SQLskills Online Immersion Event

Fixing Slow Queries, Inefficient Code, and Caching/Statistics Problems

Part I: Capturing Query Information and Analyzing Plans

Erin Stellato
Erin@SQLskills.com

Erin Stellato

Principal Consultant, SQLskills

Erin@SQLskills.com

@erinstellato

www.sqlskills.com/blogs/erin

Trainer/Speaker

In addition to consulting, I teach content for our IE0: Accidental DBA course, and our IEPTO2: Performance Tuning and Optimization course

PASS Volunteer

I was a member of the PASS Nomination Committee this past year, have previously served on the board of my local user group (ONSSUG) and supported the Performance Virtual Chapter

Data Platform MVP

I have been fortunate to be recognized as an MVP by Microsoft since 2012

© SQLskills, All rights reserved.
http://www.SQLskills.com

Schedule

- **Tuesday:** Erin Stellato
  - Capturing Query Information and Analyzing Plans
- **Wednesday:** Jonathan Kehayias
  - Removing Anti-Patterns in Transact-SQL
- **Thursday:** Kimberly Tripp
  - How to Differentiate Caching/Statistics problems and SOLVE THEM!

© SQLskills, All rights reserved.
http://www.SQLskills.com

Schedule – Tuesday, Wednesday & Thursday

- **Lecture 1:** 90 minutes from 10:00am to 11:30am PDT
  - Please use the WebEx chat for questions and responses **during** the lecture
- **Open Q&A 1:** 30 minutes from 11:30am to 12:00 noon PDT
  - I'll address any unanswered questions from chat
  - May have time for "open mic" questions
- **Mandatory break:** 30 minutes from 12:00 noon until 12:30pm PDT
  - Get up, stretch, get something to eat/drink, recharge!

Time conversions:
10:00am PDT
= 1:00pm EDT
= 5:00pm UTC

© SQLskills, All rights reserved.
http://www.SQLskills.com

Schedule – Tuesday, Wednesday & Thursday

- **Lecture 2:** 90 minutes from 12:30pm to 2:00pm PDT
  - Again, please use WebEx chat for questions/responses **during** the lecture
- **Open Q&A 2:** Up to 60 minutes from 2:00pm to 3:00pm PDT
  - Address any unanswered questions from chat
  - Time for "open mic" questions

Time conversions:
10:00am PDT
= 1:00pm EDT
= 5:00pm UTC

© SQLskills, All rights reserved.
http://www.SQLskills.com

Course Resources & Pluralsight Access

- Paul (Paul@sqlskills.com) will send you an email for a FREE 30-day access code to ALL SQLskills online training class on Pluralsight
  - No strings attached, no credit card needed
  - If you do not receive an email by end of day Thursday, email Paul
- Course resources will be sent upon completion (Friday)
  - I will update slides and add notes/resources based on questions
  - Please do not distribute it; it is for course attendees only.
- Email Paul (Paul@sqlskills.com) if you have problems finding or opening the resource file

© SQLskills, All rights reserved.
http://www.SQLskills.com

Course Resources

- Course content (this deck)
- Demos
  - All instructor-led demos
  - All reference scripts
- Reference links (in the zip file)
- Online recordings for the sessions

Please do not redistribute: These resources are for IEQuery attendees only. We ask that you please respect our IP and time in creating these resources and do not share/forward/sell on eBay ☺

© SQLskills, All rights reserved.
http://www.SQLskills.com

Overview

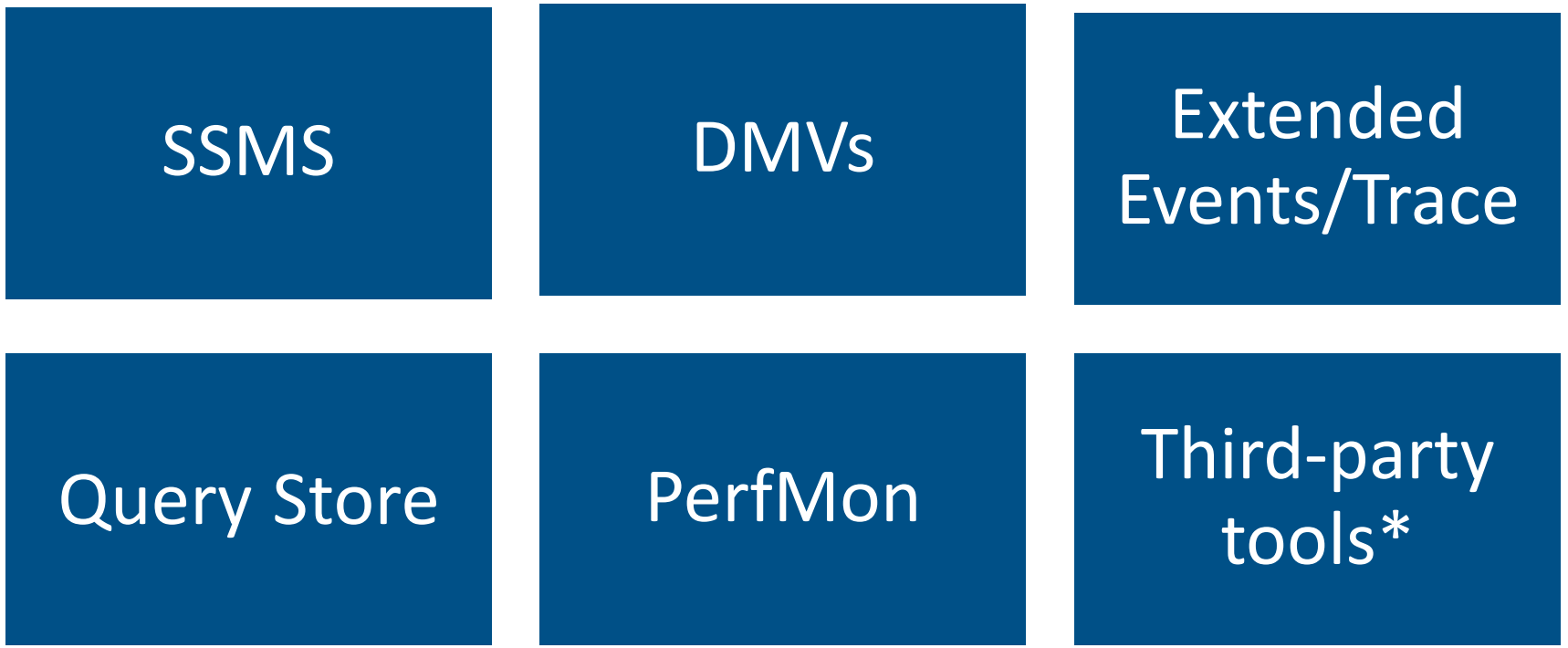
- Query Metrics
- Capturing and comparing query plans
- Information in a plan
- Plan analysis

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Metrics

© SQLskills, All rights reserved.
http://www.SQLskills.com

Sources for Data Related to Query Performance



© SQLskills, All rights reserved.
http://www.SQLskills.com

Purpose of a Baseline

- Provide a starting point for comparison of additional data over time
  - Often represents the "normal" or typical state of the environment
  - Helps you understand where the system is today
- Baseline data is invaluable during a performance crisis
- Can also be used to identify usage patterns and trending, and can be extremely helpful for capacity planning
- Used to measure the impact of changes

