Understanding Key SQL Server Concepts

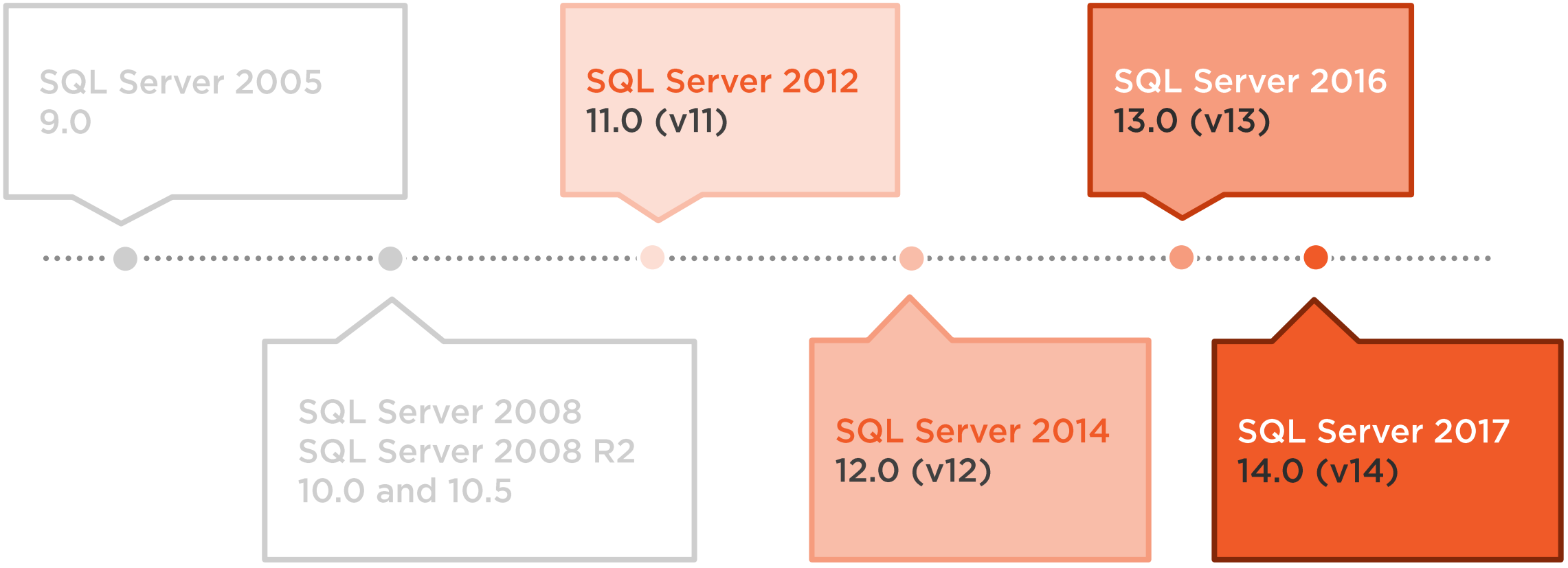
Viktor Suha
DATABASE DEVELOPER / DBA

@realeddiesson www.linkedin.com/in/viktor-suha-86b27893

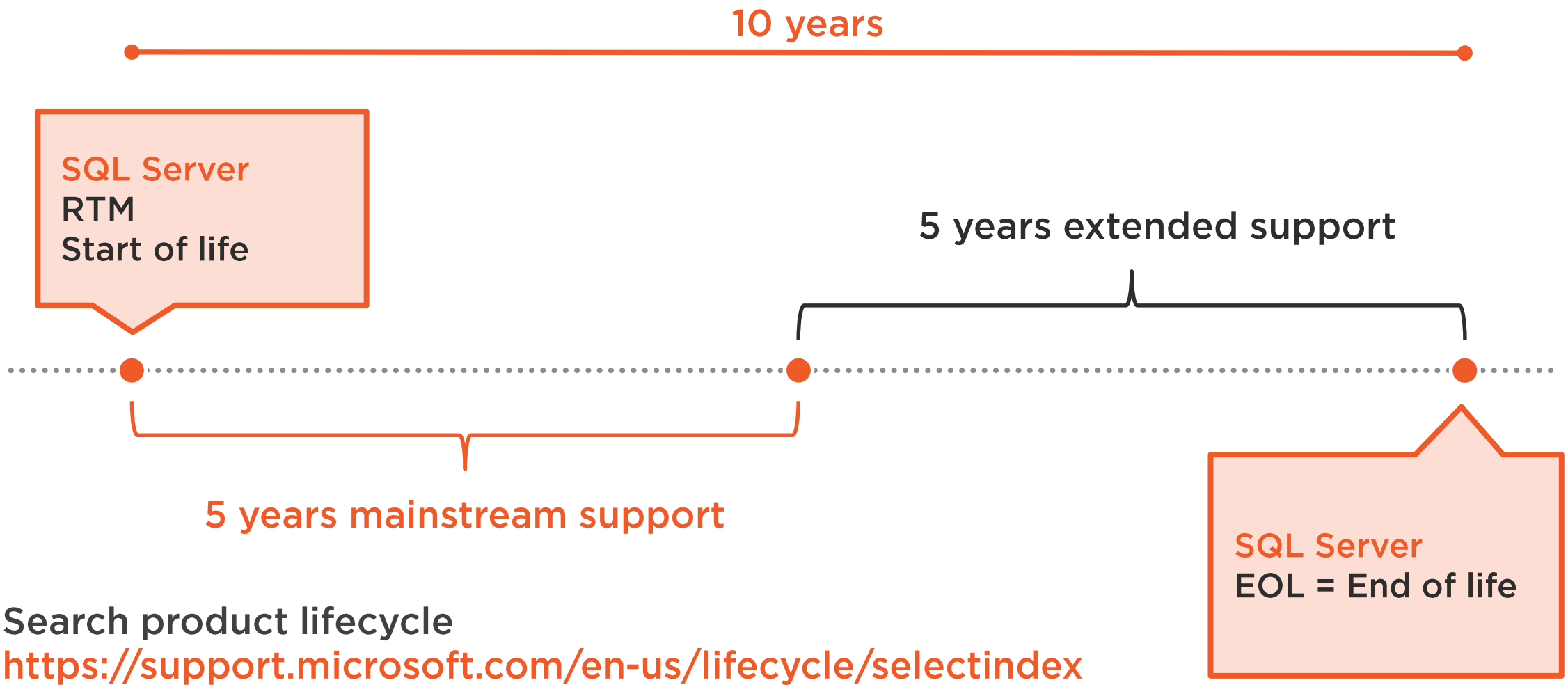
Version, Edition, Patching

SQL Server Major Versions

Supportability and servicing



SQL Server Product Lifecycle



Search product lifecycle
https://support.microsoft.com/en-us/lifecycle/selectindex

Database Compatibility Levels

Major Version	Supported Compatibility Levels
SQL2017	140 130 120 110 100
SQL2016	130 120 110 100
SQL2014	120 110 100
SQL2012	110 100 90

Determines database and workload behaviour

Compatibility Level Impact on Performance

Compatibility Level 110

Legacy Cardinality Estimator (CE) v70

Serial SELECT INTO plans

Compatibility Level 120

New CE v120

Parallel SELECT INTO plans

Compatibility Level Impact on Performance

Compatibility Level 120

SQL2014 CE v120

Serial INSERT in INSERT-SELECT plans

TF2371 behaviour is OFF

Serial statistics sampling

Compatibility Level 130

SQL2016 CE v130

Parallel INSERT in INSERT-SELECT plans

TF2371 behaviour is ON

Parallel statistics sampling

Patch Level and Servicing Example

- **CU6 for SP2 – 13.0.5292.0**
- **…**
- **CU1 for SP2 – 13.0.5149.0**

SP2 (Service pack 2) – 13.0.5026.0

- CU14 for SP1 – 13.0.4560.0
- …
- CU1 for SP1 – 13.04411.0

SP1 (Service pack 1) – 13.0.4001.0

- CU9 for RTM – 13.0.2216.0
- …
- CU1 (Cumulative update 1) for RTM – 13.0.2149.0
- Security update 4019088 – 13.0.1742.0

RTM (Release to manufacture) – 13.0.1601.5

```
/* returns:  major version,
             patch version,
             latest CU/KB article applied,
             edition,
             OS version,
             CPU architecture
*/
SELECT @@VERSION AS [Server_Version];
```

Check Version Info with @@VERSION

How to query detailed server version information with T-SQL

Microsoft SQL Server 2017 (RTM-CU13-OD) (KB4483666) - 14.0.3049.1 (X64)
Dec 15 2018 11:16:42 Copyright (C) 2017 Microsoft Corporation
Developer Edition (64-bit) on
Windows 10 Pro 10.0 <X64> (Build 17763:)

```sql
SELECT
 /* ProductBuild applies to SQL2014+ only */
 SERVERPROPERTY('ProductBuild') AS BuildNumber,
 SERVERPROPERTY('ProductLevel') AS VersionLevel,
 SERVERPROPERTY('ProductMajorVersion') AS MajorVersion,
 SERVERPROPERTY('ProductMinorVersion') AS MinorVersion,
 SERVERPROPERTY('ProductUpdateLevel') AS UpdateLevel,
 SERVERPROPERTY('ProductUpdateReference') AS UpdateReference,
 SERVERPROPERTY('ProductVersion') AS ProductVersion;
```

Check Version Info with SERVERPROPERTY()

How to query detailed server version information with T-SQL

Search the Microsoft Knowledge Base

How to use the KB articles

- *search term* site: support.microsoft.com

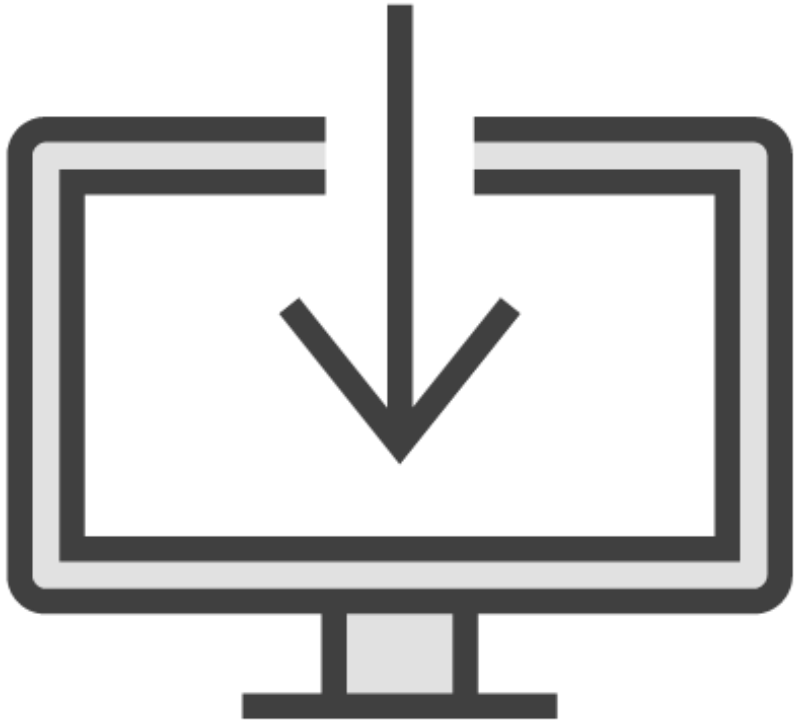
KB article URL format

- https://support.microsoft.com/en-us/help/321185

KB321185 to read

- How to determine the version and the edition, plus the list of available build numbers

Important Servicing Updates



No Service Packs from SQL2017 onwards

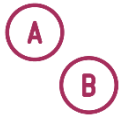
- https://support.microsoft.com/en-us/help/4041553
- Modern Servicing Model (MSM)

Service Packs still released for earlier versions

Proactive and ongoing patching with CUs

- For all supported versions

Why Does the Major Version Matter?



Set of available features

Workload behaviour

Troubleshooting, configuration, and problem resolution options

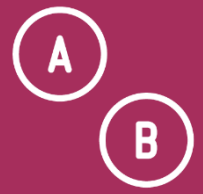
Supportability and patching options

Why Does the Patch Level Matter?

Security

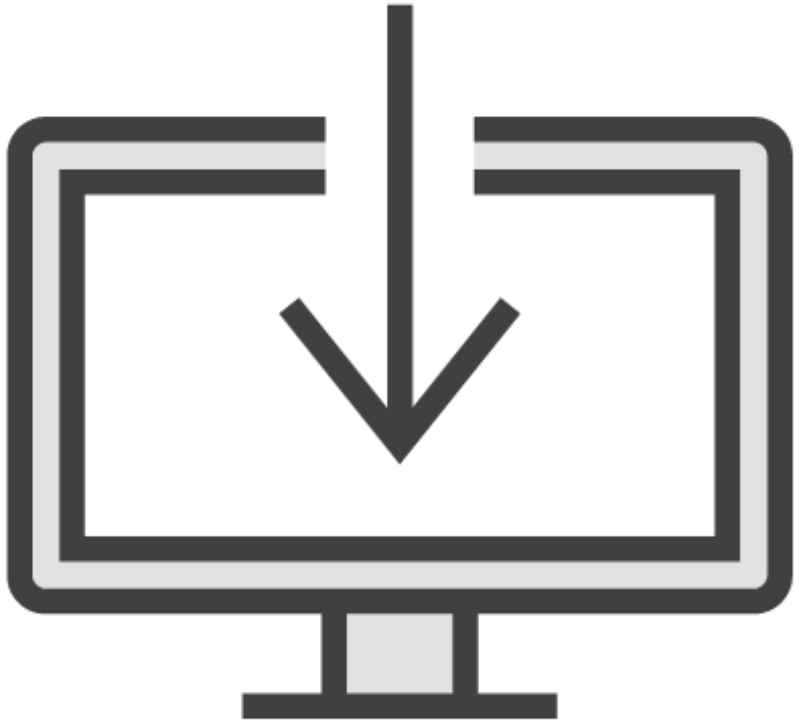
Stability

Performance



New features and behaviour

What Is Included in a CU Package?



CU14 for SQL2017 (KB4484710)

- https://support.microsoft.com/en-us/help/4484710

Bug fixes for multiple services

- SQL Engine, SSIS, SSAS

Performance fixes and improvements

Feature improvements

Security fixes and improvements

SQL Server Editions

Enterprise

Standard

Developer

Express

Web

Why Does the Edition Matter?

