurrency issues and SQLCLR punishment messages logged to the error log.”  
Link: <https://support.microsoft.com/help/3005300/>

#### Trace Flag: 6532

Function: Enables performance improvement of query operations with spatial data types in SQL Server 2012 and SQL Server 2014. The performance gain will vary, depending on the configuration, the types of queries, and the objects.  
Link: [KB3107399](https://support.microsoft.com/help/3107399)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 6533

Function: Enables performance improvement of query operations with spatial data types in SQL Server 2012 and SQL Server 2014. The performance gain will vary, depending on the configuration, the types of queries, and the objects.  
Link: [KB3107399](https://support.microsoft.com/help/3107399)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 6534

Function: Enables performance improvement of query operations with spatial data types in SQL Server 2012, SQL Server 2014 and SQL Server 2016. The performance gain will vary, depending on the configuration, the types of queries, and the objects.  
Link: <https://support.microsoft.com/help/3054180>  
Link: [KB3107399](https://support.microsoft.com/help/3107399)  
Link: <https://blogs.msdn.microsoft.com/bobsql/2016/06/03/sql-2016-it-just-runs-faster-native-spatial-implementations/>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 6545

Function: Enables "CLR strict security" behavior (introduced in SQL Server 2017) in SQL Server 2012, SQL Server 2014, and SQL Server 2016. When enabled, this option will require that all assemblies, regardless of PERMISSION\_SET, be signed, have an associated signature-based Login, and that the associated Login be granted the UNSAFE ASSEMBLY permission.  
Please note:

1. This TF can only be specified as a startup parameter!
2. This TF is only available in instances that have been updated / patched with a Service Pack (SP), Cumulative Update (CU), or GDR that was released on or after 2017-08-08.

Link: [SQLCLR vs. SQL Server 2012 & 2014 & 2016, Part 7: “CLR strict security” – The Problem Continues … in the Past (Wait, What?!?)](https://SqlQuantumLeap.com/2018/02/23/sqlclr-vs-sql-server-2012-2014-2016-part-7-clr-strict-security-the-problem-continues-in-the-past-wait-what/)  
Link: [Update adds the "CLR strict security" feature to SQL Server 2016](https://support.microsoft.com/help/4018930/) ( KB4018930 )  
Scope: global

#### Trace Flag: 7103

**Undocumented trace flag**  
Function: Disable table lock promotion for text columns  
Link: None

#### Trace Flag: 7300

Function: Outputs extra info about linked server errors  
Link: <https://support.microsoft.com/help/314530>  
Link: <https://support.microsoft.com/help/280106/>  
Link: <https://support.microsoft.com/help/280102/>  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/306380/trace-flag-issue-7300-3604>

#### Trace Flag: 7301

Function: Fixes a problem in SQL 6.5 where SELECT INTO queries with text/image types were not bulk-logged.  
Link: None

#### Trace Flag: 7311

Function: Offers a new alternative to handling the tricky problem of converting Oracle NUMBER types (across OLEDB linked server queries) with unknown precision/scale to a valid SQL Server data type, by treating all such types as NUMERIC(38,10).  
Link: <https://support.microsoft.com/help/3051993/>

#### Trace Flag: 7314

Function: Forces NUMBER values with unknown precision/scale to be treated as double values with OLE DB provider  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/3051993>  
Scope: global or session

#### Trace Flag: 7352

Function: Show the optimizer output and the post-optimization rewrite in action (After Post Optimization Rewrite)  
Link: [Internals of the Seven SQL Server Sorts – Part 1](https://sqlperformance.com/2015/04/sql-plan/internals-of-the-seven-sql-server-sorts-part-1)  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/08/31/sql-server-internals-nested-loops-prefetching.aspx>  
Link: <http://www.queryprocessor.com/batch-sort-and-nested-loops>  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)  
Link: [Few Outer Rows Optimization](https://www.sqlshack.com/few-outer-rows-optimization/)  
Related to [8607](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#8607) trace flag  
Scope: session only

#### Trace Flag: 7356

**Undocumented trace flag**  
Function: Added a probe residual to an adaptive join. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 7357

Function: Outputs info re: hashing operators, including role reversal, recursion levels, whether the Unique Hash optimization could be used, info about the hash-related bitmap, etc. Dima’s article is a must-read. For parallel query plans, 7357 does NOT send output to the console window. However, output to the SQL Server error log can be enabled by enabling 3605.  
Link: <http://www.queryprocessor.com/hash-join-execution-internals>  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)

#### Trace Flag: 7359

Function: Disables the bitmap associated with hash matching. This bitmap is used for “bit-vector filtering” and can reduce the amount of data written to TempDB during hash spills.  
Link: [www.queryprocessor.com/hash-join-execution-internals](http://www.queryprocessor.com/hash-join-execution-internals)

#### Trace Flag: 7398

**Undocumented trace flag**  
Function: Changed a nested loop join to have ordered prefetch.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 7470

Function: Fixes a problem where under certain (unknown) conditions, a sort spill occurs for large sorts  
Link: <https://support.microsoft.com/help/3088480/>

#### Trace Flag: 7412

Function: Enables the lightweight query execution statistics profiling infrastructure. unless your server is already CPU bound, like you’re running all the time with 95% CPU, unless you are at that point, turn on this trace flag at any server you have. This would be my advice here because this enables that lightweight profiling infrastructure there and then you’ll see in a few minutes what it unleashes here. So one thing that happens when I enable the lightweight profiling is that the sys.dm\_exec\_query\_profiles DMV, which is something that actually populates the live query stats ability or feature of SSMS, now also is also populated with this lightweight profiling, which means that for all essence, we are now able to run a live query stats on all fashions at any given point in time, and this is extremely useful for let’s say a production DBA that someone calls and says, “Hey, you have a problem. To tap into running system and look at what it’s doing.”  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/3170113>  
Link: <https://www.brentozar.com/archive/2017/10/get-live-query-plans-sp_blitzwho/>  
Link: <https://groupby.org/conference-session-abstracts/enhancements-that-will-make-your-sql-database-engine-roar-2016-sp1-edition/>  
Link: <https://www.scarydba.com/2018/06/11/plan-metrics-without-the-plan-trace-flag-7412/>  
Scope: global only

#### Trace Flag: 7470

Function: Fix for sort operator spills to tempdb in SQL Server 2012 or SQL Server 2014 when estimated number of rows and row size are correct  
Link: <https://support.microsoft.com/help/3088480>

#### Trace Flag: 7471

Function: Running multiple UPDATE STATISTICS for different statistics on a single table concurrently  
Link: <https://support.microsoft.com/help/3156157>  
Link: <http://sqlperformance.com/2016/05/sql-performance/parallel-rebuilds>

#### Trace Flag: 7497

Function: Behavior and intended purpose unknown, but in this post Paul White uses it in concert with 7498 to disable “optimized bitmaps”.  
Link: <https://sqlperformance.com/2015/11/sql-plan/hash-joins-on-nullable-columns>

#### Trace Flag: 7498

Function: Behavior and intended purpose unknown, but in this post Paul White uses it in concert with 7497 to disable “optimized bitmaps”.  
Link: <https://sqlperformance.com/2015/11/sql-plan/hash-joins-on-nullable-columns>

#### Trace Flag: 7501

Function: Dynamic cursors are used by default on forward-only cursors. Dynamic cursors are faster than in earlier versions and no longer require unique indexes. This flag disables the dynamic cursor enhancements and reverts to version 6.0 behavior.  
Link: None

#### Trace Flag: 7502

Function: Disable cursor plan caching for extended stored procedures  
Link: <http://basitaalishan.com/2012/02/20/essential-trace-flags-for-recovery-debugging/>

#### Trace Flag: 7505

Function: Enables version 6.x handling of return codes when calling dbcursorfetchex and the resulting cursor position follows the end of the cursor result set  
Link: None

#### Trace Flag: 7525

Function: SQL 8 - Reverts to ver 7 behavior of closing nonstatic cursors regardless of the SET CURSOR\_CLOSE\_ON\_COMMIT state  
Link: None

#### Trace Flag: 7601, 7603, 7604, 7605

Function: Helps in gathering more information in full text search by turning on full text tracing which gathers information on indexing process using the error log. Also 7603, 7604, 7605 trace flags.  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/526343/looking-for-documentation-on-trace-flags-7601-7603-7604-and-7605>

#### Trace Flag: 7608

Function: Performance fix for slow full text population with a composite clustered index  
Link: None

#### Trace Flag: 7613

Function: SQL 9 - Search results are missing when performing a full-text search operation on Win SharePoint Services 2.0 site after upgrading  
Link: None

#### Trace Flag: 7614

Function: SQL 9 - Full-text index population for the indexed view is very slow  
Link: None

#### Trace Flag: 7646

Function: SQL 10 - Avoids blocking when using full text indexing. An issue we experienced that full text can be slow when there is a high number of updates to the index and is caused by blocking on the docidfilter internal table.  
Link: None

#### Trace Flag: 7745

Function: Forces Query Store to not flush data to disk on database shutdown.  
Note: Using this trace may cause Query Store data not previously flushed to disk to be lost in case of shutdown. For a SQL Server shutdown, the command SHUTDOWN WITH NOWAIT can be used instead of this trace flag to force an immediate shutdown.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [Query Store Trace Flags](https://www.sqlskills.com/blogs/erin/query-store-trace-flags/)  
Scope: global only

