



Monitoring And Troubleshooting

Module 2

Learning Units covered in this Module

- Lesson 1: Monitoring Tools
- Lesson 2: Waits Methodology To Monitor Performance
- Lesson 3: Execution Plans
- Lesson 4: Query Store

Lesson 1: Monitoring Tools

Objectives

After completing this learning, you will be able to:

- Explain various tools that can be used to monitor SQL Server administration tasks and performance.
 - Dynamic Management Views
 - Activity Monitor
 - Management Studio Built In Reports
 - Performance Dashboard Reports
 - Policy Based Management
 - Performance Monitor
 - PAL
 - SQL Traces And Extended Events
 - SQL Server Error Logs



What to Monitor

CPU

Memory

Disk IO

Network

Error and Event
Logs

Query Duration

Waits

Blocking

Activity Monitor

Overview

Processes

Resource Waits

Recent Expensive Queries

- Show Execution Plan

Active Expensive Queries

- Show Execution Plan and Live Query Plan

Dynamic Management Views (DMVs)

Can be used to monitor the health of SQL Server instances, identify optimization opportunities, and troubleshoot various kinds of problems

Expose Memory Structures of SQL Server

Not saved to database files on disk

Exist at server and database-level scope

Require VIEW SERVER STATE and VIEW DATABASE STATE permission

Dynamic Management Views (DMVs)

Viewing user connections

- Sys.dm_exec_connections
- Sys.dm_exec_sessions
- Sys.dm_exec_requests

Identifying bottlenecks

- Sys.dm_os_wait_stats
- Sys.dm_os_waiting_tasks

Identifying costly Queries

- Sys.dm_exec_query_stats
- Sys.dm_exec_sql_text
- Sys.dm_exec_query_plan

Management Studio Built In Reports

Server Reports

- Server Dashboard
- Configuration Changes History
- Schema Changes History
- Scheduler Health
- Memory Consumption
- Current Activity Reports
- Performance Reports
- Download Performance Dashboard Reports

Database Reports

- Disk Usage Reports
- Database Consistency History
- Index Statistics Reports
- User Statistics

Login Reports

- Login Statistics
- Login Failures
- Resource Locking Statistics by Logins

Management Reports

- Tasks
- Number of Errors

SQL Server Agent Reports

- Job Steps Execution History
- Top Jobs

Performance Dashboard Reports

Custom downloadable reports available to visualize SQL DMVs

Common performance problems that the dashboard reports may help to resolve include:

- CPU bottlenecks (and what queries are consuming the most CPU)
- IO bottlenecks (and what queries are performing the most IO)
- Index recommendations generated by the query optimizer (missing indexes)
- Blocking
- Latch contention

Host reports on SSRS Report Server

- Reports accessible anytime, anywhere using a browser
- Schedule automatic report delivery
- Performance Base lining using Report Snapshots
- Linked Reports for Application owners and other stakeholders

Performance Monitor Counters

Resource limits must be investigated in proper order:

- Memory
- Disk I/O
- Processor
- Network I/O

Limits in one resource can mask problems in another resource

Enables you to gather evidence to:

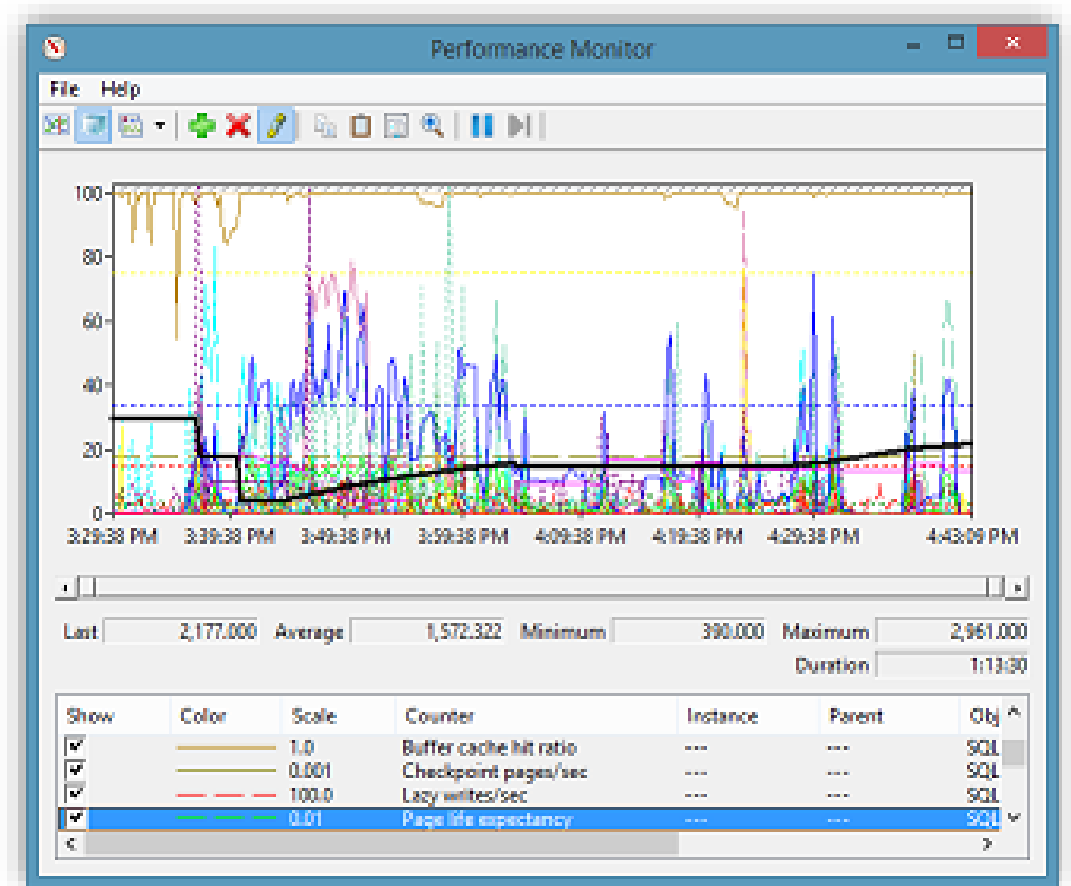
- Support your hypothesis
- Disprove theories so you can move on to other areas

Capturing overall state of the computer

- Enables you to find out WHAT happened rather than WHY or HOW

Capturing changes in the state of the computer

- Having a baseline is important for comparison purposes; enables you to find out what is "normal" and what is not