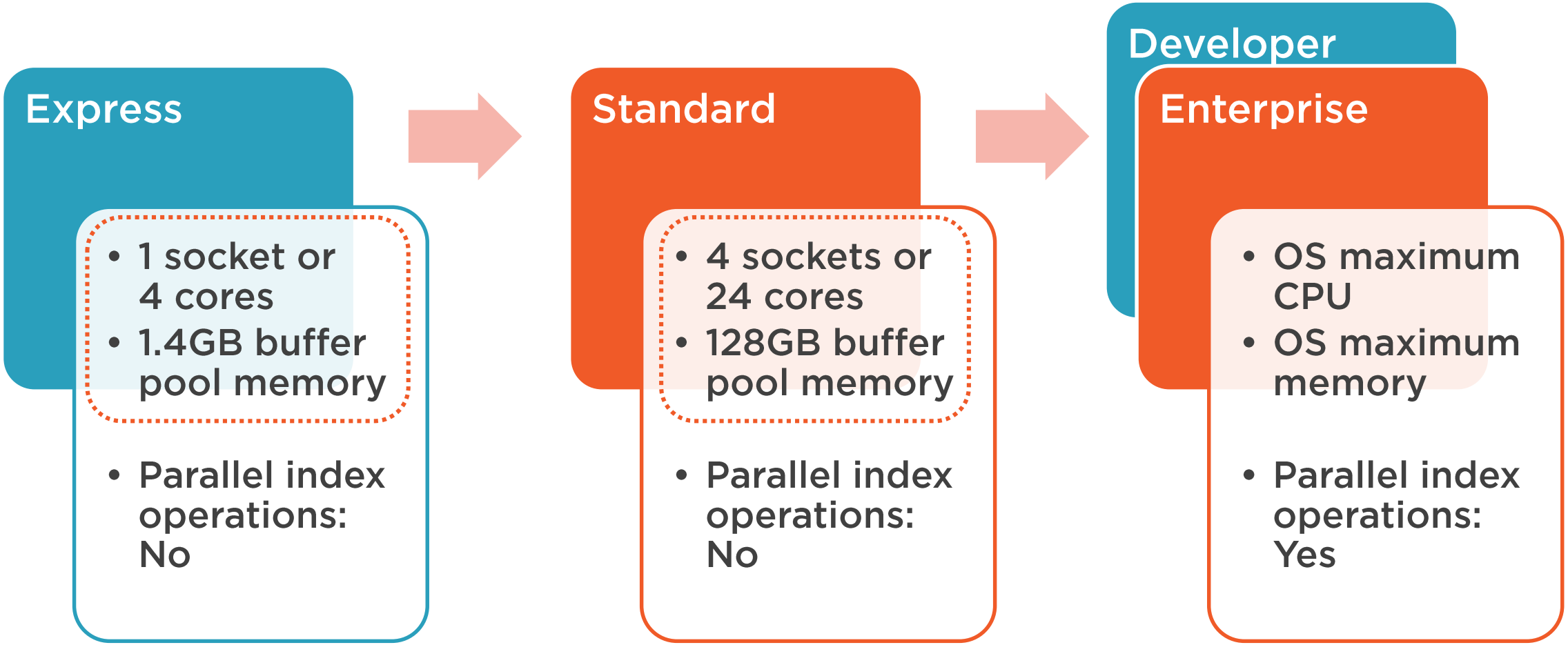
Feature limitation

Scale and resource utilization limits

Licensing cost

SQL Server Edition and Feature Matrix

Editions and supported features of SQL Server 2017
https://bit.ly/2JWsmHz



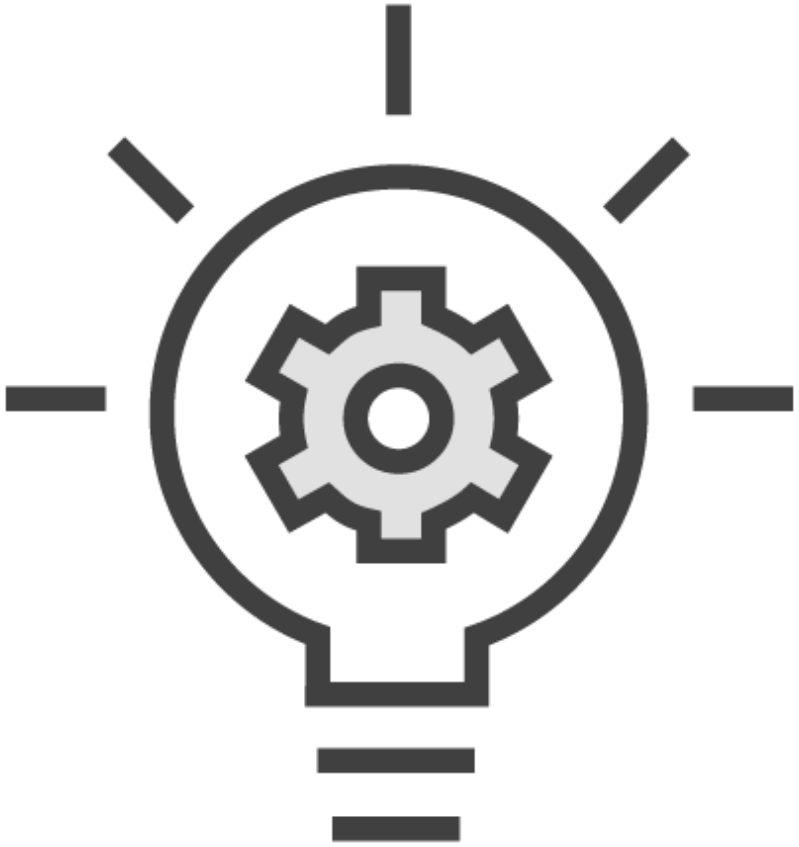
```
SELECT @@VERSION AS [Server_Version_Edition];

SELECT SERVERPROPERTY('Edition') AS Edition;

/* reading the current SQL ERRORLOG header */
EXEC sp_readerrorlog 0, 1, 'Copyright';
```

Check Edition Info

How to query server version and edition information with T-SQL

Open the raw SQL ERRORLOG files in a text editor to see the version and edition without connecting to the server instance

Version and Edition Best Practices



Keep the major version supported

- Preferably within the mainstream support phase

Keep the patch level supported

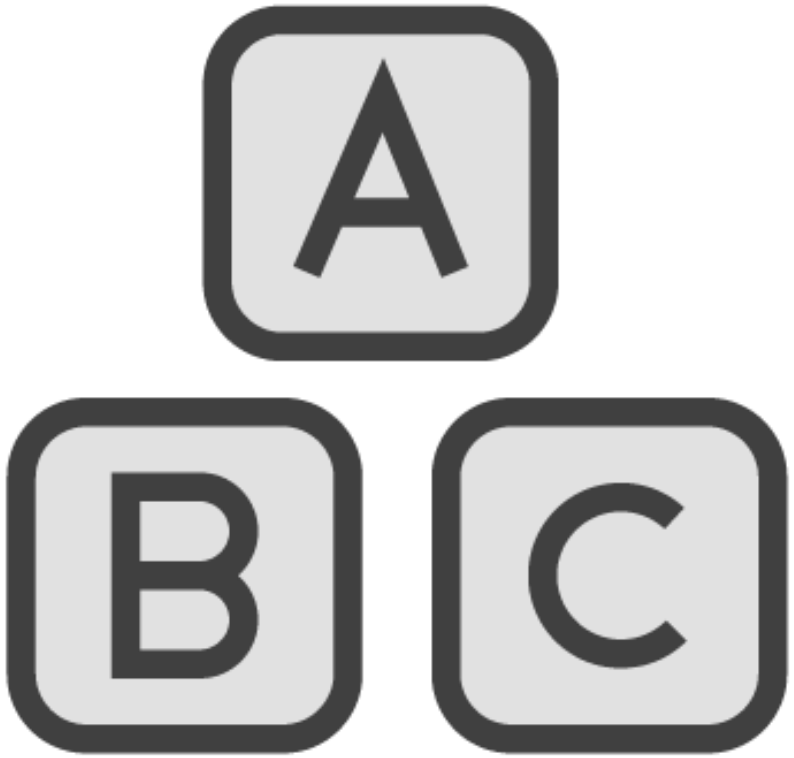
- Patch proactively and regularly with SPs and CUs
- Set up retention policy and test

Use Developer Edition for testing

Research the edition and feature matrix

Server Instances

What Is an Instance?



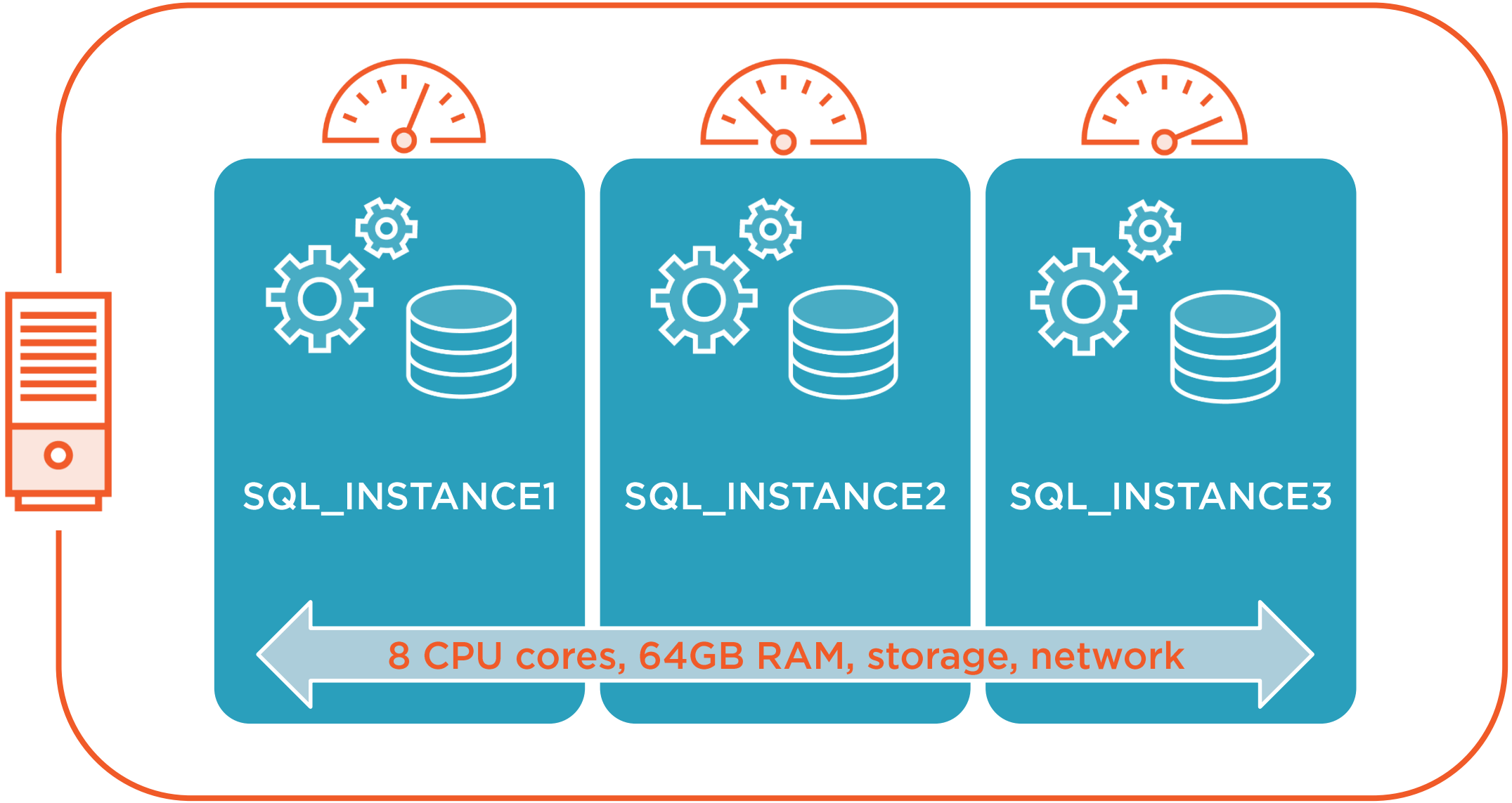
Each instance is a separate SQL Server with:

