Critical Wait Types in SQL Server — Meaning + Causes + Resolution Steps

1. PAGEIOLATCH_ — Slow I/O on Data Files

Meaning: SQL Server is waiting for data pages to be read from disk into memory (I/O latency).
Common Causes: Slow storage, overloaded disks, missing indexes, memory pressure.
Resolutions:

- Check data file latency using sys.dm_io_virtual_file_stats.
- Move data files to faster storage (SSD/NVMe).
- Add more memory to reduce physical reads.
- Optimize queries and fix missing indexes.
- Reduce page splits and table scans.

2. WRITELOG — Log File I/O Bottleneck

Meaning: SQL Server is waiting to write log records to the transaction log.
Common Causes: Slow log storage, VLF fragmentation, frequent small transactions.
Resolutions:

- Place log file on the fastest available disk.
- Pre-size log file and avoid autogrowth fragmentation.
- Reduce unnecessary log activity (batch commits).
- Tune high-frequency write workloads (ETL, large inserts).
- Fix excessive checkpoints or aggressive logging.

3. LCK_M_ — Locking / Blocking

Meaning: Queries are waiting for locks held by other processes.
Common Causes: Long-running queries, poor indexing, large updates running during peak hours.

Resolutions:

- Identify blockers with sp_whoisactive or sys.dm_tran_locks.
- Add missing indexes to avoid table scans.
- Break large transactions into smaller batches.
- Use appropriate isolation levels (e.g., RCSI).
- Evaluate deadlock patterns and fix query design.

4. SOS_SCHEDULER_YIELD — CPU Pressure

Meaning: Query exhausted its CPU quantum and needs to yield; indicates CPU pressure.
Common Causes: Parallelism issues, CPU-intensive queries, insufficient CPU cores.
Resolutions:

- Check top CPU consumers using sys.dm_exec_query_stats.
- Fix expensive queries (missing indexes, bad joins).
- Reduce MAXDOP or use query-level MAXDOP hints.
- Upgrade or scale CPUs where necessary.
- Reduce overuse of scalar functions or RBAR processing.

5. CXPACKET — Parallelism Imbalance

Meaning: Occurs when threads in a parallel plan are not finishing at the same time.
Common Causes: Poor parallel plan, skewed distributions, incorrect MAXDOP/CTFP settings.
Resolutions:

- Set appropriate MAXDOP (generally 4 or 8).
- Tune cost threshold for parallelism (raise to 30–50).
- Fix skewed data distributions.
- Improve indexes to avoid unnecessary parallel scans.

6. ASYNC_NETWORK_IO — Slow Result Consumption

Meaning: SQL Server is sending results faster than the application can consume them.

Common Causes: Application fetching row-by-row, slow client, network bottlenecks.

Resolutions:

- Fix client-side cursor loops (use set-based operations).
- Reduce result set size (select only needed columns).
- Check for network latency and packet drops.
- Update application drivers and connection libraries.

Additional Important Wait Types

7. PAGE_LATCH_ — In-Memory Contention (Not Disk I/O)

Meaning: Contention for in-memory latch/no I/O involved.

Causes: Hot pages (same page accessed by many threads), tempdb allocation.

Resolution:

- Add more tempdb data files.
- Optimize queries hitting the same hot page.
- Use bucketizing or hashing to distribute inserts.

8. RESOURCE_SEMAPHORE — Query Memory Pressure

Meaning: SQL Server cannot grant memory to new queries due to large memory grants.

Causes: Memory-hungry queries, poor cardinality estimates.

Resolution:

- Fix missing stats and outdated statistics.
- Identify large-memory-grant queries with Query Store.
- Add more RAM if workload requires it.
- Enable row-mode instead of batch-mode where needed.

9. THREADPOOL — Worker Thread Exhaustion

Meaning: SQL Server has no free worker threads; new requests must wait.

Causes: Long blocking chains, excessive parallel queries, connection storms.

Resolution:

- Fix blocking first (root cause).
- Reduce MAXDOP to lower parallel thread usage.
- Increase max worker threads (rare cases).
- Refactor application to avoid spikes.

10. PAGELATCH_UP / EX on TempDB — TempDB Contention

Meaning: Heavy latch contention in tempdb, usually on PFS/SGAM/GAM pages.

Resolution:

- Increase tempdb files (start with 8, scale up).
- Equal file sizes and identical autogrowth.
- Use trace flags (older versions) or enable TF 1117/1118 equivalents in modern SQL.

11. IO_COMPLETION — General I/O Delays

Meaning: SQL Server waiting for an I/O operation to complete.

Resolution:

- Analyze storage latency (PerfMon / DMVs).
- Offload heavy ETL to non-peak hours.
- Move to faster disks if necessary.

Summarized View

Wait Type	**Meaning**	**Resolution**
PAGEIOLATCH	Slow disk reads	Faster storage, more RAM, indexing
WRITELOG	Slow log writes	Faster log disk, presizing, batching
LCK_M_	Blocking	Indexing, fix blockers, RCSI
SOS_SCHEDULER_YIELD	CPU pressure	Tune queries, adjust MAXDOP
CXPACKET	Parallelism imbalance	Tune CTFP & MAXDOP
ASYNC_NETWORK_IO	Slow client consumption	Fix client loops, network
PAGE_LATCH	Tempdb / in-memory contention	Add tempdb files
RESOURCE_SEMAPHORE	Memory pressure	Fix stats, big memory grants
THREADPOOL	Worker thread starvation	Fix blocking, reduce parallelism
IO_COMPLETION	Slow I/O	Analyze & improve storage

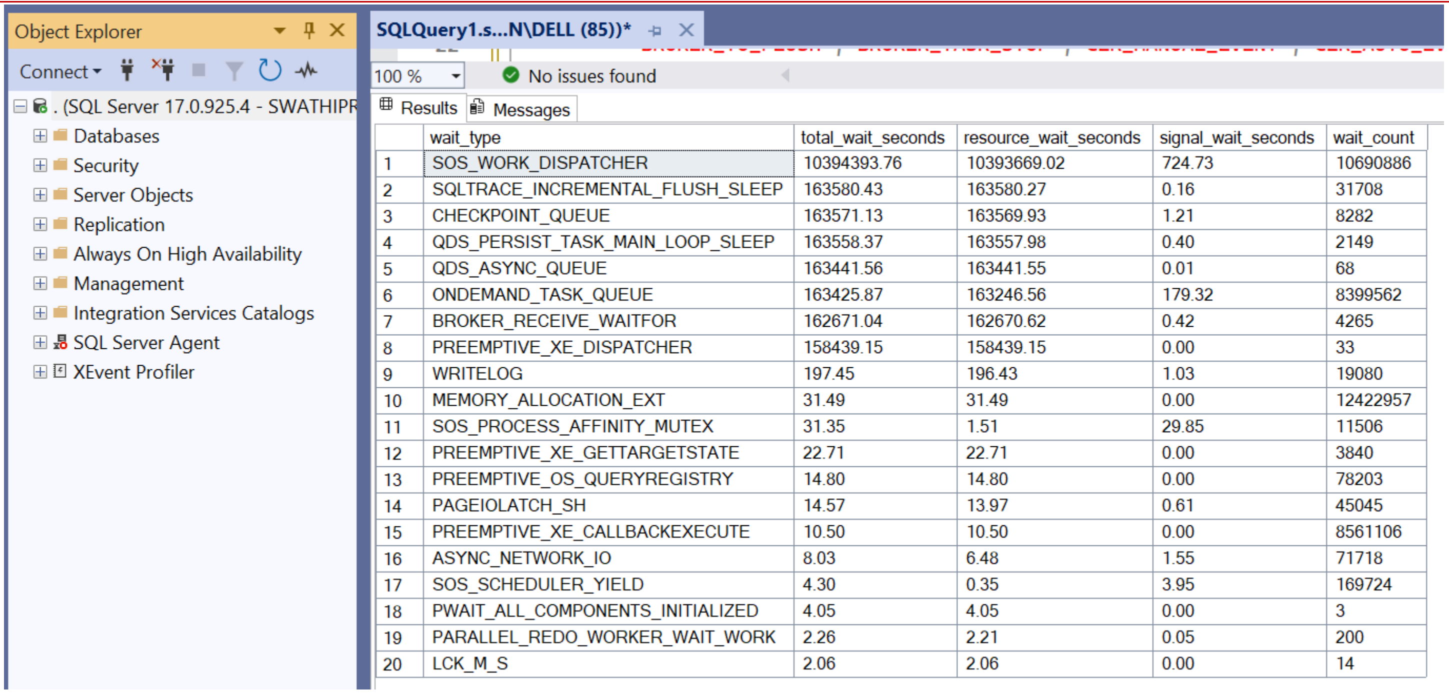
SQL Server Wait Statistics Diagnostic Kit.

