SQLskills Online Immersion Event

Solving Common Performance Problems Using Query Store

Erin Stellato
Erin@SQLskills.com

SQLskills
immerse yourself in sql server

Erin Stellato

Principal Consultant, SQLskills

Erin@SQLskills.com

@erinstellato

www.sqlskills.com/blogs/erin

Trainer/Speaker

In addition to consulting, I teach content for our IE0: Accidental DBA course, and our IEPTO2: Performance Tuning and Optimization course

PASS Volunteer

I am a member of the PASS Nomination Committee this year, have previously served on the board of my local user group (ONSSUG) and supported the Performance Virtual Chapter

Data Platform MVP

I have been fortunate to be recognized as an MVP by Microsoft since 2012

© SQLskills, All rights reserved.
http://www.SQLskills.com

Daily Course Format: Tue, Wed, and Th

- Class runs for three half days
  - Tuesday, May 4 through Thursday, May 6, 2021
- Each day
  - Lecture: 90 minutes from 9am until 10:30am PT
    - Please use "CHAT" for questions ***during*** the lecture
  - Open Q&A: 30 minutes from 10:30am until 11:00am PT
    - Will address unanswered questions from chat
  - Mandatory break: 30 minutes from 11:00am until 11:30am PT
    - Everyone needs a bit of a break!
  - Lecture: 90 minutes from 11:30am until 1:00pm PT
    - Please use "CHAT" for questions ***during*** the lecture
  - Open Q&A: up to 60 minutes from 1:00pm until 2:00pm PT
    - Will address unanswered questions from chat
    - Can open up for "open mic" questions

Time Conversions

9:00am PT

= Noon ET

= 4:00pm UTC

Use this to see the exact time in YOUR time zone:

Event Time Announcer - SQLskills IEQS (timeanddate.com)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Class Format: No LABS???

- Class time – lecture / demo / questions! And, please ASK questions!
  - There are no stupid questions… everyone started with ZERO knowledge about SQL Server!
  - If you're thinking it – you're probably not alone… speak up!
- Why not lab time?
  - Historically folks defer on labs to respond to email, etc.
    ⇒ losing valuable class time
  - We can spend more time on content
    ⇒ covering both MORE content and in more depth!
  - You receive all of the demo code and samples; you can reproduce everything we show you and pick where to really focus your attention!
    - You can take time to really study the content that's most appropriate to you
    - You can skip content that's not appropriate
    - You can take the RIGHT time for each/every exercise (rather than some being done quickly and others needing more time… potentially getting frustrated without enough time)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Course Resources, Pluralsight Access, and Alumni

- Send Paul (paul@sqlskills.com) an email for a FREE 30-day access to ALL SQLskills online training classes on Pluralsight
  - No strings attached, no credit-card required
  - If you don't receive it, send mail to Paul
- Course resources will be posted on the course membership page for class upon course completion
- We may want to update a slide or two, add some notes/resources based on questions and discussions, and process the recordings
- Email us: Training@SQLskills.com if you have any problems finding or accessing any of your course materials
- And, now that you've attended one of our courses; you are an alumnus in our system with discounts for additional training courses!

© SQLskills, All rights reserved.
http://www.SQLskills.com

Course Resources

- Course content (this slide deck)
  - One “two-slides per side” PDF for optimal printing (save some trees!)
  - One “full slide” PDF for optimal viewing and/or online note-taking
- Demos
  - All instructor-led demo scripts
  - All reference scripts
- Reference links and additional resources
- Online recordings for these sessions
- Upon completion don’t forget to request: “BadgeMe”

Please do not redistribute: These resources are for attendees only.
Please respect our IP and time in creating these resources
and do not share/forward.

© SQLskills, All rights reserved.
http://www.SQLskills.com

Recommendation:
Leverage Great Products!

SentryOne™

- Even with as much training as we will provide, you can't know everything
- SQL Server is powerful, complex, and has lots of different potential issues let third party software help!
- SQLskills partners with SentryOne because we really feel they've made the best suite of SQL Server products available
- LEARN MORE:
  - SQL Server Monitoring: https://www.sentryone.com/sql-server/sql-server-monitoring
  - Plus, get *your* questions answered!
    - Email Andy / Devon your specific monitoring questions
      -- MailTo: ayun@sentryone.com and MailTo: dwilson@sentryone.com
- Plan Explorer
  - https://www.sentryone.com/plan-explorer
  - Advanced Plan Explorer Usage for Tuning Execution Plans
- Special SQLskills-Only Presentations Coming in May!
  - Details will be posted on your course contents page!

SQL Server Monitoring

© SQLskills, All rights reserved.
http://www.SQLskills.com

SentryOne™

SQLskills' strategic partners and class sponsor

http://www.SentryOne.com

© SQLskills, All rights reserved.
http://www.SQLskills.com

Overview

- Query Store Fundamentals
- Understanding the Query Store Data
- Finding Performance Issues
- Query Store Performance
- Query Tuning
  - Plan Forcing
  - Finding Patterns
- Other Uses of Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Fundamentals

© SQLskills, All rights reserved.
http://www.SQLskills.com

Common Performance Problems

- Queries that are slow/take too long to execute
- Queries that consume significant resources
- Queries that execute with high frequency
- Queries with high variability in performance

© SQLskills, All rights reserved.
http://www.SQLskills.com

What data is available AFTER a problem?

ERRORLOG

system_health and default trace

Anything you've set up to capture data

© SQLskills, All rights reserved.
http://www.SQLskills.com

What tools do you use DURING a problem?

- Performance Monitor
- DMVs
- Extended Events/Trace
- Query plans
- DBCC commands
- Third Party Monitoring Software
- Xperf
- CU/SP Release notes
- Others?

© SQLskills, All rights reserved.
http://www.SQLskills.com

Sources for Query Performance Data

Performance Monitor

DMVs

Extended Events

SSMS

© SQLskills, All rights reserved.
http://www.SQLskills.com

Example Data

What about baseline data?

PerfMon
- SQL Compilations/sec
- SQL Re-compilcations/sec

DMVs
- sys.dm_exec_query_stats
- sys.dm_exec_sql_text
- sys.dm_exec_query_plan

Extended Events
- sp_statement_completed
- sql_statement_completed
- query_post_execution_showplan

SSMS
- Estimated plan
- Actual plan
- STATISTICS IO
- STATISTICS TIME

© SQLskills, All rights reserved.
http://www.SQLskills.com

How do you baseline query performance?

- Snapshot data from multiple DMVs
- Use sp_whoisactive to snapshot data
- Capture data using Extended Events
  - Trace for SQL Server 2008R2 and earlier
- Use a third-party monitoring tool
- OpenQueryStore
  - SQL Server 2008 – SQL Server 2014
- Query Store
  - SQL Server 2016+

© SQLskills, All rights reserved.
http://www.SQLskills.com

Who should use Query Store?

You do not need to be a performance-tuning expert to use Query Store. In fact, Query Store was written so that *anyone* who uses SQL Server can find the worst-performers and regressed queries.

If you are a performance-tuning expert or have extensive experience with SQL Server, then you'll be able to work beyond the standard functionality of Query Store to fully leverage all it has to offer.

© SQLskills, All rights reserved.
http://www.SQLskills.com

History of Query Store

- 2014 Q3: First public session by Conor Cunningham at PASS Summit
- 2015: Private preview in Azure SQL Database
- 2015 Q3: Public session by Borko Novakovic at PASS Summit
- 2015 Q4: Publicly available in Azure SQL Database
- 2016 Q2: Generally available in SQL Server 2016
- 2016 Q3/Q4: Enabled by default for all Azure SQL Databases

"Data Driven" approach

© SQLskills, All rights reserved.
http://www.SQLskills.com

What is Query Store?

Billed as a flight data recorder

What kind of a history do you have to look at *after* a performance issue or server restart/crash?

