# Using Query Store to Understand and Control Query Performance

# Goals

- Learn the kinds of data that query store collects in support of knowledge about query performance
- Understand the mechanisms of control that are offered to fine tune Query Store data collection
- Comprehend the ability to take control of query behaviors through plan forcing and hints

redgate | About me

# Grant Fritchey

## DevOps Advocate

Microsoft Data Platform MVP
AWS Community Builder


@gfritchey

grant@scarydba.com

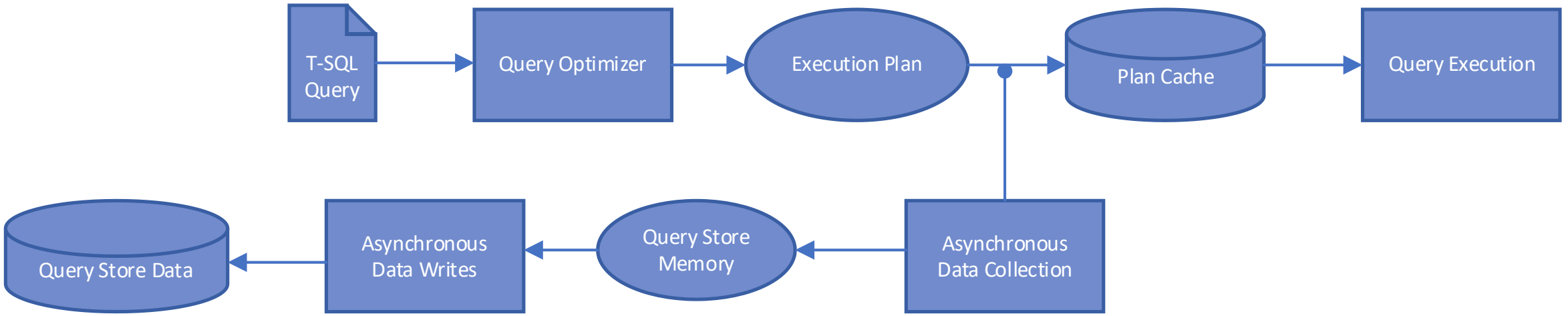
scarydba.com



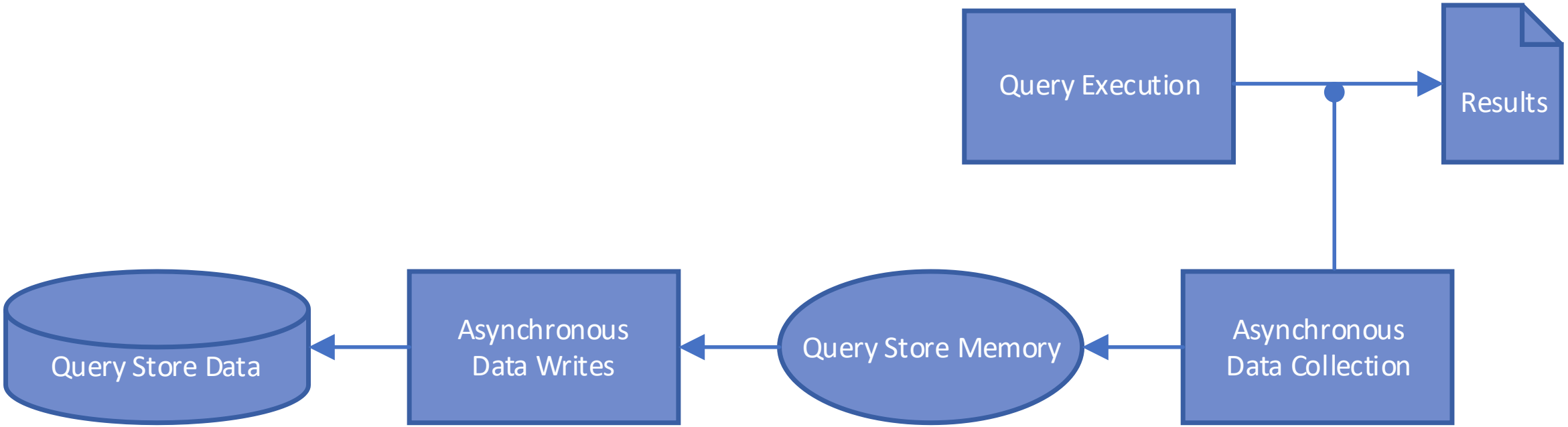

# Query Store Functionality

- Individual database setting
  - On by default in Azure SQL Database and SQL Server 2022
- Asynchronous data collection
  - First to memory
  - Then to disk
- Basis of data is the query
  - A query has one or more plans
  - A query has runtime metrics
  - A query experiences waits
- Data collected before and after query execution

# How Query Store Works: Before Execution



# How Query Store Works: After Execution



# Data Query Store Collects

- Query
  - Including T-SQL text
  - No parameter values… but
- Execution plan
  - Plan w/o runtime values (AKA “Estimated Plan”)
- Runtime metrics
  - Aggregated
  - Multiple aggregates
- Wait statistics
  - Aggregated
  - Multiple aggregates

# Data Is Stored in System Views

- Sys.query_context_settings – information about the settings for the query, including things like ANSI settings
- Sys.query_store_query – properties around the query such as hash, count of compiles, last compile duration, object_id & more
- Sys.query_store_query_text – the T-SQL for the query itself
- Sys.query_store_plan – execution plans and properties around them
- Sys.query_store_runtime_stats_interval – the defined start & stop time of runtime data aggregations
- Sys.query_store_runtime_stats – the aggregated information like average execution time, standard deviation, reads, writes, etc.
- Sys.query_store_wait_stats – the aggregated wait statistics for the query & plan in question