#### Trace Flag: 7752

Function: Enables asynchronous load of Query Store.  
Note: Use this trace flag if SQL Server is experiencing high number of QDS\_LOADDB waits related to Query Store synchronous load (default behavior).  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [Query Store Trace Flags](https://www.sqlskills.com/blogs/erin/query-store-trace-flags/)  
Scope: global only

#### Trace Flag: 7806

Function: Enables a dedicated administrator connection ([DAC](https://docs.microsoft.com/sql/database-engine/configure-windows/diagnostic-connection-for-database-administrators)) on SQL Server Express. By default, no [DAC](https://docs.microsoft.com/sql/database-engine/configure-windows/diagnostic-connection-for-database-administrators) resources are reserved on SQL Server Express.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://msdn.microsoft.com/en-us/library/ms189595.aspx>  
Link: <https://sqlperformance.com/2012/08/sql-memory/test-your-dac-connection>  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-7806/>  
Scope: global only

#### Trace Flag: 7826

Function: Disable Connectivity ring buffer  
Link: <http://blogs.msdn.com/b/sql_protocols/archive/2008/05/20/connectivity-troubleshooting-in-sql-server-2008-with-the-connectivity-ring-buffer.aspx>

#### Trace Flag: 7827

Function: Record connection closure info in ring buffer  
Link: <http://blogs.msdn.com/b/sql_protocols/archive/2008/05/20/connectivity-troubleshooting-in-sql-server-2008-with-the-connectivity-ring-buffer.aspx>  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/518158/-packet-error-a-fatal-error-occurred-while-reading-the-input-stream-from-the-network>

#### Trace Flag: 7833

Function: SQL 2012 SP2 CU8 introduced a fix for a “silent error” condition in the sqlcmd tool. The CU also included this flag to allow customers to revert to pre-CU fix behavior.  
Link: <https://support.microsoft.com/help/3082877/>

#### Trace Flag: 8001

Function: Khen2005, p2: “For SQL Server 2005, the SQL Server product team opted not to include some wait types that fall under one of the following three categories:

* Wait types that are never used in SQL Server 2005; note that some wait types not excluded are also never used.
* Wait types that can occur only at times when they do not affect user activity, such as during initial server startup and shutdown, and are not visible to users.
* Wait types that are innocuous but have caused concern among users because of their high occurrence or duration The complete list of wait types is available by enabling trace flag 8001. The only effect of this trace flag is to force sys.dm\_os\_wait\_stats to display all wait types.”  
  Link: None

#### Trace Flag: 8002

Function: Changes CPU Affinity behaviour  
Link: <https://blogs.msdn.microsoft.com/psssql/2011/11/11/sql-server-clarifying-the-numa-configuration-information>

#### Trace Flag: 8004

Function: SQL server to create a mini-dump once you enable 2551 and a out of memory condition is hit  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/342691/not-enough-memory-was-available-for-trace-error-when-attempting-to-profile-sql-2008>

#### Trace Flag: 8008

**Undocumented trace flag**  
Function: Force the scheduler hint to be ignored. Always assign to the scheduler with the least load (pool based on SQL 2012 EE SKU or Load Factor for previous versions and SKUs.)  
Link: [How It Works: SQL Server 2012 Database Engine Task Scheduling](https://blogs.msdn.microsoft.com/psssql/2013/08/13/how-it-works-sql-server-2012-database-engine-task-scheduling/)  
Link: <https://blogs.msdn.microsoft.com/psssql/2013/08/13/how-it-works-sql-server-2012-database-engine-task-scheduling>  
Link: <http://www.stillhq.com/sqldownunder/archives/msg05089.html>

#### Trace Flag: 8009

**Undocumented trace flag**  
Function: Enables the “idle state behavior” (see IO Basics, Chapter 2 document) that a SQL instance can enter under certain conditions.  
Link: <https://technet.microsoft.com/en-us/library/cc917726.aspx>

#### Trace Flag: 8010

Function: Disables the “idle state” behavior that a SQL instance can enter (see TF 8009). Fixes problem that SQL Server services can not be stopped  
Link: <https://support.microsoft.com/help/2633271> Link: <https://technet.microsoft.com/en-us/library/cc917726.aspx>

#### Trace Flag: 8011

Function: Trace flag 8011 disables the collection of additional diagnostic information for Resource Monitor. You can use the information in this ring buffer to diagnose out-of-memory conditions. Trace flag 8011 always applies across the server and has global scope. You can turn on trace flag 8011 at startup or in a user session.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlservergeeks.com/sql-server-ring-buffer-trace-flag-8011/>  
Scope: global only

#### Trace Flag: 8012

Function: Disable the ring buffer for schedulers. SQL Server records an event in the schedule ring buffer every time that one of the following events occurs: a scheduler switches context to another worker, a worker is suspended, a worker is resumed, a worker enters the preemptive mode or the non-preemptive mode. You can use the diagnostic information in this ring buffer to analyze scheduling problems. For example, you can use the information in this ring buffer to troubleshoot problems when SQL Server stops responding. Trace flag 8012 disables recording of events for schedulers. You can turn on trace flag 8012 only at startup. The exception ring buffer records the last 256 exceptions that are raised on a node. Each record contains some information about the error and contains a stack trace. A record is added to the ring buffer when an exception is raised.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlservergeeks.com/sql-server-ring-buffer-trace-flag-8012/>  
Scope: global only

#### Trace Flag: 8015

Function: Disable auto-detection and NUMA setup  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://sql-sasquatch.blogspot.se/2013/04/startup-trace-flags-i-love.html>  
Link: [Upgrading an expired SQL Server 2016 Evaluation Edition](https://www.codykonior.com/2017/11/30/upgrading-an-expired-sql-server-2016-evaluation-edition/)  
Scope: global only

#### Trace Flag: 8016

**Undocumented trace flag**  
Function: Force load balancing to be ignored. Always assign to the preferred scheduler.  
Link: [How It Works: SQL Server 2012 Database Engine Task Scheduling](https://blogs.msdn.microsoft.com/psssql/2013/08/13/how-it-works-sql-server-2012-database-engine-task-scheduling/)

#### Trace Flag: 8017

Function: Upgrade version conflict  
Link: <http://social.msdn.microsoft.com/Forums/eu/sqlexpress/thread/dd6fdc16-9d8d-4186-9549-85ba4c322d10>  
Link: <http://connect.microsoft.com/SQLServer/feedback/details/407692/indicateur-de-trace-8017-reported-while-upgrading-from-ssee2005-to-ssee2008>  
Link: <http://dba.stackexchange.com/questions/48580/trace-flag-and-which-need-to-be-turned-off-and-why>

#### Trace Flag: 8018

Function: Disables the creation of the ring buffer, and no exception information is recorded. Disabling the exception ring buffer makes it more difficult to diagnose problems that are related to internal server errors.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlservergeeks.com/sql-server-ring-buffer-trace-flag-8018/>  
Link: <http://www.sqlservergeeks.com/sql-server-ring-buffer-trace-flag-8019/>  
Scope: global only

#### Trace Flag: 8019

Function: Disable stack collection for the exception ring buffer Disables stack collection during the record creation. Trace flag 8019 has no effect if trace flag [8018](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#8018) is turned on.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlservergeeks.com/sql-server-ring-buffer-trace-flag-8019/>  
Scope: global only

#### Trace Flag: 8020

Function: Disable working set monitoring. SQL Server uses the size of the working set when SQL Server interprets the global memory state signals from the operating system. Trace flag 8020 removes the size of the working set from consideration when SQL Server interprets the global memory state signals. If you use this trace flag incorrectly, heavy paging occurs, and the performance is poor. Therefore, contact Microsoft Support before you turn on trace flag 8020.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 8021

Function: On some lower end hardware we used to get reported that each CPU has its own NUMA node. This was usually incorrect and when we detected only a single CPU per NODE we would assume NO NUMA. Trace flag 8021 disables this override.  
Link: <https://blogs.msdn.microsoft.com/psssql/2011/11/11/sql-server-clarifying-the-numa-configuration-information/>

#### Trace Flag: 8022

Function: This flag gives more information about the conditions when a non-yielding scheduler/situation was encountered. The whitepaper linked to on the right gives example output for this flag  
Link: None

#### Trace Flag: 8024

Function: When this TF is on, it affects the mini-dump generation logic for the 1788\* errors: "To capture a mini-dump, one of the following checks must also be met.

