

Gems to help you troubleshoot query performance

Pedro Lopes
Senior Program Manager



Pedro Lopes

@sqlpto

pedro.lopes@microsoft.com

Senior Program Manager

Focused on SQL Server
Relational Engine

7+ years at Microsoft

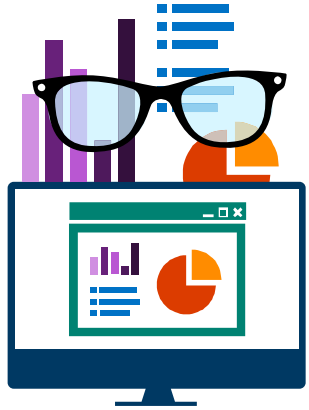
14+ years with SQL
Server



Session Objectives And Takeaways



SQL Server Tiger Team



Show new diagnostics
improvements for SQL Server
engine



Learn how to use the new
diagnostics to troubleshoot
common performance
issues



SQL Server Tiger Team

When was the last time you
dealt with a **query**
performance issue?





SQL Server Tiger Team

Query Performance Troubleshooting Fundamentals



Why does a query slow down?

- Excessive resource consumption
- Poor indexing strategy
- Lack of useful statistics
- Lack of useful partitioning
- Consequence of blocked queries
- Incorrect server configurations



Context for Slow-Running Query Analysis



- Is the performance problem related to a component other than queries?
 - For example, is the problem slow network performance?
 - Are there any other components that might be causing or contributing to performance degradation?
- If the performance issue is related to queries, which query or set of queries is involved?
- Was the query optimized with useful statistics?
- Are suitable indexes available?
- Are there any data or index hot spots?
- If you have a large volume of data, do you need to partition it?
- Is the Query Optimizer provided with the best opportunity to optimize a complex query?



SQL Server Tiger Team

How do I analyze the
performance of a slow-running
query?



Some tools we will use today

- SHOWPLAN XML (a.k.a. Actual Execution Plan)
- Query Store
- Plan Comparison Tool
- Live Query Stats
- xEvents
- ... and more 😊



A map to the execution context

- Query plans include:
 - How data is accessed
 - How data is joined
 - Sequence of operations
 - Use of temporary worktables and sorts
 - Estimated rowcounts, iterations, and costs from each step
 - Actual rowcounts and iterations
 - How data is aggregated
 - Use of parallelism
 - Query execution warnings
 - Query execution stats
 - Hardware/Resource stats

Query and Query Plan Fingerprints



- Query Fingerprint

- query_hash
- Explicitly identifies a specific query in the cache.
- *sys.dm_exec_requests*
- *sys.dm_exec_query_stats*

- SQL Handle

- sql_handle
- Token for the SQL text that relates to a batch.
- *sys.dm_exec_sql_text*
- *sys.dm_exec_query_stats*
- *sys.dm_exec_query_memory_grants*

- Query Plan Fingerprints

- query_plan_hash
- Useful to determine queries that share the same execution plan.
- Can be used to determine if the query plan has changed.
- *sys.dm_exec_requests*
- *sys.dm_exec_query_stats*

- Plan Handle

- plan_handle
- Token for a cached execution plan.
- *sys.dm_exec_query_plan*
- *sys.dm_exec_cached_plans*



SQL Server Tiger Team

Diagnostic and Troubleshooting Enhancements

The pain of joining DMVs and xEvents



- *query_hash* and *query_plan_hash* actions in xEvents
 - Not the same data types as respective columns in DMVs *sys.dm_exec_requests* and *sys.dm_exec_query_stats*
 - That makes it difficult to correlate the information
- In SQL Server 2016 RTM and 2014 SP2
 - New actions *query_hash_signed* and *query_plan_hash_signed* allow you to join these DMVs with xEvents such as *rpc_completed* and *sql_batch_completed*

	Name ^	Description
<input type="checkbox"/>	query_hash	Collect query hash. Use this to identify queries with similar logic. You can use the query hash to determine the aggregate resource usage for queries that differ only by literal
<input checked="" type="checkbox"/>	query_hash_signed	Collect query hash. Use this to identify queries with similar logic. You can use the query hash to determine the aggregate resource usage for queries that differ only by literal
<input type="checkbox"/>	query_plan_hash	Collect query plan hash. Use this to identify similar query execution plans. You can use query plan hash to find the cumulative cost of queries with similar execution plans
<input checked="" type="checkbox"/>	query_plan_hash_signed	Collect query plan hash. Use this to identify similar query execution plans. You can use query plan hash to find the cumulative cost of queries with similar execution plans



SQL Server Tiger Team

It's all about avoiding
roundtrips to collect additional
information.



Missing perf insights on query plan nodes

- Per operator performance statistics for node and threads
- Showplan extended to include *RunTimeCountersPerThread*
- Node costs for parent and children:
 - Cumulative values for Row mode operators
 - Singleton values for Batch mode operators

Runtime Info	Up to SQL 2016	SQL 2016 / SQL 2014 SP2
ActualRows	X	X
ActualRowsRead		X
Batches		X
ActualEndOfScans	X	X
ActualExecutions	X	X
ActualExecutionMode		X
ActualElapseddms		X
ActualCPUms		X
ActualScans		X
ActualLogicalReads		X
ActualPhysicalReads		X
ActualReadAheads		X
ActualLobLogicalReads		X
ActualLobPhysicalReads		X
ActualLobReadAheads		X
InputMemoryGrant		
OutputMemoryGrant		
UsedMemoryGrant		

M

Properties	
Clustered Index Scan (Clustered)	
Misc	
Actual Execution Mode	Row
Actual I/O Statistics	
Actual Lob Logical Reads	0
Actual Lob Physical Reads	0
Actual Lob Read Aheads	0
Actual Logical Reads	1345
Actual Physical Reads	3
Actual Read Aheads	1376
Actual Scans	5
Actual Number of Batches	0
Actual Number of Rows	121317
Thread 0	0
Thread 1	40604
Thread 2	17684
Thread 3	27027
Thread 4	36002
Actual Rebinds	0
Actual Rewinds	0
Actual Time Statistics	
Actual Elapsed CPU Time (ms)	74
Actual Elapsed Time (ms)	456

on query plan nodes



SQL Server Tiger Team

```
<RunTimeInformation>
<RunTimeCountersPerThread Thread="0" ActualRows="121317"
ActualRowsRead="10000000" Batches="0" ActualEndOfScans="3"
ActualExecutions="1" ActualExecutionMode="Row"
ActualElapseddms="456" ActualCPUs="74" ActualScans="3"
ActualLogicalReads="1345" ActualPhysicalReads="3"
ActualReadAheads="1376" ActualLobLogicalReads="0"
ActualLobPhysicalReads="0" ActualLobReadAheads="0" />
</RunTimeInformation>
```

SET STATISTICS IO not needed

SET STATISTICS TIME not needed



Per-operator level performance stats xEvent

New Extended Event query_thread_profile

Showplan time scale = milliseconds

xEvent time scale = microseconds for CPU and total time.

