

Wait type

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:27 AM PST


Contents

- [Query Store Wait Stats](#)
- [DB Wait Stats](#)
- [OS Wait Stats](#)

Wait Types

This TSG provides you with lists of wait types from several sources. Detailed comments are included as far as available.


Query Store Wait Stats

The main source of the values in this table are from the [sys.query_store_wait_stats](#)  documentation. Additional entries can also be added manually.

Integer value	Wait category	Wait types include in the category
0	Unknown	Unknown
1	CPU	SOS_SCHEDULER_YIELD
2	Worker Thread*	THREADPOOL
3	Lock	LCK_M_%
4	Latch	LATCH_%
5	Buffer Latch	PAGELATCH_%
6	Buffer IO	PAGEIOLATCH_%
7	Compilation*	RESOURCE_SEMAPHORE_QUERY_COMPILE
8	SQL CLR	CLR%, SQLCLR%
9	Mirroring	DBMIRROR%
10	Transaction	XACT%, DTC%, TRAN_MARKLATCH_%, MSQL_XACT_%, TRANSACTION_MUTEX
11	Idle	SLEEP_%, LAZYWRITER_SLEEP, SQLTRACE_BUFFER_FLUSH, SQLTRACE_INCREMENTAL_FLUSH_SLEEP, SQLTRACE_WAIT_ENTRIES, FT_IFTS_SCHEDULER_IDLE_WAIT, XE_DISPATCHER_WAIT, REQUEST_FOR_DEADLOCK_SEARCH, LOGMGR_QUEUE, ONDEMAND_TASK_QUEUE, CHECKPOINT_QUEUE, XE_TIMER_EVENT
12	Preemptive	PREEMPTIVE_%
13	Service Broker	BROKER_% (but not BROKER_RECEIVE_WAITFOR)
14	Tran Log IO	LOGMGR, LOGBUFFER, LOGMGR_RESERVE_APPEND, LOGMGR_FLUSH, LOGMGR_PMM_LOG, CHKPT, WRITELOG
15	Network IO	ASYNC_NETWORK_IO, NET_WAITFOR_PACKET, PROXY_NETWORK_IO, EXTERNAL_SCRIPT_NETWORK_IOF
16	Parallelism	CXPACKET, EXCHANGE, HT%, BMP%, BP%

Integer value	Wait category	Wait types include in the category
17	Memory	RESOURCE_SEMAPHORE, CMEMTHREAD, CMEMPARTITIONED, EE_PMOLOCK, MEMORY_ALLOCATION_EXT, RESERVED_MEMORY_ALLOCATION_EXT, MEMORY_GRANT_UPDATE
18	User Wait	WAITFOR, WAIT_FOR_RESULTS, BROKER_RECEIVE_WAITFOR
19	Tracing	TRACEWRITE, SQLTRACE_LOCK, SQLTRACE_FILE_BUFFER, SQLTRACE_FILE_WRITE_IO_COMPLETION, SQLTRACE_FILE_READ_IO_COMPLETION, SQLTRACE_PENDING_BUFFER_WRITERS, SQLTRACE_SHUTDOWN, QUERY_TRACEOUT, TRACE_EVTNOTIFF
20	Full Text Search	FT_RESTART_CRAWL, FULLTEXT GATHERER, MSSEARCH, FT_METADATA_MUTEX, FT_IFTSHC_MUTEX, FT_IFTSISM_MUTEX, FT_IFTS_RWLOCK, FT_COMPROWSET_RWLOCK, FT_MASTER_MERGE, FT_PROPERTYLIST_CACHE, FT_MASTER_MERGE_COORDINATOR, PWAIT_RESOURCE_SEMAPHORE_FT_PARALLEL_QUERY_SYNC
21	Other Disk IO	ASYNC_IO_COMPLETION, IO_COMPLETION, BACKUPIO, WRITE_COMPLETION, IO_QUEUE_LIMIT, IO_RETRY
22	Replication	SE_REPL_%, REPL_%, HADR_% (but not HADR_THROTTLE_LOG_RATE_GOVERNOR), PWAIT_HADR_%, REPLICA_WRITES, FCB_REPLICA_WRITE, FCB_REPLICA_READ, PWAIT_HADRSIM
23	Log Rate Governor	LOG_RATE_GOVERNOR, POOL_LOG_RATE_GOVERNOR, HADR_THROTTLE_LOG_RATE_GOVERNOR, INSTANCE_LOG_RATE_GOVERNOR

DB Wait Stats

The main source of the values in this table are from the [sys.dm_db_wait_stats \(Azure SQL Database\)](#)  documentation. Additional entries can also be added manually. There is some overlap with the OS Wait Stats listed further below.

Wait type	Description
ABR	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
ASSEMBLY_LOAD	Occurs during exclusive access to assembly loading.
ASYNC_DISKPOOL_LOCK	Occurs when there is an attempt to synchronize parallel threads that are performing tasks such as creating or initializing a file.
ASYNC_IO_COMPLETION	Occurs when a task is waiting for I/Os to finish.
ASYNC_NETWORK_IO	Occurs on network writes when the task is blocked behind the network. Verify that the client is processing data from the server.
AUDIT_GROUPCACHE_LOCK	Occurs when there is a wait on a lock that controls access to a special cache. The cache contains information about which audits are being used to audit each audit action group.
AUDIT_LOGINCACHE_LOCK	Occurs when there is a wait on a lock that controls access to a special cache. The cache contains information about which audits are being used to audit login audit action groups.
AUDIT_ON_DEMAND_TARGET_LOCK	Occurs when there is a wait on a lock that is used to ensure single initialization of audit related Extended Event targets.
AUDIT_XE_SESSION_MGR	Occurs when there is a wait on a lock that is used to synchronize the starting and stopping of audit related Extended Events sessions.
BACKUP	Occurs when a task is blocked as part of backup processing.

Wait type	Description
BACKUP_OPERATOR	Occurs when a task is waiting for a tape mount.
BACKUPBUFFER	Occurs when a backup task is waiting for data, or is waiting for a buffer in which to store data. This type is not typical, except when a task is waiting for a tape mount.
BACKUPIO	Occurs when a backup task is waiting for data, or is waiting for a buffer in which to store data. This type is not typical, except when a task is waiting for a tape mount.
BACKUPTHREAD	Occurs when a task is waiting for a backup task to finish. Wait times may be long, from several minutes to several hours. If the task that is being waited on is in an I/O process, this type does not indicate a problem.
BAD_PAGE_PROCESS	Occurs when the background suspect page logger is trying to avoid running more than every five seconds. Excessive suspect pages cause the logger to run frequently.
BROKER_CONNECTION_RECEIVE_TASK	Occurs when waiting for access to receive a message on a connection endpoint. Receive access to the endpoint is serialized.
BROKER_ENDPOINT_STATE_MUTEX	Occurs when there is contention to access the state of a Service Broker connection endpoint. Access to the state for changes is serialized.
BROKER_EVENTHANDLER	Occurs when a task is waiting in the primary event handler of the Service Broker. This should occur very briefly.
BROKER_INIT	Occurs when initializing Service Broker in each active database. This should occur infrequently.

Wait type	Description
BROKER_MASTERSTART	Occurs when a task is waiting for the primary event handler of the Service Broker to start. This should occur very briefly.
BROKER_RECEIVE_WAITFOR	Occurs when the RECEIVE WAITFOR is waiting. This is typical if no messages are ready to be received.
BROKER_REGISTERALLENDPOINTS	Occurs during the initialization of a Service Broker connection endpoint. This should occur very briefly.
BROKER_SERVICE	Occurs when the Service Broker destination list that is associated with a target service is updated or re-prioritized.
BROKER_SHUTDOWN	Occurs when there is a planned shutdown of Service Broker. This should occur very briefly, if at all.
BROKER_TASK_STOP	Occurs when the Service Broker queue task handler tries to shut down the task. The state check is serialized and must be in a running state beforehand.
BROKER_TO_FLUSH	Occurs when the Service Broker lazy flusher flushes the in-memory transmission objects to a work table.
BROKER_TRANSMITTER	Occurs when the Service Broker transmitter is waiting for work.
BUILTIN_HASHKEY_MUTEX	May occur after startup of instance, while internal data structures are initializing. Will not recur once data structures have initialized.
CHECK_PRINT_RECORD	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.

Wait type	Description
CHECKPOINT_QUEUE	Occurs while the checkpoint task is waiting for the next checkpoint request.
CHKPT	Occurs at server startup to tell the checkpoint thread that it can start.
CLEAR_DB	Occurs during operations that change the state of a database, such as opening or closing a database.
CLR_AUTO_EVENT	Occurs when a task is currently performing common language runtime (CLR) execution and is waiting for a particular autoevent to be initiated. Long waits are typical, and do not indicate a problem.
CLR_CRST	Occurs when a task is currently performing CLR execution and is waiting to enter a critical section of the task that is currently being used by another task.
CLR_JOIN	Occurs when a task is currently performing CLR execution and waiting for another task to end. This wait state occurs when there is a join between tasks.
CLR_MANUAL_EVENT	Occurs when a task is currently performing CLR execution and is waiting for a specific manual event to be initiated.
CLR_MEMORY_SPY	Occurs during a wait on lock acquisition for a data structure that is used to record all virtual memory allocations that come from CLR. The data structure is locked to maintain its integrity if there is parallel access.
CLR_MONITOR	Occurs when a task is currently performing CLR execution and is waiting to obtain a lock on the monitor.

Wait type	Description
CLR_RWLOCK_READER	Occurs when a task is currently performing CLR execution and is waiting for a reader lock.
CLR_RWLOCK_WRITER	Occurs when a task is currently performing CLR execution and is waiting for a writer lock.
CLR_SEMAPHORE	Occurs when a task is currently performing CLR execution and is waiting for a semaphore.
CLR_TASK_START	Occurs while waiting for a CLR task to complete startup.
CLRHOST_STATE_ACCESS	Occurs where there is a wait to acquire exclusive access to the CLR-hosting data structures. This wait type occurs while setting up or tearing down the CLR runtime.
CMEMTHREAD	Occurs when a task is waiting on a thread-safe memory object. The wait time might increase when there is contention caused by multiple tasks trying to allocate memory from the same memory object.
CXPACKET	Occurs when trying to synchronize the query processor exchange iterator. You may consider lowering the degree of parallelism if contention on this wait type becomes a problem.
CXROWSET_SYNC	Occurs during a parallel range scan.
DAC_INIT	Occurs while the dedicated administrator connection is initializing.
DBMIRROR_DBM_EVENT	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.

Wait type	Description
DBMIRROR_DBM_MUTEX	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
DBMIRROR_EVENTS_QUEUE	Occurs when database mirroring waits for events to process.
DBMIRROR_SEND	Occurs when a task is waiting for a communications backlog at the network layer to clear to be able to send messages. Indicates that the communications layer is starting to become overloaded and affect the database mirroring data throughput.
DBMIRROR_WORKER_QUEUE	Indicates that the database mirroring worker task is waiting for more work.
DBMIRRORING_CMD	Occurs when a task is waiting for log records to be flushed to disk. This wait state is expected to be held for long periods of time.
DEADLOCK_ENUM_MUTEX	Occurs when the deadlock monitor and sys.dm_os_waiting_tasks try to make sure that SQL Server is not running multiple deadlock searches at the same time.
DEADLOCK_TASK_SEARCH	Large waiting time on this resource indicates that the server is executing queries on top of sys.dm_os_waiting_tasks, and these queries are blocking deadlock monitor from running deadlock search. This wait type is used by deadlock monitor only. Queries on top of sys.dm_os_waiting_tasks use DEADLOCK_ENUM_MUTEX.
DEBUG	Occurs during Transact-SQL and CLR debugging for internal synchronization.

Wait type	Description
DISABLE_VERSIONING	Occurs when SQL Server polls the version transaction manager to see whether the timestamp of the earliest active transaction is later than the timestamp of when the state started changing. If this is this case, all the snapshot transactions that were started before the ALTER DATABASE statement was run have finished. This wait state is used when SQL Server disables versioning by using the ALTER DATABASE statement.
DISKIO_SUSPEND	Occurs when a task is waiting to access a file when an external backup is active. This is reported for each waiting user process. A count larger than five per user process may indicate that the external backup is taking too much time to finish.
DISPATCHER_QUEUE_SEMAPHORE	Occurs when a thread from the dispatcher pool is waiting for more work to process. The wait time for this wait type is expected to increase when the dispatcher is idle.
DLL_LOADING_MUTEX	Occurs once while waiting for the XML parser DLL to load.
DROPTEMP	Occurs between attempts to drop a temporary object if the previous attempt failed. The wait duration grows exponentially with each failed drop attempt.
DTC	Occurs when a task is waiting on an event that is used to manage state transition. This state controls when the recovery of Microsoft Distributed Transaction Coordinator (MS DTC) transactions occurs after SQL Server receives notification that the MS DTC service has become unavailable. This state also describes a task that is waiting

Wait type	Description
	when a commit of a MS DTC transaction is initiated by SQL Server and SQL Server is waiting for the MS DTC commit to finish.
DTC_ABORT_REQUEST	Occurs in a MS DTC worker session when the session is waiting to take ownership of a MS DTC transaction. After MS DTC owns the transaction, the session can roll back the transaction. Generally, the session will wait for another session that is using the transaction.
DTC_RESOLVE	Occurs when a recovery task is waiting for the master database in a cross-database transaction so that the task can query the outcome of the transaction.
DTC_STATE	Occurs when a task is waiting on an event that protects changes to the internal MS DTC global state object. This state should be held for very short periods of time.
DTC_TMDOWN_REQUEST	Occurs in a MS DTC worker session when SQL Server receives notification that the MS DTC service is not available. First, the worker will wait for the MS DTC recovery process to start. Then, the worker waits to obtain the outcome of the distributed transaction that the worker is working on. This may continue until the connection with the MS DTC service has been reestablished.
DTC_WAITFOR_OUTCOME	Occurs when recovery tasks wait for MS DTC to become active to enable the resolution of prepared transactions.
DUMP_LOG_COORDINATOR	Occurs when a main task is waiting for a subtask to generate data. Ordinarily, this state does not occur. A long wait

Wait type	Description
	indicates an unexpected blockage. The subtask should be investigated.
DUMPTRIGGER	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
EC	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
EE_PMOLOCK	Occurs during synchronization of certain types of memory allocations during statement execution.
EE_SPECPROC_MAP_INIT	Occurs during synchronization of internal procedure hash table creation. This wait can only occur during the initial accessing of the hash table after the SQL Server instance starts.
ENABLE_VERSIONING	Occurs when SQL Server waits for all update transactions in this database to finish before declaring the database ready to transition to snapshot isolation allowed state. This state is used when SQL Server enables snapshot isolation by using the ALTER DATABASE statement.
ERROR_REPORTING_MANAGER	Occurs during synchronization of multiple concurrent error log initializations.
EXCHANGE	Occurs during synchronization in the query processor exchange iterator during parallel queries.
EXECSYNC	Occurs during parallel queries while synchronizing in query processor in areas not related to the exchange iterator. Examples of such areas are bitmaps, large binary objects (LOBs), and the spool

Wait type	Description
	iterator. LOBs may frequently use this wait state.
EXECUTION_PIPE_EVENT_INTERNAL	Occurs during synchronization between producer and consumer parts of batch execution that are submitted through the connection context.
FAILPOINT	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
FCB_REPLICA_READ	Occurs when the reads of a snapshot (or a temporary snapshot created by DBCC) sparse file are synchronized.
FCB_REPLICA_WRITE	Occurs when the pushing or pulling of a page to a snapshot (or a temporary snapshot created by DBCC) sparse file is synchronized.
FS_FC_RWLOCK	Occurs when there is a wait by the FILESTREAM garbage collector to do either of the following: (1) Disable garbage collection (used by backup and restore) (2) Execute one cycle of the FILESTREAM garbage collector.
FS_GARBAGE_COLLECTOR_SHUTDOWN	Occurs when the FILESTREAM garbage collector is waiting for cleanup tasks to be completed.
FS_HEADER_RWLOCK	Occurs when there is a wait to acquire access to the FILESTREAM header of a FILESTREAM data container to either read or update contents in the FILESTREAM header file (Filestream.hdr).
FS_LOGTRUNC_RWLOCK	Occurs when there is a wait to acquire access to FILESTREAM log truncation to do either of the following: (1) Temporarily disable FILESTREAM log (FSLOG) truncation (used by backup and restore)

Wait type	Description
	(2) Execute one cycle of FSLOG truncation.
FSA_FORCE_OWN_XACT	Occurs when a FILESTREAM file I/O operation needs to bind to the associated transaction, but the transaction is currently owned by another session.
FSAGENT	Occurs when a FILESTREAM file I/O operation is waiting for a FILESTREAM agent resource that is being used by another file I/O operation.
FSTR_CONFIG_MUTEX	Occurs when there is a wait for another FILESTREAM feature reconfiguration to be completed.
FSTR_CONFIG_RWLOCK	Occurs when there is a wait to serialize access to the FILESTREAM configuration parameters.
FT_METADATA_MUTEX	Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_RESTART_CRAWL	Occurs when a full-text crawl needs to restart from a last known good point to recover from a transient failure. The wait lets the worker tasks currently working on that population to complete or exit the current step.
FULLTEXT GATHERER	Occurs during synchronization of full-text operations.
GUARDIAN	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
HTTP_ENUMERATION	Occurs at startup to enumerate the HTTP endpoints to start HTTP.