1. MASTER T-SQL SCRIPT — Critical Wait Type Analyzer

This script shows **top waits**, categorizes them, and highlights problem areas.

```
/*================================
  SQL Server Critical Wait Analysis
================================*/

-- Clear wait stats (optional)
-- DBCC SQLPERF('sys.dm_os_wait_stats', CLEAR);

WITH Waits AS
(
   SELECT
      wait_type,
      wait_time_ms / 1000.0 AS wait_s,
      (wait_time_ms - signal_wait_time_ms) / 1000.0 AS resource_s,
      signal_wait_time_ms / 1000.0 AS signal_s,
      waiting_tasks_count AS waits
   FROM sys.dm_os_wait_stats
   WHERE wait_type NOT IN
   (
      'CLR_SEMAPHORE','LAZYWRITER_SLEEP','RESOURCE_QUEUE','SLEEP_TASK',
      'SLEEP_SYSTEMTASK','SQLTRACE_BUFFER_FLUSH','WAITFOR','LOGMGR_QUEUE',
      'REQUEST_FOR_DEADLOCK_SEARCH','XE_TIMER_EVENT','XE_DISPATCHER_WAIT',
      'BROKER_TO_FLUSH','BROKER_TASK_STOP','CLR_MANUAL_EVENT','CLR_AUTO_EVENT',
      'DISPATCHER_QUEUE_SEMAPHORE','FT_IFTS_SCHEDULER_IDLE_WAIT',
      'BROKER_EVENTHANDLER','BAD_PAGE_PROCESS','DIRTY_PAGE_POLL',
      'HADR_FILESTREAM_IOMGR_IOCOMPLETION','SP_SERVER_DIAGNOSTICS_SLEEP'
   )
)
SELECT TOP 20
   wait_type,
   CAST(wait_s AS DECIMAL(12,2)) AS total_wait_seconds,
   CAST(resource_s AS DECIMAL(12,2)) AS resource_wait_seconds,
   CAST(signal_s AS DECIMAL(12,2)) AS signal_wait_seconds,
   waits AS wait_count
FROM Waits
ORDER BY total_wait_seconds DESC;
```

Sample output:

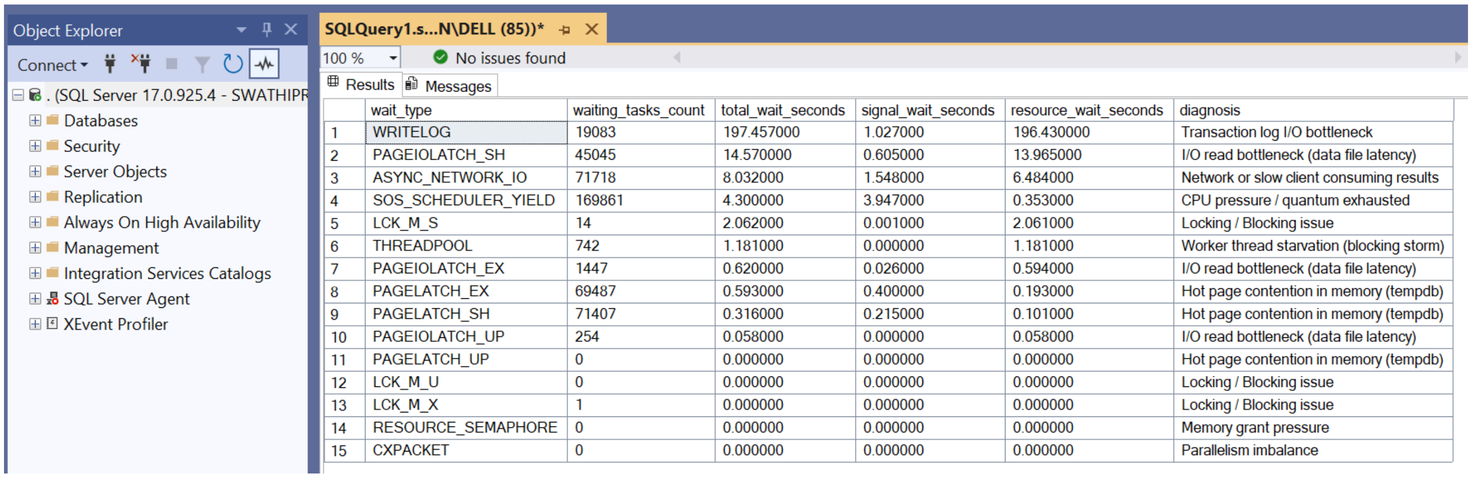


	wait_type	total_wait_seconds	resource_wait_seconds	signal_wait_seconds	wait_count
1	SOS_WORK_DISPATCHER	10394393.76	10393669.02	724.73	10690886
2	SQLTRACE_INCREMENTAL_FLUSH_SLEEP	163580.43	163580.27	0.16	31708
3	CHECKPOINT_QUEUE	163571.13	163569.93	1.21	8282
4	QDS_PERSIST_TASK_MAIN_LOOP_SLEEP	163558.37	163557.98	0.40	2149
5	QDS_ASYNC_QUEUE	163441.56	163441.55	0.01	68
6	ONDEMAND_TASK_QUEUE	163425.87	163246.56	179.32	8399562
7	BROKER_RECEIVE_WAITFOR	162671.04	162670.62	0.42	4265
8	PREEMPTIVE_XE_DISPATCHER	158439.15	158439.15	0.00	33
9	WRITELOG	197.45	196.43	1.03	19080
10	MEMORY_ALLOCATION_EXT	31.49	31.49	0.00	12422957
11	SOS_PROCESS_AFFINITY_MUTEX	31.35	1.51	29.85	11506
12	PREEMPTIVE_XE_GETTARGETSTATE	22.71	22.71	0.00	3840
13	PREEMPTIVE_OS_QUERYREGISTRY	14.80	14.80	0.00	78203
14	PAGEIOLATCH_SH	14.57	13.97	0.61	45045
15	PREEMPTIVE_XE_CALLBACKEXECUTE	10.50	10.50	0.00	8561106
16	ASYNC_NETWORK_IO	8.03	6.48	1.55	71718
17	SOS_SCHEDULER_YIELD	4.30	0.35	3.95	169724
18	PWAIT_ALL_COMPONENTS_INITIALIZED	4.05	4.05	0.00	3
19	PARALLEL_REDO_WORKER_WAIT_WORK	2.26	2.21	0.05	200
20	LCK_M_S	2.06	2.06	0.00	14