- System and user databases
- Configuration
- TCP port number

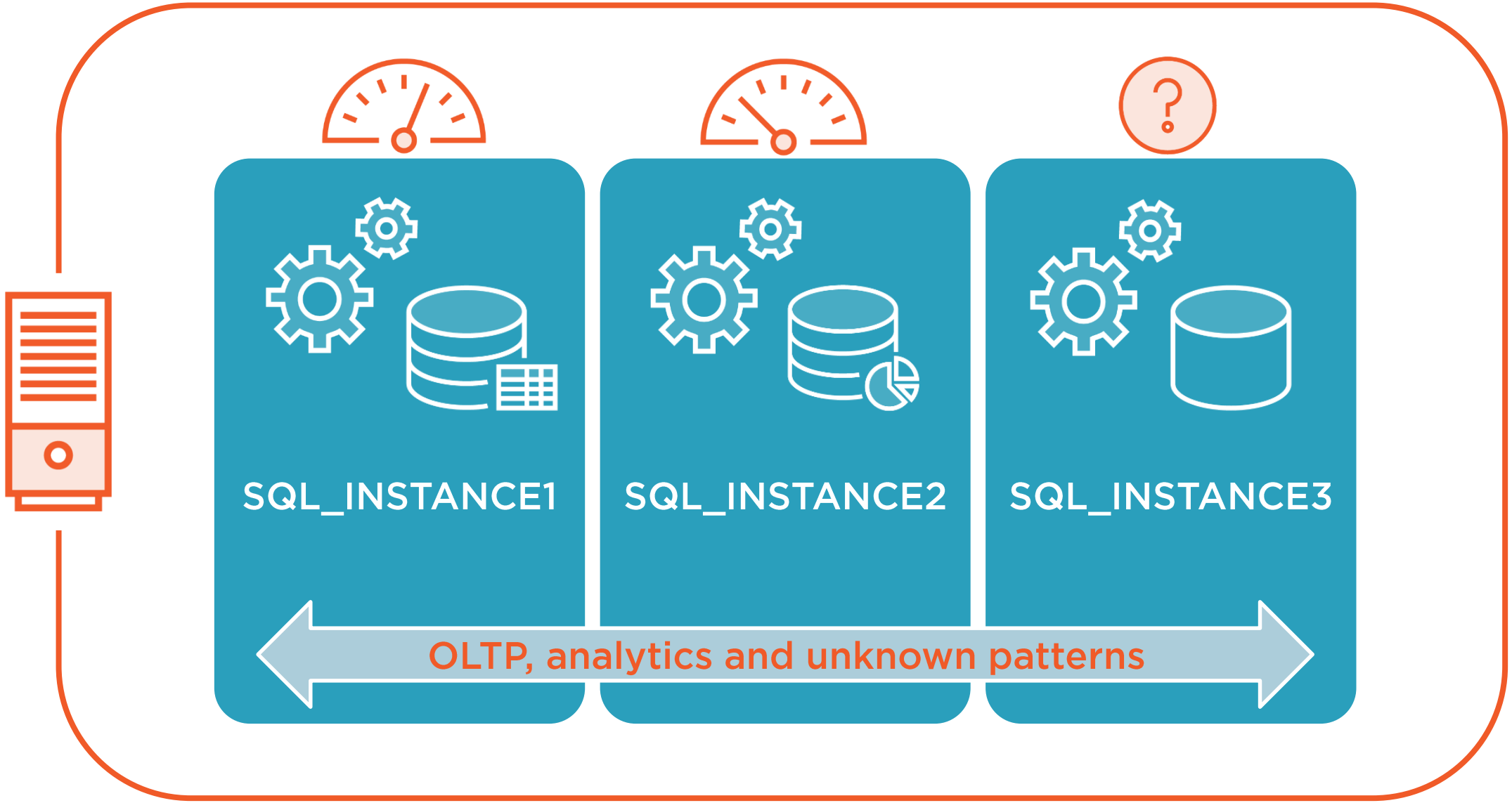
Multiple instances of the database service can be installed side-by-side on the same machine

- One default instance and multiple named instances
- Can even mix major versions

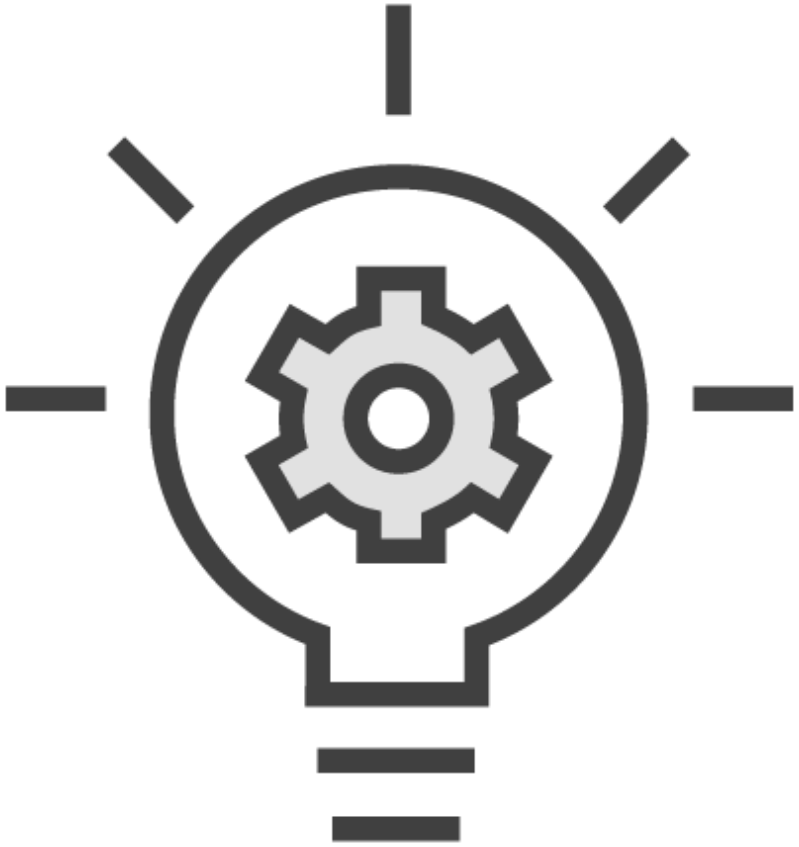
Instances and Shared Resources



Instances and Mixed Environments



Instances and Performance Optimization



Multi-instance deployments

- Create scalability problems
- Make performance troubleshooting and optimization more complex
- Make operations more complex

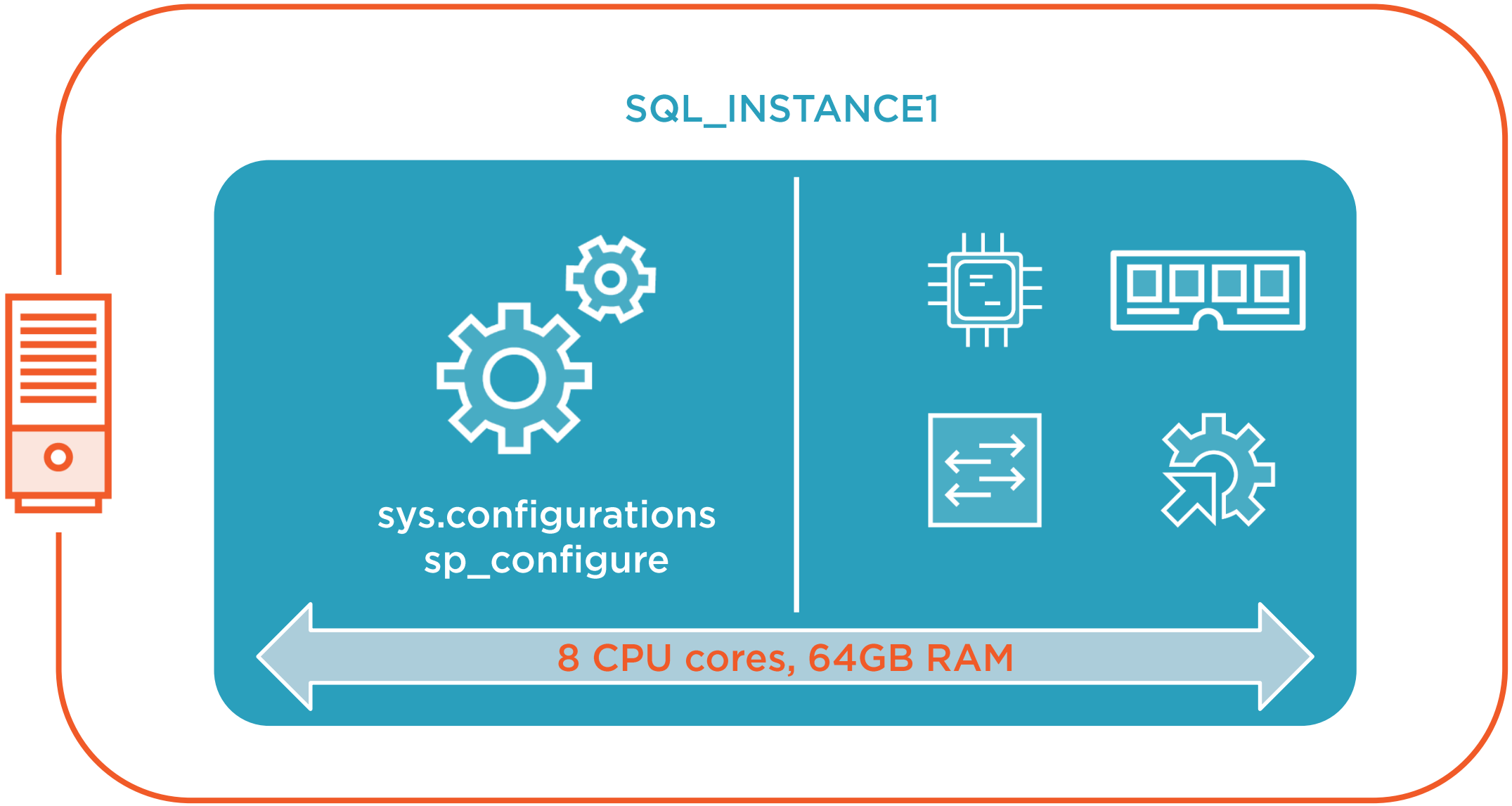
Consolidate with virtualization

Do not stack SQL Server instances in production

- Aim for one instance per server

Server Configuration Options

Server Configuration Options



Which Settings to Change from the Default?

Must change

- Max server memory
- Min server memory
- Max degree of parallelism (MAXDOP)
- Cost threshold for parallelism

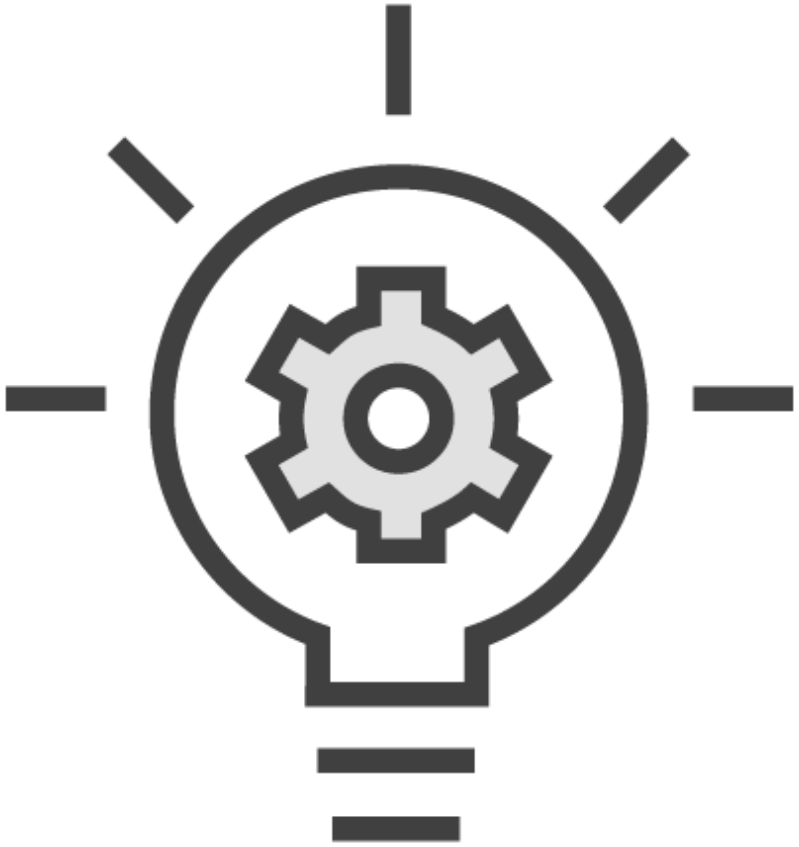
Might change

- Optimize for ad-hoc workload

Do not change

- Priority boost

Server Configuration and Performance



Validate and pre-configure the instance for:

- Memory settings
- CPU configuration
- Parallelism settings

Verify all settings in sys.configurations

Verify the SQL ERRORLOG files

Database Configuration Options

Database Configuration Options

sys.databases

SET options

ALTER DATABASE

sys.database_scoped_configurations

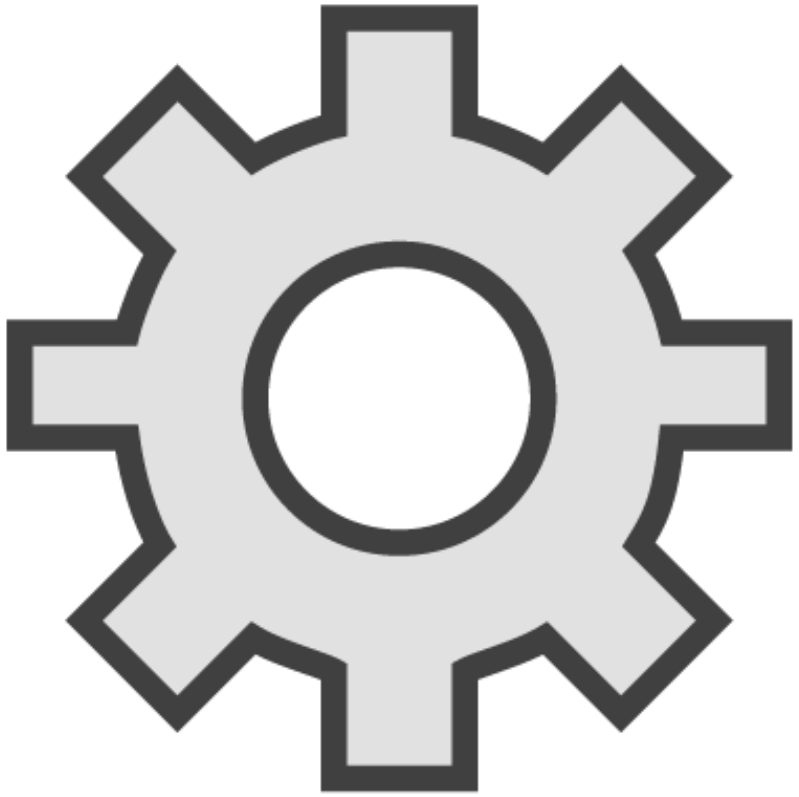
Scoped configurations

ALTER DATABASE

All versions

SQL2016+

Database SET Options



A few SET options of interest

- COMPATIBILITY_LEVEL
- AUTO_CLOSE and AUTO_SHRINK
- Auto create and update statistics options
- RECOVERY
- AUTOMATIC_TUNING (SQL2017+)

sys.databases view

sys.database_automatic_tuning_options view

DATABASEPROPERTYEX() function

Database Scoped Configurations

Common scoped configuration options

- MAXDOP
- LEGACY_CARDINALITY_ESTIMATION
- QUERY_OPTIMIZER_HOTFIXES
- CLEAR PROCEDURE_CACHE

Configuration Hierarchy

Server configuration
- MAXDOP
- TF4199

Database scoped configurations
- MAXDOP
- QUERY_OPTIMIZER_HOTFIXES

SQL2016+

Query hint
- MAXDOP
- ENABLE_QUERY_OPTIMIZER_HOTFIXES
- QUERYTRACEON

```
sp_configure 'max degree of parallelism', 4;
RECONFIGURE; /* SERVER instance level */

/* DATABASE level - Database Scoped Configuration */
ALTER DATABASE SCOPED CONFIGURATION SET MAXDOP = 2;

/* QUERY level – OPTION query hint */
SELECT ColdRoomSensorNumber, COUNT(ColdRoomTemperatureID)
FROM Warehouse.ColdRoomTemperatures_Archive
GROUP BY ColdRoomSensorNumber OPTION (MAXDOP 8);
```

Setting MAXDOP in Different Scopes

How to use the configuration hierarchy

The OPTION (MAXDOP 8) query hint overrides the upper level settings

```
SELECT ColdRoomSensorNumber, COUNT(ColdRoomTemperatureID)
FROM Warehouse.ColdRoomTemperatures_Archive
GROUP BY ColdRoomSensorNumber
OPTION (USE HINT ('FORCE_LEGACY_CARDINALITY_ESTIMATION'));
```

USE HINT Query Options

How to use the new USE HINT query options

Using the old, legacy CE just for this particular query regardless of database or server level settings

Trace Flags

What Is a Trace Flag?