Profiler Traces and Extended Events

Profiler Trace

- Monitor SQL Server events
- Can be used to:
 - Monitor SQL Server activities
 - Tune performance
 - Diagnose problems
 - Debug applications
 - Audit
 - Replay trace to simulate and reproduce problems
- Requires ALTER TRACE permissions to run
- Tracing introduces overhead
- Default Traces

Extended Events

- SQL Trace is now Deprecated
 - Still works in SQL Server 2022
 - May be removed in a future version
- Much more lightweight than SQL Trace
 - Should NOT hinder customer workload
- Much more capability than SQL Trace
 - You can capture page split events with xEvents
 - Many different targets for storing event data
- Management Studio provides a GUI for defining session and viewing Extended Events data
- System Health Session

SQL Server Error Logs

Wealth of information about the SQL Server

Monitor and review logs

Location

- Program Files\Microsoft SQL Server\MSSQL.n\MSSQL\LOG\ERRORLOG and ERRORLOG.n
- Confirm in SQL Server Configuration Manager -> Right Click on SQL Server Service -> Startup Parameters -> -e parameter

Error Log Management

- Cycling the log
- Maximum number of error log files
- Controlling the size of the log

sp_readerrorlog

Interesting messages in SQL Server error logs

- SQL IO taking longer than 15 seconds, Paging of SQL Server Buffer Pool memory, crash dumps, Access Violation, Availability Group Failover messages, database corruption errors etc.

Demonstration

Built-In Reports and XEvents

- In-built reports in SSMS
- Create in Xevent session through wizard in SSMS
- How to configure Error Logs



Extended Events

- Examining Extended Events



Questions?



Knowledge Check

What tool will you use to find missing index information?

Which tool will you use to identify the process or the application that is responsible for high CPU on the server?

What tool can you use to open extended event files and view event data?

You are seeing performance issues on the server and want to know the queries which are currently running and may be causing the issue? Which tool can you use?

True/False? You can view the execution plan of the recent or currently executing query using Activity Monitor?

What tool will you use to collect data to analyze with SQLNexus

Lesson 2: Waits Methodology To Monitor Performance

Objectives

After completing this learning, you will be able to:

- Explain how to use waits stats and related DMVs to troubleshoot performance issues in SQL Server.
- Explain Task Execution Model.
- Describe Waits and Queues.
- List common Wait types.



Waits and Queues Methodology

Troubleshooting methodology to identify resource bottlenecks

Valuable in identifying performance trends overtime

Provides information on resources SQL Server is regularly waiting on

Useful for workload measurements and benchmarking

SQL Server tracks Wait Statistics over time in DMVs

SQL Server tracks three types of waits:

- Resource waits (for example locks, latches, network, and disk I/O waits)
- Queue waits (idle worker waiting for work)
- External waits (waiting for an external event, extended stored procedure, and linked server query)

Impact of Waits on Query Response Time

Running

- Only one session can be running or executing per scheduler at a time

Runnable

- Sessions waiting for CPU. Next SPID in the runnable queue is scheduled to start running

Suspended

- Sessions wait in the waiter list until resources become available

Resource
Wait Time
(SUSPENDED)



Signal Wait
Time
(RUNNABLE)



CPU Time
(RUNNING)



Total Query
Response
Time

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

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

SPID56 moved to the bottom of the Runnable queue.

Notable Wait Types

PAGELATCH_xx

PAGEIOLATCH_xx

ASYNC_IO_COMPLETION

SOS_SCHEDULER_YIELD

CXPACKET

THREADPOOL

WRITELOG

LCK_XX

CMEMTHREAD

RESOURCE_SEMAPHORE

Wait Stats DMVs

sys.dm_os_wait_stats

- Returns cumulative 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.

sys.dm_exec_session_wait_stats

- Returns cumulative information about all the waits encountered by a session.

Questions?



Knowledge Check

What resource bottleneck does RESOURCE_SEMAPHORE wait type indicate?

Which DMV can be used to return information on tasks/requests that are currently waiting on a resource?

What is the difference between PAGELATCH and PAGEIOLATCH wait types

A query took 60 seconds to complete but when you check the CPU time consumed by the query in Profiler, it is less than 5 seconds. Where did the query spend the rest of 55 seconds?

Which DMV contains the wait stats information for an instance of SQL Server since the last server restart?

Lesson 3: Execution Plans

Objectives

After completing this learning, you will be able to:

- Explain Reading Execution Plans
- Explain Difference between Estimated vs Actual Execution Plans
- Explain the Live Query Statistics feature.



How to see the query plan

Graphical execution plan

Estimated Execution Plan (Before Execution)

- The compiled plan.

Actual Execution Plan (After Execution)