Name ^	Category ^	Channel ^
query_thread_profile	execution	Debug

query_thread_profile ^	Event Fields ^	Description
Reports the performance of each node and thread of a query plan after execution	actual_batches	Number of batches processed by this thread
	actual_execution_mode	Execution mode of the thread. 0 indicates row mode, 1 indicates batch mode
	actual_logical_reads	Number of logical pages read
	actual_physical_reads	Number of physical pages read
	actual_ra_reads	Number of read-ahead pages read
	actual_rebinds	Number of rebinds for this thread
	actual_rewinds	Number of rewinds for this thread
	actual_rows	Number of rows processed by this thread
	actual_writes	Number of pages written
	cpu_time_us	CPU time in microseconds
	io_reported	Is IO reported?
	node_id	The ID of the node in the query plan
	thread_id	The ID of the thread running in this node
	total_time_us	Cumulative time in microseconds, including waits

Getting all context info in Showplan: Trace Flags



Shows list of active trace flags:

- Query
- Session
- Global

Useful to understand if active Trace Flags influence execution context

TraceFlags	
[1]	
IsCompileTime	True
TraceFlag	
[1]	
Scope	Global
Value	2371
[2]	
Scope	Global
Value	7412
[3]	
Scope	Session
Value	9481
[2]	
IsCompileTime	False
TraceFlag	
[1]	
Scope	Global
Value	2371
[2]	
Scope	Global
Value	7412



SQL Server Tiger Team



All-up Trace Flag list

<http://aka.ms/traceflags>

Getting all context info in Showplan: Waits



Shows top 10 waits from sys.dm_exec_session_wait_stats

[-] WaitStats	
[-] [1]	
WaitCount	1049
WaitTimeMs	1
WaitType	RESERVED_MEMORY_ALLOCATION_
[-] [2]	
WaitCount	1347
WaitTimeMs	2
WaitType	MEMORY_ALLOCATION_EXT
[-] [3]	
WaitCount	6
WaitTimeMs	31
WaitType	PAGEIOLATCH_SH
[-] [4]	
WaitCount	19
WaitTimeMs	154
WaitType	ASYNC_NETWORK_IO



Getting all context info in Showplan: Parameter Data Types

Easier detection of type conversion issues

[-] Parameter List	@CustomerID, @State
[-] [1]	@CustomerID
Column	@CustomerID
Parameter Data Type	int
Parameter Runtime Value	(29401)
[-] [2]	@State
Column	@State
Parameter Compiled Value	'WA'
Parameter Data Type	char(2)
Parameter Runtime Value	'WA'

Getting all context info in Showplan: Times



Persisting information on elapsed and CPU times

[-] QueryTimeStats	
CpuTime	89
ElapsedTime	274

[-] QueryTimeStats	
CpuTime	91903
ElapsedTime	92330



Getting all context info in Showplan: RG info

List attributes of Resource Governor Settings

- MaxCompileMemory for maximum query optimizer memory in KB during compile under RG
- MaxQueryMemory for maximum query memory grant under RG
MAX_MEMORY_PERCENT hint

[-] MemoryGrantInfo	
DesiredMemory	63136
GrantedMemory	63136
GrantWaitTime	0
MaxQueryMemory	1492408
MaxUsedMemory	56024
RequestedMemory	63136
RequiredMemory	7104
SerialDesiredMemory	57544
SerialRequiredMemory	1536
Optimization Level	FULL
[-] OptimizerHardwareDependent	
EstimatedAvailableDegreeO	2
EstimatedAvailableMemory	417483
EstimatedPagesCached	104370
MaxCompileMemory	653072



SQL Server Tiger Team

Demo

Per-operator level performance stats



- Occurs when a T-SQL statement or stored procedure waits more than one second for a memory grant or when the initial attempt to get memory fails.
- Since SQL Server 2012

Understanding memory usage per execution



New columns in *sys.dm_exec_query_stats*

total_grant_kb	last_grant_kb	min_grant_kb	max_grant_kb	total_used_grant_kb	last_used_grant_kb
783288	783288	783288	783288	0	0

min_used_grant_kb	max_used_grant_kb	total_ideal_grant_kb	last_ideal_grant_kb	min_ideal_grant_kb	max_ideal_grant_kb
0	0	28592000	28592000	28592000	28592000

Showplan extended to include grant usage per thread and iterator

Memory Grant	783288
MemoryGrantInfo	
DesiredMemory	28592000
GrantedMemory	783288
GrantWaitTime	0
MaxUsedMemory	0
RequestedMemory	783288
RequiredMemory	4064
SerialDesiredMemory	28588448
SerialRequiredMemory	512

New Memory Grant Showplan Warning



SQL Server 2014 SP2 and SQL Server 2016 SP1

- 3 conditions:

- **Excessive Grant:** when max used memory is too small compared to the granted memory. This scenario can cause blocking and less efficient usage when large grants exist and a fraction of that memory was used.



SELECT	
Actual Number of Rows	0
Cached plan size	64 KB
Degree of Parallelism	0
Estimated Operator Cost	0 (0%)
Estimated Subtree Cost	0.205452
Memory Grant	67808
Estimated Number of Rows	89.3525
Statement	
SELECT	
[fo].[Order Key], [fo].[Description]	
FROM [Fact].[Order] AS [fo]	
INNER HASH JOIN [Dimension].[Stock	
Item] AS [si]	
ON [fo].[Stock Item Key] = [si].[Stock Item	
Key]	
WHERE [fo].[Lineage Key] =	
@LineageKey	
AND [si].[Lead Time Days] > 0	
ORDER BY [fo].[Stock Item Key], [fo].[Order	
Date Key] DESC	
OPTION (MAXDOP 1)	
Warnings	
The query memory grant detected	
"ExcessiveGrant", which may impact the	
reliability. Grant size: Initial 67808 KB, Final	
67808 KB, Used 1024 KB.	

KB
3172997

New Memory Grant Showplan Warning



SQL Server 2014 SP2 and SQL Server 2016 SP1

- 3 conditions:
 - **Excessive Grant**: when max used memory is too small compared to the granted memory. This scenario can cause blocking and less efficient usage when large grants exist and a fraction of that memory was used.
 - **Grant Increase**: when the dynamic grant starts to increase too much, based on the ratio between the max used memory and initial request memory. This scenario can cause server instability and unpredictable workload performance.
 - **Used More Than Granted**: when the max used memory exceeds the granted memory. This scenario can cause OOM conditions on the server.

SELECT
Cost: 13 %

Cached plan size	64 KB
Degree of Parallelism	0
Estimated Operator Cost	0 (0%)
Memory Grant	5272
Estimated Subtree Cost	0.205452
Estimated Number of Rows	89.3525

Statement
SELECT
[fo].[Order Key], [fo].[Description]
FROM [Fact].[Order] AS [fo]
INNER HASH JOIN [Dimension].[Stock Item] AS [si]
ON [fo].[Stock Item Key] = [si].[Stock Item Key]
WHERE [fo].[Lineage Key] =
@LineageKey
AND [si].[Lead Time Days] > 0
ORDER BY [fo].[Stock Item Key], [fo].[Order Date Key] DESC
OPTION (MAXDOP 1)

Warnings
The query memory grant detected "GrantIncrease", which may impact the reliability. Grant size: Initial 2200 KB, Final 5272 KB, Used 4816 KB.