2. Deep-Dive Diagnosis for Each Critical Wait Type

```
/*=============================================
  Detailed Diagnosis for Critical Wait Types
=============================================*/
SELECT
    wait_type,
    waiting_tasks_count,
    wait_time_ms / 1000.0 AS total_wait_seconds,
    signal_wait_time_ms / 1000.0 AS signal_wait_seconds,
    (wait_time_ms - signal_wait_time_ms) / 1000.0 AS resource_wait_seconds,
    CASE
        WHEN wait_type LIKE 'PAGEIOLATCH%' THEN 'I/O read bottleneck (data file latency)'
        WHEN wait_type = 'WRITELOG' THEN 'Transaction log I/O bottleneck'
        WHEN wait_type LIKE 'LCK_M%' THEN 'Locking / Blocking issue'
        WHEN wait_type = 'SOS_SCHEDULER_YIELD' THEN 'CPU pressure / quantum exhausted'
        WHEN wait_type = 'CXPACKET' THEN 'Parallelism imbalance'
        WHEN wait_type = 'ASYNC_NETWORK_IO' THEN 'Network or slow client consuming results'
        WHEN wait_type LIKE 'PAGELATCH%' THEN 'Hot page contention in memory (tempdb)'
        WHEN wait_type = 'RESOURCE_SEMAPHORE' THEN 'Memory grant pressure'
        WHEN wait_type = 'THREADPOOL' THEN 'Worker thread starvation (blocking storm)'
        ELSE 'Other'
    END AS diagnosis
FROM sys.dm_os_wait_stats
WHERE wait_type IN (
    'PAGEIOLATCH_SH','PAGEIOLATCH_EX','PAGEIOLATCH_UP',
    'WRITELOG', 'LCK_M_S','LCK_M_X','LCK_M_U',
    'SOS_SCHEDULER_YIELD','CXPACKET','ASYNC_NETWORK_IO',
    'PAGELATCH_SH','PAGELATCH_EX','PAGELATCH_UP',
    'RESOURCE_SEMAPHORE','THREADPOOL'
)
ORDER BY total_wait_seconds DESC;
```

Sample output:


SQLQuery1.s...N\DELL (85))*

No issues found

	wait_type	waiting_tasks_count	total_wait_seconds	signal_wait_seconds	resource_wait_seconds	diagnosis
1	WRITELOG	19083	197.457000	1.027000	196.430000	Transaction log I/O bottleneck
2	PAGEIOLATCH_SH	45045	14.570000	0.605000	13.965000	I/O read bottleneck (data file latency)
3	ASYNC_NETWORK_IO	71718	8.032000	1.548000	6.484000	Network or slow client consuming results
4	SOS_SCHEDULER_YIELD	169861	4.300000	3.947000	0.353000	CPU pressure / quantum exhausted
5	LCK_M_S	14	2.062000	0.001000	2.061000	Locking / Blocking issue
6	THREADPOOL	742	1.181000	0.000000	1.181000	Worker thread starvation (blocking storm)
7	PAGEIOLATCH_EX	1447	0.620000	0.026000	0.594000	I/O read bottleneck (data file latency)
8	PAGELATCH_EX	69487	0.593000	0.400000	0.193000	Hot page contention in memory (tempdb)
9	PAGELATCH_SH	71407	0.316000	0.215000	0.101000	Hot page contention in memory (tempdb)
10	PAGEIOLATCH_UP	254	0.058000	0.000000	0.058000	I/O read bottleneck (data file latency)
11	PAGELATCH_UP	0	0.000000	0.000000	0.000000	Hot page contention in memory (tempdb)
12	LCK_M_U	0	0.000000	0.000000	0.000000	Locking / Blocking issue
13	LCK_M_X	1	0.000000	0.000000	0.000000	Locking / Blocking issue
14	RESOURCE_SEMAPHORE	0	0.000000	0.000000	0.000000	Memory grant pressure
15	CXPACKET	0	0.000000	0.000000	0.000000	Parallelism imbalance

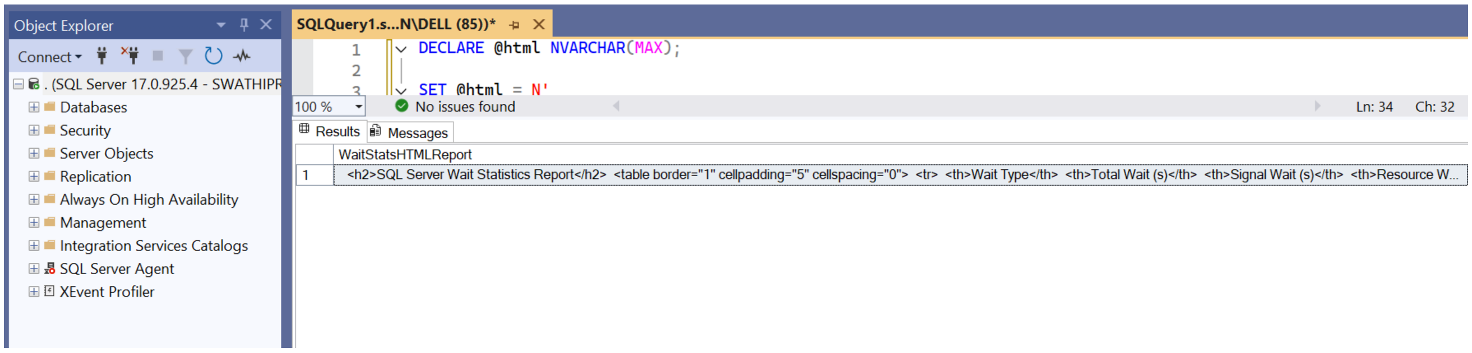
3. HTML Report (Email-Friendly) — Top Wait Statistics

Use this for SQL Agent job alerts.

```
DECLARE @html NVARCHAR(MAX);

SET @html = N'
<h2>SQL Server Wait Statistics Report</h2>
<table border="1" cellpadding="5" cellspacing="0">
<tr>
<th>Wait Type</th>
<th>Total Wait (s)</th>
<th>Signal Wait (s)</th>
<th>Resource Wait (s)</th>
<th>Wait Count</th>
<th>Category</th>
</tr>';

SELECT @html = @html +
    '<tr><td>' + wait_type + '</td><td>' +
    CAST(wait_time_ms / 1000.0 AS VARCHAR(20)) + '</td><td>' +
    CAST(signal_wait_time_ms / 1000.0 AS VARCHAR(20)) + '</td><td>' +
    CAST((wait_time_ms - signal_wait_time_ms) / 1000.0 AS VARCHAR(20)) + '</td><td>' +
    CAST(waiting_tasks_count AS VARCHAR(20)) + '</td><td>' +
    CASE
        WHEN wait_type LIKE 'PAGEIOLATCH%' THEN 'Data File I/O'
        WHEN wait_type = 'WRITELOG' THEN 'Log File I/O'
        WHEN wait_type LIKE 'LCK_M%' THEN 'locking'
        WHEN wait_type = 'SOS_SCHEDULER_YIELD' THEN 'CPU Pressure'
        WHEN wait_type = 'CXPACKET' THEN 'Parallelism'
        WHEN wait_type = 'ASYNC_NETWORK_IO' THEN 'Network / Application'
        ELSE 'Other'
    END + '</td></tr>'
FROM sys.dm_os_wait_stats
WHERE wait_type NOT LIKE 'SLEEP%'
```

```
ORDER BY wait_time_ms DESC;
SET @html = @html + '</table>';
SELECT @html AS WaitStatsHTMLReport;
```

Sample Output:



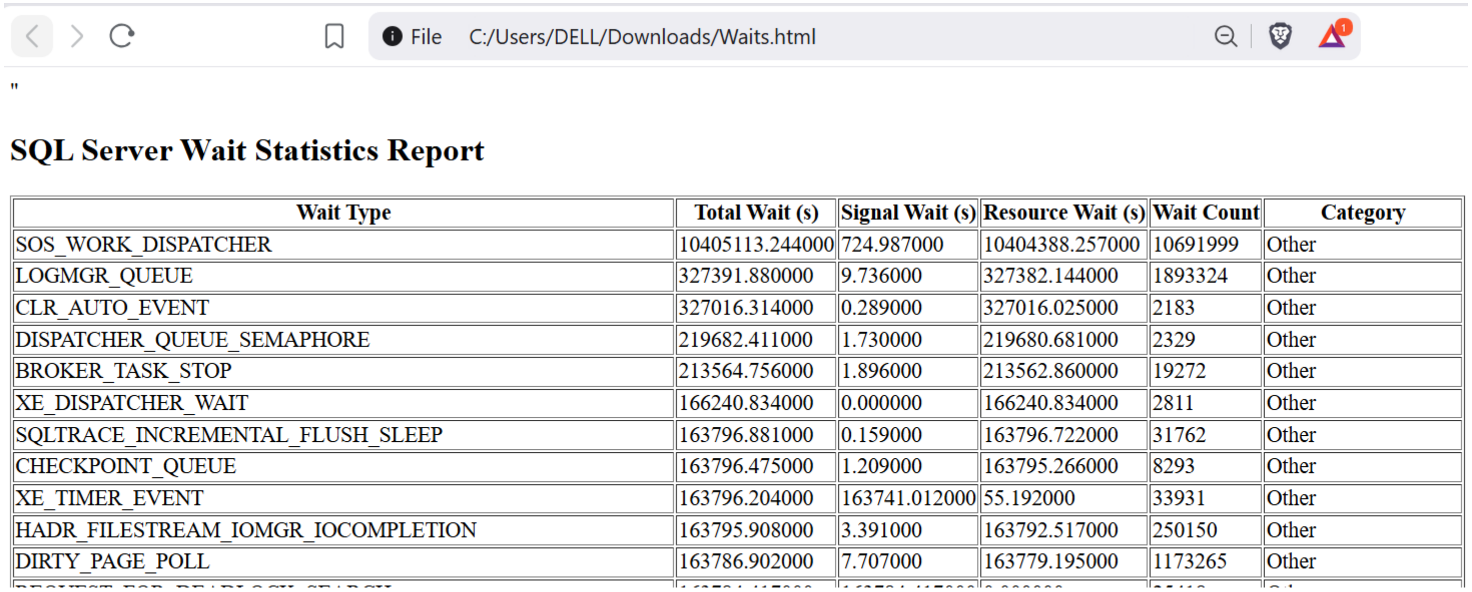
```
DECLARE @html NVARCHAR(MAX);

SET @html = N'
```

100 % No issues found Ln: 34 Ch: 32

Results Messages

```
   WaitStatsHTMLReport
1  <h2>SQL Server Wait Statistics Report</h2>  <table border="1" cellpadding="5" cellspacing="0">  <tr>  <th>Wait Type</th>  <th>Total Wait (s)</th>  <th>Signal Wait (s)</th>  <th>Resource W...
```

Save the output as "Waits.HTML" in your required path. HTML output file looks as below.



File C:/Users/DELL/Downloads/Waits.html

"

SQL Server Wait Statistics Report

Wait Type	Total Wait (s)	Signal Wait (s)	Resource Wait (s)	Wait Count	Category
SOS_WORK_DISPATCHER	10405113.244000	724.987000	10404388.257000	10691999	Other
LOGMGR_QUEUE	327391.880000	9.736000	327382.144000	1893324	Other
CLR_AUTO_EVENT	327016.314000	0.289000	327016.025000	2183	Other
DISPATCHER_QUEUE_SEMAPHORE	219682.411000	1.730000	219680.681000	2329	Other
BROKER_TASK_STOP	213564.756000	1.896000	213562.860000	19272	Other
XE_DISPATCHER_WAIT	166240.834000	0.000000	166240.834000	2811	Other
SQLTRACE_INCREMENTAL_FLUSH_SLEEP	163796.881000	0.159000	163796.722000	31762	Other
CHECKPOINT_QUEUE	163796.475000	1.209000	163795.266000	8293	Other
XE_TIMER_EVENT	163796.204000	163741.012000	55.192000	33931	Other
HADR_FILESTREAM_IOMGR_IOCOMPLETION	163795.908000	3.391000	163792.517000	250150	Other
DIRTY_PAGE_POLL	163786.902000	7.707000	163779.195000	1173265	Other

4. Interview-Ready Explanation of Top Wait Types

PAGEIOLATCH

- **Meaning:** Waiting for slow reads from disk.
- **Fix:** More RAM, faster disks, better indexes.

WRITELOG

- **Meaning:** Log file is too slow.
- **Fix:** Fast SSD for log file, presize the log, optimize large writes.

LCK_M_

- **Meaning:** Blocking due to locks.
- **Fix:** Fix slow queries, add indexes, use RCSI, break large batches.

SOS_SCHEDULER_YIELD

- **Meaning:** CPU pressure.
- **Fix:** Tune CPU-heavy queries, reduce parallelism, add CPU.

CXPACKET

- **Meaning:** Parallelism imbalance.

- **Fix:** Adjust MAXDOP + CTFP, fix data skew, improve indexing.

ASYNC_NETWORK_IO

- **Meaning:** App is reading results slowly.
- **Fix:** Fix RBAR loops, reduce result size, improve network.

RESOURCE_SEMAPHORE

- **Meaning:** Memory grant pressure.
- **Fix:** Update stats, fix cardinality, add RAM.

