Query Store Basics: A DBA's Best Friend

Ever wondered why your queries are running differently today than they did last week? Query Store in SQL Server can help you figure that out! In this session, we'll break down the basics of Query Store, designed especially for junior or accidental DBAs.

We'll talk about what Query Store does, how it works and how it evolved over time, and why it's such a great tool for keeping track of query performance over time. With explanations and demos, you'll learn how to use Query Store to spot performance issues, dig into query execution plans, and even prevent bad plans from ruining your day.

If you're new to SQL Server or just looking for a practical way to optimize query performance, this session will help you to get started.

MONIN
Database Managed Services

Peter Kruis

https://www.linkedin.com/in/peter-kruis

peterkruis@hotmail.com / peter.kruis@monin-it.be

https://github.com/peterkruis

SeaQL Netherlands: https://seaql.nl

MONIN
Database Managed Services

Before Query Store…

Mondays…

“My query was fast last week, and now it’s slow! Who broke it?!” or.. “What have you done that it is broken”

Ugh… let me go dig through the plan cache… oh wait, it’s been flushed. Guess I’ll need to capture it next time it runs!

“I swear I didn't change anything, but now it's running terribly!”

Did someone rebuild indexes, update stats, or change a setting? No? I don’t have time to play 'Guess the Execution Plan' again!

“Yesterday’s execution plan was perfect. Today? A disaster!”

I hope someone saved it manually…
because SQL Server sure didn’t!

“The query runs fine on my machine… must be the database!”

Ah yes, the classic developer excuse! Should we use your laptop in production? Are you running it with the same parameters? Same data volume? No? Thought so.

“We upgraded SQL Server, and now everything is slower… what happened?!”

Oh boy, the upgrade probably changed the cardinality estimator or optimizer behavior. Time to roll back, test, or pray.

And then… SQL Server 2016 Was Released!

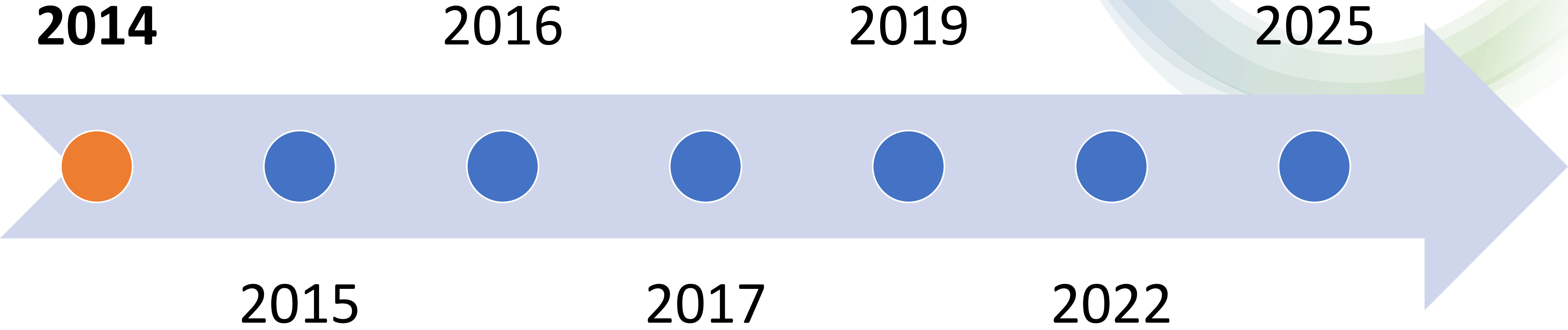
And it came with Query Store..
But.. What is it, and how does it help you?

- Black box of what happens within your database(s)
- Captures execution plans automatically over time
- Helps identifying plan changes
- Allows plan forcing
- Stores performance history

Remarks

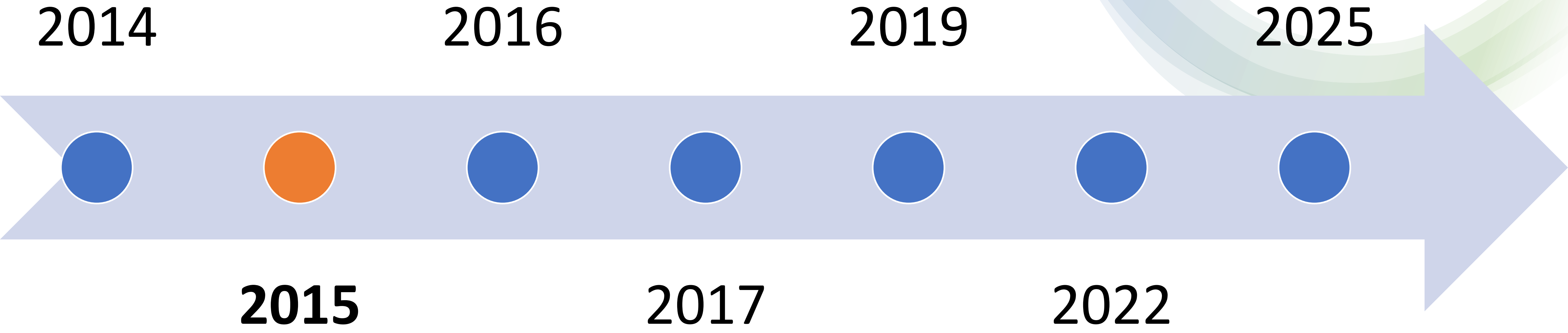
- It must be enabled at database level
- Cannot be enabled for master or tempdb
    - *model database can be done using T-SQL*
- Enabled by default for Azure SQL DB and Azure SQL Managed Instance databases
- Enabled for NEW databases by default starting SQL Server 2022

Evolution



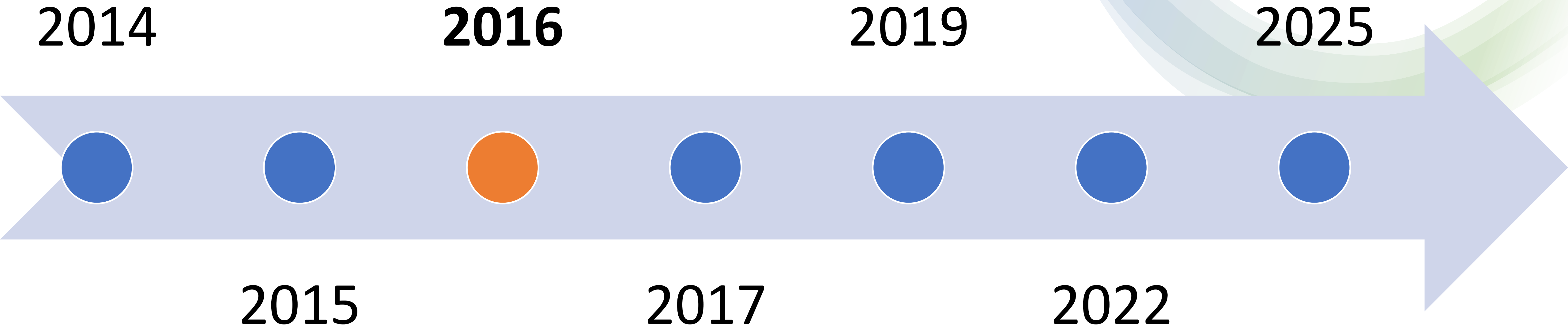
- Initial preview
    - Concept + vision

Evolution



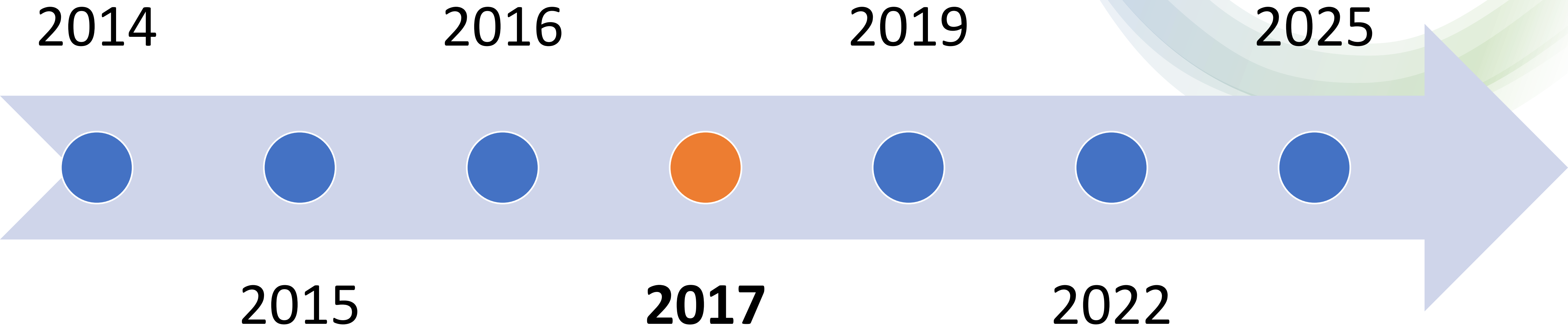
- March: Public Preview within Azure SQL DB
- Mid-year: GA within Azure SQL DB