Provides information about query execution

Captures performance and query data

Available in *ALL* editions of SQL Server

Highly recommended to run the latest release (2016 SP2 CU14, 2017 CU22, 2019 CU6) for all editions

Enabled by default in Azure SQL Database since late 2016

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Basics

- Enabled at the database level
- Data persisted in internal tables
- Cannot be enabled for master, model* or tempdb
- Requires VIEW DATABASE STATE to view data
- Requires db_owner to force/un-force plans
- Data is not captured on readable secondaries

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query and Performance Data in Query Store

sys.dm_exec_query_plan
sys.dm_exec_cached_plans
sys.dm_exec_sql_text

Plan Store

sys.dm_exec_query_stats

Runtime Stats Store

sys.dm_exec_session_wait_stats

Wait Stats Store

SQL Server 2017 and Azure SQL Database

© SQLskills, All rights reserved.
http://www.SQLskills.com

Data Captured by Query Store

Plan Store

- Compile, bind and optimization duration
- Compile memory
- Last execution time
- Context settings
- Query text
- Query plan

Runtime Stats Store

- Execution counts
- Duration
- CPU
- Logical reads
- Physical reads
- Write
- Memory use
- DOP
- Log bytes/used
- tempdb

Wait Stats Store

- Wait statistics per plan

SQL Server 2017 and Azure SQL Database

© SQLskills, All rights reserved.
http://www.SQLskills.com

What queries are captured?

- Any T-SQL DML
  - CAPTURE_MODE will change what is captured
  - SELECT includes cursors, SET assignment, internal queries
- Not the same as what is cached in the DMVs
  - queries that include OPTION (RECOMPILE)
  - ad hoc queries (even with 'Optimize for Ad Hoc Workloads' enabled)
  - internal queries (e.g. automatic updates to statistics)
- Each statement in an object is captured as a separate query
  - Stored procedures, functions, triggers
- Statement text appears in parameterized form (unless ad hoc)

© SQLskills, All rights reserved.
http://www.SQLskills.com

What queries are not captured?

- DDL
  - CREATE, ALTER, DROP
- BULK INSERT
- “commands”
  - DBCC, KILL, BACKUP
- SET SHOWPLAN
- Those executed from different database context

© SQLskills, All rights reserved.
http://www.SQLskills.com

Other Data NOT Captured by Query Store

- User/login that executed the query
- Workstation from which the query was executed
- Application from which the query was executed
- Execution plan with actual runtime statistics

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Query Store in Action

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Settings

- Query Store is *not* enabled by default in SQL Server 2016, SQL Server 2017, or SQL Server 2019
- Query Store *is* enabled by default for new databases in Azure SQL Database (included those in Managed Instances)
  - Retroactively enabled for databases
- There are multiple settings related to Query Store configuration, and they affect what data is collected and how it is stored

© SQLskills, All rights reserved.
http://www.SQLskills.com

Adapted from: https://msdn.microsoft.com/en-us/library/mt631173.aspx

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Options(1)

OPERATION_MODE = [READ_WRITE | READ_ONLY]

Why it matters: Do you want to capture new data?

QUERY_CAPTURE_MODE = [ALL | AUTO | CUSTOM | NONE]

Why it matters: Do you want to capture all queries, or just those most relevant to your workload? CUSTOM is **new** in SQL 2019

MAX_PLANS_PER_QUERY =

Why it matters: You may have more than 200 unique plans for a query (?) (!)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Options(2)

MAX_STORAGE_SIZE_MB =

Why it matters: The default is 1GB or less, “right” value depends on multiple factors

CLEANUP_POLICY = (STALE_QUERY_THRESHOLD_DAYS = #)

Why it matters: You need to think about how much data you want to keep

SIZE_BASED_CLEANUP_MODE = [AUTO | OFF]

Why it mattes: If this is OFF and MAX_STORAGE_SIZE_MB is reached, Query Store will switch to READ_ONLY

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Options (4)

DATA_FLUSH_INTERVAL_SECONDS =

Why it matters: Determines how frequently Query Store is written to disk

INTERVAL_LENGTH_MINUTES =

Why it matters: Affects the space needed by Query Store and the time windows across which you can analyze data

WAIT_STATS_CAPTURE_MODE [ON | OFF]

Why it matters: Enabled by default (and enabled when upgrade to SQL 2017)

SQL Server 2017 and Azure SQL Database

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Capture Policy: CUSTOM

- This additional option allows you to further customize what queries are captured
  - Three options establish OR conditions
- Only available in SQL Server 2019 and Azure SQL Database
- Benefits workloads that are extremely ad hoc
- There are four settings to consider

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store CUSTOM Mode Options

STALE_CAPTURE_POLICY_THRESHOLD

Why it matters: Sets the window of time for query evaluation

EXECUTION_COUNT

Why it matters: If a query executes < N times, unless compile or execution time exceed set values, it won't be captured

TOTAL_COMPILE_CPU_TIME_MS

Why it matters: Estimation based on existing data may be difficult

TOTAL_EXECUTION_CPU_TIME_MS

Why it matters: Existing data should help with estimation

© SQLskills, All rights reserved.
http://www.SQLskills.com

Default Values

Setting	2016/2017	2019	SQL DB Basic	SQL DB Standard	SQL DB Premium
QUERY_CAPTURE_MODE	ALL	AUTO	AUTO	AUTO	AUTO
MAX_STORAGE_SIZE_MB	100	1000	10	100	1024
CLEANUP_POLICY	30	30	7	30	30
SIZE_BASED_CLEANUP_MODE	AUTO	AUTO	AUTO	AUTO	AUTO
DATA_FLUSH_INTERVAL_ SECONDS	900	900	900	900	900

- Tier change in Azure SQL Databases causes a change in Query Store settings, unless they were modified using ALTER DATABASE

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Settings(5)

- Improper configuration of these settings can cause data to be removed from Query Store before expected, or Query Store can stop collecting data entirely
  - https://www.sqlskills.com/blogs/erin/query-store-settings/
- Stored in sys.database_query_store_options, along with current and desired status
- Use Extended Events to monitor
  - query_store_db_settings_and_state
  - query_store_db_settings_changed
  - query_store_disk_size_info
  - query_store_disk_size_over_limit

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Configuring Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Settings and AG Replicas

- Querying sys.database_query_store_options on a replica fetches the metadata settings cached *in memory*
- Any change made to the configuration of Query Store *is* persisted to disk on the primary and thus propagated over to the secondary and written to disk
- However, on-disk changes to the setting are only be propagated to memory cache once the secondary replica is restarted or becomes primary
- https://www.sqlskills.com/blogs/erin/different-query-store-settings-for-a-database-in-an-availability-group/

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Settings in a sqlproj

- Query Store configuration values can be defined in a sqlproj
  - Database development project extension created by Visual Studio and stores the schema of the database and can include SQL source code
- For example, a DACPAC deployment would set configuration options to the same values that are set in the project
  - If these are different than what is set in production, values can change with a deployment

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Trace Flags (1)

- Current configuration prevents queries from executing until all necessary Query Store data has been loaded into memory
  - May be an issue for larger data sets
  - Check for QDS_LOADDB wait type

- TF 7752 allows queries to execute while Query Store data loads asynchronously during startup
  - Data about query execution will not be collected until Query Store data is loaded into memory
  - This is default behavior in SQL Server 2019
- Use Extended Events to monitor initial loading and shutdown

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Trace Flags (2)

- By default, SQL Server will wait until the Query Store data that's in memory is written to disk before fully shutting down
  - This could delay a failover

- TF 7745 bypasses writing Query Store data to disk at shutdown
  - Query Store data may be lost, the amount is dependent on the value for DATA_FLUSH_INTERVAL_SECONDS

© SQLskills, All rights reserved.
http://www.SQLskills.com

Cross Database Queries

- Queries executed from different database context are not captured in a user database with Query Store enabled
- With Query Store enabled for multiple databases, with queries across the databases, a query is only captured in the database from which it is executed
- For Linked Servers, if Query Store enabled in both the originating and destination database, queries are *only* captured in the destination database if sp_executesql is used

© SQLskills, All rights reserved.
http://www.SQLskills.com

Understanding the Query Store Data

© SQLskills, All rights reserved.
http://www.SQLskills.com

Image attribution: https://msdn.microsoft.com/en-us/library/mt631173.aspx

© SQLskills, All rights reserved.
http://www.SQLskills.com

```
SELECT
    TOP (1)
    [Extent1].[UserId] AS [ID]
FROM [dbo].[UserSessions] AS [Extent1]
WHERE [UserID] =
    28ACBBC0-75DF-4BEF-B9B7-48CB8860897C;

SELECT
    TOP (1)
    [Extent1].[UserId] AS [ID]
FROM [dbo].[UserSessions] AS [Extent1]
WHERE [UserID] =
    8640610E-0627-4D9D-B07E-542EBA495922;
```

query_text

query_text_id	query_sql_text
2144475	SELECT TOP (1) [Extent1].[UserId] AS [ID] FROM [dbo].[UserS...
3135658	SELECT TOP (1) [Extent1].[UserId] AS [ID] FROM [dbo].[UserS...