A switch to alter SQL Server behaviour

- Referred as: TF*nnnn* e.g:TF1118

Mostly designed for short-term troubleshooting by MS customer support

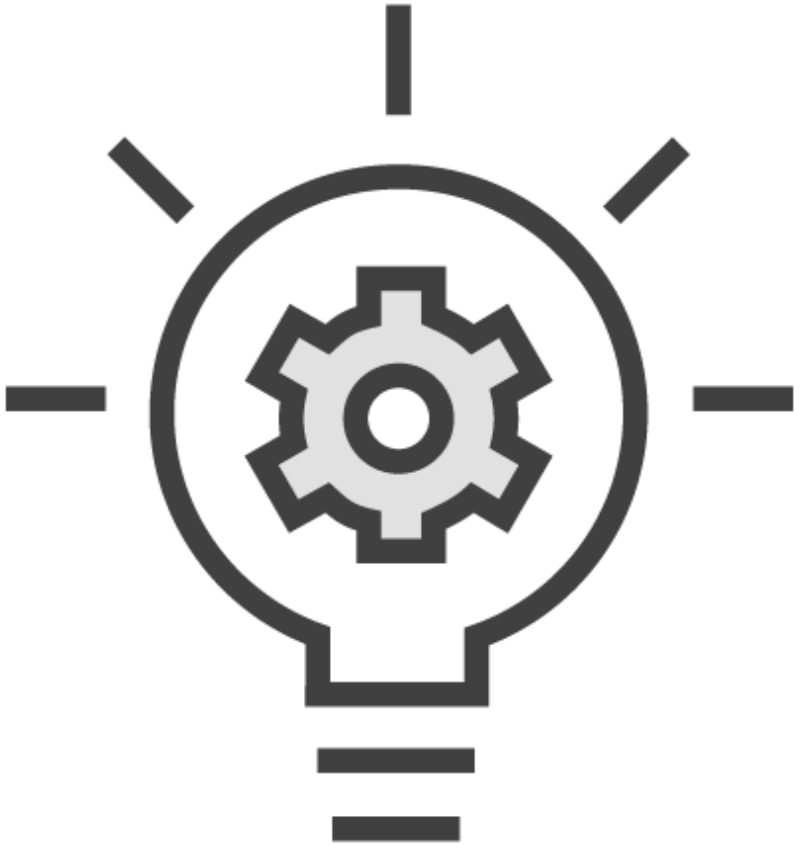
- Many should not be used in Production
- Can cause serious problems

But: a few are useful and have become the baseline or otherwise recommended

Can be set at:

- Server level (global)
- Session level
- Query level

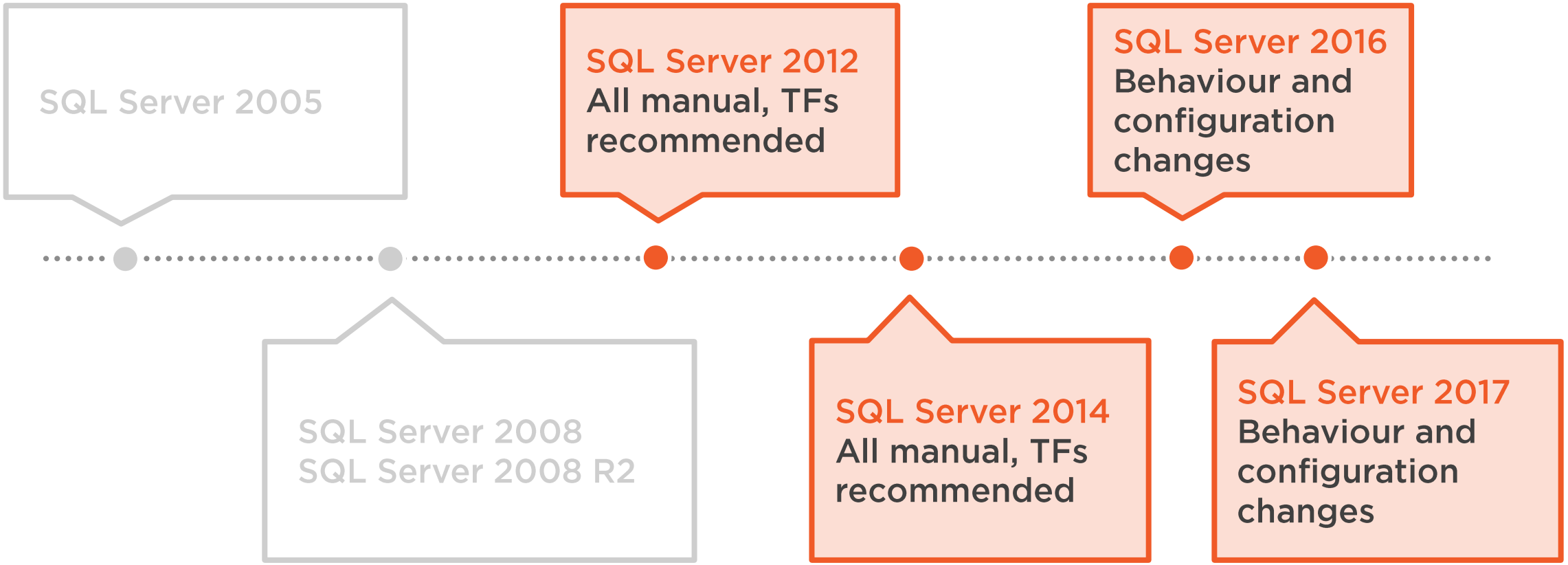
Performance Trace Flag Use Cases



Common scenarios

- Optimize tempdb in pre-SQL2016 versions
- Turn on Query Optimizer fixes
- Control the version of Cardinality Estimator (as part of the Query Optimizer) to be used
- Control how and when statistics are being automatically updated

Trace Flag Configuration Across Versions



Common Performance Trace Flags

SQL2016+	SQL2014	SQL2012
Database scoped configurations	1118 and 1117	1118 and 1117
Query hints	4199	4199
4199	2371	2371
9481 and 2312	9481 and 2312	

SQL2016+ Trace Flag Behaviour Changes

Trace Flag	Change
TF1118 and TF1117	Default behaviour, TF deprecated, ALTER DATABASE options
TF2371	Default behaviour, TF deprecated
TF4199	Pre-current version fixes enabled by default, TF can be set for current version fixes
TF9481 and TF2312	Can still be set globally, or use database scoped configurations, or use query hints

```
DBCC TRACEON (9481);     -- set TF for the current session

DBCC TRACEON (9481,-1); -- set TF globally for the server

-- set TF at query level
SELECT CustomerID, CustomerName, PostalCityID
FROM Sales.Customers
WHERE CustomerName LIKE N'Wing%’
OPTION (QUERYTRACEON 9481);
```

Trace Flag Configuration ON

How to turn on trace flags in SQL Server for different scopes

Global trace flags can be set as startup parameters (-T) too for the database service

```
-- set TF at query level with USE HINT
SELECT CustomerID, CustomerName, PostalCityID
FROM Sales.Customers
WHERE CustomerName LIKE N'Wing%’
OPTION (USE HINT ('FORCE_LEGACY_CARDINALITY_ESTIMATION’));

-- set TF at database level with Scoped Configuration
ALTER DATABASE SCOPED CONFIGURATION SET
LEGACY_CARDINALITY_ESTIMATION = ON;
```

SQL2016+ Trace Flag Alternatives

Beginning with SQL2016 trace flags are not encouraged, SP1 introduced trace flag replacement query hints too

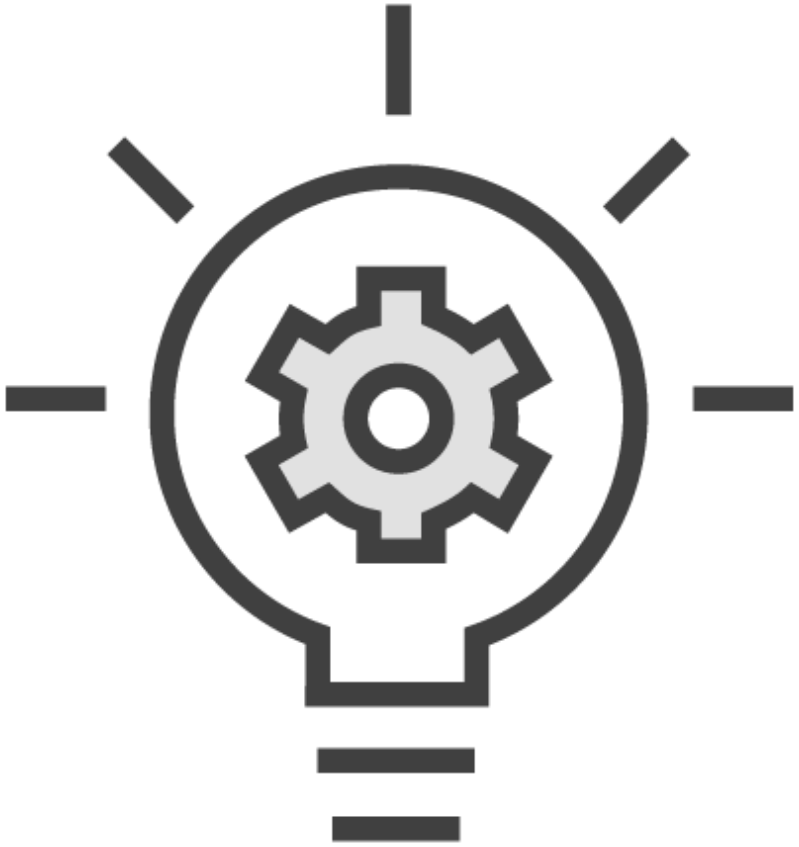
```
DBCC TRACESTATUS(-1);     -- check all configured TFs globally
DBCC TRACEOFF (9481);     -- turn off TF in the session
DBCC TRACEOFF (9481,-1); -- turn off TF globally
```

Trace Flag Configuration OFF

How to check if trace flags are configured and then turn off trace flags in SQL Server

Global trace flags set as startup parameters need to be removed from the startup parameter list

Trace Flags and Performance



TFs really can make a difference, pro and con

It is hard to track the settings and the changed behaviours across different versions and scopes

TF usage or the equivalent configuration settings must be justified and documented

Tempdb

What Is Tempdb?

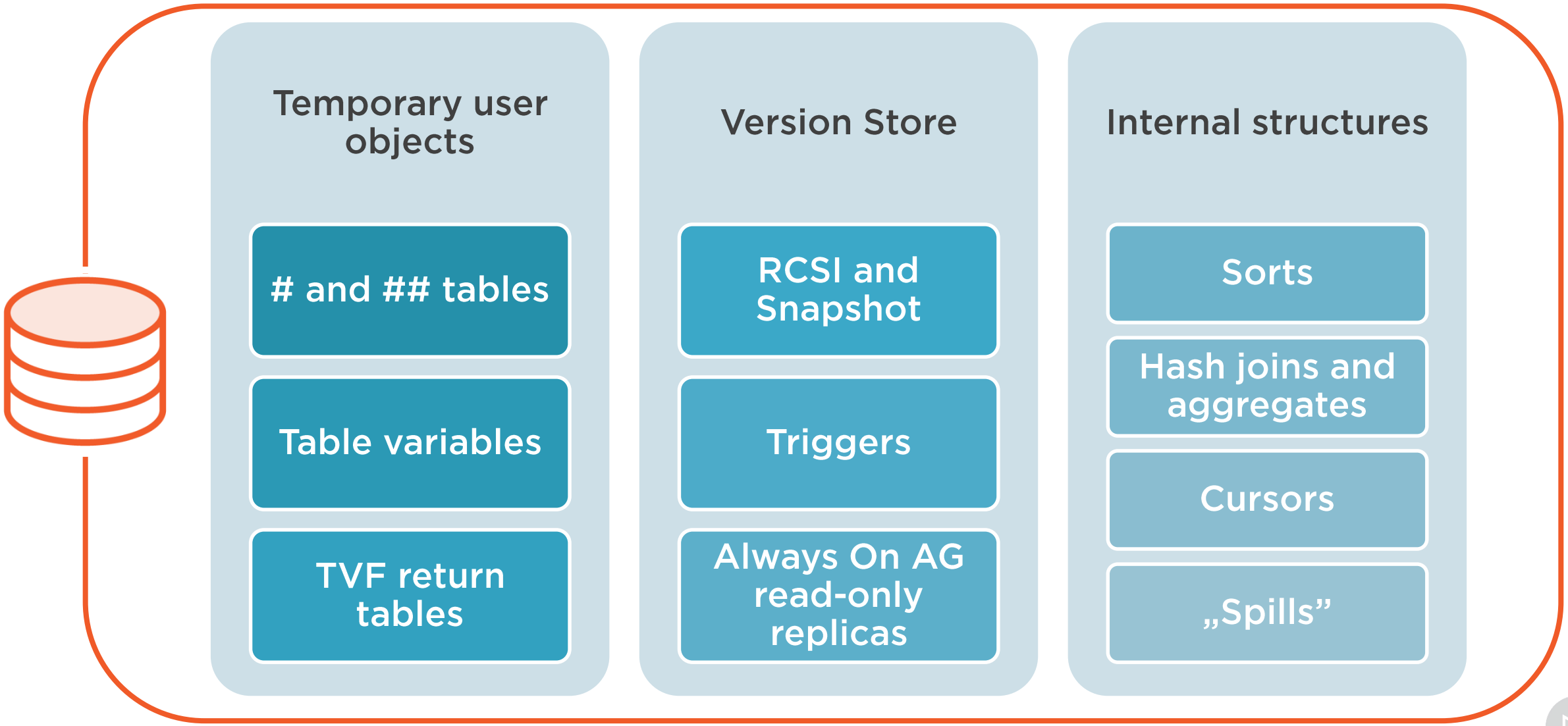
- Shared container to be used by everyone in the same server instance
- One tempdb database per instance
- You can explicitly create objects in it
- SQL Server uses it for internal objects and certain features too
- Reset with each service restart

```
-- create a local temporary table explicitly
CREATE TABLE #mytemp (id int);

-- create a local temporary table with SELECT..INTO
SELECT CustomerID, CustomerName, PostalCityID
INTO #mycustomers
FROM Sales.Customers
WHERE CustomerName LIKE N'Wing%’;
```

Tempdb Common Use Case

The most common scenario is to use tempdb for temporary user objects

Tempdb: What Is it For?