Wait type	Description
HTTP_START	Occurs when a connection is waiting for HTTP to complete initialization.
IMPPROV_IOWAIT	Occurs when SQL Server waits for a bulkload I/O to finish.
INTERNAL_TESTING	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
IO_AUDIT_MUTEX	Occurs during synchronization of trace event buffers.
IO_COMPLETION	Occurs while waiting for I/O operations to complete. This wait type generally represents non-data page I/Os. Data page I/O completion waits appear as PAGEIOLATCH_* waits.
IO_QUEUE_LIMIT	Occurs when the asynchronous IO queue for the Azure SQL Database has too many IOs pending. Tasks trying to issue another IO are blocked on this wait type until the number of pending IOs drop below the threshold. The threshold is proportional to the DTUs assigned to the database.
IO_RETRY	Occurs when an I/O operation such as a read or a write to disk fails because of insufficient resources, and is then retried.
IOAFF_RANGE_QUEUE	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
KSOURCE_WAKEUP	Used by the service control task while waiting for requests from the Service Control Manager. Long waits are expected and do not indicate a problem.
KTM_ENLISTMENT	Identified for informational purposes only. Not supported. Future compatibility

Wait type	Description
	is not guaranteed.
KTM_RECOVERY_MANAGER	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
KTM_RECOVERY_RESOLUTION	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
LATCH_DT	Occurs when waiting for a DT (destroy) latch. This does not include buffer latches or transaction mark latches. A listing of LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LATCH_EX	Occurs when waiting for an EX (exclusive) latch. This does not include buffer latches or transaction mark latches. A listing of LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LATCH_KP	Occurs when waiting for a KP (keep) latch. This does not include buffer latches or transaction mark latches. A listing of LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LATCH_NL	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
LATCH_SH	Occurs when waiting for an SH (share) latch. This does not include buffer latches or transaction mark latches. A listing of

Wait type	Description
	LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LATCH_UP	Occurs when waiting for an UP (update) latch. This does not include buffer latches or transaction mark latches. A listing of LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LAZYWRITER_SLEEP	Occurs when lazywriter tasks are suspended. This is a measure of the time spent by background tasks that are waiting. Do not consider this state when you are looking for user stalls.
LCK_M_BU	Occurs when a task is waiting to acquire a Bulk Update (BU) lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_IS	Occurs when a task is waiting to acquire an Intent Shared (IS) lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_IU	Occurs when a task is waiting to acquire an Intent Update (IU) lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_IX	Occurs when a task is waiting to acquire an Intent Exclusive (IX) lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_RIn_NL	Occurs when a task is waiting to acquire a NULL lock on the current key value, and an Insert Range lock between the current

Wait type	Description
	and previous key. A NULL lock on the key is an instant release lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_RIn_S	Occurs when a task is waiting to acquire a shared lock on the current key value, and an Insert Range lock between the current and previous key. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_RIn_U	Task is waiting to acquire an Update lock on the current key value, and an Insert Range lock between the current and previous key. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_RIn_X	Occurs when a task is waiting to acquire an Exclusive lock on the current key value, and an Insert Range lock between the current and previous key. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_RS_S	Occurs when a task is waiting to acquire a Shared lock on the current key value, and a Shared Range lock between the current and previous key. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_RS_U	Occurs when a task is waiting to acquire an Update lock on the current key value, and an Update Range lock between the current and previous key. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_RX_S	Occurs when a task is waiting to acquire a Shared lock on the current key value, and an Exclusive Range lock between the current and previous key. For a lock

Wait type	Description
	compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_RX_U	Occurs when a task is waiting to acquire an Update lock on the current key value, and an Exclusive range lock between the current and previous key. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_RX_X	Occurs when a task is waiting to acquire an Exclusive lock on the current key value, and an Exclusive Range lock between the current and previous key. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_S	Occurs when a task is waiting to acquire a Shared lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_SCH_M	Occurs when a task is waiting to acquire a Schema Modify lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_SCH_S	Occurs when a task is waiting to acquire a Schema Share lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_SIU	Occurs when a task is waiting to acquire a Shared With Intent Update lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_SIX	Occurs when a task is waiting to acquire a Shared With Intent Exclusive lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_U	Occurs when a task is waiting to acquire an Update lock. For a lock compatibility

Wait type	Description
	matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_UIX	Occurs when a task is waiting to acquire an Update With Intent Exclusive lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LCK_M_X	Occurs when a task is waiting to acquire an Exclusive lock. For a lock compatibility matrix, see sys.dm_tran_locks (Transact-SQL).
LOG_RATE_GOVERNOR	Occurs when DB is waiting for quota to write to the log.
LOGBUFFER	Occurs when a task is waiting for space in the log buffer to store a log record. Consistently high values may indicate that the log devices cannot keep up with the amount of log being generated by the server.
LOGGENERATION	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
LOGMGR	Occurs when a task is waiting for any outstanding log I/Os to finish before shutting down the log while closing the database.
LOGMGR_FLUSH	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
LOGMGR_QUEUE	Occurs while the log writer task waits for work requests.
LOGMGR_RESERVE_APPEND	Occurs when a task is waiting to see whether log truncation frees up log space to enable the task to write a new log record. Consider increasing the size

Wait type	Description
	of the log file(s) for the affected database to reduce this wait.
LOWFAIL_MEMMGR_QUEUE	Occurs while waiting for memory to be available for use.
MSQL_DQ	Occurs when a task is waiting for a distributed query operation to finish. This is used to detect potential Multiple Active Result Set (MARS) application deadlocks. The wait ends when the distributed query call finishes.
MSQL_XACT_MGR_MUTEX	Occurs when a task is waiting to obtain ownership of the session transaction manager to perform a session level transaction operation.
MSQL_XACT_MUTEX	Occurs during synchronization of transaction usage. A request must acquire the mutex before it can use the transaction.
MSQL_XP	Occurs when a task is waiting for an extended stored procedure to end. SQL Server uses this wait state to detect potential MARS application deadlocks. The wait stops when the extended stored procedure call ends.
MSSEARCH	Occurs during Full-Text Search calls. This wait ends when the full-text operation completes. It does not indicate contention, but rather the duration of full-text operations.
NET_WAITFOR_PACKET	Occurs when a connection is waiting for a network packet during a network read.
OLEDB	Occurs when SQL Server calls the SQL Server Native Client OLE DB Provider. This wait type is not used for

Wait type	Description
	synchronization. Instead, it indicates the duration of calls to the OLE DB provider.
ONDEMAND_TASK_QUEUE	Occurs while a background task waits for high priority system task requests. Long wait times indicate that there have been no high priority requests to process, and should not cause concern.
PAGEIOLATCH_DT	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Destroy mode. Long waits may indicate problems with the disk subsystem.
PAGEIOLATCH_EX	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Exclusive mode. Long waits may indicate problems with the disk subsystem.
PAGEIOLATCH_KP	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Keep mode. Long waits may indicate problems with the disk subsystem.
PAGEIOLATCH_NL	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PAGEIOLATCH_SH	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Shared mode. Long waits may indicate problems with the disk subsystem.
PAGEIOLATCH_UP	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Update mode. Long waits may indicate problems with the disk subsystem.

Wait type	Description
PAGELATCH_DT	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Destroy mode.
PAGELATCH_EX	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Exclusive mode.
PAGELATCH_KP	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Keep mode.
PAGELATCH_NL	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PAGELATCH_SH	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Shared mode.
PAGELATCH_UP	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Update mode.
PARALLEL_BACKUP_QUEUE	Occurs when serializing output produced by RESTORE HEADERONLY, RESTORE FILELISTONLY, or RESTORE LABELONLY.
PREEMPTIVE_ABR	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PREEMPTIVE_AUDIT_ACCESS_EVENTLOG	Occurs when the SQL Server Operating System (SQLOS) scheduler switches to preemptive mode to write an audit event to the Windows event log.
PREEMPTIVE_AUDIT_ACCESS_SECLOG	Occurs when the SQLOS scheduler switches to preemptive mode to write an audit event to the Windows Security log.
PREEMPTIVE_CLOSEBACKUPMEDIA	Occurs when the SQLOS scheduler switches to preemptive mode to close

Wait type	Description
	backup media.
PREEMPTIVE_CLOSEBACKUPTAPE	Occurs when the SQLOS scheduler switches to preemptive mode to close a tape backup device.
PREEMPTIVE_CLOSEBACKUPVDIDEVICE	Occurs when the SQLOS scheduler switches to preemptive mode to close a virtual backup device.
PREEMPTIVE_CLUSAPI_CLUSTERRESOURCECONTROL	Occurs when the SQLOS scheduler switches to preemptive mode to perform Windows failover cluster operations.
PREEMPTIVE_COM_COCREATEINSTANCE	Occurs when the SQLOS scheduler switches to preemptive mode to create a COM object.
PREEMPTIVE_HADR_LEASE_MECHANISM	Always On Availability Groups lease manager scheduling for CSS diagnostics.
PREEMPTIVE_SOSTESTING	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PREEMPTIVE_STRESSDRIVER	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PREEMPTIVE_TESTING	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PREEMPTIVE_XETESTING	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PRINT_ROLLBACK_PROGRESS	Used to wait while user processes are ended in a database that has been transitioned by using the ALTER DATABASE termination clause. For more information, see ALTER DATABASE (Transact-SQL).

Wait type	Description
PWAIT_HADR_CHANGE_NOTIFIER_TERMINATION_SYNC	Occurs when a background task is waiting for the termination of the background task that receives (via polling) Windows Server Failover Clustering notifications. Internal use only.
PWAIT_HADR_CLUSTER_INTEGRATION	An append, replace, and/or remove operation is waiting to grab a write lock on an Always On internal list (such as a list of networks, network addresses, or availability group listeners). Internal use only.
PWAIT_HADR_OFFLINE_COMPLETED	An Always On drop availability group operation is waiting for the target availability group to go offline before destroying Windows Server Failover Clustering objects.
PWAIT_HADR_ONLINE_COMPLETED	An Always On create or failover availability group operation is waiting for the target availability group to come online.
PWAIT_HADR_POST_ONLINE_COMPLETED	An Always On drop availability group operation is waiting for the termination of any background task that was scheduled as part of a previous command. For example, there may be a background task that is transitioning availability databases to the primary role. The DROP AVAILABILITY GROUP DDL must wait for this background task to terminate in order to avoid race conditions.
PWAIT_HADR_WORKITEM_COMPLETED	Internal wait by a thread waiting for an async work task to complete. This is an expected wait and is for CSS use.
PWAIT_MD_LOGIN_STATS	Occurs during internal synchronization in metadata on login stats.

Wait type	Description
PWAIT_MD_RELATION_CACHE	Occurs during internal synchronization in metadata on table or index.
PWAIT_MD_SERVER_CACHE	Occurs during internal synchronization in metadata on linked servers.
PWAIT_MD_UPGRADE_CONFIG	Occurs during internal synchronization in upgrading server wide configurations.
PWAIT_METADATA_LAZYCACHE_RWLOCK	Occurs during internal synchronization in metadata cache along with iterating index or stats in a table.
QPJOB_KILL	Indicates that an asynchronous automatic statistics update was canceled by a call to KILL as the update was starting to run. The terminating thread is suspended, waiting for it to start listening for KILL commands. A good value is less than one second.
QPJOB_WAITFOR_ABORT	Indicates that an asynchronous automatic statistics update was canceled by a call to KILL when it was running. The update has now completed but is suspended until the terminating thread message coordination is complete. This is an ordinary but rare state, and should be very short. A good value is less than one second.
QRY_MEM_GRANT_INFO_MUTEX	Occurs when Query Execution memory management tries to control access to static grant information list. This state lists information about the current granted and waiting memory requests. This state is a simple access control state. There should never be a long wait on this state. If this mutex is not released, all new memory-using queries will stop responding.

Wait type	Description
QUERY_ERRHDL_SERVICE_DONE	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
QUERY_EXECUTION_INDEX_SORT_EVENT_OPEN	Occurs in certain cases when offline create index build is run in parallel, and the different worker threads that are sorting synchronize access to the sort files.
QUERY_NOTIFICATION_MGR_MUTEX	Occurs during synchronization of the garbage collection queue in the Query Notification Manager.
QUERY_NOTIFICATION_SUBSCRIPTION_MUTEX	Occurs during state synchronization for transactions in Query Notifications.
QUERY_NOTIFICATION_TABLE_MGR_MUTEX	Occurs during internal synchronization within the Query Notification Manager.
QUERY_NOTIFICATION_UNITTEST_MUTEX	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
QUERY_OPTIMIZER_PRINT_MUTEX	Occurs during synchronization of query optimizer diagnostic output production. This wait type only occurs if diagnostic settings have been enabled under direction of Microsoft Product Support.
QUERY_TRACEOUT	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
QUERY_WAIT_ERRHDL_SERVICE	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
RECOVER_CHANGEDB	Occurs during synchronization of database status in warm standby database.

Wait type	Description
REPL_CACHE_ACCESS	Occurs during synchronization on a replication article cache. During these waits, the replication log reader stalls, and data definition language (DDL) statements on a published table are blocked.
REPL_SCHEMA_ACCESS	Occurs during synchronization of replication schema version information. This state exists when DDL statements are executed on the replicated object, and when the log reader builds or consumes versioned schema based on DDL occurrence.
REPLICA_WRITES	Occurs while a task waits for completion of page writes to database snapshots or DBCC replicas.
REQUEST_DISPENSER_PAUSE	Occurs when a task is waiting for all outstanding I/O to complete, so that I/O to a file can be frozen for snapshot backup.
REQUEST_FOR_DEADLOCK_SEARCH	Occurs while the deadlock monitor waits to start the next deadlock search. This wait is expected between deadlock detections, and lengthy total waiting time on this resource does not indicate a problem.
RESMGR_THROTTLED	Occurs when a new request comes in and is throttled based on the GROUP_MAX_REQUESTS setting.
RESOURCE_QUEUE	Occurs during synchronization of various internal resource queues.
RESOURCE_SEMAPHORE	Occurs when a query memory request cannot be granted immediately due to other concurrent queries. High waits and wait times may indicate excessive

Wait type	Description
	number of concurrent queries, or excessive memory request amounts.
RESOURCE_SEMAPHORE_MUTEX	Occurs while a query waits for its request for a thread reservation to be fulfilled. It also occurs when synchronizing query compile and memory grant requests.
RESOURCE_SEMAPHORE_QUERY_COMPILE	Occurs when the number of concurrent query compilations reaches a throttling limit. High waits and wait times may indicate excessive compilations, recompiles, or uncachable plans.
RESOURCE_SEMAPHORE_SMALL_QUERY	Occurs when memory request by a small query cannot be granted immediately due to other concurrent queries. Wait time should not exceed more than a few seconds, because the server transfers the request to the main query memory pool if it fails to grant the requested memory within a few seconds. High waits may indicate an excessive number of concurrent small queries while the main memory pool is blocked by waiting queries.
SE_REPL_CATCHUP_THROTTLE	Occurs when the transaction is waiting for one of the database secondaries to make progress.
SE_REPL_COMMIT_ACK	Occurs when the transaction is waiting for quorum commit acknowledgment from secondary replicas.
SE_REPL_COMMIT_TURN	Occurs when the transaction is waiting for commit after receiving quorum commit acknowledgments.
SE_REPL_ROLLBACK_ACK	Occurs when the transaction is waiting for quorum rollback acknowledgment from secondary replicas.

Wait type	Description
SE_REPL_SLOW_SECONDARY_THROTTLE	Occurs when the thread is waiting for one of the database secondary replicas.
SEC_DROP_TEMP_KEY	Occurs after a failed attempt to drop a temporary security key before a retry attempt.
SECURITY_MUTEX	Occurs when there is a wait for mutexes that control access to the global list of Extensible Key Management (EKM) cryptographic providers and the session-scoped list of EKM sessions.
SEQUENTIAL_GUID	Occurs while a new sequential GUID is being obtained.
SERVER_IDLE_CHECK	Occurs during synchronization of SQL Server instance idle status when a resource monitor is attempting to declare a SQL Server instance as idle or trying to wake up.
SHUTDOWN	Occurs while a shutdown statement waits for active connections to exit.
SLEEP_BPOOL_FLUSH	Occurs when a checkpoint is throttling the issuance of new I/Os in order to avoid flooding the disk subsystem.
SLEEP_DBSTARTUP	Occurs during database startup while waiting for all databases to recover.
SLEEP_DCOMSTARTUP	Occurs once at most during SQL Server instance startup while waiting for DCOM initialization to complete.
SLEEP_MSDBSTARTUP	Occurs when SQL Trace waits for the msdb database to complete startup.
SLEEP_SYSTEMTASK	Occurs during the start of a background task while waiting for tempdb to complete startup.

Wait type	Description
SLEEP_TASK	Occurs when a task sleeps while waiting for a generic event to occur.
SLEEP_TEMPDBSTARTUP	Occurs while a task waits for tempdb to complete startup.
SNI_CRITICAL_SECTION	Occurs during internal synchronization within SQL Server networking components.
SNI_HTTP_WAITFOR_0_DISCON	Occurs during SQL Server shutdown, while waiting for outstanding HTTP connections to exit.
SNI_LISTENER_ACCESS	Occurs while waiting for non-uniform memory access (NUMA) nodes to update state change. Access to state change is serialized.
SNI_TASK_COMPLETION	Occurs when there is a wait for all tasks to finish during a NUMA node state change.
SOAP_READ	Occurs while waiting for an HTTP network read to complete.
SOAP_WRITE	Occurs while waiting for an HTTP network write to complete.
SOS_CALLBACK_REMOVAL	Occurs while performing synchronization on a callback list in order to remove a callback. It is not expected for this counter to change after server initialization is completed.
SOS_DISPATCHER_MUTEX	Occurs during internal synchronization of the dispatcher pool. This includes when the pool is being adjusted.
SOS_LOCALALLOCATORLIST	Occurs during internal synchronization in the SQL Server memory manager.