context_settings

context_settings_id	set_options
14	0x000210FB
15	0x000214FB

query

query_id	query_text_id	context_settings_id	object_id	query_hash	query_parameterization_type_desc	count_compiles
2656838	2144475	14	0	0xF0B23C1ECE53788E	None	3
3674359	3135658	14	0	0xF0B23C1ECE53788E	None	1

plan

plan_id	query_id	query_plan_hash	query_plan	is_forced_plan
2168010	2656838	0x2B21AE35885B1F12	<ShowPlanXML x...	0
3163697	3674359	0x2B21AE35885B1F12	<ShowPlanXML x...	0

© SQLskills, All rights reserved.
http://www.SQLskills.com

Compiling and the Plan Store

In the plan cache, only execution stats are updated

© SQLskills, All rights reserved.
http://www.SQLskills.com

Understanding Runtime Statistics

```
SELECT [OrderID], [OrderDate], [CustomerPurchaseOrderNumber]
FROM [Sales].[Orders]
WHERE [CustomerID] = @CustID
ORDER BY [OrderDate];
```

query_id
768

sys.query_store_query

plan_id	query_plan
805	<ShowPlanXML xmlns="http://sc

sys.query_store_plan

runtime_stats_interval_id	plan_id	count_executions	avg_duration	last_duration	max_duration	min_duration
2932	805	1	800	800	800	800

sys.query_store_runtime_stats

© SQLskills, All rights reserved.
http://www.SQLskills.com

Understanding Runtime Statistics

```
SELECT [OrderID], [OrderDate], [CustomerPurchaseOrderNumber]
FROM [Sales].[Orders]
WHERE [CustomerID] = @CustID
ORDER BY [OrderDate];
```

query_id
768

sys.query_store_query

plan_id	query_plan
805	<ShowPlanXML xmlns="http://sc

sys.query_store_plan

runtime_stats_interval_id	plan_id	count_executions	avg_duration	last_duration	max_duration	min_duration
2932	805	432	621	589	816	300

sys.query_store_runtime_stats

SQLskills immerse yourself in sql server
© SQLskills, All rights reserved.
http://www.SQLskills.com

Understanding Runtime Statistics

```
SELECT [OrderID], [OrderDate], [CustomerPurchaseOrderNumber]
FROM [Sales].[Orders]
WHERE [CustomerID] = @CustID
ORDER BY [OrderDate];
```

query_id
768

sys.query_store_query

plan_id	query_plan
805	<ShowPlanXML xmlns="http://sc

sys.query_store_plan

runtime_stats_interval_id	plan_id	count_executions	avg_duration	last_duration	max_duration	min_duration
2932	805	432	621	589	816	300
2933	805	1	350	350	350	350

sys.query_store_runtime_stats

© SQLskills, All rights reserved.
http://www.SQLskills.com

Understanding Runtime Statistics

```
SELECT [OrderID], [OrderDate], [CustomerPurchaseOrderNumber]
FROM [Sales].[Orders]
WHERE [CustomerID] = @CustID
ORDER BY [OrderDate];
```

runtime_stats_interval_id	start_time	end_time
2932	2021-05-05 08:00:00.0000000 +00:00	2021-05-05 09:00:00.0000000 +00:00
2933	2021-05-05 09:00:00.0000000 +00:00	2021-05-05 10:00:00.0000000 +00:00

sys.query_store_runtime_stats_interval

runtime_stats_interval_id	plan_id	count_executions	avg_duration	last_duration	max_duration	min_duration
2932	805	432	621	589	816	300
2933	805	1	350	350	350	350

sys.query_store_runtime_stats

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Exploring Query Store Data

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Views

Text	• sys.query_store_query_text • sys.query_store_query • sys.query_context_settings
Plan	• sys.query_store_plan
Runtime Stats	• sys.query_store_runtime_statistics • sys.query_store_runtime_statistics_interval
Wait Stats	• sys.query_store_wait_stats

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store System Tables

- Data resides in PRIMARY filegroup
- The Query Store data is exposed through catalog views
- The underlying tables all have primary keys, *but none have foreign keys*
  - Integrity is logically enforced within the engine
- If you find Query Store in an ERROR state (actual_state = 3), run sp_query_store_consistency_check to fix
  - Very extreme edge case
- Tables are reorganized regularly (internally managed)
  - Use Extended Events to monitor

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Dates

- Stored as DATETIMEOFFSET data type
- Use AT TIME ZONE get dates in your time zone

```
SELECT
    [start_time] AT TIME ZONE 'Eastern Standard Time' [EST StartTime]
FROM [sys].[query_store_runtime_stats_interval]
```

  - Reference time zone list in sys.time_zone_info
- Use DATEADD to search date ranges
- https://www.sqlskills.com/blogs/erin/handling-dates-in-query-store/

© SQLskills, All rights reserved.
http://www.SQLskills.com

How Much Space Do I Need?(1)

- There are multiple factors to consider:
  - How long are you keeping the data?
  - Across what interval are you capturing data?
  - Are you capturing wait statistics (2017+, Azure SQL DB)
  - What does your workload look like?
    - Number of unique queries?
    - Number of unique plans?
  - What is the capture mode?

Biggest impact on space required

© SQLskills, All rights reserved.
http://www.SQLskills.com

How Much Space Do I Need? (2)

- Default size for Azure SQL Database is either 10MB, 100MB, or 1GB (depends on tier)
  - Counts against **total space** for the database
  - Interval for all is 60 minutes
- My recommendation: start with 2GB and monitor
  - current_storage_size_mb in sys.database_query_store_options
  - Extended Events
    - query_store_database_out_of_disk_space
    - query_store_disk_size_over_limit

© SQLskills, All rights reserved.
http://www.SQLskills.com

How the Interval Setting Affects Space

- sys.query_store_runtime_statistics

INTERVAL_LENGTH_MINUTES	Rows/Day	Rows/Month	Approximate space (KB)	Total space for 30 days for 1000 unique queries (MB)	Total space for 30 days for 5000 unique queries (MB)
60	24	732	20	20	100
30	48	1,464	40	40	195
15	96	2,928	80	80	390
10	144	4,392	120	118	590
5	288	8,784	240	235	1175
1	1,440	43,920	1,201	1175	5865

Each row in sys.query_store_runtime_stats is 28 bytes

© SQLskills, All rights reserved.
http://www.SQLskills.com

Estimating Space Needed for Plans(1)

- The following table represents a calculation/estimation
  - (Average size of plan (bytes) *
    Unique number of plans)
    / 1024 / 1024
- To know the average size of your plans, you'll have to interrogate the system tables (sys.query_store_plan)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Estimating Space Needed for Plans(2)

Approximate size of each plan (bytes)	Total space (MB) 50,000 plans	Total space (MB) 100,000 plans	Total space (MB) 250,000 plans	Total space (MB) 500,000 plans	Total space (MB) 1,000,000 plans
2000	95	191	477	954	1907
5000	238	477	1192	2384	4768
7000	334	668	1669	3338	6676
10000	477	954	2384	4768	9537

© SQLskills, All rights reserved.
http://www.SQLskills.com

Example 1 of Used Space

- INTERVAL_LENGTH_MINUTES = 30
- CLEANUP_POLICY = (STALE_QUERY_THRESHOLD_DAYS = 30)

Data	Rows	MB
query_text	581905	102
query	582140	38
plan	596200	420
runtime_stats	1105562	795
runtime_stats_interval	1388	<1
context_settings	25	<1

© SQLskills, All rights reserved.
http://www.SQLskills.com

Example 2 of Used Space

- INTERVAL_LENGTH_MINUTES = 60
- CLEANUP_POLICY = (STALE_QUERY_THRESHOLD_DAYS = 30)