© SQLskills, All rights reserved.
http://www.SQLskills.com

Benchmarking

- A benchmark measures performance using a *specific set of indicators* to determine the performance level in a way that can be compared to other systems, business requirements, or previous benchmarks
- It is a ***standard*** for comparison
- May be established to determine the capacity limits of a system
  - Maximum number of concurrent connections
  - Maximum number of transactions/batches per second
  - Use to forecast replacement/upgrade requirements before exceeding limits

© SQLskills, All rights reserved.
http://www.SQLskills.com

Benchmark vs. Baseline

- Use benchmarks and baselines to reach a target or specific goal
  - "This stored procedure used to run in 200 ms." (*this is your benchmark*)
  - "The SP now takes 3 seconds." (*this is your baseline*)
  - Compare the baseline to the benchmark
  - Improve the current value in steps (*this is tuning*)

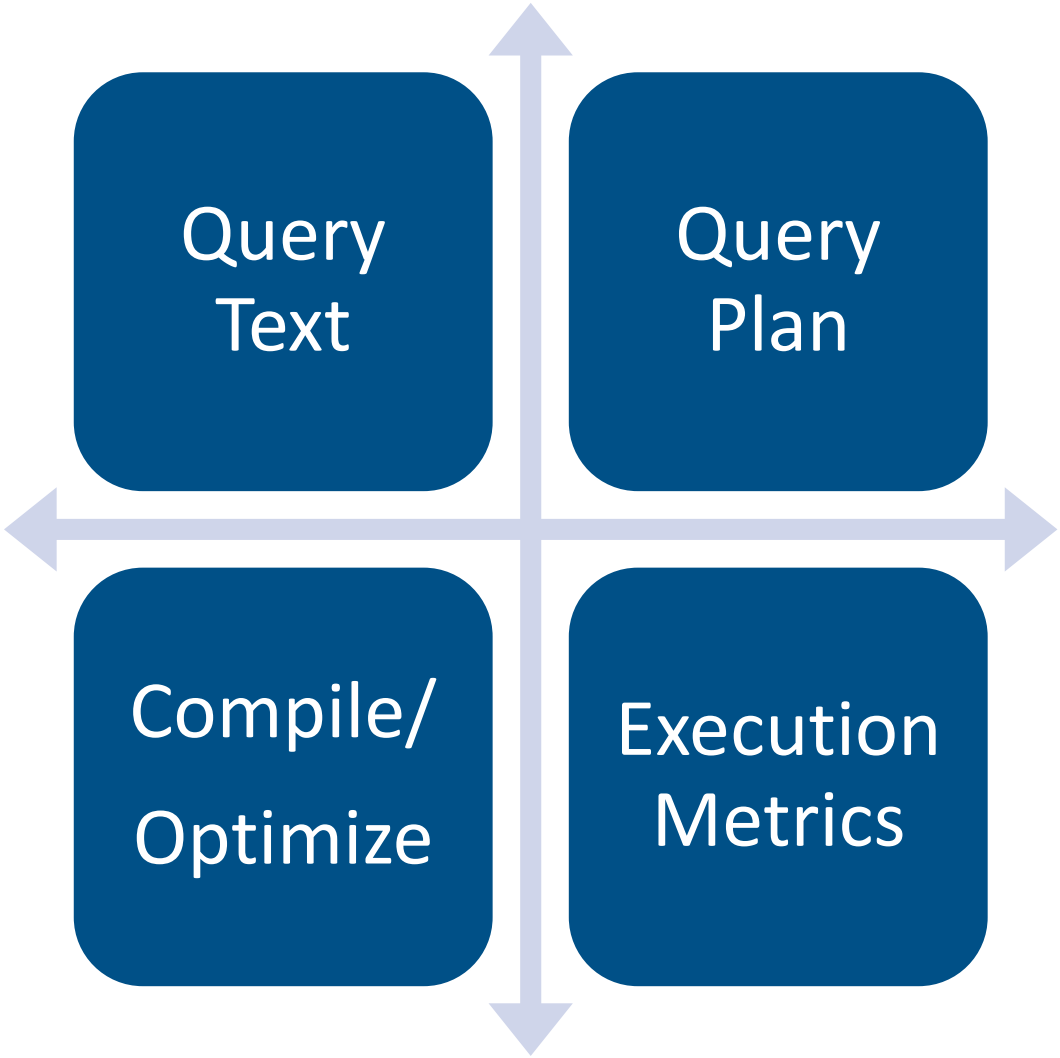
© SQLskills, All rights reserved.
http://www.SQLskills.com

How do you baseline query performance?

- Use a third-party monitoring tool
- Snapshot data from multiple DMVs
- Use sp_whoisactive to snapshot data
- Capture data using Extended Events
  - Trace for SQL Server 2008R2 and earlier
- OpenQueryStore
  - SQL Server 2008 – SQL Server 2014
- Query Store
  - SQL Server 2016+

© SQLskills, All rights reserved.
http://www.SQLskills.com

Specific Data of Interest



Data source will depend on SQL Server version and the problem you're trying to solve

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Execution Data

Individual Metrics

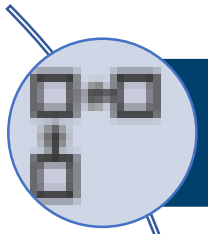
- SSMS
- Extended Events
- Trace

Aggregated Metrics

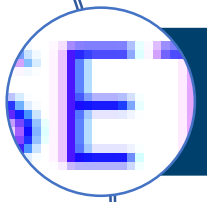
- DMVs
- Query Store
- PerfMon

© SQLskills, All rights reserved.
http://www.SQLskills.com

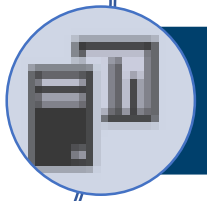
SSMS



Actual and estimated plans



STATISTICS IO and STATISTICS TIME



Client Statistics

Live Query Statistics

© SQLskills, All rights reserved.
http://www.SQLskills.com

DMVs

- sys.dm_exec_sql_text
  - Provide a sql_handle, returns text of the batch
- sys.dm_exec_cached_plans
  - One row for each query plan currently in memory
- sys.dm_exec_query_plan
  - Provided a plan_handle, it returns the plan in XML format
- sys.dm_exec_text_query_plan
  - Output is in text format
- sys.dm_exec_query_stats
  - Aggregate performance statistics for cached plans
- sys.dm_exec_procedure_stats
  - Aggregate performance statistics for cached stored procedures
- sys.dm_exec_function_stats (added in 2016)
  - Aggregate performance statistics for cached functions

© SQLskills, All rights reserved.
http://www.SQLskills.com

Extended Events

- sql_batch_completed
  - Completion of T-SQL batch
- rpc_completed
  - Completion of a remote procedure call
- sp_statement_completed
  - Completion of T-SQL statement within a stored procedure
- sql_statement_completed
  - Completion of T-SQL statement
- query_post_compilation_showplan
- query_pre_execution_showplan
- query_post_execution_showplan

Not recommended

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Store

- Described as a flight data recorder
- Captures information about query execution
  - Query text
  - Query plan
  - Compilation time
  - Last execution time
  - Duration, CPU, logical reads, physical reads, writes – aggregated across a time interval

© SQLskills, All rights reserved.
http://www.SQLskills.com

Data Captured by Query Store

Plan Store

- Compile time and duration
- Last execution time
- Query text
- Query plan

Runtime Stats Store

- Execution counts
- Duration
- CPU
- Logical reads
- Physical reads
- Write
- Memory use
- DOP
- Log bytes/used
- tempdb