Evolution



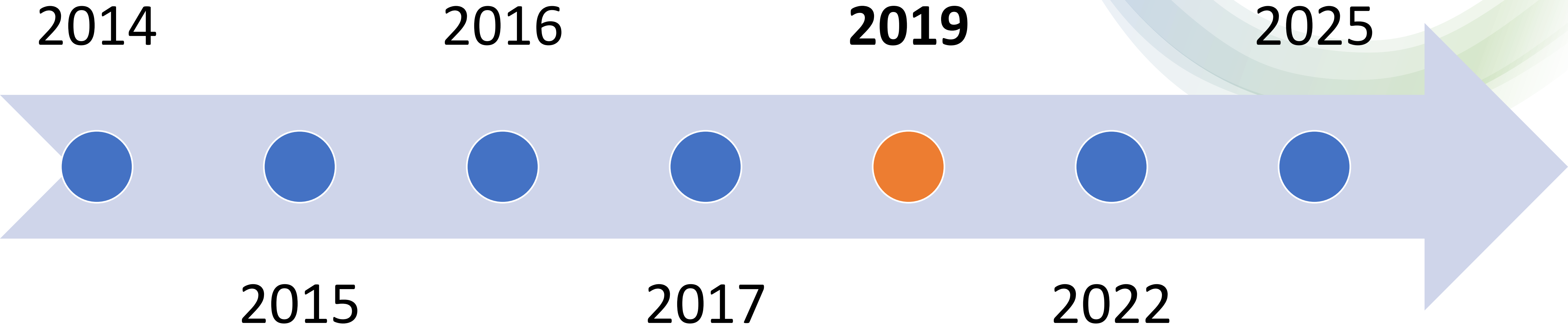
- June: SQL Server 2016
- GUI Support within SSMS
- Plan forcing
- Capture policies

Evolution



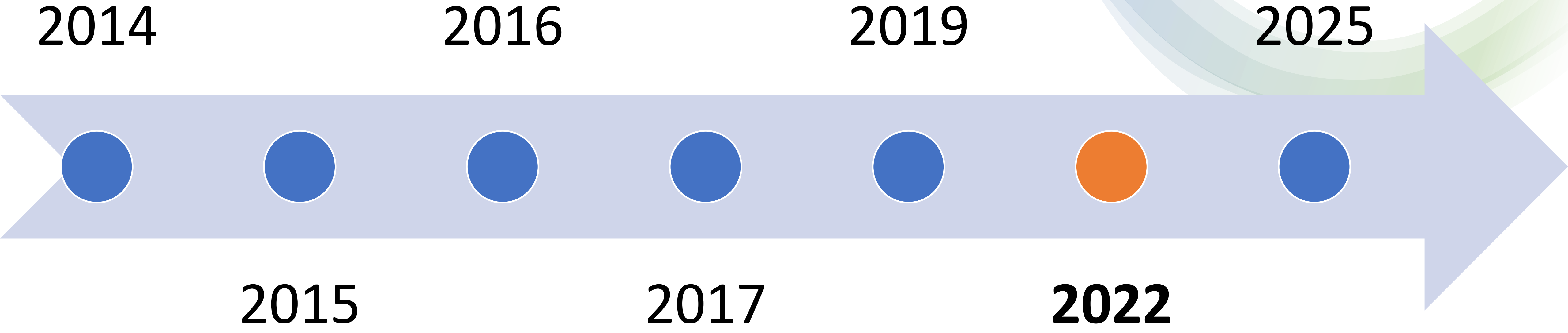
- Query store wait stats
- (Performance) Optimizations

Evolution



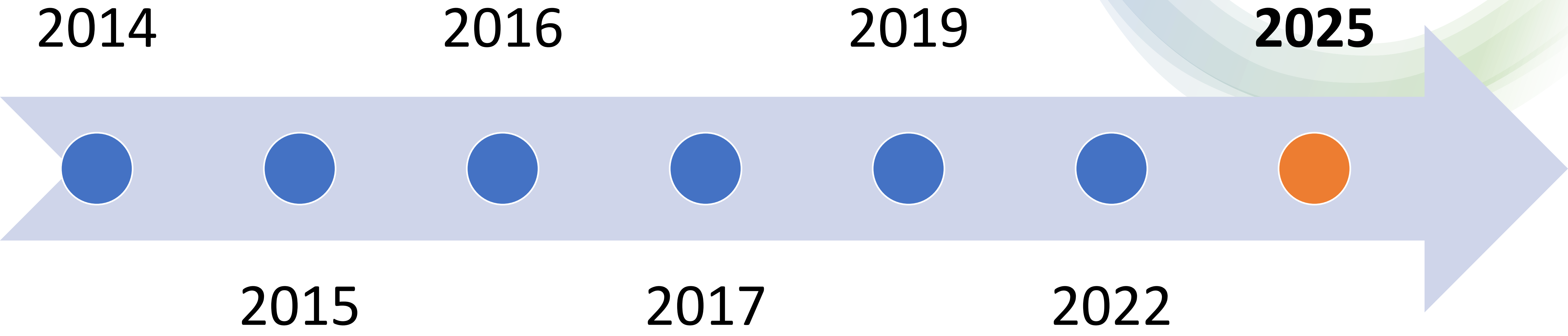
- Custom plan capture policies
- Memory and concurrency improvements

Evolution



- Enabled by default for new databases
- Automatic plan correction
- Integration with automatic tuning
- Inject query hints
- Query store on readable secondaries (preview)
  - Use SSMS 21.X

Evolution



- Integration with Intelligent Query Processing
- Degree of Parallelism auto adjustments
- Optional Parameter Plan Optimization
- Additional hints

How does it work?

