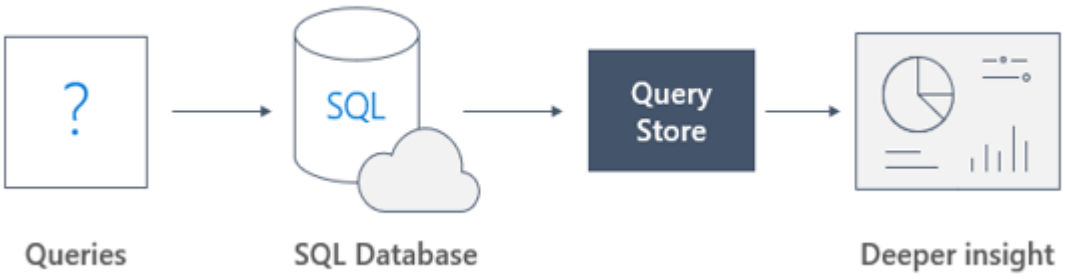
Query_Store_Hints

Last updated by | Peter Hewitt | Oct 27, 2022 at 7:29 AM PDT

Contents

Introduction

The Query Store feature was introduced in SQL Server 2016 and is enabled by default for SQL Database. It provides insights on query plan choice and performance. The Query Store simplifies performance troubleshooting by helping you quickly find performance differences caused by query plan changes. Query Store automatically captures a history of queries, plans, and runtime statistics, and retains these for review. It separates data by time windows so you can see database usage patterns and understand when query plan changes happened on the server.



Ideally the Query Optimizer selects an optimal execution plan for a query. When this doesn't happen, a developer or DBA may want to manually optimize the query for specific conditions by adding a Query hint via the OPTION clause to influence the query execution plan and execution behavior.

For example, this Transact-SQL (T-SQL) uses a query hint to set MAXDOP equal to 1:

```
SELECT ProductID, OrderQty, SUM(LineTotal) AS Total
FROM SalesLT.SalesOrderDetail
WHERE UnitPrice < 5.00
GROUP BY ProductID, OrderQty
ORDER BY ProductID, OrderQty
OPTION (MAXDOP 1);
GO
```

While query hints help provide localized solutions to various performance-related issues, they do require a rewrite of the original query text. However, DBAs and developers may not always be able to make changes directly to T-SQL code to add a query hint. This could be due to reasons such as:

- The T-SQL queries may be hard-coded into the application.
- The T-SQL queries are automatically generated by the application.

In addition, they might be able to make the changes to the code but not immediately with some wait time involved. An option for the DBA or Developer is to use plan guides, which have been around for years to attach query hints or a fixed query plan without directly changing the actual query text. However, customer feedback returned is often that "plan guides are complex to use".

Query Store Hints

Query Store hints were introduced as an extension to Query Store to help address these types of problems. If a query is captured by Query Store, then Query Store hints provides an easy to use method for shaping query plans and behavior without changing application code.

Query Store hints are GA for SQL Database and SQL Mangaged Instance since October 2022.

Using Query Store Hints

Using Query Store Hints is a two step process. The first step is to look up the query ID, the second step is to then apply the Query Store hint to the query.

1. Identify the Query Store `query_id` of the query statement you wish to modify. You can do this in various ways:
   a) Querying the Query Store catalog views.
   b) Using SQL Server Management Studio (SSMS) built-in Query Store reports.
   c) From the Azure portal Query Performance Insight blade for SQL Database.

Example of how to identify the query in the query store using the system catalog views:

```
SELECT q.query_id, qt.query_sql_text
FROM sys.query_store_query_text qt
INNER JOIN sys.query_store_query q ON
    qt.query_text_id = q.query_text_id
WHERE query_sql_text like N'%PostalCode =%'
  AND query_sql_text not like N'%query_store%';
GO
```

2. Execute sys.sp_query_store_set_hints with the `query_id` and query hint string you wish to apply to the query. This string can contain one or more query hints.

```
EXEC sys.sp_query_store_set_hints @query_id= 39, @query_hints = N'OPTION(MAX_GRANT_PERCENT=10)';
```

The hint can be removed using sp_query_store_clear_hints

```
EXEC sys.sp_query_store_clear_hints @query_id = 39;
```

Once created, Query Store hints are persisted and survive restarts and failovers.

Feature interoperability

- Query Store hints will override other hard-coded statement level hints and plan guides.
- Queries will always execute where any opposing Query Store hints, that would otherwise cause an error, will be ignored.
- If Query Store hints contradict, SQL Server will not block query execution and Query Store hint will not be applied.
- Simple parameterization - Query Store hints are not supported for statements that qualify for simple parameterization.
- Forced parameterization - The RECOMPILE hint is not compatible with forced parameterization set at the database level. If the database has forced parameterization set, and the RECOMPILE hint is part of the hints string set in Query Store for a query, SQL Server will ignore the RECOMPILE hint and will apply any other hints if they are applied.
  - Additionally, SQL Server will issue a warning (error code 12461) stating that the RECOMPILE hint was ignored.
  - For more information on forced parameterization use case considerations, see Guidelines for Using Forced Parameterization.
- Manually created Query Store hints are exempt from cleanup. The hint and the query will not be cleaned up from Query Store by the automatic retention of the capture policy.
  - Queries can be manually removed by users, which would also remove the associated Query Store hint.
  - Query Store hints automatically generated by the CE Feedback are subject to clean up by the automatic retention of the capture policy.
  - DOP feedback and memory grant feedback shape query behavior without using Query Store hints. When queries are cleanup by automatic retention of the capture policy, DOP feedback and memory grant feedback data is also cleaned up.
  - You can manually create the same Query Store hint that CE feedback implemented, and then the query with the hint would no longer be subject to clean up by the automatic retention of the capture policy.