Wait Stats Store

- Wait statistics per plan

SQL Server 2017 and Azure SQL Database

© SQLskills, All rights reserved.
http://www.SQLskills.com

Capturing and Comparing Query Plans

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Plans

- The majority of plans are stored in cache
- This means that the plan cache can be queried
- The plan retrieved from cache is the one that has been used
  - This is often referred to as the *estimated* plan
  - When you retrieve the plan from cache, you only see estimates (number of rows, number of executions)
- The *actual plan* can only be retrieved during query execution
  - The "actual" query plan is *typically* the same as the plan that exists in cache, and it *also includes* actuals (actual number of rows, actual number of executions)
  - Used to determine if there is disparity between estimates and actuals

© SQLskills, All rights reserved.
http://www.SQLskills.com

Finding a Query Plan

Estimated Plan

- sys.dm_exec_query_plan
- sys.dm_exec_text_query_plan
- SSMS
  - SET SHOWPLAN_XML
  - Graphical Showplan
- Query Store

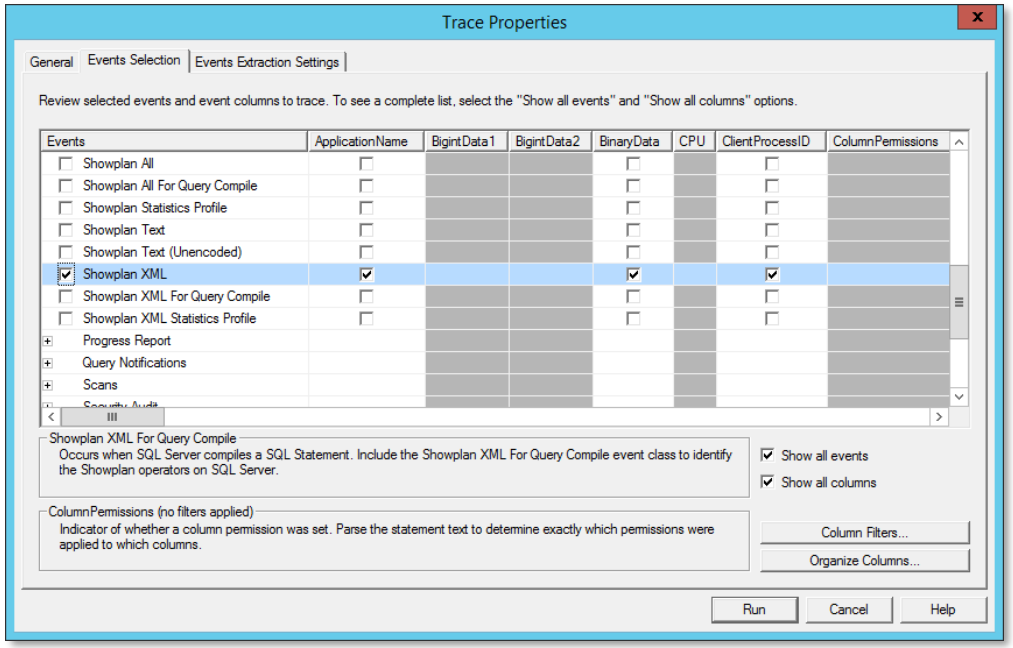
Actual Plan

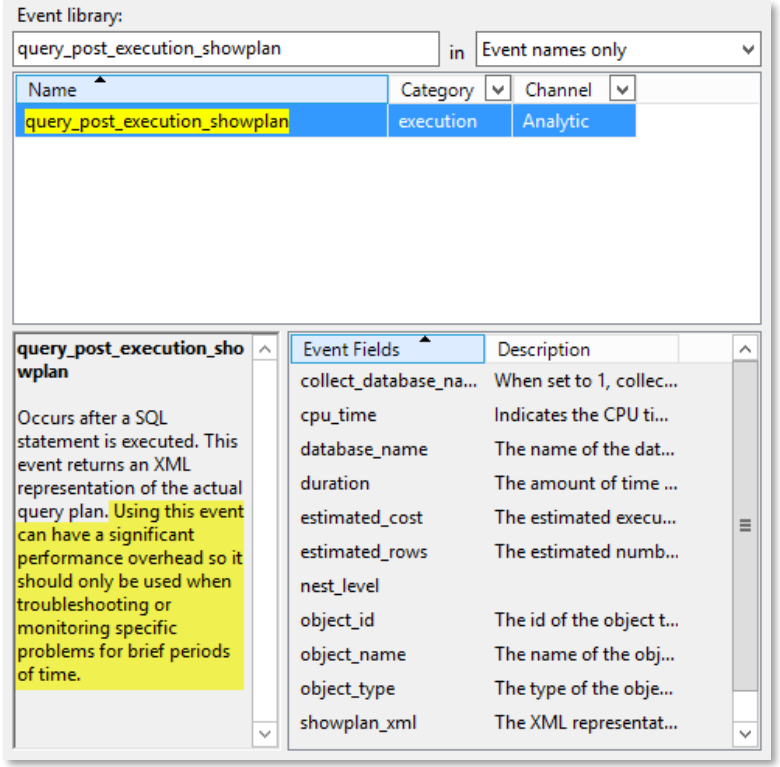
- SSMS
  - SET STATISTICS XML
  - Graphical Showplan
- Trace
- Extended Events

© SQLskills, All rights reserved.
http://www.SQLskills.com

Capturing Plans with Trace or Extended Events

- Plans can be captured with both SQL Trace and Extended Events
  - Neither is a recommended option

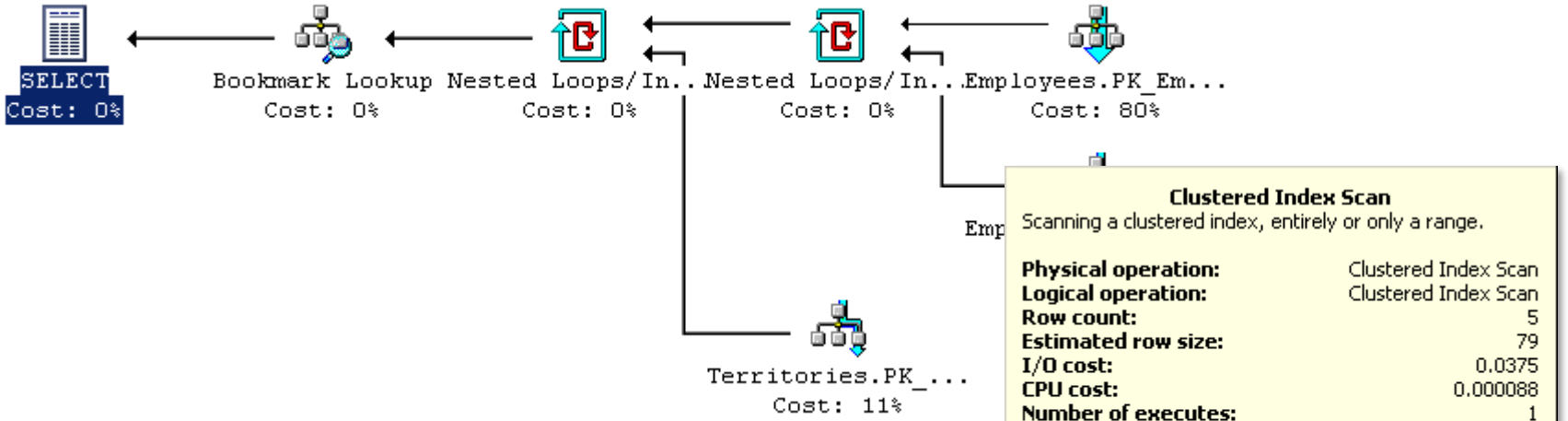




SQLskills immerse yourself in sql server
© SQLskills, All rights reserved.
http://www.SQLskills.com