Wait type	Description
SOS_MEMORY_USAGE_ADJUSTMENT	Occurs when memory usage is being adjusted among pools.
SOS_OBJECT_STORE_DESTROY_MUTEX	Occurs during internal synchronization in memory pools when destroying objects from the pool.
SOS_PROCESS_AFFINITY_MUTEX	Occurs during synchronizing of access to process affinity settings.
SOS_RESERVEDMEMBLOCKLIST	Occurs during internal synchronization in the SQL Server memory manager.
SOS_SCHEDULER_YIELD	Occurs when a task voluntarily yields the scheduler for other tasks to execute. During this wait the task is waiting for its quantum to be renewed.
SOS_SMALL_PAGE_ALLOC	Occurs during the allocation and freeing of memory that is managed by some memory objects.
SOS_STACKSTORE_INIT_MUTEX	Occurs during synchronization of internal store initialization.
SOS_SYNC_TASK_ENQUEUE_EVENT	Occurs when a task is started in a synchronous manner. Most tasks in SQL Server are started in an asynchronous manner, in which control returns to the starter immediately after the task request has been placed on the work queue.
SOS_VIRTUALMEMORY_LOW	Occurs when a memory allocation waits for a resource manager to free up virtual memory.
SOSHOST_EVENT	Occurs when a hosted component, such as CLR, waits on a SQL Server event synchronization object.
SOSHOST_INTERNAL	Occurs during synchronization of memory manager callbacks used by hosted components, such as CLR.

Wait type	Description
SOSHOST_MUTEX	Occurs when a hosted component, such as CLR, waits on a SQL Server mutex synchronization object.
SOSHOST_RWLOCK	Occurs when a hosted component, such as CLR, waits on a SQL Server reader-writer synchronization object.
SOSHOST_SEMAPHORE	Occurs when a hosted component, such as CLR, waits on a SQL Server semaphore synchronization object.
SOSHOST_SLEEP	Occurs when a hosted task sleeps while waiting for a generic event to occur. Hosted tasks are used by hosted components such as CLR.
SOSHOST_TRACELOCK	Occurs during synchronization of access to trace streams.
SOSHOST_WAITFORDONE	Occurs when a hosted component, such as CLR, waits for a task to complete.
SQLCLR_APPDOMAIN	Occurs while CLR waits for an application domain to complete startup.
SQLCLR_ASSEMBLY	Occurs while waiting for access to the loaded assembly list in the appdomain.
SQLCLR_DEADLOCK_DETECTION	Occurs while CLR waits for deadlock detection to complete.
SQLCLR_QUANTUM_PUNISHMENT	Occurs when a CLR task is throttled because it has exceeded its execution quantum. This throttling is done in order to reduce the effect of this resource-intensive task on other tasks.
SQLSORT_NORMMUTEX	Occurs during internal synchronization, while initializing internal sorting structures.

Wait type	Description
SQLSORT_SORTMUTEX	Occurs during internal synchronization, while initializing internal sorting structures.
SQLTRACE_BUFFER_FLUSH	Occurs when a task is waiting for a background task to flush trace buffers to disk every four seconds.
SQLTRACE_LOCK	Occurs during synchronization on trace buffers during a file trace.
SQLTRACE_SHUTDOWN	Occurs while trace shutdown waits for outstanding trace events to complete.
SQLTRACE_WAIT_ENTRIES	Occurs while a SQL Trace event queue waits for packets to arrive on the queue.
SRVPROC_SHUTDOWN	Occurs while the shutdown process waits for internal resources to be released to shutdown cleanly.
TEMPOBJ	Occurs when temporary object drops are synchronized. This wait is rare, and only occurs if a task has requested exclusive access for temp table drops.
THREADPOOL	Occurs when a task is waiting for a worker to run on. This can indicate that the maximum worker setting is too low, or that batch executions are taking unusually long, thus reducing the number of workers available to satisfy other batches.
TIMEPRIV_TIMEPERIOD	Occurs during internal synchronization of the Extended Events timer.
TRACEWRITE	Occurs when the SQL Trace rowset trace provider waits for either a free buffer or a buffer with events to process.
TRAN_MARKLATCH_DT	Occurs when waiting for a destroy mode latch on a transaction mark latch.



Wait type	Description
	Transaction mark latches are used for synchronization of commits with marked transactions.
TRAN_MARKLATCH_EX	Occurs when waiting for an exclusive mode latch on a marked transaction. Transaction mark latches are used for synchronization of commits with marked transactions.
TRAN_MARKLATCH_KP	Occurs when waiting for a keep mode latch on a marked transaction. Transaction mark latches are used for synchronization of commits with marked transactions.
TRAN_MARKLATCH_NL	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
TRAN_MARKLATCH_SH	Occurs when waiting for a shared mode latch on a marked transaction. Transaction mark latches are used for synchronization of commits with marked transactions.
TRAN_MARKLATCH_UP	Occurs when waiting for an update mode latch on a marked transaction. Transaction mark latches are used for synchronization of commits with marked transactions.
TRANSACTION_MUTEX	Occurs during synchronization of access to a transaction by multiple batches.
UTIL_PAGE_ALLOC	Occurs when transaction log scans wait for memory to be available during memory pressure.
VIA_ACCEPT	Occurs when a Virtual Interface Adapter (VIA) provider connection is completed during startup.

Wait type	Description
VIEW_DEFINITION_MUTEX	Occurs during synchronization on access to cached view definitions.
WAIT_FOR_RESULTS	Occurs when waiting for a query notification to be triggered.
WAITFOR	Occurs as a result of a WAITFOR Transact-SQL statement. The duration of the wait is determined by the parameters to the statement. This is a user-initiated wait.
WAITFOR_TASKSHUTDOWN	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
WAITSTAT_MUTEX	Occurs during synchronization of access to the collection of statistics used to populate sys.dm_os_wait_stats.
WCC	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
WORKTBL_DROP	Occurs while pausing before retrying, after a failed worktable drop.
WRITE_COMPLETION	Occurs when a write operation is in progress.
WRITELOG	Occurs while waiting for a log flush to complete. Common operations that cause log flushes are checkpoints and transaction commits.
XACT_OWN_TRANSACTION	Occurs while waiting to acquire ownership of a transaction.
XACT_RECLAIM_SESSION	Occurs while waiting for the current owner of a session to release ownership of the session.

Wait type	Description
XACTLOCKINFO	Occurs during synchronization of access to the list of locks for a transaction. In addition to the transaction itself, the list of locks is accessed by operations such as deadlock detection and lock migration during page splits.
XACTWORKSPACE_MUTEX	Occurs during synchronization of defections from a transaction, as well as the number of database locks between enlist members of a transaction.
XE_BUFFERMGR_ALLPROCESSED_EVENT	Occurs when Extended Events session buffers are flushed to targets. This wait occurs on a background thread.
XE_BUFFERMGR_FREEBUF_EVENT	Occurs when either of the following conditions is true: (1) An Extended Events session is configured for no event loss, and all buffers in the session are currently full. This can indicate that the buffers for an Extended Events session are too small, or should be partitioned. (2) Audits experience a delay. This can indicate a disk bottleneck on the drive where the audits are written.
XE_DISPATCHER_CONFIG_SESSION_LIST	Occurs when an Extended Events session that is using asynchronous targets is started or stopped. This wait indicates either of the following: (1) An Extended Events session is registering with a background thread pool. (2) The background thread pool is calculating the required number of threads based on current load.
XE_DISPATCHER_JOIN	Occurs when a background thread that is used for Extended Events sessions is terminating.
XE_DISPATCHER_WAIT	Occurs when a background thread that is used for Extended Events sessions is

Wait type	Description
	waiting for event buffers to process.
XE_MODULEMGR_SYNC	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
XE_OLS_LOCK	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
XE_PACKAGE_LOCK_BACKOFF	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_COMPROWSET_RWLOCK	Full-text is waiting on fragment metadata operation. Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_IPTS_RWLOCK	Full-text is waiting on internal synchronization. Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_IPTS_SCHEDULER_IDLE_WAIT	Full-text scheduler sleep wait type. The scheduler is idle.
FT_IPTSHC_MUTEX	Full-text is waiting on an fdhost control operation. Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_IPTSISM_MUTEX	Full-text is waiting on communication operation. Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_MASTER_MERGE	Full-text is waiting on master merge operation. Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.

OS Wait Stats

The main source of the values in this table are from the [sys.dm_os_wait_stats \(Transact-SQL\)](#)  documentation. Additional entries can also be added manually. There are several entries marked as "Internal Use Only", for which there is no public documentation. Take a look at the wait type name and you might be able to guess its meaning. You can also search on the [SQLSkills external site](#)  which has some additional information (source and quality unknown).

Wait Type	Description
ABR	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
ASSEMBLY_LOAD	Occurs during exclusive access to assembly loading.
ASYNC_DISKPOOL_LOCK	Occurs when there is an attempt to synchronize parallel threads that are performing tasks such as creating or initializing a file.
ASYNC_IO_COMPLETION	Occurs when a task is waiting for asynchronous non-data I/Os to finish. Examples include I/O involved in warm standby log shipping, database mirroring, some bulk import related operations.
ASYNC_NETWORK_IO	Occurs on network writes when the task is blocked waiting for the client application to acknowledge it has processed all the data sent to it. Verify that the client application is processing data from the server as fast as possible or that no network delays exist. Reasons the client application cannot consume data fast enough include: application design issues like writing results to a file while the results arrive, waiting for user input, client-side filtering on a large dataset instead of server-side filtering, or an intentional wait introduced. Also the client computer may be experiencing slow response due to issues like low virtual/physical memory, 100% CPU consumption, etc. Network delays can also lead to

Wait Type	Description
	this wait - typically caused by network adapter driver issues, filter drivers, firewalls or misconfigured routers.
ASYNC_OP_COMPLETION	Internal use only.
ASYNC_OP_CONTEXT_READ	Internal use only.
ASYNC_OP_CONTEXT_WRITE	Internal use only.
ASYNC_SOCKETDUP_IO	Internal use only.
AUDIT_GROUPCACHE_LOCK	Occurs when there is a wait on a lock that controls access to a special cache. The cache contains information about which audits are being used to audit each audit action group.
AUDIT_LOGINCACHE_LOCK	Occurs when there is a wait on a lock that controls access to a special cache. The cache contains information about which audits are being used to audit login audit action groups.
AUDIT_ON_DEMAND_TARGET_LOCK	Occurs when there is a wait on a lock that is used to ensure single initialization of audit related Extended Event targets.
AUDIT_XE_SESSION_MGR	Occurs when there is a wait on a lock that is used to synchronize the starting and stopping of audit related Extended Events sessions.
BACKUP	Occurs when a task is blocked as part of backup processing.
BACKUP_OPERATOR	Occurs when a task is waiting for a tape mount. To view the tape status, query sys.dm_io_backup_tapes. If

Wait Type	Description
	a mount operation is not pending, this wait type may indicate a hardware problem with the tape drive.
BACKUPBUFFER	Occurs when a backup task is waiting for data, or is waiting for a buffer in which to store data. This type is not typical, except when a task is waiting for a tape mount.
BACKUPIO	Occurs when a backup task is waiting for data, or is waiting for a buffer in which to store data. This type is not typical, except when a task is waiting for a tape mount.
BACKUPTHREAD	Occurs when a task is waiting for a backup task to finish. Wait times may be long, from several minutes to several hours. If the task that is being waited on is in an I/O process, this type does not indicate a problem.
BAD_PAGE_PROCESS	Occurs when the background suspect page logger is trying to avoid running more than every five seconds. Excessive suspect pages cause the logger to run frequently.
BLOB_METADATA	Internal use only.
BMPALLOCATION	Occurs with parallel batch-mode plans when synchronizing the allocation of a large bitmap filter. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.

Wait Type	Description
BMPBUILD	Occurs with parallel batch-mode plans when synchronizing the building of a large bitmap filter. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.
BMPREPARTITION	Occurs with parallel batch-mode plans when synchronizing the repartitioning of a large bitmap filter. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.
BMPREPLICATION	Occurs with parallel batch-mode plans when synchronizing the replication of a large bitmap filter across worker threads. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.
BPSORT	Occurs with parallel batch-mode plans when synchronizing the sorting of a dataset across multiple threads. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.

Wait Type	Description
BROKER_CONNECTION_RECEIVE_TASK	Occurs when waiting for access to receive a message on a connection endpoint. Receive access to the endpoint is serialized.
BROKER_DISPATCHER	Internal use only.
BROKER_ENDPOINT_STATE_MUTEX	Occurs when there is contention to access the state of a Service Broker connection endpoint. Access to the state for changes is serialized.
BROKER_EVENTHANDLER	Occurs when a task is waiting in the primary event handler of the Service Broker. This should occur very briefly.
BROKER_FORWARDER	Internal use only.
BROKER_INIT	Occurs when initializing Service Broker in each active database. This should occur infrequently.
BROKER_MASTERSTART	Occurs when a task is waiting for the primary event handler of the Service Broker to start. This should occur very briefly.
BROKER_RECEIVE_WAITFOR	Occurs when the RECEIVE WAITFOR is waiting. This may mean that either no messages are ready to be received in the queue or a lock contention is preventing it from receiving messages from the queue.
BROKER_REGISTERALLENDPOINTS	Occurs during the initialization of a Service Broker connection endpoint. This should occur very briefly.

Wait Type	Description
BROKER_SERVICE	Occurs when the Service Broker destination list that is associated with a target service is updated or re-prioritized.
BROKER_SHUTDOWN	Occurs when there is a planned shutdown of Service Broker. This should occur very briefly, if at all.
BROKER_START	Internal use only.
BROKER_TASK_SHUTDOWN	Internal use only.
BROKER_TASK_STOP	Occurs when the Service Broker queue task handler tries to shut down the task. The state check is serialized and must be in a running state beforehand.
BROKER_TASK_SUBMIT	Internal use only.
BROKER_TO_FLUSH	Occurs when the Service Broker lazy flusher flushes the in-memory transmission objects to a work table.
BROKER_TRANSMISSION_OBJECT	Internal use only.
BROKER_TRANSMISSION_TABLE	Internal use only.
BROKER_TRANSMISSION_WORK	Internal use only.
BROKER_TRANSMITTER	Occurs when the Service Broker transmitter is waiting for work. Service Broker has a component known as the Transmitter which schedules messages from multiple dialogs to be sent across the wire over one or more connection endpoints. The transmitter has 2 dedicated threads for this purpose. This wait type is charged when these transmitter threads

Wait Type	Description
	are waiting for dialog messages to be sent using the transport connections. High values of <code>waiting_tasks_count</code> for this wait type point to intermittent work for these transmitter threads and are not indications of any performance problem. If service broker is not used at all, <code>waiting_tasks_count</code> should be 2 (for the 2 transmitter threads) and <code>wait_time_ms</code> should be twice the duration since instance startup. See Service broker wait stats.
BUILTIN_HASHKEY_MUTEX	May occur after startup of instance, while internal data structures are initializing. Will not recur once data structures have initialized.
CHANGE_TRACKING_WAITFORCHANGES	Internal use only.
CHECK_PRINT_RECORD	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
CHECK_SCANNER_MUTEX	Internal use only.
CHECK_TABLES_INITIALIZATION	Internal use only.
CHECK_TABLES_SINGLE_SCAN	Internal use only.
CHECK_TABLES_THREAD_BARRIER	Internal use only.
CHECKPOINT_QUEUE	Occurs while the checkpoint task is waiting for the next checkpoint request.
CHKPT	Occurs at server startup to tell the checkpoint thread that it can start.

Wait Type	Description
CLEAR_DB	Occurs during operations that change the state of a database, such as opening or closing a database.
CLR_AUTO_EVENT	Occurs when a task is currently performing common language runtime (CLR) execution and is waiting for a particular autoevent to be initiated. Long waits are typical, and do not indicate a problem.
CLR_CRST	Occurs when a task is currently performing CLR execution and is waiting to enter a critical section of the task that is currently being used by another task.
CLR_JOIN	Occurs when a task is currently performing CLR execution and waiting for another task to end. This wait state occurs when there is a join between tasks.
CLR_MANUAL_EVENT	Occurs when a task is currently performing CLR execution and is waiting for a specific manual event to be initiated.
CLR_MEMORY_SPY	Occurs during a wait on lock acquisition for a data structure that is used to record all virtual memory allocations that come from CLR. The data structure is locked to maintain its integrity if there is parallel access.
CLR_MONITOR	Occurs when a task is currently performing CLR execution and is waiting to obtain a lock on the monitor.

Wait Type	Description
CLR_RWLOCK_READER	Occurs when a task is currently performing CLR execution and is waiting for a reader lock.
CLR_RWLOCK_WRITER	Occurs when a task is currently performing CLR execution and is waiting for a writer lock.
CLR_SEMAPHORE	Occurs when a task is currently performing CLR execution and is waiting for a semaphore.
CLR_TASK_START	Occurs while waiting for a CLR task to complete startup.
CLRHOST_STATE_ACCESS	Occurs where there is a wait to acquire exclusive access to the CLR-hosting data structures. This wait type occurs while setting up or tearing down the CLR runtime.
CMEMPARTITIONED	Internal use only.
CMEMTHREAD	Occurs when a task is waiting on a thread-safe memory object. The wait time might increase when there is contention caused by multiple tasks trying to allocate memory from the same memory object.
COLUMNSTORE_BUILD_THROTTLE	Internal use only.
COLUMNSTORE_COLUMNDATASET_SESSION_LIST	Internal use only.
COMMIT_TABLE	Internal use only.
CONNECTION_ENDPOINT_LOCK	Internal use only.
COUNTRECOVERYMGR	Internal use only.
CREATE_DATINISERVICE	Internal use only.

