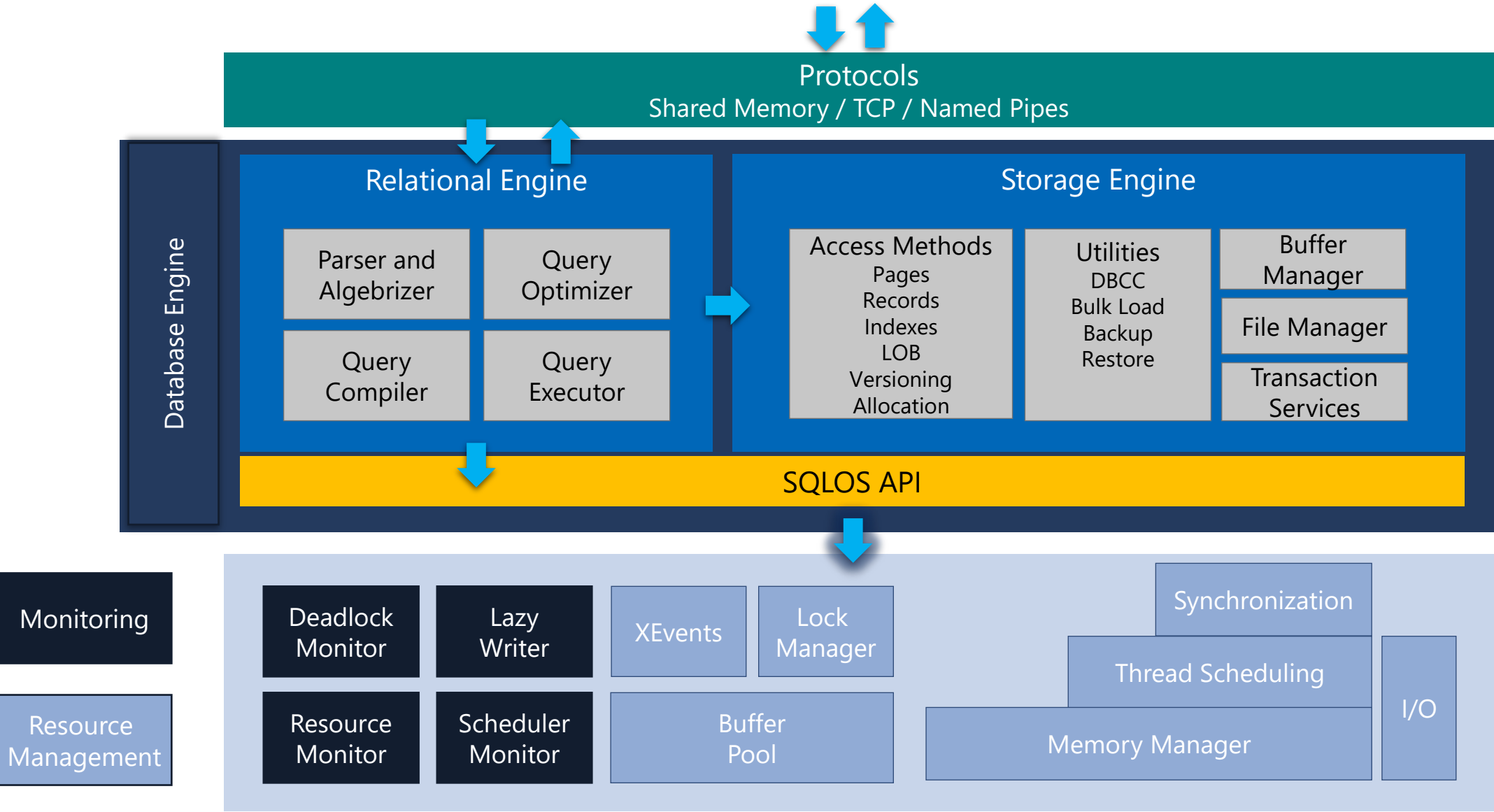# Learning Units covered in this Module

- Lesson 1: Introduction to SQL Operating System

- Lesson 2: SQL Server Task Scheduling

- Lesson 3: SQL Server Waits and Queues

# Lesson 1: Introduction to SQL Operating System

# Inside the Database Engine

# SQL Server Operating System (SQLOS)

Application layer between Microsoft SQL Server components and the Windows Operating System.

Centralizes resource allocation to provide more efficient management and accounting.

The SQLOS is used by the SQL Server relational database engine for system-level services.