1. The non-yielding workers CPU utilization must be > 40 percent.
2. The SQL Server process is not starved for overall CPU resource utilization. Additional check #1 is targeted at runaway CPU users. Additional check #2 is targeted at workers with lower utilizations that are probably stuck in an API call or similar activity."  
   Link: [How To Diagnose and Correct Errors 17883, 17884, 17887, and 17888](https://msdn.microsoft.com/library/cc917684.aspx)

#### Trace Flag: 8025

Function: SQL on NUMA normally does most of its allocation on Node 1, because usually Windows and other programs will allocate from Node 0. However, if you want SQL to do its resource allocation on the default node (node 0), turn on this flag.  
Link: <https://blogs.msdn.microsoft.com/psssql/2011/11/11/sql-server-clarifying-the-numa-configuration-information>

#### Trace Flag: 8026

Function: SQL Server will clear a dump trigger after generating the dump once  
Link: [KB917825](https://support.microsoft.com/help/917825/)  
Link: [Controlling SQL Server memory dumps](https://blogs.msdn.microsoft.com/psssql/2009/11/17/how-it-works-controlling-sql-server-memory-dumps)

#### Trace Flag: 8030

Function: Fix for performance bug  
Link: <https://support.microsoft.com/help/917035>

#### Trace Flag: 8032

Function: Alters cache limit settings  
**Warning: Trace flag 8032 can cause poor performance if large caches make less memory available for other memory consumers, such as the buffer pool.**  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 8033

Function: Suppresses messages of the form “The time stamp counter of CPU on scheduler id 1 is not synchronized with other CPUs” from being placed in the SQL Error log when CPU drift is noticed  
**Warning: SQL 9 - Disable the reporting of CPU Drift errors in the SQL Server error log like time stamp counter of CPU on scheduler id 1 is not synchronized with other CPUs.**  
Link: <https://support.microsoft.com/help/931279/>  
Link: <https://blogs.msdn.microsoft.com/psssql/2007/08/19/sql-server-2005-rdtsc-truths-and-myths-discussed>

#### Trace Flag: 8038

Function: Will drastically reduce the number of context switches when running SQL 2005 or 2008  
Link: [KB972767](https://support.microsoft.com/help/972767)  
Link: <http://forum.proxmox.com/threads/15844-Win7-x64-guest-with-SQLServer-2012-High-CPU-usage>  
Link: [TECHNET List Of SQL Server Trace Flags](http://social.technet.microsoft.com/wiki/contents/articles/13105.trace-flags-in-sql-server.aspx)

#### Trace Flag: 8040

Function: Disables Resource Governor  
Link: <http://www.sqlservergeeks.com/blogs/AmitBansal/sql-server-bi/64/sql-server-disabling-resource-governor-permanently-somewhat>

#### Trace Flag: 8048

**Note: Beginning with SQL Server 2014 SP2 and SQL Server 2016 this behavior is controlled by the engine and trace flag 8048 has no effect.**  
Function: Converts NUMA partitioned memory objects into CPU partitioned  
Link: <http://sql-sasquatch.blogspot.se/2013/04/startup-trace-flags-i-love.html>  
Link: <https://support.microsoft.com/help/2809338>  
Link: <http://blogs.msdn.com/b/psssql/archive/2012/12/20/how-it-works-cmemthread-and-debugging-them.aspx>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://blogs.msdn.com/b/psssql/archive/2011/09/01/sql-server-2008-2008-r2-on-newer-machines-with-more-than-8-cpus-presented-per-numa-node-may-need-trace-flag-8048.aspx>  
Link: [Hidden Performance & Manageability Improvements in SQL Server 2012 / 2014](https://sqlperformance.com/2018/01/sql-performance/hidden-performance-manageability-improvements-sql-server-2012-2014)  
Related to: [8015](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#8015), [9024](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#9024)  
Scope: global only

#### Trace Flag: 8049

Function: SQL 9+ Startup only – Allows use of 1ms times even when patched. Check 8038 for details.  
Link: [KB972767](https://support.microsoft.com/help/972767)  
Link: <https://blogs.msdn.microsoft.com/psssql/2010/08/18/how-it-works-timer-outputs-in-sql-server-2008-r2-invariant-tsc>

#### Trace Flag: 8050

Function: Causes "optional" wait types (see the CSS article) to be excluded when querying sys.dm\_os\_wait\_stats  
Link: <https://blogs.msdn.microsoft.com/psssql/2009/11/02/the-sql-server-wait-type-repository/>

#### Trace Flag: 8075

Function: Enables a fix (after applying the appropriate CU) for x64 VAS exhaustion.  
Link: <https://support.microsoft.com/help/3074434/>

#### Trace Flag: 8079

Function: Allows SQL Server 2014 SP2 to interrogate the hardware layout and automatically configure Soft-NUMA on systems reporting 8 or more CPUs per NUMA node. The automatic Soft-NUMA behavior is Hyperthread (HT/logical processor) aware. The partitioning and creation of additional nodes scales background processing by increasing the number of listeners, scaling and network and encryption capabilities. When Trace Flag 8079 is enabled during startup, SQL Server 2012 SP4 will interrogate the hardware layout and automatically configure Soft NUMA on systems reporting 8 or more CPUs per NUMA node. It is recommended to first test the performance of workload with Auto-Soft NUMA before it is turned ON in production. **Note: This trace flag applies to SQL Server 2014 SP2 and SQL Server 2012 SP4. Beginning with SQL Server 2016 this behavior is controlled by the engine and trace flag 8048 has no effect.**  
Link: [KB972767](https://support.microsoft.com/help/972767)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://blogs.msdn.microsoft.com/sqlreleaseservices/sql-server-2012-service-pack-4-sp4-released/>  
Link: [Hidden Performance & Manageability Improvements in SQL Server 2012 / 2014](https://sqlperformance.com/2018/01/sql-performance/hidden-performance-manageability-improvements-sql-server-2012-2014)  
Scope: global only

#### Trace Flag: 8202

Function: Used to replicate UPDATE as DELETE/INSERT pair at the publisher. i.e. UPDATE commands at the publisher can be run as an "on-page DELETE/INSERT" or a "full DELETE/INSERT". If the UPDATE command is run as an "on-page DELETE/INSERT," the Logreader send UDPATE command to the subscriber, If the UPDATE command is run as a "full DELETE/INSERT," the Logreader send UPDATE as DELETE/INSERT Pair. If you turn on trace flag 8202, then UPDATE commands at the publisher will be always send to the subscriber as DELETE/INSERT pair.  
Link: None

#### Trace Flag: 8203

Function: Display statement and transaction locks on a deadlock error  
Link: None

#### Trace Flag: 8206

Function: SQL 8 - Supports stored procedure execution with a user specified owner name for SQL Server subscribers or without owner qualification for heterogeneous subscribers  
Link: None

#### Trace Flag: 8207

Function: Enables singleton updates for Transactional Replication. Updates to subscribers can be replicated as a DELETE and INSERT pair. This might not meet business rules, such as firing an UPDATE trigger. With trace flag 8207 an update to a unique column that affects only one row (a singleton update) is replicated as an UPDATE and not as a DELETE or INSERT pair. If the update affects a column on which has a unique constraint or if the update affects multiple rows, the update is still replicated as a DELETE or INSERT pair.  
Link: <https://blogs.msdn.microsoft.com/psssql/2009/11/02/the-sql-server-wait-type-repository/>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only

#### Trace Flag: 8209

Function: Output extra information to error log regarding replication of schema changes in SQL Server Replication  
Link: None

#### Trace Flag: 8218

Function: Determine whether trace flag to bypass proc generation has been set. Referenced in the system procedure [master].[sys].[sp\_cdc\_vupgrade]  
Link: None

#### Trace Flag: 8295

Function: Creates a secondary index on the identifying columns on the change tracking side table at enable time  
Link: <https://social.msdn.microsoft.com/forums/sqlserver/en-US/00250311-7991-47b0-b788-7fae2e102254/trace-flag-8295>  
Link: <https://support.microsoft.com/help/2476322/>  
Link: <https://www.brentozar.com/archive/2014/06/performance-tuning-sql-server-change-tracking>  
Link: <https://blogs.technet.microsoft.com/smartinez/2013/03/06/sql-server-for-configmgr-2012-ebook-and-top-10-database-issues>  
Thanks to: Wilfred van Dijk

#### Trace Flag: 8446

Function: Databases in SQL 8 do not have a Service Broker ID. If you restore these databases on SQL 9 by using the WITH NORECOVERY option, these databases will not be upgraded causing mirroring & log-shipping configurations to fail.  
Link: <https://support.microsoft.com/help/959008>

#### Trace Flag: 8501

Function: Writes detailed information about Ms-DTC context & state changes to the log  
Link: None

#### Trace Flag: 8599

Function: Allows you to use a save-point within a distributed transaction  
Link: None

#### Trace Flag: 8602

Function: This trace flag is used to ignore all the index hints specified in query or stored procedure. We can use this trace flag to troubleshooting the query performance without changing index hints.  
Link: <http://download.microsoft.com/download/6/e/5/6e52bf39-0519-42b7-b806-c32905f4a066/eim_perf_flowchart_final.pdf>  
Link: <http://sqlblog.com/blogs/kalen_delaney/archive/2008/02/26/lost-without-a-trace.aspx>  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-8602/>  
Scope: global only  
Demo: <https://github.com/ktaranov/sqlserver-kit/blob/master/Scripts/Trace_Flag/Trace_Flag_8602.sql>

#### Trace Flag: 8605

**Undocumented trace flag**  
Function: Displays logical and physical trees used during the optimization process  
Link: [More Undocumented Query Optimizer Trace Flags](http://www.benjaminnevarez.com/2012/04/more-undocumented-query-optimizer-trace-flags/)  
Link: [Yet another X-Ray for the QP](http://www.queryprocessor.com/tf_8628/)  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)

#### Trace Flag: 8606

**Undocumented trace flag**  
Function: Show LogOp Trees  
Link: [Cardinality Estimation Framework 2014 First Look](http://www.somewheresomehow.ru/cardinality-estimation-framework-2014-first-look/)  
Link: [Yet another X-Ray for the QP](http://www.queryprocessor.com/tf_8628/)  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)