Data	Rows	MB
query_text	709719	1076
query	709833	534
plan	721567	3674
runtime_stats	1695027	1465
runtime_stats_interval	153	24
context_settings	19	8

© SQLskills, All rights reserved.
http://www.SQLskills.com

Finding Performance Issues

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store vs. Third-Party Monitoring Tools

Query Store

- Enable for *each* user database
- Configure collection options
- Retention policy affects size of data *in* user DB
- Does not capture runtime parameters
- Captures *query-level* wait stats
- SQL Server 2016+
- Ability to force plans

3rd Party

- Captures data for all DBs on an instance
- Configure collection options
- Data stored in a separate DB (on a separate instance)
- Captures runtime parameters with queries
- Captures instance-level wait stats
- All SQL Server versions
- Cannot force plans

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Reports

- Available within Management Studio
  - Current SSMS version is 18.x
  - Can be used for SQL 2016/2017/2019, or Azure SQL DB
- Current reports:
  - Regressed Queries
  - Overall Resource Consumption
  - Top Resource Consuming Queries
  - Queries With Forced Plans
  - Queries with High Variation
  - Query Wait Statistics*
  - Tracked Queries

© SQLskills, All rights reserved.
http://www.SQLskills.com

Using the Reports

- Performance reports are configurable
- Can specify resource(s), time frame, and number of queries to return
- Filtering options include CPU, Duration, Execution Count, Logical Reads, Logical Writes, Memory Consumption, Physical Reads
  - Wait Time, Log Memory, tempdb Memory, Row Count also available in SQL Server 2017
  - Avg, Min, Max, Total, StDev

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Regression

Calculated using captured values

CPU, duration, I/O, memory

Displayed in UI via “Regressed Queries” report

You can select a single resource on which to filter, as well as the time frame

You can also query the catalog views directly

Write your own T-SQL to find regressed queries based on changes in one or more resources (e.g. duration increased by 20% *and* IO increased by 40%)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Azure SQL Database

- Azure Portal contains Intelligent Performance data for each database
  - Performance overview
  - Performance recommendations
  - Query Performance Insight
  - Automatic Tuning
- Azure SQL Analytics can also be configured for additional monitoring and recommendations
  - Available for Azure SQL DB (single and pooled) and Managed Instance databaes

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Using Query Store Reports

© SQLskills, All rights reserved.
http://www.SQLskills.com

Notes About Querying the Catalog Views

- Every underlying system table has a clustered index, and most have one nonclustered index
- You cannot create additional indexes to support queries, thus index scans will be required for a lot of ad-hoc queries that search non-indexed columns
- Options
  - Use DBCC CLONEDATABASE to create a schema-only copy of the database with Query Store data and query it
  - Use read-only secondaries for queries against Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Indexes for the Query Store Tables

Data	Clustered Index	Nonclustered Index
query_text	query_text_id	statement_sql_handle
query	query_id	query_text_id, context_settings_id
plan	plan_id	query_id (-)
runtime_stats	plan_id, runtime_stats_interval_id, execution_type	runtime_stats_id
runtime_stats_interval	runtime_stats_interval_id	end_time
context_settings	context_settings_id(-)	
wait_stats	runtime_stats_interval_id, plan_id, wait_category, execution_type	wait_stats_id

© SQLskills, All rights reserved.
http://www.SQLskills.com

Comparing Data Across Sources

XE and Trace

- Individual query metrics
- Captured across a window of time
- Data may or may not be filtered

DMVs

- Aggregate metrics for a query
- Performance of query as long as it's been in cache
- Based on what plans are in cache at that moment

Query Store

- Aggregate metrics for a query plan
- Performance of plan during pre-defined, consistent intervals
- Continuously captured

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: DMVs vs. XE vs. Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Wait Statistics in SQL Server

- The term “wait” means that a thread running on a processor cannot proceed because a resource it requires is unavailable
  - The thread has to wait until that resource is available
- The resource the thread is waiting for is tracked by SQL Server
  - Each resource maps to a wait type
- Example resources a thread may wait for:
  - A lock (LCK_M_XX wait type)
  - A data file page to be read into the buffer pool (PAGEIOLATCH_XX)
  - Results from part of a parallel query (CXPACKET)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Using Wait Statistics

- Information historically only available at the instance level or at the thread level
  - sys.dm_os_wait_stats
  - sys.dm_os_waiting_tasks
- Excellent for understanding what the system is “normally” waiting on, or troubleshooting to understand what’s waiting at this moment

© SQLskills, All rights reserved.
http://www.SQLskills.com

Enhancements to Wait Statistics

- SQL Server 2016 added the ability to look at session-level wait statistics
  - sys.dm_exec_session_wait_stats
  - Only available for current sessions
- SQL Server 2016 SP1 added wait statistics to the actual execution plan
- SQL Server 2017 includes wait statistics in Query Store
  - sys.query_store_wait_stats

© SQLskills, All rights reserved.
http://www.SQLskills.com

Wait Statistics and Query Store

- Wait statistics are tied to the query plan and the runtime stats interval
  - Not the same as wait stats information *in* a plan
- Wait types are grouped into categories for simplicity
  - Consider the resource requirements needed to track 900+ wait types
  - Slight change in approach for those who are familiar with wait stats
- Information captured:
  - Wait category
  - Wait time (total, average, last, min, max, stdev)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Wait Statistics Categories

Wait Category	Wait types included
CPU	SOS_SCHEDULER_YIELD
Buffer IO	PAGEIOLATCH%
Parallelism	CXPACKET, EXECSYNC
Lock	LCK_M_%
Latch	LATCH_%
Buffer Latch	PAGELATCH%
Network IO	ASYNCH_NETWORK_IO, NET_WAITFOR_PACKET, PROXY_NETWORK_IO, …
Other Disk IO	ASYNCH_IO_COMPLETION, IO_COMPLETION, BACKUPIO, WRITE_COMPLETION, …
Tran Log IO	WRITELOG, LOGMGR, LOGBUFFER, LOGMGR_RESERVE_APPEND, CHKPT, …
Memory	RESOURCE_SEMAPHORE, CMEMTHREAD, CMEMPARTITIONED, MEMORY_GRANT_UPDATE, …

https://www.sqlskills.com/help/waits/

© SQLskills, All rights reserved.
http://www.SQLskills.com

Top Waits in Azure

Wait stats in Query Store on SQL Azure worldwide

BY WAIT_CATEGORY • REFRESHED: 6:08:21 PM

wait_category	max_query_wait_tim...	total_query_wait_tim...	percent
LOCK	693728	7060938862	27.99
PARALLELISM	1818930	4863058449	19.28
CPU	835701	4414964929	17.50
BUFFERIO	482195	2666986143	10.57
NETWORKIO	1803494	1420400350	5.63
UNKNOWN	591342	1377305042	5.46
OTHERDISKIO	111690	865909119	3.43
LOGRATEGOVERNOR	57612	536220709	2.13
LATCH	186490	502279984	1.99
BUFFERLATCH	76716	490459332	1.94
IDLE	217673	418901616	1.66
MEMORY	86406	372126921	1.48
USERWAIT	162	185536422	0.74
TRANLOGIO	22299	15596692	0.06
PREEMPTIVE	61567	15012519	0.06

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Wait Statistics in Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

The Relationship Between Plans and Statistics

```
SELECT [OrderID], [OrderDate],
[CustomerPurchaseOrderNumber]
FROM [Sales].[Orders]
WHERE [CustomerID] = @CustID
ORDER BY [OrderDate];
```

query_id = 768

plan_id = 805

runtime_stats_interval_id	start_time	end_time
2932	2021-05-05 08:00:00.0000000 +00:00	2021-05-05 09:00:00.0000000 +00:00
2933	2021-05-05 09:00:00.0000000 +00:00	2021-05-05 10:00:00.0000000 +00:00
2934	2021-05-05 10:00:00.0000000 +00:00	2021-05-05 11:00:00.0000000 +00:00

sys.query_store_runtime_stats_interval

© SQLskills, All rights reserved.
http://www.SQLskills.com

runtime_stats_interval_id	start_time	end_time
2932	2021-05-05 08:00:00.0000000 +00:00	2021-05-05 09:00:00.0000000 +00:00
2933	2021-05-05 09:00:00.0000000 +00:00	2021-05-05 10:00:00.0000000 +00:00
2934	2021-05-05 10:00:00.0000000 +00:00	2021-05-05 11:00:00.0000000 +00:00

runtime_stats_interval_id	plan_id	count_executions	avg_duration	last_duration	max_duration	min_duration
2932	1397	432	621	589	912	303
2933	1397	581	493	350	816	291
2934	1397	634	587	502	865	447

runtime_stats_interval_id	plan_id	wait_category_desc	total_query_wait_time_ms	avg_query_wait_time_ms
2932	1397	CPU	234	35
2932	1397	Buffer IO	1305	24
2932	1397	Parallelism	21846	158
2932	1397	Other Disk IO	124	8