Tempdb Performance Factors

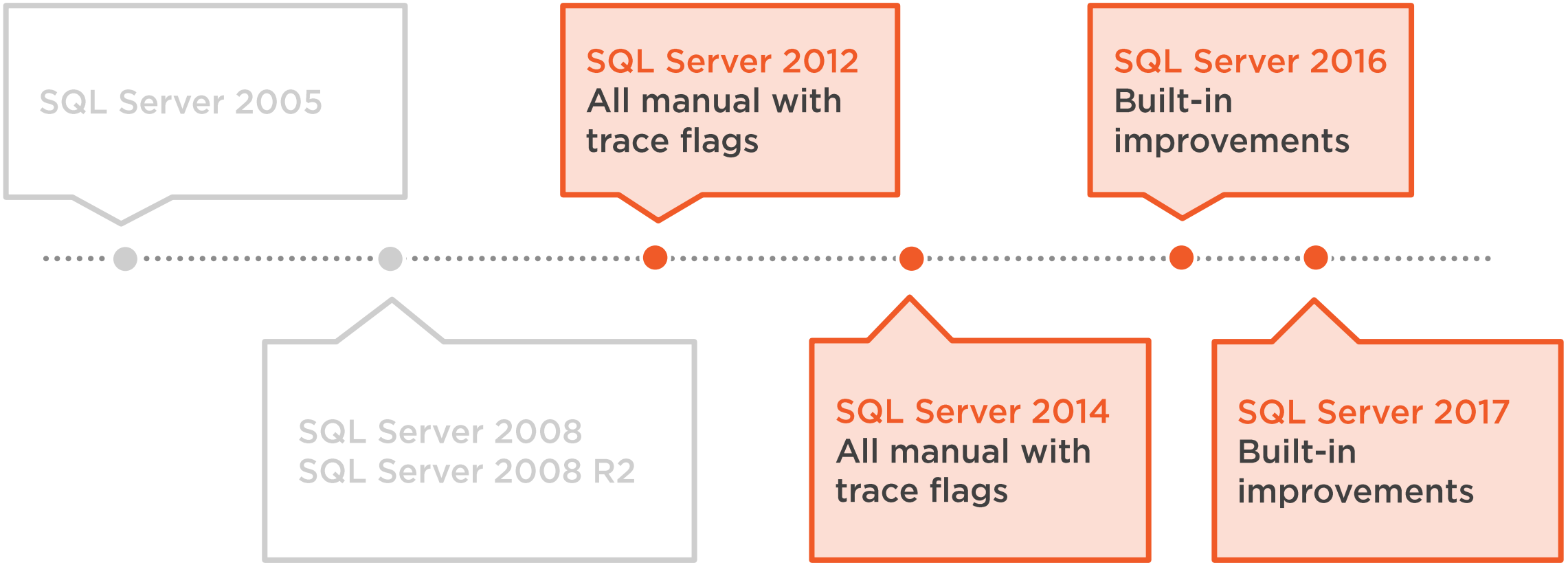
IO

Read/Write the data file(s) and the log file with varying IO sizes

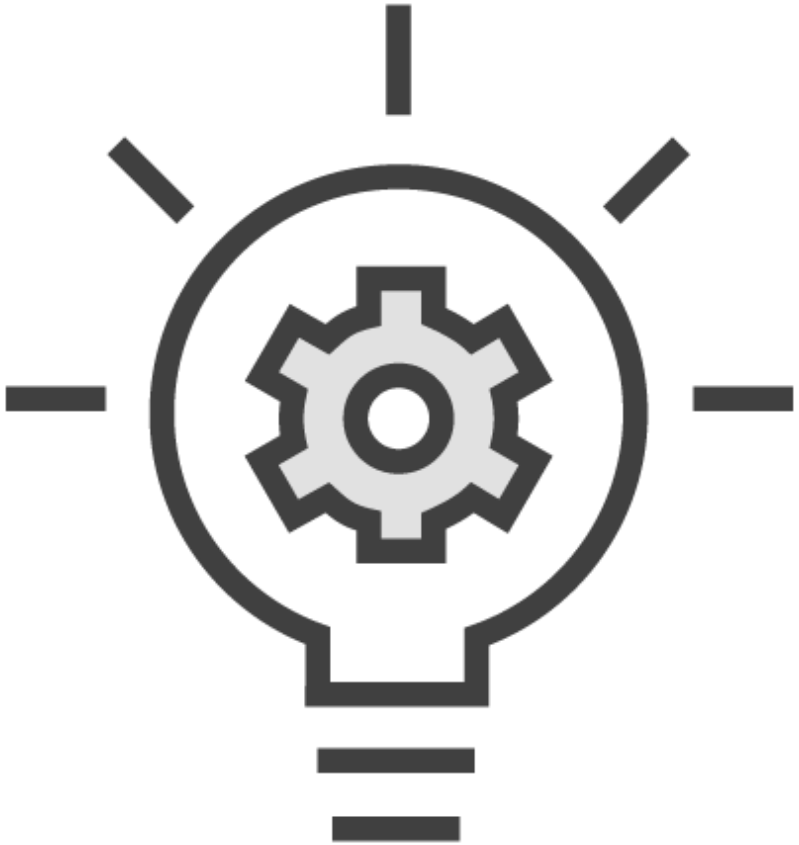
Internal allocation

SQL Server internal algorithms to allocate objects and manage metadata

Tempdb Configuration Across Versions



Tempdb and Performance



Common performance problems due to

- Slow storage
- Memory bottlenecks
- Configuration
- Badly written application

Plan with tempdb usage in-advance

Pre-configure

Monitor and adjust

Database Recovery Models

Recovery Models

Simple

Automatic log clearing: Full and differential backups only

Full

Log clearing only with log backup: Full, differential, and log backups

Bulk-logged

Transaction Log

What Is the Transaction Log?

.ldf file

- Each database has one and should have only one log file

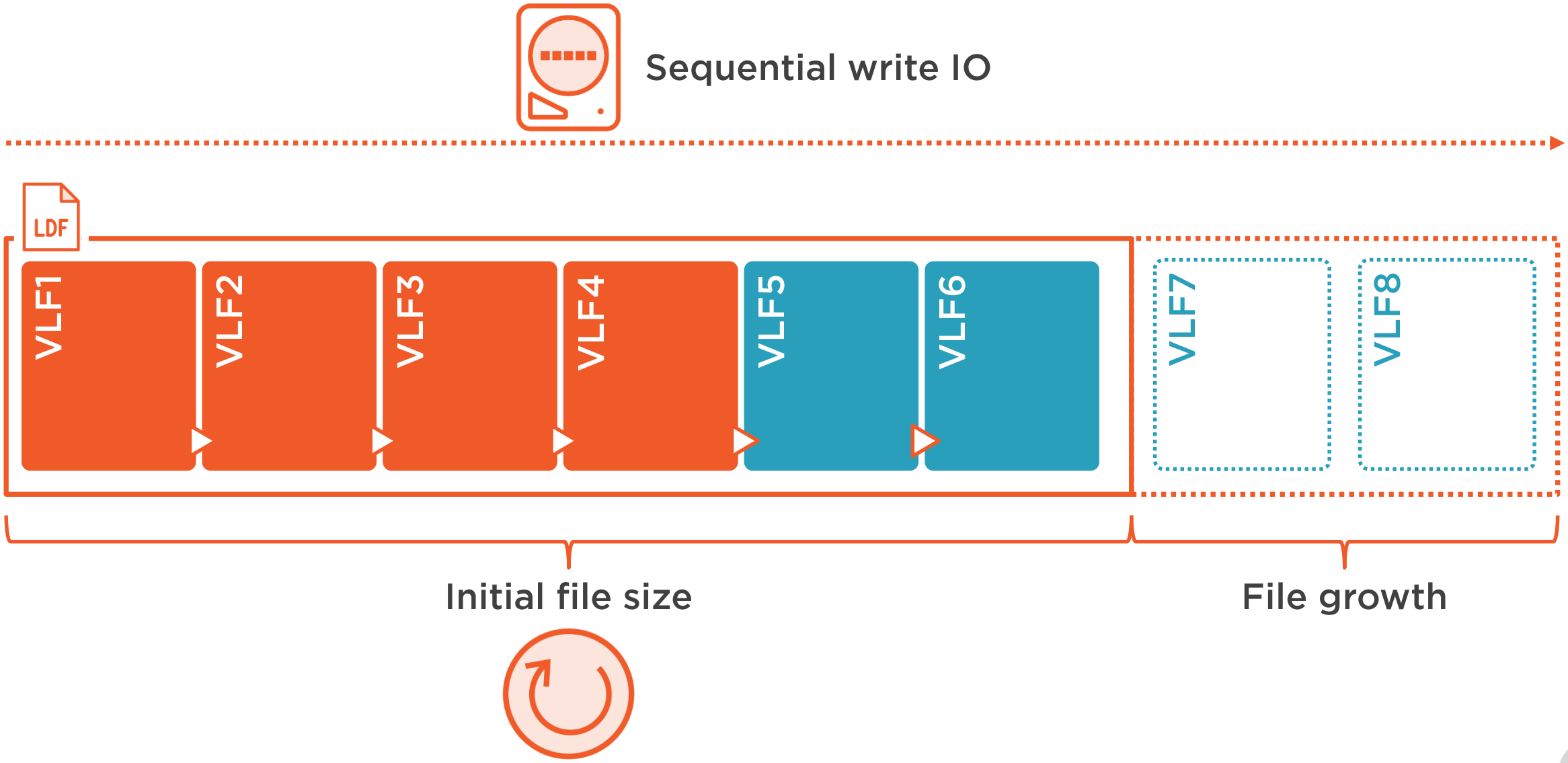
Write-ahead logging mechanism

- Everything is a transaction

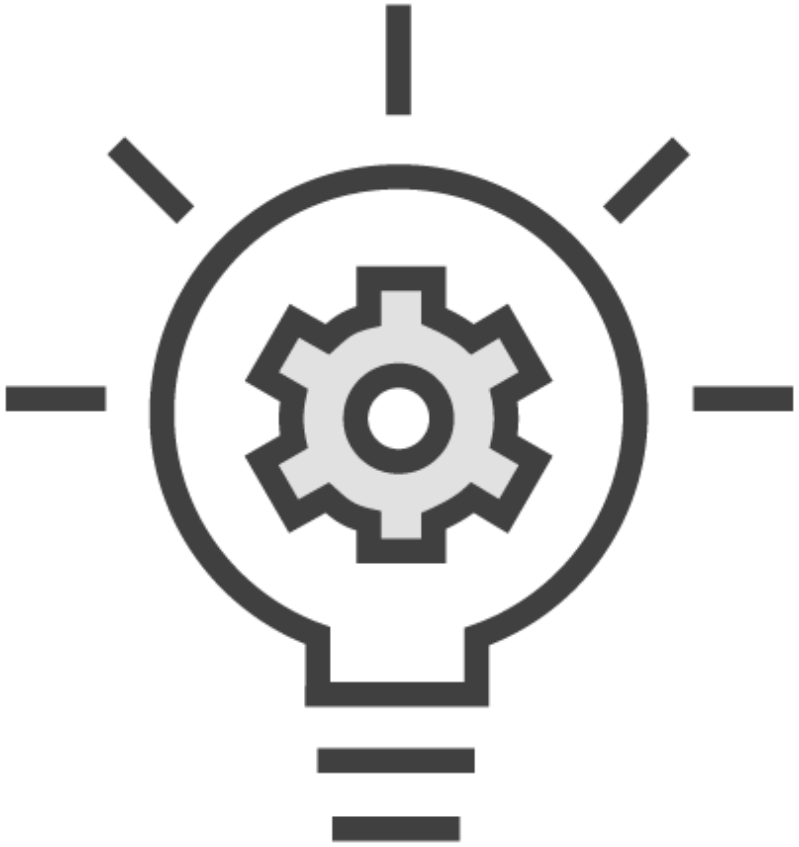
ACID properties (Durability)