Remember Plans in SQL 2000?

```
Query 1: Query cost (relative to the batch): 100.00%
Query text: SELECT e.LastName, e.Firstname, t.TerritoryDescription  FROM dbo.Employees e  JOIN (
```



	StmtText	StmtId	NodeId	Parent	PhysicalOp	LogicalOp
1	SELECT e.LastName, e.Firstname, t.TerritoryDescription ...	1	1	0	NULL	NULL
2	|--Bookmark Lookup(BOOKMARK:([Bmk1002]), OBJECT:([Nor...	1	3	1	Bookmark Lookup	Bookmark Lookup
3	|--Nested Loops(Inner Join, OUTER REFERENCES:([e...	1	4	3	Nested Loops	Inner Join
4	|--Nested Loops(Inner Join, OUTER REFERENCE...	1	5	4	Nested Loops	Inner Join
5	|--Clustered Index Scan(OBJECT:([North...	1	6	5	Clustered Index Scan	Clustered Index Scan
6	|--Index Seek(OBJECT:([Northwind].[dbo...	1	7	5	Index Seek	Index Seek
7	|--Index Seek(OBJECT:([Northwind].[dbo].[Te...	1	8	4	Index Seek	Index Seek

SQLskills immerse yourself in sql server

© SQLskills, All rights reserved.
http://www.SQLskills.com

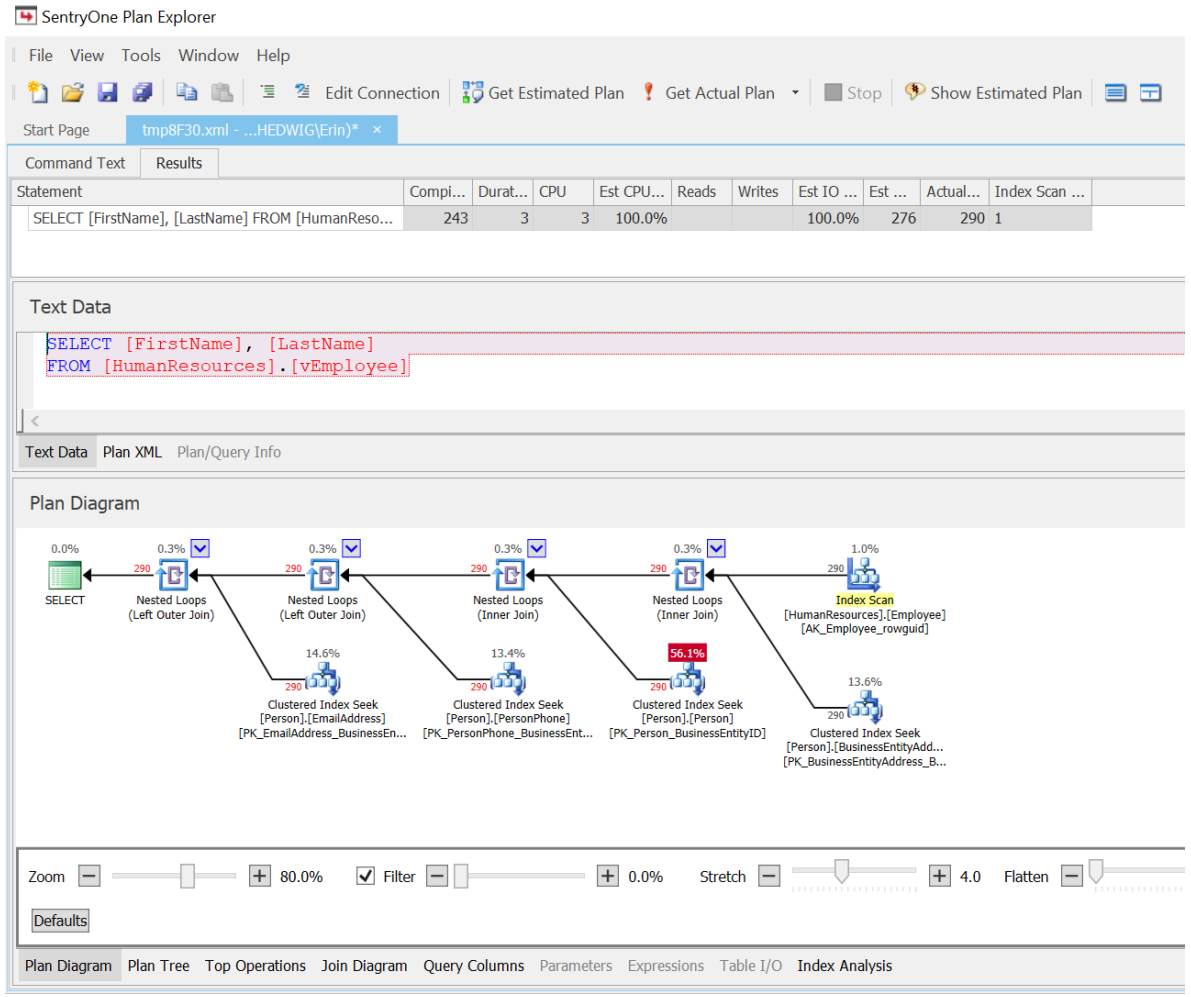
Plans in XML vs. Text

- XML showplan formats may seem more unwieldy and verbose than their text-based / tabular counterparts, but there are some key advantages:
  - Can be processed via XPath and XQuery
  - Easier to add attributes/elements from version to version
  - Save with .sqlplan extension to open in SSMS
- XML schema describes the structure of an XML document
  - http://schemas.microsoft.com/sqlserver/2004/07/showplan/
  - Much of what you can find in the graphical plan you can see in XML, so the primary benefit is in parsing and finding key elements and attributes programmatically

© SQLskills, All rights reserved.
http://www.SQLskills.com

Viewing Plans in SentryOne Plan Explorer

- Free, third-party tool
- Good for all sizes of plans
- Pulls out key information in a compact way
- Can open a .sqlplan file, or copy in the XML



SQLskills immerse yourself in sql server

© SQLskills, All rights reserved.
http://www.SQLskills.com

Plan Comparison

- Introduced in SSMS 17.0
  - Compare against a saved plan
  - Compare against a generated, actual plan
  - Compare two plans in Query Store
- Differences in operator properties are highlighted
- Includes additional details in the Scenarios tab when large differences in estimates and actuals exist

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Capturing and comparing query performance data

© SQLskills, All rights reserved.
http://www.SQLskills.com

Information in a plan

© SQLskills, All rights reserved.
http://www.SQLskills.com

Plans

Information Included

- Tables/indexes access
- Operators
  - Scans vs. seeks
  - Join type
- Estimated cost of each operation
- Number of rows expected
- More information continues to be added
  - Duration, I/O, wait statistics

Information *NOT* Included

- Locks
- Who executed the query
- Application/host name
- Wait statistics*

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Parameters

- The plan contains valuable information on compiled vs. runtime value
  - ParameterCompiledValue
  - ParameterRuntimeValue
- New attribute, ParameterDataType, added in 2016 SP1
  - https://support.microsoft.com/en-us/help/3190761/
- Some queries are very sensitive to different parameters
- Test different values using recompile (or a cold cache) to see if different plans are created for different values