Wait Type	Description
CXCONSUMER	Occurs with parallel query plans when a consumer thread (parent) waits for a producer thread to send rows. CXCONSUMER waits are caused by an Exchange Iterator that runs out of rows from its producer thread. This is a normal part of parallel query execution.
CXPACKET	Occurs with parallel query plans when waiting to synchronize the Query Processor Exchange Iterator, and when producing and consuming rows. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the Cost Threshold for Parallelism or lowering the Max Degree of Parallelism (MaxDOP).
CXSYNC_PORT	Occurs with parallel query plans when waiting to open, close, and synchronize Exchange Iterator ports between producer and consumer threads. For example, if a query plan has a long sort operation, CXSYNC_PORT waits may be higher because the sort must complete before the Exchange Iterator port can be synchronized. Thread execution might be delayed because auto-update stats gets triggered. As first mitigation step, set MAXDOP to 8 (or between 1 and 8) and SET AUTO_UPDATE_STATISTICS_ASYNC ON.
CXSYNC_CONSUMER	Occurs with parallel query plans when waiting to reach

Wait Type	Description
	an Exchange Iterator synchronization point among all consumer threads.
CXROWSET_SYNC	Occurs during a parallel range scan.
DAC_INIT	Occurs while the dedicated administrator connection is initializing.
DBCC_SCALE_OUT_EXPR_CACHE	Internal use only.
DBMIRROR_DBM_EVENT	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
DBMIRROR_DBM_MUTEX	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
DBMIRROR_EVENTS_QUEUE	Occurs when database mirroring waits for events to process.
DBMIRROR_SEND	Occurs when a task is waiting for a communications backlog at the network layer to clear to be able to send messages. Indicates that the communications layer is starting to become overloaded and affect the database mirroring data throughput.
DBMIRROR_WORKER_QUEUE	Indicates that the database mirroring worker task is waiting for more work.
DBMIRRORING_CMD	Occurs when a task is waiting for log records to be flushed to disk. This wait state is expected to be held for long periods of time.

Wait Type	Description
DBSEEDING_FLOWCONTROL	Internal use only.
DBSEEDING_OPERATION	Internal use only.
DEADLOCK_ENUM_MUTEX	Occurs when the deadlock monitor and sys.dm_os_waiting_tasks try to make sure that SQL Server is not running multiple deadlock searches at the same time.
DEADLOCK_TASK_SEARCH	Large waiting time on this resource indicates that the server is executing queries on top of sys.dm_os_waiting_tasks, and these queries are blocking deadlock monitor from running deadlock search. This wait type is used by deadlock monitor only. Queries on top of sys.dm_os_waiting_tasks use DEADLOCK_ENUM_MUTEX.
DEBUG	Occurs during Transact-SQL and CLR debugging for internal synchronization.
DIRECTLOGCONSUMER_LIST	Internal use only.
DIRTY_PAGE_POLL	Internal use only.
DIRTY_PAGE_SYNC	Internal use only.
DIRTY_PAGE_TABLE_LOCK	Internal use only.
DISABLE_VERSIONING	Occurs when SQL Server polls the version transaction manager to see whether the timestamp of the earliest active transaction is later than the timestamp of when the state started changing. If this is this case, all the snapshot transactions that were started

Wait Type	Description
	before the ALTER DATABASE statement was run have finished. This wait state is used when SQL Server disables versioning by using the ALTER DATABASE statement.
DISKIO_SUSPEND	Occurs when a task is waiting to access a file when an external backup is active. This is reported for each waiting user process. A count larger than five per user process may indicate that the external backup is taking too much time to finish.
DISPATCHER_PRIORITY_QUEUE_SEMAPHORE	Internal use only.
DISPATCHER_QUEUE_SEMAPHORE	Occurs when a thread from the dispatcher pool is waiting for more work to process. The wait time for this wait type is expected to increase when the dispatcher is idle.
DLL_LOADING_MUTEX	Occurs once while waiting for the XML parser DLL to load.
DPT_ENTRY_LOCK	Internal use only.
DROP_DATABASE_TIMER_TASK	Internal use only.
DROPTEMP	Occurs between attempts to drop a temporary object if the previous attempt failed. The wait duration grows exponentially with each failed drop attempt.
DTC	Occurs when a task is waiting on an event that is used to manage state transition. This state controls when the recovery of Microsoft Distributed Transaction Coordinator (MS DTC)

Wait Type	Description
	transactions occurs after SQL Server receives notification that the MS DTC service has become unavailable.
DTC_ABORT_REQUEST	Occurs in a MS DTC worker session when the session is waiting to take ownership of a MS DTC transaction. After MS DTC owns the transaction, the session can roll back the transaction. Generally, the session will wait for another session that is using the transaction.
DTC_RESOLVE	Occurs when a recovery task is waiting for the master database in a cross-database transaction so that the task can query the outcome of the transaction.
DTC_STATE	Occurs when a task is waiting on an event that protects changes to the internal MS DTC global state object. This state should be held for very short periods of time.
DTC_TMDOWN_REQUEST	Occurs in a MS DTC worker session when SQL Server receives notification that the MS DTC service is not available. First, the worker will wait for the MS DTC recovery process to start. Then, the worker waits to obtain the outcome of the distributed transaction that the worker is working on. This may continue until the connection with the MS DTC service has been reestablished.
DTC_WAITFOR_OUTCOME	Occurs when recovery tasks wait for MS DTC to become active to

Wait Type	Description
	enable the resolution of prepared transactions.
DTCNEW_ENLIST	Internal use only.
DTCNEW_PREPARE	Internal use only.
DTCNEW_RECOVERY	Internal use only.
DTCNEW_TM	Internal use only.
DTCNEW_TRANSACTION_ENLISTMENT	Internal use only.
DTCPNTSYNC	Internal use only.
DUMP_LOG_COORDINATOR	Occurs when a main task is waiting for a subtask to generate data. Ordinarily, this state does not occur. A long wait indicates an unexpected blockage. The subtask should be investigated.
DUMP_LOG_COORDINATOR_QUEUE	Internal use only.
DUMPTRIGGER	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
EC	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
EE_PMOLOCK	Occurs during synchronization of certain types of memory allocations during statement execution.
EE_SPECPROC_MAP_INIT	Occurs during synchronization of internal procedure hash table creation. This wait can only occur during the initial accessing of the

Wait Type	Description
	hash table after the SQL Server instance starts.
ENABLE_EMPTY_VERSIONING	Internal use only.
ENABLE_VERSIONING	Occurs when SQL Server waits for all update transactions in this database to finish before declaring the database ready to transition to snapshot isolation allowed state. This state is used when SQL Server enables snapshot isolation by using the ALTER DATABASE statement.
ERROR_REPORTING_MANAGER	Occurs during synchronization of multiple concurrent error log initializations.
EXCHANGE	Occurs during synchronization in the query processor exchange iterator during parallel queries.
EXECSYNC	Occurs during parallel queries while synchronizing in query processor in areas not related to the exchange iterator. Examples of such areas are bitmaps, large binary objects (LOBs), and the spool iterator. LOBs may frequently use this wait state.
EXECUTION_PIPE_EVENT_INTERNAL	Occurs during synchronization between producer and consumer parts of batch execution that are submitted through the connection context.
EXTERNAL_RG_UPDATE	Internal use only.
EXTERNAL_SCRIPT_NETWORK_IO	Internal use only.
EXTERNAL_SCRIPT_PREPARE_SERVICE	Internal use only.

Wait Type	Description
EXTERNAL_SCRIPT_SHUTDOWN	Internal use only.
EXTERNAL_WAIT_ON_LAUNCHER,	Internal use only.
FABRIC_HADR_TRANSPORT_CONNECTION	Internal use only.
FABRIC_REPLICA_CONTROLLER_LIST	Internal use only.
FABRIC_REPLICA_CONTROLLER_STATE_AND_CONFIG	Internal use only.
FABRIC_REPLICA_PUBLISHER_EVENT_PUBLISH	Internal use only.
FABRIC_REPLICA_PUBLISHER_SUBSCRIBER_LIST	Internal use only.
FABRIC_WAIT_FOR_BUILD_REPLICA_EVENT_PROCESSING	Internal use only.
FAILPOINT	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
FCB_REPLICA_READ	Occurs when the reads of a snapshot (or a temporary snapshot created by DBCC) sparse file are synchronized.
FCB_REPLICA_WRITE	Occurs when the pushing or pulling of a page to a snapshot (or a temporary snapshot created by DBCC) sparse file is synchronized.
FEATURE_SWITCHES_UPDATE	Internal use only.
FFT_NSO_DB_KILL_FLAG	Internal use only.
FFT_NSO_DB_LIST	Internal use only.
FFT_NSO_FCB	Internal use only.
FFT_NSO_FCB_FIND	Internal use only.
FFT_NSO_FCB_PARENT	Internal use only.

Wait Type	Description
FFT_NSO_FCB_RELEASE_CACHED_ENTRIES	Internal use only.
FFT_NSO_FCB_STATE	Internal use only.
FFT_NSO_FILEOBJECT	Internal use only.
FFT_NSO_TABLE_LIST	Internal use only.
FFT_NTFS_STORE	Internal use only.
FFT_RECOVERY	Internal use only.
FFT_RSFX_COMM	Internal use only.
FFT_RSFX_WAIT_FOR_MEMORY	Internal use only.
FFT_STARTUP_SHUTDOWN	Internal use only.
FFT_STORE_DB	Internal use only.
FFT_STORE_ROWSET_LIST	Internal use only.
FFT_STORE_TABLE	Internal use only.
FILE_VALIDATION_THREADS	Internal use only.
FILESTREAM_CACHE	Internal use only.
FILESTREAM_CHUNKER	Internal use only.
FILESTREAM_CHUNKER_INIT	Internal use only.
FILESTREAM_FCB	Internal use only.
FILESTREAM_FILE_OBJECT	Internal use only.
FILESTREAM_WORKITEM_QUEUE	Internal use only.
FILETABLE_SHUTDOWN	Internal use only.
FOREIGN_REDO	Internal use only.
FORWARDER_TRANSITION	Internal use only.

Wait Type	Description
FS_FC_RWLOCK	Occurs when there is a wait by the FILESTREAM garbage collector to do either of the following:
FS_GARBAGE_COLLECTOR_SHUTDOWN	Occurs when the FILESTREAM garbage collector is waiting for cleanup tasks to be completed.
FS_HEADER_RWLOCK	Occurs when there is a wait to acquire access to the FILESTREAM header of a FILESTREAM data container to either read or update contents in the FILESTREAM header file (Filestream.hdr).
FS_LOGTRUNC_RWLOCK	Occurs when there is a wait to acquire access to FILESTREAM log truncation to do either of the following:
FSA_FORCE_OWN_XACT	Occurs when a FILESTREAM file I/O operation needs to bind to the associated transaction, but the transaction is currently owned by another session.
FSAGENT	Occurs when a FILESTREAM file I/O operation is waiting for a FILESTREAM agent resource that is being used by another file I/O operation.
FSTR_CONFIG_MUTEX	Occurs when there is a wait for another FILESTREAM feature reconfiguration to be completed.
FSTR_CONFIG_RWLOCK	Occurs when there is a wait to serialize access to the FILESTREAM configuration parameters.
FT_COMPROWSET_RWLOCK	Full-text is waiting on fragment metadata operation. Documented for informational purposes only.

Wait Type	Description
	Not supported. Future compatibility is not guaranteed.
FT_ISTS_RWLOCK	Full-text is waiting on internal synchronization. Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_ISTS_SCHEDULER_IDLE_WAIT	Full-text scheduler sleep wait type. The scheduler is idle.
FT_ISTSHC_MUTEX	Full-text is waiting on an fdhost control operation. Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_ISTSISM_MUTEX	Full-text is waiting on communication operation. Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_MASTER_MERGE	Full-text is waiting on master merge operation. Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_MASTER_MERGE_COORDINATOR	Internal use only.
FT_METADATA_MUTEX	Documented for informational purposes only. Not supported. Future compatibility is not guaranteed.
FT_PROPERTYLIST_CACHE	Internal use only.
FT_RESTART_CRAWL	Occurs when a full-text crawl needs to restart from a last known good point to recover from a transient failure. The wait lets the

Wait Type	Description
	worker tasks currently working on that population to complete or exit the current step.
FULLTEXT GATHERER	Occurs during synchronization of full-text operations.
GDMA_GET_RESOURCE_OWNER	Internal use only.
GHOSTCLEANUP_UPDATE_STATS	Internal use only.
GHOSTCLEANUPSYNCMGR	Internal use only.
GLOBAL_QUERY_CANCEL	Internal use only.
GLOBAL_QUERY_CLOSE	Internal use only.
GLOBAL_QUERY_CONSUMER	Internal use only.
GLOBAL_QUERY_PRODUCER	Internal use only.
GLOBAL_TRAN_CREATE	Internal use only.
GLOBAL_TRAN_UCS_SESSION	Internal use only.
GUARDIAN	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
HADR_AG_MUTEX	Occurs when an Always On DDL statement or Windows Server Failover Clustering command is waiting for exclusive read/write access to the configuration of an availability group.
HADR_AR_CRITICAL_SECTION_ENTRY	Occurs when an Always On DDL statement or Windows Server Failover Clustering command is waiting for exclusive read/write access to the runtime state of the

Wait Type	Description
	local replica of the associated availability group.
HADR_AR_MANAGER_MUTEX	Occurs when an availability replica shutdown is waiting for startup to complete or an availability replica startup is waiting for shutdown to complete. Internal use only.
HADR_AR_UNLOAD_COMPLETED	Internal use only.
HADR_ARCONTROLLER_NOTIFICATIONS_SUBSCRIBER_LIST	The publisher for an availability replica event (such as a state change or configuration change) is waiting for exclusive read/write access to the list of event subscribers. Internal use only.
HADR_BACKUP_BULK_LOCK	The Always On primary database received a backup request from a secondary database and is waiting for the background thread to finish processing the request on acquiring or releasing the BulkOp lock.
HADR_BACKUP_QUEUE	The backup background thread of the Always On primary database is waiting for a new work request from the secondary database. (Typically, this occurs when the primary database is holding the BulkOp log and is waiting for the secondary database to indicate that the primary database can release the lock).
HADR_CLUSAPI_CALL	A SQL Server thread is waiting to switch from non-preemptive mode (scheduled by SQL Server) to preemptive mode (scheduled by the operating system) in order

Wait Type	Description
	to invoke Windows Server Failover Clustering APIs.
HADR_COMPRESSED_CACHE_SYNC	Waiting for access to the cache of compressed log blocks that is used to avoid redundant compression of the log blocks sent to multiple secondary databases.
HADR_CONNECTIVITY_INFO	Internal use only.
HADR_DATABASE_FLOW_CONTROL	Waiting for messages to be sent to the partner when the maximum number of queued messages has been reached. Indicates that the log scans are running faster than the network sends. This is an issue only if network sends are slower than expected.
HADR_DATABASE_VERSIONING_STATE	Occurs on the versioning state change of an Always On secondary database. This wait is for internal data structures and usually is very short with no direct effect on data access.
HADR_DATABASE_WAIT_FOR_RECOVERY	Internal use only.
HADR_DATABASE_WAIT_FOR_RESTART	Waiting for the database to restart under Always On Availability Groups control. Under normal conditions, this is not a customer issue because waits are expected here.
HADR_DATABASE_WAIT_FOR_TRANSITION_TO_VERSIONING	A query on object(s) in a readable secondary database of an Always On availability group is blocked on row versioning while waiting for commit or rollback of all transactions that were in-flight when the secondary replica was

Wait Type	Description
	enabled for read workloads. This wait type guarantees that row versions are available before execution of a query under snapshot isolation.
HADR_DB_COMMAND	Waiting for responses to conversational messages (which require an explicit response from the other side, using the Always On conversational message infrastructure). A number of different message types use this wait type.
HADR_DB_OP_COMPLETION_SYNC	Waiting for responses to conversational messages (which require an explicit response from the other side, using the Always On conversational message infrastructure). A number of different message types use this wait type.
HADR_DB_OP_START_SYNC	An Always On DDL statement or a Windows Server Failover Clustering command is waiting for serialized access to an availability database and its runtime state.
HADR_DBR_SUBSCRIBER	The publisher for an availability replica event (such as a state change or configuration change) is waiting for exclusive read/write access to the runtime state of an event subscriber that corresponds to an availability database. Internal use only.
HADR_DBR_SUBSCRIBER_FILTER_LIST	The publisher for an availability replica event (such as a state change or configuration change) is waiting for exclusive read/write access to the list of event

Wait Type	Description
	subscribers that correspond to availability databases. Internal use only.
HADR_DBSEEDING	Internal use only.
HADR_DBSEEDING_LIST	Internal use only.
HADR_DBSTATECHANGE_SYNC	Concurrency control wait for updating the internal state of the database replica.
HADR_FABRIC_CALLBACK	Internal use only.
HADR_FILESTREAM_BLOCK_FLUSH	The FILESTREAM Always On transport manager is waiting until processing of a log block is finished.
HADR_FILESTREAM_FILE_CLOSE	The FILESTREAM Always On transport manager is waiting until the next FILESTREAM file gets processed and its handle gets closed.
HADR_FILESTREAM_FILE_REQUEST	An Always On secondary replica is waiting for the primary replica to send all requested FILESTREAM files during UNDO.
HADR_FILESTREAM_IOMGR	The FILESTREAM Always On transport manager is waiting for R/W lock that protects the FILESTREAM Always On I/O manager during startup or shutdown.
HADR_FILESTREAM_IOMGR_IOCOMPLETION	The FILESTREAM Always On I/O manager is waiting for I/O completion.
HADR_FILESTREAM_MANAGER	The FILESTREAM Always On transport manager is waiting for the R/W lock that protects the