Basis for database recovery

Transaction Log Management



Transaction Log and Performance



Do frequent enough log backups in Full

Instant file initialization *does not* help

Use fixed and meaningful growth values, pre-size the file

Watch out for long running transactions

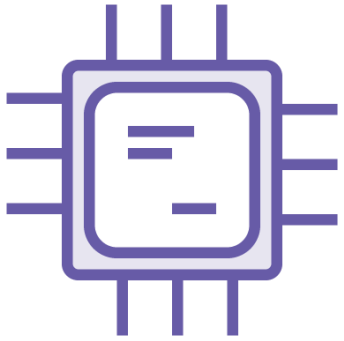
Do not shrink the file regularly

Importance of log IO performance

- Sequential write IO patterns

SQLOS: Resource Management

SQL Server Resource Management

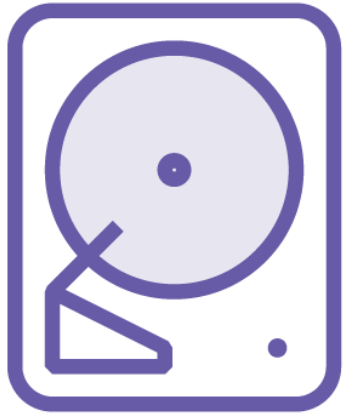


CPU/Scheduling

Serial and parallel plans, number of cores and performance

Memory

Buffer pool, plan cache, and query workspace



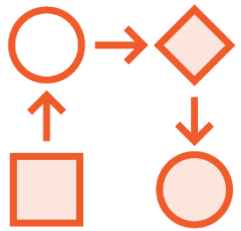
IO

Random and sequential read/write IO patterns

SQLOS:Memory Management

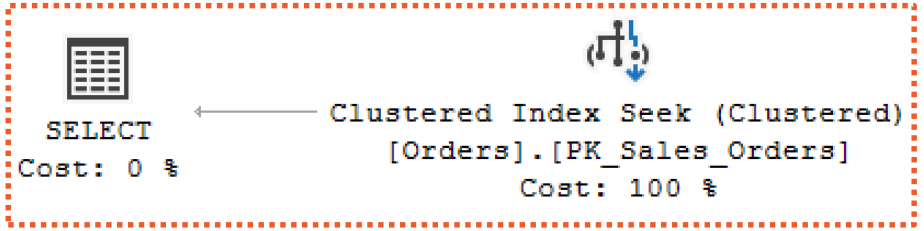
Plan Cache Memory

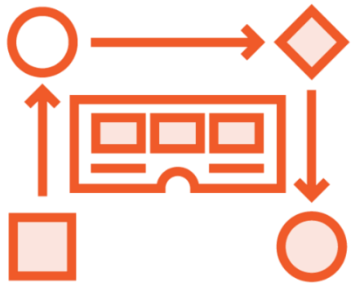
```
SELECT OrderDate
FROM Sales.Orders
WHERE OrderID = 1;
```



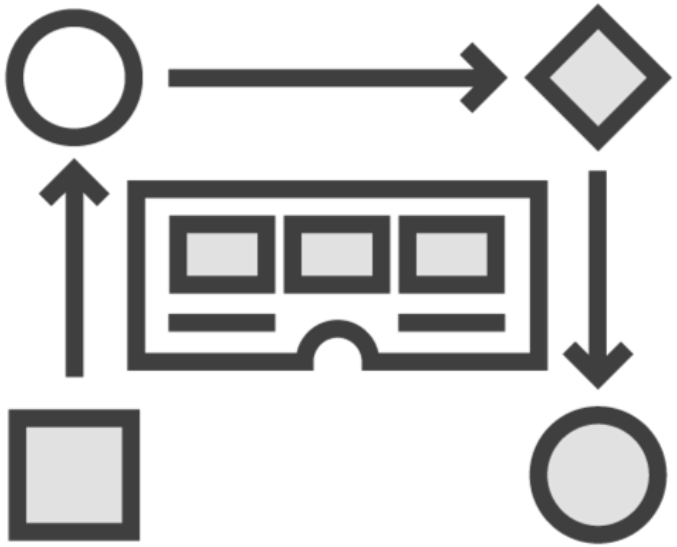
Compile

- Adhoc queries
- Prepared statements
- Stored procedures





Plan Cache Memory and Performance



Managed dynamically

Under throttling by SQL Server

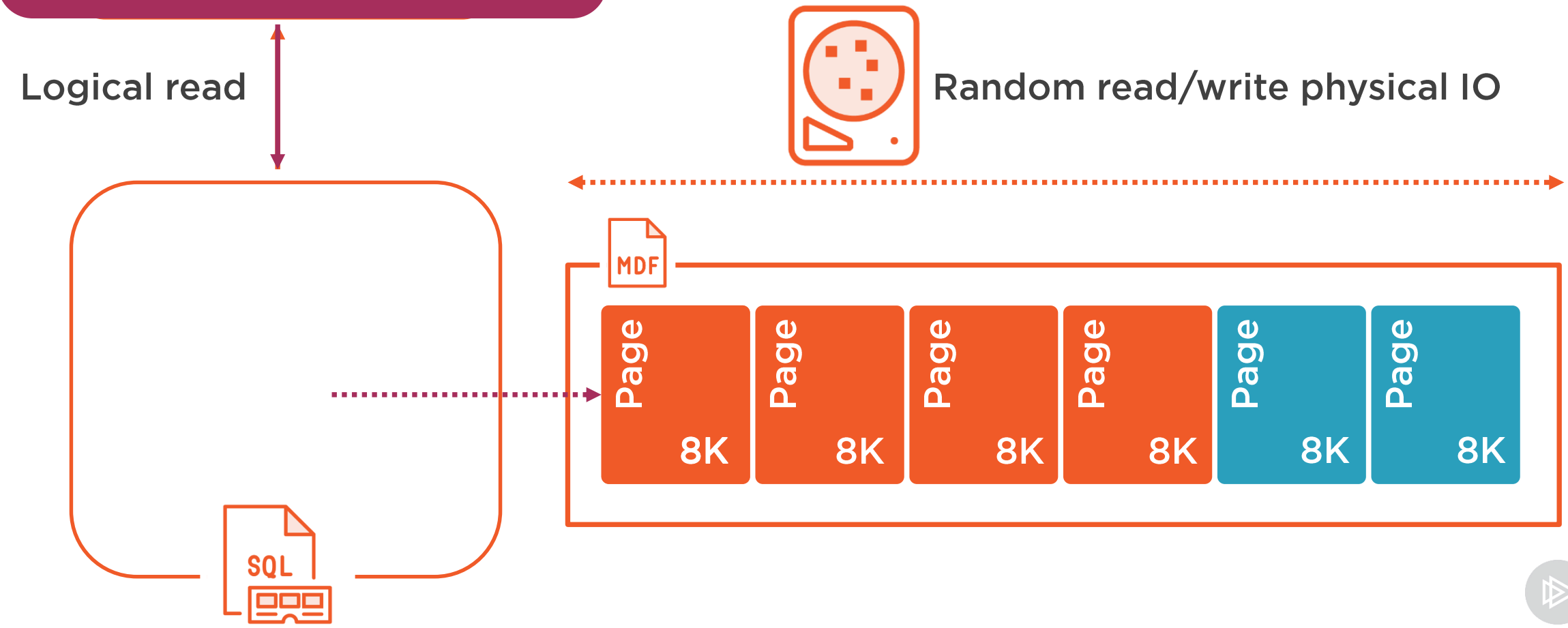
Importance of solution design

- Can be bloated in size with adhoc plans

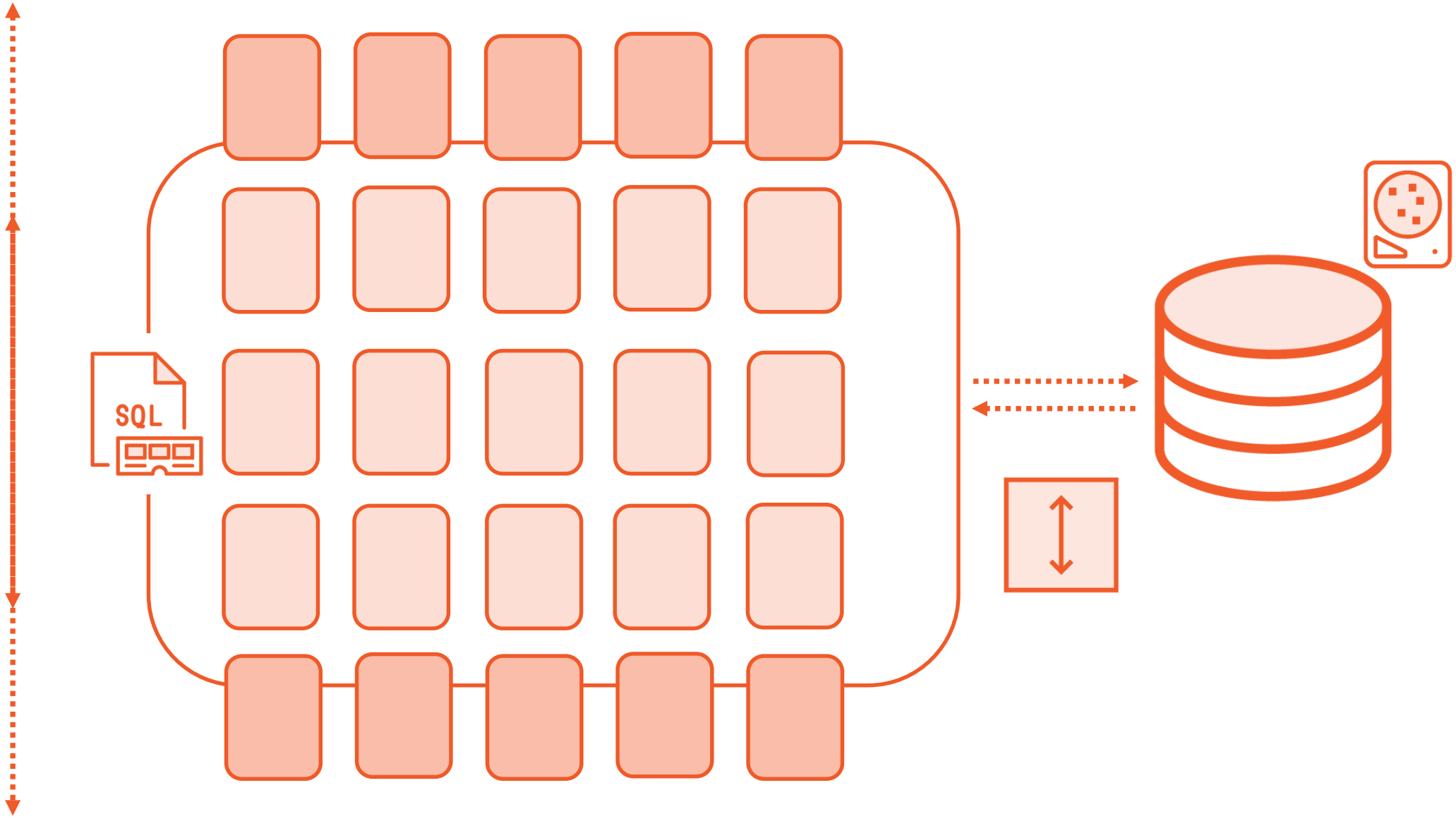
sys.dm_exec_cached_plans view

Buffer Pool Memory Usage

```
UPDATE Sales.Orders
SET OrderDate = GETDATE()
WHERE OrderID = 1;
```