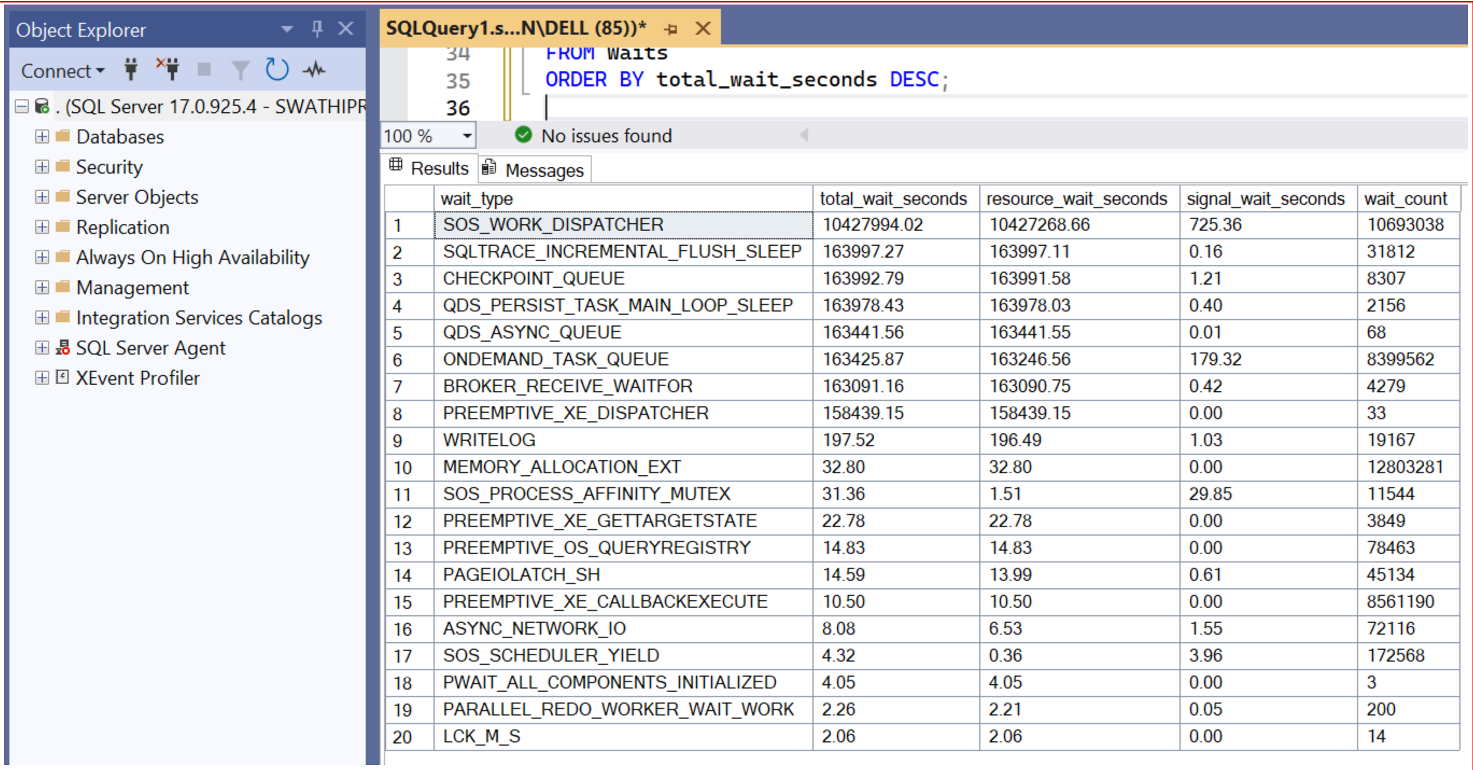
THREADPOOL

- **Meaning:** Worker thread shortage.
- **Fix:** Fix blocking, reduce parallelism, tune connection spikes.

----
1. MASTER WAIT STATISTICS ANALYZER (T-SQL)

```
/*================================
  SQL Server Critical Wait Analysis
================================*/

-- OPTIONAL: Clear to start fresh baseline
-- DBCC SQLPERF('sys.dm_os_wait_stats', CLEAR);

WITH Waits AS
(
   SELECT
      wait_type,
      wait_time_ms / 1000.0 AS wait_s,
      (wait_time_ms - signal_wait_time_ms) / 1000.0 AS resource_s,
      signal_wait_time_ms / 1000.0 AS signal_s,
      waiting_tasks_count AS waits
   FROM sys.dm_os_wait_stats
   WHERE wait_type NOT IN
   (
      'CLR_SEMAPHORE','LAZYWRITER_SLEEP','RESOURCE_QUEUE','SLEEP_TASK',
      'SLEEP_SYSTEMTASK','SQLTRACE_BUFFER_FLUSH','WAITFOR','LOGMGR_QUEUE',
      'REQUEST_FOR_DEADLOCK_SEARCH','XE_TIMER_EVENT','XE_DISPATCHER_WAIT',
      'BROKER_TO_FLUSH','BROKER_TASK_STOP','CLR_MANUAL_EVENT','CLR_AUTO_EVENT',
      'DISPATCHER_QUEUE_SEMAPHORE','FT_IFTS_SCHEDULER_IDLE_WAIT',
      'BROKER_EVENTHANDLER','BAD_PAGE_PROCESS','DIRTY_PAGE_POLL',
      'HADR_FILESTREAM_IOMGR_IOCOMPLETION','SP_SERVER_DIAGNOSTICS_SLEEP'
   )
)
SELECT TOP 20
   wait_type,
   CAST(wait_s AS DECIMAL(12,2)) AS total_wait_seconds,
   CAST(resource_s AS DECIMAL(12,2)) AS resource_wait_seconds,
   CAST(signal_s AS DECIMAL(12,2)) AS signal_wait_seconds,
   waits AS wait_count
FROM Waits
ORDER BY total_wait_seconds DESC;
```

Sample output:



```
FROM Waits
ORDER BY total_wait_seconds DESC;
```

No issues found

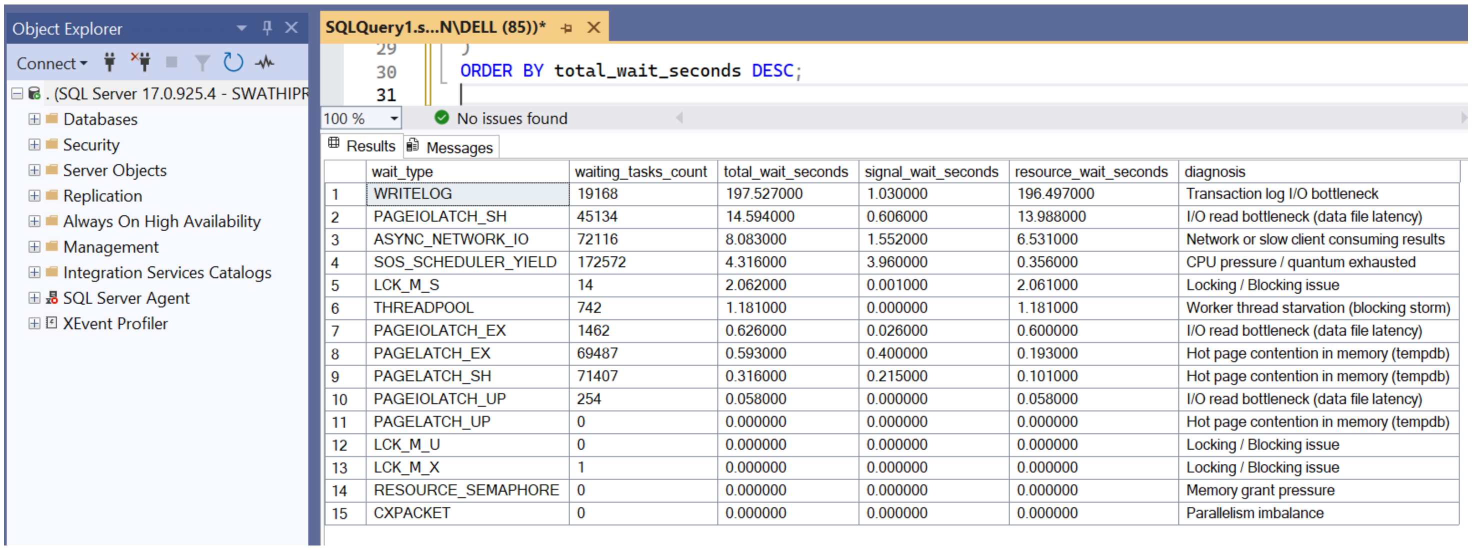
	wait_type	total_wait_seconds	resource_wait_seconds	signal_wait_seconds	wait_count
1	SOS_WORK_DISPATCHER	10427994.02	10427268.66	725.36	10693038
2	SQLTRACE_INCREMENTAL_FLUSH_SLEEP	163997.27	163997.11	0.16	31812
3	CHECKPOINT_QUEUE	163992.79	163991.58	1.21	8307
4	QDS_PERSIST_TASK_MAIN_LOOP_SLEEP	163978.43	163978.03	0.40	2156
5	QDS_ASYNC_QUEUE	163441.56	163441.55	0.01	68
6	ONDEMAND_TASK_QUEUE	163425.87	163246.56	179.32	8399562
7	BROKER_RECEIVE_WAITFOR	163091.16	163090.75	0.42	4279
8	PREEMPTIVE_XE_DISPATCHER	158439.15	158439.15	0.00	33
9	WRITELOG	197.52	196.49	1.03	19167
10	MEMORY_ALLOCATION_EXT	32.80	32.80	0.00	12803281
11	SOS_PROCESS_AFFINITY_MUTEX	31.36	1.51	29.85	11544
12	PREEMPTIVE_XE_GETTARGETSTATE	22.78	22.78	0.00	3849
13	PREEMPTIVE_OS_QUERYREGISTRY	14.83	14.83	0.00	78463
14	PAGEIOLATCH_SH	14.59	13.99	0.61	45134
15	PREEMPTIVE_XE_CALLBACKEXECUTE	10.50	10.50	0.00	8561190
16	ASYNC_NETWORK_IO	8.08	6.53	1.55	72116
17	SOS_SCHEDULER_YIELD	4.32	0.36	3.96	172568
18	PWAIT_ALL_COMPONENTS_INITIALIZED	4.05	4.05	0.00	3
19	PARALLEL_REDO_WORKER_WAIT_WORK	2.26	2.21	0.05	200
20	LCK_M_S	2.06	2.06	0.00	14