Wait Type	Description
	FILESTREAM Always On transport manager during startup or shutdown.
HADR_FILESTREAM_PREPROC	Internal use only.
HADR_GROUP_COMMIT	Transaction commit processing is waiting to allow a group commit so that multiple commit log records can be put into a single log block. This wait is an expected condition that optimizes the log I/O, capture, and send operations.
HADR_LOGCAPTURE_SYNC	Concurrency control around the log capture or apply object when creating or destroying scans. This is an expected wait when partners change state or connection status.
HADR_LOGCAPTURE_WAIT	Waiting for log records to become available. Can occur either when waiting for new log records to be generated by connections or for I/O completion when reading log not in the cache. This is an expected wait if the log scan is caught up to the end of log or is reading from disk.
HADR_LOGPROGRESS_SYNC	Concurrency control wait when updating the log progress status of database replicas.
HADR_NOTIFICATION_DEQUEUE	A background task that processes Windows Server Failover Clustering notifications is waiting for the next notification. Internal use only.
HADR_NOTIFICATION_WORKER_EXCLUSIVE_ACCESS	The Always On availability replica manager is waiting for serialized access to the runtime state of a background task that processes

Wait Type	Description
	Windows Server Failover Clustering notifications. Internal use only.
HADR_NOTIFICATION_WORKER_STARTUP_SYNC	A background task is waiting for the completion of the startup of a background task that processes Windows Server Failover Clustering notifications. Internal use only.
HADR_NOTIFICATION_WORKER_TERMINATION_SYNC	A background task is waiting for the termination of a background task that processes Windows Server Failover Clustering notifications. Internal use only.
HADR_PARTNER_SYNC	Concurrency control wait on the partner list.
HADR_READ_ALL_NETWORKS	Waiting to get read or write access to the list of WSFC networks. Internal use only. Note: The engine keeps a list of WSFC networks that is used in dynamic management views (such as sys.dm_hadr_cluster_networks) or to validate Always On Transact-SQL statements that reference WSFC network information. This list is updated upon engine startup, WSFC related notifications, and internal Always On restart (for example, losing and regaining of WSFC quorum). Tasks will usually be blocked when an update in that list is in progress.
HADR_RECOVERY_WAIT_FOR_CONNECTION	Waiting for the secondary database to connect to the primary database before running recovery. This is an expected wait, which can lengthen if the

Wait Type	Description
	connection to the primary is slow to establish.
HADR_RECOVERY_WAIT_FOR_UNDO	Database recovery is waiting for the secondary database to finish the reverting and initializing phase to bring it back to the common log point with the primary database. This is an expected wait after failovers. Undo progress can be tracked through the Windows System Monitor (perfmon.exe) and dynamic management views.
HADR_REPLICAINFO_SYNC	Waiting for concurrency control to update the current replica state.
HADR_SEEDING_CANCELLATION	Internal use only.
HADR_SEEDING_FILE_LIST	Internal use only.
HADR_SEEDING_LIMIT_BACKUPS	Internal use only.
HADR_SEEDING_SYNC_COMPLETION	Internal use only.
HADR_SEEDING_TIMEOUT_TASK	Internal use only.
HADR_SEEDING_WAIT_FOR_COMPLETION	Internal use only.
HADR_SYNC_COMMIT	Waiting for a transaction commit processing on the synchronized secondary databases to harden the log. This wait is also reflected by the Transaction Delay performance counter. This wait type is expected for synchronous-commit Availability Groups and indicates the time to send, write, and acknowledge log commit to the secondary databases.
HADR_SYNCHRONIZING_THROTTLE	Waiting for transaction commit processing to allow a synchronizing secondary database

Wait Type	Description
	to catch up to the primary end of log in order to transition to the synchronized state. This is an expected wait when a secondary database is catching up.
HADR_TDS_LISTENER_SYNC	Either the internal Always On system or the WSFC cluster will request that listeners are started or stopped. The processing of this request is always asynchronous, and there is a mechanism to remove redundant requests. There are also moments that this process is suspended because of configuration changes. All waits related with this listener synchronization mechanism use this wait type. Internal use only.
HADR_TDS_LISTENER_SYNC_PROCESSING	Used at the end of an Always On Transact-SQL statement that requires starting and/or stopping an availability group listener. Since the start/stop operation is done asynchronously, the user thread will block using this wait type until the situation of the listener is known.
HADR_THROTTLE_LOG_RATE_GOVERNOR	Internal use only.
HADR_THROTTLE_LOG_RATE_MISMATCHED_SLO	Occurs when a geo-replication secondary is configured with lower compute size (lower SLO) than the primary. A primary database is throttled due to delayed log consumption by the secondary. This is caused by the secondary database having insufficient compute capacity to keep up with the primary database's rate of change.

Wait Type	Description
HADR_THROTTLE_LOG_RATE_LOG_SIZE	Internal use only.
HADR_THROTTLE_LOG_RATE_SEEDING	Internal use only.
HADR_THROTTLE_LOG_RATE_SEND_RECV_QUEUE_SIZE	Internal use only.
HADR_TIMER_TASK	Waiting to get the lock on the timer task object and is also used for the actual waits between times that work is being performed. For example, for a task that runs every 10 seconds, after one execution, Always On Availability Groups waits about 10 seconds to reschedule the task, and the wait is included here.
HADR_TRANSPORT_DBRLIST	Waiting for access to the transport layer's database replica list. Used for the spinlock that grants access to it.
HADR_TRANSPORT_FLOW_CONTROL	Waiting when the number of outstanding unacknowledged Always On messages is over the out flow control threshold. This is on an availability replica-to-replica basis (not on a database-to-database basis).
HADR_TRANSPORT_SESSION	Always On Availability Groups is waiting while changing or accessing the underlying transport state.
HADR_WORK_POOL	Concurrency control wait on the Always On Availability Groups background work task object.
HADR_WORK_QUEUE	Always On Availability Groups background worker thread waiting for new work to be assigned. This is an expected wait when there

Wait Type	Description
	are ready workers waiting for new work, which is the normal state.
HADR_XRF_STACK_ACCESS	Accessing (look up, add, and delete) the extended recovery fork stack for an Always On availability database.
HCCO_CACHE	Internal use only.
HK_RESTORE_FILEMAP	Internal use only.
HKCS_PARALLEL_MIGRATION	Internal use only.
HKCS_PARALLEL_RECOVERY	Internal use only.
HTBUILD	Occurs with parallel batch-mode plans when synchronizing the building of the hash table on the input side of a hash join/aggregation. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.
HTDELETE	Occurs with parallel batch-mode plans when synchronizing at the end of a hash join/aggregation. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.
HTMEMO	Occurs with parallel batch-mode plans when synchronizing before scanning hash table to output matches / non-matches in hash join/aggregation. If waiting is excessive and cannot be reduced

Wait Type	Description
	by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.
HTREINIT	Occurs with parallel batch-mode plans when synchronizing before resetting a hash join/aggregation for the next partial join. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.
HTREPARTITION	Occurs with parallel batch-mode plans when synchronizing the repartitioning of the hash table on the input side of a hash join/aggregation. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.
HTTP_ENUMERATION	Occurs at startup to enumerate the HTTP endpoints to start HTTP.
HTTP_START	Occurs when a connection is waiting for HTTP to complete initialization.
HTTP_STORAGE_CONNECTION	Internal use only.
IMPPROV_IOWAIT	Occurs when SQL Server waits for a bulkload I/O to finish.
INSTANCE_LOG_RATE_GOVERNOR	Internal use only.

Wait Type	Description
INTERNAL_TESTING	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
IO_AUDIT_MUTEX	Occurs during synchronization of trace event buffers.
IO_COMPLETION	Occurs while waiting for I/O operations to complete. This wait type generally represents non-data page I/Os. Data page I/O completion waits appear as PAGEIOLATCH_* waits.
IO_QUEUE_LIMIT	Internal use only.
IO_RETRY	Occurs when an I/O operation such as a read or a write to disk fails because of insufficient resources, and is then retried.
IOAFF_RANGE_QUEUE	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
KSOURCE_WAKEUP	Used by the service control task while waiting for requests from the Service Control Manager. Long waits are expected and do not indicate a problem.
KTM_ENLISTMENT	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
KTM_RECOVERY_MANAGER	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.

Wait Type	Description
KTM_RECOVERY_RESOLUTION	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
LATCH_DT	Occurs when waiting for a DT (destroy) latch. This does not include buffer latches or transaction mark latches. A listing of LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LATCH_EX	Occurs when waiting for an EX (exclusive) latch. This does not include buffer latches or transaction mark latches. A listing of LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LATCH_KP	Occurs when waiting for a KP (keep) latch. This does not include buffer latches or transaction mark latches. A listing of LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LATCH_NL	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.

Wait Type	Description
LATCH_SH	Occurs when waiting for an SH (share) latch. This does not include buffer latches or transaction mark latches. A listing of LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LATCH_UP	Occurs when waiting for an UP (update) latch. This does not include buffer latches or transaction mark latches. A listing of LATCH_* waits is available in sys.dm_os_latch_stats. Note that sys.dm_os_latch_stats groups LATCH_NL, LATCH_SH, LATCH_UP, LATCH_EX, and LATCH_DT waits together.
LAZYWRITER_SLEEP	Occurs when lazy writer tasks are suspended. This is a measure of the time spent by background tasks that are waiting. Do not consider this state when you are looking for user stalls.
LCK_M_BU	Occurs when a task is waiting to acquire a Bulk Update (BU) lock. See Bulk Update Locks for more information
LCK_M_BU_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a Bulk Update (BU) lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Bulk Update Locks for more information

Wait Type	Description
LCK_M_BU_LOW_PRIORITY	Occurs when a task is waiting to acquire a Bulk Update (BU) lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Bulk Update Locks for more information
LCK_M_IS	Occurs when a task is waiting to acquire an Intent Shared (IS) lock. See Intent Locks for more information
LCK_M_IS_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Intent Shared (IS) lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.) See Intent Locks for more information.
LCK_M_IS_LOW_PRIORITY	Occurs when a task is waiting to acquire an Intent Shared (IS) lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Intent Locks for more information.
LCK_M_IU	Occurs when a task is waiting to acquire an Intent Update (IU) lock. See Intent Locks for more information
LCK_M_IU_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Intent Update (IU) lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.) See Intent Locks for more information.

Wait Type	Description
LCK_M_IU_LOW_PRIORITY	Occurs when a task is waiting to acquire an Intent Update (IU) lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.). See Intent Locks for more information.
LCK_M_IX	Occurs when a task is waiting to acquire an Intent Exclusive (IX) lock. See Intent Locks for more information
LCK_M_IX_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Intent Exclusive (IX) lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Intent Locks for more information.
LCK_M_IX_LOW_PRIORITY	Occurs when a task is waiting to acquire an Intent Exclusive (IX) lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Intent Locks for more information.
LCK_M_RIn_NL	Occurs when a task is waiting to acquire a NULL lock on the current key value, and an Insert Range lock between the current and previous key. A NULL lock on the key is an instant release lock.
LCK_M_RIn_NL_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a NULL lock with Abort Blockers on the current key value, and an Insert Range lock with Abort Blockers between the current and previous key. A NULL lock on the key is an instant

Wait Type	Description
	release lock. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RIn_NL_LOW_PRIORITY	Occurs when a task is waiting to acquire a NULL lock with Low Priority on the current key value, and an Insert Range lock with Low Priority between the current and previous key. A NULL lock on the key is an instant release lock. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RIn_S	Occurs when a task is waiting to acquire a shared lock on the current key value, and an Insert Range lock between the current and previous key.
LCK_M_RIn_S_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a shared lock with Abort Blockers on the current key value, and an Insert Range lock with Abort Blockers between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RIn_S_LOW_PRIORITY	Occurs when a task is waiting to acquire a shared lock with Low Priority on the current key value, and an Insert Range lock with Low Priority between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RIn_U	Task is waiting to acquire an Update lock on the current key value, and an Insert Range lock

Wait Type	Description
	between the current and previous key.
LCK_M_RIn_U_ABORT_BLOCKERS	Task is waiting to acquire an Update lock with Abort Blockers on the current key value, and an Insert Range lock with Abort Blockers between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RIn_U_LOW_PRIORITY	Task is waiting to acquire an Update lock with Low Priority on the current key value, and an Insert Range lock with Low Priority between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RIn_X	Occurs when a task is waiting to acquire an Exclusive lock on the current key value, and an Insert Range lock between the current and previous key.
LCK_M_RIn_X_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Exclusive lock with Abort Blockers on the current key value, and an Insert Range lock with Abort Blockers between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RIn_X_LOW_PRIORITY	Occurs when a task is waiting to acquire an Exclusive lock with Low Priority on the current key value, and an Insert Range lock with Low Priority between the current and previous key. (Related to the low

Wait Type	Description
	priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RS_S	Occurs when a task is waiting to acquire a Shared lock on the current key value, and a Shared Range lock between the current and previous key.
LCK_M_RS_S_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a Shared lock with Abort Blockers on the current key value, and a Shared Range lock with Abort Blockers between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RS_S_LOW_PRIORITY	Occurs when a task is waiting to acquire a Shared lock with Low Priority on the current key value, and a Shared Range lock with Low Priority between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RS_U	Occurs when a task is waiting to acquire an Update lock on the current key value, and an Update Range lock between the current and previous key.
LCK_M_RS_U_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Update lock with Abort Blockers on the current key value, and an Update Range lock with Abort Blockers between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RS_U_LOW_PRIORITY	Occurs when a task is waiting to acquire an Update lock with Low

Wait Type	Description
	Priority on the current key value, and an Update Range lock with Low Priority between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RX_S	Occurs when a task is waiting to acquire a Shared lock on the current key value, and an Exclusive Range lock between the current and previous key.
LCK_M_RX_S_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a Shared lock with Abort Blockers on the current key value, and an Exclusive Range with Abort Blockers lock between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RX_S_LOW_PRIORITY	Occurs when a task is waiting to acquire a Shared lock with Low Priority on the current key value, and an Exclusive Range with Low Priority lock between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RX_U	Occurs when a task is waiting to acquire an Update lock on the current key value, and an Exclusive range lock between the current and previous key.
LCK_M_RX_U_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Update lock with Abort Blockers on the current key value, and an Exclusive range lock with Abort Blockers between the current and previous key. (Related

Wait Type	Description
	to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RX_U_LOW_PRIORITY	Occurs when a task is waiting to acquire an Update lock with Low Priority on the current key value, and an Exclusive range lock with Low Priority between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RX_X	Occurs when a task is waiting to acquire an Exclusive lock on the current key value, and an Exclusive Range lock between the current and previous key.
LCK_M_RX_X_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Exclusive lock with Abort Blockers on the current key value, and an Exclusive Range lock with Abort Blockers between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_RX_X_LOW_PRIORITY	Occurs when a task is waiting to acquire an Exclusive lock with Low Priority on the current key value, and an Exclusive Range lock with Low Priority between the current and previous key. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX.),
LCK_M_S	Occurs when a task is waiting to acquire a Shared lock. See Shared Locks for more information.
LCK_M_S_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a Shared lock with Abort Blockers. (Related to the low priority wait option of ALTER

Wait Type	Description
	TABLE and ALTER INDEX). See Shared Locks for more information.
LCK_M_S_LOW_PRIORITY	Occurs when a task is waiting to acquire a Shared lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Shared Locks for more information.
LCK_M_SCH_M	Occurs when a task is waiting to acquire a Schema Modify lock. See Schema Locks for more information.
LCK_M_SCH_M_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a Schema Modify lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Schema Locks for more information.
LCK_M_SCH_M_LOW_PRIORITY	Occurs when a task is waiting to acquire a Schema Modify lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Schema Locks for more information.
LCK_M_SCH_S	Occurs when a task is waiting to acquire a Schema Share lock. See Schema Locks for more information.
LCK_M_SCH_S_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a Schema Share lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX).

Wait Type	Description
	See Schema Locks for more information.
LCK_M_SCH_S_LOW_PRIORITY	Occurs when a task is waiting to acquire a Schema Share lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX) See Schema Locks for more information.
LCK_M_SIU	Occurs when a task is waiting to acquire a Shared With Intent Update lock. See Intent Locks for more information.
LCK_M_SIU_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a Shared With Intent Update lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Intent Locks for more information.
LCK_M_SIU_LOW_PRIORITY	Occurs when a task is waiting to acquire a Shared With Intent Update lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Intent Locks for more information.
LCK_M_SIX	Occurs when a task is waiting to acquire a Shared With Intent Exclusive lock. See Intent Locks for more information.
LCK_M_SIX_ABORT_BLOCKERS	Occurs when a task is waiting to acquire a Shared With Intent Exclusive lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER

Wait Type	Description
	INDEX). See Intent Locks for more information.
LCK_M_SIX_LOW_PRIORITY	Occurs when a task is waiting to acquire a Shared With Intent Exclusive lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Intent Locks for more information.
LCK_M_U	Occurs when a task is waiting to acquire an Update lock. See Update Locks for more information.
LCK_M_U_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Update lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Update Locks for more information.
LCK_M_U_LOW_PRIORITY	Occurs when a task is waiting to acquire an Update lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Update Locks for more information.
LCK_M_UIX	Occurs when a task is waiting to acquire an Update With Intent Exclusive lock. See Intent Locks for more information.
LCK_M_UIX_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Update With Intent Exclusive lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER

Wait Type	Description
	INDEX). See Intent Locks for more information.
LCK_M_UIX_LOW_PRIORITY	Occurs when a task is waiting to acquire an Update With Intent Exclusive lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Intent Locks for more information.
LCK_M_X	Occurs when a task is waiting to acquire an Exclusive lock. See Exclusive Locks for more information.
LCK_M_X_ABORT_BLOCKERS	Occurs when a task is waiting to acquire an Exclusive lock with Abort Blockers. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Exclusive Locks for more information.
LCK_M_X_LOW_PRIORITY	Occurs when a task is waiting to acquire an Exclusive lock with Low Priority. (Related to the low priority wait option of ALTER TABLE and ALTER INDEX). See Exclusive Locks for more information.
LOG_POOL_SCAN	Internal use only.
LOG_RATE_GOVERNOR	Internal use only.
LOGBUFFER	Occurs when a task is waiting for space in the log buffer to store a log record. Consistently high values may indicate that the log devices cannot keep up with the amount of log being generated by the server.