© SQLskills, All rights reserved.
http://www.SQLskills.com

Trace Flags Enabled

- The TraceFlags element is available in SQL Server 2014 SP2 which lists the trace flags enabled (global or session) at the time of query compilation (IsCompileTime = true) and execution (IsCompileTime = false)
  - https://support.microsoft.com/en-us/kb/3170115

```
<TraceFlags IsCompileTime="true">
  <TraceFlag Value="4199" Scope="Global" />
</TraceFlags>
<TraceFlags IsCompileTime="false">
  <TraceFlag Value="4199" Scope="Global" />
</TraceFlags>
```

```
<TraceFlags IsCompileTime="true">
  <TraceFlag Value="8649" Scope="Session" />
</TraceFlags>
```


© SQLskills, All rights reserved.
http://www.SQLskills.com

Operators

- Operators are building blocks for a query, each one has a specific functionality
  - Some operators access data (scan, seek), others perform operations such as aggregations or joins
  - There is a one-to-many mapping of logical to physical
    - For example, an INNER JOIN could be implemented as a Loop, Hash, or Merge join
- Specific operators are not "good" or "bad"
  - Some can consume more resources than others or have overhead of which you should be aware
  - Some are more appropriate in given contexts
- SQL Server 2017 has 100+ operators
- Operators may also be referred to as iterators

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Optimization and Cost

- Each operation in the query tree is given a cost, and those are totaled to determine total cost for each possible plan
- Cost-based optimization looks at statistics and the size of the data that would be retrieved to estimate I/O
- Cost used to equate to elapsed time in seconds required to run on a specific Microsoft employee's machine (from SQL Server 7 era)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Operator Cost

- "Cost" today in the context of query plans is a unit-less measure
  - Cost <> time
  - Cost <> CPU
- Cost is used for relative comparison across plan operators and between plans
- "cost threshold for parallelism" option refers to this very same cost (default 5)
  - The default value of 5 is very low considering today's CPU architecture
- Estimated costs **remain estimates**, even for actual plans

© SQLskills, All rights reserved.
http://www.SQLskills.com

Cost Sensitivity to Row and Page Count

- Estimated costs are sensitive to specific pieces of data
  - e.g. the number of anticipated data pages and rows
- Consider the Clustered Index Scan operator
  - Estimated **I/O** cost of a Clustered Index Scan shows sensitivity to anticipated data **page** counts, but not row counts.
    - Think about the estimated I/O cost impact that high fragmentation may have (same number of rows across many partially-filled data pages)
  - Estimated **CPU** cost of a Clustered Index Scan sensitivity to **row** counts, but not data page counts.
    - Regarding row counts, even for narrow rows, consider the estimated CPU cost impact that high row counts may have.

© SQLskills, All rights reserved.
http://www.SQLskills.com

Time and Wait Stats

- In SQL Server 2016 SP1, time and wait type information is added to the plan
- Within the QueryTimeStats attribute you can see CpuTime and ElapsedTime
  - Also get UDF execution statistics in SQL 2016 SP2 and SQL 2017 CU3
- The top 10 wait types are also included within the WaitStats element
  - https://support.microsoft.com/en-us/help/3201552

```
<QueryTimeStats CpuTime="4" ElapsedTime="4" />
```

```
<WaitStats>
  <Wait WaitType="PAGEIOLATCH_SH" WaitTimeMs="4" WaitCount="31" />
  <Wait WaitType="SOS_SCHEDULER_YIELD" WaitTimeMs="5" WaitCount="843" />
  <Wait WaitType="ASYNC_NETWORK_IO" WaitTimeMs="72" WaitCount="6" />
  <Wait WaitType="MEMORY_ALLOCATION_EXT" WaitTimeMs="496" WaitCount="535073" />
</WaitStats>
```


© SQLskills, All rights reserved.
http://www.SQLskills.com

Operator Execution Statistics

- In SQL Server 2014 SP2 and SQL Server 2016+ per-operator query execution statistics are available in the XML
  - RunTimeCountersPerThread
  - Includes CPU time, elapsed time, IO information
  - https://support.microsoft.com/en-us/kb/3170113

```
<RunTimeInformation>
  <RunTimeCountersPerThread Thread="0" ActualRebinds="1" ActualRewinds="0" ActualRows="18097" ActualEndOfScans="1"
                            ActualExecutions="1" ActualElapsedms="11" ActualCPUms="11" ActualScans="0" ActualLogicalReads="0"
                            ActualPhysicalReads="0" ActualReadAheads="0" ActualLobLogicalReads="0" ActualLobPhysicalReads="0"
                            ActualLobReadAheads="0" />
</RunTimeInformation>
```

```
<RunTimeInformation>
  <RunTimeCountersPerThread Thread="4" ActualRows="5574" ActualRowsRead="5574" Batches="0" ActualEndOfScans="1" ActualExecut
  <RunTimeCountersPerThread Thread="3" ActualRows="7401" ActualRowsRead="7401" Batches="0" ActualEndOfScans="1" ActualExecut
  <RunTimeCountersPerThread Thread="1" ActualRows="4312" ActualRowsRead="4312" Batches="0" ActualEndOfScans="1" ActualExecut
  <RunTimeCountersPerThread Thread="2" ActualRows="14178" ActualRowsRead="14178" Batches="0" ActualEndOfScans="1" ActualExec
  <RunTimeCountersPerThread Thread="0" ActualRows="0" ActualEndOfScans="0" ActualExecutions="0" ActualElapsedms="0" ActualCP
</RunTimeInformation>
```


© SQLskills, All rights reserved.
http://www.SQLskills.com

Live Query Statistics and Query Profiling

- Introduced in SQL Server 2016 (SSMS 13.x), and can work with SQL Server 2014 through the current release
- Using this option enables the statistics profile infrastructure for the *current session*
  - This infrastructure can be enabled globally using TF 7412 or with an Extended Events session that captures the query_thread_profile event
    - **This introduces performance overhead**
  - In SQL Server 2016 SP2 CU3 and SQL Server 2017 CU11, you can enable the profile infrastructure globally with the query_plan_profile event, but metrics will only be collected for queries that include the QUERY_PLAN_PROFILE hint
    - **Note this does not include the statement**