----

2. Detailed Diagnosis Script

```
/*==============================================
 Detailed Diagnosis for Critical Wait Types
============================================*/
SELECT
  wait_type,
  waiting_tasks_count,
  wait_time_ms / 1000.0 AS total_wait_seconds,
  signal_wait_time_ms / 1000.0 AS signal_wait_seconds,
  (wait_time_ms - signal_wait_time_ms) / 1000.0 AS resource_wait_seconds,
  CASE
    WHEN wait_type LIKE 'PAGEIOLATCH%' THEN 'I/O read bottleneck (data file latency)'
    WHEN wait_type = 'WRITELOG' THEN 'Transaction log I/O bottleneck'
    WHEN wait_type LIKE 'LCK_M%' THEN 'Locking / Blocking issue'
    WHEN wait_type = 'SOS_SCHEDULER_YIELD' THEN 'CPU pressure / quantum exhausted'
    WHEN wait_type = 'CXPACKET' THEN 'Parallelism imbalance'
    WHEN wait_type = 'ASYNC_NETWORK_IO' THEN 'Network or slow client consuming results'
    WHEN wait_type LIKE 'PAGELATCH%' THEN 'Hot page contention in memory (tempdb)'
    WHEN wait_type = 'RESOURCE_SEMAPHORE' THEN 'Memory grant pressure'
    WHEN wait_type = 'THREADPOOL' THEN 'Worker thread starvation (blocking storm)'
    ELSE 'Other'
  END AS diagnosis
FROM sys.dm_os_wait_stats
WHERE wait_type IN (
  'PAGEIOLATCH_SH','PAGEIOLATCH_EX','PAGEIOLATCH_UP',
  'WRITELOG', 'LCK_M_S','LCK_M_X','LCK_M_U',
  'SOS_SCHEDULER_YIELD','CXPACKET','ASYNC_NETWORK_IO',
  'PAGELATCH_SH','PAGELATCH_EX','PAGELATCH_UP',
    'RESOURCE_SEMAPHORE','THREADPOOL'
)
ORDER BY total_wait_seconds DESC;
```

Sample Output:



```
)
ORDER BY total_wait_seconds DESC;
```

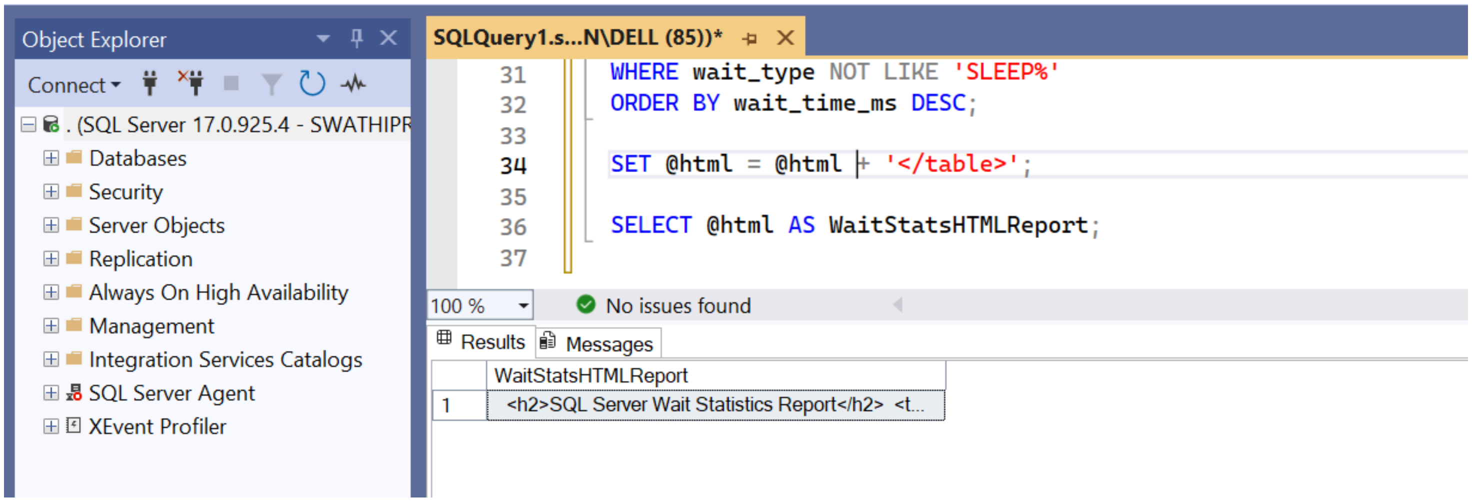
	wait_type	waiting_tasks_count	total_wait_seconds	signal_wait_seconds	resource_wait_seconds	diagnosis
1	WRITELOG	19168	197.527000	1.030000	196.497000	Transaction log I/O bottleneck
2	PAGEIOLATCH_SH	45134	14.594000	0.606000	13.988000	I/O read bottleneck (data file latency)
3	ASYNC_NETWORK_IO	72116	8.083000	1.552000	6.531000	Network or slow client consuming results
4	SOS_SCHEDULER_YIELD	172572	4.316000	3.960000	0.356000	CPU pressure / quantum exhausted
5	LCK_M_S	14	2.062000	0.001000	2.061000	Locking / Blocking issue
6	THREADPOOL	742	1.181000	0.000000	1.181000	Worker thread starvation (blocking storm)
7	PAGEIOLATCH_EX	1462	0.626000	0.026000	0.600000	I/O read bottleneck (data file latency)
8	PAGELATCH_EX	69487	0.593000	0.400000	0.193000	Hot page contention in memory (tempdb)
9	PAGELATCH_SH	71407	0.316000	0.215000	0.101000	Hot page contention in memory (tempdb)
10	PAGEIOLATCH_UP	254	0.058000	0.000000	0.058000	I/O read bottleneck (data file latency)
11	PAGELATCH_UP	0	0.000000	0.000000	0.000000	Hot page contention in memory (tempdb)
12	LCK_M_U	0	0.000000	0.000000	0.000000	Locking / Blocking issue
13	LCK_M_X	1	0.000000	0.000000	0.000000	Locking / Blocking issue
14	RESOURCE_SEMAPHORE	0	0.000000	0.000000	0.000000	Memory grant pressure
15	CXPACKET	0	0.000000	0.000000	0.000000	Parallelism imbalance