Wait Type	Description
LOGCAPTURE_LOGPOOLTRUNCPOINT	Internal use only.
LOGGENERATION	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
LOGMGR	Occurs when a task is waiting for any outstanding log I/Os to finish before shutting down the log while closing the database.
LOGMGR_FLUSH	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
LOGMGR_PMM_LOG	Internal use only.
LOGMGR_QUEUE	Occurs while the log writer task waits for work requests.
LOGMGR_RESERVE_APPEND	Occurs when a task is waiting to see whether log truncation frees up log space to enable the task to write a new log record. Consider increasing the size of the log file(s) for the affected database to reduce this wait.
LOGPOOL_CACHESIZE	Internal use only.
LOGPOOL_CONSUMER	Internal use only.
LOGPOOL_CONSUMERSET	Internal use only.
LOGPOOL_FREEPOOLS	Internal use only.
LOGPOOL_MGRSET	Internal use only.
LOGPOOL_REPLACEMENTSET	Internal use only.
LOGPOOLREFCOUNTEDOBJECT_REFDONE	Internal use only.

Wait Type	Description
LOWFAIL_MEMMGR_QUEUE	Occurs while waiting for memory to be available for use.
MD_AGENT_YIELD	Internal use only.
MD_LAZYCACHE_RWLOCK	Internal use only.
MEMORY_ALLOCATION_EXT	Occurs while allocating memory from either the internal SQL Server memory pool or the operation system.
MEMORY_GRANT_UPDATE	Internal use only.
METADATA_LAZYCACHE_RWLOCK	Internal use only.
MIGRATIONBUFFER	Internal use only.
MISCELLANEOUS	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
MISCELLANEOUS	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
MSQL_DQ	Occurs when a task is waiting for a distributed query operation to finish. This is used to detect potential Multiple Active Result Set (MARS) application deadlocks. The wait ends when the distributed query call finishes.
MSQL_XACT_MGR_MUTEX	Occurs when a task is waiting to obtain ownership of the session transaction manager to perform a session level transaction operation.

Wait Type	Description
MSQL_XACT_MUTEX	Occurs during synchronization of transaction usage. A request must acquire the mutex before it can use the transaction.
MSQL_XP	Occurs when a task is waiting for an extended stored procedure to end. SQL Server uses this wait state to detect potential MARS application deadlocks. The wait stops when the extended stored procedure call ends.
MSSEARCH	Occurs during Full-Text Search calls. This wait ends when the full-text operation completes. It does not indicate contention, but rather the duration of full-text operations.
NET_WAITFOR_PACKET	Occurs when a connection is waiting for a network packet during a network read.
NETWORKSXMLMGRLOAD	Internal use only.
NODE_CACHE_MUTEX	Internal use only.
OLEDB	Occurs when SQL Server calls the SNAC OLE DB Provider (SQLNCLI) or the Microsoft OLE DB Driver for SQL Server (MSOLEDBSQL). This wait type is not used for synchronization. Instead, it indicates the duration of calls to the OLE DB provider.
ONDEMAND_TASK_QUEUE	Occurs while a background task waits for high priority system task requests. Long wait times indicate that there have been no high priority requests to process, and should not cause concern.

Wait Type	Description
PAGEIOLATCH_DT	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Destroy mode. Long waits may indicate problems with the disk subsystem.
PAGEIOLATCH_EX	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Exclusive mode - a mode used when the buffer is being written to disk. Long waits may indicate problems with the disk subsystem.
PAGEIOLATCH_KP	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Keep mode. Long waits may indicate problems with the disk subsystem.
PAGEIOLATCH_NL	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PAGEIOLATCH_SH	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Shared mode - a mode used when the buffer is being read from disk. Long waits may indicate problems with the disk subsystem.
PAGEIOLATCH_UP	Occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Update mode. Long waits may indicate problems with the disk subsystem.

Wait Type	Description
PAGELATCH_DT	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Destroy mode. Destroy mode must be acquired before deleting contents of a page. See Latch Modes for more information.
PAGELATCH_EX	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Exclusive mode - it blocks other threads from writing to or reading from the page (buffer).
PAGELATCH_KP	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Keep mode which prevents the page from being destroyed by another thread. See Latch Modes for more information.
PAGELATCH_NL	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PAGELATCH_SH	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Shared mode which allows multiple threads to read, but not modify, a buffer (page). See Latch Modes for more information.
PAGELATCH_UP	Occurs when a task is waiting on a latch for a buffer that is not in an I/O request. The latch request is in Update mode. Commonly this wait type may be observed when a system page (buffer) like PFS,

Wait Type	Description
	GAM, SGAM is latched. See Latch Modes for more information.
PARALLEL_BACKUP_QUEUE	Occurs when serializing output produced by RESTORE HEADERONLY, RESTORE FILELISTONLY, or RESTORE LABELONLY.
PARALLEL_REDO_DRAIN_WORKER	Internal use only.
PARALLEL_REDO_FLOW_CONTROL	Internal use only.
PARALLEL_REDO_LOG_CACHE	Internal use only.
PARALLEL_REDO_TRAN_LIST	Internal use only.
PARALLEL_REDO_TRAN_TURN	Internal use only.
PARALLEL_REDO_WORKER_SYNC	Internal use only.
PARALLEL_REDO_WORKER_WAIT_WORK	Internal use only.
PERFORMANCE_COUNTERS_RWLOCK	Internal use only.
PHYSICAL_SEEDING_DMV	Internal use only.
POOL_LOG_RATE_GOVERNOR	Internal use only.
PREEMPTIVE_ABR	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PREEMPTIVE_AUDIT_ACCESS_EVENTLOG	Occurs when the SQL Server Operating System (SQLOS) scheduler switches to preemptive mode to write an audit event to the Windows event log.
PREEMPTIVE_AUDIT_ACCESS_SECLOG	Occurs when the SQLOS scheduler switches to preemptive mode to

Wait Type	Description
	write an audit event to the Windows Security log.
PREEMPTIVE_CLOSEBACKUPMEDIA	Occurs when the SQLOS scheduler switches to preemptive mode to close backup media.
PREEMPTIVE_CLOSEBACKUPTAPE	Occurs when the SQLOS scheduler switches to preemptive mode to close a tape backup device.
PREEMPTIVE_CLOSEBACKUPVDIDEVICE	Occurs when the SQLOS scheduler switches to preemptive mode to close a virtual backup device.
PREEMPTIVE_CLUSAPI_CLUSTERRESOURCECONTROL	Occurs when the SQLOS scheduler switches to preemptive mode to perform Windows failover cluster operations.
PREEMPTIVE_COM_COCREATEINSTANCE	Occurs when the SQLOS scheduler switches to preemptive mode to create a COM object.
PREEMPTIVE_COM_COGETCLASSOBJECT	Internal use only.
PREEMPTIVE_COM_CREATEACCESSOR	Internal use only.
PREEMPTIVE_COM_DELETEROWS	Internal use only.
PREEMPTIVE_COM_GETCOMMANDTEXT	Internal use only.
PREEMPTIVE_COM_GETDATA	Internal use only.
PREEMPTIVE_COM_GETNEXTROWS	Internal use only.
PREEMPTIVE_COM_GETRESULT	Internal use only.
PREEMPTIVE_COM_GETROWSBYBOOKMARK	Internal use only.
PREEMPTIVE_COM_LBFLUSH	Internal use only.
PREEMPTIVE_COM_LBLOCKREGION	Internal use only.

Wait Type	Description
PREEMPTIVE_COM_LBREADAT	Internal use only.
PREEMPTIVE_COM_LBSETSIZE	Internal use only.
PREEMPTIVE_COM_LBSTAT	Internal use only.
PREEMPTIVE_COM_LBUNLOCKREGION	Internal use only.
PREEMPTIVE_COM_LBWRITEAT	Internal use only.
PREEMPTIVE_COM_QUERYINTERFACE	Internal use only.
PREEMPTIVE_COM_RELEASE	Internal use only.
PREEMPTIVE_COM_RELEASEACCESSOR	Internal use only.
PREEMPTIVE_COM_RELEASEROWS	Internal use only.
PREEMPTIVE_COM_RELEASESESSION	Internal use only.
PREEMPTIVE_COM_RESTARTPOSITION	Internal use only.
PREEMPTIVE_COM_SEQSTRMREAD	Internal use only.
PREEMPTIVE_COM_SEQSTRMREADANDWRITE	Internal use only.
PREEMPTIVE_COM_SETDATAFAILURE	Internal use only.
PREEMPTIVE_COM_SETPARAMETERINFO	Internal use only.
PREEMPTIVE_COM_SETPARAMETERPROPERTIES	Internal use only.
PREEMPTIVE_COM_STRMLOCKREGION	Internal use only.
PREEMPTIVE_COM_STRMSEEKANDREAD	Internal use only.
PREEMPTIVE_COM_STRMSEEKANDWRITE	Internal use only.
PREEMPTIVE_COM_STRMSETSIZE	Internal use only.
PREEMPTIVE_COM_STRMSTAT	Internal use only.
PREEMPTIVE_COM_STRMUNLOCKREGION	Internal use only.

Wait Type	Description
PREEMPTIVE_CONSOLEWRITE	Internal use only.
PREEMPTIVE_CREATEPARAM	Internal use only.
PREEMPTIVE_DEBUG	Internal use only.
PREEMPTIVE_DFSADDLINK	Internal use only.
PREEMPTIVE_DFSLINKEXISTCHECK	Internal use only.
PREEMPTIVE_DFSLINKHEALTHCHECK	Internal use only.
PREEMPTIVE_DFSREMOVELINK	Internal use only.
PREEMPTIVE_DFSREMOVEROOT	Internal use only.
PREEMPTIVE_DFSROOTFOLDERCHECK	Internal use only.
PREEMPTIVE_DFSROOTINIT	Internal use only.
PREEMPTIVE_DFSROOTSHARECHECK	Internal use only.
PREEMPTIVE_DTC_ABORT	Internal use only.
PREEMPTIVE_DTC_ABORTREQUESTDONE	Internal use only.
PREEMPTIVE_DTC_BEGINTRANSACTION	Internal use only.
PREEMPTIVE_DTC_COMMITREQUESTDONE	Internal use only.
PREEMPTIVE_DTC_ENLIST	Internal use only.
PREEMPTIVE_DTC_PREPAREREQUESTDONE	Internal use only.
PREEMPTIVE_FILESIZEGET	Internal use only.
PREEMPTIVE_FSAOLEDB_ABORTTRANSACTION	Internal use only.
PREEMPTIVE_FSAOLEDB_COMMITTRANSACTION	Internal use only.
PREEMPTIVE_FSAOLEDB_STARTTRANSACTION	Internal use only.
PREEMPTIVE_FSRECOVER_UNCONDITIONALUNDO	Internal use only.

Wait Type	Description
PREEMPTIVE_GETRMINFO	Internal use only.
PREEMPTIVE_HADR_LEASE_MECHANISM	Always On Availability Groups lease manager scheduling for Microsoft Support diagnostics.
PREEMPTIVE_HTTP_EVENT_WAIT	Internal use only.
PREEMPTIVE_HTTP_REQUEST	Internal use only.
PREEMPTIVE_LOCKMONITOR	Internal use only.
PREEMPTIVE_MSS_RELEASE	Internal use only.
PREEMPTIVE_ODBCOPS	Internal use only.
PREEMPTIVE_OLE_UNINIT	Internal use only.
PREEMPTIVE_OLEDB_ABORTORCOMMITTRAN	Internal use only.
PREEMPTIVE_OLEDB_ABORTTRAN	Internal use only.
PREEMPTIVE_OLEDB_GETDATASOURCE	Internal use only.
PREEMPTIVE_OLEDB_GETLITERALINFO	Internal use only.
PREEMPTIVE_OLEDB_GETPROPERTIES	Internal use only.
PREEMPTIVE_OLEDB_GETPROPERTYINFO	Internal use only.
PREEMPTIVE_OLEDB_GETSCHEMALOCK	Internal use only.
PREEMPTIVE_OLEDB_JOINTRANSACTION	Internal use only.
PREEMPTIVE_OLEDB_RELEASE	Internal use only.
PREEMPTIVE_OLEDB_SETPROPERTIES	Internal use only.
PREEMPTIVE_OLEDBOPS	Internal use only.
PREEMPTIVE_OS_ACCEPTSECURITYCONTEXT	Internal use only.
PREEMPTIVE_OS_ACQUIRECREDENTIALSHANDLE	Internal use only.

Wait Type	Description
PREEMPTIVE_OS_AUTHENTICATIONOPS	Internal use only. Likely related to waiting for an Active Directory operation to complete.
PREEMPTIVE_OS_AUTHORIZATIONOPS	Internal use only. Likely related to waiting for a Windows function related to authorization and security permissions.
PREEMPTIVE_OS_AUTHZGETINFORMATIONFROMCONTEXT	Internal use only.
PREEMPTIVE_OS_AUTHZINITIALIZECONTEXTFROMSID	Internal use only.
PREEMPTIVE_OS_AUTHZINITIALIZERESOURCEMANAGER	Internal use only.
PREEMPTIVE_OS_BACKUPREAD	Internal use only.
PREEMPTIVE_OS_CLOSEHANDLE	Internal use only.
PREEMPTIVE_OS_CLUSTEROPS	Internal use only.
PREEMPTIVE_OS_COMOPS	Internal use only.
PREEMPTIVE_OS_COMPLETEAUTHTOKEN	Internal use only.
PREEMPTIVE_OS_COPYFILE	Internal use only.
PREEMPTIVE_OS_CREATEDIRECTORY	Internal use only.
PREEMPTIVE_OS_CREATEFILE	Internal use only.
PREEMPTIVE_OS_CRYPTACQUIRECONTEXT	Internal use only. Likely related to waiting for the Windows CryptAcquireContext function.
PREEMPTIVE_OS_CRYPTIMPORTKEY	Internal use only. Likely related to waiting for the CRYPTIMPORTKEY Windows function.
PREEMPTIVE_OS_CRYPTOPS	Internal use only. Likely related to waiting for a Windows function related to cryptography.

Wait Type	Description
PREEMPTIVE_OS_DECRYPTMESSAGE	Internal use only.
PREEMPTIVE_OS_DELETEFILE	Internal use only.
PREEMPTIVE_OS_DELETESECURITYCONTEXT	Internal use only. Likely related to waiting for the Windows DeleteSecurityContext function.
PREEMPTIVE_OS_DEVICEIOCONTROL	Internal use only.
PREEMPTIVE_OS_DEVICEOPS	Internal use only.
PREEMPTIVE_OS_DIRSVCS_NETWORKOPS	Internal use only.
PREEMPTIVE_OS_DISCONNECTNAMEDPIPE	Internal use only.
PREEMPTIVE_OS_DOMAINSERVICESOPS	Internal use only.
PREEMPTIVE_OS_DSGETDCNAME	Internal use only.
PREEMPTIVE_OS_DTCOPS	Internal use only.
PREEMPTIVE_OS_ENCRYPTMESSAGE	Internal use only.
PREEMPTIVE_OS_FILEOPS	Internal use only.
PREEMPTIVE_OS_FINDFILE	Internal use only.
PREEMPTIVE_OS_FLUSHFILEBUFFERS	Internal use only.
PREEMPTIVE_OS_FORMATMESSAGE	Internal use only.
PREEMPTIVE_OS_FREECREDENTIALSHANDLE	Internal use only.
PREEMPTIVE_OS_FREELIBRARY	Internal use only.
PREEMPTIVE_OS_GENERICOPS	Internal use only.
PREEMPTIVE_OS_GETADDRINFO	Internal use only.
PREEMPTIVE_OS_GETCOMPRESSEDFILESIZE	Internal use only.
PREEMPTIVE_OS_GETDISKFREESPACE	Internal use only.

Wait Type	Description
PREEMPTIVE_OS_GETFILEATTRIBUTES	Internal use only.
PREEMPTIVE_OS_GETFILESIZE	Internal use only.
PREEMPTIVE_OS_GETFINALFILEPATHBYHANDLE	Internal use only.
PREEMPTIVE_OS_GETLONGPATHNAME	Internal use only.
PREEMPTIVE_OS_GETPROCADDRESS	Internal use only.
PREEMPTIVE_OS_GETVOLUMENAMEFORVOLUMEMOUNTPOINT	Internal use only.
PREEMPTIVE_OS_GETVOLUMEPATHNAME	Internal use only.
PREEMPTIVE_OS_INITIALIZESECURITYCONTEXT	Internal use only.
PREEMPTIVE_OS_LIBRARYOPS	Internal use only.
PREEMPTIVE_OS_LOADLIBRARY	Internal use only.
PREEMPTIVE_OS_LOGONUSER	Internal use only.
PREEMPTIVE_OS_LOOKUPACCOUNTSID	Internal use only.
PREEMPTIVE_OS_MESSAGEQUEUEOPS	Internal use only.
PREEMPTIVE_OS_MOVEFILE	Internal use only.
PREEMPTIVE_OS_NETGROUPGETUSERS	Internal use only.
PREEMPTIVE_OS_NETLOCALGROUPGETMEMBERS	Internal use only.
PREEMPTIVE_OS_NETUSERGETGROUPS	Internal use only.
PREEMPTIVE_OS_NETUSERGETLOCALGROUPS	Internal use only.
PREEMPTIVE_OS_NETUSERMODALSGET	Internal use only.
PREEMPTIVE_OS_NETVALIDATEPASSWORDPOLICY	Internal use only.
PREEMPTIVE_OS_NETVALIDATEPASSWORDPOLICYFREE	Internal use only.
PREEMPTIVE_OS_OPENDIRECTORY	Internal use only.