#### Trace Flag: 8607

Function: Displays the optimization output tree during the optimization process (Before Post Optimization Rewrite).  
Link: [Internals of the Seven SQL Server Sorts – Part 1](https://sqlperformance.com/2015/04/sql-plan/internals-of-the-seven-sql-server-sorts-part-1)  
Link: [More Undocumented Query Optimizer Trace Flags](http://www.benjaminnevarez.com/2012/04/more-undocumented-query-optimizer-trace-flags/)  
Link: [Yet another X-Ray for the QP](http://www.queryprocessor.com/tf_8628/)  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)  
Link: [Few Outer Rows Optimization](https://www.sqlshack.com/few-outer-rows-optimization/)  
Scope: session only

#### Trace Flag: 8608

Function: Shows the initial Memo structure  
Link: <http://www.queryprocessor.ru/optimizer-part-3-full-optimiztion-optimization-search0>  
Link: <http://www.benjaminnevarez.com/2012/04/inside-the-query-optimizer-memo-structure>  
Link: <http://sqlblog.com/blogs/paul_white/archive/2012/04/29/query-optimizer-deep-dive-part-3.aspx>  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)

#### Trace Flag: 8609

Function: PWhite: “Task and operation type counts”.  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)  
Link: <http://www.queryprocessor.ru/good-enough-plan>  
Scope: session only

#### Trace Flag: 8612

Function: Add Extra Info to the Trees Output  
Link: [Cardinality Estimation Framework 2014 First Look](http://www.somewheresomehow.ru/cardinality-estimation-framework-2014-first-look/)  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/06/11/hello-operator-my-switch-is-bored.aspx>

#### Trace Flag: 8615

**Undocumented trace flag**  
Function: Display the final memo structure  
Link: <http://www.benjaminnevarez.com/2012/04/inside-the-query-optimizer-memo-structure/>  
Link: <http://www.somewheresomehow.ru/optimizer-part-3-full-optimiztion-optimization-search0/>  
Link: [A Row Goal Riddle](https://orderbyselectnull.com/2018/03/30/a-row-goal-riddle/)  
Scope: session only

#### Trace Flag: 8619

**Undocumented trace flag**  
Function: Show Applied Transformation Rules  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/02/06/incorrect-results-with-indexed-views.aspx>  
Link: [Cardinality Estimation Framework 2014 First Look](http://www.somewheresomehow.ru/cardinality-estimation-framework-2014-first-look/)  
Link: [Yet another X-Ray for the QP](http://www.queryprocessor.com/tf_8628/)  
Link: [A Row Goal Riddle](https://orderbyselectnull.com/2018/03/30/a-row-goal-riddle/)  
Scope: session only

#### Trace Flag: 8620

**Undocumented trace flag**  
Function: Add memo arguments to trace flag 8619  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)  
Link: [Yet another X-Ray for the QP](http://www.queryprocessor.com/tf_8628/)  
Link: [A Row Goal Riddle](https://orderbyselectnull.com/2018/03/30/a-row-goal-riddle/)  
Scope: session only

#### Trace Flag: 8621

**Undocumented trace flag**  
Function: Rule with resulting tree  
Link: [Query Optimizer Deep Dive - Part 4](http://sqlblog.com/blogs/paul_white/archive/2012/05/01/query-optimizer-deep-dive-part-4.aspx)  
Link: [Yet another X-Ray for the QP](http://www.queryprocessor.com/tf_8628/)  
Link: [A Row Goal Riddle](https://orderbyselectnull.com/2018/03/30/a-row-goal-riddle/)  
Scope: session only

#### Trace Flag: 8628

**Undocumented trace flag**  
Function: When used with TF [8666](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#8666), causes extra information about the transformation rules applied to be put into the XML showplan.  
Link: [Yet another X-Ray for the QP](http://www.queryprocessor.com/tf_8628/)

#### Trace Flag: 8633

Function: PWhite: “Enable prefetch (CUpdUtil::FPrefetchAllowedForDML and CPhyOp\_StreamUpdate::FDoNotPrefetch)”  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/01/26/optimizing-t-sql-queries-that-change-data.aspx>

#### Trace Flag: 8649

Function: Set Cost Threshold for parallelism from 1 to 0  
Link: <http://sqlblog.com/blogs/paul_white/archive/2011/12/23/forcing-a-parallel-query-execution-plan.aspx>  
Link: <http://sqlblog.com/blogs/adam_machanic/archive/2013/07/11/next-level-parallel-plan-porcing.aspx>  
Link: [What You Need to Know about the Batch Mode Window Aggregate Operator in SQL Server 2016: Part 1](http://sqlmag.com/sql-server/what-you-need-know-about-batch-mode-window-aggregate-operator-sql-server-2016-part-1)  
Link: [Few Outer Rows Optimization](https://www.sqlshack.com/few-outer-rows-optimization/)  
Link: [Next-Level Parallel Plan Forcing: An Alternative to 8649](http://dataeducation.com/next-level-parallel-plan-forcing-an-alternative-to-8649/)  
Scope: session only

#### Trace Flag: 8665

**Undocumented trace flag**  
Function: Disables local/global aggregation.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 8666

**Undocumented trace flag**  
Function: Included in execution plans are the names of the stats it used to come up with the plan. Using a bit o' XML magic and over time, it allows you to clearly identify which stats are actually in use so that you can delete unused stats. CQScanPartitionSortNew is one of only two sort classes that sets the Soft Sort property exposed when Sort operator execution plan properties are generated with undocumented trace flag 8666 enabled  
Link: [Internals of the Seven SQL Server Sorts – Part 1](https://sqlperformance.com/2015/04/sql-plan/internals-of-the-seven-sql-server-sorts-part-1)  
Link: [Yet another X-Ray for the QP](http://www.queryprocessor.com/tf_8628/)  
Link: <https://blogfabiano.com/2012/07/03/statistics-used-in-a-cached-query-plan>  
Link: <http://dataidol.com/davebally/2014/04/12/reasons-why-your-plans-suck-no-56536>  
Link: <https://www.mssqltips.com/sqlservertip/4269/how-to-identify-useful-sql-server-table-statistics/>  
Link: [http://sql-sasquatch.blogspot.com/2018/06/harvesting-sql-server-trace-flag-8666.htmlhttp://sql-sasquatch.blogspot.com/2018/06/harvesting-sql-server-trace-flag-8666.html](http://sql-sasquatch.blogspot.com/2018/06/harvesting-sql-server-trace-flag-8666.htmlhttp:/sql-sasquatch.blogspot.com/2018/06/harvesting-sql-server-trace-flag-8666.html)  
Scope: session only

#### Trace Flag: 8671

**Undocumented trace flag**  
Function: According to Dima, disables the logic that prunes the memo and prevents the optimization process from stopping due to “Good Enough Plan found”. Can significantly increase the amount of time, CPU, and memory used in the compilation process  
Link: <http://www.queryprocessor.ru/optimizer_unleashed_2>

#### Trace Flag: 8675

Function: Displays the query optimization phases for a specific optimization  
Link: [More Undocumented Query Optimizer Trace Flags](http://www.benjaminnevarez.com/2012/04/more-undocumented-query-optimizer-trace-flags/)  
Link: <http://sqlblog.com/blogs/paul_white/archive/2012/04/29/query-optimizer-deep-dive-part-3.aspx>  
Link: <https://sqlperformance.com/2013/06/sql-indexes/recognizing-missed-optimizations>

#### Trace Flag: 8677

**Undocumented trace flag**  
Function: Skips “Search 1” phase of query optimization, and only Search 0 and Search 2 execute.  
Link: <https://sqlbits.com/Sessions/Event12/Query_Optimizer_Internals_Traceflag_fun>

#### Trace Flag: 8678

**Undocumented trace flag**  
Function: For one query this changed a bushy plan to a left deep one. There was no change in cost. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 8679

**Undocumented trace flag**  
Function: Prevents the SQL Server optimizer from using a Hash Match Team operator  
Link: None

#### Trace Flag: 8687

**Undocumented trace flag**  
Function: Prevents the SQL Server optimizer from using a Hash Match Team operator  
Link: None

#### Trace Flag: 8688

**Undocumented trace flag**  
Function: Disables parallel scans.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 8690

**Undocumented trace flag**  
Function: Disable the spool on the inner side of nested loop. Spools improve performance in majority of the cases. But it’s based on estimates. Sometimes, this can be incorrect due to unevenly distributed or skewed data, causing slow performance. But in vast majority of situations, you don’t need to manually disable spool with this trace flag.  
Link: <https://blogs.msdn.microsoft.com/psssql/2015/12/15/spool-operator-and-trace-flag-8690/>  
Link: <http://dba.stackexchange.com/questions/52552/index-not-making-execution-faster-and-in-some-cases-is-slowing-down-the-query>  
Link: <http://connect.microsoft.com/SQL/feedback/ViewFeedback.aspx?FeedbackID=453982>

#### Trace Flag: 8691

**Undocumented trace flag**  
Function: 'performance spool' optimization to the RegEx execution, reduces the number of executions of the RegEx function.  
Link: [Splitting Strings Based on Patterns](https://www.sqlservercentral.com/Forums/Topic1390297-3122-5.aspx)  
Scope: ?