Abstracts the concepts of resource management from components, providing:

- **Scheduling and synchronization support**
- **Memory management and caching**
- **Resource governance**
- **Diagnostics and debug infrastructure**
- **Scalability and performance optimization**

# Two Main Functions of SQLOS

| Management | Monitoring |
|---|---|
| • Memory Manager<br>• Process Scheduler<br>• Synchronization<br>• I/O<br>• Support for Non-Uniform Memory Access (NUMA) and Resource Governor | • Resource Monitor<br>• Deadlock Monitor<br>• Scheduler Monitor<br>• Lazy Writer (Buffer Pool management)<br>• Dynamic Management Views (DMVs)<br>• Extended Events<br>• Dedicated Administrator Connection (DAC) |

# Dynamic Management Views and Functions

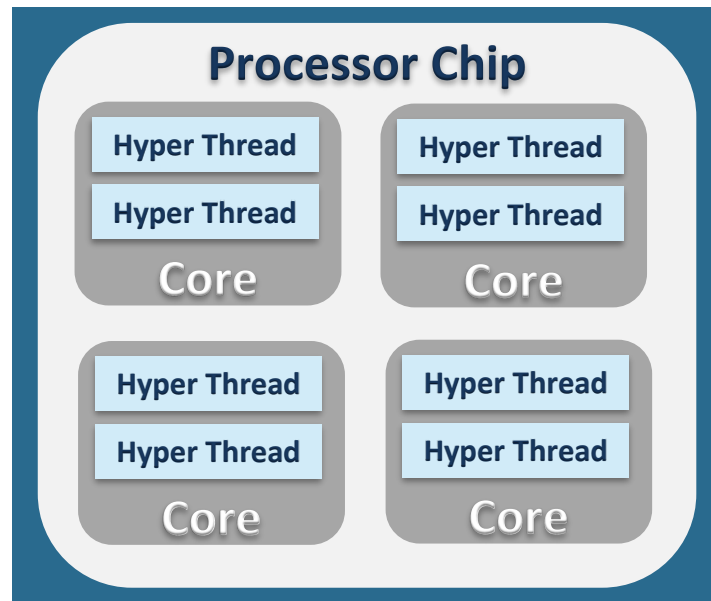| Category | Description |
|---|---|
| sys.dm_exec_% | Execution and connection information |
| sys.dm_os_% | Operating system related information |
| sys.dm_tran_% | Transaction management information |
| sys.dm_io_% | I/O related information |
| sys.dm_db_% | Database information |

# Using Dynamic Management Objects (DMOs)

- Must reference using the sys schema
- Two basic types:
  - Real-time state information
  - Historical information

```sql
SELECT cpu_count, hyperthread_ratio,
    scheduler_count, scheduler_total_count,
    affinity_type, affinity_type_desc,
    softnuma_configuration, softnuma_configuration_desc,
    socket_count, cores_per_socket, numa_node_count,
    sql_memory_model, sql_memory_model_desc
FROM sys.dm_os_sys_info
```
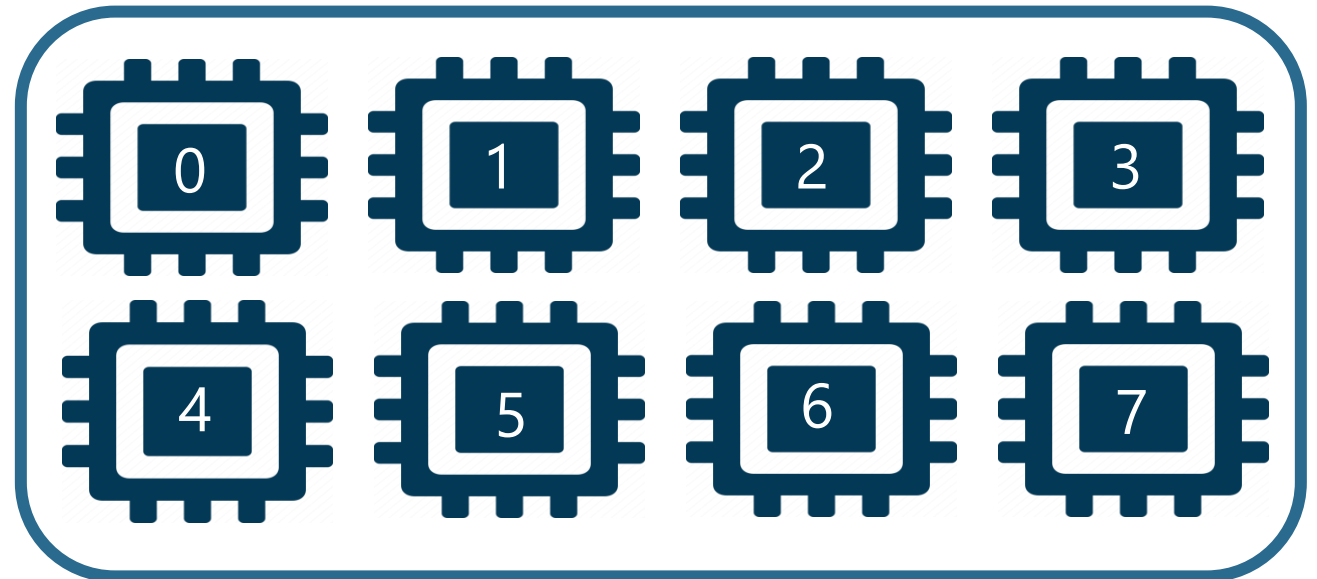
# CPU Architecture

