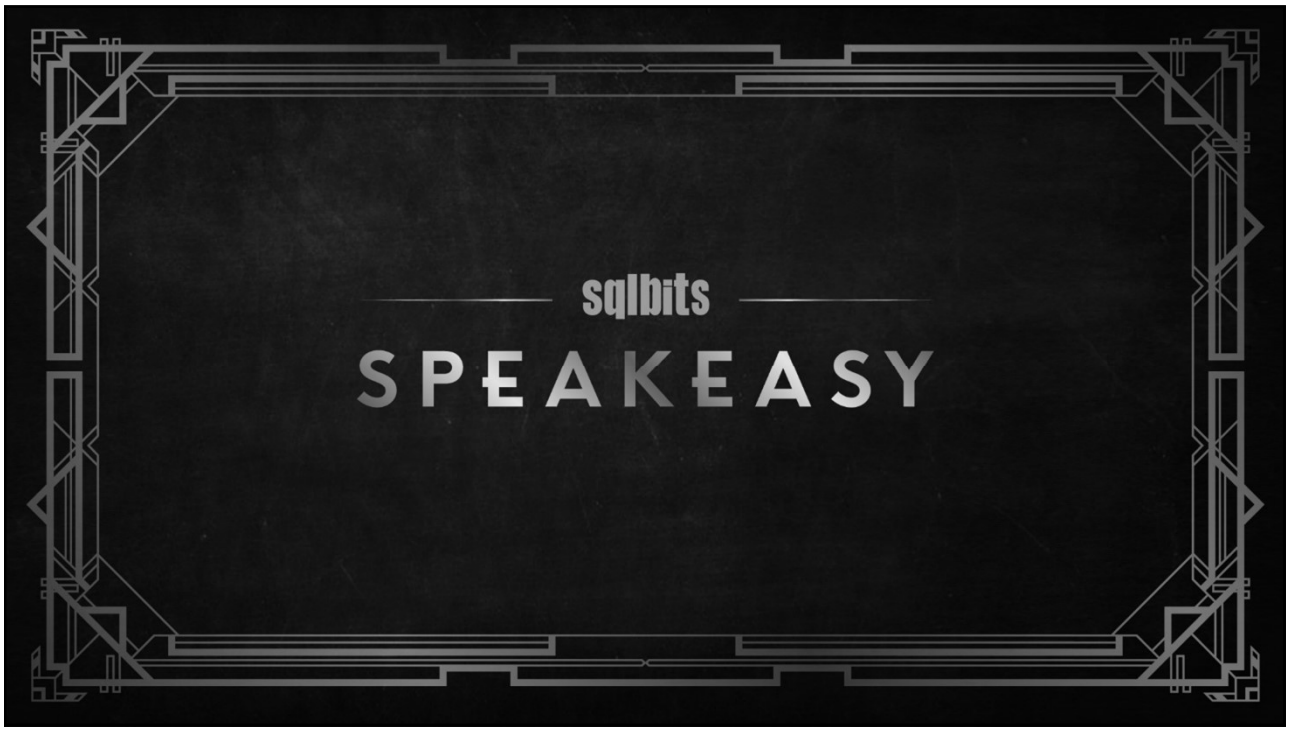


SQL Server Query Store

Kalen Delaney
www.SQLServerInternals.com

sqlbits

MANCHESTER 2019

Kalen Delaney

www.SQLServerInternals.com

- /kalendelaney
- @sqlqueen
- SQLInternals

Working with SQL Server for over 31 years!

- Employed by Sybase 1987 - 1992
- Contracted by MS for course development, course delivery, writing books and whitepapers
- SQL Server MVP for 25 years

Author of numerous SQL Server books, most recently:

- Author: SQL Server In-Memory OLTP Internals (RedGate, 2014, 2017)
- Primary Author: SQL Server 2012 Internals (O'Reilly, 2013)

Teaching and troubleshooting for clients around the world

- 5-day and 3-day SQL Server Internals classes
- Troubleshooting query and application performance issues

© Kalen Delaney, 2019

Problems With Query Performance

Website Is down — SQL Server is not available

Temporary perf issues — Impossible to predict / root cause

DB upgraded — Regression caused by new bits

Fixing query plan choice regressions is difficult

- Query plan cache is not well-suited for performance troubleshooting
- Cleared out when SQL Server comes down

Long time to detect the issue (TTD)

- Which query is slow? Why is it slow?
- What was the previous plan?