#### Trace Flag: 8692

Function: Force optimizer to use an Eager Spool for Halloween Protection  
Link: <http://www.sqlperformance.com/2013/02/sql-plan/halloween-problem-part-4>  
Link: <https://sqlperformance.com/2016/03/sql-plan/changes-to-a-writable-partition-may-fail>

#### Trace Flag: 8719

Function: In SQL 2000, apparently would show IO prefetch on loop joins and bookmarks. I (Aaron) was unable to replicate the query plan behavior on SQL 2012 using the same test, so this flag may be obsolete.  
Link: <http://www.hanlincrest.com/SQLServerLockEscalation.htm>

#### Trace Flag: 8720

Function: In SQL 2000, apparently would have the same effect as OPTION(KEEPFIXED PLAN)  
Link: <http://www.hanlincrest.com/SQLserverStoredProcRecompiles.htm>

#### Trace Flag: 8721

Function: Reports to the error log when auto-update statistics executes  
Link: <https://support.microsoft.com/help/195565>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-8721/>  
Scope: global only

#### Trace Flag: 8722

Function: Disable all hints except locking hints  
Link: <http://sqlmag.com/sql-server/investigating-trace-flags>

#### Trace Flag: 8738

Function: (Apparently) disables an optimization where rows are sorted before a Key Lookup operator. (The optimization is meant to promote Sequential IO rather than the random nature of IO from Key Lookups). Note that the context in which this flag is described means that the above description may not be very precise, or even the only use of this flag.  
Link: <https://answers.sqlperformance.com/questions/603/why-is-the-sort-operator-needed-in-this-plan.html>

#### Trace Flag: 8739

Function: Group Optimization Information  
Link: <http://www.queryprocessor.ru/good-enough-plan>

#### Trace Flag: 8741

**Undocumented trace flag**  
Function: Resulted in a different join order for some queries with a higher estimated cost. Perhaps this disables Transitive Predicates? Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 8742

**Undocumented trace flag**  
Function: Resulted in a different join order for some queries. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 8743

**Undocumented trace flag**  
Function: Disable SM join.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 8744

Function: Disable pre-fetching for the Nested Loop operator. Incorrect use of this trace flag may cause additional physical reads when SQL Server executes plans that contain the Nested Loops operator.  
Link: [KB920093](https://support.microsoft.com/help/920093)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/03/08/execution-plan-analysis-the-mystery-work-table.aspx>  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/780194/make-dbcc-trace-flags-available-as-option-querytraceon>  
Scope: global or session

#### Trace Flag: 8746

Function: Whatever else it does, one effect is to disable the “rowset sharing” optimization described in the 2 PWhite posts.  
Link: <https://sqlperformance.com/2016/03/sql-plan/changes-to-a-writable-partition-may-fail>

#### Trace Flag: 8750

**Undocumented trace flag**  
Function: Skips search 0 optimization phase and moves to search 1.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 8755

Function: Disable all locking hints  
Link: <http://sqlmag.com/sql-server/investigating-trace-flags>

#### Trace Flag: 8757

Function: Skip trivial plan optimization and force a full optimization  
Link: [More Undocumented Query Optimizer Trace Flags](http://www.benjaminnevarez.com/2012/04/more-undocumented-query-optimizer-trace-flags/)  
Link: <http://sqlblog.com/blogs/paul_white/archive/2012/04/28/query-optimizer-deep-dive-part-1.aspx>

#### Trace Flag: 8758

Function: “A [workaround to the MERGE bug described] is to apply Trace Flag 8758 –unfortunately this disables a number of optimisations, not just the one above, so it’s not really recommended for long term use.” “Disable rewrite to a single operator plan (CPhyOp\_StreamUpdate::PqteConvert)”  
Link: <http://sqlblog.com/blogs/paul_white/archive/2010/08/04/another-interesting-merge-bug.aspx>  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/01/26/optimizing-t-sql-queries-that-change-data.aspx>

#### Trace Flag: 8759

**Undocumented trace flag**  
Function: Detect and write part of the query to the error log when it has been autoparameterized.  
Link: <https://github.com/ktaranov/sqlserver-kit/issues/146#issue-358855110>  
Scope: ?

#### Trace Flag: 8765

Function: Allows use of variable length data, from ODBC driver; fixes the issue of a field returning the wrong data length  
Link: <http://jacob.steelsmith.org/content/sql-server-and-ole-db>  
Scope: global or session

#### Trace Flag: 8780

Function: Give the optimizer more time to find a better plan  
Link: <http://www.queryprocessor.ru/optimizer_unleashed_1>  
Link: <http://www.sqlservice.se/sql-server-trace-flag-8780/>  
Scope: global or session

#### Trace Flag: 8783

Function: Allows DELETE, INSERT, and UPDATE statements to honor the SET ROWCOUNT ON setting when enabled  
Link: None

#### Trace Flag: 8790

Function: PWhite: “Undocumented trace flag 8790 forces a wide update plan for any data-changing query (remember that a wide update plan is always possible)”  
Link: <https://support.microsoft.com/help/956718/>  
Link: <http://sqlblog.com/blogs/paul_white/archive/2012/12/10/merge-bug-with-filtered-indexes.aspx>  
Link: <https://sqlperformance.com/2014/06/sql-plan/filtered-index-side-effect>

#### Trace Flag: 8795

Function: PWhite: “Disable DML Request Sort (CUpdUtil::FDemandRowsSortedForPerformance)”  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/01/26/optimizing-t-sql-queries-that-change-data.aspx>  
Link: <https://sqlperformance.com/2014/10/t-sql-queries/performance-tuning-whole-plan>

#### Trace Flag: 8799

**Undocumented trace flag**  
Function: Forces unordered scans.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 8809

Function: Extended Page Heap Activities. Referenced in passing in the CSS article in relation to debugging memory scribbler problems.  
Link: <https://blogs.msdn.microsoft.com/psssql/2012/11/12/how-can-reference-counting-be-a-leading-memory-scribbler-cause/>

#### Trace Flag: 8816

Function: Logs every two-digit year conversion to a four-digit year  
Link: None

#### Trace Flag: 8901

Function: Enables new (in 7.0) code to correct a problem with the SHRINK command and empty text or image extents  
Link: None

#### Trace Flag: 8903

Function: Allows SQL Server to use a specific API (SetFileIoOverlappedRange) when Locked Pages in Memory is enabled.  
Link: <https://blogs.msdn.microsoft.com/psssql/2012/03/20/setfileiooverlappedrange-can-lead-to-unexpected-behavior-for-sql-server-2008-r2-or-sql-server-2012-denali>  
Link: <https://support.microsoft.com/help/2679255/>  
Link: <https://blogs.msdn.microsoft.com/psssql/2013/10/16/every-time-i-attach-database-sql-logs-error-1314-for-setfileiooverlappedrange>

#### Trace Flag: 9024

Function: Converts a global log pool memory object into NUMA node partitioned memory object  
**Note: Beginning with SQL Server 2012 SP3 and SQL Server 2014 SP1 this behavior is controlled by the engine and trace flag 9024 has no effect.**  
Link: <https://support.microsoft.com/help/2809338>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global only  
Related to: [8048](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#8048)

#### Trace Flag: 9050

Function: “FIX: The compile time for a query that uses at least one outer join may be greater for SQL Server post-SP3 builds”  
Link: None

#### Trace Flag: 9052

Function: “FIX: Queries that join a view may run slowly if the view contains outer joins”  
Link: None

#### Trace Flag: 9054

Function: “FIX: SQL Server 2000 Service Pack 1 (SP1) and later builds may not generate an execution plan for a query, and you receive error message 8623”  
Link: None

#### Trace Flag: 9055

Function: “FIX: The performance of a DML operation that fires a trigger may decrease when the trigger execution plan recompiles repeatedly”  
Link: None

#### Trace Flag: 9056

Function: “FIX: A user-defined function returns results that are not correct for a query”  
Link: None

#### Trace Flag: 9059

Function: SQL 8 - Turns back behavior to SP3 after a SP4 installation, this allows to choose an index seek when comparing numeric columns or numeric constants that are of different precision or scale; else would have to change schema/code.  
Link: None

#### Trace Flag: 9061

Function: “FIX: Build 8.00.0837: A query that contains a correlated subquery runs slowly”  
Link: None

#### Trace Flag: 9062

Function: “FIX: Some complex queries are slower after you install SQL Server 2000 Service Pack 2 or SQL Server 2000 Service Pack 3”  
Link: None

#### Trace Flag: 9063

Function: “FIX: Query performance may be slower if the query contains both a GROUP BY clause and a DISTINCT keyword on the same column”  
Link: None

#### Trace Flag: 9065

Function: “FIX: The query plan may take longer than expected to compile, and you may receive error message 701, error message 8623, or error message 8651 in SQL Server 2000”  
Link: None

#### Trace Flag: 9068

Function: “FIX: A query may run more slowly against SQL Server 2000 post-SP3 hotfix build 8.00.0988 than a query that you run against SQL Server 2000 post-SP3 hotfix builds that are earlier than build 8.00.0988”  
Link: None

#### Trace Flag: 9079

Function: “FIX: The query performance may be slow when you query data from a view in SQL Server 2000”  
Link: None

#### Trace Flag: 9082

Function: SQL 9 - Stored procedure using views, perform slow compared to ver 8 if views use JOIN operator and contain sub queries  
Link: None

#### Trace Flag: 9109

Function: Used to workaround a problem with query notifications and restoring a DB with the NEW\_BROKER option enabled.  
Link: <https://support.microsoft.com/help/2483090/>

#### Trace Flag: 9114

**Undocumented trace flag**  
Function: Implemented a (SELECT 1) = 1 predicate as a join instead of optimizing it away.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9115