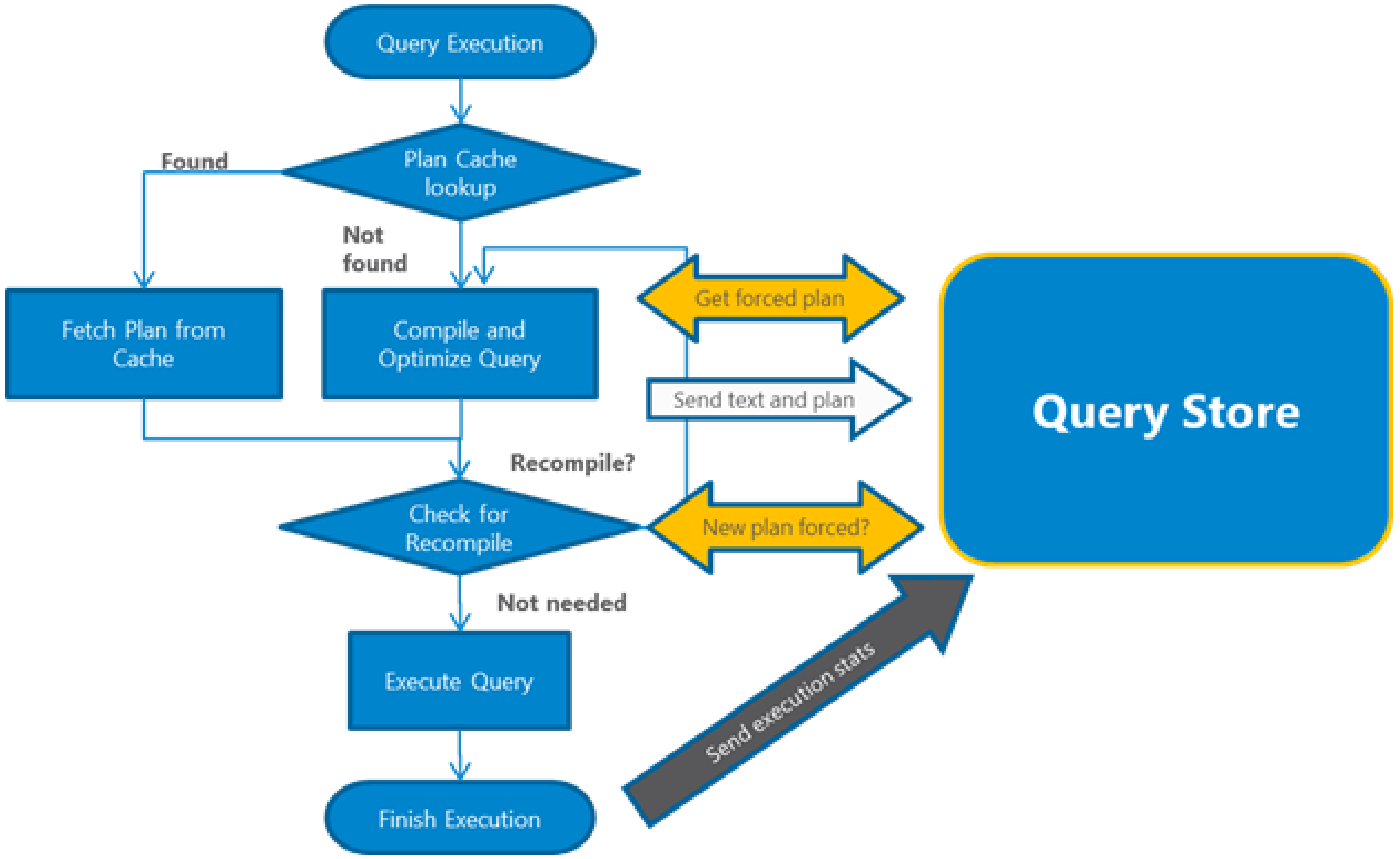
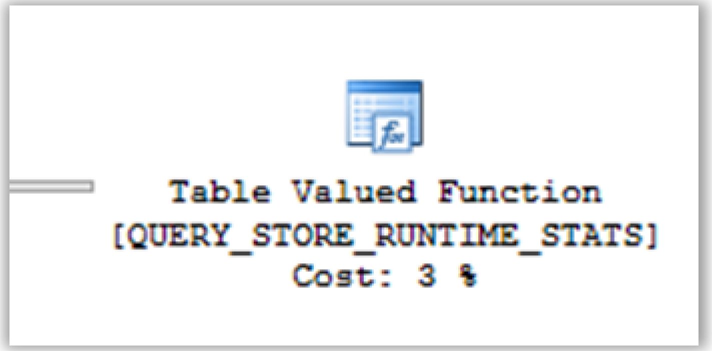
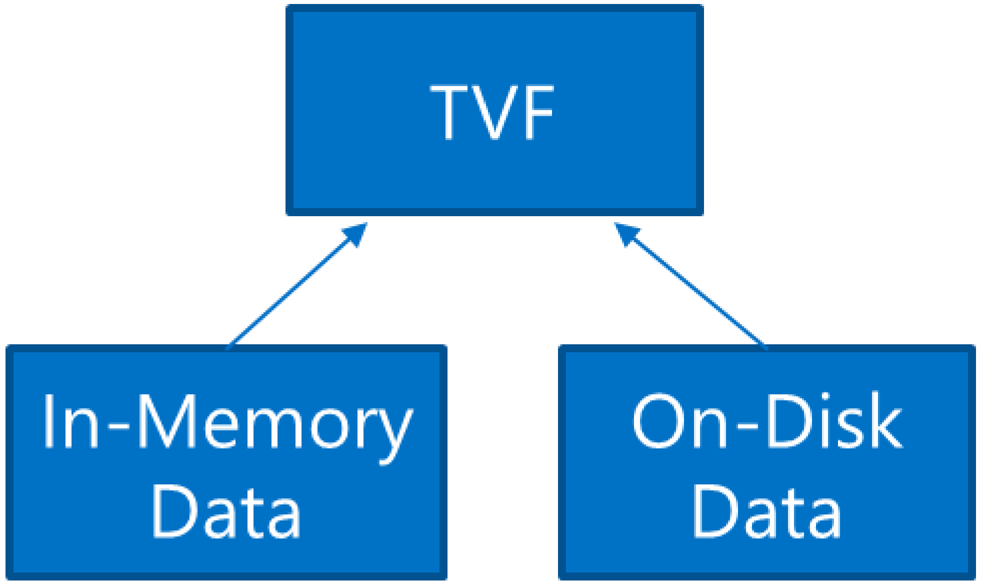


Image from https://learn.microsoft.com/en-us/sql/relational-databases/performance/how-query-store-collects-data?view=sql-server-ver16

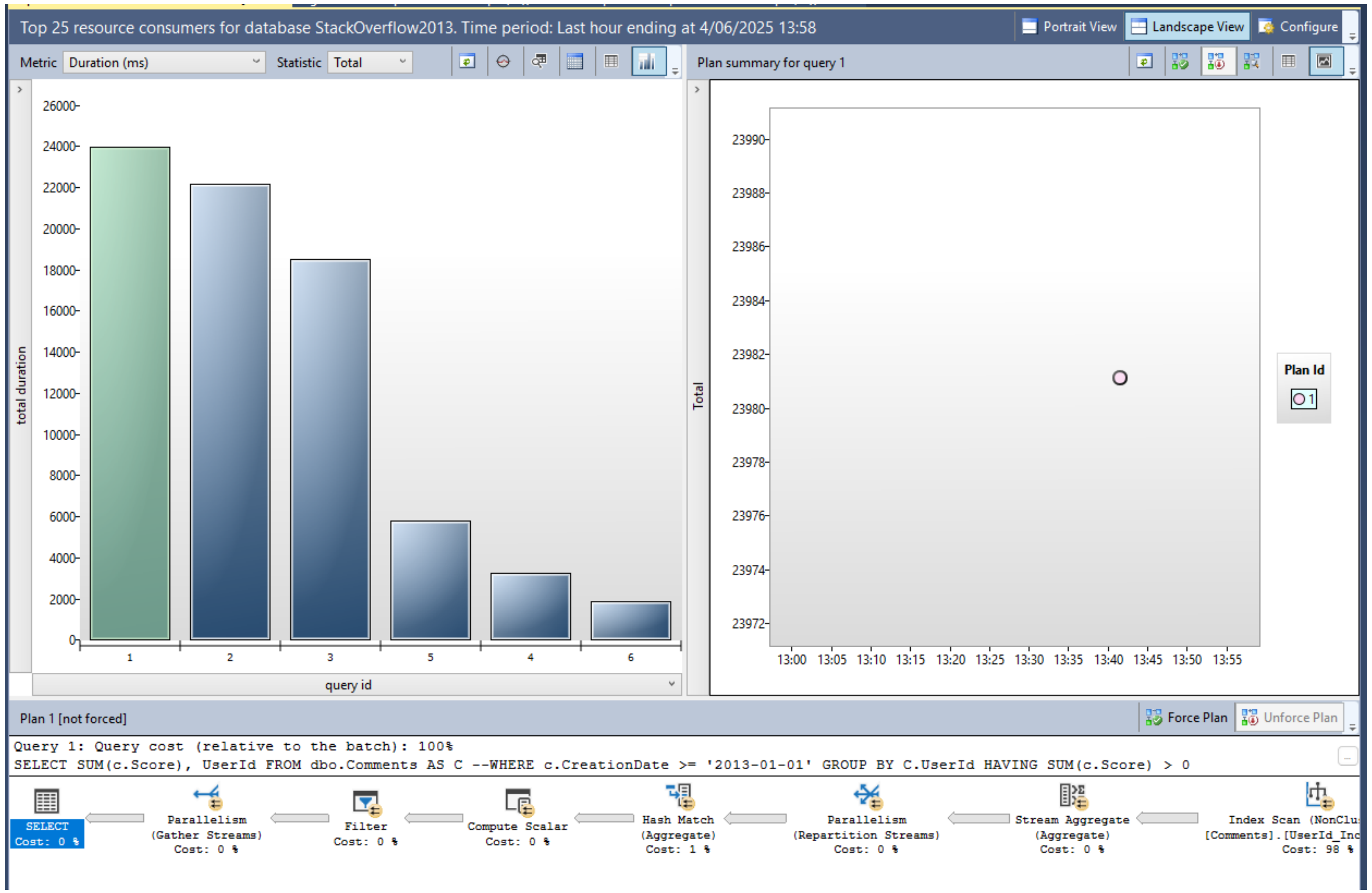
How does it work?





https://learn.microsoft.com/en-us/sql/relational-databases/performance/how-query-store-collects-data?view=sql-server-ver16

What will we see?