Wait Type	Description
PREEMPTIVE_OS_PDH_WMI_INIT	Internal use only.
PREEMPTIVE_OS_PIPEOPS	Internal use only.
PREEMPTIVE_OS_PROCESSOPS	Internal use only.
PREEMPTIVE_OS_QUERYCONTEXTATTRIBUTES	Internal use only.
PREEMPTIVE_OS_QUERYREGISTRY	Internal use only.
PREEMPTIVE_OS_QUERYSECURITYCONTEXTTOKEN	Internal use only.
PREEMPTIVE_OS_REMOVEDIRECTORY	Internal use only.
PREEMPTIVE_OS_REPORTEVENT	Internal use only.
PREEMPTIVE_OS_REVERTTOSELF	Internal use only.
PREEMPTIVE_OS_RSFXDEVICEOPS	Internal use only.
PREEMPTIVE_OS_SECURITYOPS	Internal use only.
PREEMPTIVE_OS_SERVICEOPS	Internal use only.
PREEMPTIVE_OS_SETENDOFFILE	Internal use only.
PREEMPTIVE_OS_SETFILEPOINTER	Internal use only.
PREEMPTIVE_OS_SETFILEVALIDDATA	Internal use only.
PREEMPTIVE_OS_SETNAMEDSECURITYINFO	Internal use only.
PREEMPTIVE_OS_SQLCLROPS	Internal use only.
PREEMPTIVE_OS_SQMLAUNCH	Internal use only.
PREEMPTIVE_OS_VERIFYSIGNATURE	Internal use only.
PREEMPTIVE_OS_VERIFYTRUST	Internal use only.
PREEMPTIVE_OS_VSSOPS	Internal use only.
PREEMPTIVE_OS_WAITFORSINGLEOBJECT	Internal use only.

Wait Type	Description
PREEMPTIVE_OS_WINSOCKOPS	Internal use only.
PREEMPTIVE_OS_WRITEFILE	Internal use only.
PREEMPTIVE_OS_WRITEFILEGATHER	Internal use only.
PREEMPTIVE_OS_WSASETLASTERROR	Internal use only.
PREEMPTIVE_REENLIST	Internal use only.
PREEMPTIVE_RESIZELOG	Internal use only.
PREEMPTIVE_ROLLFORWARDREDO	Internal use only.
PREEMPTIVE_ROLLFORWARDUNDO	Internal use only.
PREEMPTIVE_SB_STOPENDPOINT	Internal use only.
PREEMPTIVE_SERVER_STARTUP	Internal use only.
PREEMPTIVE_SETRMINFO	Internal use only.
PREEMPTIVE_SHAREDMEM_GETDATA	Internal use only.
PREEMPTIVE_SNIOPEN	Internal use only.
PREEMPTIVE_SOSHOST	Internal use only.
PREEMPTIVE_SOSTESTING	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PREEMPTIVE_SP_SERVER_DIAGNOSTICS	Internal use only.
PREEMPTIVE_STARTRM	Internal use only.
PREEMPTIVE_STREAMFCB_CHECKPOINT	Internal use only.
PREEMPTIVE_STREAMFCB_RECOVER	Internal use only.
PREEMPTIVE_STRESSDRIVER	Identified for informational purposes only. Not supported.

Wait Type	Description
	Future compatibility is not guaranteed.
PREEMPTIVE_TESTING	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PREEMPTIVE_TRANSIMPORT	Internal use only.
PREEMPTIVE_UNMARSHALPROPAGATIONTOKEN	Internal use only.
PREEMPTIVE_VSS_CREATESNAPSHOT	Internal use only.
PREEMPTIVE_VSS_CREATEVOLUMESNAPSHOT	Internal use only.
PREEMPTIVE_XE_CALLBACKEXECUTE	Internal use only.
PREEMPTIVE_XE_CX_FILE_OPEN	Internal use only.
PREEMPTIVE_XE_CX_HTTP_CALL	Internal use only.
PREEMPTIVE_XE_DISPATCHER	Internal use only.
PREEMPTIVE_XE_ENGINEINIT	Internal use only.
PREEMPTIVE_XE_GETTARGETSTATE	Internal use only.
PREEMPTIVE_XE_SESSIONCOMMIT	Internal use only.
PREEMPTIVE_XE_TARGETFINALIZE	Internal use only.
PREEMPTIVE_XE_TARGETINIT	Internal use only.
PREEMPTIVE_XE_TIMERRUN	Internal use only.
PREEMPTIVE_XETESTING	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
PRINT_ROLLBACK_PROGRESS	Used to wait while user processes are ended in a database that has been transitioned by using the

Wait Type	Description
	ALTER DATABASE termination clause. For more information, see ALTER DATABASE (Transact-SQL).
PRU_ROLLBACK_DEFERRED	Internal use only.
PWAIT_ALL_COMPONENTS_INITIALIZED	Internal use only.
PWAIT_COOP_SCAN	Internal use only.
PWAIT_DIRECTLOGCONSUMER_GETNEXT	Internal use only.
PWAIT_EVENT_SESSION_INIT_MUTEX	Internal use only.
PWAIT_FABRIC_REPLICA_CONTROLLER_DATA_LOSS	Internal use only.
PWAIT_HADR_ACTION_COMPLETED	Internal use only.
PWAIT_HADR_CHANGE_NOTIFIER_TERMINATION_SYNC	Occurs when a background task is waiting for the termination of the background task that receives (via polling) Windows Server Failover Clustering notifications.
PWAIT_HADR_CLUSTER_INTEGRATION	An append, replace, and/or remove operation is waiting to grab a write lock on an Always On internal list (such as a list of networks, network addresses, or availability group listeners). Internal use only,
PWAIT_HADR_FAILOVER_COMPLETED	Internal use only.
PWAIT_HADR_JOIN	Internal use only.
PWAIT_HADR_OFFLINE_COMPLETED	An Always On drop availability group operation is waiting for the target availability group to go offline before destroying Windows Server Failover Clustering objects.

Wait Type	Description
PWAIT_HADR_ONLINE_COMPLETED	An Always On create or failover availability group operation is waiting for the target availability group to come online.
PWAIT_HADR_POST_ONLINE_COMPLETED	An Always On drop availability group operation is waiting for the termination of any background task that was scheduled as part of a previous command. For example, there may be a background task that is transitioning availability databases to the primary role. The DROP AVAILABILITY GROUP DDL must wait for this background task to terminate in order to avoid race conditions.
PWAIT_HADR_SERVER_READY_CONNECTIONS	Internal use only.
PWAIT_HADR_WORKITEM_COMPLETED	Internal wait by a thread waiting for an async work task to complete. This is an expected wait and is for CSS use.
PWAIT_HADRSIM	Internal use only.
PWAIT_LOG_CONSOLIDATION_IO	Internal use only.
PWAIT_LOG_CONSOLIDATION_POLL	Internal use only.
PWAIT_MD_LOGIN_STATS	Occurs during internal synchronization in metadata on login stats.
PWAIT_MD_RELATION_CACHE	Occurs during internal synchronization in metadata on table or index.
PWAIT_MD_SERVER_CACHE	Occurs during internal synchronization in metadata on linked servers.

Wait Type	Description
PWAIT_MD_UPGRADE_CONFIG	Occurs during internal synchronization in upgrading server wide configurations.
PWAIT_PREEMPTIVE_APP_USAGE_TIMER	Internal use only.
PWAIT_PREEMPTIVE_AUDIT_ACCESS_WINDOWSLOG	Internal use only.
PWAIT_QRY_BPMEMORY	Internal use only.
PWAIT_REPLICA_ONLINE_INIT_MUTEX	Internal use only.
PWAIT_RESOURCE_SEMAPHORE_FT_PARALLEL_QUERY_SYNC	Internal use only.
PWAIT_SBS_FILE_OPERATION	Internal use only.
PWAIT_XTP_FSSTORAGE_MAINTENANCE	Internal use only.
PWAIT_XTP_HOST_STORAGE_WAIT	Internal use only.
QDS_ASYNC_CHECK_CONSISTENCY_TASK	Internal use only.
QDS_ASYNC_PERSIST_TASK	Internal use only.
QDS_ASYNC_PERSIST_TASK_START	Internal use only.
QDS_ASYNC_QUEUE	Internal use only.
QDS_BCKG_TASK	Internal use only.
QDS_BLOOM_FILTER	Internal use only.
QDS_CLEANUP_STALE_QUERIES_TASK_MAIN_LOOP_SLEEP	Internal use only.
QDS_CTXS	Internal use only.
QDS_DB_DISK	Internal use only.
QDS_DYN_VECTOR	Internal use only.
QDS_EXCLUSIVE_ACCESS	Internal use only.
QDS_HOST_INIT	Internal use only.

Wait Type	Description
QDS_LOADDB	Internal use only.
QDS_PERSIST_TASK_MAIN_LOOP_SLEEP	Internal use only.
QDS_QDS_CAPTURE_INIT	Internal use only.
QDS_SHUTDOWN_QUEUE	Internal use only.
QDS_STMT	Internal use only.
QDS_STMT_DISK	Internal use only.
QDS_TASK_SHUTDOWN	Internal use only.
QDS_TASK_START	Internal use only.
QE_WARN_LIST_SYNC	Internal use only.
QPJOB_KILL	Indicates that an asynchronous automatic statistics update was canceled by a call to KILL as the update was starting to run. The terminating thread is suspended, waiting for it to start listening for KILL commands. A good value is less than one second.
QPJOB_WAITFOR_ABORT	Indicates that an asynchronous automatic statistics update was canceled by a call to KILL when it was running. The update has now completed but is suspended until the terminating thread message coordination is complete. This is an ordinary but rare state, and should be very short. A good value is less than one second.
QRY_MEM_GRANT_INFO_MUTEX	Occurs when Query Execution memory management tries to control access to static grant information list. This state lists information about the current

Wait Type	Description
	granted and waiting memory requests. This state is a simple access control state. There should never be a long wait on this state. If this mutex is not released, all new memory-using queries will stop responding.
QRY_PARALLEL_THREAD_MUTEX	Internal use only.
QRY_PROFILE_LIST_MUTEX	Internal use only.
QUERY_ERRHDL_SERVICE_DONE	Identified for informational purposes only. Not supported.
QUERY_WAIT_ERRHDL_SERVICE	Identified for informational purposes only. Not supported.
QUERY_EXECUTION_INDEX_SORT_EVENT_OPEN	Occurs in certain cases when offline create index build is run in parallel, and the different worker threads that are sorting synchronize access to the sort files.
QUERY_NOTIFICATION_MGR_MUTEX	Occurs during synchronization of the garbage collection queue in the Query Notification Manager.
QUERY_NOTIFICATION_SUBSCRIPTION_MUTEX	Occurs during state synchronization for transactions in Query Notifications.
QUERY_NOTIFICATION_TABLE_MGR_MUTEX	Occurs during internal synchronization within the Query Notification Manager.
QUERY_NOTIFICATION_UNITTEST_MUTEX	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
QUERY_OPTIMIZER_PRINT_MUTEX	Occurs during synchronization of query optimizer diagnostic output

Wait Type	Description
	production. This wait type only occurs if diagnostic settings have been enabled under direction of Microsoft Product Support.
QUERY_TASK_ENQUEUE_MUTEX	Internal use only.
QUERY_TRACEOUT	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
RBIO_WAIT_VLF	Internal use only.
RBIO_RG_STORAGE	Occurs when a Hyperscale database compute node is being throttled due to delayed log consumption at the page server(s).
RBIO_RG_STORAGE_CHECKPOINT	Internal use only - Occurs when a Hyperscale database Writebehind checkpoint on one/some of the PageServers is lagging far behind in terms of number of dirty pages that has to be flushed to remote store.
RBIO_RG_DESTAGE	Occurs when a Hyperscale database compute node is being throttled due to delayed log consumption by the long term log storage.
RBIO_RG_REPLICA	Occurs when a Hyperscale database compute node is being throttled due to delayed log consumption by the readable secondary replica node(s).
RBIO_RG_LOCALDESTAGE	Occurs when a Hyperscale database compute node is being

Wait Type	Description
	throttled due to delayed log consumption by the log service.
RBPEX_WRITEBEHIND_DB_STATE	Internal use only - Occurs when a Hyperscale database there is wait due to synchronization of Writebehind activities, for example, registering or unregistering database with Writebehind or reporting Writebehind throttling progress.
RECOVER_CHANGEDB	Occurs during synchronization of database status in warm standby database.
RECOVERY_MGR_LOCK	Internal use only.
REDO_THREAD_PENDING_WORK	Internal use only.
REDO_THREAD_SYNC	Internal use only.
REMOTE_BLOCK_IO	Internal use only.
REMOTE_DATA_ARCHIVE_MIGRATION_DMV	Internal use only.
REMOTE_DATA_ARCHIVE_SCHEMA_DMV	Internal use only.
REMOTE_DATA_ARCHIVE_SCHEMA_TASK_QUEUE	Internal use only.
REPL_CACHE_ACCESS	Occurs during synchronization on a replication article cache. During these waits, the replication log reader stalls, and data definition language (DDL) statements on a published table are blocked.
REPL_HISTORYCACHE_ACCESS	Internal use only.
REPL_SCHEMA_ACCESS	Occurs during synchronization of replication schema version information. This state exists when DDL statements are executed on the replicated object, and when

Wait Type	Description
	the log reader builds or consumes versioned schema based on DDL occurrence. Contention can be seen on this wait type if you have many published databases on a single publisher with transactional replication and the published databases are very active.
REPL_TRANFSINFO_ACCESS	Internal use only.
REPL_TRANHASHTABLE_ACCESS	Internal use only.
REPL_TRANTEXTINFO_ACCESS	Internal use only.
REPLICA_WRITES	Occurs while a task waits for completion of page writes to database snapshots or DBCC replicas.
REQUEST_DISPENSER_PAUSE	Occurs when a task is waiting for all outstanding I/O to complete, so that I/O to a file can be frozen for snapshot backup.
REQUEST_FOR_DEADLOCK_SEARCH	Occurs while the deadlock monitor waits to start the next deadlock search. This wait is expected between deadlock detections, and lengthy total waiting time on this resource does not indicate a problem.
RESERVED_MEMORY_ALLOCATION_EXT	Internal use only.
RESMGR_THROTTLED	Occurs when a new request comes in and is throttled based on the GROUP_MAX_REQUESTS setting.
RESOURCE_GOVERNOR_IDLE	Internal use only.
RESOURCE_QUEUE	Occurs during synchronization of

Wait Type	Description
	various internal resource queues.
RESOURCE_SEMAPHORE	Occurs when a query memory request during query execution cannot be granted immediately due to other concurrent queries. High waits and wait times may indicate excessive number of concurrent queries, or excessive memory request amounts. Excessive waits of this type may raise SQL error 8645, "A time out occurred while waiting for memory resources to execute the query. Rerun the query."
RESOURCE_SEMAPHORE_MUTEX	Occurs while a query waits for its request for a thread reservation to be fulfilled. It also occurs when synchronizing query compile and memory grant requests.
RESOURCE_SEMAPHORE_QUERY_COMPILE	Occurs when the number of concurrent query compilations reaches a throttling limit. High waits and wait times may indicate excessive compilations, recompiles, or uncacheable plans.
RESOURCE_SEMAPHORE_SMALL_QUERY	Occurs when memory request by a small query cannot be granted immediately due to other concurrent queries. Wait time should not exceed more than a few seconds, because the server transfers the request to the main query memory pool if it fails to grant the requested memory within a few seconds. High waits may indicate an excessive number of concurrent small queries while the main memory pool is blocked by waiting queries.

