Lesson Objectives

- In this Lesson you will learn the following:
  - SQL Server Operating System (SQLOS)
  - SQL Server Memory Management
  - SQL Server Disk I/O
  - SQL Server Process Scheduling
  - SQL Server Wait Statistics
  - SQL Server Resource Governor

SQL Server Operating System

SQL Server Task Scheduling

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

SQL Server Scheduling Terminology

- Task
  - A unit of work that is scheduled by SQL Server
  - A Batch will have one or more Tasks (aligns with statements)
  - Monitor with sys.dm_os_tasks

SQL Server Scheduling Terminology

- Worker Thread
  - Logical thread within the SQL Server process
  - Mapped 1:1 to a Windows thread
  - Each Task will be assigned to a single Worker Thread for the life of the Task
  - Monitor with sys.dm_os_workers

SQL Server Task Scheduling

- One SQLOS Scheduler per core/logical processor
- Handles scheduling tasks, I/O and synchronization of other resources
- Work requests are balanced across schedulers based on number of active tasks
- Monitor using sys.dm_os_schedulers

Scheduling Types

Non-Preemptive (Cooperative)

- SQL Server manages CPU scheduling for most activity (instead of the OS)
- SQL decides when a thread should wait, or get switched out
- SQL developers also programmed some pre-determined voluntary yields to avoid starvation of other threads

Preemptive

- Preemption is act of an OS temporarily interrupting an executing task
- Higher priority tasks can preempt lower priority tasks
- When this happens, it can get expensive
- Preemptive mode used in SQL Server for external code calls, CLR with an UNSAFE assemblies, APIs that could block for extended periods (example –extended stored procedures)