Top 25 resource consumers for database StackOverflow2013. Time period: Last hour ending at 4/06/2025 13:58

Plan 1 [not forced]

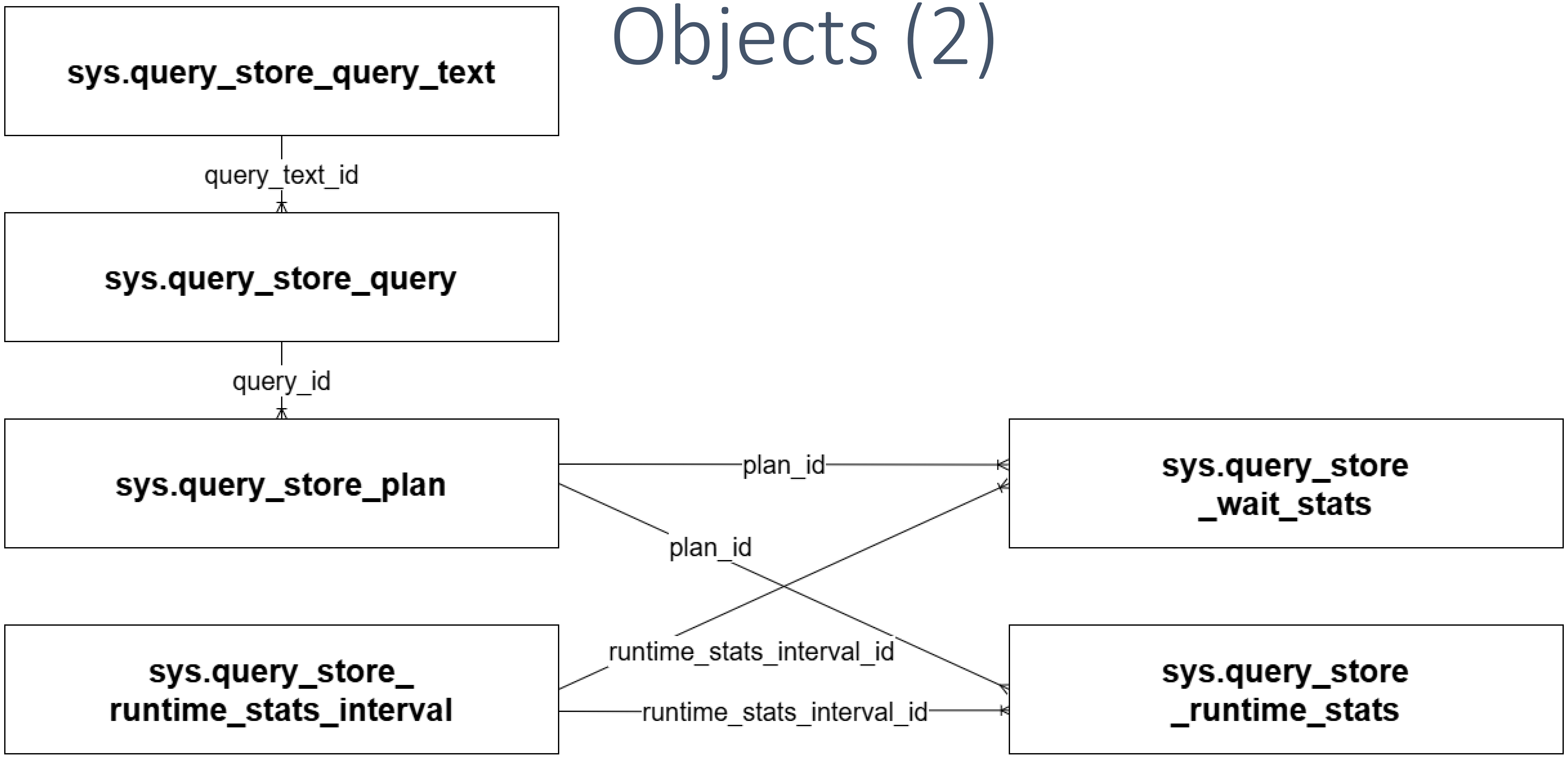
```
Query 1: Query cost (relative to the batch): 100%
SELECT SUM(c.Score), UserId FROM dbo.Comments AS C --WHERE c.CreationDate >= '2013-01-01' GROUP BY C.UserId HAVING SUM(c.Score) > 0
```

Objects (1)

Object Name	Description
sys.query_store_query	Metadata about the captured query (e.g., query ID, object ID, parameterization).
sys.query_store_query_text	The actual text of the query as executed.
sys.query_store_plan	Execution plans linked to each query.
sys.query_store_runtime_stats	Runtime execution statistics per plan per interval (CPU, IO, duration).
sys.query_store_runtime_stats_interval	Time intervals used to group runtime statistics.
sys.query_store_wait_stats	Wait statistics per plan per interval.

https://learn.microsoft.com/en-us/sql/relational-databases/performance/how-query-store-collects-data?view=sql-server-ver16

Objects (2)


sys.query_store_query_text
query_text_id
sys.query_store_query
query_id
sys.query_store_plan
plan_id
plan_id
sys.query_store_wait_stats
runtime_stats_interval_id
sys.query_store_runtime_stats_interval
runtime_stats_interval_id
sys.query_store_runtime_stats

Objects (3)

Object Name	Description
sys.query_store_plan_feedback	Stores feedback about query plans and suggestions for improvements.
sys.query_store_plan_forcing_locations	Details about where a plan forcing is applied.
sys.query_store_query_hints	Query hints applied via Query Store.
sys.query_store_query_variant	Query variants, for example due to parameterization.
sys.query_store_replicas	Replica information for AlwaysOn Availability Groups.
sys.query_context_settings	Query context settings.
sys.database_query_store_internal_state	Internal status and space usage of Query Store.
sys.database_query_store_options	Configuration options of Query Store.

Wait statistics

Group	Common wait types
CPU	SOS_SCHEDULER_YIELD, THREADPOOL
Memory	RESOURCE_SEMAPHORE, RESOURCE_SEMAPHORE_QUERY_COMPILE
I/O	PAGEIOLATCH_XX, IO_COMPLETION, ASYNC_IO_COMPLETION
Latches	PAGELATCH_XX
Locks	LCK_M_XX
Network	ASYNC_NETWORK_IO, NETWORK_IO
Parallelism	CXPACKET, CXCONSUMER
HADR/AG	HADR_SYNC_COMMIT, HADR_DATABASE_FLOW_CONTROL
Other	BROKER_RECEIVE_WAITFOR, PREEMPTIVE_XX, TRACEWRITE, etc.

Captures (1)

Captured (depends on capture mode)

- DML
- Statements
    - Stored Proc
    - Functions
    - Triggers

Not captured

- DDL
- BULK INSERT
- DBCC, KILL etc.
- Showplan
- Other database context
    - Executed from other database
    - Cross database joins
- “Who/Where”

Captures (2)

- Assuming that StackOverflow2013 has Query Store enabled.