© SQLskills, All rights reserved.
http://www.SQLskills.com

Estimating Cardinality in SQL Server

- The Query Optimizer evaluates the cost of one or more plans when deciding which plan to ultimately execute
- One factor used to determine cost is the number of estimated rows that will need to be processed for each operator
  - This is the cardinality estimate
- The cardinality estimator (CE) component was significantly changed in SQL Server 2014
  - First redesign since SQL Server 7.0

© SQLskills, All rights reserved.
http://www.SQLskills.com

Cardinality Estimate Issues

- Major red flag to watch for, not just when upgrading to 2014+
  - Skewed estimate vs. actual
- Magnification and distortion as we move through the plan tree
- Other symptoms:
  - Performance may be good sometimes and bad other times
  - Differences in object access (seeks vs. scans)
  - Differences in join types (loop vs. hash)
  - Inadequate query memory allocated, causing spills
    - Conversely, too much memory allocated, reducing concurrency
  - Query performs badly or doesn't execute at all due to memory error

© SQLskills, All rights reserved.
http://www.SQLskills.com

Cardinality Estimator Version

- The new CE will be used in SQL Server 2014 if the database has the compatibility level set to 120
- CE version is determined by the LEGACY_CARDINALITY_ESTIMATION database scoped setting*
  - Database compatibility level is relevant for new CE
  - Trace flags 9481 and 2312 can still be used to change CE for individual queries (with QUERYTRACEON hint)
- CE version for tempdb is relevant if you use temporary tables

© SQLskills, All rights reserved.
http://www.SQLskills.com

Compatibility Mode

- The database compatibility mode affects T-SQL and query processing behavior
  - Not changed automatically as part of an upgrade
  - https://docs.microsoft.com/en-us/sql/t-sql/statements/alter-database-transact-sql-compatibility-level?view=sql-server-ver15
- Current compatibility mode can be used with the legacy CE
- Testing is still recommended
- Compatibility Mode is tracked in Query Store (sys.query_store_plan)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: CE and Compat Mode with Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Performance

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query and Plan Hashes ("fingerprints")

- Added in SQL Server 2008
- Available in sys.dm_exec_query_stats
- Improved ability to find queries that are syntactically the same with different literal values
- Easier to find patterns in query execution
- Blog posts:
  - https://blogs.msdn.microsoft.com/bartd/2008/09/03/query-fingerprints-and-plan-fingerprints-the-best-sql-2008-feature-that-youve-never-heard-of/
  - https://blogs.msdn.microsoft.com/bartd/2010/05/26/finding-procedure-cache-bloat/

© SQLskills, All rights reserved.
http://www.SQLskills.com

query_hash

- Generated from a tree of logical operators *after* parsing
- Queries do not *have* to have the same text to have the same query_hash
  - Whitespace and comments do not matter
- Object names must be the same (e.g. table, view) and same aliases
- Hints, if used, must be exactly the same
- SET options can change query_hash if they change semantics
- Not affected by database or instance
  - Same query in two different databases will have the same query_hash

© SQLskills, All rights reserved.
http://www.SQLskills.com

query_plan_hash

- Generated from a tree of physical operators
- Plans must have the same shape and use the same operators
  - Same shape but different join operator will have different query_plan_hash values
- Certain operator attributes must be the same
  - Table name = yes, Row estimates = no
- Not affected by database or instance
  - The same shaped plan in two different databases can have the same query_plan_hash
- query_hash is included in query_plan_hash

© SQLskills, All rights reserved.
http://www.SQLskills.com

Parameterization in SQL Server

- Default is SIMPLE (database setting)
- If a statement is deemed to be “safe”, then it can be *automatically* parameterized

© SQLskills, All rights reserved.
http://www.SQLskills.com

If No Parameterization Then…

- If the statement is not safe and cannot be parameterized, then the un-parameterized statement (containing specific literal values) will be placed in the plan cache
  - It can be re-used if the **exact same query** is submitted – as long it matches EXACTLY in terms of textual matching
  - This consumes space in the plan cache
  - It also consumes space in Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Example Ad Hoc Queries

```
SELECT
     w.ColorID,
     s.StockItemName
FROM Warehouse.Colors w
JOIN Warehouse.StockItems s
     ON w.ColorID = s.ColorID
WHERE w.ColorName = 'Blue';
```

```
SELECT
     TOP 1 o.SalesPersonPersonID,
     o.OrderDate,
     ol.StockItemID
FROM Sales.Orders o
JOIN Sales.OrderLines ol
    ON o.OrderID = ol.OrderID
WHERE o.CustomerID = 3284;
```


© SQLskills, All rights reserved.
http://www.SQLskills.com

Ad hoc Statements and the Plan Cache

- Plan cache space is limited
  - There are known issues related to ad hoc (un-parameterized) statements bloating the plan cache
- View cached plans: `sys.dm_exec_cached_plans`
- Plan cache memory limits:
  - SQL Server 2008+ and SQL Server 2005 SP2
    - 75% of visible target memory from 0-4GB
      + **10%** of visible target memory from 4Gb-64GB
      + **5%** of visible target memory > 64GB

2005 SP2 +	
Memory	**Plan Cache**
4GB	3.0GB
8GB	3.5GB
16GB	4.2GB
32GB	5.8GB
64GB	9.0GB
128GB	12.2GB
256GB	18.6GB
512GB	31.4GB

© SQLskills, All rights reserved.
http://www.SQLskills.com

Managing the Plan Cache with Ad Hoc Stmts

- Check the size of the plan cache and how it's being used:
  - https://www.sqlskills.com/blogs/kimberly/plan-cache-adhoc-workloads-and-clearing-the-single-use-plan-cache-bloat/
- Enable the Optimize for Ad Hoc Workloads instance setting
  - This exists as a DATABASE SCOPED option in Azure SQL DB
  - When a query first executes, only the compiled plan stub will go into cache
    - If the query executes a second time, the full plan will go into cache
- Option: create a job that runs on a regular basis to check the size of single-use plans and clear them if they exceed a certain amount (e.g. > 2GB)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Ad Hoc Statements and Query Store

- Queries are uniquely identified based on:
  - Query text
  - Context settings
  - OBJECT_ID
  - Type of query parameterization
  - batch_sql_handle
- Query Store space is limited, based on MAX_STORAGE_SIZE_MB setting
- You can control what statements are saved in Query Store using QUERY_CAPTURE_MODE

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Query Store and Performance

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Overhead

- Designed to have minimal overhead
- High volume, ad hoc workloads may appear as though they have performance issues
  - This can be related to synchronous overhead on the query compilation path, due to conflicts when writing to the statement hash map
  - This can be related to how the asynchronous flush of data from memory to disk is managed internally
  - This can be related to the amount of memory needed to track ad hoc queries
- Performance optimizations added in 2017
  - Back-ported to SQL Server 2016 SP2 CU2
    - https://support.microsoft.com/en-us/help/4340759
    - Don't let the reference to spinlocks concern you

Will show up as QDS_STMT waits

© SQLskills, All rights reserved.
http://www.SQLskills.com

Key performance optimizations

- Internal memory limits
- Smaller transactions for background flushes of data
- Changes to Query Store's cleanup mechanism

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Memory Use

- Unique key for each query is stored in a hash map in memory (per database)
- Workloads that are more ad-hoc typically have a larger hash map
  - More unique queries based on query text
  - Potentially same query_hash
  - Potentially same query_plan_hash
- Runtime statistics information is stored in a separate hash map in memory (per database)
  - Changed to be stored in one hash map per instance
  - Implemented for SQL 2016+ in latest CUs