Long time to mitigate (TTM)

- Can I modify the query?
- How to use plan guide?

Plan choice change can cause these problems

© Kalen Delaney, 2019

The Solution: Query Store

Dedicated store for query workload performance data

- Captures the history of plans for each query
- Captures the performance of each plan over time
- Persists the data to disk (works across restarts, upgrades, and recompiles)

Significantly reduces TTD/TTM

- Find regressions and other issues in seconds
- Allows you to force previous plans from history

DBA is now in control

Query Store Feature

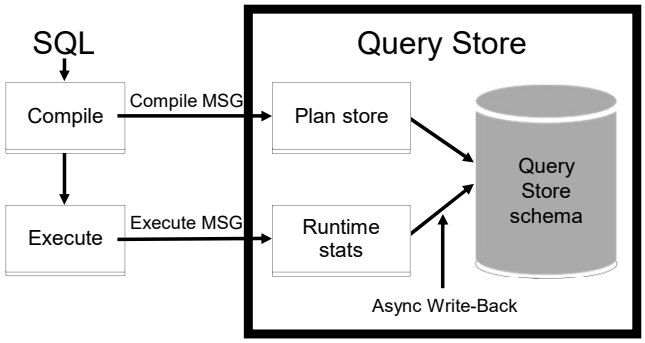
Planned since SQL Server 2008

Added in Azure SQL Database prior to release of SQL Server 2016

On-prem version SQL Server 2016 included Query Store

All metadata was available in SQL Server 2014

Query Data Store



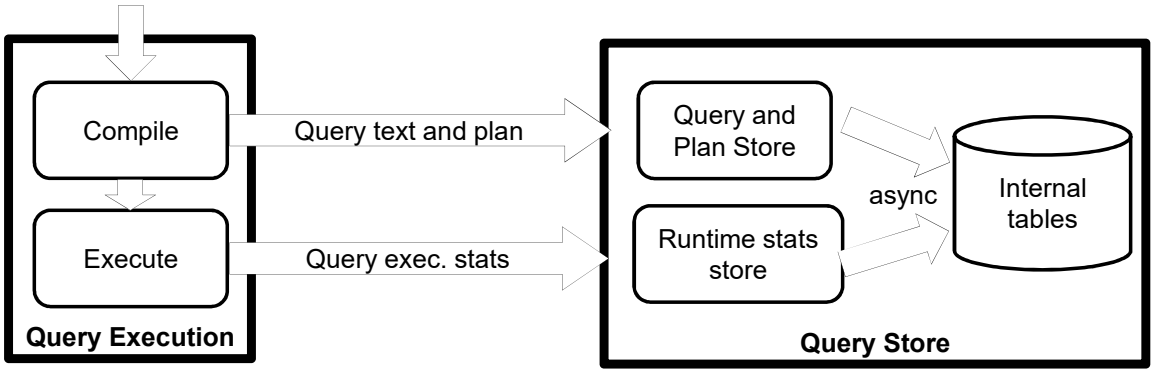
Durability latency controlled by DB option DATA_FLUSH_INTERNAL_SECONDS

- Collects query texts (plus all relevant properties)
- Stores all plan choices and performance metrics
- Works across restarts / upgrades / recompiles
- Dramatically lowers the bar for performance troubleshooting
- New Views
- Intuitive and easy plan forcing