Tracked:

```
USE StackOverflow2013
SELECT COUNT(*) FROM dbo.Users AS u
```

Not tracked:

```
USE StackOverflow2013
SELECT COUNT(*) FROM dbo.Users AS u
JOIN OtherDatabase.dbo.Users AS i ON i.id = u.Id

USE Master
SELECT COUNT(*) FROM StackOverflow2013.dbo.Users AS u
```

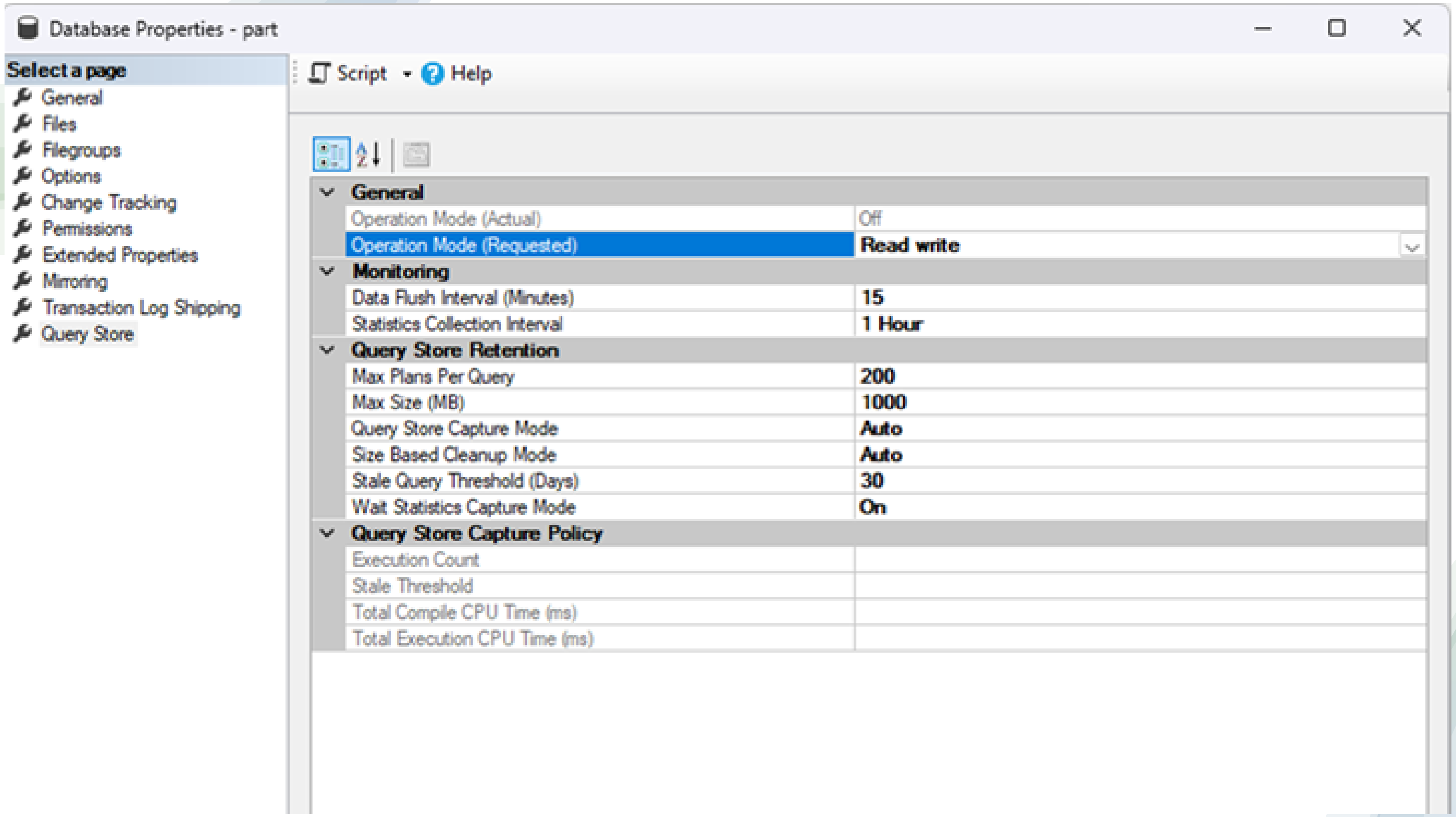
Configuring

- SSMS GUI
  - In Object Explorer, right-click a database, select **properties**
  - In the dialog box, select the **Query Store** page
  - In the **Operation Mode (Requested),** select **Read Write**

- T-SQL

  -
    ```
    ALTER DATABASE <database_name> SET QUERY_STORE = ON (OPERATION_MODE = READ_WRITE);
    ```

Configuring



Database Properties - part

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

Script · Help

General	
Operation Mode (Actual)	Off
Operation Mode (Requested)	Read write
Monitoring	
Data Flush Interval (Minutes)	15
Statistics Collection Interval	1 Hour
Query Store Retention	
Max Plans Per Query	200
Max Size (MB)	1000
Query Store Capture Mode	Auto
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)	

Configuring