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Memory Structures

Memory Clerk/Object	Description
MEMORYCLERK_QUERYDISKSTORE_HASHMAP	Hash table of queries and plans for the instance/node Uses MEMOBJ_QUERYDISKSTORE (NUMA enabled) Repopulated from disk at startup
CACHESTORE_QDSRUNTIMESTATS	Cache of aggregated runtime stats before they are persisted to disk
MEMOBJ_QUERYSTOREPARTITIONEDHEAP	CPU partitioned heap for execution stats for the instance
USERSTORE_QDSSTMT	Temporary buffers per DB to store statements before persisted to disk
MEMORYCLERK_QUERYDISKSTORE	General clerk for overall Query Store for instance Should be fairly fixed in size and small Uses MEMOBJ_QUERYDISKSTORE (NUMA enabled)
CAHCHESTORE_QDSCONTEXTSETTINGS	Tracks unique context settings across all queries per DB
MEMORYCLERK_QUERYDISKSTORE_STATS	Used for allocating partitioned runtime stats objects (partitioned per CPU core) in the Query Store fast path

© SQLskills, All rights reserved.
http://www.SQLskills.com

Internal Memory Limits

- "Query Store now imposes internal limits to the amount of memory it can use and automatically changes the operation mode to READ-ONLY until enough memory has been returned to the Database Engine, preventing performance issues."

© SQLskills, All rights reserved.
http://www.SQLskills.com

Adapted from: https://msdn.microsoft.com/en-us/library/mt631173.aspx

© SQLskills, All rights reserved.
http://www.SQLskills.com

Internal Memory Limits

- "Query Store now imposes internal limits to the amount of memory it can use and automatically changes the operation mode to READ-ONLY until enough memory has been returned to the Database Engine, preventing performance issues."

- Azure SQL DB
- SQL Server 2019 CU8
- SQL Server 2017 CU22
- SQL Server 2016 SP2 CU15

© SQLskills, All rights reserved.
http://www.SQLskills.com

Smaller transactions for background flushes of data

- Data is regularly flushed, asynchronously, from memory to disk
  - Background activity
  - Previously, for high-volume workloads, this could take an extended period of time

© SQLskills, All rights reserved.
http://www.SQLskills.com

Adapted from: https://msdn.microsoft.com/en-us/library/mt631173.aspx

© SQLskills, All rights reserved.
http://www.SQLskills.com

Smaller transactions for background flushes of data

- Data is regularly flushed, asynchronously, from memory to disk
  - Background activity
  - Previously, for high-volume workloads, this could take an extended period of time

- Azure SQL DB
- SQL Server 2019
- SQL Server 2017
- SQL Server 2016 SP2 CU2

© SQLskills, All rights reserved.
http://www.SQLskills.com

Changes to Query Store's cleanup mechanism

- Query Store cleanup kicks in when it consumes 90% of its maximum size (MAX_STORAGE_SIZE_MB)
- Cleanup also regularly occur to keep only the last *N* days of data (STALE_QUERY_THRESHOLD_DAYS)
  - Cleanup removes queries based on age and significance, and also removes their plans, runtime statistics and wait statistics
  - Cleanup continues until it drops to 80% of maximum size

© SQLskills, All rights reserved.
http://www.SQLskills.com

Adapted from: https://msdn.microsoft.com/en-us/library/mt631173.aspx

© SQLskills, All rights reserved.
http://www.SQLskills.com

Changes to Query Store's cleanup mechanism

- Query Store cleanup kicks in when it consumes 90% of its maximum size (MAX_STORAGE_SIZE_MB)
- Cleanup also regularly occur to keep only the last *N* days of data (STALE_QUERY_THRESHOLD_DAYS)
  - Cleanup removes queries based on age and significance, and also removes their plans, runtime statistics and wait statistics
  - Cleanup continues until it drops to 80% of maximum size
- SQL Server 2019
- SQL Server 2017 CU16
- SQL Server 2016 SP2 CU8

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Capture Mode Considerations

- Default for QUERY_CAPTURE_MODE is now AUTO
  - Default in 2016 and 2017 was ALL
    - Not recommended, even with parameterized workload
  - AUTO ignores infrequent queries and any that have insignificant compile and execution CPU times.
    - Thresholds (execution count, compile and runtime CPU) are internally determined
  - Queries that are part of a stored procedure are ALWAYS captured in Query Store, regardless of compile and execution durations
- Note that Query Store still has to track (in memory) what queries have executed that are insignificant

© SQLskills, All rights reserved.
http://www.SQLskills.com

Capture Mode Considerations in 2019

- Implementing a QUERY_CAPTURE_MODE of CUSTOM allows you control what queries are captured in Query Store
  - Determined by execution count, total compile CPU time OR total execution CPU time
- Interval across which these are tracked also must be set
  - Range is 1 hour to 7 days
- Query Store *still* has to track what queries have executed that do not meet established thresholds

© SQLskills, All rights reserved.
http://www.SQLskills.com

Monitoring Query Store Impact

- CPU (PerfMon)
- Memory
  - sys.dm_os_memory_clerks
  - Monitor aforementioned types
- query_store_db_diagnostics Event
  - Database specific
  - In SQL Server 2016+
- query_store_global_mem_obj_size_kb Event
  - Instance level
  - Exists in SQL Server 2016+

© SQLskills, All rights reserved.
http://www.SQLskills.com

Troubleshooting Query Store(1)

- Status is READ_ONLY
  - Reached MAX_STORAGE_SIZE_MB
    - What is set for SIZE_BASED_CLEANUP?
  - Database maximum size on disk reached
  - Memory limit reached
    - Azure SQL DB
    - SQL Server 2016 SP2 CU15, SQL Server 2017 CU22, SQL Server 2019 CU8
  - Confirm database allows writes (e.g. READ_ONLY, EMERGENCY mode, SINGLE_USER)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Troubleshooting Query Store(2)

- Queries are not being captured
  - What are capture policy settings?
    - QUERY_CAPTURE_MODE
    - Custom settings in 2019
  - Not all queries are captured (e.g. DDL, DBCC)
  - If lock timeouts are occurring (error 1222), Query Store may not be able to acquire necessary internal locks *in the allowed time*, and it will stop captured data by design, to prevent introducing a bigger performance issue

© SQLskills, All rights reserved.
http://www.SQLskills.com

Troubleshooting Query Store(3)

- Unable to turn off Query Store
  - Customers previously reported being unable to turn off Query Store when the instance was starting up, and when the system was not performing well.
  - Now have an option to forcibly and immediately turn Query Store OFF in case of severe issues.

```
ALTER DATABASE OPTION SET QUERY_STORE = OFF (FORCED)
```

  - Available in:
    - SQL Server 2019 CU6+
    - SQL Server 2017 CU21+
    - SQL Server 2016 SP2 CU14+

© SQLskills, All rights reserved.
http://www.SQLskills.com

How ad-hoc is your workload?

```
/* Perform cardinality analysis when suspect ad hoc workloads */


SELECT COUNT(*) AS CountQueryTextRows
FROM sys.query_store_query_text;

SELECT COUNT(*) AS CountQueryRows
FROM sys.query_store_query;

SELECT COUNT(DISTINCT query_hash) AS CountDifferentQueryRows
FROM sys.query_store_query;

SELECT COUNT(*) AS CountPlanRows
FROM sys.query_store_plan;

SELECT COUNT(DISTINCT query_plan_hash) AS CountDifferentPlanRows
FROM sys.query_store_plan;
```


© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Performing Workload Analysis

© SQLskills, All rights reserved.
http://www.SQLskills.com

Workload Considerations

Number of compiles and recompiles	Number of distinct query texts	Number of distinct query shapes	Potential issue
LOW	LOW	LOW	No problem
HIGH	LOW	LOW	Problems with excessive compiles, potentially affecting plan cache and predictable performance
HIGH	HIGH	LOW	Problems with excessive number of non-parameterized queries (similar but larger problems as above)
HIGH	HIGH	HIGH	Could be a completely ad-hoc workload such as random query generator, which is extremely rare; more likely due to excessive table name versioning or excessive schema use.