Query Store Write Architecture

Query Store captures data in-memory to minimize I/O overhead

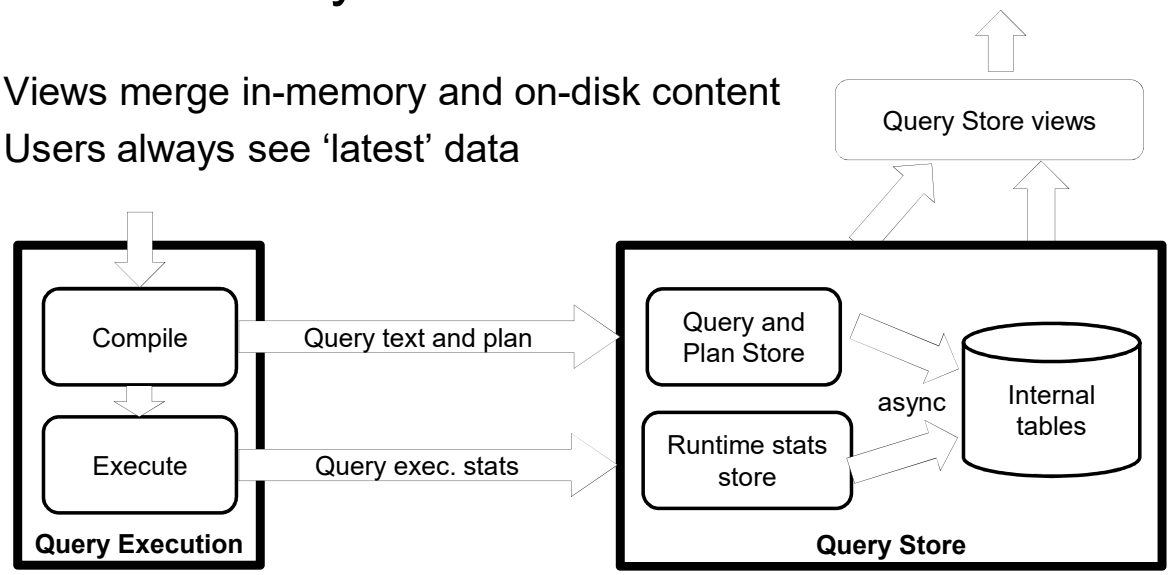
Data is persisted to disk asynchronously in the background