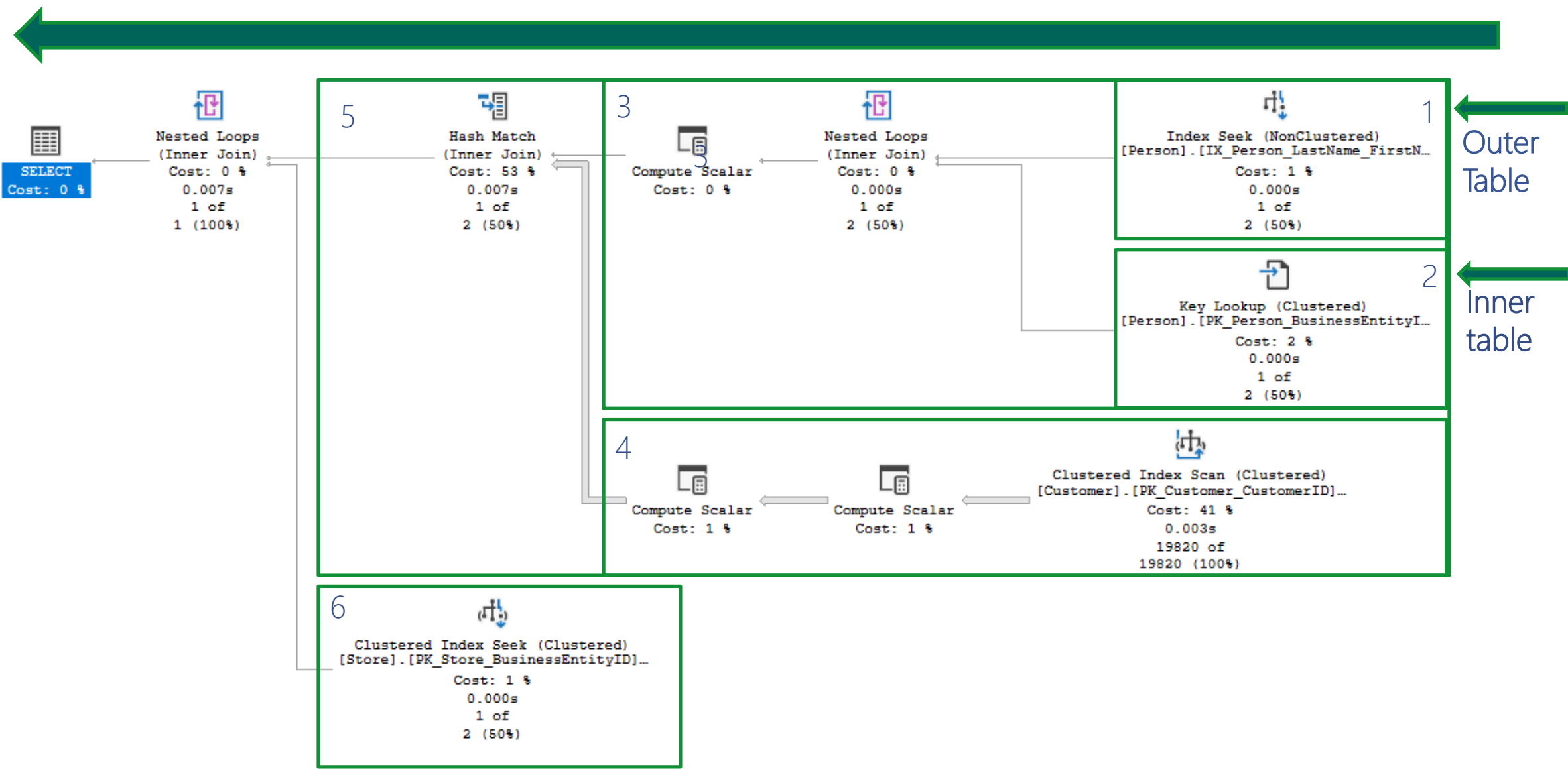
- The same as the compiled plan plus its execution context.
- This includes runtime information available after the execution completes, such as execution warnings, or in newer versions of the Database Engine, the elapsed and CPU time used during execution.

Live Query Statistics (During Execution)

- The same as the compiled plan plus its execution context.
- This includes runtime information during execution progress and is updated every second. Runtime information includes for example the actual number of rows flowing through the operators.
- Enables rapid identification of potential bottlenecks.

SSMS Graphical Plan

Execution Flow



Live Query Statistics

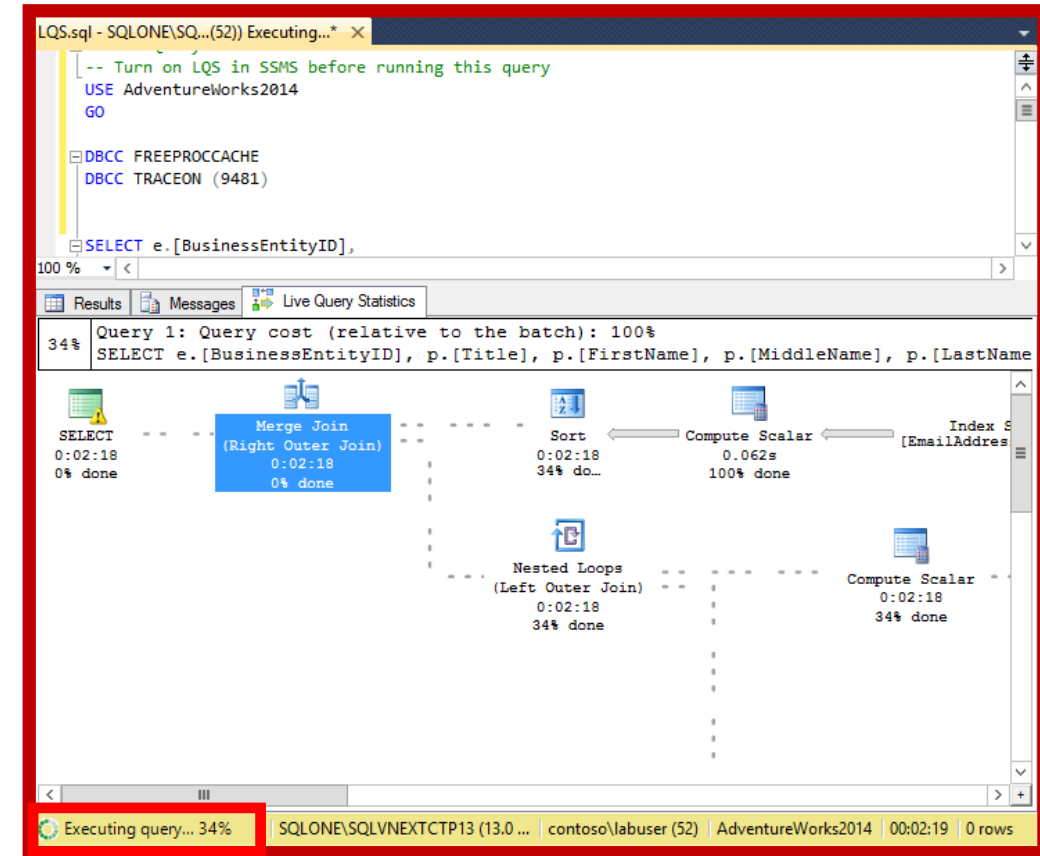
Introduction

View real time CPU, I/O, execution time and query progress

Information available in the sys.dm_exec_query_profiles DMV

Accessible via SSMS query window or Activity Monitor

Slight performance overhead to monitor live statistics



Live Query Statistics

How To Enable

SQL Server Management Studio (SSMS):

- Query menu -> Include Live Query Statistics
- Right-click on the query in query window -> Include Live Query Statistics

Extended Events

- Enable query_post_execution_showplan extended event

Sessions

- SET STATISTICS XML ON or SET STATISTICS PROFILE ON in the target session

Activity Monitor

- Right-click a session and select Show Live Execution Plan
- Right click the query under "Active Expensive Queries" pane and select Show Live Execution Plan
- Session must have SET STATISTICS PROFILE/XML ON or the extended event for showplan must be turned on

Demonstration

- Live Query Statistics



Questions?