New Memory Grant xEvent

- Query_memory_grant_usage XE in SQL Server 2016
 - Detect inaccurate or insufficient memory grant, when grant is >5MB as minimum

Selected events:

Name ^	⚡	🔍
query_memory_grant_usage	0	

query_memory_grant_usage
Occurs at the end of query processing for queries with memory grant over 5MB to let users know about memory grant inaccuracies

Event configuration options:

⚡ Global Fields (Actions) 🔍 Filter (Predicate) Event Fields

Name ^	Description	Data Type
dop	Degree of Parallelism	uint32
granted_memory_kb	Granted memory in KB	uint64
granted_percent	Percentage of Granted vs. Ideal	uint32
ideal_memory_kb	Ideal memory grant size in KB	uint64
usage_percent	Percentage of Used vs. Granted	uint32
used_memory_kb	Used memory in KB	uint64

min and max query memory grant option



SQL Server 2016 and SQL Server 2014 SP2

- User control over min and max grant size in percentages
 - `OPTION (MAX_GRANT_PERCENT=0.1)`, meaning 0.1% of max allowed query memory under Resource Governor
 - The valid value is between 0 and 100%
 - `MAX_GRANT_PERCENT >= MIN_GRANT_PERCENT`
- Why use a floating point value?
 - On a high end machine with 1 TB of memory, 1% can be already 10GB



SQL Server Tiger Team

Demo

Memory grant issue detection

New Spills Warnings - Sort



Up to SQL Server 2016

Sort	Cost: 32 %
Hash Match (Inner Join)	Cost: 6 %
Parall (Repartitio	Cost: 6 %

Sort	
Sort the input.	
Physical Operation	Sort
Logical Operation	Sort
Actual Execution Mode	Row
Estimated Execution Mode	Row
Actual Number of Rows	121317
Actual Number of Batches	0
Estimated Operator Cost	1.23741 (32%)
Estimated I/O Cost	0.0018769
Estimated CPU Cost	1.23553
Estimated Subtree Cost	2.71983
Estimated Number of Executions	1
Number of Executions	12
Estimated Number of Rows	97454.1
Estimated Row Size	332 B
Actual Rebinds	12
Actual Rewinds	0
Node ID	2
Warnings	
Operator used tempdb to spill data during execution with spill level 1	
Order By	
[AdventureWorks2014].[Production].[Product].Style Ascending	

SQL Server 2016 and 2014 SP2

Sort	Cost: 32 %
Hash Match (Inner Join)	Cost: 6 %
Parall (Repartitio	Cost: 6 %

Sort	
Sort the input.	
Physical Operation	Sort
Logical Operation	Sort
Actual Execution Mode	Row
Estimated Execution Mode	Row
Actual Number of Rows	121317
Actual Number of Batches	0
Estimated Operator Cost	1.23741 (32%)
Estimated I/O Cost	0.0018769
Estimated CPU Cost	1.23553
Estimated Subtree Cost	2.71983
Estimated Number of Executions	1
Number of Executions	12
Estimated Number of Rows	97454.1
Estimated Row Size	332 B
Actual Rebinds	12
Actual Rewinds	0
Node ID	2
Warnings	
Operator used tempdb to spill data during execution with spill level 1 and 12 spilled thread(s), Sort wrote 4432 pages to and read 4432 pages from tempdb with granted memory 50400KB and used memory 39704KB	
Order By	
[AdventureWorks2014].[Production].[Product].Style Ascending	

New Spills Warnings - Hash



Up to SQL Server 2016

SQL Server 2016 and 2014 SP2

Hash Match (Inner Join) Cost: 0.1200468 (20%)	Hash Match Use each row from the top input to build a hash table, and each row from the bottom input to probe into the hash table, outputting all matching rows.
Physical Operation	Hash Match
Logical Operation	Inner Join
Actual Execution Mode	Row
Estimated Execution Mode	Row
Actual Number of Rows	19620
Actual Number of Batches	0
Estimated I/O Cost	0
Estimated Operator Cost	0.1200468 (20%)
Estimated CPU Cost	0.11053
Estimated Subtree Cost	0.591696
Number of Executions	1
Estimated Number of Executions	1
Estimated Number of Rows	200
Estimated Row Size	11 B
Actual Rebinds	0
Actual Rewinds	0
Node ID	0
Output List	[AdventureWorks2014].[Sales].[Customer].CustomerID
Warnings	Operator used tempdb to spill data during execution with spill level 1
Hash Keys Probe	[AdventureWorks2014].[Sales].[Customer].CustomerID

Hash Match (Inner Join) Cost: 0.1200468 (20%)	Hash Match Use each row from the top input to build a hash table, and each row from the bottom input to probe into the hash table, outputting all matching rows.
Physical Operation	Hash Match
Logical Operation	Inner Join
Actual Execution Mode	Row
Estimated Execution Mode	Row
Actual Number of Rows	19620
Actual Number of Batches	0
Estimated I/O Cost	0
Estimated Operator Cost	0.1200468 (20%)
Estimated CPU Cost	0.11053
Estimated Subtree Cost	0.591696
Number of Executions	1
Estimated Number of Executions	1
Estimated Number of Rows	200
Estimated Row Size	11 B
Actual Rebinds	0
Actual Rewinds	0
Node ID	0
Output List	[AdventureWorks2014].[Sales].[Customer].CustomerID
Warnings	Operator used tempdb to spill data during execution with spill level 1 and 1 spilled thread(s), Hash wrote 32 pages to and read 32 pages from tempdb with granted memory 1152KB and used memory 992KB
Hash Keys Probe	[AdventureWorks2014].[Sales].[Customer].CustomerID



Spill xEvents - Hash Warning

Up to SQL Server 2016

Selected events:

Name ^		
hash_warning	1	
sort_sqlserver.hash_warning	1	

Event configuration options:

Global Fields (Actions) Filter (Predicate) Event Fields

Name ^	Description
hash_warning_type	Indicates either a hash recursion or a hash bailout warni
query_operation_nod...	Identifies the node ID of the operation that caused the h
recursion_level	Indicates the number of times that the build input was :

SQL Server 2016
SQL Server 2014
SP2

Selected events:

Name ^		
hash_spill_details	1	
hash_warning	1	
sort_warning	0	

Event configuration options:

Global Fields (Actions) Filter (Predicate) Event Fields

Name	Description ^
actual_row_count	Actual number of uniquely hashed rows
dop	Degree of parallelism
granted_memory_kb	Granted memory in KB
query_operation_nod...	Identifies the node ID of the operation that caused the h
thread_id	Identifies worker thread id which matches to showplan
hash_warning_type	Indicates either a hash recursion or a hash bailout warni
recursion_level	Indicates the number of times that the build input was :
workfile_physical_writ...	Number of pages written to workfile
worktable_physical_w...	Number of pages written to worktable
used_memory_kb	Used memory in KB



Spill xEvents - Sort Warning

Up to SQL Server 2016

Selected events:

Name ^		
hash_warning	1	
sort_warning	1	

Event configuration options:

Global Fields (Actions) Filter (Predicate) Event Fields

Name ^	Description
query_operation_nod...	Identifies the node ID of the operation that caused the s
sort_warning_type	Indicates whether sorting a query required a single or m

SQL Server 2016
SQL Server 2014
SP2

Selected events:

Name ^		
hash_spill_details	1	
hash_warning	1	
sort_warning	0	

Event configuration options:

Global Fields (Actions) Filter (Predicate) Event Fields

Name	Description ^
actual_row_count	Actual number of sorted rows
dop	Degree of parallelism
granted_memory_kb	Granted memory in KB
query_operation_nod...	Identifies the node ID of the operation that caused the s
thread_id	Identifies worker thread id which matches to showplan
sort_warning_type	Indicates whether sorting a query required a single or m
worktable_physical_re..	Number of pages read from worktable
worktable_physical_w..	Number of pages written to worktable
used_memory_kb	Used memory in KB



Detecting predicate search inefficiencies?

Actual number of rows returned are rows after the predicate is applied.

- Not the actual number of rows that are scanned from a table or index.

Scenario hidden from an actual execution plan:

- SCAN or SEEK returns only 10 rows, why is it taking so long?
- You see high CPU or many logical reads, but the query plan doesn't reflect that.