Function: PWhite: “Disable prefetch (CUpdUtil::FPrefetchAllowedForDML)” Dima: “Disables both [NLoop Implicit Batch Sort {TF 2340} and NL Prefetching {TF 8744}], and not only on the Post Optimization, but the explicit Sort also”  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/01/26/optimizing-t-sql-queries-that-change-data.aspx>  
Link: <http://www.hanlincrest.com/SQLServerLockEscalation.htm>  
Link: <http://www.queryprocessor.com/batch-sort-and-nested-loops>

#### Trace Flag: 9130

Function: Disables the particular copy out stage rewrite from Filter + (Scan or Seek) to (Scan or Seek) + Residual Predicate. Enabling this flag retains the Filter in the final execution plan, resulting in a SQL Server 2008+ plan that mirrors the 2005 version.  
Link: <http://sqlblog.com/blogs/paul_white/archive/2012/10/15/cardinality-estimation-bug-with-lookups-in-sql-server-2008-onward.aspx>  
Link: <http://sqlblogcasts.com/blogs/sqlandthelike/archive/2012/12/06/my-new-favourite-traceflag.aspx>  
Link: <http://sqlblog.com/blogs/paul_white/archive/2013/06/11/hello-operator-my-switch-is-bored.aspx>  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/767395/cardinality-estimation-error-with-pushed-predicate-on-a-lookup>  
Link: <http://www.theboreddba.com/Categories/FunWithFlags/Revealing-Predicates-in-Execution-Plans-(TF-9130).aspx>

#### Trace Flag: 9134

Function: SQL 8 - Does additional reads to test if the page is allocated & linked correctly this checks IAM & PFS. Fixes error 601 for queries under Isolation level read uncommitted. In case performance is affected (because of a bug) apply SP4.  
Link: <https://support.microsoft.com/help/815008/>

#### Trace Flag: 9136

Function: “PRB: You receive error message 8623 when you try to run a query that joins multiple tables”  
Link: None

#### Trace Flag: 9164

**Undocumented trace flag**  
Function: Disables HM (hash joins).  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9165

**Undocumented trace flag**  
Function: Disable NL join and remove an index recommendation from a plan.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9182

**Undocumented trace flag**  
Function: Resulted in a very strange cost change to a clustered index delete.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9183

**Undocumented trace flag**  
Function: Resulted in a very strange cost change to a clustered index delete.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9185

Function: Cardinality estimates for literals that are outside the histogram range are very low  
Link: None Related to: [9205](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#9205)

#### Trace Flag: 9204

Function: Output Statistics used by Query Optimizer. When enabled and a plan is compiled or recompiled there is a listing of statistics which is being fully loaded & used to produce cardinality and distribution estimates for some plan alternative or other.  
Link: [How to Find the Statistics Used to Compile an Execution Plan](http://sqlblog.com/blogs/paul_white/archive/2011/09/21/how-to-find-the-statistics-used-to-compile-an-paul_white)  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-9204/>  
Scope: global only Related to: [9292](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#9292)

#### Trace Flag: 9205

Function: Cardinality estimates for literals that are outside the histogram range are very low for tables that have parent-child relationships  
Link: None Related to: [9185](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#9185)

#### Trace Flag: 9207

Function: Fixes that SQL Server underestimates the cardinality of a query expression and query performance may be slow  
Link: None

#### Trace Flag: 9209

Function: “FIX: Some queries that have a left outer join and an IS NULL filter run slower after you install SQL Server 2000 post-SP3 hotfix”  
Link: None

#### Trace Flag: 9210

Function: “FIX: A query filter condition that has a LEFT OUTER JOIN clause may cause an incorrect row count estimate in the query execution plan”  
Link: None

#### Trace Flag: 9236

**Undocumented trace flag**  
Function: Resulted in a different join order for some queries. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9251

**Undocumented trace flag**  
Function: Change in cardinality estimates for some queries. It might only work with the legacy CE. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9259

Function: Disables Project Normalization step  
**Note: Please, don’t use TF 9259 that disables Project Normalization step in a real production system, besides it is undocumented and unsupported, it may hurt your performance.**  
Link: <http://www.queryprocessor.com/sudf-ce/>

#### Trace Flag: 9260

**Undocumented trace flag**  
Function: Adds an explicit sort before creation of an index spool. Almost doesn’t change the total estimated cost. Might be identical plans with just more detail shown at that step.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9268

Function: SQL 8 - When SQL Server runs a parameterized query that contains several IN clauses, each with a large number of values, SQL Server may return the following error message after a minute or more of high CPU utilization: KB 325658. Server: Msg 8623, Level 16, State 1. Internal Query Processor Error: The query processor could not produce a query plan. Contact your primary support provider for more information.  
Link: None

#### Trace Flag: 9275

Function: “FIX: A DML Operation on a Large Table Can Cause Performance Problems” Enables SQL 2000 optimizations that sort data in DML statements before the changes are applied to a clustered index  
Link: None

#### Trace Flag: 9284

**Undocumented trace flag**  
Function: Changed the order of a scalar operator comparison in a single join for certain queries. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9287

**Undocumented trace flag**  
Function: Appears to disable partial aggreation.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9288

**Undocumented trace flag**  
Function: Effects around local and global aggregates - choose coerce partial and global aggregation over scalar aggregation.  
Link: <https://github.com/ktaranov/sqlserver-kit/issues/93>  
Scope: local only

#### Trace Flag: 9292

Function: Output Statistics considered to be used by Query Optimizer  
Link: [How to Find the Statistics Used to Compile an Execution Plan](http://sqlblog.com/blogs/paul_white/archive/2011/09/21/how-to-find-the-statistics-used-to-compile-an-paul_white)  
Link: <http://www.sqlservergeeks.com/sql-server-trace-flag-9292/>  
Scope: session only Related to: [9204](https://github.com/ktaranov/sqlserver-kit/blob/master/SQL%20Server%20Trace%20Flag.md#9204)

#### Trace Flag: 9341

**Undocumented trace flag**  
Function: Resulted in a rather odd plan for a COUNT(DISTINCT) query against a CCI.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9346

**Undocumented trace flag**  
Function: Appears to disable batch mode window aggregates.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9347

Function: Disables batch mode for sort operator. SQL Server 2016 introduces a new batch mode sort operator that boosts performance for many analytical queries.  
Link: <https://support.microsoft.com/help/3172787>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [Niko Neugebauer Columnstore Indexes – part 86](http://www.nikoport.com/2016/07/29/columnstore-indexes-part-86-new-trace-flags-in-sql-server-2016/)  
Scope: global only

#### Trace Flag: 9348

Function: Sets a row limit (based on cardinality estimates) that controls whether a bulk insert is attempted or not (assuming conditions are met for a bulk insert). Introduced as a workaround for memory errors encountered with bulk insert.  
Link: <https://support.microsoft.com/help/2998301/>

#### Trace Flag: 9349

Function: Disables batch mode for top N sort operator. SQL Server 2016 introduces a new batch mode top sort operator that boosts performance for many analytical queries.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [Niko Neugebauer Columnstore Indexes – part 86](http://www.nikoport.com/2016/07/29/columnstore-indexes-part-86-new-trace-flags-in-sql-server-2016/)  
Link: <https://support.microsoft.com/help/3172787>  
Scope: global or session or query

#### Trace Flag: 9354

**Undocumented trace flag**  
Function: Disable [aggregate pushdown](http://www.nikoport.com/2015/07/11/columnstore-indexes-part-59-aggregate-pushdown/) operations for columnstore indexes. The number of rows aggregated at the level of the scan is displayed in the new property plan [Actual Number Of Locally Aggregated Rows](http://www.nikoport.com/2016/03/21/clustered-columnstore-indexes-part-80-local-aggregation/). TF 9354 can be used to disable the push of aggregation, the difference can be observed by the runtime, according to the number of rows in the plan Actual Number Of Locally Aggregated Rows and number Actual Number Of Rows output from the scan operator.  
Example:

use AdventureworksDW2016CTP3;

set nocount on;

go

-- Undocumented TF 9354 disables this optimization, run to see Aggregation Pushdown Performance Gain

set statistics xml, time on;

select count\_big(\*) from dbo.FactResellerSalesXL\_CCI;

select count\_big(\*) from dbo.FactResellerSalesXL\_CCI option(querytraceon 9354); -- undocumented/unsupported TF 9354 to disable aggregate pushdown

set statistics xml, time off;

#### Trace Flag: 9358

Function: Disable batch mode sort operations in a complex parallel query. For example, this flag could apply if the query contains merge join operations.  
Link: [Niko Neugebauer Columnstore Indexes – part 86](http://www.nikoport.com/2016/07/29/columnstore-indexes-part-86-new-trace-flags-in-sql-server-2016/)  
Link: <https://support.microsoft.com/help/3171555>

#### Trace Flag: 9384

**Undocumented trace flag**  
Function: Very slightly changed the memory grant of a query with a batch mode window aggregate.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9389

Function: Enables dynamic memory grant for batch mode operators. If a query does not get all the memory it needs, it spills data to tempdb, incurring additional I/O and potentially impacting query performance. If the dynamic memory grant trace flag is enabled, a batch mode operator may ask for additional memory and avoid spilling to tempdb if additional memory is available.  
Link: [Niko Neugebauer Columnstore Indexes – part 86](http://www.nikoport.com/2016/07/29/columnstore-indexes-part-86-new-trace-flags-in-sql-server-2016/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 9390

**Undocumented trace flag**  
Function: Resulted in plan changes including parallelism for queries that shouldn’t have been eligible for parallelism based on CTFP. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9394

Function: Apparently enables a fix for an access violation when a table with Japanese characters has an indexed changed.  
Link: <https://support.microsoft.com/help/3142595/>  
Link: <https://support.microsoft.com/help/3138659/>

#### Trace Flag: 9398

**Undocumented trace flag**  
Function: Disable adaptive join.  
Link: [SQL Server 2017: Adaptive Join Internals](http://www.queryprocessor.com/adaptive-join-internals/)  
Scope: ?