Examples of when to use Query Store Hints

Ideally query hints when you use them they're temporary or shorter-term because you want the Query Optimizer to have all the choices available at its disposal. But there are times when they might be very useful, especially mitigating emergencies, and this is where Query hints are used the most. Some examples:

- Recompile a query on each execution.
- Cap the memory grant size for a bulk insert operation.

- Limit the maximum degree of parallelism when updating statistics.
- Use a Hash join instead of a Nested Loops join.
- Use compatibility level 110 for a specific query while keeping everything else in the database at compatibility level 150.
- Disable row goal optimization for a SELECT TOP query.

Query Store hints best practices

- Complete index and statistics maintenance before evaluating queries for potential new Query Store hints.
- Test your application database on the latest compatibility level before using Query Store hints.
  - For example, Parameter Sensitive Plan (PSP) optimization was introduced in SQL Server 2022 (16.x) Preview (compatibility level 160), which leverages multiple active plans per query to address non-uniform data distributions. If your environment cannot use the latest compatibility level, Query Store hints using the RECOMPILE hint can be leveraged on any supporting compatibility level.
- Query Store hints override SQL Server query plan behavior. It is recommended to only leverage Query Store hints when it is necessary to address performance related issues.
- It is recommended to reevaluate Query Store hints, statement level hints, plan guides, and Query Store forced plans any time data distributions change and during database migrations projects. Changes in data distribution may cause Query Store hints to generate suboptimal execution plans.

For more information, refer to Query Store hints best practices

Execution Plan XML attributes

When hints are applied, the following result set appears in the `StmtSimple` element of the Execution Plan in XML format:

Attribute	Description
`QueryStoreStatementHintText`	Actual Query Store hint(s) applied to the query
`QueryStoreStatementHintId`	Unique identifier of a query hint
`QueryStoreStatementHintSource`	Source of Query Store hint (ex: "User")

Note: These XML elements are available via the output of the Transact-SQL commands SET STATISTICS XML and SET SHOWPLAN XML.

Using XEvents for troubleshooting

If query hint was applied successfully, QDS hints will fire **query_store_hints_application_success** event. If query hint application failed, QDS hints will fire **query_store_hints_application_failed** event. Data present in each is listed below.

- `query_store_hints_application_success`
  - database_id: Id of database associated with hinted query

    - query_hint_id: Id of hint in Query Store
- `query_store_hints_application_failed`
    - database_id: Id of database associated with hinted query
    - query_hint_id: Id of hint in Query Store
    - query_id: Id of query in Query Store
    - error_number: Error number (corresponds to message_id in sys.messages)

Using Kusto for troubleshooting

Kusto telemetry for the query store hints application XEvents can be found in the table **MonQueryStoreInfo**. You can use variants of the following query to check for hint application for a particular Server/Database:

```
MonQueryStoreInfo
| where LogicalServerName =~ "myserver" and logical_database_name =~ "mydb"
| where event in ("query_store_hints_application_success", "query_store_hints_application_failed")
| where TIMESTAMP >= ago(1d)
```

FAQ

1. When Query Store is OFF, can Query Store Hints still be used?
   Query Store hints stored procedures and usage are blocked if Query Store is off. The catalog view is still available.

2. What is the behavior Query Store is in a READ_ONLY state?

Action	Query Store READ_ONLY
Create a new hint	New hints allowed
Update an existing hint	Updated hints allowed
Apply hint to query executions	Query hint is used for query executions
Remove hint	Hint deletions allowed

3. What happens to Query Store Hints when we execute ALTER DATABASE [db_name] SET QUERY_STORE CLEAR?
   All Query Store hints in the database are also purged to avoid orphaned and mismatched query ID references.

4. Are Query Store hints removed as part of automatic stale query cleanup?
   No. Query Store hinted queries are protected from stale query cleanup.

5. Does the query text change when a Query Store hint is applied?
   No.

6. What happens when an applied query hint contradicts what is possible for query optimization?
   If a query hint contradicts what is possible for query optimization, we will not block query execution and the hint will not be applied.

7. Where can customer check if a Query hint is was successfully applied or not?
   If query hint was applied successfully, QDS hints will fire Xevents query_store_hints_application_success event. If query hint application failed, QDS hints will fire query_store_hints_application_failed event. When hints are applied, the following will be surfaced in the StmtSimple element of the ShowPlan XML.

Attribute	Description
QueryStoreStatementHintText	Query Store hint(s) applied to the query
QueryStoreStatementHintId	Unique identifier of a query hint
QueryStoreStatementHintSource	Source of Query Store hint

You can further check the last failure reason in exposed in the new `sys.query_store_query_hints` *view.*

8. What happens if there is already a Queryhint associated with the query ID? *Query Store Hint already exists for a specific query_id, the last hint(s) provided will override previously specified values for the associated query.*

9. Can I use a query store hint for a query with no query ID? *Query Store hints stored procedures and usage are blocked if Query Store is off. The catalog view is still available. If a query_id doesn't exist, an error will be raised.*

10. What permissions are required for Query Store Hints stored procedures
    ALTER permission for the database.

Public Doc Reference

- Query Store hints
- Data Exposed - Query Store Hints video with Microsoft PM

Internal Reference

- Azure Databases - CSS Mentoring Series - Query Store Hints

How good have you found this content?