```
ALTER DATABASE <database_name>
SET QUERY_STORE = ON (
    OPERATION_MODE = READ_WRITE,
    CLEANUP_POLICY =
(STALE_QUERY_THRESHOLD_DAYS = 30),
    DATA_FLUSH_INTERVAL_SECONDS = 900,
    INTERVAL_LENGTH_MINUTES = 60,
    MAX_STORAGE_SIZE_MB = 1000,
    QUERY_CAPTURE_MODE = AUTO,
    SIZE_BASED_CLEANUP_MODE = AUTO,
    WAIT_STATS_CAPTURE_MODE = ON
);
```

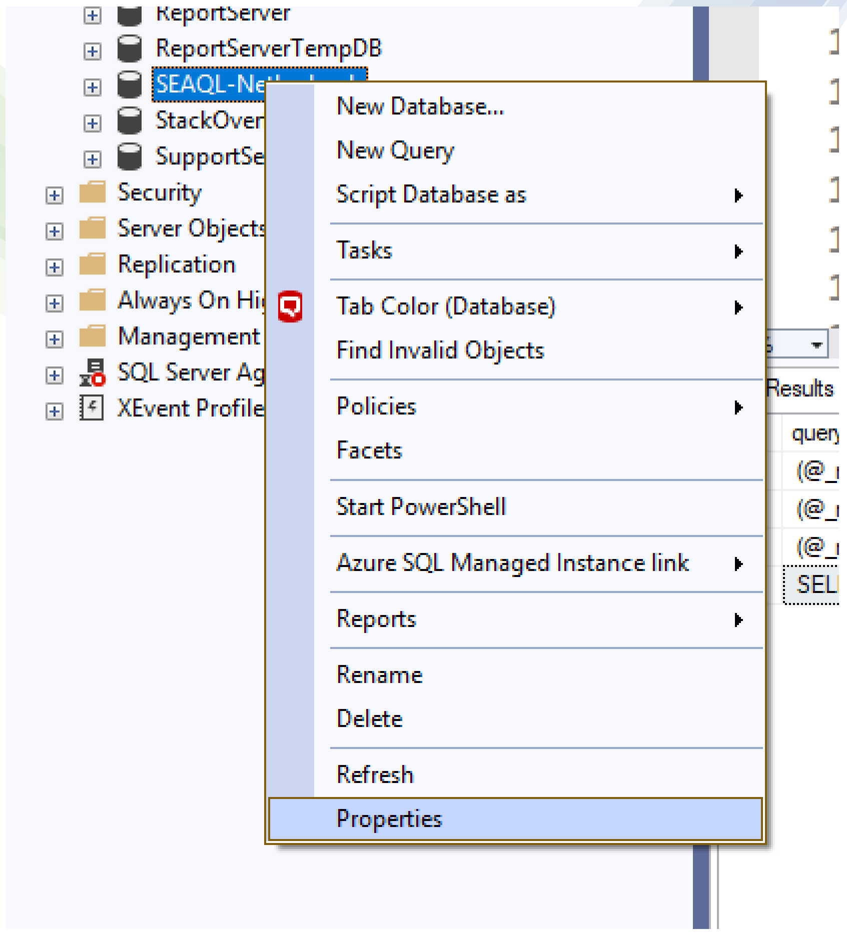
Trace Flags

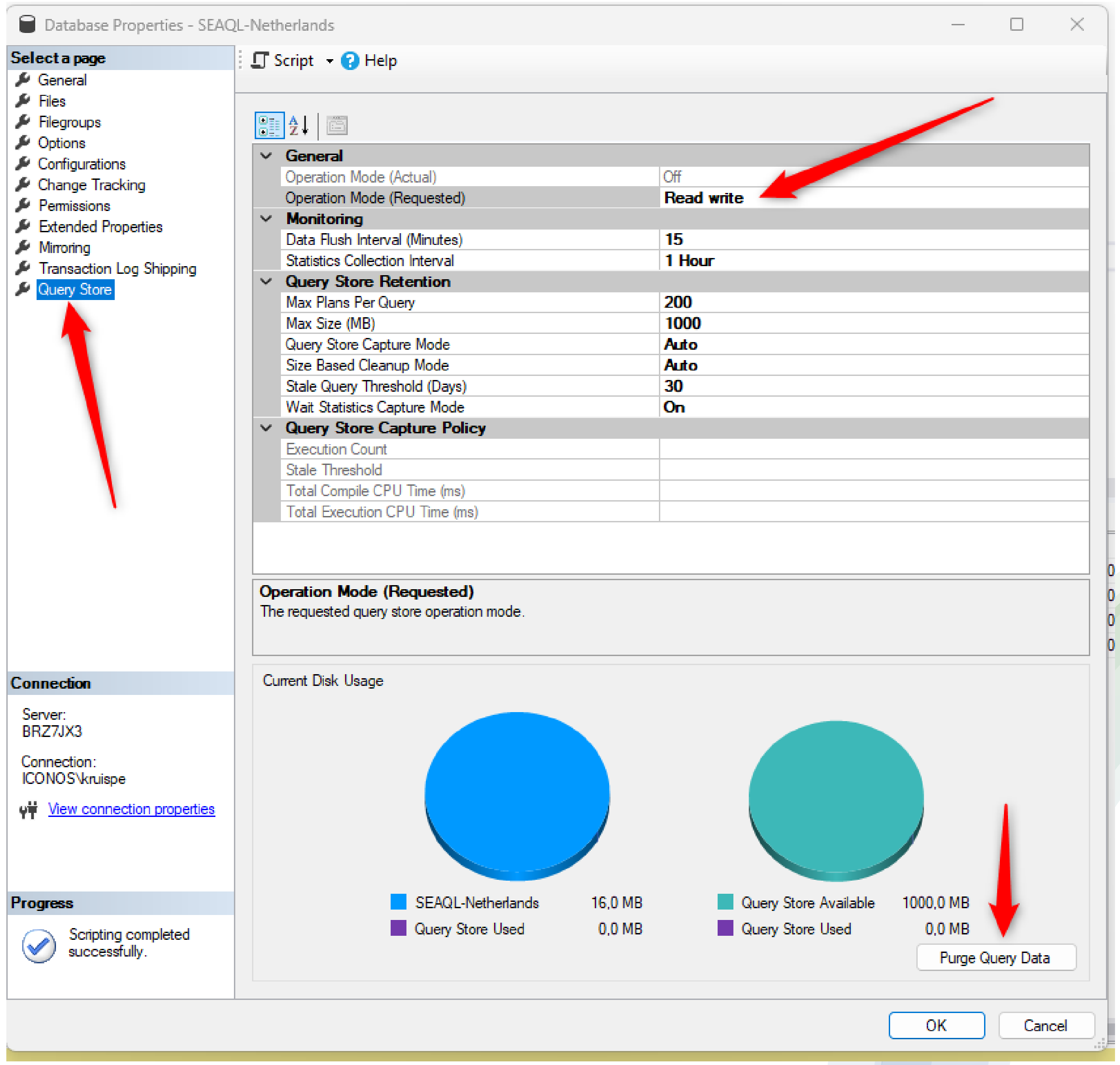
- **Traceflag 7752**
- This traceflag will allow queries to execute while Query Store loads it’s data asynchronously when starting up.
  - New data won’t be collected during this period
  - This is the default behaviour since SQL Server 2019
- **Traceflag 7745**
- This traceflag will bypass the waiting for the Query Store data being written to disk at a shutdown.
  - Some data may be lost, the amount depends on the configuration

Yeah, yeah, enough Powerpoint..

- **Demotime..**
  - Enable / Configure Query Store
  - Look into reports
  - (Un)Force plans
  - Add/remove query hints
  - Check query store tables

Configuring (1)





General	
Operation Mode (Actual)	Off
Operation Mode (Requested)	Read write
Monitoring	
Data Flush Interval (Minutes)	15
Statistics Collection Interval	1 Hour
Query Store Retention	
Max Plans Per Query	200
Max Size (MB)	1000
Query Store Capture Mode	Auto
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)	

Operation Mode (Requested)
The requested query store operation mode.

Connection
Server:
BRZ7JX3
Connection:
ICONOS\kruispe
View connection properties

Progress
Scripting completed successfully.

Current Disk Usage

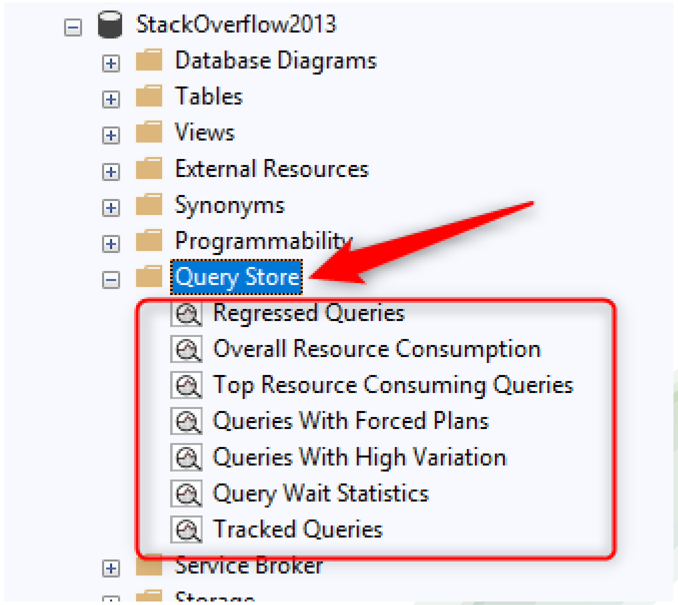
🟦 SEAQL-Netherlands	16,0 MB	🟩 Query Store Available	1000,0 MB
🟪 Query Store Used	0,0 MB	🟪 Query Store Used	0,0 MB

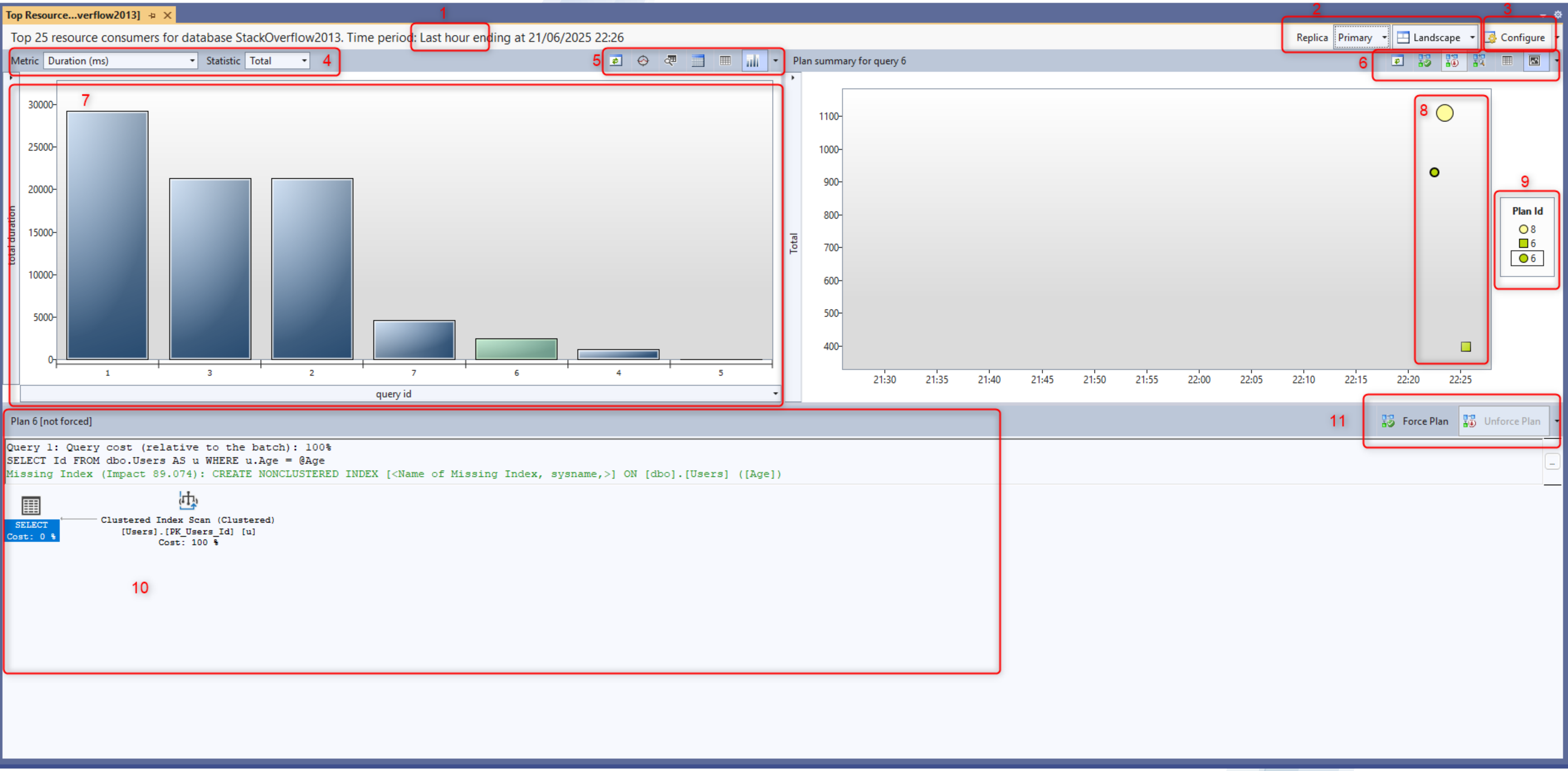
Configuring (2)

- **Properties**
  - Make sure Read-Write is selected to actually gather data
  - You probably do not want to use ALL for Capture mode, as this has the most overhead
  - Make sure the size and days match your needs
  - Probably want to change the Collection Interval to be more granular
  - Purge button will clear all information

Reports (1)

- **Reports**
  - Regressed Queries
    - Shows queries with worse performance than before
  - Overall Resource Consumption
    - Shows total CPU, reads, writes, and duration for all queries
  - Top Resource Consuming Queries
    - Shows queries that consume the most resources
  - Queries With Forced Plans
    - Shows queries running with a manually forced execution plan
  - Queries With High Variations
    - Shows queries with large differences between executions
  - Query Wait Statistics
    - Shows waits experienced by queries during execution
  - Tracked Queries
    - Shows queries manually selected for detailed tracking
    - Also search options possible





Top 25 resource consumers for database StackOverflow2013. Time period: Last hour ending at 21/06/2025 22:26

Replica: Primary

Metric: Duration (ms) Statistic: Total

Plan summary for query 6

Plan 6 [not forced]

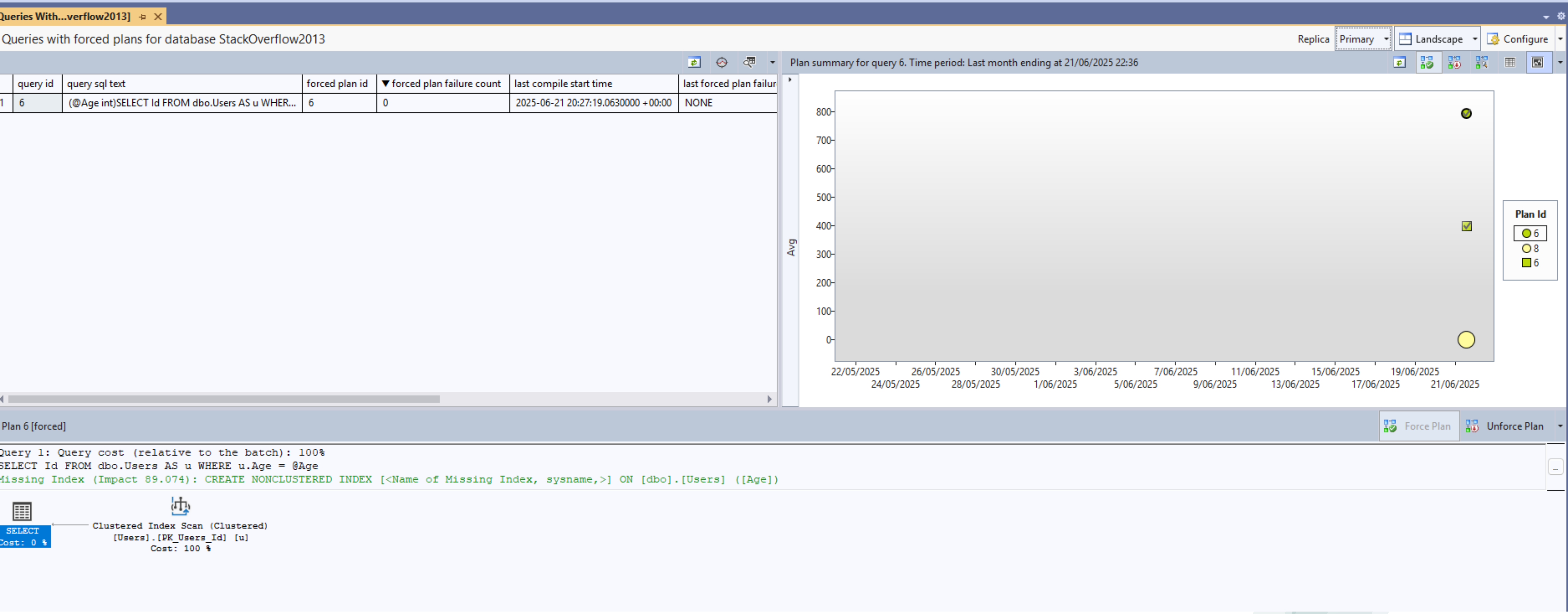
```
Query 1: Query cost (relative to the batch): 100%
SELECT Id FROM dbo.Users AS u WHERE u.Age = @Age
Missing Index (Impact 89.074): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] ON [dbo].[Users] ([Age])
```