## Physical Hardware

**Processor Chip**

Hyper Thread
Hyper Thread
**Core**

Hyper Thread
Hyper Thread
**Core**

Hyper Thread
Hyper Thread
**Core**

Hyper Thread
Hyper Thread
**Core**

**Socket**

## Logical Processors as seen by the OS

0  1  2  3

4  5  6  7

# Demonstration

Dynamic Management Views

# Questions?

# Lesson 2: SQL Server Task Scheduling

# Microsoft SQL Server Scheduling Terminology

## Batch

- A statement or set of statements submitted to SQL Server by the user (a query), also referred to as a request
- Monitor with sys.dm_exec_requests

## Task

- A batch will have one or more tasks (aligns with statements)
- Monitor with sys.dm_os_tasks

## Worker Thread

- Each task will be assigned to a single worker thread for the life of the task
- Monitor with sys.dm_os_workers

# Hierarchy of Common Terms

```sql
SELECT *
FROM sys.dm_exec_connections;
-- relevant data:
-- session_id --> spid
-- most_recent_sql_handle --> last query
-- net_transport, protocol_type --> connectivity
```

Connection → Session → Request (Batch) → Task → Worker

# Hierarchy of Common Terms

```sql
SELECT *
FROM sys.dm_exec_sessions;
-- relevant data:
-- session_id --> spid
-- host_name, program_name --> client identity
-- login_name, nt_user_name --> login identity
-- status --> activity
-- database_id --> database being accessed
-- open_transaction_count --> blocking identification
```

| Connection | Session | Request (Batch) | Task | Worker |

# Hierarchy of Common Terms

```sql
SELECT  *
FROM sys.dm_exec_requests;
-- relevant data:
-- session_id --> spid
-- status --> background, running, runnable, suspended
-- sql_handle, offset --> query text
-- database_id --> database being accessed
-- wait_type, wait_time --> blocking information
-- open_transaction_count --> blocking others
-- cpu_time, total_elapsed_time, reads, writes --> telemetry
```

| Connection | Session | Request (Batch) | Task | Worker |

# Hierarchy of Common Terms

```sql
SELECT *
FROM sys.dm_os_tasks;
-- relevant data:
-- task_state --> running, suspended
-- pending_io_* --> I/O activity
-- scheduler_id -->processor info
-- session_id --> spid
```

| Connection | Session | Request (Batch) | Task | Worker |

# Hierarchy of Common Terms

```sql
SELECT *
FROM sys.dm_os_workers;
-- relevant data:
-- worker_address --> memory address of the worker
-- wait_start_ms_ticks --> Point in time worker Suspended.
-- wait_resumed_ms_ticks --> Worker in Runnable state.
-- state -- > Running, Runnable, Susspended
```

| Connection | Session | Request (Batch) | Task | Worker |

# Scheduling Types

## Non-Preemptive (Cooperative)

- SQL Server manages CPU scheduling for most activity (instead of the operating system).
- SQL Server decides when a thread should wait or get switched out (known as yielding).
- SQL Server developers also programmed some predetermined voluntary yields to avoid starvation of other threads

## Preemptive

- Preemption is the act of an operating system temporarily interrupting an executing task.
- Higher priority tasks can preempt lower priority tasks.
- Preemptive mode used in SQL Server for external code calls, CLR with an UNSAFE assemblies, extended stored procedures

# SQL Server Task Scheduling

One SQLOS Scheduler per core/logical processor

Handles scheduling tasks, I/O, and synchronization of resources

Work requests are balanced across schedulers based on the number of active tasks

Monitor using sys.dm_os_schedulers

Worker Migration – better use of schedulers in SQL Server 2019

# Yielding

In SQL Server, each thread is assigned a quantum (duration 4ms), with SQL Server using a cooperative model to ensure its CPU resources are shared amongst all the threads that are in a runnable state, preventing the 'starving' condition of any individual thread.