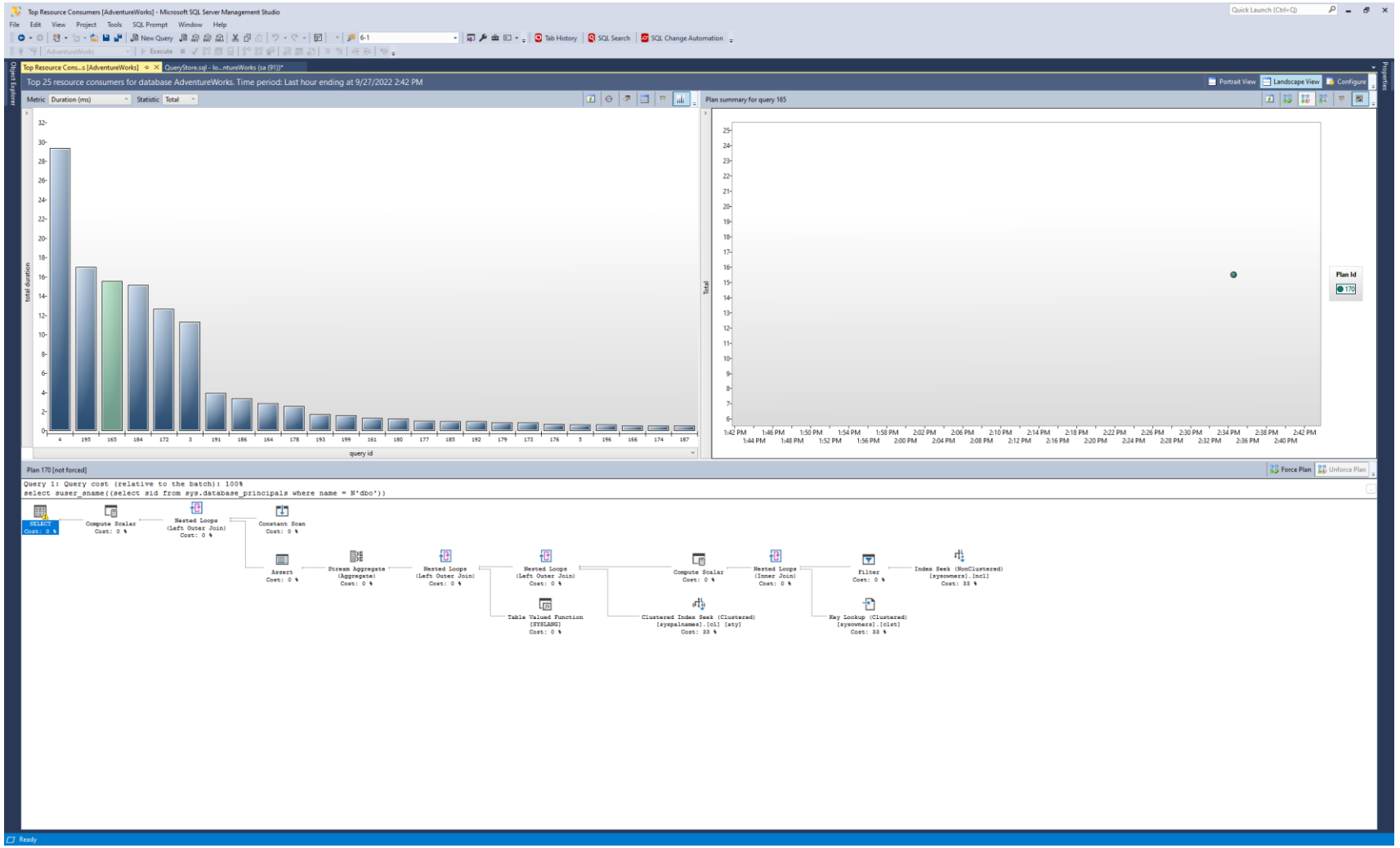
# Controlling Query Store: Basics

- Enable/Disable Query Store
  - Also READ ONLY
- Data flush interval (default 15 minutes)
- Max storage (varies per version)
- Aggregation interval length (default 60 minutes)
- Cleanup mechanisms (both days & size-based cleanup)

# Controlling Query Store: Data Collection

- Defaults vary by version
- Modes:
  - All – simple, all queries, probably not the safest approach
  - Auto – “capture queries based on resource consumption”, 3 executions or longer than 1 second
  - None – Query Store is enabled, but doesn’t capture new queries
  - Custom – 2019+
    - EXECUTION_COUNT
    - TOTAL_COMPILE_CPU_TIME
    - TOTAL_EXECUTION_CPU_TIME_MS
    - STALE_CAPTURE_POLICY_THRESHOLD

# Query Store Reporting and GUI



# Plan Forcing

- Forcing execution plans
  - T-SQL
  - GUI
- Forcing query hints
  - T-SQL
- Optimized plan forcing

# Demo

# Some Suggested Best Practices

- Monitor data size closely
- Data capture should be Auto or, better still, Custom
- Enable TraceFlags 7752 (prior to 2019) and 7745
- Minimize your historical data
- Regularly check Query Store status
- Where possible, avoid non-parameterized queries
- Use ALTER instead of DROP/CREATE where you can

# Why You Need Query Store

- Performance data collection with unique data
- Ability to force plans
- As an aid to upgrades
- Basis for most Intelligent Query Processing
- Traditional query tuning and performance troubleshooting

# Additional Resources

- Query Store for SQL Server 2019 – Tracy Boggiano, Grant Fritchey
- dbaTools
- SQL Server 2022 Query Performance Tuning – Grant Fritchey
- Best Practices With Query Store – Erin Stellato
- Best Practices With Query Store - Microsoft

redgate | About me

# Grant Fritchey

## DevOps Advocate

Microsoft Data Platform MVP
AWS Community Builder


@gfritchey

grant@scarydba.com

scarydba.com