© SQLskills, All rights reserved.
http://www.SQLskills.com

Workload Analysis (1)

Data	Count
Number of Query Text Rows (query_text_id)	709719
Number of Query Rows (query_id)	709833
Number of Distinct Query Rows/Shapes (query_hash)	4033
Number of Plans (query_plan_id)	721567
Number of Distinct Plans/Shapes (query_plan_hash)	7003

- LOW number of distinct queries
- LOW number of distinct plans
- Compiles probably high, potentially affecting plan cache and predictable performance

© SQLskills, All rights reserved.
http://www.SQLskills.com

Workload Analysis (2)

Data	Count
Number of Query Text Rows (query_text_id)	581905
Number of Query Rows (query_id)	582140
Number of Distinct Query Rows/Shapes (query_hash)	347292
Number of Plans (query_plan_id)	596200
Number of Distinct Plans/Shapes (query_plan_hash)	216565

- HIGH number of distinct queries
- HIGH number of distinct plans
- Extremely random workload (e.g. table versioning)

© SQLskills, All rights reserved.
http://www.SQLskills.com

To be clear…

- If you enable Query Store and find that query duration, CPU, etc. increases, you need to look at your workload
- On *any* version of SQL Server, you run the risk of running into performance issues because of the way your workload is designed
- However, with SQL 2016+, there are ways to detect these issues by using Query Store, but it doesn't mean that these issues weren't present in your workload before the upgrade
- Query Store Performance Overhead…Updated
  - https://www.sqlskills.com/blogs/erin/query-store-performance-updated/

© SQLskills, All rights reserved.
http://www.SQLskills.com

Options for Addressing AdHoc Statements

- Parameterization via code changes
- Use “templatized” plan guides
- Enable FORCED parameterization at the database level
  - This is typically NOT something we recommended
  - **Many** more statements will be parameterized
    - A benefit is that will reduce plan cache bloat and the number of queries in Query Store (decreased compilation, decreased CPU)
    - A drawback is that if you have variability in your data and parameter sensitivity, performance can start to vary wildly
    - Also: this applies to ALL queries in the database

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Tuning: Forcing Plans

© SQLskills, All rights reserved.
http://www.SQLskills.com

How do you fix a poorly-performing query?

Change code and/or schema

Add RECOMPILE

Manually get the “best” plan in cache

Use a plan guide

Force a plan in Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Plan Guides

- Multiple options exist for plan guide creation (OBJECT, SQL, TEMPLATE)
- Allow you to add hints without changing existing code, and allow you to parameterize ad-hoc queries
- Schema-bound
- Can be tricky to implement
- It is not always easy to understand why a plan guide is not being used
- Monitor success/failure with Trace or Extended Events

© SQLskills, All rights reserved.
http://www.SQLskills.com

Forcing Plans with Query Store

- Query Store allows you to easily find queries with multiple plans and force one plan
- Not schema-bound
- Monitor failures with Extended events
- If a plan is no longer optimal, Query Store will continue to use it, unless you un-force it or forcing fails

A bad plan is not the one which failed, but the one which succeeded at the greatest cost.

-Anonymous DBA

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Creating Plan Stability

© SQLskills, All rights reserved.
http://www.SQLskills.com

Image attribution: https://msdn.microsoft.com/en-us/library/mt631173.aspx

© SQLskills, All rights reserved.
http://www.SQLskills.com

Plan Forcing Internals

- Plan forcing in Query Store is a wrapper around the USE PLAN query hint
  - This is a best effort; there are no guarantees
  - Theoretically, this should shorten the optimization phase
- When you initially force a plan, a recompile of the query is triggered on the next execution (to get the forced plan into cache, and this evicts the previous plan)
  - **Any** plan forcing change (force or unforce, automatic or manual) for a query triggers recompilation on its next execution, even if there is an existing plan for that query already in cache

© SQLskills, All rights reserved.
http://www.SQLskills.com

Plan Forcing Internals

- Optimizer goes through different trees and finds the "best" plan
  - It has to validate the plan in some way
  - This is done the first time the query is executed (compiled), if the forced plan is not already in cache
  - Subsequent query executions reuse the plan in cache, like every other query, until the query is recompiled or evicted from cache
- If the forced plan fails, the optimizer will choose a different plan
- The Query Optimizer is a factor here – if forcing doesn’t work, don't blame Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Why isn't a forced plan used?

- There are cases where the optimizer will choose a plan that has the same shape as the one that is forced - it's morally equivalent (and doesn't show up as a forcing failure).
- In other scenarios, the optimizer cannot use the plan that was forced
  - Schema changes
  - TIME_OUT

© SQLskills, All rights reserved.
http://www.SQLskills.com

Comparison

PLAN GUIDES

- Can use a plan guide for an ad hoc query or stored procedure
- Schema bound
- Adding a hint in a plan guide does not alter the query text
- Can be tricky to implement
- Understanding failures can be difficult

FORCING PLANS

- Can force a plan for an ad hoc query* or stored procedure
- Not schema bound
- Adding a hint changes the query text
- Very simple to implement
- Understanding failures is easier

© SQLskills, All rights reserved.
http://www.SQLskills.com

Points to Remember with Plan Forcing(1)

- It may not always be obvious that a plan is forced – check the actual plan and Query Store to determine
- Query performance can be different across environments for multiple reasons – including forced plans!
- Pay attention to forced plans when testing code and schema changes
  - Changing index names
  - Changing object names
- A forced plan overrides a plan guide*

© SQLskills, All rights reserved.
http://www.SQLskills.com

Points to Remember with Plan Forcing (2)

- You cannot force a plan on read-only secondary by forcing the plan on the primary
- You cannot force a plan for a query if it hasn't been generated by that query
- Turning off Query Store negates the ability to use forced plans
  - They cannot be removed manually, until un-forced
- Forced plans will not be aged out of Query Store
- *Be aware that forced plans are removed if you installed SQL Server 2017 CU2 and then upgrade to a later CU*
  - https://www.sqlskills.com/blogs/erin/query-store-fix-in-sql-server-2017/

© SQLskills, All rights reserved.
http://www.SQLskills.com

Typical Reasons Forcing Can Fail

- Drop an index (NO_INDEX)
- Change an index name (NO_INDEX)
- Remove columns from an index (NO_PLAN)
- Change the object_id due to DROP/CREATE rather than ALTER

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store plan example from class:
Query 1464 would run well the majority of the time, but then it would generate plan 3484 which was extremely slow comparatively. The query executed on a regular basis as part of a screen refresh, so this created problems for users and created high CPU on the server.

Developers decided to add an OPTION (RECOMPILE) to the query (which created a new query_id, 3320). This seemed to address the issue based on the new plans.

Solutions here could also include forcing the plan (e.g. force plan 1838 for the original query), or re-writing the query. It's important to understand WHY different plans were generated.

© SQLskills, All rights reserved.
http://www.SQLskills.com

Automatic Tuning

- Reduces manual intervention required from data professionals
- Monitors workload performance, makes changes, continues to monitor and make additional changes if needed (e.g. revert)
  - Query Store must be enabled
- Two components:
  - Automatic plan correction
  - Automatic index management

© SQLskills, All rights reserved.
http://www.SQLskills.com

Automatic Tuning

- SQL Server 2017
  - Automatic Plan Correction
  - Disabled by default
- Azure SQL Database
  - Automatic Plan Correction
  - Automatic Index Management
  - Starting January 15, 2018 a change was rolled out to start to enable Automatic Tuning for all existing Azure SQL DBs
    - Users notified in advance
  - Eventually will be enabled by default for all new databases

© SQLskills, All rights reserved.
http://www.SQLskills.com

Automatic Plan Correction

- Available in SQL Server 2017 and Azure SQL Database
- Only available in Enterprise Edition
- Disabled by default

```
ALTER DATABASE <database_name_here>
SET AUTOMATIC_TUNING ( FORCE_LAST_GOOD_PLAN = ON );
```

- Can use the information captured to make corrections manually
  - Stored in sys.dm_db_tuning_recommendations
  - Recommendations will not appear in Standard Edition

© SQLskills, All rights reserved.
http://www.SQLskills.com

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Automatic Plan Correction

© SQLskills, All rights reserved.
http://www.SQLskills.com

Can I trust it?