By design, a worker owns the scheduler until it yields to another worker on the same scheduler.

When no worker is currently on the Runnable list, the **yielding worker is allowed another quantum** or performs the necessary idle scheduler maintenance.
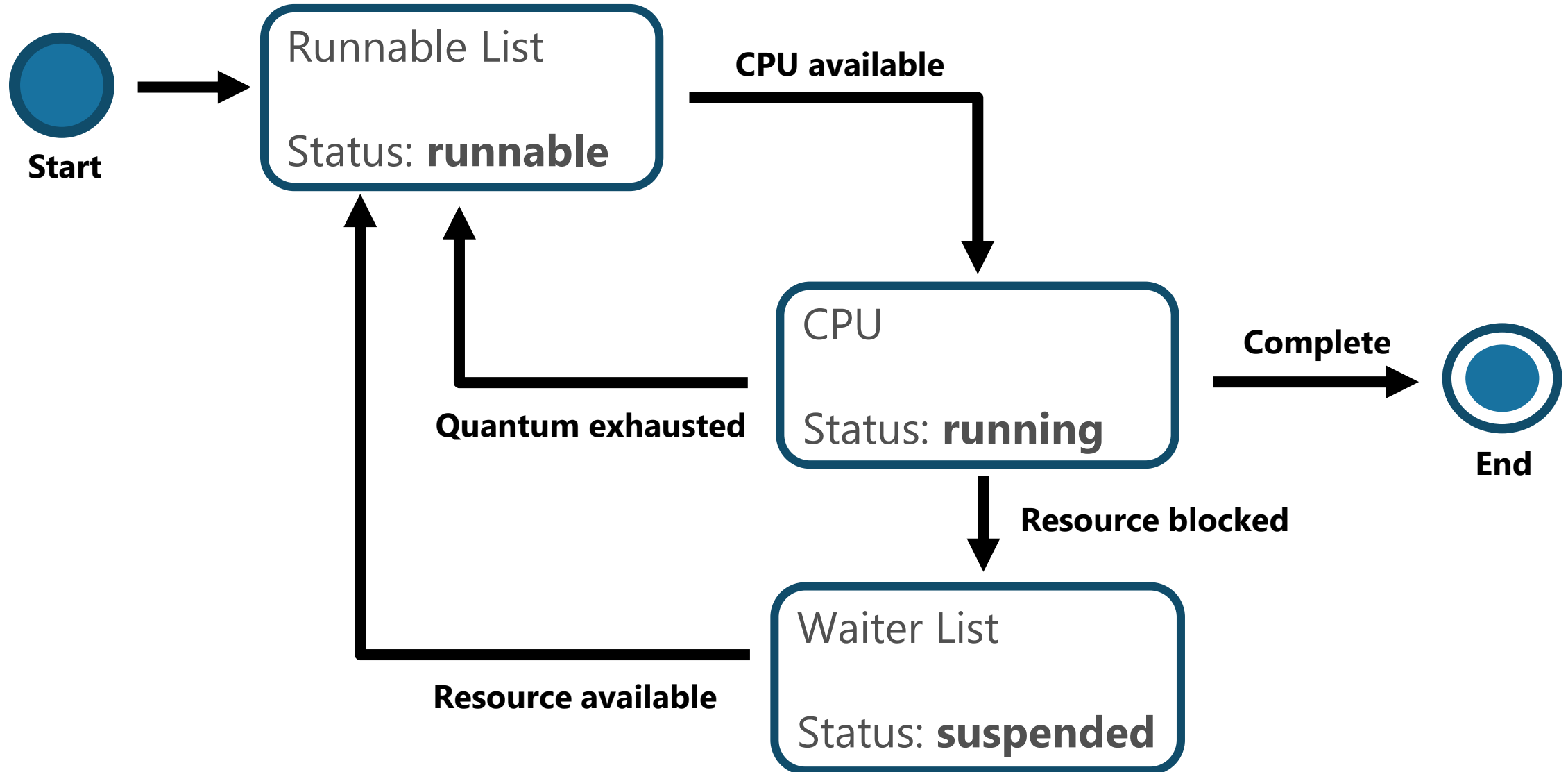
# Thread States and Queues