© SQLskills, All rights reserved.
http://www.SQLskills.com

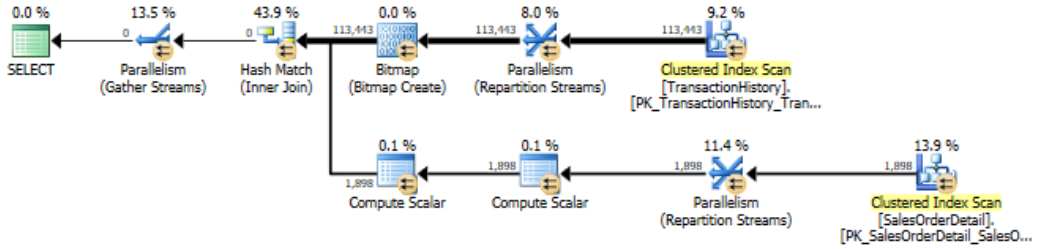
Demo: Essential information in a plan

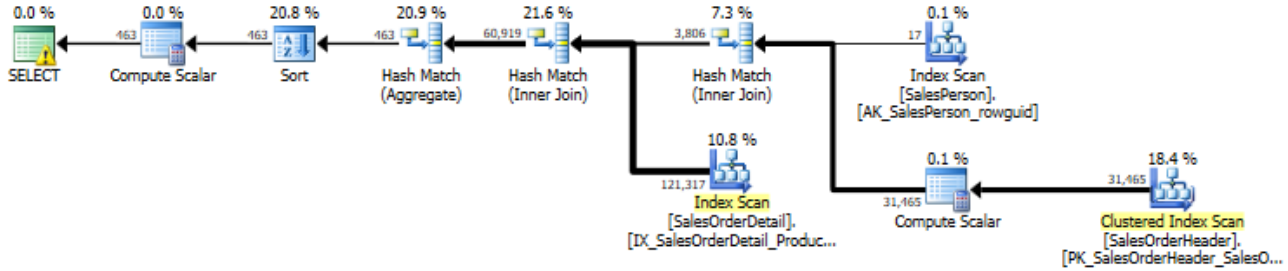
© SQLskills, All rights reserved.
http://www.SQLskills.com

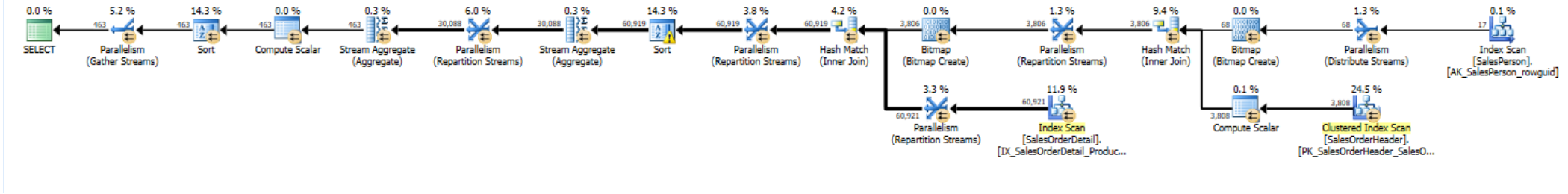
Plan analysis

© SQLskills, All rights reserved.
http://www.SQLskills.com

Where do you start with a plan?







© SQLskills, All rights reserved.
http://www.SQLskills.com

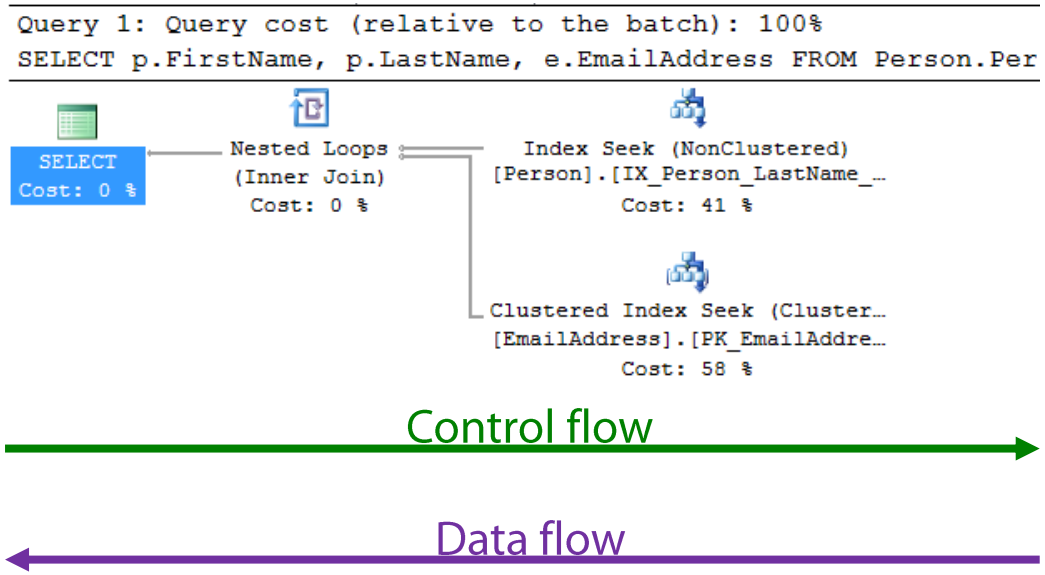
Where “should” you start with a plan?

- There is no one “right” answer
- Don’t believe statements that specific operators or behaviors translate to an absolute problem
  - We’ll discuss things to watch for, but see them as areas of investigation
    - Some might be red herrings instead of red flags
- The emphasis is on patterns you are more likely to see
- You have to practice reading and understanding plans to determine where you can improve a query, and this takes time

© SQLskills, All rights reserved.
http://www.SQLskills.com

Reading Plans

- An operator reads rows from a leaf-level data source OR from child operators and return rows to the parent
- Control flow starts at the root (left-to-right)
- Data flow starts at the leaf level (right-to-left)



SQLskills immerse yourself in sql server
© SQLskills, All rights reserved.
http://www.SQLskills.com

Data Access Operators

Table Scan

- Retrieve all rows from a table
- Indicates a heap table
- Is it a red flag?

Clustered Index Scan

- Retrieve rows from a clustered index
- Is it a red flag?

Nonclustered Index Scan

- Retrieve rows from a non-clustered index
- Is it a red flag?

Clustered Index Seek

- Retrieve rows based on a SEEK predicate from a clustered index
- Is it always a good thing?

Nonclustered Index Seek

- Retrieve rows based on a SEEK predicate from a non-clustered index
- Is it always a good thing?

Columnstore Index Scan

- Retrieve rows from a column-store index

© SQLskills, All rights reserved.
http://www.SQLskills.com

Predicates

- Seek Predicate
  - Used in actual index seek operation
    - Leveraging index keys
- Predicate (Residual Predicate)
  - Search condition that isn't SARGable – so it remains as an extra predicate
    - For Merge Join: check Graphical Showplan Properties for "Residual" value
    - For Hash Match: check "Probe Residual" value in plan itself (tooltip over operator)

© SQLskills, All rights reserved.
http://www.SQLskills.com

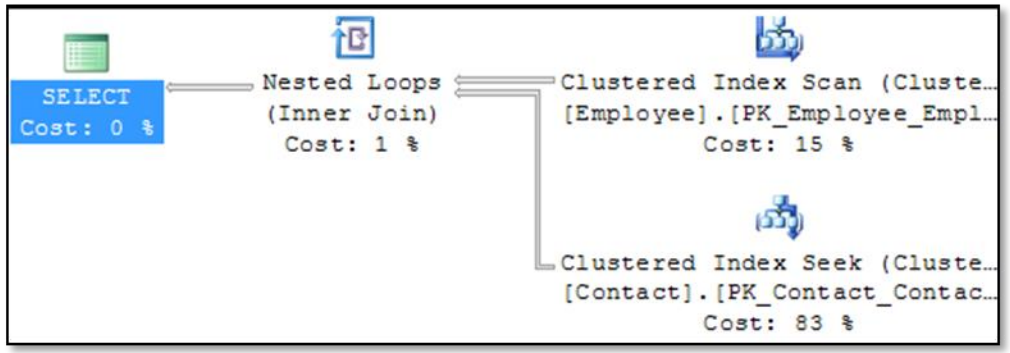
Filter

- Predicates can be evaluated within operators that read data from table/indexes
- Query Optimizer aims (when possible) to “push” filter down the tree (leaf level) to reduce rows moved
- If a predicate is high in cost or complexity a separate Filter operator may be used
- When you see these, take note of where they are happening
  - Late in the data flow can translate to higher overhead as the operators pull data

© SQLskills, All rights reserved.
http://www.SQLskills.com

Seek Predicate with No Residual

```
SELECT
[e].[EmployeeID],
[c].[Title],
[c].[FirstName],
[c].[MiddleName],
[c].[LastName],
[c].[Suffix]
FROM [HumamResources].[Employee] [e]
INNER JOIN [Person].[Contact] [c]
ON [c].[ContactID] = [e].[ContactID];
```



Clustered Index Seek (Clustered)

Scanning a particular range of rows from a clustered index.