Knowledge Check

True/False? Trace Flag 7412 is required to enable Lightweight query statistics profiling in SQL Server 2019?

How can you enable/disable Lightweight query statistics profiling in SQL Server 2019?

What is the requirement to be able to see the Live Query Statistics for a query/session in Activity Monitor?

What new DMF was added in SQL Server 2019 to return the equivalent of the last known actual execution plan for most queries?

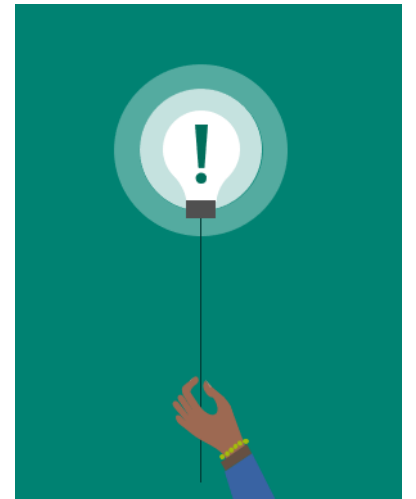
Can you see the Live Query Statistics for a query that just finished but not executing anymore?

Lesson 4: Query Store

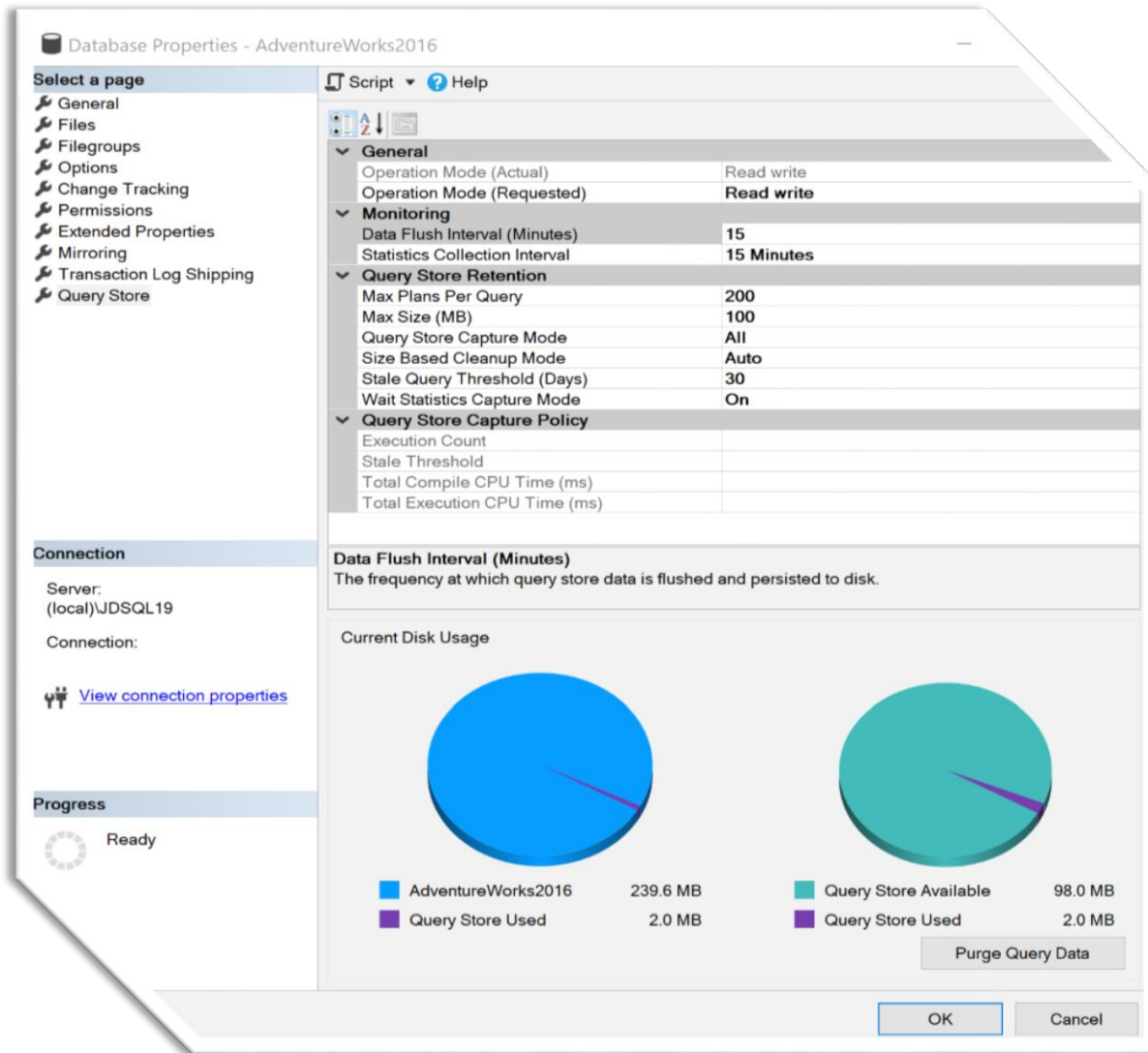
Objectives

After completing this learning, you will be able to:

- Explain the importance of the Query Store feature in SQL Server to monitor performance of queries and stored procedures.
 - Introduction
 - Usage Scenarios
- How does the Query Store work?
- Explain scenarios where Query Store can be a helpful tool
- What is new?



Introducing the Query Store



Query Store is set at the database level

Cannot be used for Master or TempDB system databases but can be enabled for the Model and MSDB system databases.

The user database stores the data in internal tables that can be accessed by using built-in Query Store views.

SQL Server retains this data until the space allocated to Query Store is full or manually purged.

Why use Query Store?

Before Query Store

- Requires manual proactive monitoring to identify execution plan problems.
- Only the latest plan was stored in the procedure cache
- Restart caused data to be lost
- Frequent recompiles of procedures or use of DBCC FREEPROCACHE
- No history or aggregated gathering of data available.