----

3. HTML REPORT SCRIPT (Email-Friendly)

```
DECLARE @html NVARCHAR(MAX);

SET @html = N'
<h2>SQL Server Wait Statistics Report</h2>
<table border="1" cellpadding="5" cellspacing="0">
<tr>
<th>Wait Type</th>
<th>Total Wait (s)</th>
<th>Signal Wait (s)</th>
<th>Resource Wait (s)</th>
<th>Wait Count</th>
<th>Category</th>
</tr>';

SELECT @html = @html +
    '<tr><td>' + wait_type + '</td><td>' +
    CAST(wait_time_ms / 1000.0 AS VARCHAR(20)) + '</td><td>' +
    CAST(signal_wait_time_ms / 1000.0 AS VARCHAR(20)) + '</td><td>' +
    CAST((wait_time_ms - signal_wait_time_ms) / 1000.0 AS VARCHAR(20)) + '</td><td>' +
    CAST(waiting_tasks_count AS VARCHAR(20)) + '</td><td>' +
    CASE
        WHEN wait_type LIKE 'PAGEIOLATCH%' THEN 'Data File I/O'
        WHEN wait_type = 'WRITELOG' THEN 'Log File I/O'
        WHEN wait_type LIKE 'LCK_M%' THEN 'Blocking'
        WHEN wait_type = 'SOS_SCHEDULER_YIELD' THEN 'CPU Pressure'
        WHEN wait_type = 'CXPACKET' THEN 'Parallelism'
        WHEN wait_type = 'ASYNC_NETWORK_IO' THEN 'Network/App'
        ELSE 'Other'
    END + '</td></tr>'
FROM sys.dm_os_wait_stats
WHERE wait_type NOT LIKE 'SLEEP%'
ORDER BY wait_time_ms DESC;

SET @html = @html + '</table>';

SELECT @html AS WaitStatsHTMLReport;
```

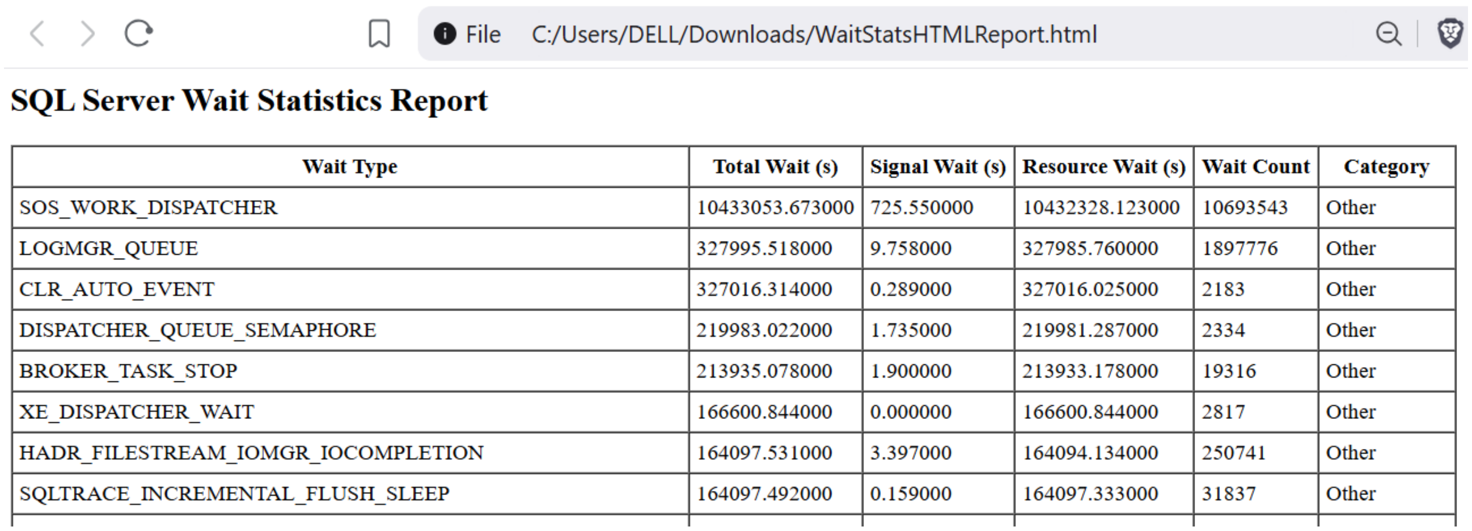
Sample output:



```
WHERE wait_type NOT LIKE 'SLEEP%'
ORDER BY wait_time_ms DESC;

SET @html = @html + '</table>';

SELECT @html AS WaitStatsHTMLReport;
```

	WaitStatsHTMLReport
1	<h2>SQL Server Wait Statistics Report</h2> <t...

Save the output as “WaitStatsHTMLReport.HTML” in your required path. HTML output file looks as below.



SQL Server Wait Statistics Report

Wait Type	Total Wait (s)	Signal Wait (s)	Resource Wait (s)	Wait Count	Category
SOS_WORK_DISPATCHER	10433053.673000	725.550000	10432328.123000	10693543	Other
LOGMGR_QUEUE	327995.518000	9.758000	327985.760000	1897776	Other
CLR_AUTO_EVENT	327016.314000	0.289000	327016.025000	2183	Other
DISPATCHER_QUEUE_SEMAPHORE	219983.022000	1.735000	219981.287000	2334	Other
BROKER_TASK_STOP	213935.078000	1.900000	213933.178000	19316	Other
XE_DISPATCHER_WAIT	166600.844000	0.000000	166600.844000	2817	Other
HADR_FILESTREAM_IOMGR_IOCOMPLETION	164097.531000	3.397000	164094.134000	250741	Other
SQLTRACE_INCREMENTAL_FLUSH_SLEEP	164097.492000	0.159000	164097.333000	31837	Other

----

4. Interview-Ready Wait Type Explanations

PAGEIOLATCH

Slow I/O reads on data files.

✓ Fix: Faster storage, more RAM, better indexing.

WRITELOG

Log file throughput slow.

✓ Fix: Put log on fastest disk, presize, batch commits.

LCK_M_

Blocking due to locks.

✓ Fix: Indexing, RCSI, break large transactions.

SOS_SCHEDULER_YIELD

CPU pressure / worker exhausted quantum.

✓ Fix: Tune CPU-heavy queries, MAXDOP tuning.

CXPACKET

Parallelism threads not finishing equally.

✓ Fix: MAXDOP, CTFP, fix skew, indexing.

ASYNC_NETWORK_IO

Application consuming results slowly.

✓ Fix: Optimize client loops, reduce result-set size.

RESOURCE_SEMAPHORE

Memory grant pressure.

✓ Fix: Fix stats, grants, add RAM.

THREADPOOL

Worker thread starvation.

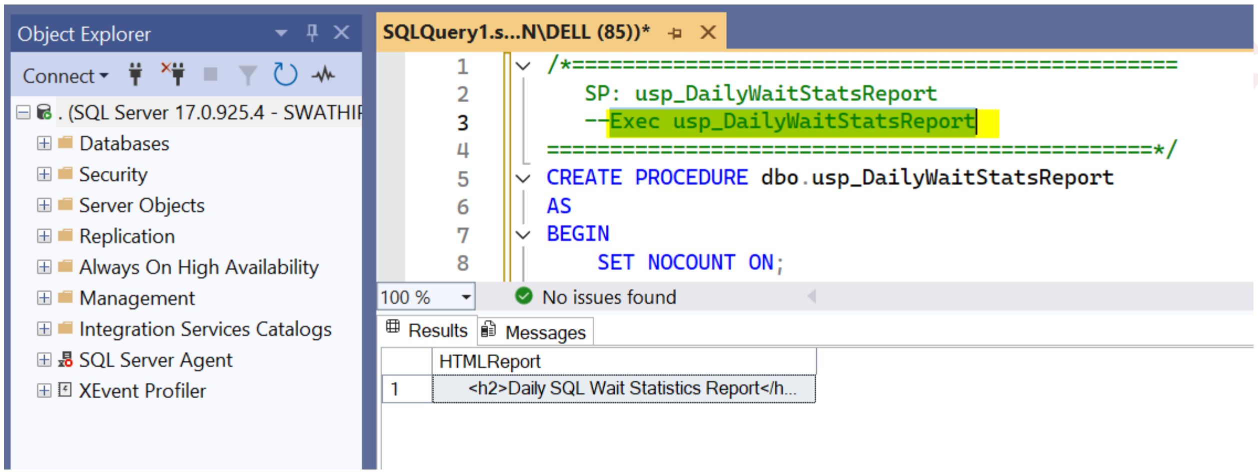
✓ Fix: Reduce parallelism, eliminate blocking storms.