```
SELECT          Clustered Index Scan (Clustered)
Cost: 0 %         [Users].[PK_Users_Id] [u]
                       Cost: 100 %
```

Reports (3)

- Focus has been on the Top Resource Consuming Queries during the demo, however, applies for most of the reports
- 1. The period where the report has been configured for
- 2. The replica
- 3. Possibility to change the period and metrics
- 4. Change metric / statistic
- 5. Refresh – track query – open query text in editor – grid view (2x) – chart view
- 6. Refresh – force query – unforce query – compare plans – grid view – chart view
- 7. Graph visualization of longest total duration (as configured at 1)
- 8. Multiple query plans, within the interval period, signs have separate meanings:
  - Succesfully executed – Circle
  - Triangle – Error
  - Timeout/aborted - Square
- 9. List of the query plans
- 10. Query plan
- 11. Force or unforce plan

- Queries with forced plans
- Same information, however top left shows a list of queries with forced plans.



Queries with forced plans for database StackOverflow2013

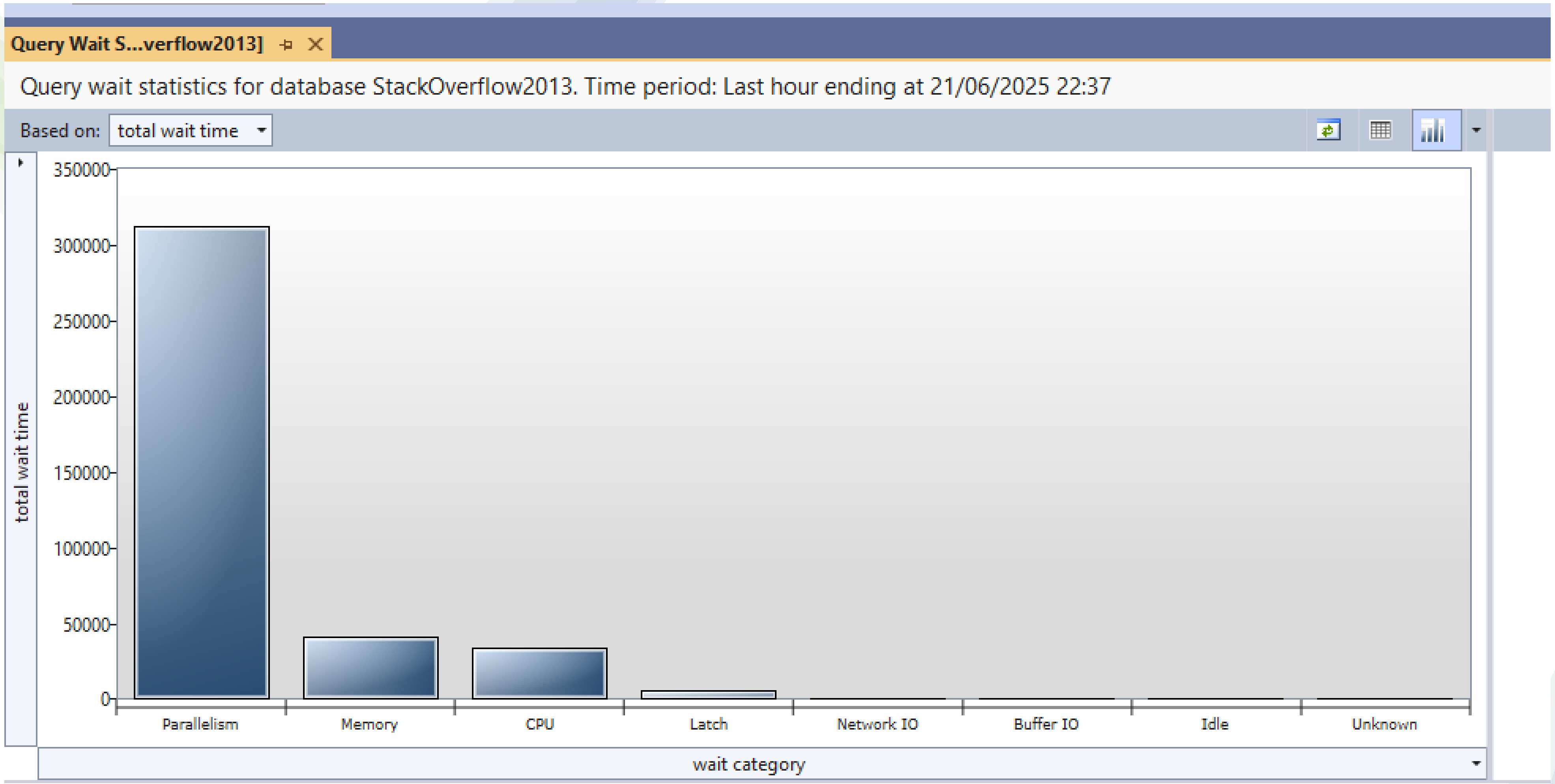
	query id	query sql text	forced plan id	▼ forced plan failure count	last compile start time	last forced plan failur
1	6	(@Age int)SELECT Id FROM dbo.Users AS u WHER...	6	0	2025-06-21 20:27:19.0630000 +00:00	NONE

Plan summary for query 6. Time period: Last month ending at 21/06/2025 22:36

Plan 6 [forced]