Now what?? 😞

Searching without pushdown



```
SELECT [ProductID]
FROM [Sales].[SalesOrderDetail]
WHERE [ModifiedDate] BETWEEN '2011-01-01' AND '2012-01-01'
AND [OrderQty] >= 10
```



Result Set

ModifiedDate	ProductID	StoreID	OrderQty	SalesAmount
2011-04-16	106	02	10	40
2011-07-20	102	02	12	50
2011-10-21	106	03	16	55
2011-12-15	103	03	20	55
(...)	(...)	(...)	(...)	(...)
2012-01-01	109	01	11	16

Range Scan

Sales.SalesOrderDetail

ModifiedDate	ProductID	StoreID	OrderQty	SalesAmount
2010-12-31	106	01	1	30
2011-01-07	103	04	1	17
2011-01-07	109	04	7	25
2011-02-12	103	03	5	16
2011-03-08	106	05	7	40
2011-04-16	106	02	10	40
2011-07-20	102	02	12	50
2011-10-21	106	03	16	55
2011-12-15	103	03	20	55
(...)	(...)	(...)	(...)	(...)
2012-01-01	109	01	11	16
2012-01-11	102	05	5	10

Actual Rows

Filter

Searching with pushdown



```
SELECT [ProductID]
FROM [Sales].[SalesOrderDetail]
WHERE [ModifiedDate] BETWEEN '2011-01-01' AND '2012-01-01'
AND [OrderQty] >= 10
```



Range
Scan

Result Set

ModifiedDate	ProductID	StoreID	OrderQty	SalesAmount
2011-04-16	106	02	10	40
2011-07-20	102	02	12	50
2011-10-21	106	03	16	55
2011-12-15	103	03	20	55
(...)	(...)	(...)	(...)	(...)
2012-01-01	109	01	11	16

Sales.SalesOrderDetail

ModifiedDate	ProductID	StoreID	OrderQty	SalesAmount
2010-12-31	106	01	12	30
2011-01-07			1	17
2011-01-07			7	20
2011-02-12	106	03	5	40
2011-03-08	103	05	7	25
2011-04-16	106	02	10	40
2011-07-20	102	02	12	50
2011-10-21	106	03	16	55
2011-12-15	103	03	20	55
(...)	(...)	(...)	(...)	(...)
2012-01-01	109	01	11	16
2012-01-11	102	05	5	10

Actual
Rows



SQL Server Tiger Team

Demo

Predicate Pushdown in Showplan

Production Alert: Application is slow!



SQL Server Tiger Team



Run data collection tools:

PSSDiag

xEvents

Profiler



...

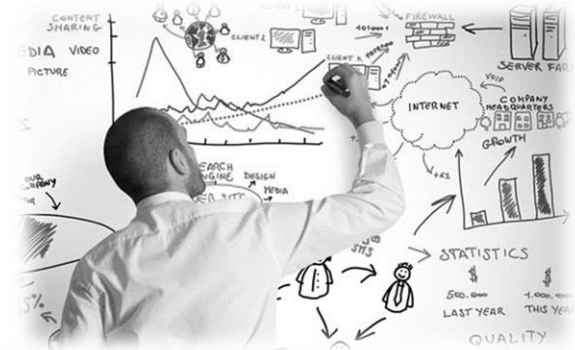
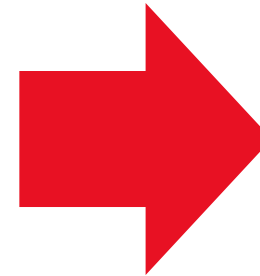
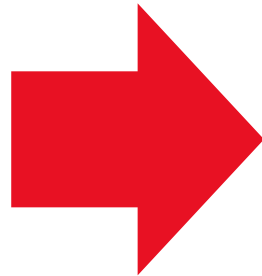
All after the fact

Take the data:

Repro

Analyze

Deploy mitigation





Production Alert: Query Perf Issues!

What if I could do live query troubleshooting?

- To have in-flight query execution statistics, the *query execution statistics profile infrastructure* must be enabled **on demand**.
- But cost overhead goes up to 75%.
 - It makes bad things worse if running all the time.
 - This is why customers still revert back to previous pattern.

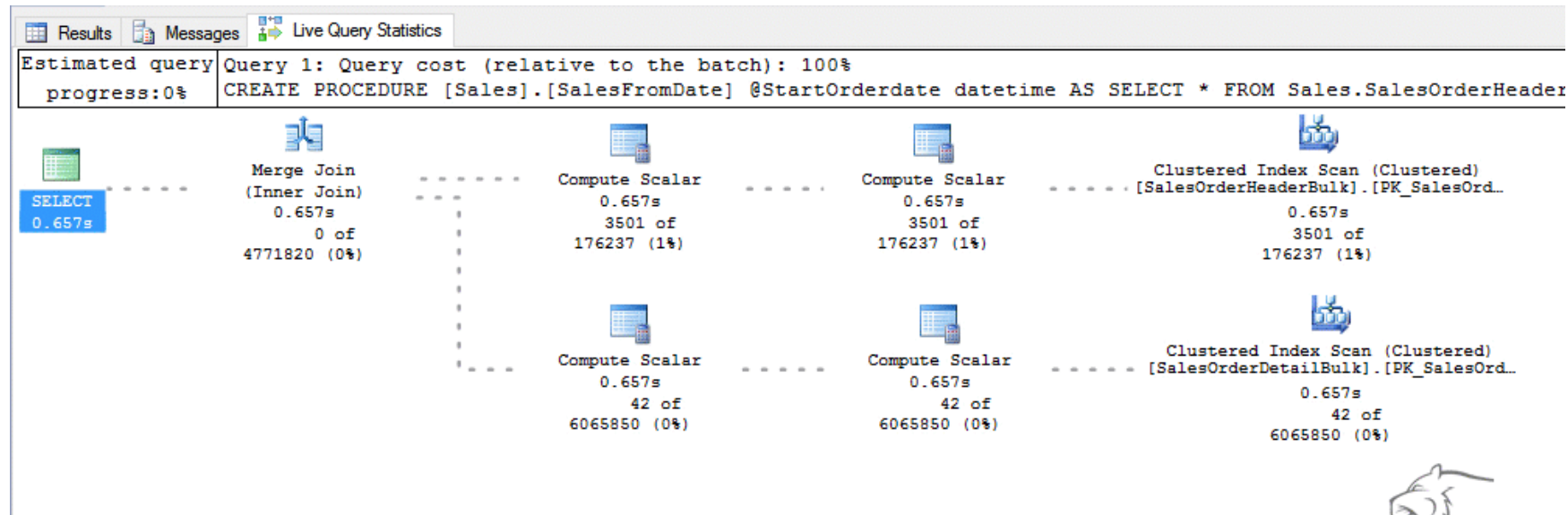
Unleash Lightweight Profiling



SQL Server Tiger Team



Tap to in-flight execution
Find the hotspot
Deploy mitigation



SQL Server Tiger Team

Tracking query progress (estimated)



- To have in-flight query execution statistics, the *query execution statistics profile infrastructure* must be enabled on demand.
- Can be enabled for a target session:
 - Specifying Include Live Query Statistics in SSMS.
 - SET STATISTICS XML ON
 - SET STATISTICS PROFILE ON
- Or globally to view the LQS from other sessions (such as from Activity Monitor):
 - Enabling *query_post_execution_showplan* extended event.
- **High overhead** (75% with TPC-C like workload)



Lightweight Tracking query progress (estimated)



- Lightweight query execution profiling **dramatically reduces performance overhead** of continuously collecting per-operator query execution statistics (such as actual number of rows)
- Can be enabled by:
 - Using global TF 7412.
 - Enabling *query_thread_profile* extended event.
 - When lightweight profiling is on, *sys.dm_exec_query_profiles* is also populated for all sessions.
- This enables usage of LQS feature in SSMS (including Activity Monitor) and of the new DMF **sys.dm_exec_query_statistics_xml**.
- The following still use regular profiling infra:
 - SET STATISTICS XML (or Include Actual Plan).
 - *query_post_execution_showplan* extended event.



What is the impact of live query troubleshooting?