Wait Type	Description
RESTORE_FILEHANDLECACHE_ENTRYLOCK	Internal use only.
RESTORE_FILEHANDLECACHE_LOCK	Internal use only.
RG_RECONFIG	Internal use only.
ROWGROUP_OP_STATS	Internal use only.
ROWGROUP_VERSION	Internal use only.
RTDATA_LIST	Internal use only.
SATELLITE_CARGO	Internal use only.
SATELLITE_SERVICE_SETUP	Internal use only.
SATELLITE_TASK	Internal use only.
SBS_DISPATCH	Internal use only.
SBS_RECEIVE_TRANSPORT	Internal use only.
SBS_TRANSPORT	Internal use only.
SCAN_CHAR_HASH_ARRAY_INITIALIZATION	Internal use only.
SEC_DROP_TEMP_KEY	Occurs after a failed attempt to drop a temporary security key before a retry attempt.
SECURITY_CNG_PROVIDER_MUTEX	Internal use only.
SECURITY_CRYPTOCONTEXT_MUTEX	Internal use only.
SECURITY_DBE_STATE_MUTEX	Internal use only.
SECURITY_KEYRING_RWLOCK	Internal use only.
SECURITY_MUTEX	Occurs when there is a wait for mutexes that control access to the global list of Extensible Key Management (EKM) cryptographic

Wait Type	Description
	providers and the session-scoped list of EKM sessions.
SECURITY_RULETABLE_MUTEX	Internal use only.
SEMPLAT_DSI_BUILD	Internal use only.
SEQUENCE_GENERATION	Internal use only.
SEQUENTIAL_GUID	Occurs while a new sequential GUID is being obtained.
SERVER_IDLE_CHECK	Occurs during synchronization of SQL Server instance idle status when a resource monitor is attempting to declare a SQL Server instance as idle or trying to wake up.
SERVER_RECONFIGURE	Internal use only.
SESSION_WAIT_STATS_CHILDREN	Internal use only.
SHARED_DELTASTORE_CREATION	Internal use only.
SHUTDOWN	Occurs while a shutdown statement waits for active connections to exit.
SLEEP_BPOOL_FLUSH	Occurs when a checkpoint is throttling the issuance of new I/Os in order to avoid flooding the disk subsystem.
SLEEP_BUFFERPOOL_HELPPLW	Internal use only.
SLEEP_DBSTARTUP	Occurs during database startup while waiting for all databases to recover.
SLEEP_DCOMSTARTUP	Occurs once at most during SQL Server instance startup while

Wait Type	Description
	waiting for DCOM initialization to complete.
SLEEP_MASTERDBREADY	Internal use only.
SLEEP_MASTERMDREADY	Internal use only.
SLEEP_MASTERUPGRADED	Internal use only.
SLEEP_MEMORYPOOL_ALLOCATEPAGES	Internal use only.
SLEEP_MSDBSTARTUP	Occurs when SQL Trace waits for the msdb database to complete startup.
SLEEP_RETRY_VIRTUALALLOC	Internal use only.
SLEEP_SYSTEMTASK	Occurs during the start of a background task while waiting for tempdb to complete startup.
SLEEP_TASK	Occurs when a task sleeps while waiting for a generic event to occur.
SLEEP_TEMPDBSTARTUP	Occurs while a task waits for tempdb to complete startup.
SLEEP_WORKSPACE_ALLOCATEPAGE	Internal use only.
SLO_UPDATE	Internal use only.
SMSYNC	Internal use only.
SNI_CONN_DUP	Internal use only.
SNI_CRITICAL_SECTION	Occurs during internal synchronization within SQL Server networking components.
SNI_HTTP_WAITFOR_0_DISCON	Occurs during SQL Server shutdown, while waiting for

Wait Type	Description
	outstanding HTTP connections to exit.
SNI_LISTENER_ACCESS	Occurs while waiting for non-uniform memory access (NUMA) nodes to update state change. Access to state change is serialized.
SNI_TASK_COMPLETION	Occurs when there is a wait for all tasks to finish during a NUMA node state change.
SNI_WRITE_ASYNC	Internal use only.
SOAP_READ	Occurs while waiting for an HTTP network read to complete.
SOAP_WRITE	Occurs while waiting for an HTTP network write to complete.
SOCKETDUPLICATEQUEUE_CLEANUP	Internal use only.
SOS_CALLBACK_REMOVAL	Occurs while performing synchronization on a callback list in order to remove a callback. It is not expected for this counter to change after server initialization is completed.
SOS_DISPATCHER_MUTEX	Occurs during internal synchronization of the dispatcher pool. This includes when the pool is being adjusted.
SOS_LOCALALLOCATORLIST	Occurs during internal synchronization in the SQL Server memory manager.
SOS_MEMORY_TOPLEVELBLOCKALLOCATOR	Internal use only.
SOS_MEMORY_USAGE_ADJUSTMENT	Occurs when memory usage is being adjusted among pools.

Wait Type	Description
SOS_OBJECT_STORE_DESTROY_MUTEX	Occurs during internal synchronization in memory pools when destroying objects from the pool.
SOS_PHYS_PAGE_CACHE	Accounts for the time a thread waits to acquire the mutex it must acquire before it allocates physical pages or before it returns those pages to the operating system. Waits on this type only appear if the instance of SQL Server uses AWE memory.
SOS_PROCESS_AFFINITY_MUTEX	Occurs during synchronizing of access to process affinity settings.
SOS_RESERVEDMEMBLOCKLIST	Occurs during internal synchronization in the SQL Server Memory Manager.
SOS_SCHEDULER_YIELD	Occurs when a task voluntarily yields the scheduler for other tasks to execute. During this wait, the task is waiting in a runnable queue for its quantum to be renewed, i.e. waiting to be scheduled to run on the CPU again. Prolonged waits on this wait type most frequently indicate opportunities to optimize queries that perform index or table scans. Focus on plan regression, missing index, stats updates, query re-writes. Optimizing runtimes reduces the need for tasks to be yielding multiple times. If query times for such CPU-consuming tasks are acceptable, then this wait type is expected and can be ignored.

Wait Type	Description
SOS_SMALL_PAGE_ALLOC	Occurs during the allocation and freeing of memory that is managed by some memory objects.
SOS_STACKSTORE_INIT_MUTEX	Occurs during synchronization of internal store initialization.
SOS_SYNC_TASK_ENQUEUE_EVENT	Occurs when a task is started in a synchronous manner. Most tasks in SQL Server are started in an asynchronous manner, in which control returns to the starter immediately after the task request has been placed on the work queue.
SOS_VIRTUALMEMORY_LOW	Occurs when a memory allocation waits for a Resource Manager to free up virtual memory.
SOSHOST_EVENT	Occurs when a hosted component, such as CLR, waits on a SQL Server event synchronization object.
SOSHOST_INTERNAL	Occurs during synchronization of memory manager callbacks used by hosted components, such as CLR.
SOSHOST_MUTEX	Occurs when a hosted component, such as CLR, waits on a SQL Server mutex synchronization object.
SOSHOST_RWLOCK	Occurs when a hosted component, such as CLR, waits on a SQL Server reader-writer synchronization object.
SOSHOST_SEMAPHORE	Occurs when a hosted component, such as CLR, waits on

Wait Type	Description
	a SQL Server semaphore synchronization object.
SOSHOST_SLEEP	Occurs when a hosted task sleeps while waiting for a generic event to occur. Hosted tasks are used by hosted components such as CLR.
SOSHOST_TRACELOCK	Occurs during synchronization of access to trace streams.
SOSHOST_WAITFORDONE	Occurs when a hosted component, such as CLR, waits for a task to complete.
SP_PREEMPTIVE_SERVER_DIAGNOSTICS_SLEEP	Internal use only.
SP_SERVER_DIAGNOSTICS_BUFFER_ACCESS	Internal use only.
SP_SERVER_DIAGNOSTICS_INIT_MUTEX	Internal use only.
SP_SERVER_DIAGNOSTICS_SLEEP	Internal use only.
SQLCLR_APPDOMAIN	Occurs while CLR waits for an application domain to complete startup.
SQLCLR_ASSEMBLY	Occurs while waiting for access to the loaded assembly list in the appdomain.
SQLCLR_DEADLOCK_DETECTION	Occurs while CLR waits for deadlock detection to complete.
SQLCLR_QUANTUM_PUNISHMENT	Occurs when a CLR task is throttled because it has exceeded its execution quantum. This throttling is done in order to reduce the effect of this resource-intensive task on other tasks.
SQLSORT_NORMMUTEX	Occurs during internal synchronization, while initializing

Wait Type	Description
	internal sorting structures.
SQLSORT_SORTMUTEX	Occurs during internal synchronization, while initializing internal sorting structures.
SQLTRACE_BUFFER_FLUSH	Occurs when a task is waiting for a background task to flush trace buffers to disk every four seconds.
SQLTRACE_FILE_BUFFER	Occurs during synchronization on trace buffers during a file trace.
SQLTRACE_FILE_READ_IO_COMPLETION	Internal use only.
SQLTRACE_FILE_WRITE_IO_COMPLETION	Internal use only.
SQLTRACE_INCREMENTAL_FLUSH_SLEEP	Internal use only.
SQLTRACE_LOCK	Internal use only.
SQLTRACE_PENDING_BUFFER_WRITERS	Internal use only.
SQLTRACE_SHUTDOWN	Occurs while trace shutdown waits for outstanding trace events to complete.
SQLTRACE_WAIT_ENTRIES	Occurs while a SQL Trace event queue waits for packets to arrive on the queue.
SRVPROC_SHUTDOWN	Occurs while the shutdown process waits for internal resources to be released to shutdown cleanly.
STARTUP_DEPENDENCY_MANAGER	Internal use only.
TDS_BANDWIDTH_STATE	Internal use only.
TDS_INIT	Internal use only.
TDS_PROXY_CONTAINER	Internal use only.

Wait Type	Description
TEMPOBJ	Occurs when temporary object drops are synchronized. This wait is rare, and only occurs if a task has requested exclusive access for temp table drops.
TEMPORAL_BACKGROUND_PROCEED_CLEANUP	Internal use only.
TERMINATE_LISTENER	Internal use only.
THREADPOOL	Occurs when a task (query or login/logout) is waiting for a worker thread to execute it. This can indicate that the maximum worker thread setting is misconfigured, or that, most commonly, batch executions are taking unusually long, thus reducing the number of worker threads available to satisfy other batches. Examine the performance of batches (queries) and reduce query duration by either reducing bottlenecks (blocking, parallelism, I/O, latch waits), or providing proper indexing or query design.
TIMEPRIV_TIMEPERIOD	Occurs during internal synchronization of the Extended Events timer.
TRACE_EVTNOTIF	Internal use only.
TRACEWRITE	Occurs when the SQL Trace rowset trace provider waits for either a free buffer or a buffer with events to process.
TRAN_MARKLATCH_DT	Occurs when waiting for a destroy mode latch on a transaction mark latch. Transaction mark latches are used for synchronization of

Wait Type	Description
	commits with marked transactions.
TRAN_MARKLATCH_EX	Occurs when waiting for an exclusive mode latch on a marked transaction. Transaction mark latches are used for synchronization of commits with marked transactions.
TRAN_MARKLATCH_KP	Occurs when waiting for a keep mode latch on a marked transaction. Transaction mark latches are used for synchronization of commits with marked transactions.
TRAN_MARKLATCH_NL	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
TRAN_MARKLATCH_SH	Occurs when waiting for a shared mode latch on a marked transaction. Transaction mark latches are used for synchronization of commits with marked transactions.
TRAN_MARKLATCH_UP	Occurs when waiting for an update mode latch on a marked transaction. Transaction mark latches are used for synchronization of commits with marked transactions.
TRANSACTION_MUTEX	Occurs during synchronization of access to a transaction by multiple batches.
UCS_ENDPOINT_CHANGE	Internal use only.
UCS_MANAGER	Internal use only.

Wait Type	Description
UCS_MEMORY_NOTIFICATION	Internal use only.
UCS_SESSION_REGISTRATION	Internal use only.
UCS_TRANSPORT	Internal use only.
UCS_TRANSPORT_STREAM_CHANGE	Internal use only.
UTIL_PAGE_ALLOC	Occurs when transaction log scans wait for memory to be available during memory pressure.
VDI_CLIENT_COMPLETECOMMAND	Internal use only.
VDI_CLIENT_GETCOMMAND	Internal use only.
VDI_CLIENT_OPERATION	Internal use only.
VDI_CLIENT_OTHER	Internal use only.
VERSIONING_COMMITTING	Internal use only.
VIA_ACCEPT	Occurs when a Virtual Interface Adapter (VIA) provider connection is completed during startup.
VIEW_DEFINITION_MUTEX	Occurs during synchronization on access to cached view definitions.
WAIT_FOR_RESULTS	Occurs when waiting for a query notification to be triggered.
WAIT_ON_SYNC_STATISTICS_REFRESH	Occurs when waiting for synchronous statistics update to complete before query compilation and execution can resume.
WAIT_SCRIPTDEPLOYMENT_REQUEST	Internal use only.
WAIT_SCRIPTDEPLOYMENT_WORKER	Internal use only.
WAIT_XLOGREAD_SIGNAL	Internal use only.

Wait Type	Description
WAIT_XTP_ASYNC_TX_COMPLETION	Internal use only.
WAIT_XTP_CKPT_AGENT_WAKEUP	Internal use only.
WAIT_XTP_CKPT_CLOSE	Occurs when waiting for a checkpoint to complete.
WAIT_XTP_CKPT_ENABLED	Occurs when checkpointing is disabled, and waiting for checkpointing to be enabled.
WAIT_XTP_CKPT_STATE_LOCK	Occurs when synchronizing checking of checkpoint state.
WAIT_XTP_COMPILE_WAIT	Internal use only.
WAIT_XTP_GUEST	Occurs when the database memory allocator needs to stop receiving low-memory notifications.
WAIT_XTP_HOST_WAIT	Occurs when waits are triggered by the database engine and implemented by the host.
WAIT_XTP_OFFLINE_CKPT_BEFORE_REDO	Internal use only.
WAIT_XTP_OFFLINE_CKPT_LOG_IO	Occurs when offline checkpoint is waiting for a log read IO to complete.
WAIT_XTP_OFFLINE_CKPT_NEW_LOG	Occurs when offline checkpoint is waiting for new log records to scan.
WAIT_XTP_PROCEDURE_ENTRY	Occurs when a drop procedure is waiting for all current executions of that procedure to complete.
WAIT_XTP_RECOVERY	Occurs when database recovery is waiting for recovery of memory-optimized objects to finish.

Wait Type	Description
WAIT_XTP_SERIAL_RECOVERY	Internal use only.
WAIT_XTP_SWITCH_TO_INACTIVE	Internal use only.
WAIT_XTP_TASK_SHUTDOWN	Occurs when waiting for an In-Memory OLTP thread to complete.
WAIT_XTP_TRAN_DEPENDENCY	Occurs when waiting for transaction dependencies.
WAITFOR	Occurs as a result of a WAITFOR Transact-SQL statement. The duration of the wait is determined by the parameters to the statement. This is a user-initiated wait.
WAITFOR_PER_QUEUE	Internal use only.
WAITFOR_TASKSHUTDOWN	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
WAITSTAT_MUTEX	Occurs during synchronization of access to the collection of statistics used to populate sys.dm_os_wait_stats.
WCC	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
WINDOW_AGGREGATES_MULTIPASS	Internal use only.
WINFAB_API_CALL	Internal use only.
WINFAB_REPLICA_BUILD_OPERATION	Internal use only.
WINFAB_REPORT_FAULT	Internal use only.

Wait Type	Description
WORKTBL_DROP	Occurs while pausing before retrying, after a failed worktable drop.
WRITE_COMPLETION	Occurs when a write operation is in progress.
WRITELOG	Occurs while waiting for a log flush to complete. Common operations that cause log flushes are transaction commits and checkpoints. Common reasons for long waits on WRITELOG are: disk latency (where transaction log files reside), the inability for I/O to keep up with transactions, or, a large number of transaction log operations and flushes (commits, rollback)
XACT_OWN_TRANSACTION	Occurs while waiting to acquire ownership of a transaction.
XACT_RECLAIM_SESSION	Occurs while waiting for the current owner of a session to release ownership of the session.
XACTLOCKINFO	Occurs during synchronization of access to the list of locks for a transaction. In addition to the transaction itself, the list of locks is accessed by operations such as deadlock detection and lock migration during page splits.
XACTWORKSPACE_MUTEX	Occurs during synchronization of defections from a transaction, as well as the number of database locks between enlist members of a transaction.
XDB_CONN_DUP_HASH	Internal use only.

Wait Type	Description
XDES_HISTORY	Internal use only.
XDES_OUT_OF_ORDER_LIST	Internal use only.
XDES_SNAPSHOT	Internal use only.
XDESTSVERMGR	Internal use only.
XE_BUFFERMGR_ALLPROCESSED_EVENT	Occurs when Extended Events session buffers are flushed to targets. This wait occurs on a background thread.
XE_BUFFERMGR_FREEBUF_EVENT	Occurs when either of the following conditions is true:
XE_CALLBACK_LIST	Internal use only.
XE_CX_FILE_READ	Internal use only.
XE_DISPATCHER_CONFIG_SESSION_LIST	Occurs when an Extended Events session that is using asynchronous targets is started or stopped. This wait indicates either of the following:
XE_DISPATCHER_JOIN	Occurs when a background thread that is used for Extended Events sessions is terminating.
XE_DISPATCHER_WAIT	Occurs when a background thread that is used for Extended Events sessions is waiting for event buffers to process.
XE_FILE_TARGET_TVF	Internal use only.
XE_LIVE_TARGET_TVF	Internal use only.
XE_MODULEMGR_SYNC	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.

Wait Type	Description
XE_OLS_LOCK	Identified for informational purposes only. Not supported. Future compatibility is not guaranteed.
XE_PACKAGE_LOCK_BACKOFF	Identified for informational purposes only. Not supported.
XE_SERVICES_EVENTMANUAL	Internal use only.
XE_SERVICES_MUTEX	Internal use only.
XE_SERVICES_RWLOCK	Internal use only.
XE_SESSION_CREATE_SYNC	Internal use only.
XE_SESSION_FLUSH	Internal use only.
XE_SESSION_SYNC	Internal use only.
XE_STM_CREATE	Internal use only.
XE_TIMER_EVENT	Internal use only.
XE_TIMER_MUTEX	Internal use only.
XE_TIMER_TASK_DONE	Internal use only.
XIO_CREDENTIAL_MGR_RWLOCK	Internal use only.
XIO_CREDENTIAL_RWLOCK	Internal use only.
XIO_EDS_MGR_RWLOCK	Internal use only.
XIO_EDS_RWLOCK	Internal use only.
XIO_IOSTATS_BLOBLIST_RWLOCK	Internal use only.
XIO_IOSTATS_FCBLIST_RWLOCK	Internal use only.
XIO_LEASE_RENEW_MGR_RWLOCK	Internal use only.
XTP_HOST_DB_COLLECTION	Internal use only.

Wait Type	Description
XTP_HOST_LOG_ACTIVITY	Internal use only.
XTP_HOST_PARALLEL_RECOVERY	Internal use only.
XTP_PREEMPTIVE_TASK	Internal use only.
XTP_TRUNCATION_LSN	Internal use only.
XTPPROC_CACHE_ACCESS	Occurs when for accessing all natively compiled stored procedure cache objects.
XTPPROC_PARTITIONED_STACK_CREATE	Occurs when allocating per-NUMA node natively compiled stored procedure cache structures (must be done single threaded) for a given procedure.

How good have you found this content?