**Runnable:** The thread is currently in the Runnable Queue waiting to execute. (First In, First Out).

**Running:** One active thread executing on a processor.

**Suspended:** Placed on a Waiter List waiting for a resource other than a processor. (No specific order).

# Yielding

# Demonstration

Runnable Tasks

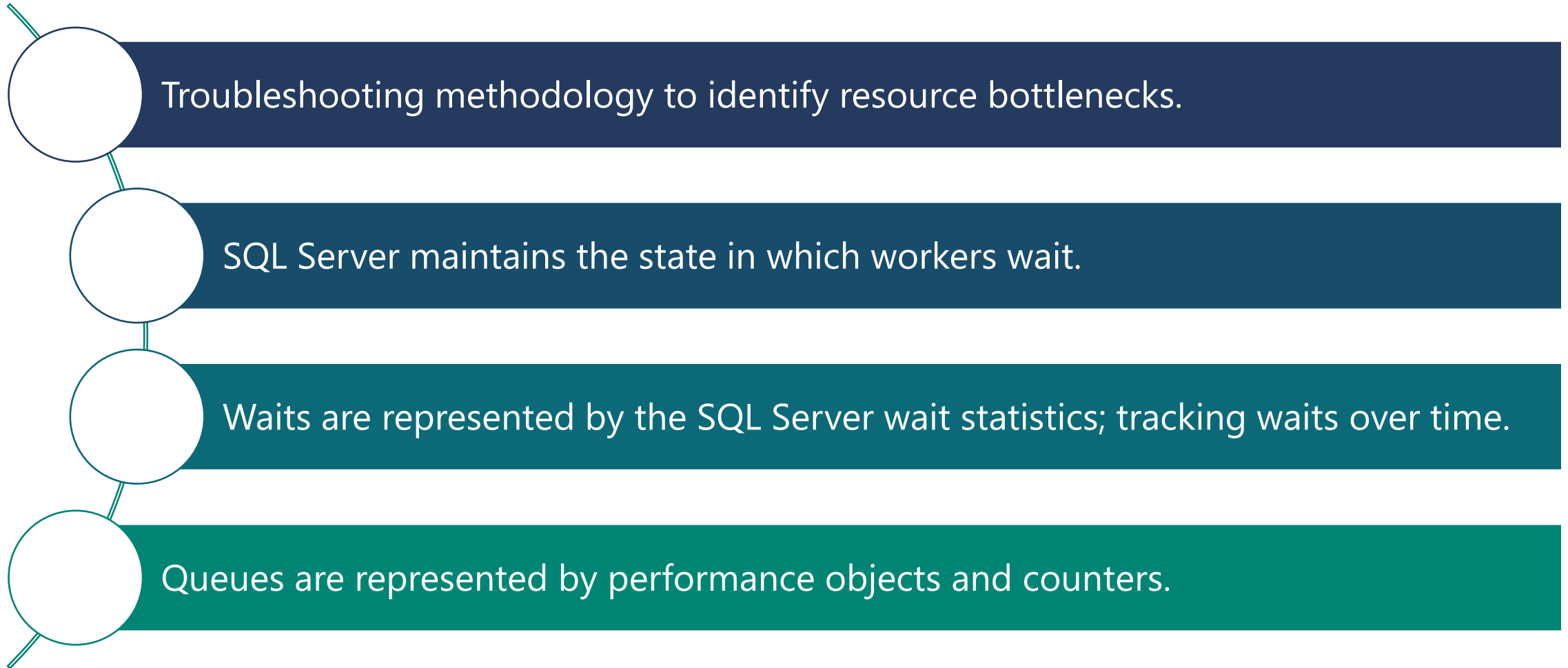# Questions?

# Lesson 3: SQL Server Waits and Queues

# Objectives

After completing this learning, you will be able to:

· Understand common wait types

· Troubleshoot the resource bottleneck

# Waits and Queues

Troubleshooting methodology to identify resource bottlenecks.

SQL Server maintains the state in which workers wait.

Waits are represented by the SQL Server wait statistics; tracking waits over time.

Queues are represented by performance objects and counters.

# Using Waits and Queues

Useful to assist in troubleshooting an active performance issue

Valuable to track the resources SQL Server is regularly waiting on

Useful for workload measurements and benchmarking

Valuable for identifying performance trends

# Task Execution Model

· The full cycle between the several task states, for how many times it needs to cycle, is what we experience as the total query response time.