Query Execution Statistics Profiling Infrastructure tests with TPC-C like workloads

Infra Type	Overhead percent (up to)	
	no active xEvents	Active xEvent query_post_execution_showplan
Legacy	75.5	93.17
Lightweight in SQL Server 2014 SP2/2016	3.5	62.02
Lightweight in SQL Server 2016 SP1	2	14.3



SQL Server Tiger Team

Demo

Live Query Troubleshooting

What Diagnostic choices do you have?



SQL Server Tiger Team

Regular Profiling

Full runtime statistics for a query plan

Most expensive overhead

Can be enabled per session or globally

Consume data from live queries or post execution

Lightweight Profiling

Limited runtime statistics in query plan (no CPU tracking)

Least expensive overhead

Only enabled globally

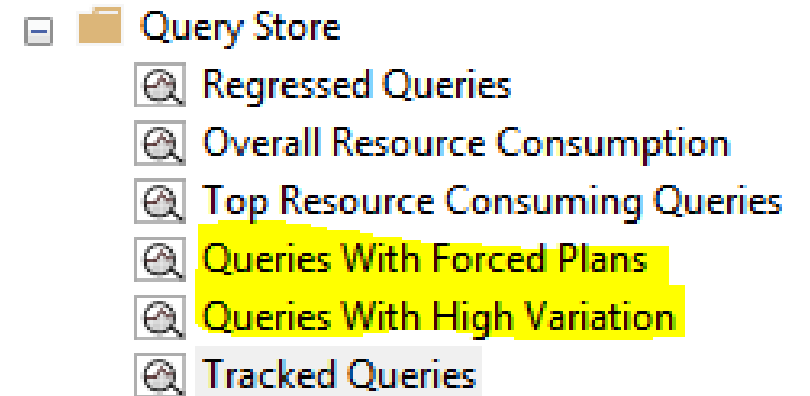
Consume data from live queries or post execution

More optimized in SQL Server 2016 SP1

Query performance insights in SSMS



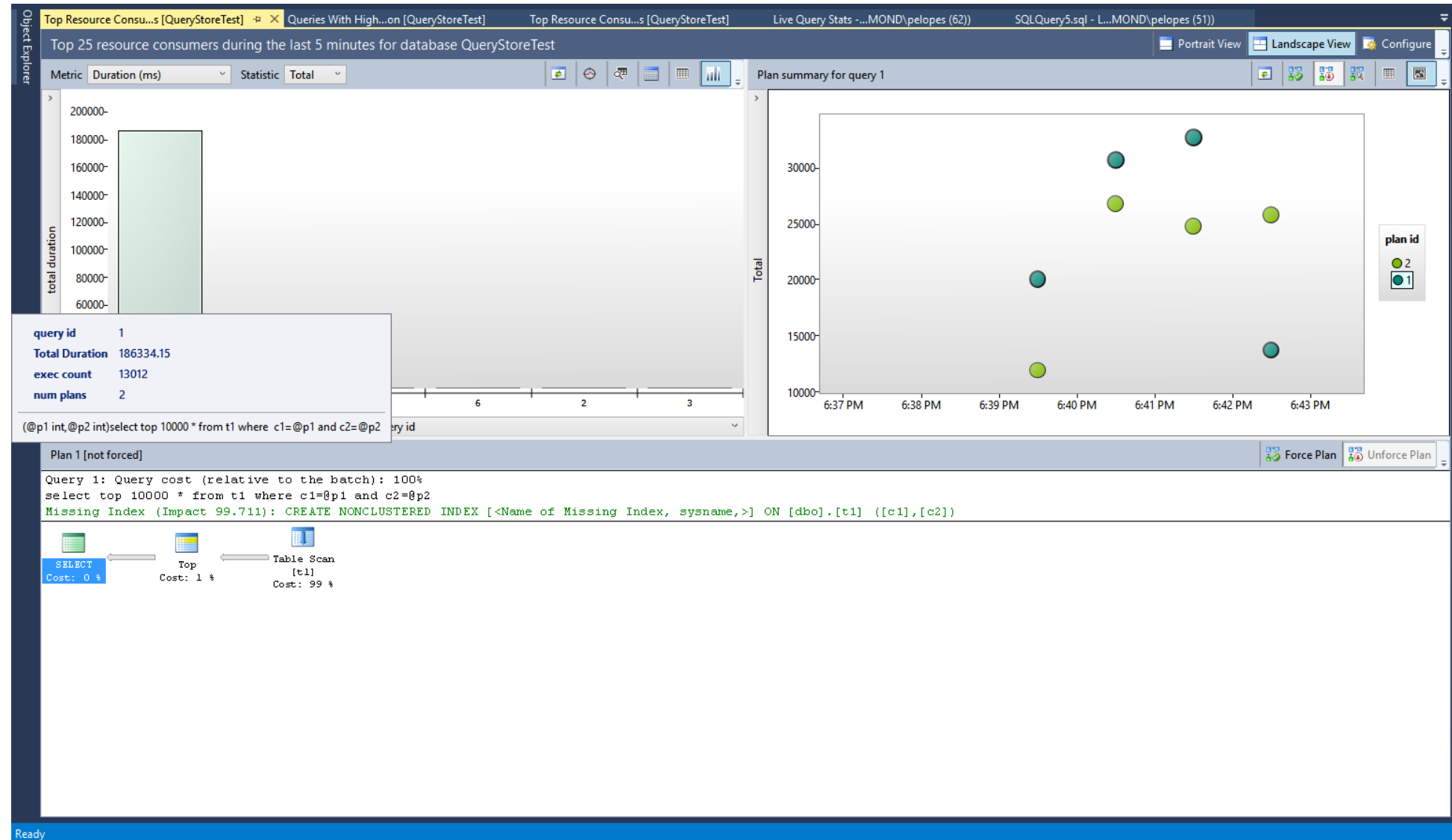
- Still in last v16:
 - Support for multi-statement showplan comparison
 - Per-operator level performance stats in showplan Properties window
- Query Store
 - Filter by number of different plans
- New with v17:
 - Query Store: new reports
 - Query analysis scenarios: Introduces CE diff search



Query Store UI



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



SSMS Plan Comparison



SSMS Plan Comparison interface showing two query plans side-by-side for comparison.

Top Plan (Slow): C:\IP\Demos\SSMS Comp Tool\Compare1_Plan1_Slow.sqlplan

Bottom Plan (Fast): C:\IP\Demos\SSMS Comp Tool\Compare1_Plan2_Fast.sqlplan

Query Text: SELECT e.[BusinessEntityID], p.[Title], p.[FirstName], p.[MiddleName], p.[LastName], p.[Suffix], e.[JobTitle], pp.[PhoneNumber], pnt.[Name] AS [...]