With Query Store

- It stores the history of the execution plans for each query
- It establishes a performance baseline for each plan over time
- It identifies queries that may have regressed
- It is possible to force plans quickly and easily
- It works across server restarts, upgrades, and query recompilation

Query Store

Usage Scenarios

Pinpoint and fix queries with plan choice regressions

- Audit the history of query plans for a given query

Force a specific query plan easily without need of plan guides

Identify and tune top resource consuming queries

- A baseline for the performance of each plan over time (CPU and IO consumption for each plan)

Upgrade testing

- keep performance stability during the upgrade to newer SQL Server

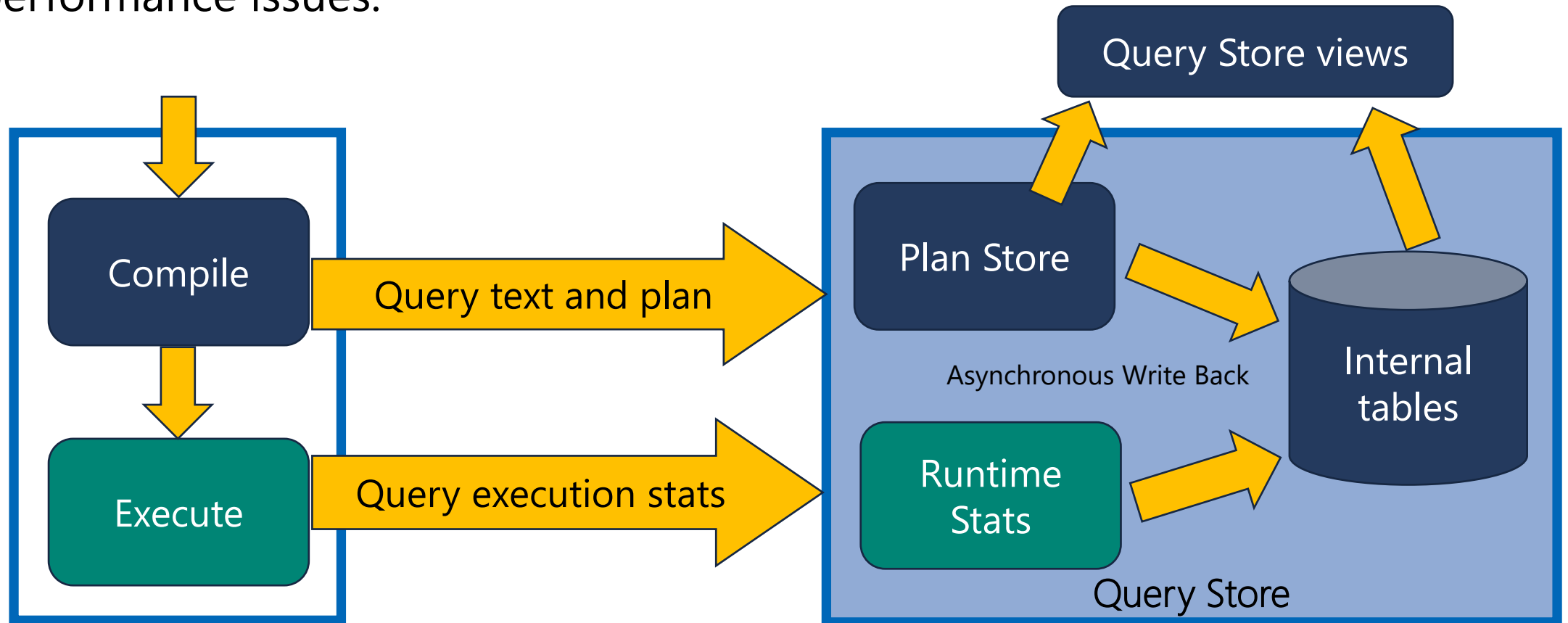
Identify and improve ad hoc workloads

Captures wait statistics

Query Store

Introduction

- Query Store persists execution plans and runtime statistics per database.
- Graphical interface allows you to quickly and easily troubleshoot query performance issues.



Query Store

What Is Captured By Query Store?