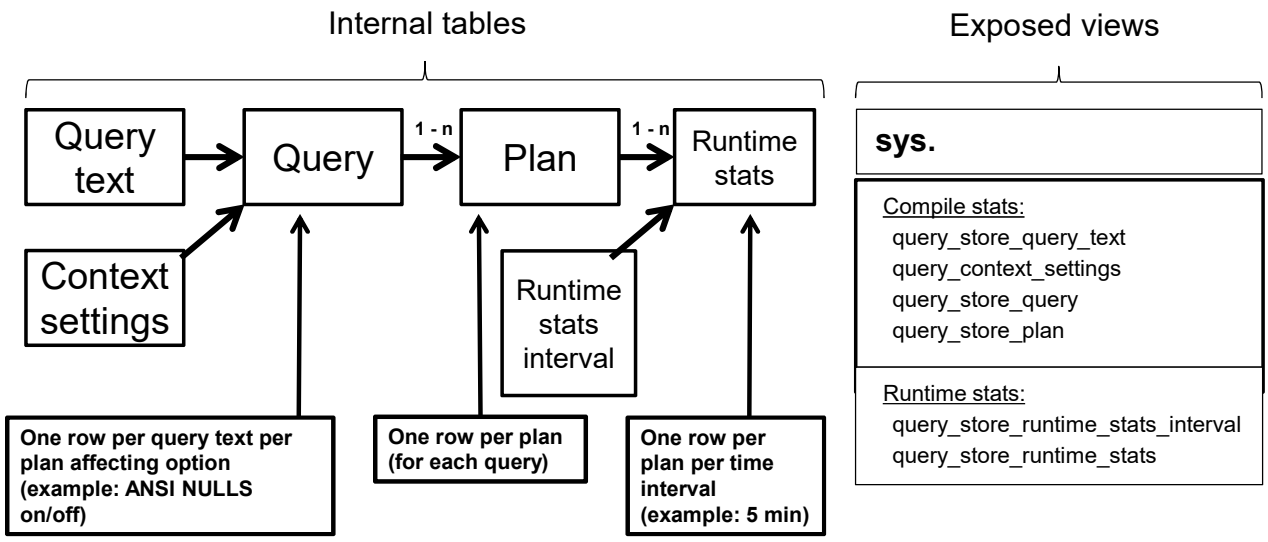
Query Store Read Architecture

Views merge in-memory and on-disk content

Users always see 'latest' data

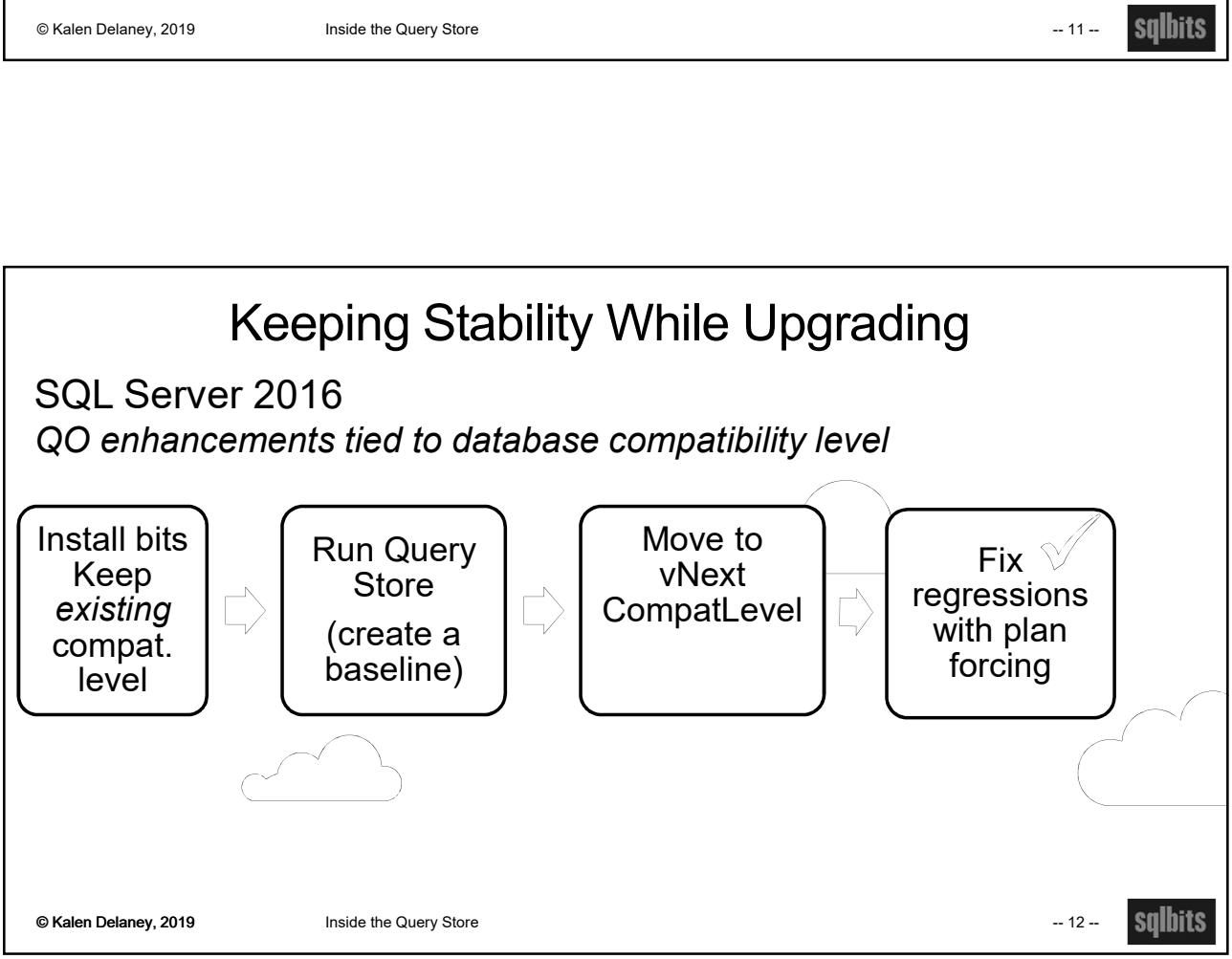


Query Store Schema

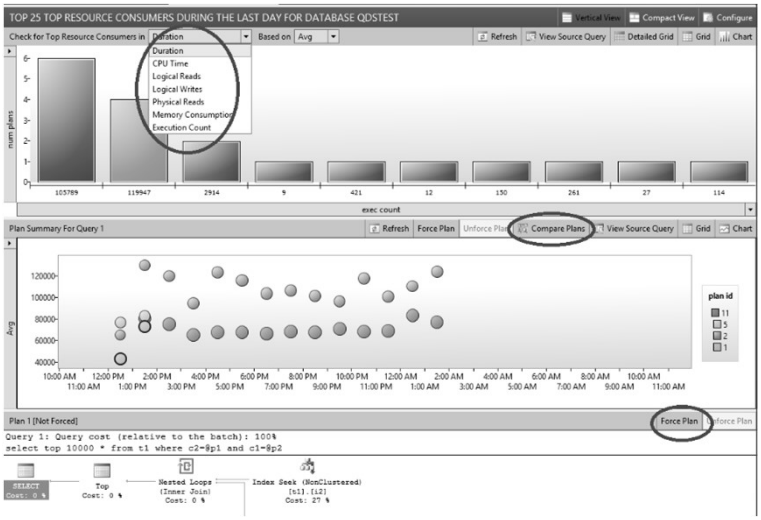


Query Store Storage