- It is not perfect, but it has been developed with operational telemetry from Azure SQL Database implementations
- It may not “catch” every regression you expect, and it may make a not-so-great decision
- Its ability to recover from any “bad decision” is highly reliable as there is continuous validation of forced plans and automatic back-off logic built-in

© SQLskills, All rights reserved.
http://www.SQLskills.com

Notes About Forcing

- If you manually force a plan – either because you determined it was needed or based on a recommendation from sys.dm_db_tuning_recommendations – it will never be automatically un-forced
- Only plans that are forced with the Automatic Plan Correction feature will be automatically un-forced
- APC only considers recent plans, older plans may exist for a query that seem "better" than what is selected
  - Consider: the older a plan is, the ability to reliably predict if past performance is the same now/in the future decreases

© SQLskills, All rights reserved.
http://www.SQLskills.com

Monitoring Automatic Plan Correction

- Information in sys.dm_db_tuning_recommendations is lost on instance restart
  - Snapshot to a table if you want to retain information
- Create an Extended Events session that captures automatic tuning events, writes to an event_file target, and starts when the instance starts (always running)
  - automatic_tuning_error
  - automatic_tuning_plan_regression_detection_check_completed
  - automatic_tuning_plan_regression_verification_check_completed
  - automatic_tuning_recommendation_expired

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Tuning: Finding Patterns

© SQLskills, All rights reserved.
http://www.SQLskills.com

Mining Data from Query Store

- Very common to use the plan cache to find problems and potential issues
- Examples:
  - Implicit conversions
  - Specific operators used in plans
  - Missing indexes
  - Specific index use
- Largest issue with using the plan cache: it's transient

© SQLskills, All rights reserved.
http://www.SQLskills.com

Using sys.query_store_plan

- The same information from the plan cache is now persisted in sys.query_store_plan
- The query plan is stored as VARBINARY(MAX)
  - This means you don't have to know XQuery…unless you want to find specific operations or details
- The XML for query plans is bound to a published schema
  - Converting to XML allows you to parse the data efficiently
  - http://schemas.microsoft.com/sqlserver/2004/07/showplan/

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Mining Query Store Plans

© SQLskills, All rights reserved.
http://www.SQLskills.com

Other Uses of Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Using Query Store for Testing

- Query Store captures data that can be used as a baseline for testing:
  - Hardware upgrades
  - Code changes
  - Changes in the Cardinality Estimator
- While data collection is built in to SQL Server, analysis is not
- A restored copy of a production database can be used for testing

© SQLskills, All rights reserved.
http://www.SQLskills.com

Testing Hardware Upgrades

- Execute code in production, data is captured in Query Store
- Backup the database
- Restore the database in the new environment
- Execute the code
- Look for changes via Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Testing Code Changes

- Execute code in production, data is captured in Query Store
- Backup the database
- Restore the database in the new environment
- Make code changes
  - Any changes to stored procedures, functions, etc. should be done using ALTER, not DROP/CREATE
  - When testing index changes, be aware that the index name is what is stored in the plan
- Execute the code
- Look for changes via Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Testing CE and Compat Mode Changes

- Execute code in production, data is captured in Query Store
- Backup the database
- Restore the database in the new environment
- Change the CE version or Compatibility Mode version
- Confirm if any plans are forced, and if they should still be forced for testing
- Execute the code
- Look for changes via Query Store
  - Queries with a new/additional plan

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Testing Considerations

© SQLskills, All rights reserved.
http://www.SQLskills.com

Additional Considerations

- DBCC CLONEDATABASE can be used for testing, as Query Store data can be retained in the clone
  - Testing is done *without* data in the database, therefore performance metrics are not useful, only plan information
  - https://sqlperformance.com/2017/02/sql-performance/clonedatabase-query-store-testing
- Query Store can also be used in QA/Dev/Test databases for testing
  - If these databases are restored nightly, Query Store data is lost unless you export it manually and store/reference it elsewhere
- Values for query_id and plan_id can/will be different

© SQLskills, All rights reserved.
http://www.SQLskills.com

Methods for Replaying Workloads(1)

- Profiler
  - Replays a trace file using Profiler from a single client
    - https://docs.microsoft.com/en-us/sql/tools/sql-server-profiler/replay-traces?view=sql-server-2017
- Distributed Replay
  - Replays a trace file using DRU services from up to 16 clients
  - Requires Enterprise Edition to replay from more than one client
    - https://www.sqlskills.com/blogs/jonathan/category/distributed-replay/
  - Most appropriate for SQL Server 2012 through SQL Server 2016

© SQLskills, All rights reserved.
http://www.SQLskills.com

Methods for Replaying Workloads(2)

- Database Experimentation Assistant
  - Graphical tool (built on DRU) to replay and evaluate a workload after a change (A/B testing)
  - https://www.microsoft.com/en-us/download/details.aspx?id=54090
- Ostress
  - Component of RML Utilities that allows replay from multiple sessions
    - https://support.microsoft.com/en-us/help/944837/description-of-the-replay-markup-language-rml-utilities-for-sql-server

© SQLskills, All rights reserved.
http://www.SQLskills.com

Uses of Query Store

- Troubleshooting query performance
- Plan Forcing
- Proactively analyze workload patterns
  - Determine patterns in coding and plan execution
- A/B testing
  - Testing query performance before an upgrade (hardware, software, application)
  - Testing changes in the Cardinality Estimator and with different compatibility modes
- New functionality will continue to build upon Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store Fixes of Note(1)

- Access violation when SQL Server 2016 tries to start Query Store during startup
  - Fixed in SQL Server 2016 SP2 CU4
    - https://support.microsoft.com/en-us/help/4052133/access-violation-when-sql-server-starts-query-store-during-startup
  - Fixed as part of improvements in SQL Server 2017
- Transactions and log truncation blocked in SQL Server 2017 when using ALTER DATABASE <*DatabaseName*> SET QUERY_STORE CLEAR
  - Fixed in SQL Server 2016 SP2 CU5 and SQL Server 2017 CU 11
    - https://support.microsoft.com/en-us/help/4461562/transactions-and-log-truncation-may-be-blocked-when-using-query-store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Other Query Store Fixes of Note(2)

- "Non-yielding Scheduler" occurs when you clean up in-memory runtime statistics in Query Store in SQL Server 2016
  - Fixed in SQL Server 2016 SP2 CU7
    - https://support.microsoft.com/en-us/help/4501205/fix-non-yielding-scheduler-occurs-when-you-clean-up-in-memory-runtime
  - Fixed as part of improvements in SQL Server 2017
- Filled transaction log causes outages when you run Query Store in SQL Server 2016 and 2017
  - Fixed in SQL Server 2016 SP2 CU8
  - Fixed in SQL Server 2017 CU 16
    - https://support.microsoft.com/en-us/help/4511715/fix-filled-transaction-log-causes-outages-when-you-run-query-store-in

© SQLskills, All rights reserved.
http://www.SQLskills.com

Other Query Store Fixes of Note(3)

- Memory limitations implemented
  - Added in SQL Server 2016 SP2 CU15, SQL Server 2017 CU22, SQL Server 2019 CU8
  - "Query Store scalability improvement for adhoc workloads. Query Store now imposes internal limits to the amount of memory, it can use and automatically changes the operation mode to READ-ONLY until enough memory has been returned to the Database Engine, preventing performance issues."

© SQLskills, All rights reserved.
http://www.SQLskills.com

Resources

- SQL Server: Introduction to Query Store
  - https://www.pluralsight.com/courses/sqlserver-query-store-introduction
- Automatic Tuning in SQL Server 2017 and Azure SQL Database
  - https://www.pluralsight.com/courses/sqlserver-azure-database
- Blog:
  - https://www.sqlskills.com/blogs/erin/category/query-store/
- Post from Kendra about “morally equivalent” plans
  - https://sqlworkbooks.com/2018/03/what-is-a-morally-equivalent-execution-plan-and-why-is-it-good/

© SQLskills, All rights reserved.
http://www.SQLskills.com

Review

- Query Store Fundamentals
- Understanding the Query Store Data
- Finding Performance Issues
- Query Store Performance
- Query Tuning
  - Plan Forcing
  - Finding Patterns
- Other Uses of Query Store

© SQLskills, All rights reserved.
http://www.SQLskills.com

Thank you!

SQLskills
immerse yourself in sql server