Plan Comparison Options:

- ☒ Highlight similar operations
- ☐ Highlight operators not matching similar segments
- ☒ Ignore database name when comparing operators

Properties Panel (Top Plan):

Property	Value
Actual Number of Rows	290
Cached plan size	160 KB
CardinalityEstimationModelVersion	70
CompileCPU	79
CompileMemory	2424
CompileTime	79
DatabaseContextSettingsId	1
Degree of Parallelism	1
Estimated Number of Rows	4598570000000000
Estimated Operator Cost	0 (0%)
Estimated Subtree Cost	6451880000000000
Memory Grant	115520
Optimization Level	FULL
OptimizerHardwareDependentPro	0
ParentObjectId	0
QueryHash	0xF7AF7F558ADB4
QueryPlanHash	0xF472609C6CC25
QueryTimeStats	CpuTime: 77197, ElapsedTime: 77416
RetrievedFromCache	true
SecurityPolicyApplied	False
Set Options	ANSI_NULLS: True, ANSI_PADDING: True, ANSI_WARNINGS: True, ARITHABORT: True, CONCAT_NULL_YIELDS_NULL: True, FIPS_FLAGGER: SQL_SERVER, IMPLICIT_TRANSACTIONS: False, ISOLATION: ReadCommitted, LANGUAGE: English, NUMERIC_ROUNDABORT: False, QUOTED_IDENTIFIER: True, RECURSIVE_QUERY_TIMEOUT: 0, ROWCOUNT_LIMIT: 0, SHOWPLAN_ALL: False, SHOWPLAN_TEXT: True, SHOWPLAN_XML: False, TABLOCK: False, TRANSACTION_ISOLATION: ReadCommitted, XACT_ABORT: False
TraceFlags	1204, 1205, 1206, 1207, 1208, 1209, 1210, 1211, 1212, 1213, 1214, 1215, 1216, 1217, 1218, 1219, 1220, 1221, 1222, 1223, 1224, 1225, 1226, 1227, 1228, 1229, 1230, 1231, 1232, 1233, 1234, 1235, 1236, 1237, 1238, 1239, 1240, 1241, 1242, 1243, 1244, 1245, 1246, 1247, 1248, 1249, 1250, 1251, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 1259, 1260, 1261, 1262, 1263, 1264, 1265, 1266, 1267, 1268, 1269, 1270, 1271, 1272, 1273, 1274, 1275, 1276, 1277, 1278, 1279, 1280, 1281, 1282, 1283, 1284, 1285, 1286, 1287, 1288, 1289, 1290, 1291, 1292, 1293, 1294, 1295, 1296, 1297, 1298, 1299, 1300, 1301, 1302, 1303, 1304, 1305, 1306, 1307, 1308, 1309, 1310, 1311, 1312, 1313, 1314, 1315, 1316, 1317, 1318, 1319, 1320, 1321, 1322, 1323, 1324, 1325, 1326, 1327, 1328, 1329, 1330, 1331, 1332, 1333, 1334, 1335, 1336, 1337, 1338, 1339, 1340, 1341, 1342, 1343, 1344, 1345, 1346, 1347, 1348, 1349, 1350, 1351, 1352, 1353, 1354, 1355, 1356, 1357, 1358, 1359, 1360, 1361, 1362, 1363, 1364, 1365, 1366, 1367, 1368, 1369, 1370, 1371, 1372, 1373, 1374, 1375, 1376, 1377, 1378, 1379, 1380, 1381, 1382, 1383, 1384, 1385, 1386, 1387, 1388, 1389, 1390, 1391, 1392, 1393, 1394, 1395, 1396, 1397, 1398, 1399, 1400, 1401, 1402, 1403, 1404, 1405, 1406, 1407, 1408, 1409, 1410, 1411, 1412, 1413, 1414, 1415, 1416, 1417, 1418, 1419, 1420, 1421, 1422, 1423, 1424, 1425, 1426, 1427, 1428, 1429, 1430, 1431, 1432, 1433, 1434, 1435, 1436, 1437, 1438, 1439, 1440, 1441, 1442, 1443, 1444, 1445, 1446, 1447, 1448, 1449, 1450, 1451, 1452, 1453, 1454, 1455, 1456, 1457, 1458, 1459, 1460, 1461, 1462, 1463, 1464, 1465, 1466, 1467, 1468, 1469, 1470, 1471, 1472, 1473, 1474, 1475, 1476, 1477, 1478, 1479, 1480, 1481, 1482, 1483, 1484, 1485, 1486, 1487, 1488, 1489, 1490, 1491, 1492, 1493, 1494, 1495, 1496, 1497, 1498, 1499, 1500, 1501, 1502, 1503, 1504, 1505, 1506, 1507, 1508, 1509, 1510, 1511, 1512, 1513, 1514, 1515, 1516, 1517, 1518, 1519, 1520, 1521, 1522, 1523, 1524, 1525, 1526, 1527, 1528, 1529, 1530, 1531, 1532, 1533, 1534, 1535, 1536, 1537, 1538, 1539, 1540, 1541, 1542, 1543, 1544, 1545, 1546, 1547, 1548, 1549, 1550, 1551, 1552, 1553, 1554, 1555, 1556, 1557, 1558, 1559, 1560, 1561, 1562, 1563, 1564, 1565, 1566, 1567, 1568, 1569, 1570, 1571, 1572, 1573, 1574, 1575, 1576, 1577, 1578, 1579, 1580, 1581, 1582, 1583, 1584, 1585, 1586, 1587, 1588, 1589, 1590, 1591, 1592, 1593, 1594, 1595, 1596, 1597, 1598, 1599, 1600, 1601, 1602, 1603, 1604, 1605, 1606, 1607, 1608, 1609, 1610, 1611, 1612, 1613, 1614, 1615, 1616, 1617, 1618, 1619, 1620, 1621, 1622, 1623, 1624, 1625, 1626, 1627, 1628, 1629, 1630, 1631, 1632, 1633, 1634, 1635, 1636, 1637, 1638, 1639, 1640, 1641, 1642, 1643, 1644, 1645, 1646, 1647, 1648, 1649, 1650, 1651, 1652, 1653, 1654, 1655, 1656, 1657, 1658, 1659, 1660, 1661, 1662, 1663, 1664, 1665, 1666, 1667, 1668, 1669, 1670, 1671, 1672, 1673, 1674, 1675, 1676, 1677, 1678, 1679, 1680, 1681, 1682, 1683, 1684, 1685, 1686, 1687, 1688, 1689, 1690, 1691, 1692, 1693, 1694, 1695, 1696, 1697, 1698, 1699, 1700, 1701, 1702, 1703, 1704, 1705, 1706, 1707, 1708, 1709, 1710, 1711, 1712, 1713, 1714, 1715, 1716, 1717, 1718, 1719, 1720, 1721, 1722, 1723, 1724, 1725, 1726, 1727, 1728, 1729, 1730, 1731, 1732, 1733, 1734, 1735, 1736, 1737, 1738, 1739, 1740, 1741, 1742, 1743, 1744, 1745, 1746, 1747, 1748, 1749, 1750, 1751, 1752, 1753, 1754, 1755, 1756, 1757, 1758, 1759, 1760, 1761, 1762, 1763, 1764, 1765, 1766, 1767, 1768, 1769, 1770, 1771, 1772, 1773, 1774, 1775, 1776, 1777, 1778, 1779, 1780, 1781, 1782, 1783, 1784, 1785, 1786, 1787, 1788, 1789, 1790, 1791, 1792, 1793, 1794, 1795, 1796, 1797, 1798, 1799, 1800, 1801, 1802, 1803, 1804, 1805, 1806, 1807, 1808, 1809, 1810, 1811, 1812, 1813, 1814, 1815, 1816, 1817, 1818, 1819, 1820, 1821, 1822, 1823, 1824, 1825, 1826, 1827, 1828, 1829, 1830, 1831, 1832, 1833, 1834, 1835, 1836, 1837, 1838, 1839, 1840, 1841, 1842, 1843, 1844, 1845, 1846, 1847, 1848, 1849, 1850, 1851, 1852, 1853, 1854, 1855, 1856, 1857, 1858, 1859, 1860, 1861, 1862, 1863, 1864, 1865, 1866, 1867, 1868, 1869, 1870, 1871, 1872, 1873, 1874, 1875, 1876, 1877, 1878, 1879, 1880, 1881, 1882, 1883, 1884, 1885, 1886, 1887, 1888, 1889, 1890, 1891, 1892, 1893, 1894, 1895, 1896, 1897, 1898, 1899, 1900, 1901, 1902, 1903, 1904, 1905, 1906, 1907, 1908, 1909, 1910, 1911, 1912, 1913, 1914, 1915, 1916, 1917, 1918, 1919, 1920, 1921, 1922, 1923, 1924, 1925, 1926, 1927, 1928, 1929, 1930, 1931, 1932, 1933, 1934, 1935, 1936, 1937, 1938, 1939, 1940, 1941, 1942, 1943, 1944, 1945, 1946, 1947, 1948, 1949, 1950, 1951, 1952, 1953, 1954, 1955, 1956, 1957, 1958, 1959, 1960, 1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970, 1971, 1972, 1973, 1974, 1975, 1976, 1977, 1978, 1979, 1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030, 2031, 2032, 2033, 2034, 2035, 2036, 2037, 2038, 2039, 2040, 2041, 2042, 2043, 2044, 2045, 2046, 2047, 2048, 2049, 2050, 2051, 2052, 2053, 2054, 2055, 2056, 2057, 2058, 2059, 2060, 2061, 2062, 2063, 2064, 2065, 2066, 2067, 2068, 2069, 2070, 2071, 2072, 2073, 