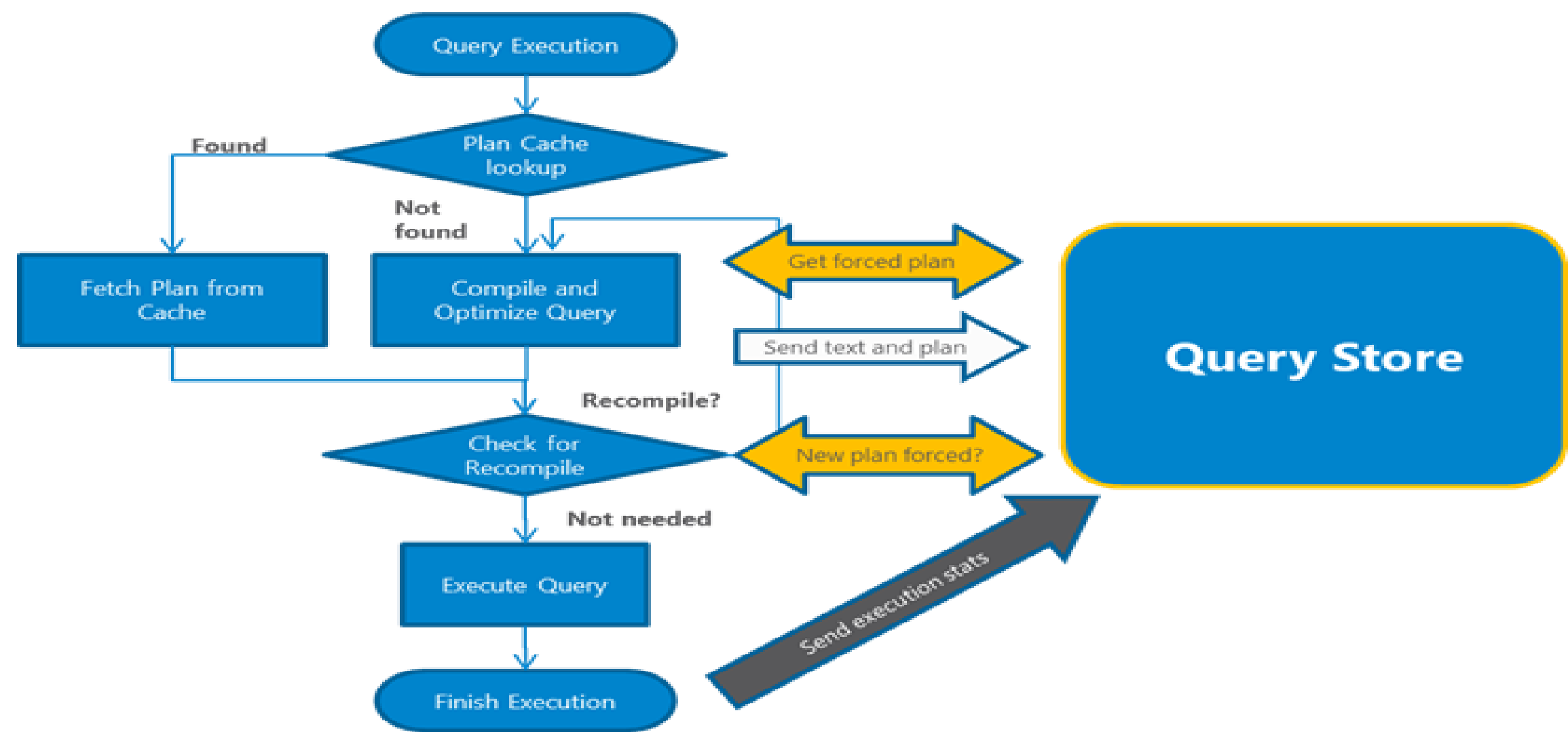
Query texts

Query plans

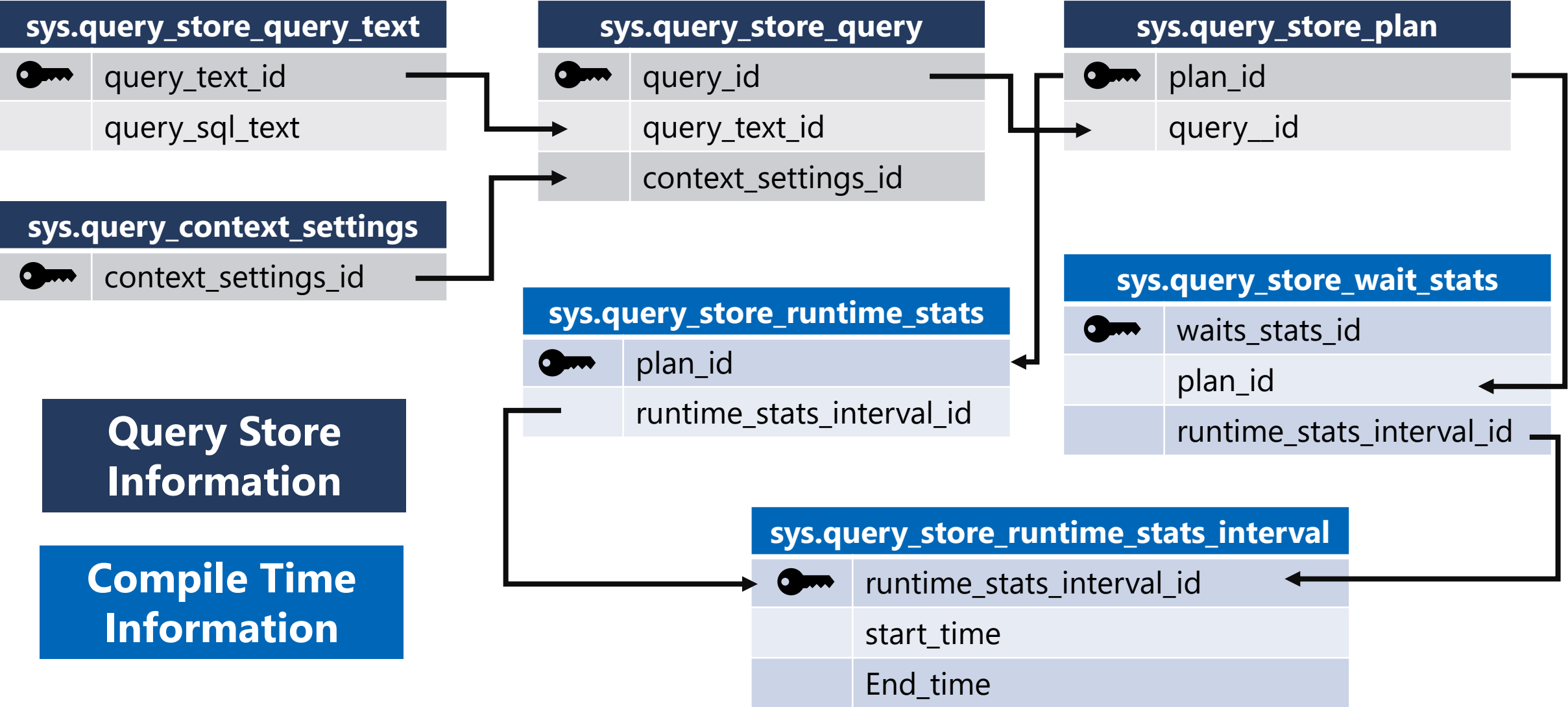
Runtime statistics

- Aggregated in selected time intervals
- Count of executions of each captured plan
- Metrics: duration, cpu_time, logical_io_reads, DOP, query_max_used_memory, and rowcount
- Waits per query

Query Store Data Storage



Query Store Catalog Views



Query Store

Wait types In Query Store

Waits are captured per query execution

- What queries are causing blocking
- For my most expensive queries, how much time is spent on CPU vs waiting?