CPU Time (RUNNING) **+** Signal Wait Time (RUNNABLE) **+** Resource Wait Time (SUSPENDED) **=** Total Query Response Time
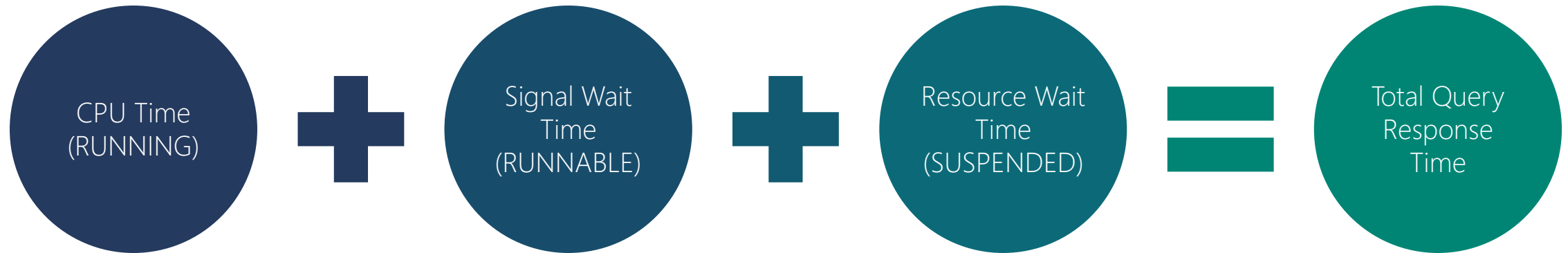
# Task Execution Model

Status: Running

| | |
|---|---|
| session_id 51 | Running |

Wait Queue (Resource Waits)
Status: Suspended

| | |
|---|---|
| session_id 73 | LCK_M_S |
| session_id 59 | NETWORKIO |
| ~~session_id 56~~ | ~~Runnable~~ |
| session_id 55 | RESOURCE_SEMAPHORE |
| session_id 60 | IO_Completion |

FIFO List

Unordered List

Runnable Queue (Signal Waits)
Status: Runnable

| | |
|---|---|
| session_id 51 | Runnable |
| session_id 64 | Runnable |
| session_id 87 | Runnable |
| session_id 52 | Runnable |
| session_id 56 | Runnable |

SPID56 moved to the bottom of the Runnable queue.

# Relevant Dynamic Management Views (DMVs)

## sys.dm_os_wait_stats

- Returns information about all the waits encountered by threads that ran.
- Includes wait type, number of tasks that waited in the specific wait type, total and max wait times, and the amount of signal waits.

## sys.dm_os_waiting_tasks

- Returns information about the wait queue of tasks actively waiting on some resource.

## sys.dm_exec_requests

- Returns information about each request that is in-flight.
- Includes session owning the request and status of the request, which will reflect the status of one or more tasks assigned to the request.

# Waiting Tasks DMV

```sql
SELECT w.session_id, w.wait_duration_ms, w.wait_type,
       w.blocking_session_id, w.resource_description,
       s.program_name, t.text, t.dbid, s.cpu_time, s.memory_usage
  FROM sys.dm_os_waiting_tasks as w
       INNER JOIN sys.dm_exec_sessions as s
          ON w.session_id = s.session_id
       INNER JOIN sys.dm_exec_requests as r
          ON s.session_id = r.session_id
       OUTER APPLY sys.dm_exec_sql_text (r.sql_handle) as t
  WHERE s.is_user_process = 1;
```

| session_id | wait_duration_ms | wait_type | blocking_session_id | resource_description |
|---|---|---|---|---|
| 58 | 8563 | LCK_M_S | 62 | keylock hobtid=72057594047365120 dbid=5 id=lock1... |

# Troubleshooting Wait Types

Aaron Bertrand – Top Wait Types
https://sqlperformance.com/2018/10/sql-performance/top-wait-stats

Paul Randal – SQL Skills Wait Types Library
https://www.sqlskills.com/help/waits/