2074, 2075, 2076, 2077, 2078, 2079, 2080, 2081, 2082, 2083, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2091, 2092, 2093, 2094, 2095, 2096, 2097, 2098, 2099, 2100, 2101, 2102, 2103, 2104, 2105, 2106, 2107, 2108, 2109, 2110, 2111, 2112, 2113, 2114, 2115, 2116, 2117, 2118, 2119, 2120, 2121, 2122, 2123, 2124, 2125, 2126, 2127, 2128, 2129, 2130, 2131, 2132, 2133, 2134, 2135, 2136, 2137, 2138, 2139, 2140, 2141, 2142, 2143, 2144, 2145, 2146, 2147, 2148, 2149, 2150, 2151, 2152, 2153, 2154, 2155, 2156, 2157, 2158, 2159, 2160, 2161, 2162, 2163, 2164, 2165, 2166, 2167, 2168, 2169, 2170, 2171, 2172, 2173, 2174, 2175, 2176, 2177, 2178, 2179, 2180, 2181, 2182, 2183, 2184, 2185, 2186, 2187, 2188, 2189, 2190, 2191, 2192, 2193, 2194, 2195, 2196, 2197, 2198, 2199, 2200, 2201, 2202, 2203, 2204, 2205, 2206, 2207, 2208, 2209, 2210, 2211, 2212, 2213, 2214, 2215, 2216, 2217, 2218, 2219, 2220, 2221, 2222, 2223, 2224, 2225, 2226, 2227, 2228, 2229, 2230, 2231, 2232, 2233, 2234, 2235, 2236, 2237, 2238, 2239, 2240, 2241, 2242, 2243, 2244, 2245, 2246, 2247, 2248, 2249, 2250, 2251, 2252, 2253, 2254, 2255, 2256, 2257, 2258, 2259, 2260, 2261, 2262, 2263, 2264, 2265, 2266, 2267, 2268, 2269, 2270, 2271, 2272, 2273, 2274, 2275, 2276, 2277, 2278, 2279, 2280, 2281, 2282, 2283, 2284, 2285, 2286, 2287, 2288, 2289, 2290, 2291, 2292, 2293, 2294, 2295, 2296, 2297, 2298, 2299, 2300, 2301, 2302, 2303, 2304, 2305, 2306, 2307, 2308, 2309, 2310, 2311, 2312, 2313, 2314, 2315, 2316, 2317, 2318, 2319, 2320, 2321, 2322, 2323, 2324, 2325, 2326, 2327, 2328, 2329, 2330, 2331, 2332, 2333, 2334, 2335, 2336, 2337, 2338, 2339, 2340, 2341, 2342, 2343, 2344, 2345, 2346, 2347, 2348, 2349, 2350, 2351, 2352, 2353, 2354, 2355, 2356, 2357, 2358, 2359, 2360, 2361, 2362, 2363, 2364, 2365, 2366, 2367, 2368, 2369, 2370, 2371, 2372, 2373, 2374, 2375, 2376, 2377, 2378, 2379, 2380, 2381, 2382, 2383, 2384, 2385, 2386, 2387, 2388, 2389, 2390, 2391, 2392, 2393, 2394, 2395, 2396, 2397, 2398, 2399, 2400, 2401, 2402, 2403, 2404, 2405, 2406, 2407, 2408, 2409, 2410, 2411, 2412, 2413, 2414, 2415, 2416, 2417, 2418, 2419, 2420, 2421, 2422, 2423, 2424, 2425, 2426, 2427, 2428, 2429, 2430, 2431, 2432, 2433, 2434, 2435, 2436, 2437, 2438, 2439, 2440, 2441, 2442, 2443, 2444, 2445, 2446, 2447, 2448, 2449, 2450, 2451, 2452, 2453, 2454, 2455, 2456, 2457, 2458, 2459, 2460, 2461, 2462, 2463, 2464, 2465, 2466, 2467, 2468, 2469, 2470, 2471, 2472, 2473, 2474, 2475, 2476, 2477, 2478, 2479, 2480, 2481, 2482, 2483, 2484, 2485, 2486, 2487, 2488, 2489, 2490, 2491, 2492, 2493, 2494, 2495, 2496, 2497, 2498, 2499, 2500, 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 2509, 2510, 2511, 2512, 2513, 2514, 2515, 2516, 2517, 2518, 2519, 2520, 2521, 2522, 2523, 2524, 2525, 2526, 2527, 2528, 2529, 2530, 2531, 2532, 2533, 2534, 2535, 2536, 2537, 2538, 2539, 2540, 2541, 2542, 2543, 2544, 2545, 2546, 2547, 2548, 2549, 2550, 2551, 2552, 2553, 2554, 2555, 2556, 2557, 2558, 2559, 2560, 2561, 2562, 2563, 2564, 2565, 2566, 2567, 2568, 2569, 2570, 2571, 2572, 2573, 2574, 2575, 2576, 2577, 2578, 2579, 2580, 2581, 2582, 2583, 2584, 2585, 2586, 2587, 2588, 2589, 2590, 2591, 2592, 2593, 2594, 2595, 2596, 2597, 2598, 2599, 2600, 2601, 2602, 2603, 2604, 2605, 2606, 2607, 2608, 2609, 2610, 2611, 2612, 2613, 2614, 2615, 2616, 2617, 2618, 2619, 2620, 2621, 2622, 2623, 2624, 2625, 2626, 2627, 2628, 2629, 2630, 2631, 2632, 2633, 2634, 2635, 2636, 2637, 2638, 2639, 2640, 2641, 2642, 2643, 2644, 2645, 2646, 2647, 2648, 2649, 2650, 2651, 2652, 2653, 2654, 2655, 2656, 2657, 2658, 2659, 2660, 2661, 2662, 2663, 2664, 2665, 2666, 2667, 2668, 2669, 2670, 2671, 2672, 2673, 2674, 2675, 2676, 2677, 2678, 2679, 2680, 2681, 2682, 2683, 2684, 2685, 2686, 2687, 2688, 2689, 2690, 2691, 2692, 2693, 2694, 2695, 2696, 2697, 2698, 2699, 2700, 2701, 2702, 2703, 2704, 2705, 2706, 2707, 2708, 2709, 2710, 2711, 2712, 2713, 2714, 2715, 2716, 2717, 2718, 2719, 2720, 2721, 2722, 2723, 2724, 2725, 2726, 2727, 2728, 2729, 2730, 2731, 2732, 2733, 2734, 2735, 2736, 2737, 2738, 2739, 2740, 2741, 2742, 2743, 2744, 2745, 2746, 2747, 2748, 2749, 2750, 2751, 2752, 2753, 2754, 2755, 2756, 2757, 2758, 2759, 2760, 2761, 2762, 2763, 2764, 2765, 2766, 2767, 2768, 2769, 2770, 2771, 2772, 2773, 2774, 2775, 2776, 2777, 2778, 2779, 2780, 2781, 2782, 2783, 2784, 2785, 2786, 2787, 2788, 2789, 2790, 2791, 2792, 2793, 2794, 2795, 2796, 2797, 2798, 2799, 2800, 2801, 2802, 2803, 2804, 2805, 2806, 2807, 2808, 2809, 2810, 2811, 2812, 2813, 2814, 2815, 2816, 2817, 2818, 2819, 2820, 2821, 2822, 2823, 2824, 2825, 2826, 2827, 2828, 2829, 2830, 2831, 2832, 2833, 2834, 2835, 2836, 2837, 2838, 2839, 2840, 2841, 2842, 2843, 2844, 2845, 2846, 2847, 2848, 2849, 2850, 2851, 2852, 2853, 2854, 2855, 2856, 2857, 2858, 2859, 2860, 2861, 2862, 2863, 2864, 2865, 2866, 2867, 2868, 2869, 2870, 2871, 2872, 2873, 2874, 2875, 2876, 2877, 2878, 2879, 2880, 2881, 2882, 2883, 2884, 2885, 2886, 2887, 2888, 2889, 2890, 2891, 2892, 2893, 2894, 2895, 2896, 2897, 2898, 2899, 2900, 2901, 2902, 2903, 2904, 2905, 2906, 2907, 2908, 2909, 2910, 2911, 2912, 2913, 2914, 2915, 2916, 2917, 2918, 2919, 2920, 2921, 2922, 2923, 2924, 2925, 2926, 2927, 2928, 2929, 2930, 2931, 2932, 2933, 2934, 2935, 2936, 2937, 2938, 2939, 2940, 2941, 2942, 2943, 2944, 2945, 2946, 2947, 2948, 2949, 2950, 2951, 2952, 2953, 2954, 2955, 2956, 2957, 2958, 2959, 2960, 2961, 2962, 2963, 2964, 2965, 2966, 2967, 2968, 2969, 2970, 2971, 2972, 2973, 2974, 2975, 2976, 2977, 2978, 2979, 2980, 2981, 2982, 2983, 2984, 2985, 2986, 2987, 2988, 2989, 2990, 2991, 2992, 2993, 2994, 2995, 2996, 2997, 2998, 2999, 3000, 3001, 3002, 3003, 3004, 3005, 3006, 3007, 3008, 3009, 3010, 3011, 3012, 3013, 3014, 3015, 3016, 3017, 3018, 3019, 3020, 3021, 3022, 3023, 3024, 3025, 3026, 3027, 3028, 3029, 3030, 3031, 3032, 3033, 3034, 3035, 3036, 3037, 3038, 3039, 3040, 3041, 3042, 3043, 3044, 3045, 3046, 3047, 3048, 3049, 3050, 3051, 3052, 3053, 3054, 3055, 3056, 3057, 3058, 3059, 3060, 3061, 3062, 3063, 3064, 3065, 3066, 3067, 3068, 3069, 3070, 3071, 3072, 3073, 3074, 3075, 3076, 3077, 3078, 3079, 3080, 3081, 3082, 3083, 3084, 3085, 3086, 3087, 3088, 3089, 3090, 3091, 3092, 3093, 3094, 3095, 3096, 3097, 3098, 3099, 3100, 3101, 3102, 3103, 3104, 3105, 310