#### Trace Flag: 9399

**Undocumented trace flag**  
Function: Optimization adaptive threshold rows. The adaptive threshold to the minimum estimate.  
Link: [SQL Server 2017: Adaptive Join Internals](http://www.queryprocessor.com/adaptive-join-internals/)  
Scope: ?

#### Trace Flag: 9410

**Undocumented trace flag**  
Function: Fix slowly query runs when SQL Server uses hash aggregate in the query plan.  
Link: <https://support.microsoft.com/help/3167159/>  
Scope: ?

#### Trace Flag: 9412

**Undocumented trace flag**  
Function: Removes the new OptimizerStatsUsage information from estimated query plans.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9415

**Undocumented trace flag**  
Function: Optimization adaptive join internals.  
Link: [SQL Server 2017: Adaptive Join Internals](http://www.queryprocessor.com/adaptive-join-internals/)  
Scope: ?

#### Trace Flag: 9447

**Undocumented trace flag**  
Function: Forces query plans to use the new referential integrity operator when validating UPDATE and DELETE queries against foreign key parent tables.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9448

**Undocumented trace flag**  
Function: Disables the referential integrity operator.  
Link: <https://orderbyselectnull.com/2017/12/05/the-referential-integrity-operator/>

#### Trace Flag: 9453

Function: Disables Batch Mode in Parallel Columnstore query plans. (Note that a plan using batch mode appears to require a recompile before the TF takes effect) Sunil Agarwal also used this trace flag in demo scripts for a PASS 2014 session on column store indexing  
Link: [Niko Neugebauer Columnstore Indexes – part 35](http://www.nikoport.com/2014/07/24/clustered-columnstore-indexes-part-35-trace-flags-query-optimiser-rules/)  
Link: [What You Need to Know about the Batch Mode Window Aggregate Operator in SQL Server 2016: Part 1](http://sqlmag.com/sql-server/what-you-need-know-about-batch-mode-window-aggregate-operator-sql-server-2016-part-1)

#### Trace Flag: 9471

Function: Causes SQL Server to generate a plan using minimum selectivity for single-table filters, under the query optimizer cardinality estimation model of SQL Server 2014 through SQL Server 2016 versions. Beginning with SQL Server 2016 SP1, to accomplish this at the query level, add the USE HINT query hint instead of using this trace flag.  
**Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session or query

#### Trace Flag: 9472

Function: Assumes independence for multiple WHERE predicates in the SQL 2014 cardinality estimation model. Predicate independence was the default for versions prior to SQL Server 2014, and thus this flag can be used to more closely emulate pre-SQL 2014 cardinality estimate behavior in a more specific fashion than TF 9481.  
Link: <https://sqlperformance.com/2014/01/sql-plan/cardinality-estimation-for-multiple-predicates>  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/801908/sql-server-2014-cardinality-estimation-regression>

#### Trace Flag: 9473

**Undocumented trace flag**  
Function: Allowing the outer join to keep a zero-row inner-side estimate (instead of raising to one row) (so all outer rows qualify) gives a 'bug-free' join estimation with either calculator. If you're interested in exploring this, the undocumented trace flag is 9473 (alone).  
Link: <https://dba.stackexchange.com/a/141533/107045>  
Scope: ?

#### Trace Flag: 9474

**Undocumented trace flag**  
Function: Change in cardinality estimates for some joins in certain queries. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9476

Function: Causes SQL Server to generate a plan using the Simple Containment assumption instead of the default Base Containment assumption, under the query optimizer cardinality estimation model of SQL Server 2014 through SQL Server 2016 versions. Beginning with SQL Server 2016 SP1, to accomplish this at the query level, add the USE HINT query hint instead of using this trace flag. **Note: Please ensure that you thoroughly test this option, before rolling it into a production environment.**  
Link: <https://support.microsoft.com/help/3189675>  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session or query

#### Trace Flag: 9477

**Undocumented trace flag**  
Function: Slight change in ratio of EstimateRebinds and EstimateRewinds was observed. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9478

**Undocumented trace flag**  
Function: Change in cardinality estimates for some joins in certain queries. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9479

**Undocumented trace flag**  
Function: Forces the optimizer to use Simple Join [estimation] even if a histogram is available. Will force optimizer to use a simple join estimation algorithm, it may be CSelCalcSimpleJoinWithDistinctCounts, CSelCalcSimpleJoin or CSelCalcSimpleJoinWithUpperBound, depending on the compatibility level and predicate comparison type.  
Link: [Statistics and Cardinality Estimation](http://topicaltraceflags.readthedocs.io/en/latest/cat/qry_StatsAndEst.html)  
Scope: ?

#### Trace Flag: 9480

**Undocumented trace flag**  
Function: Reduced the selectivity of a bitmap filter from 0.001 to 0.000001. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9481

Function: Enables you to set the query optimizer cardinality estimation model to the SQL Server 2012 and earlier version independent of the compatibility level of the database. To accomplish this at the database level, see [ALTER DATABASE SCOPED CONFIGURATION (Transact-SQL)](https://docs.microsoft.com/sql/t-sql/statements/alter-database-scoped-configuration-transact-sql). To accomplish this at the query level, add the QUERYTRACEON query hint  
Link: [New Features in SQL Server 2016 Service Pack 1](https://www.mssqltips.com/sqlservertip/4574/new-features-in-sql-server-2016-service-pack-1/)  
Link: <https://sqlserverscotsman.wordpress.com/2016/11/28/a-guide-on-forcing-the-legacy-ce/>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [KB2801413](https://support.microsoft.com/help/2801413)  
Link: <http://www.sqlservergeeks.com/sql-server-2014-trace-flags-9481/>  
Scope: global or session or query

#### Trace Flag: 9482

**Undocumented trace flag**  
Function: Implements a “model variation” in the SQL 2014 cardinality estimator. The flag turns off the “overpopulated primary key” adjustment that the optimizer might use when determining that a “dimension” table (the schema could be OLTP as well) has many more distinct values than the “fact” table. (The seminal example is where a Date dimension is populated out into the future, but the fact table only has rows up to the current date). Since join cardinality estimation occurs based on the contents of the histograms of the joined columns, an “overpopulated primary key” can result in higher selectivity estimates, causing rowcount estimates to be too low.  
Link: <http://www.queryprocessor.com/ce_opk>

#### Trace Flag: 9483

**Undocumented trace flag**  
Function: Implements a “model variation” in the SQL 2014 cardinality estimator. The flag will force the optimizer to create (if possible) a filtered statistics object based on a predicate in the query. This filtered stat object is not persisted and thus would be extremely resource intensive for frequent compilations. In Dima’s example, the filtered stat object is actually created on the join column...i.e. “CREATE STATISTICS [filtered stat obj] ON [table] (Join column) WHERE (predicate column = ‘literal’)”  
Link: <http://www.queryprocessor.com/ce_filteredstats>

#### Trace Flag: 9484

**Undocumented trace flag**  
Function: Slight change in estimated number of rewinds. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9485

Function: Disables SELECT permission for DBCC SHOW\_STATISTICS  
Link: <https://support.microsoft.com/help/2683304>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <http://www.benjaminnevarez.com/2013/02/dbcc-show_statistics-works-with-select-permission>  
Scope: global only

#### Trace Flag: 9488

**Undocumented trace flag**  
Function: Implements a “model variation” in the SQL 2014 cardinality estimator. This flag reverts the estimation behavior for multi-statement TVFs back to 1 row (instead of the 100-row estimate behavior that was adopted in SQL 2014).  
Link: <http://www.queryprocessor.com/ce_mtvf>

#### Trace Flag: 9489

**Undocumented trace flag**  
Function: Implements a “model variation” in the SQL 2014 cardinality estimator and turns off the new logic that handles ascending keys.  
Link: <http://www.queryprocessor.com/ce_asckey>

#### Trace Flag: 9490

**Undocumented trace flag**  
Function: Change in cardinality estimate. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 9494

**Undocumented trace flag**  
Function: The behaviour of the join cardinality estimation with CSelCalcExpressionComparedToExpression can also be modified to not account for ``bId` with another undocumented variation flag (9494)  
Link: <https://dba.stackexchange.com/a/141533/107045>  
Scope: ?