Buffer Pool Memory Sizing



Buffer Pool Memory and Performance

Managed dynamically, can grow and shrink

Aim is to have a proper size to limit costly physical IO operations

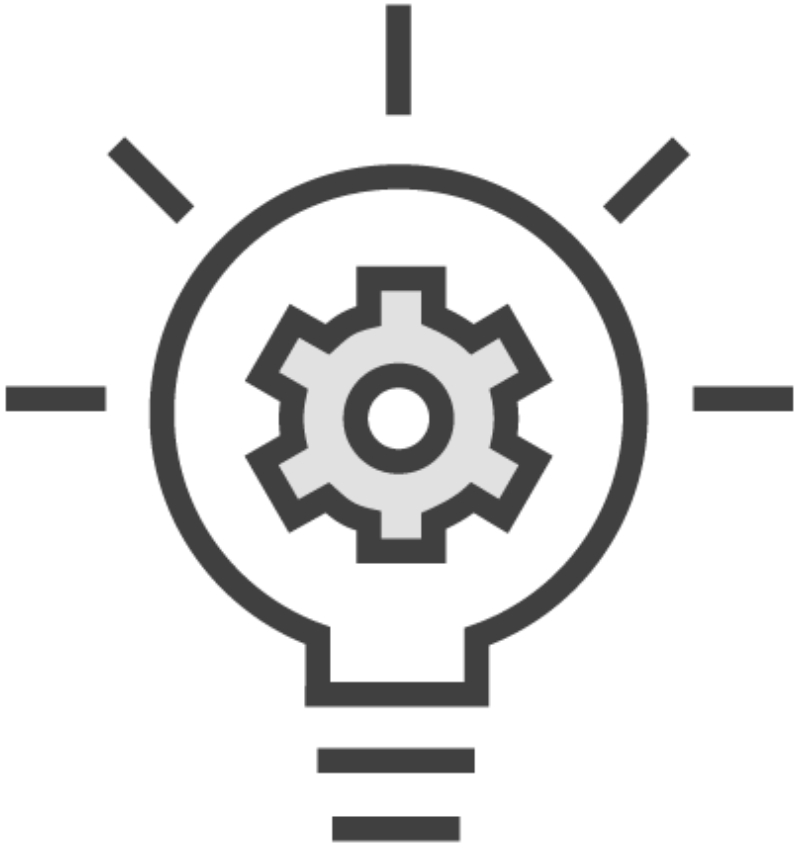
Aim is to have a stable size

Server configuration options

- Max_server_memory
- Min_server_memory

sys.dm_os_buffer_descriptors view

Memory Management and Performance



KB2663912 to read

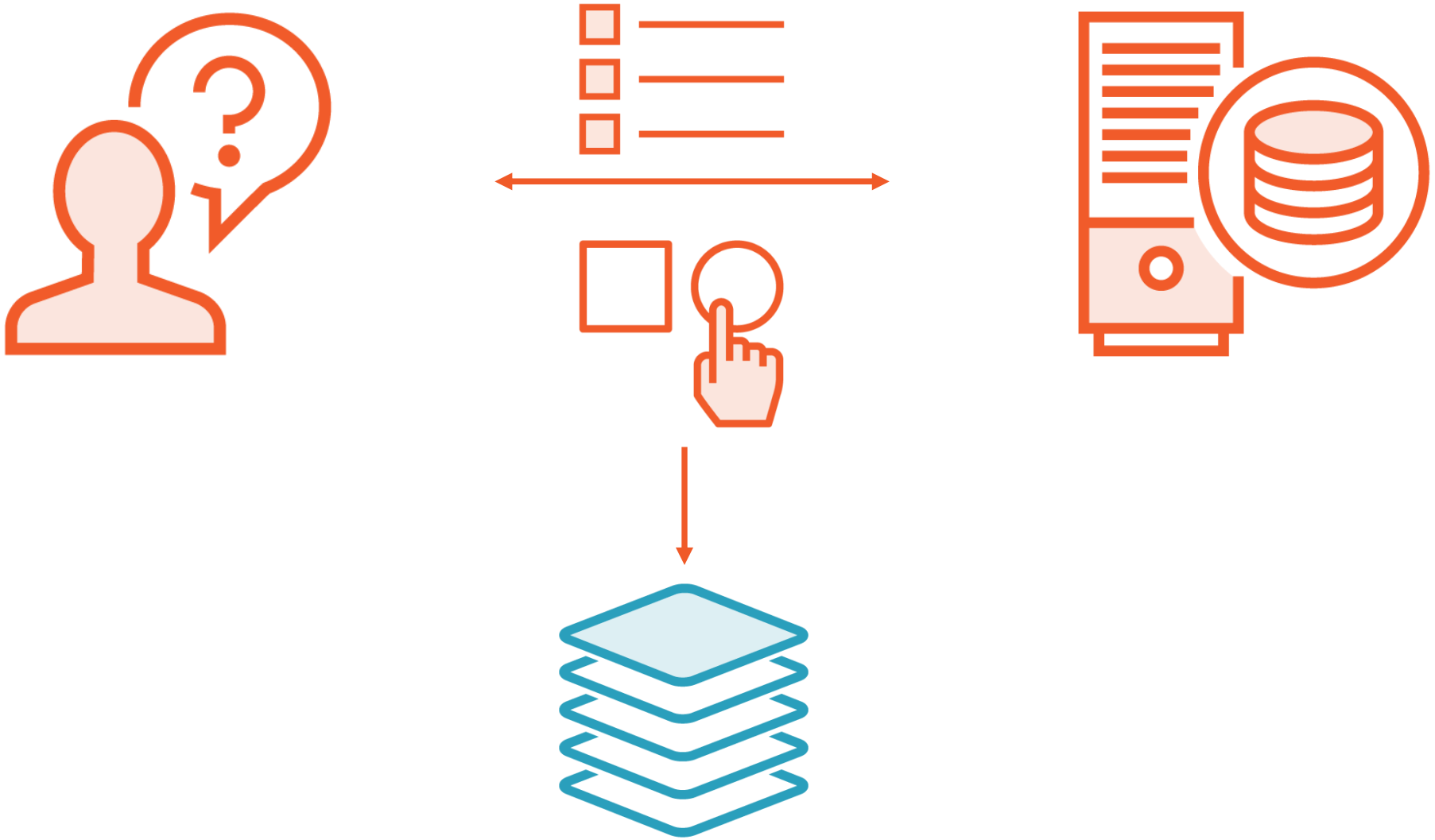
SQL Server uses many other types of memory allocations

- Plan cache
- Connections
- Query executions
- External modules

sys.dm_os_memory_clerks view

SQLOS: Wait Statistics

Hey SQL Server, What Is the Problem?