```
Query 1: Query cost (relative to the batch): 100%
SELECT Id FROM dbo.Users AS u WHERE u.Age = @Age
Missing Index (Impact 89.074): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] ON [dbo].[Users] ([Age])
```

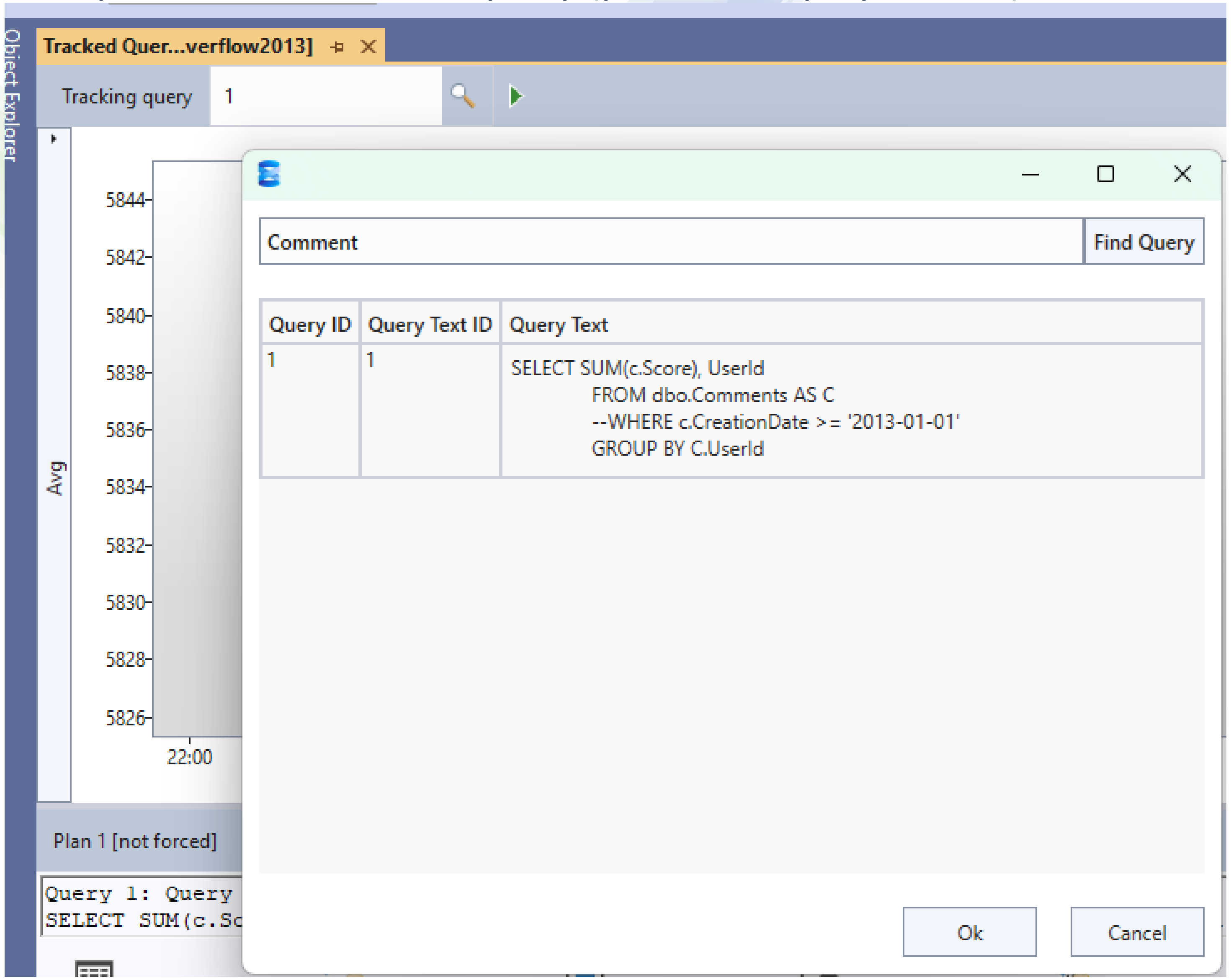
- Query wait statistics
- Grouped information, clicking will drill-down



Query wait statistics for database StackOverflow2013. Time period: Last hour ending at 21/06/2025 22:37

Based on: total wait time

- Tracked queries
- Option to search for a query (press OK + play button)



Query ID	Query Text ID	Query Text
1	1	SELECT SUM(c.Score), UserId FROM dbo.Comments AS C --WHERE c.CreationDate >= '2013-01-01' GROUP BY C.UserId

```
SELECT SUM(c.Score), UserId
        FROM dbo.Comments AS C
        --WHERE c.CreationDate >= '2013-01-01'
        GROUP BY C.UserId
```

```
Query 1: Query
SELECT SUM(c.Sc
```

Recap

- **Learned:**
  - What is query store and how did it evolve
  - How can we configure it
  - How can we see the results
  - How can we force plans
  - How can we apply query hints

- So, query store is a great tool to have in the DBA toolbox

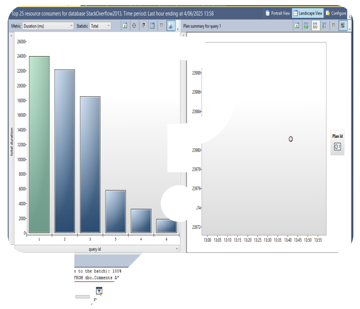
Additional resources

- Learn more:
  - Microsoft Learn
  - Erin Stellato (Pluralsight and/or SQLSkills course)

- Dig into results
  - Erin Stellato (Github)
  - Erik Darling (sp_QuickieStore)
  - Bart Vernaillen (PlanInspector)

Questions?

Feedback?!



Peter Kruis

https://www.linkedin.com/in/peter-kruis

peterkruis@hotmail.com / peter.kruis@monin-it.be

https://github.com/peterkruis

MONIN
Database Managed Services