SQL Server
Performance Tuning
Using Wait Statistics:
A Beginner's Guide

By Jonathan Kehayias
and Erin Stellato

SQLskills  simple talk
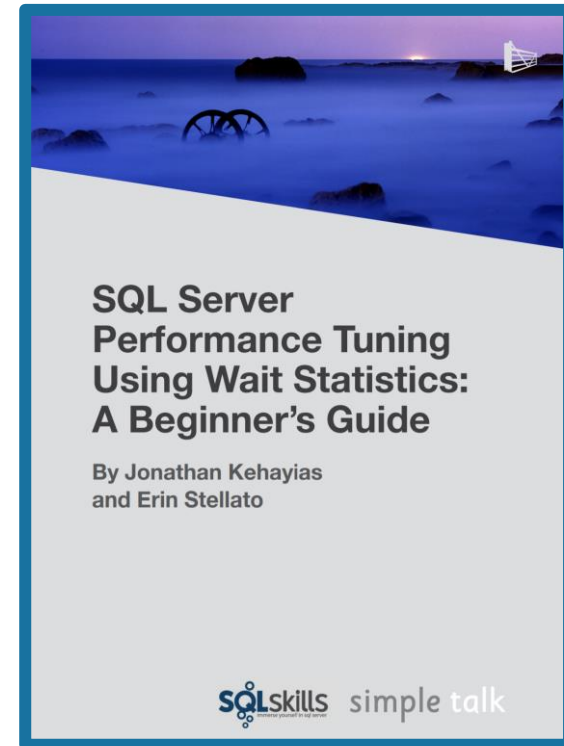
# Notable Waits
CPU related waits

## SOS_SCHEDULER_YIELD

- Normally means a thread has yielded after exhausting the 4ms quantum.
- Might indicate CPU pressure if very high overall percentage of Processor Time. Example: Large amount of Signal Waits (Runnable Queue)

## CXPACKET

- If it's an OLTP system, check for parallelism issues if above 20%
- If combined with a high number of PAGEIOLATCH_xx waits, it could be due to large parallel table scans going on because of incorrect non-clustered indexes, or out-of-date statistics causing a bad query plan

# Notable Waits
PAGE access related waits

## PAGELATCH_xx

- Indicates contention for access to buffers (in-memory copies of pages).
- If PFS, SGAM, and GAM then it is allocation contention (updating allocation metadata), namely in TempDB.

## PAGEIOLATCH_xx

- Indicates I/O problems in data pages.
- Validate disk and memory perfmon counters and the SQL Errorlogs (for error 833 "I/O taking longer than 15 seconds").
- Examine the virtual file stats DMV.

# Notable Waits

I/O related waits

## IO_COMPLETION

- If on TempDB, usually means spilling or high static cursor rate

## ASYNC_IO_COMPLETION

- Usually when not using Instant File Initialization, or waiting on backups

## WRITELOG

- Waiting for a log flush to disk
- Examine the I/O latency for the log file
- Can show up with PREEMPTIVE_OS_WRITEFILEGATHERER in aggressive autogrow scenarios

# Notable Waits

Memory related

## CMEMTHREAD

- Contention in the insertion of entries into the plan cache

## RESOURCE_SEMAPHORE

- Indicates queries are waiting for execution memory (memory grants).
- Look for plans with excessive hashing or large sorts.
- Confirm with the Memory Grants Pending and Memory Grants Outstanding perfmon counters.

# Notable Waits
Other common waits

## THREADPOOL

- Look for high blocking or contention problems with workers.
- This will not show up in sys.dm_exec_requests.

## LCK_xx

- Indicates contention for access to locked resources such as index keys or pages.
- Examine the transactions Isolation Level, maybe using a less concurrent such as SERIALIZABLE.
- Look for queries that do large serialized UPSERTs.

# Performance Monitor Counters

## Important Operating System Counters

- % Processor Time
  - Less than 80% is preferred
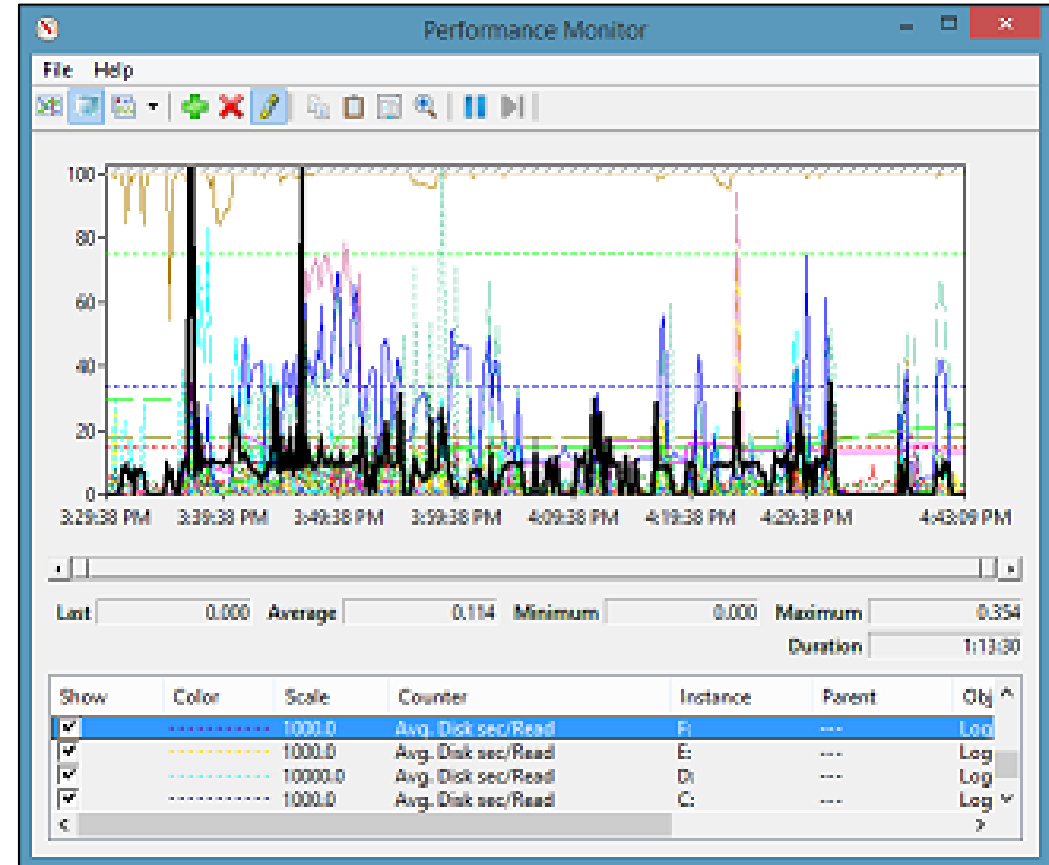
# Performance Monitor Counters