SQL Server Operating System

Waits and Queues

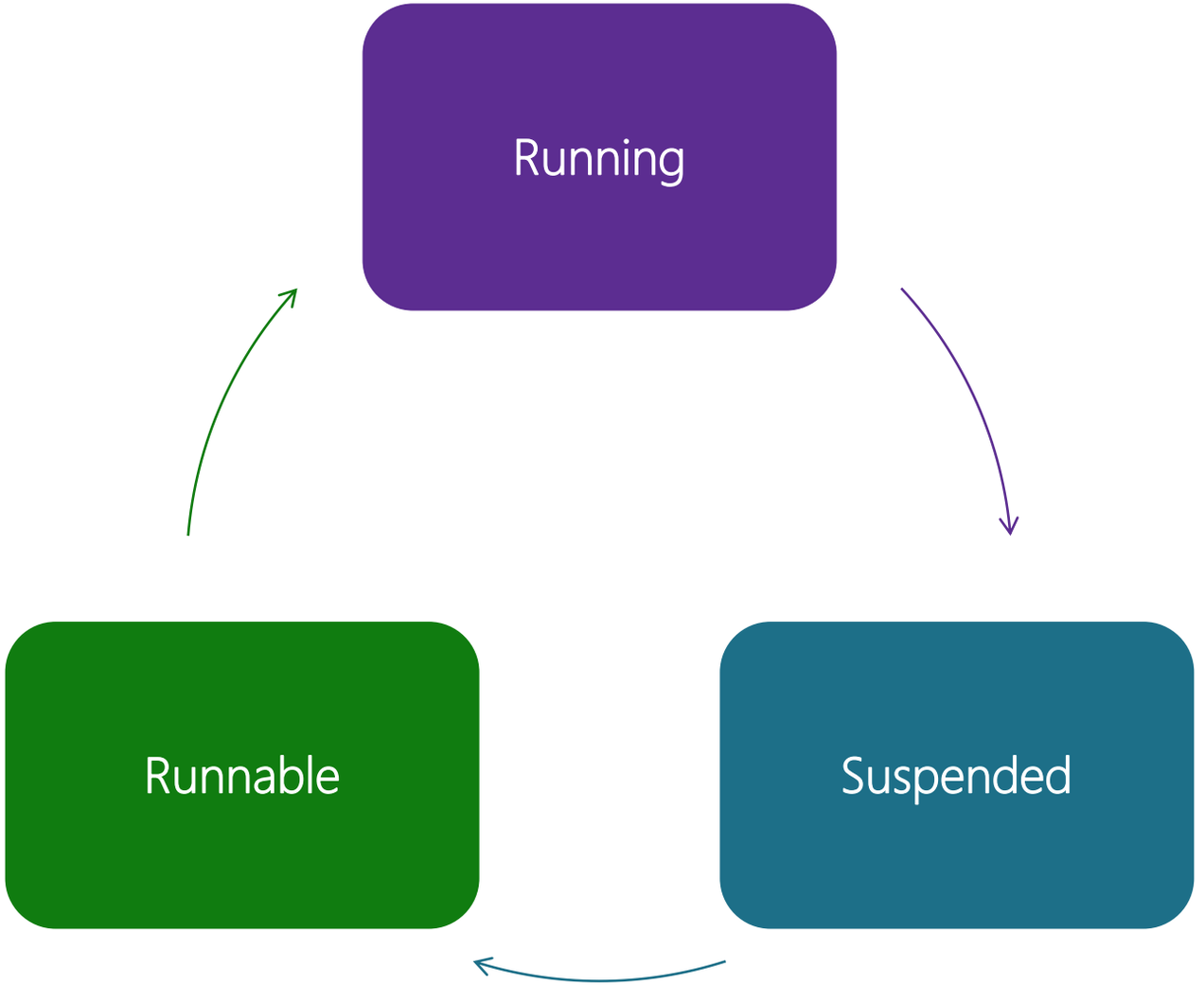
Waits and Queues

- The only way SQL Server can attend a volume of requests with a limited set of resources is to implement a methodology known as Waits and Queues
- SQL Server maintains the state in which workers wait
- Waits are the measurable resource capable of being tracked by SQL Server
- Queues maintain the runnable and waiting resource states in the execution model

Leveraging 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

Waits and Queues



Task Execution Model

Status: Running

session_id 51	Running

FIFO 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
session_id 56	Runnable

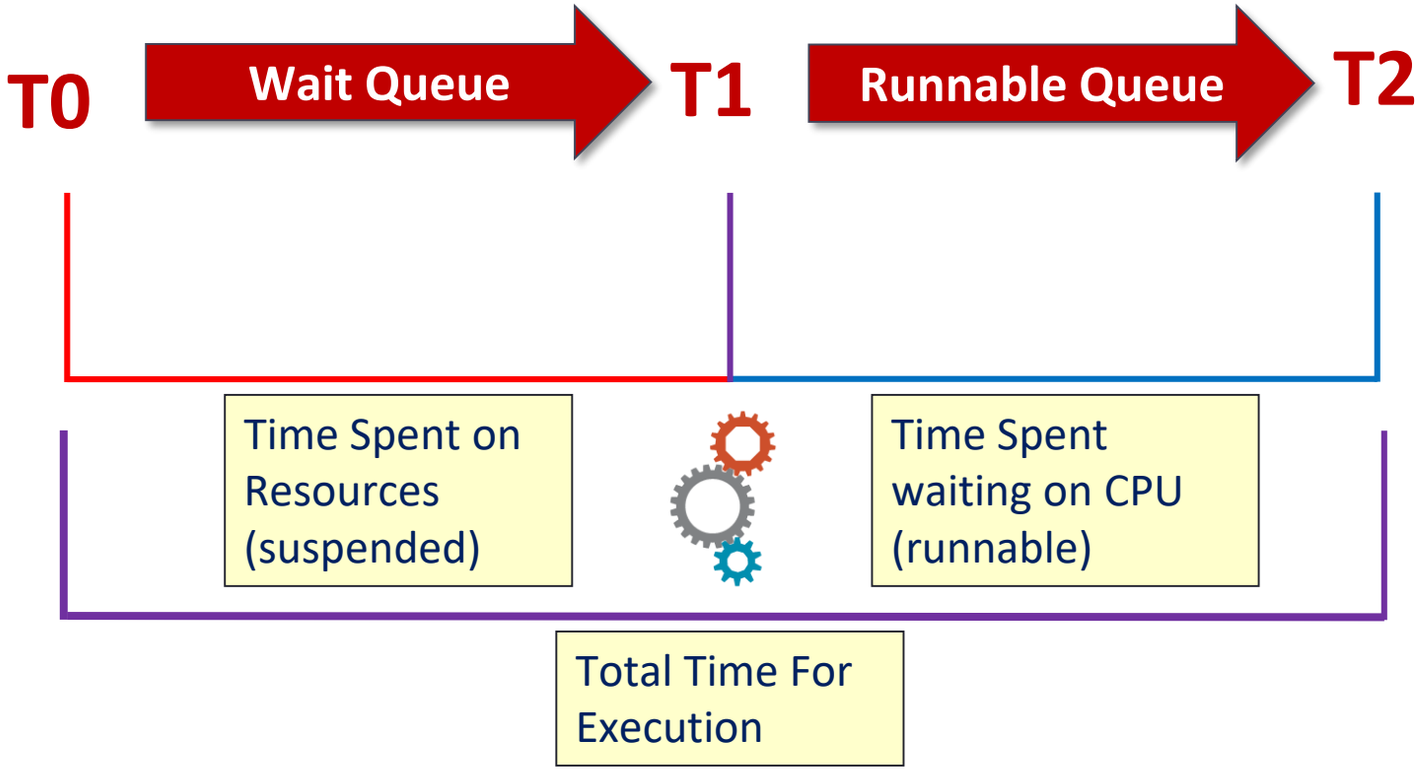
SPID56 moved to the bottom of the Runnable queue.