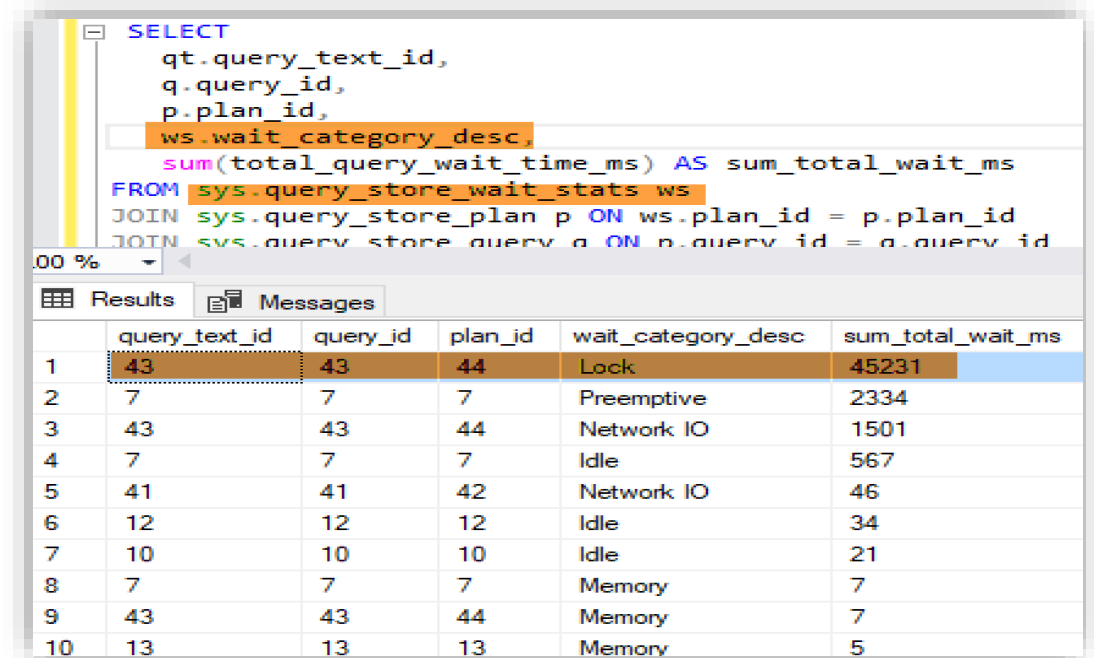
Combined into Wait Categories

Categories group similar wait types to simplify reporting and troubleshooting

Group examples:

- Lock – LCK_M_S, LCK_M_X, LCK_M_SCH_S, etc
- Buffer IO – pageiolatch_sh, pageiolatch_ex, etc

Stored in sys.query_store_wait_stats



The screenshot shows a SQL query in the 'Query' tab of SQL Server Enterprise Manager. The query is a SELECT statement that joins sys.query_store_wait_stats (ws), sys.query_store_plan (p), and sys.query_store_query (q) to retrieve wait statistics. The results are displayed in the 'Results' tab as a table with 10 rows. The first row is highlighted in blue, indicating it is the current row. The columns are: query_text_id, query_id, plan_id, wait_category_desc, and sum_total_wait_ms.

```
SELECT
    qt.query_text_id,
    q.query_id,
    p.plan_id,
    ws.wait_category_desc,
    sum(total_query_wait_time_ms) AS sum_total_wait_ms
FROM sys.query_store_wait_stats ws
JOIN sys.query_store_plan p ON ws.plan_id = p.plan_id
JOIN sys.query_store_query q ON p.query_id = q.query_id
```

	query_text_id	query_id	plan_id	wait_category_desc	sum_total_wait_ms
1	43	43	44	Lock	45231
2	7	7	7	Preemptive	2334
3	43	43	44	Network IO	1501
4	7	7	7	Idle	567
5	41	41	42	Network IO	46
6	12	12	12	Idle	34
7	10	10	10	Idle	21
8	7	7	7	Memory	7
9	43	43	44	Memory	7
10	13	13	13	Memory	5

Query Store Reports

Regressed Queries

Overall Resource Consumption










Top Resource Consuming Queries

Queries With Forced Plans

Queries With High Variation

Query Wait Statistics

Tracked Queries

-   Query Store
 -  Regressed Queries
 -  Overall Resource Consumption
 -  Top Resource Consuming Queries
 -  Queries With Forced Plans
 -  Queries With High Variation
 -  Query Wait Statistics
 -  Tracked Queries

Query Store Configuration Options

General and Monitoring

Operation Mode (Actual)

Operation Mode (Requested)

Data Flush Interval (Minutes)

Statistics Collection Interval

Database Properties - AdventureWorks2016

Select a page: General, Files, Filegroups, Options, Change Tracking, Permissions, Extended Properties, Mirroring, Transaction Log Shipping, Query Store

Script Help

General	
Operation Mode (Actual)	Read write
Operation Mode (Requested)	Read write

Monitoring	
Data Flush Interval (Minutes)	15
Statistics Collection Interval	15 Minutes

Query Store Retention	
Max Plans Per Query	200
Max Size (MB)	100
Query Store Capture Mode	All
Size Based Cleanup Mode	Auto
Stale Query Threshold (Days)	30
Wait Statistics Capture Mode	On

Query Store Capture Policy	
Execution Count	
Stale Threshold	
Total Compile CPU Time (ms)	
Total Execution CPU Time (ms)	

Connection

Server: (local)\JDSQL19

Connection:


[View connection properties](#)

Progress

Ready

Data Flush Interval (Minutes)
The frequency at which query store data is flushed and persisted to disk.

Current Disk Usage



AdventureWorks2016	239.6 MB	Query Store Available	98.0 MB
Query Store Used	2.0 MB	Query Store Used	2.0 MB

Purge Query Data

OK Cancel

Query Store Configuration Options

Retention Settings

Max Plans Per Query

Max Size (MB)

Query Store Capture Mode

Size Based Cleanup Mode

Stale Query Threshold (Days)

Wait Statistics Capture Mode

Database Properties - AdventureWorks2016

Select a page: General, Files, Filegroups, Options, Change Tracking, Permissions, Extended Properties, Mirroring, Transaction Log Shipping, Query Store

Script Help

General

Operation Mode (Actual)	Read write
Operation Mode (Requested)	Read write

Monitoring

Data Flush Interval (Minutes)	15
Statistics Collection Interval	15 Minutes

Query Store Retention

Max Plans Per Query	200
Max Size (MB)	100
Query Store Capture Mode	All
Size Based Cleanup Mode	Auto
Stale Query Threshold (Days)	30
Wait Statistics Capture Mode	On

Query Store Capture Policy

Execution Count	
Stale Threshold	
Total Compile CPU Time (ms)	
Total Execution CPU Time (ms)	

Connection

Server: (local)\JDSQL19

Connection:

[View connection properties](#)

Progress

Ready

Data Flush Interval (Minutes)

The frequency at which query store data is flushed and persisted to disk.