SSMS Plan Analysis



er Team

Query 1: Query cost (relative to the batch): 100%

SELECT * FROM Sales.SalesOrderHeaderBulk AS h INNER JOIN Sales.SalesOrderDetailBulk AS d ON h.SalesOrderID = d.SalesOrderID WHERE (h.OrderDate >= @StartOrderdate)

SELECT
Cost: 0 %

Compute Scalar
Cost: 0 %

Parallelism
(Gather Streams)
Cost: 1 %

Nested Loops
(Inner Join)
Cost: 0 %

Compute Scalar
Cost: 1 %

Compute Scalar
Cost: 0 %

Clustered Index Scan (Clustered)
[SalesOrderHeaderBulk].[PK_SalesOrd...
Cost: 66 %

Clustered Index Seek (Clustered)
[SalesOrderDetailBulk].[PK_SalesOrd...
Cost: 31 %

Showplan Analysis

Multi Statement Scenarios

Each tab below shows details on a category of potential issues found in the plans.

Inaccurate Cardinality Estimation

Findings

< Back

Forward >

	Difference %	Actual	Estimated	Node
1	14314	405300	2811.67	Clustered Index Scan (Clustered) [SalesOrderHeaderBulk].[PK_SalesOrderHeaderBulk]
2	12718	1389650	10840.7	Parallelism (Gather Streams)
3	12718	1389650	10840.7	Nested Loops (Inner Join)
4	12718	1389650	10840.7310854	Clustered Index Seek (Clustered) [SalesOrderDetailBulk].[PK_SalesOrderDetailBulk]

Finding Details

Click here for more information about this scenario

1) The predicate for this operator depends on parameter @StartOrderdate. The compile-time value was unknown or different from the runtime value, so the estimate may not be accurate for the run-time value. Refer to [here](#) for more details.

2) One of the common reasons for estimation differences is the use of different statistics. Check if statistics for table [SalesOrderHeaderBulk] are different or stale. Refer to [here](#) for more information.

SELECT

Misc

Cached plan size 80 KB

CardinalityEstimationModelVersion 130

CompileCPU 6

CompileMemory 592

CompileTime 6

Degree of Parallelism 4

Estimated Number of Rows 10840.7

Estimated Operator Cost 0 (0%)

Estimated Subtree Cost 28.8878

Memory Grant 72

MemoryGrantInfo

Optimization Level FULL

OptimizerHardwareDependentProperties

Parameter List

@StartOrderdate

Column @StartOrderdate

Parameter Compiled Value '2004-07-31 00:00:00.000'

Parameter Data Type datetime

Parameter Runtime Value '2004-03-28 00:00:00.000'

QueryHash 0x504FBC112C813CD6

QueryPlanHash 0x99DC0A67C61621F4

QueryTimeStats

RetrievedFromCache true

SecurityPolicyApplied False

Set Options

ANSI_NULLS: True, ANSI_PADDING: True, ANSI_WARNINGS: Off

Statement SELECT *FROM Sales.SalesOrderHeaderBulk AS h

ThreadStat

TraceFlags

WaitStats

Parameter List

Parameter list.

Ready



SQL Server Tiger Team

Demo

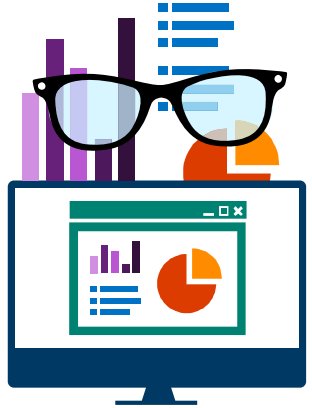
Query Store UI and Plan Comparison

Plan Analysis

In Review: Objectives And Takeaways



SQL Server Tiger Team



Learn about new diagnostics improvements for SQL Server engine



Learn how to use the new diagnostics to troubleshoot common performance issues

Bookmarks



SQL Server Tiger Team Blog <http://aka.ms/sqlserverteam>

Tiger Toolbox GitHub <http://aka.ms/tigertoolbox>

SQL Server Release Blog <http://aka.ms/sqlreleases>

BP Check <http://aka.ms/bpcheck>

SQL Server Standards Support <http://aka.ms/sqlstandards>

Trace Flags <http://aka.ms/traceflags>

SQL Server Support lifecycle <http://aka.ms/sqlifecycle>

SQL Server Updates <http://aka.ms/sqlupdates>

Twitter [@mssqltiger](#)