- Database Specific
- Control options per database
- Can be backed up and restored!



Monitoring Performance By Using the Query Store



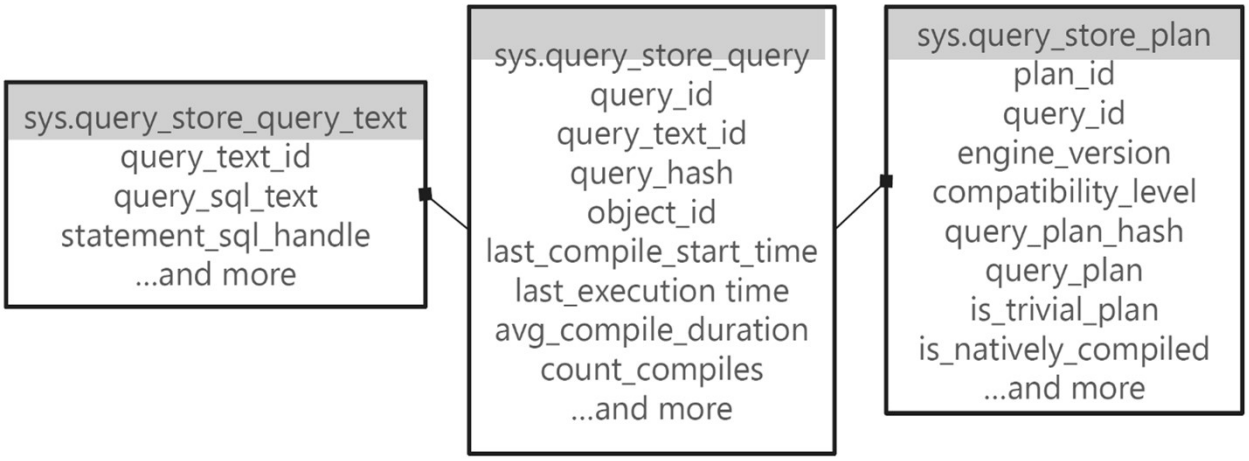
The Query Store feature provides DBAs with insight on query plan choice and performance