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

Unordered List

Wait Statistics Execution Model



Wait Types

Queue

- Locks
- Latches
- Network
- I/O

Resource

- Idle worker
- Background tasks (Deadlock monitor; Resource Monitor)

External

- Extended Stored Procedures
- Linked servers
- OLEDB
- SOS waits

Relevant DMVs

sys.dm_os_wait_stats

- Returns information about all the waits encountered by threads that executed
- Includes wait type, number of tasks that waited in the specific wait type, total and max wait times, as well as 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

Wait Stats DMV – Trending vs. Benchmarking

- What are the waits from an **historic** perspective?

- SELECT * FROM sys.dm_os_wait_stats

- This DMV is cleared on server restart or by using
  DBCC SQLPERF ('sys.dm_os_wait_stats', CLEAR)

What are the waits **right now**?

- Select from *sys.dm_os_wait_stats*
- Wait for a few seconds to allow some waits to be collected
- Clear the DMV status and check again

Or preferably…

- Persist output from *sys.dm_os_wait_stats*
- Wait for a few seconds to allow some waits to be collected
- Compare new collection and compare deltas

Waiting Tasks DMV

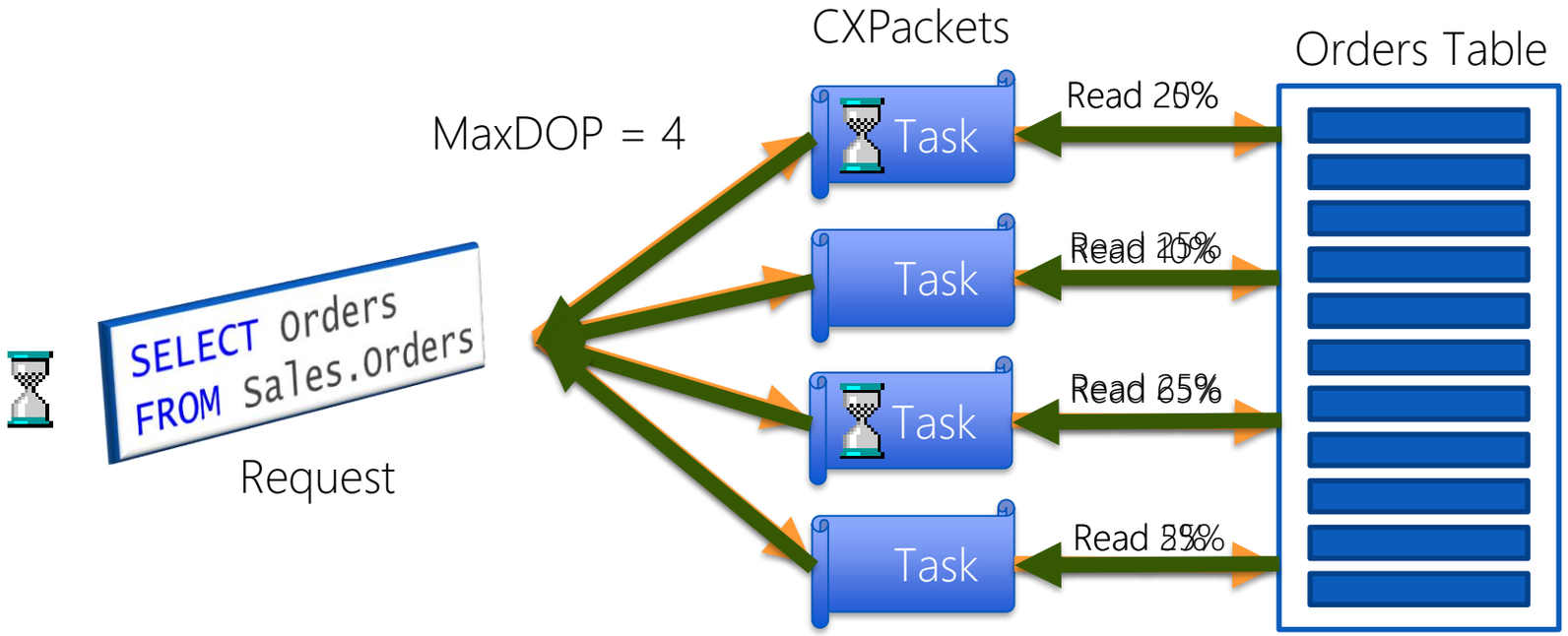
- Use *sys.dm_os_waiting_tasks* to show the waiter list at the current moment and join with *sys.dm_exec_requests*
- This allows us to find what is causing blocking at a given point in time
- Reports on all in-flight requests and their assigned tasks to provide their status and hierarchy (running, runnable, waiting)

SQL Server Operating System

Parallelism

Parallelism Overview

- Parallelism occurs when a request is divided into several tasks, each of them executing a piece of the required work.