Current Disk Usage

AdventureWorks2016	239.6 MB	Query Store Available	98.0 MB
Query Store Used	2.0 MB	Query Store Used	2.0 MB

Purge Query Data

OK Cancel

Query Store Configuration Options

Capture Policy Settings

Execution Count

Stale Threshold

Total Compile CPU Time

Total Execution CPU Time

Database Properties - AdventureWorks2016

Select a page: General, Files, Filegroups, Options, Change Tracking, Permissions, Extended Properties, Mirroring, Transaction Log Shipping, Query Store

Script Help

General

Operation Mode (Actual)	Read write
Operation Mode (Requested)	Read write

Monitoring

Data Flush Interval (Minutes)	15
Statistics Collection Interval	15 Minutes

Query Store Retention

Max Plans Per Query	200
Max Size (MB)	100
Query Store Capture Mode	All
Size Based Cleanup Mode	Auto
Stale Query Threshold (Days)	30
Wait Statistics Capture Mode	On

Query Store Capture Policy

Execution Count	
Stale Threshold	
Total Compile CPU Time (ms)	
Total Execution CPU Time (ms)	

Connection

Server: (local)\JDSQL19

Connection: [View connection properties](#)

Progress

Ready

Data Flush Interval (Minutes)

The frequency at which query store data is flushed and persisted to disk.

Current Disk Usage

AdventureWorks2016	239.6 MB	Query Store Available	98.0 MB
Query Store Used	2.0 MB	Query Store Used	2.0 MB

Purge Query Data

OK Cancel

Query Store

Considerations and Best Practices

Use latest version of SQL Server Management Studio

Use ALTER to modify objects, otherwise duplicate queries will be tracked

For Ad hoc workloads use capture mode = Auto

Query store will revert to read-only mode when:

- The database is not in a state to allow writing such as emergency, read-only or single user
- Reaches the maximum storage size

Avoid renaming databases if you have queries with Forced Plans

- If you restore the database with a different database name, forced plans may fail

Failure forcing plans

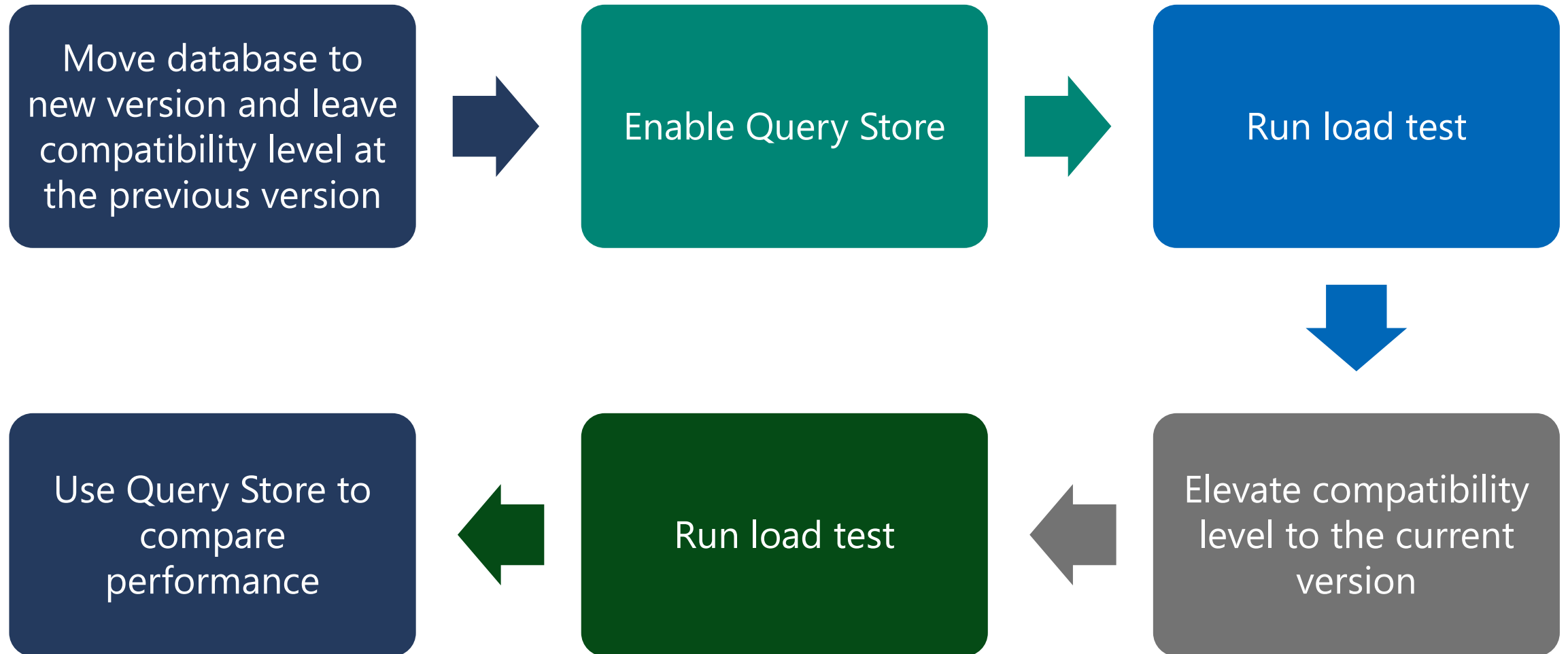
- Query still compiles and executes

Use Query Store Trace Flags on mission critical servers

- Trace flags 7745 and 7752

Query Store

How To Use Query Store During Migration and Upgrades



Query Store hints – Use Cases

Use Cases

- When code can't be changed
- Override other hints/plan guides
- Recompile a query on each execution.
- Cap the memory grant size for a bulk insert operation.
- Limit the maximum degree of parallelism when updating statistics.
- Use a Hash join instead of a Nested Loops join.
- Use [compatibility level](#) 110 for a specific query while keeping everything else in the database at compatibility level 150.

Query Store read replica support for Availability Groups

New feature in SQL Server 2022

Execution metrics for queries run on secondary replicas

Data is sent from secondaries back to the primary replica

Persisted in the primary replica's Query Store

You must enable trace flag 12606 before you can enable Query Store for secondary replicas.

Considerations

- Sharing bandwidth with outgoing transaction records
- A shared Query Store will be larger
- Impact of *ad hoc* workloads run on secondary replicas

Query Store

Analyzing Plan Regression with
Query Store Reports



Questions?



Knowledge Check

Which version of SQL Server introduced the ability to track query wait stats in Query Store?

Why are Query Store Trace Flags 7745 and 7752 recommended on mission critical servers?

Under what circumstances can Query Store revert to Read Only mode?

What option should we choose for the "Query Store Capture Mode" to ignore capturing infrequent queries and queries with insignificant compile and execution duration?

True\False? If Query Store is unable to force a query plan, the query will fail to execute?