Controlling the Query Store

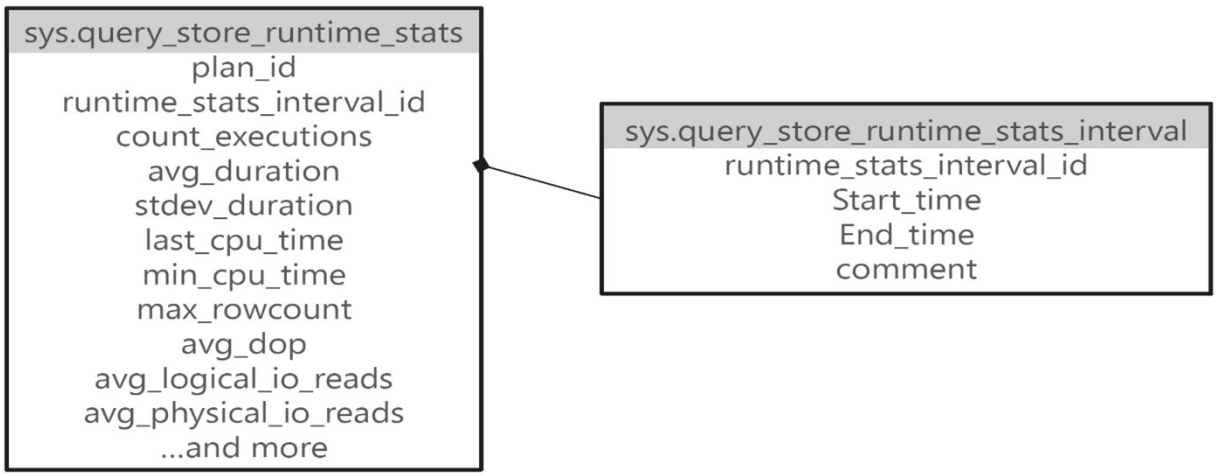
```
/* Turn ON Query Store */

ALTER DATABASE MyDB SET QUERY_STORE = ON;

/* Review current Query Store parameters */
SELECT * FROM  sys.database_query_store_options

/* Set new parameter values */
ALTER DATABASE MyDB
SET QUERY_STORE (
  OPERATION_MODE = READ_WRITE,
  CLEANUP_POLICY = (
    STALE_QUERY_THRESHOLD_DAYS = 30  ),
  DATA_FLUSH_INTERVAL_SECONDS = 3000,
  MAX_SIZE_MB = 500, INTERVAL_LENGTH_MINUTES = 15);

/* Clear all Query Store data */
ALTER DATABASE MyDB SET QUERY_STORE CLEAR;

/* (5) Turn OFF Query Store */
ALTER DATABASE MyDB SET QUERY_STORE = OFF;
```

- DB-level feature exposed through T-SQL extensions
- **ALTER DATABASE**
- Catalog views (settings, compile, and runtime stats)
- Stored Procs (plan forcing, query/plan/stats cleanup)

Plan Store Views



Runtime Views



Examining Query Store Metadata

```
/* Performance analysis using Query Store views*/
SELECT q.query_id, qt.query_text_id, qt.query_sql_text,
        SUM(rs.count_executions) AS total_execution_count
FROM
    sys.query_store_query_text qt
    JOIN sys.query_store_query q
          ON qt.query_text_id = q.query_text_id
    JOIN sys.query_store_plan p
          ON q.query_id = p.query_id
     JOIN sys.query_store_runtime_stats rs
          ON p.plan_id = rs.plan_id
GROUP BY q.query_id, qt.query_text_id, qt.query_sql_text
ORDER BY total_execution_count DESC;

/* Force plan for a given query */
EXEC sp_query_store_force_plan  12 /*@query_id*/, 14 /*@plan_id*/
```


New in SQL Server 2017

- Waitstats, grouped by ‘category’
  sys.query_store_wait_stats

Integer value	Wait category	Wait types include in the category
1	CPU	SOS_SCHEDULER_YIELD
2	Worker Thread	THREADPOOL
3	Lock	LCK_M_%
4	Latch	LATCH_%
5	Buffer Latch	PAGELATCH_%
6	Buffer IO	PAGEIOLATCH_%
7	Compilation	RESOURCE_SEMAPHORE_QUERY_COMPILE
15	Network IO	ASYNC_NETWORK_IO, NET_WAITFOR_PACKET
16	Parallelism	CXPACKET, EXCHANGE
17	Memory	RESOURCE_SEMAPHORE, CMEMTHREAD, CMEMPARTITIONED, EE_PMOLOCK, MEMORY_ALLOCATION_EXT, RESERVED_MEMORY_ALLOCATION_EXT, MEMORY_GRANT_UPDATE

Query Store Summary

- Capability
  - Query Store helps customers quickly find and fix query performance issues
  - Query Store is a ‘flight data recorder’ for database workloads
- Benefits
  - Greatly simplifies query performance troubleshooting
  - Provides performance stability across SQL Server upgrades
  - Allows deeper insight into workload performance

Thank You!

FRIDAY NIGHT, SQLBITS

PROHIBITION PARTY

FEEDBACK FORMS

PLEASE FILL OUT AND PASS TO YOUR ROOM
HELPER BEFORE YOU LEAVE THE SESSION