Physical Operation	Clustered Index Seek
Logical Operation	Clustered Index Seek
Actual Number of Rows	290
Estimated I/O Cost	0.003125
Estimated CPU Cost	0.0001581
Number of Executions	290
Estimated Number of Executions	290
Estimated Operator Cost	0.071616 (83%)
Estimated Subtree Cost	0.071616
Estimated Number of Rows	1
Estimated Row Size	187 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	True
Node ID	3

Object
[AdventureWorks].[Person].[Contact].
[PK_Contact_ContactID] [c]

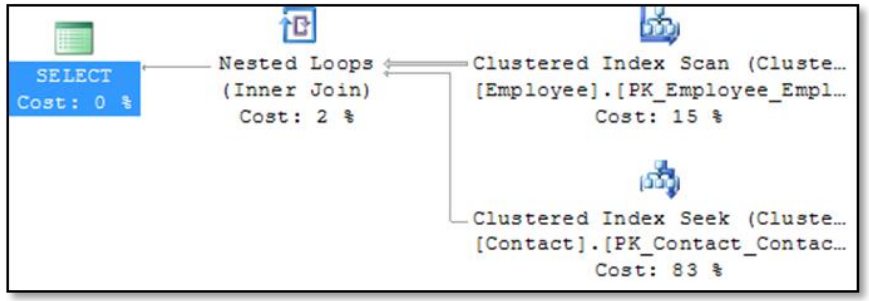
Output List
[AdventureWorks].[Person].[Contact].Title,
[AdventureWorks].[Person].[Contact].FirstName,
[AdventureWorks].[Person].[Contact].MiddleName,
[AdventureWorks].[Person].[Contact].LastName,
[AdventureWorks].[Person].[Contact].Suffix

Seek Predicates
Seek Keys[1]: Prefix: [AdventureWorks].[Person].
[Contact].ContactID = Scalar Operator([AdventureWorks].
[HumanResources].[Employee].[ContactID] as [e].
[ContactID])

© SQLskills, All rights reserved.
http://www.SQLskills.com

Seek Predicate *with* a Residual

```
SELECT
[e].[EmployeeID],
[c].[Title],
[c].[FirstName],
[c].[MiddleName],
[c].[LastName],
[c].[Suffix],
[c].[EmailAddress]
FROM [HumamResources].[Employee] [e]
INNER JOIN [Person].[Contact] [c]
ON [c].[ContactID] = [e].[ContactID]
WHERE [LastName] = 'Ellerbrock';
```



Clustered Index Seek (Clustered)

Scanning a particular range of rows from a clustered index.

Physical Operation	Clustered Index Seek
Logical Operation	Clustered Index Seek
Actual Number of Rows	1
Estimated I/O Cost	0.003125
Estimated CPU Cost	0.0001581
Number of Executions	290
Estimated Number of Executions	290
Estimated Operator Cost	0.071616 (83%)
Estimated Subtree Cost	0.071616
Estimated Number of Rows	1
Estimated Row Size	200 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	True
Node ID	3

Predicate

[AdventureWorks].[Person].[Contact].[LastName] as [c].[LastName]=N'Ellerbrock'

Object

[AdventureWorks].[Person].[Contact].[PK_Contact_ContactID] [c]

Output List

[AdventureWorks].[Person].[Contact].Title, [AdventureWorks].[Person].[Contact].FirstName, [AdventureWorks].[Person].[Contact].MiddleName, [AdventureWorks].[Person].[Contact].LastName, [AdventureWorks].[Person].[Contact].Suffix, [AdventureWorks].[Person].[Contact].EmailAddress

Seek Predicates

Seek Keys[1]: Prefix: [AdventureWorks].[Person].[Contact].ContactID = Scalar Operator([AdventureWorks].[HumanResources].[Employee].[ContactID] as [e].[ContactID])

© SQLskills, All rights reserved.
http://www.SQLskills.com

Additional Predicate Diagnostics

- ActualRowsRead attribute was added in SQL Server 2012 SP3 and SQL Server 2014 SP2 (also in SQL Server 2016+) to provide additional information about predicate evaluation
  - https://support.microsoft.com/en-us/kb/3107397
- Compare against ActualRows, which is the number of rows output from the operator

Index Seek (NonClustered)

Scan a particular range of rows from a nonclustered index.

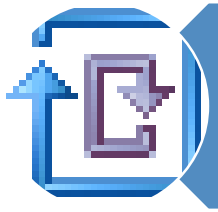
Physical Operation	Index Seek
Logical Operation	Index Seek
Actual Execution Mode	Row
Estimated Execution Mode	Row
Storage	RowStore
Actual Number of Rows	59
Number of Rows Read	348
Actual Number of Batches	0
Estimated I/O Cost	0.0046065
Estimated Operator Cost	0.0051463 (100%)
Estimated CPU Cost	0.0005398
Estimated Subtree Cost	0.0051463
Estimated Number of Executions	1
Number of Executions	1
Estimated Number of Rows	69.6
Estimated Row Size	53 B

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Analyzing plans Part I

© SQLskills, All rights reserved.
http://www.SQLskills.com

Join Operators

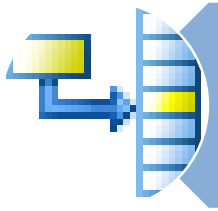


Nested Loop

- Uses each row from one input to find rows from a second input that satisfy the join predicate
- Usually seen with smaller data sets and lookups, where the inner input is indexed on the join predicate
- See with Key Lookup and RID Lookup

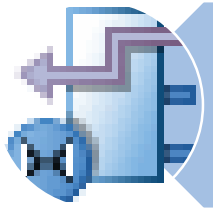
Merge Join

- Joins two inputs which are sorted on the joining columns and returns matching rows
- Typically benefits moderate-sized data sets



Hash Join

- Joins two unsorted inputs and outputs the matching rows
- Often seen with large data sets and when grouping aggregates



Adaptive Join

- Indicates that the optimizer has the choice of a Hash Join or Nested Loop
- Plan is still cached, join type is determined at run-time
- Ideal for workloads with varied inputs/skewed data

© SQLskills, All rights reserved.
http://www.SQLskills.com

Join Considerations

- Beware of advice telling you that specific join types (or operators, for that matter) are "good" or "bad"
- Each join type has performance characteristics of which to be aware – take note of times when the join type is not expected
- Join hints and/or forcing order = red flag
  - Generally, "edge" cases or extreme tuning scenarios warrant their use
  - Otherwise, ask questions and find out why this is happening