#### Trace Flag: 9495

Function: Disables parallelism during insertion for INSERT...SELECT operations and it applies to both user and temporary tables  
Link: <https://support.microsoft.com/help/3180087>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 9532

Function: SQL 11 CTP3 - to get more than 1 availability group replica in CTP3 Scope Startup  
Link: <http://connect.microsoft.com/SQLServer/feedback/details/682581/denali-hadron-read-only-routing-url-is-not-yet-implemente>

#### Trace Flag: 9559

**Undocumented trace flag**  
Function: For AGs, “when enabled on the secondary ignores the redo target provided from the primary progress message and always set the redo target at the Max LSN value.”  
Link: <https://blogs.msdn.microsoft.com/alwaysonpro/2013/12/04/recovery-on-secondary-lagging-shared-redo-target>

#### Trace Flag: 9567

Function: Enables compression of the data stream for availability groups during automatic seeding. Compression can significantly reduce the transfer time during automatic seeding and will increase the load on the processor.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://www.mssqltips.com/sqlservertip/4537/sql-server-2016-availability-group-automatic-seeding/>  
Link: <https://msdn.microsoft.com/en-us/library/mt735149.aspx>  
Link: [Tune compression for availability group](https://docs.microsoft.com/sql/database-engine/availability-groups/windows/tune-compression-for-availability-group)  
Scope: global or session

#### Trace Flag: 9576

Function: Revert to the original (SQL Server 2016) implementation of database level health detection using TF 9576 as either a startup parameter or enabled using DBCC TRACEON command. This new implementation is currently only available for SQL Server running on Windows and will be ported to SQL Server 2017 on Linux in an upcoming cumulative update.  
Link: <https://blogs.msdn.microsoft.com/sql_server_team/sql-server-availability-groups-enhanced-database-level-failover/>  
Scope: global only

#### Trace Flag: 9591

Function: Disables log block compression in Always On Availability Groups. Log block compression is the default behavior used with both synchronous and asynchronous replicas in SQL Server 2012 and SQL Server 2014. In SQL Server 2016, compression is only used with asynchronous replica.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 9592

Function: Enables log stream compression for synchronous availability groups. This feature is disabled by default on synchronous availability groups because compression adds latency.  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [Tune compression for availability group](https://docs.microsoft.com/sql/database-engine/availability-groups/windows/tune-compression-for-availability-group)  
Scope: global or session

#### Trace Flag: 9706

**Undocumented trace flag**  
Function: Software Usage Metrics is disabled.  
Link: [Bad Idea Jeans: Finding Undocumented Trace Flags](https://www.brentozar.com/archive/2017/10/bad-idea-jeans-finding-undocumented-trace-flags/)

#### Trace Flag: 9806

**Undocumented trace flag**  
Function: Unknown. Is turned on on SQL Server 2014 CTP1 standard installation in Windows Azure VM  
Link: None

#### Trace Flag: 9807

**Undocumented trace flag**  
Function: Unknown. Is turned on on SQL Server 2014 CTP1 standard installation in Windows Azure VM  
Link: None

#### Trace Flag: 9808

**Undocumented trace flag**  
Function: Unknown. Is turned on on SQL Server 2014 CTP1 standard installation in Windows Azure VM  
Link: None

#### Trace Flag: 9830

**Undocumented trace flag**  
Function: Activate the trace flag before creating a natively compiled procedure. If you now open up the SQL Server error log you should see the compilation process for the natively compiled procedure. This is an undocumented trace flag so please don’t use this on a production system.  
Link: [https://web.archive.org/web/20160327221828/http://speedysql.com/2015/10/28/new-trace-flag-for-in-memory-oltp-hekaton/](https://web.archive.org/web/20160327221828/http:/speedysql.com/2015/10/28/new-trace-flag-for-in-memory-oltp-hekaton/)

#### Trace Flag: 9837

**Undocumented trace flag**  
Function: According to Bob Ward’s PASS 2014 talk on SQL Server IO, enables “extra tracing but massive output” for Hekaton checkpoint files.  
Link: None

#### Trace Flag: 9850

**Undocumented trace flag**  
Function: Dumps more diagnostic stuff in the log.  
Link: [Bad Idea Jeans: Finding Undocumented Trace Flags](https://www.brentozar.com/archive/2017/10/bad-idea-jeans-finding-undocumented-trace-flags/)

#### Trace Flag: 9851

**Undocumented trace flag**  
Function: For testing purposes, you might want to turn off automatic merging of files, so that you can more readily explore this metadata. You can do that by turning on the undocumented trace flag 9851. And of course, be sure to turn off the trace flag when done testing.  
Link: <http://gsl.azurewebsites.net/Portals/0/Users/dewitt/talks/HekatonWhitePaper.pdf>

#### Trace Flag: 9929

Function: Enables an update that reduces the “disk footprint [of In-Memory OLTP] by reducing the In-Memory checkpoint files to 1 MB (megabytes) each.”  
Link: <https://support.microsoft.com/help/3147012/>

#### Trace Flag: 9939

Function: Disables merge/recompress during columnstore index reorganization. In SQL Server 2016, when a columnstore index is reorganized, there is new functionality to automatically merge any small compressed rowgroups into larger compressed rowgroups, as well as recompressing any rowgroups that have a large number of deleted rows.  
**Note: Trace flag 10204 does not apply to columnstore indexes which are created on memory-optimized tables.**  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: [Parallelism in Hekaton (In-Memory OLTP)](http://www.nikoport.com/2018/01/20/parallelism-in-hekaton-in-memory-oltp/) Scope: global or session

#### Trace Flag: 9989

Function: In CTP2, enabled functionality for reading in-memory tables on a readable secondary  
Link: <https://connect.microsoft.com/SQLServer/feedback/details/795360/secondary-db-gets-suspect-when-i-add-in-memory-table-to-db-which-is-part-of-alwayson-availability-group>

#### Trace Flag: 10202

**Undocumented trace flag**  
Function: According to demo scripts from a Sunil Agarwal session at PASS 2014, enables a new DMV named sys.dm\_db\_column\_store\_row\_group\_physical\_stats. This DMV is not in SQL 2014 RTM and Sunil did not perform this demo during the session, so this DMV appears to be in a future (or internal) version of SQL Server.  
Link: None

#### Trace Flag: 10204

Function: Disables merge/recompress during columnstore index reorganization. In SQL Server 2016, when a columnstore index is reorganized, there is new functionality to automatically merge any small compressed rowgroups into larger compressed rowgroups, as well as recompressing any rowgroups that have a large number of deleted rows.  
**Note: Trace flag 10204 does not apply to column store indexes which are created on memory-optimized tables.**  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 10207

Function: When a Clustered Columnstore index has corrupted segments, turning on this flag suppresses errors 5288 and 5289 and allows a scan of a clustered columns store to skip corrupt segments and complete (though with results that do not include the corrupted segment(s)). This flag is helpful when attempting to copy-out data in a corrupt CCI.  
Link: <https://support.microsoft.com/help/3067257/>  
Link: <https://blogs.msdn.microsoft.com/sqlreleaseservices/partial-results-in-a-query-of-a-clustered-columnstore-index-in-sql-server-2014>

#### Trace Flag: 10213

**Undocumented trace flag**  
Function: Enables the option to configure compression delay in columnstore indexes in SQL Server 2016  
Link: <http://www.nikoport.com/2016/02/04/columnstore-indexes-part-76-compression-delay/>  
Scope: session only

#### Trace Flag: 10264

**Undocumented trace flag**  
Function: Polybase mode enabled for SqlComposable.  
Link: [Bad Idea Jeans: Finding Undocumented Trace Flags](https://www.brentozar.com/archive/2017/10/bad-idea-jeans-finding-undocumented-trace-flags/)

#### Trace Flag: 10316

Function: Enables creation of additional indexes on internal memory-optimized staging temporal table, beside the default one. If you have specific query pattern that includes columns which are not covered by the default index you may consider adding additional ones.  
**Note: System-versioned temporal tables for Memory-Optimized Tables are designed to provide high transactional throughput. Please be aware that creating additional indexes may introduce overhead for DML operations that update or delete rows in the current table. With the additional indexes you should aim to find the right balance between performance of temporal queries and additional DML overhead.**  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Link: <https://support.microsoft.com/help/3198846>  
Link: <https://blogs.msdn.microsoft.com/sqlcat/2016/12/08/improve-query-performance-on-memory-optimized-tables-with-temporal-using-new-index-creation-enhancement-in-sp1/>  
Scope: global or session

#### Trace Flag: 10809

**Undocumented trace flag**  
Function: Force stream Aggregates for scalar aggregation in batch mode.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 11001

**Undocumented trace flag**  
Function: Results in a different join order for some queries. Full effect unknown.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)  
Scope: ?

#### Trace Flag: 11023

Function: Disables the use of the last persisted sample rate, for all subsequent statistics update where a sample rate is not specified explicitly as part of the [UPDATE STATISTICS](https://docs.microsoft.com/en-us/sql/t-sql/statements/update-statistics-transact-sql) statement.  
Link: <https://support.microsoft.com/help/4039284>  
Link: [Docs Trace Flags](https://docs.microsoft.com/sql/t-sql/database-console-commands/dbcc-traceon-trace-flags-transact-sql)  
Scope: global or session

#### Trace Flag: 11024

Function: In Microsoft SQL Server 2017, when incremental statistics are built on the top of partitioned tables, the sum of modification counts of all partitions is stored as the modification count of the root node. When the modification count of the root node exceeds a threshold, the auto update of statistics is triggered. However, if the modification count of any single partition does not exceed the local threshold, the statistics are not updated. Additionally, the modification count of the root node is reset to zero. This may cause delay in the auto update of incremental statistics. When trace flag 11024 is enabled, the modification count of the root node is kept as the sum of modification counts of all partitions.  
**Note: This trace flag applies to SQL Server 2017 CU3 and higher builds.**  
Link: <https://support.microsoft.com/help/4041811>  
Scope: global or session

#### Trace Flag: 11029

**Undocumented trace flag**  
Function: Prevents new information about row goals from getting logged to the plan cache.  
Link: [New Undocumented Trace Flags](https://orderbyselectnull.com/2018/01/09/45-new-trace-flags/)

Scope: ?