Scheduling Types

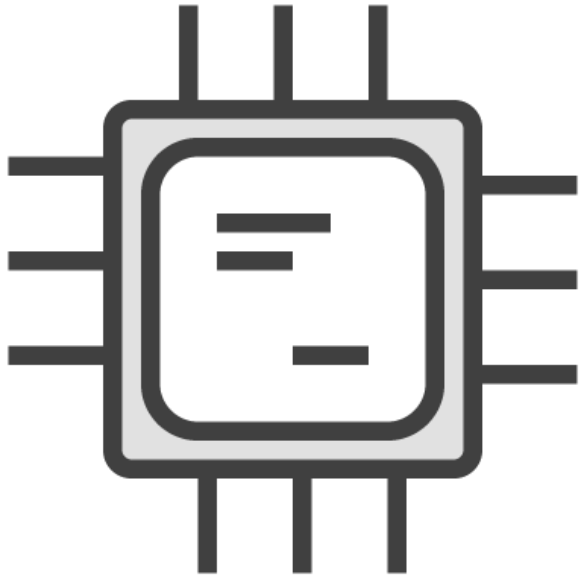
Windows OS

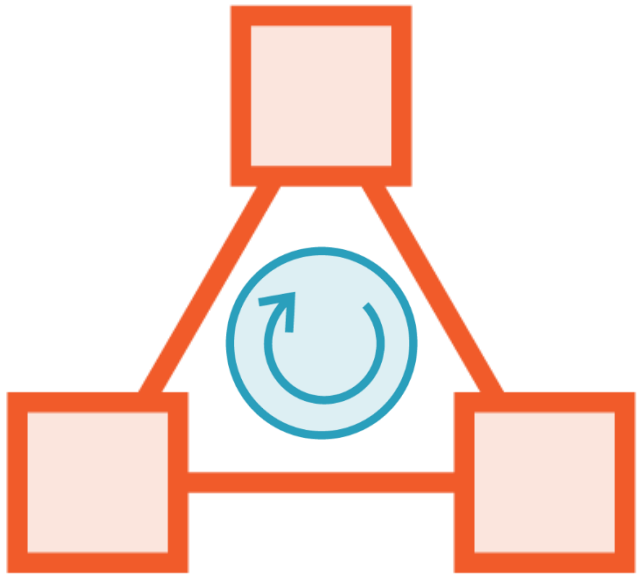
Preemptive

SQL Server

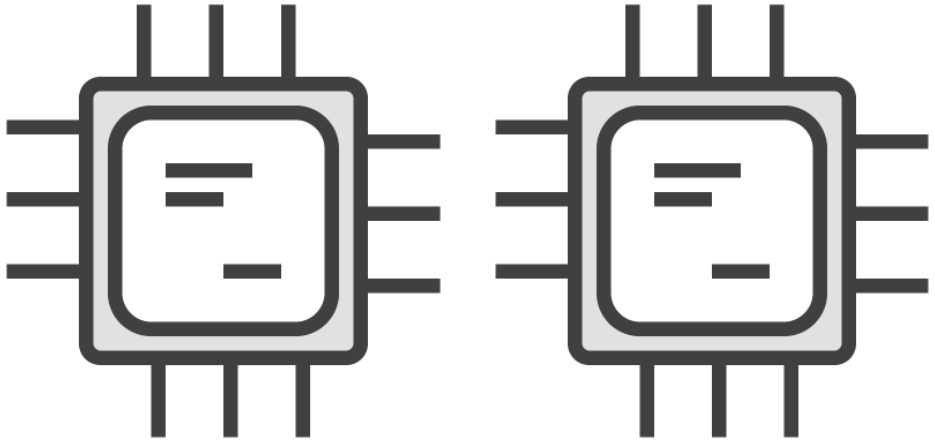
Non-preemptive

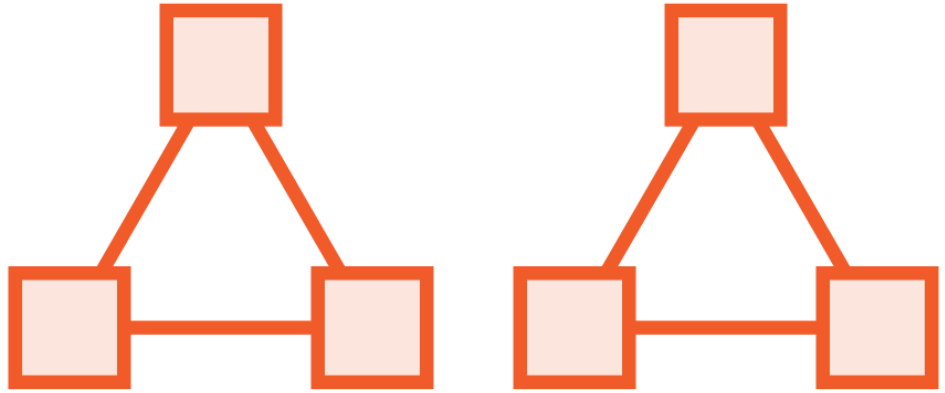
SQL Server Scheduler: Overview



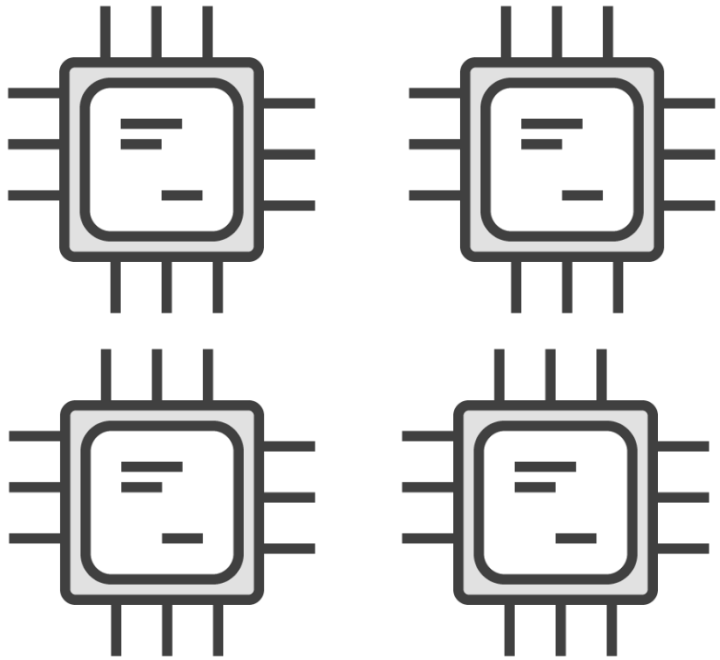


SQL Server Scheduler: Overview

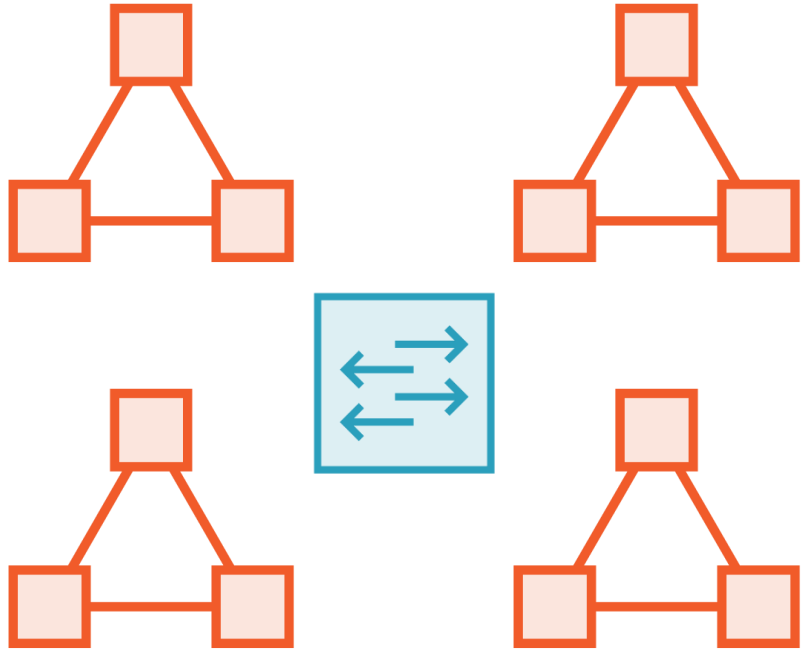




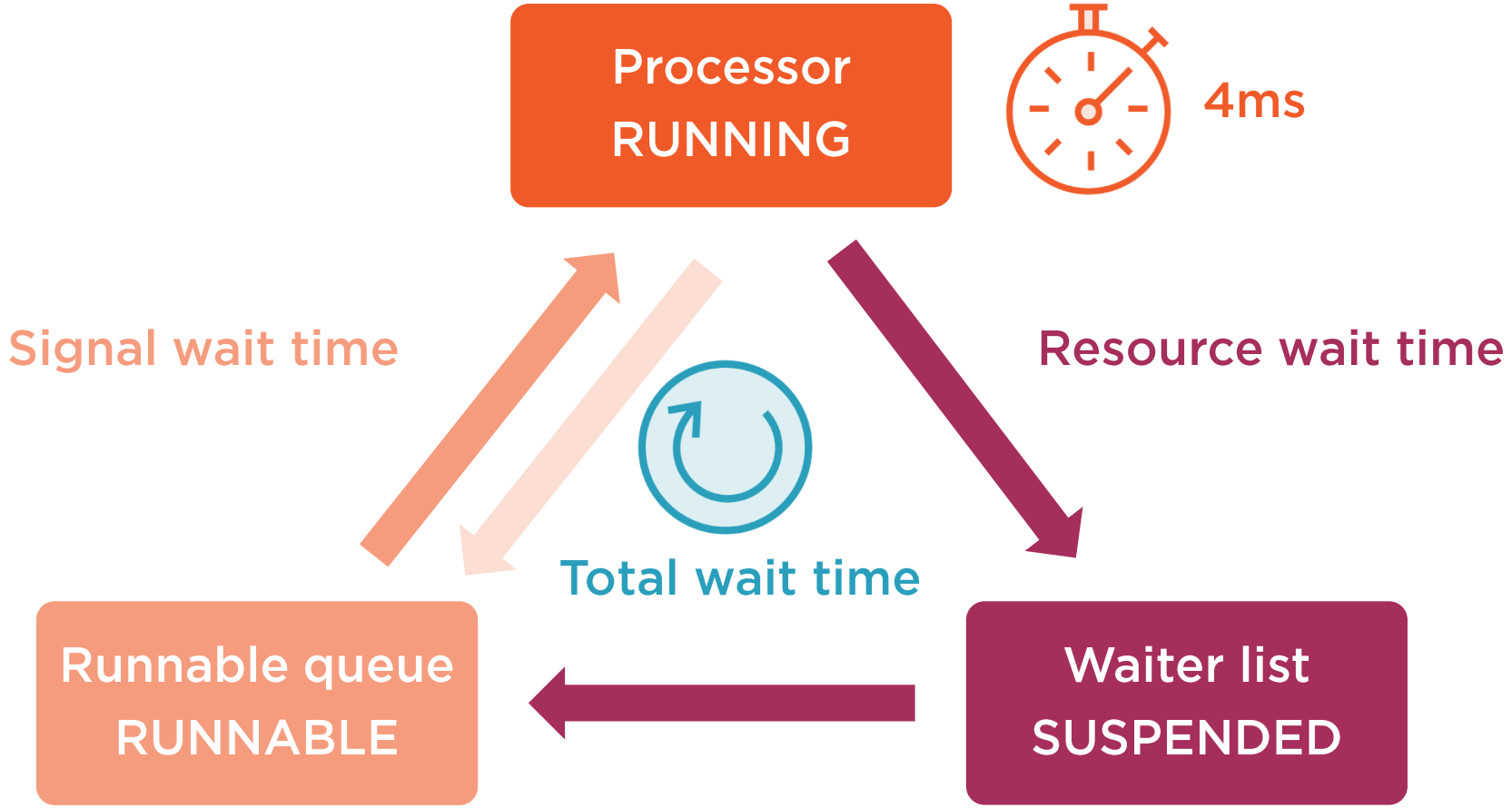
SQL Server Scheduler: Overview



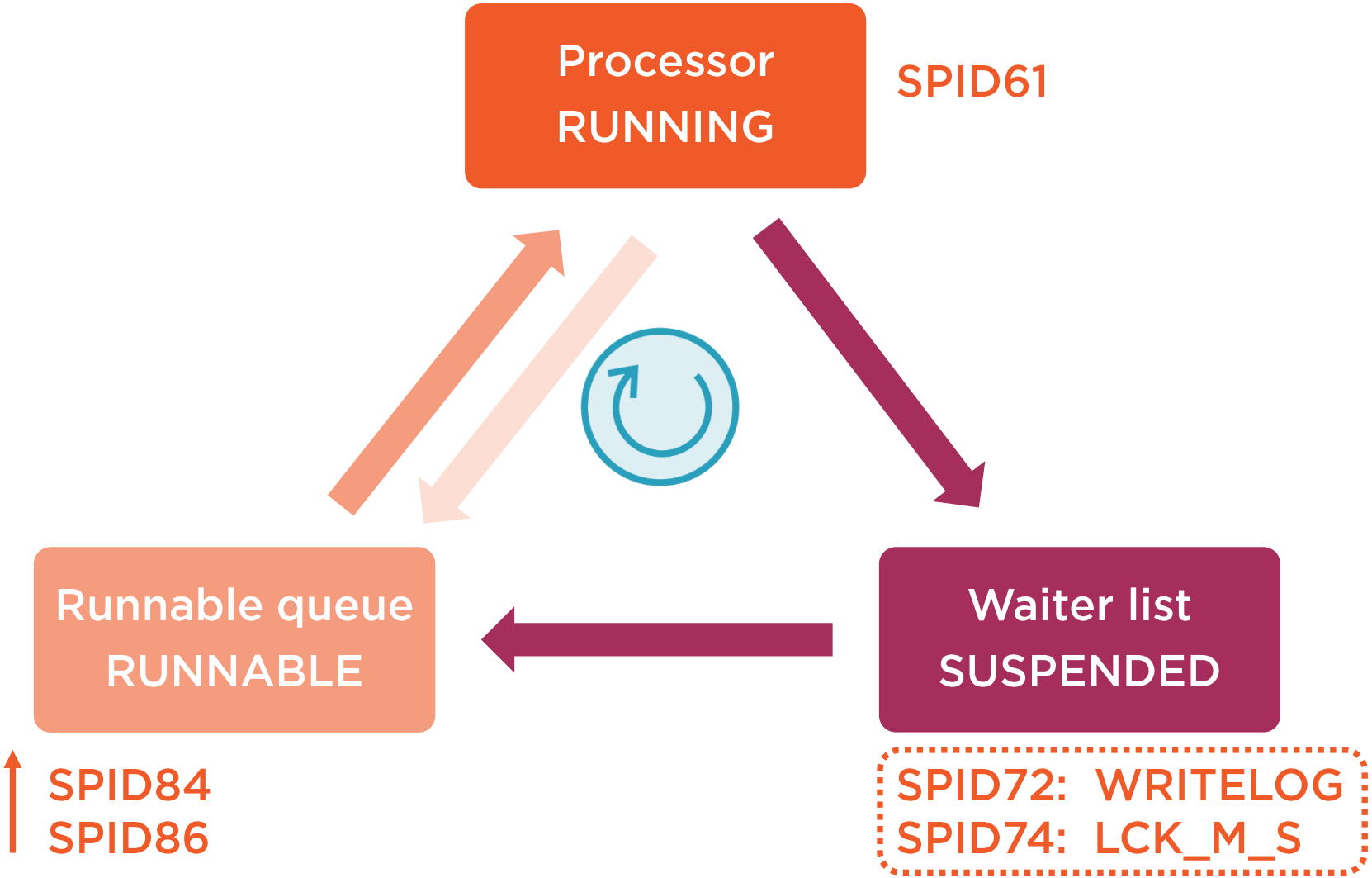
sys.dm_os_schedulers view

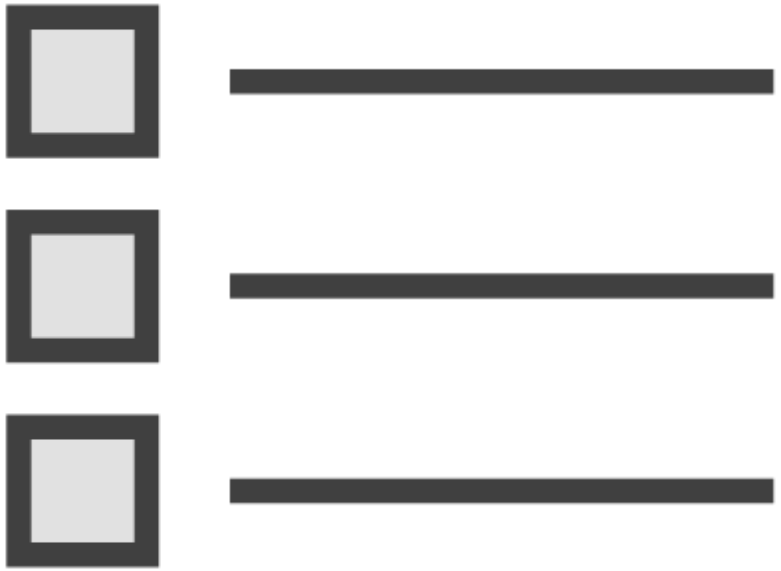


Threading Model and Wait Times



Threading Model and Wait Types

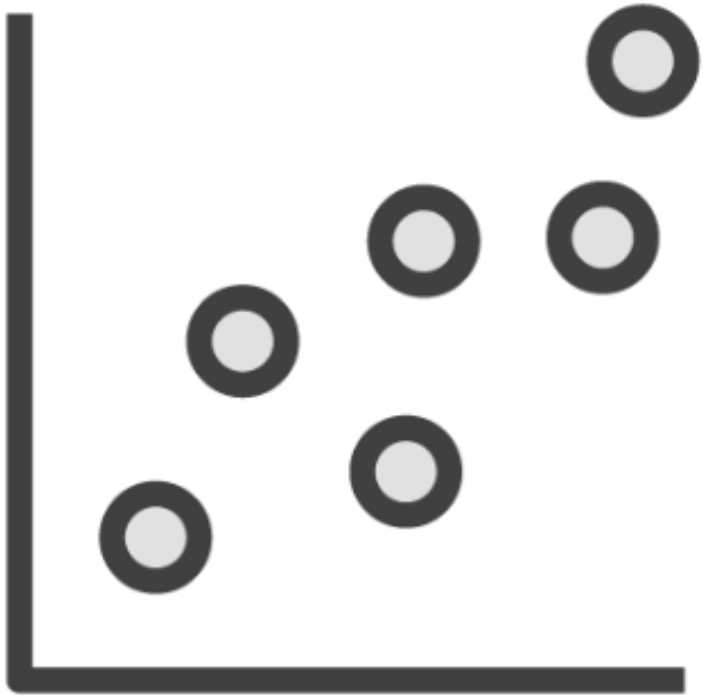




SUSPENDED: The waiter list

- Multiple SPIDs waiting
- No order
- Wait type assigned
- Resource wait time calculated

sys.dm_os_waiting_tasks view

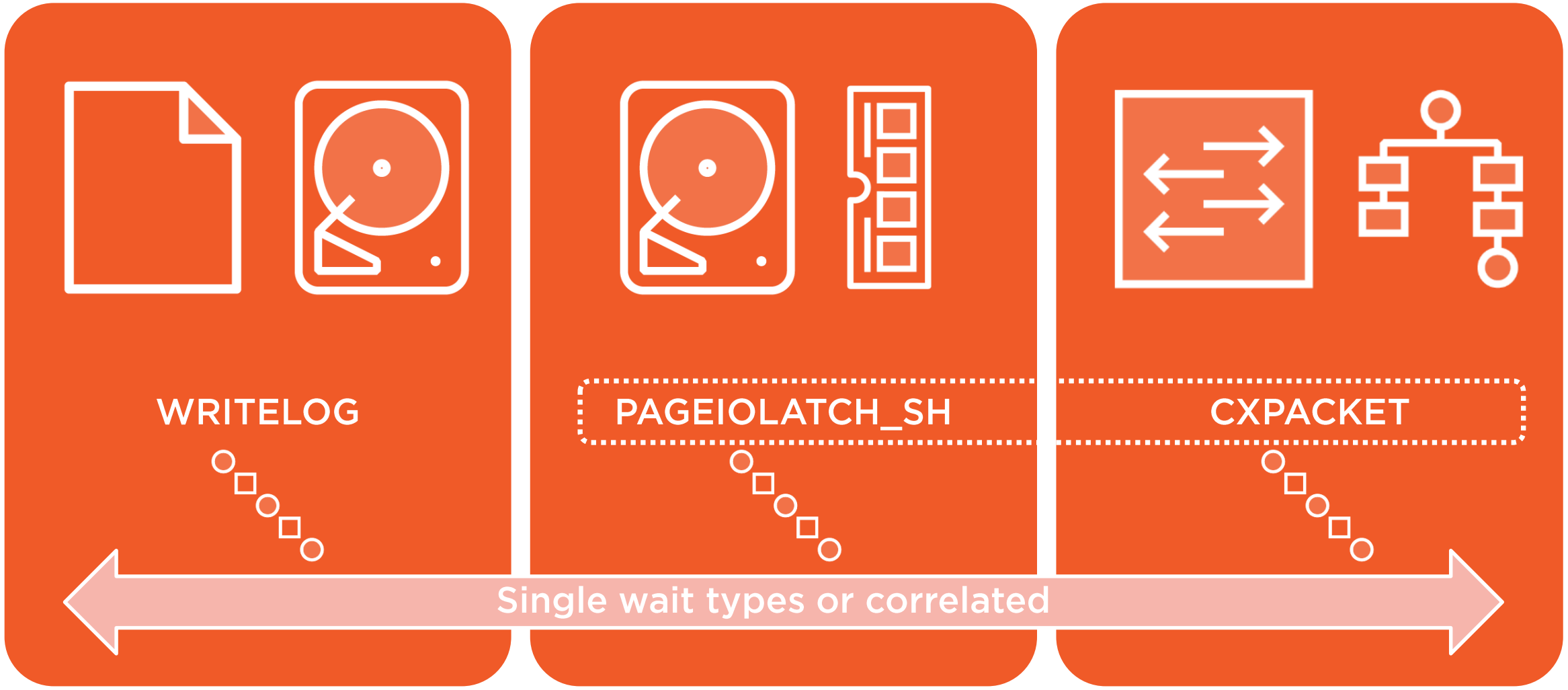


Wait statistics

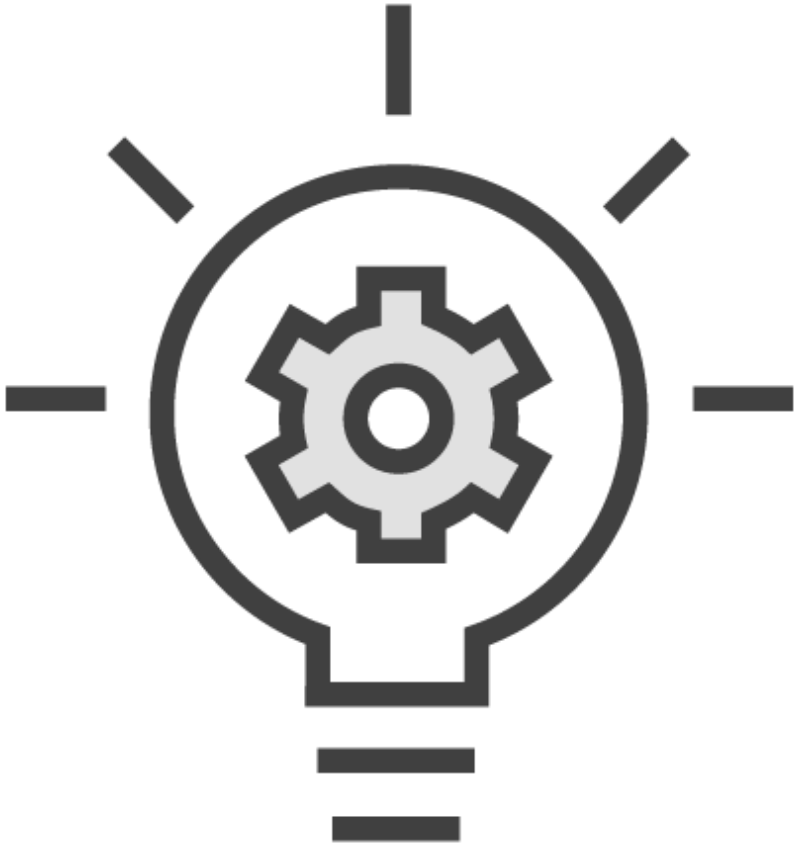
- Server level
- Database level (Azure SQL Database)
- Query level (Query Store 2017)
- Custom (eg: session level)

sys.dm_os_wait_stats view

Troubleshooting Patterns and Wait Types



Wait Statistics and Performance



Troubleshooting and optimization entry points

We know what SQL Server waits for and for how long

- Wait counts
- Wait types
- Maximum and average wait times

Wait types

- https://www.sqlskills.com/help/waits/
- https://bit.ly/2UZG7tm

Wait statistics is the basis
for SQL Server performance
troubleshooting and optimization

Summary

- SQL Server versions, editions, and patching
- Server instances
- Server and database configuration options
- Trace flags
- Tempdb
- Recovery models and the transaction log
- Memory management
- Threading model and wait statistics