Execution Statistics

## SQL Errors\Errors/sec

- Error types must be investigated and possibly resolved.

## SQL Statistics\Batch Requests/sec

- Batch Requests > 1000 indicates busy server.

## SQL Statistics\SQL Compilations/sec

- A high number can be an indicator of ad hoc queries, this must be cross referenced with ad hoc plans in the plan cache.

## SQL Statistics\SQL Recompilations/sec

- If high determine recompilation reason with Xevent session. Usually stale statistics, Temp table usage and option WITH Recompile.
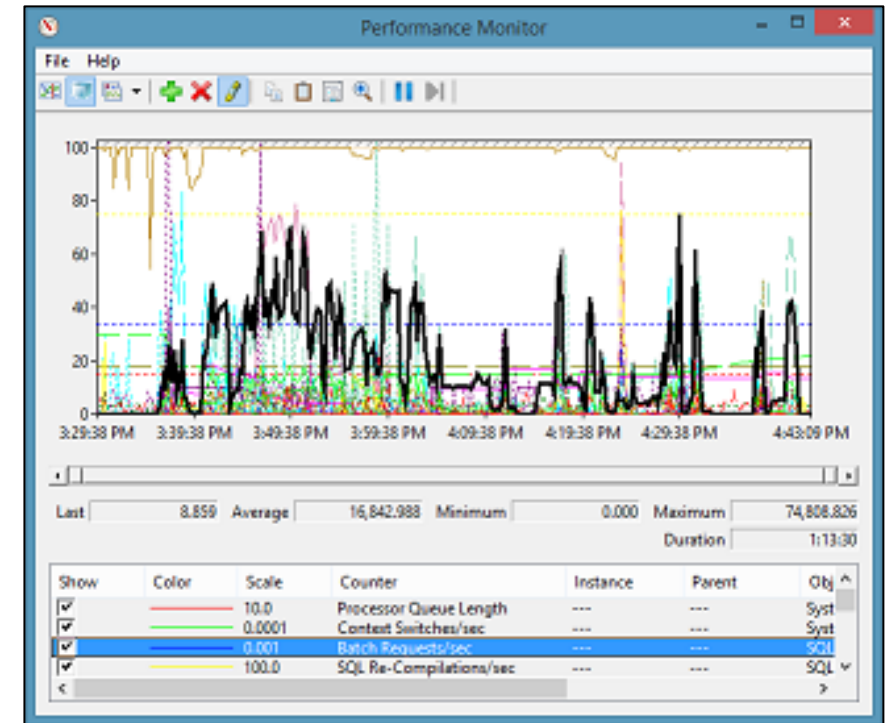
# Demonstration

Examining SQL Server
Scheduling

# Waits and queues

- Using waits and queues troubleshooting methodology


LAB

# Questions?

# Knowledge Check

While troubleshooting a resource bottleneck what two objects should you examine?

What are the three components that make up query execution time?

What is the difference between a PAGELATCH wait and a PAGEIOLATCH wait?

Which DMV should you examine to view historical wait information?

Which DMV should you examine to view current wait information?