Parallelism Overview

- On OLTP systems, parallelism may be a sign of inefficient workloads
- On OLAP/Reporting/DSS workloads, parallelism more of an advantage
- For mixed systems, it may be necessary to tune MAXDOP and the Cost Threshold for Parallelism

Influencing Parallelism

Default value of **Max Degree of Parallelism** is 0.

Use the following guidelines when you configure the **Max Degree of Parallelism** value on an OLTP system:

- For servers that use more than 8 processors, use MAXDOP=8
- For servers that use up to 8 processors, use MAXDOP=0 to N, where N equals the number of required processors
- For servers that have NUMA configured, MAXDOP should not exceed the number of CPUs that are assigned to a NUMA node
- For servers that have hyper-threading enabled, the MAXDOP value should not exceed the number of physical processors

Overriding the server configuration:

- MAXDOP query hint
- MAXDOP value configured with Resource Governor

Influencing Parallelism

- SQL Server creates and runs a parallel plan for a query only when the estimated cost to run a serial plan for the same query is higher than the value set in **Cost Threshold for Parallelism.**

- The default value for this configuration is 5 and is appropriate for most SQL Server instances. Thus, it rarely needs to be configured.
- For mixed systems, **look at the query costs** of those queries that really use parallelism to their advantage, and use this setting to disallow mote OLTP-centric workloads to use parallelism.
- Better yet, **tune your queries** so that the ones that might use parallelism in an inefficient way, stop using it if possible.

Parallelism information in Query plans

Starting SQL Server 2012, XML showplan is enhanced to include the reason why the plan is not or cannot be parallelized:

- **MaxDOPSetToOne**: Max Degree of Parallelism set to 1 at query or server level
- **NoParallelDynamicCursor**: Dynamic cursor doesn't support parallel plan
- **NoParallelFastForwardCursor**: Fast Forward cursor doesn't support parallel plan
- **NoParallelCreateIndexInNonEnterpriseEdition**: We don't' support parallel index operations for non Enterprise editions
- **NoParallelPlansInDesktopOrExpressEdition**: No parallel plan for express edition (SQL 2000 desktop edition is the same as express edition for later builds)
- **TSQLUserDefinedFunctionsNotParallelizable**: Scalar TSQL user defined function used in the query
- **CLRUserDefinedFunctionRequiresDataAccess**: If a CLR user defined function ends up access data via context connection, the query can't be parallelized. But a CLR user defined function that doesn't do data access via context connection can be parallelized.
- **NoParallelForMemoryOptimizedTables**: This is for any query accessing memory optimized tables (part of SQL 2014 in-memory OLTP feature)

Notable 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 data pages I/O problems
- 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

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

SOS_SCHEDULER_YIELD

- Might indicate CPU pressure if very high overall percentage of Processor Time

CXPACKET

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

Notable Waits

THREADPOOL

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

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

LCK_xx

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

Notable Waits

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 perfmon counters Memory Grants Pending and Memory Grants Outstanding

Lesson Knowledge Check

- If a request is experiencing signal waits, what will be its status in sys.dm_exec_requests?
- Which scheduler will a new task be assigned to?
- How might you determine why a query is running slow?

Demonstration: Examining SQL Server Scheduling

- Comparing requests, sessions, and connections
- Currently executing spids with Wait Type with statement and with Plan

Demonstration: Examining SQL Server Scheduling

- Calculating Wait Stats and Waiting Tasks

Demonstration: Examining SQL Server Scheduling

- Wait Stat Deltas
- Sys.dm_os_waiting_tasks
- Wait Statistics in Extended Events

(Optional) Demonstration: Wait Statistics

- Isolating Top Waits for Server Instance Since Last Restart
- Wait Stat Deltas

(Optional) Demonstration: Wait Statistics

- Wait Stat Deltas

Lesson Knowledge Check

- What makes up the entire execution time of a query?
- How can I get the waits stats from the last 60s?
- What is the difference between a PAGELATCH and a PAGEIOLATCH?