----

5. Monitoring Stored Procedure (Daily Job)

```
/*================================================
  SP: usp_DailyWaitStatsReport
================================================*/
        CREATE PROCEDURE dbo.usp_DailyWaitStatsReport
        AS
        BEGIN
          SET NOCOUNT ON;
          DECLARE @html NVARCHAR(MAX);
          SET @html = N'
          <h2>Daily SQL Wait Statistics Report</h2>
          <table border="1" cellpadding="4" cellspacing="0">
          <tr>
          <th>Wait Type</th>
          <th>Total Wait (s)</th>
          <th>Signal Wait (s)</th>
          <th>Resource Wait (s)</th>
          <th>Wait Count</th>
          <th>Category</th>
          </tr>';
        SELECT @html = @html +
          '<tr><td>' + wait_type + '</td><td>' +
          CAST(wait_time_ms / 1000.0 AS VARCHAR(20)) + '</td><td>' +
          CAST(signal_wait_time_ms / 1000.0 AS VARCHAR(20)) + '</td><td>' +
          CAST((wait_time_ms - signal_wait_time_ms) / 1000.0 AS VARCHAR(20)) + '</td><td>' +
          CAST(waiting_tasks_count AS VARCHAR(20)) + '</td><td>' +
          CASE
            WHEN wait_type LIKE 'PAGEIOLATCH%' THEN 'Data File I/O'
            WHEN wait_type = 'WRITELOG' THEN 'Log I/O'
            WHEN wait_type LIKE 'LCK_M%' THEN 'Blocking'
            WHEN wait_type = 'SOS_SCHEDULER_YIELD' THEN 'CPU Pressure'
            WHEN wait_type = 'CXPACKET' THEN 'Parallelism'
            WHEN wait_type = 'ASYNC_NETWORK_IO' THEN 'Network'
            ELSE 'Other'
          END + '</td></tr>'
        FROM sys.dm_os_wait_stats
        WHERE wait_type NOT LIKE 'SLEEP%'
        ORDER BY wait_time_ms DESC;
        SET @html = @html + '</table>';
        SELECT @html AS HTMLReport;
      END;
```

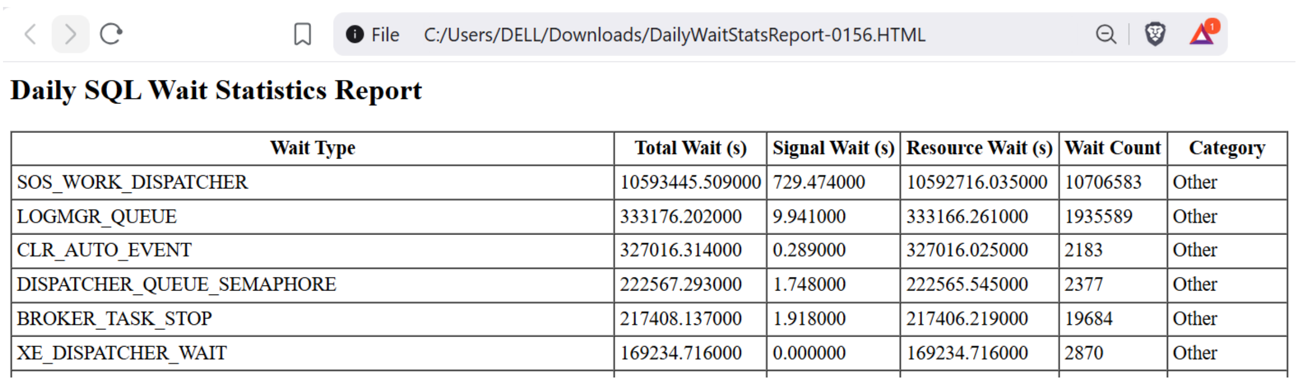
Sample output:



```
/*==================================================
   SP: usp_DailyWaitStatsReport
   --Exec usp_DailyWaitStatsReport
==================================================*/
CREATE PROCEDURE dbo.usp_DailyWaitStatsReport
AS
BEGIN
    SET NOCOUNT ON;
```

	HTMLReport
1	<h2>Daily SQL Wait Statistics Report</h...

Save the output as "Daily SQL Wait Statistics.HTML" in your required path. HTML output file looks as below.


File   C:/Users/DELL/Downloads/DailyWaitStatsReport-0156.HTML

Daily SQL Wait Statistics Report

Wait Type	Total Wait (s)	Signal Wait (s)	Resource Wait (s)	Wait Count	Category
SOS_WORK_DISPATCHER	10593445.509000	729.474000	10592716.035000	10706583	Other
LOGMGR_QUEUE	333176.202000	9.941000	333166.261000	1935589	Other
CLR_AUTO_EVENT	327016.314000	0.289000	327016.025000	2183	Other
DISPATCHER_QUEUE_SEMAPHORE	222567.293000	1.748000	222565.545000	2377	Other
BROKER_TASK_STOP	217408.137000	1.918000	217406.219000	19684	Other
XE_DISPATCHER_WAIT	169234.716000	0.000000	169234.716000	2870	Other

----
6. PowerShell Script — Email the HTML Wait Report Daily

```
#============================================
# PowerShell Wait Stats Reporter
#============================================

$server = "YourSQLServer"
$database = "master"
$sp = "dbo.usp_DailyWaitStatsReport"
$to = "recipient@company.com"
$from = "sqlalerts@company.com"
$smtp = "smtp.server.com"
$subject = "Daily SQL Server Wait Statistics Report"

# Run SP
$query = "EXEC $sp;"
$html = Invoke-Sqlcmd -ServerInstance $server -Database $database -Query $query

Send-MailMessage -To $to -From $from -Subject $subject -BodyAsHtml $html.HTMLReport -SmtpServer $smtp
```

----
7. One-Page Wait Statistics DBA Cheat Sheet

```
==========================================
SQL SERVER WAIT STATISTICS – QUICK CHEAT SHEET
==========================================

1. PAGEIOLATCH_  – Data file I/O read latency
   Fix: Faster storage, add RAM, missing indexes, reduce scans.

2. WRITELOG – Log file write bottleneck
   Fix: Fast SSD, pre-size log, reduce autogrowth, batch writes.

3. LCK_M_ – Locking / Blocking
   Fix: Indexing, reduce long transactions, use RCSI, optimize queries.

4. SOS_SCHEDULER_YIELD – CPU pressure
   Fix: Tune CPU-heavy queries, MAXDOP tuning, add CPU.

5. CXPACKET – Parallelism imbalance
   Fix: MAXDOP (4 or 8), increase cost threshold for parallelism, fix skew.

6. ASYNC_NETWORK_IO – Client/app consuming results slowly
   Fix: Optimize client code, reduce result set, check network latency.

7. PAGELATCH_ – TempDB / in-memory hot latch contention
   Fix: Add tempdb files, align sizes, optimize temp usage.

8. RESOURCE_SEMAPHORE – Memory grant pressure
```

Fix: Update stats, fix large memory grant queries, add RAM.

9. THREADPOOL – Worker thread starvation
Fix: Resolve blocking, reduce parallelism, fix connection storms.

10. IO_COMPLETION – General I/O slowness
Fix: Tune I/O subsystem, reduce heavy ETL during peak.