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Analyzing plans Part II

© SQLskills, All rights reserved.
http://www.SQLskills.com

Query Memory

- Some queries require memory to store data while sorting and joining rows, thus a memory grant is requested
  - Lifetime of the grant is equivalent to the lifetime of the query
- When available memory is insufficient, queries that require lots of memory may wait to execute (RESOURCE_SEMAPHORE wait type)
- Under-estimating memory (due to cardinality estimation issues) can cause spills to tempdb (I/O)
- Over-estimating memory can reduce concurrency!
- Memory grant information can be found in the plan

© SQLskills, All rights reserved.
http://www.SQLskills.com

Operator Memory

- Each type of operator requires varying amounts of memory in order to perform the associated operation
- Some operators require *more* memory because they cache rows
  - More rows = more memory required
    - SQL Server performs estimates of the required memory and tries to reserve the memory grant prior to execution
    - This is where cardinality estimation is critical

© SQLskills, All rights reserved.
http://www.SQLskills.com

Stop-and-Go Operators

- Stop-and-go operators must read ALL rows from the child operator before it can pass rows to the parent or perform specific actions
- Examples of stop-and-go operators include:
  - Hash Join
    - Outer (top) table in “blocking input”, not inner
    - Must read and process the entire build phase before the probe phase can start
  - Sort
  - Hash Aggregate
- Examples of operators that can keep streaming one row for every row that is read:
  - Nested Loop
  - Merge Join

© SQLskills, All rights reserved.
http://www.SQLskills.com

Sort

- As named, this operator orders rows received from an input
- Variations include:
  - Distinct Sort
  - Top N Sort
- Noteworthy: Sort tempdb spills
- Keep an eye on these for the following reasons:
  - Sort can occur in tempdb for large data sets and memory constraints (see Sort Warnings)
  - Has resource overhead (CPU / I/O / memory)
  - May not be needed if you have supporting indexes or unnecessary ORDER BY

© SQLskills, All rights reserved.
http://www.SQLskills.com

Batch Mode Memory Grant Feedback

- To better manage query memory grants, a feedback mechanism was introduced in SQL Server 2017 for batch mode operators
- The memory required for a query can be recalculated after query execution, and the grant numbers are updated in the cached plan
- Memory grants respect the limits set by Resource Governor or query hints
- Feedback loop can be disabled after multiple executions with variable memory requirements
  - Monitor with Extended Events

© SQLskills, All rights reserved.
http://www.SQLskills.com

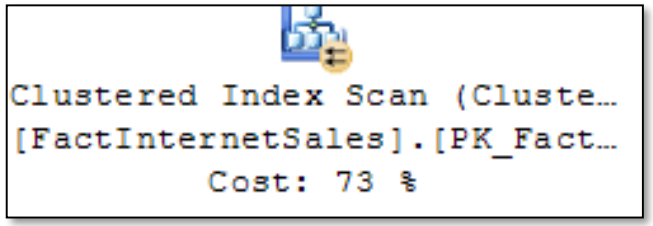
Parallelism

- Query optimization can generate either a serial or parallel plan
  - First determine cost of a serial plan, and if that cost is greater than CTP, consider parallel plans
  - The Query Optimizer ultimately caches one of them, not both
  - Parallel plans can then be used as intended, or run in serial with parallelism operators removed
- The MAXDOP server setting limits:
  - The maximum number of threads that can be used per operator
  - It does not limit the total number of threads for the plan

© SQLskills, All rights reserved.
http://www.SQLskills.com

Identifying Parallelism in the Plan

- When parallelism occurs in query execution, you will see exchange operators in the plan
  - Distribute Streams
  - Repartition Streams
  - Gather Streams
- Within the XML Showplan you'll see RelOp:
  - Parallel="true"
- You will also see the parallelism icon in the graphic for other operators that can run in parallel or serial modes
- Not all operators are parallel-aware



© SQLskills, All rights reserved.
http://www.SQLskills.com

NonParallelPlanReason

- Introduced in SQL Server 2012
  - Identify why a plan didn't run in parallel
  - Not all-encompassing, but a step in the right direction
  - Examples:
    - MaxDOPSetToOne (hint or max degree of parallelism)
    - EstimatedDOPIsOne (processor affinity)
  - Post by Simon Sabin that lists reasons for 2012:
    - http://sqlblogcasts.com/blogs/simons/archive/2015/04/26/non-parallelizable-operations-in-sql-server.aspx

© SQLskills, All rights reserved.
http://www.SQLskills.com

Parallelism Performance Aspects

- Not inherently bad or good
  - Context matters, for example OLTP vs. DW
- Indicates higher cost query, so that should attract attention
- Some objects and operators inhibit parallelism
  - UDFs and TVFs (CLR, scalar, multi-statement), built-in functions (like OBJECT_ID), TOP
- Watch for data skew across threads
  - You can check this in XML or Properties window
  - Remember to also check for CXPACKET waits on non-zero thread IDs
- Watch for memory pressure
  - Increased memory grant requirements for parallel operations

© SQLskills, All rights reserved.
http://www.SQLskills.com

Demo: Analyzing plans Part III

© SQLskills, All rights reserved.
http://www.SQLskills.com

Summary: the "Watch List"

- "Thick" lines
- High estimated cost within a query
- Scans (Index/Table)
- Lookups
- Late filtering or residual predicates
- Unexpected join selection
- Stop-and-go operators
- Sort operations (Query Optimizer added or not)
- Warnings
- Parameter sensitivity
- Hint or plan guide usage (forcing specific plan operations)

© SQLskills, All rights reserved.
http://www.SQLskills.com

Key Takeaways

- There are many sources of data for query performance, and numerous components worth investigating in a query plan
- Where to start when analyzing performance, and a plan, is a personal preference, and something that evolves over time…there is no "right way"
- Query tuning doesn't just consist of reviewing a plan in isolation, take advantage of multiple sources of information to improve performance
  - Understand what behavior to "expect" for operators you see repeatedly, and take note of behaviors outside the norm
  - Delve into plan details via the Properties window, and by mining the XML for information not easily surfaced in the UI

© SQLskills, All rights reserved.
http://www.SQLskills.com

Additional Resources

- Pluralsight Courses
  - SQL Server: Analyzing Query Performance for Developers
    - https://bit.ly/2HKaoYo
  - SQL Server: Query Plan Analysis
    - http://bit.ly/ZyExm6
  - SQL Server: Introduction to Query Store
    - https://bit.ly/2J42aJP

© SQLskills, All rights reserved.
http://www.SQLskills.com

Additional Resources

- Whitepapers
  - *Statistics Used by the Query Optimizer in Microsoft SQL Server 2008*
    - http://msdn.microsoft.com/en-us/library/dd535534.aspx
  - *Optimizing Your Query Plans with the SQL Server 2014 Cardinality Estimator*
    - *http://msdn.microsoft.com/en-us/library/dn673537.aspx*
- Books
  - SQL Server Execution Plans by Grant Fritchey

© SQLskills, All rights reserved.
http://www.SQLskills.com

Additional Resources

- Craig Freedman's SQL Blog (not updated recently but very good!):
  - http://blogs.msdn.com/b/craigfr/
- Conor Cunningham Blog(s):
  - http://blogs.msdn.com/b/conor_cunningham_msft/
  - http://www.sqlskills.com/blogs/conor/
- Paul White's Blog
  - http://sqlblog.com/blogs/paul_white/

© SQLskills, All rights reserved.
http://www.SQLskills.com

Review

- Query Metrics
- Capturing and comparing query plans
- Information in a plan
- Plan analysis

© SQLskills, All rights reserved.
http://www.SQLskills.com

Thank you!