SQLServerFast.com Execution Plan Video Training

Block 2: Reading data

Level: Basic

Chapter 3: Seek operators

Seek operators

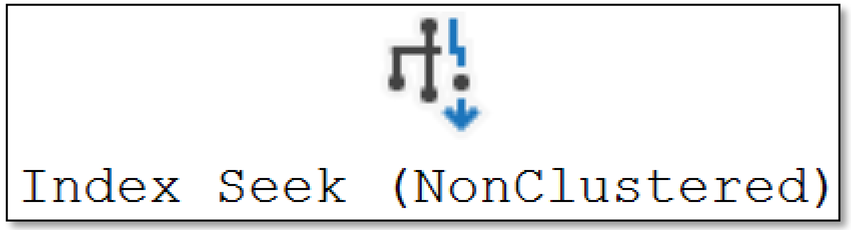
- Targeted search for specific data
  - Uses structure of the object searched
    - Not supported for heaps

Seek operators

- Targeted search for specific data
  - Uses structure of the object searched
    - Not supported for heaps and columnstore indexes
    - Supported for all other indexes
      - Nonclustered index → Index Seek
      - Clustered index → Clustered Index Seek
      - Memory-optimized nonclustered index → Index Seek
      - Memory-optimized nonclustered hash index → Index Seek

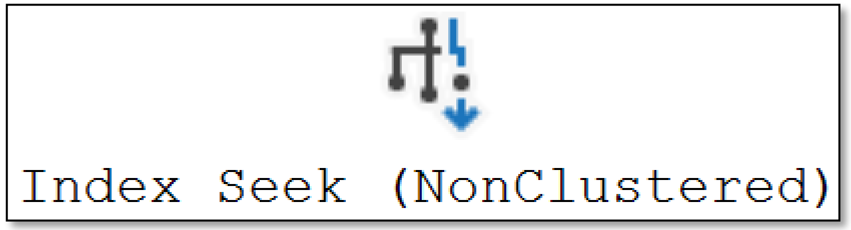
Seek operators

- Index Seek
  - Sometimes called nonclustered Index Seek for clarity
  - Reads data from a nonclustered index
    - On-disk rowstore
    - Memory-optimized (Bw-tree)
    - Memory-optimized Hash



Seek operators

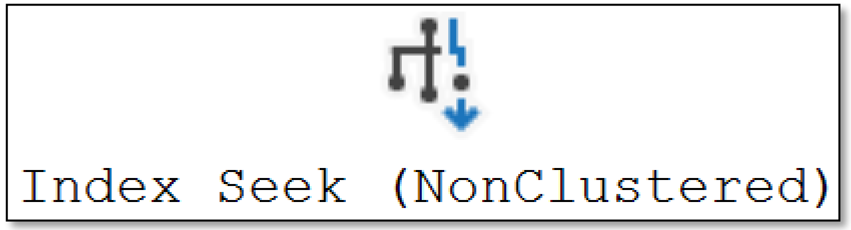
- Index Seek
  - Sometimes called nonclustered Index Seek for clarity
  - Reads data from a nonclustered index
  - Uses structure of the index to find specific information
    - Finds these rows directly, without reading other information
    - Some overhead is unavoidable



Seek operators

Index Seek

- Sometimes called nonclustered Index Seek for clarity
- Reads data from a nonclustered index
- Uses structure of the index to find specific information
- Two behaviors
  - Singleton lookup
    - When search criteria plus index definition guarantees at most one row
  - Range seek
    - When more than one row might match

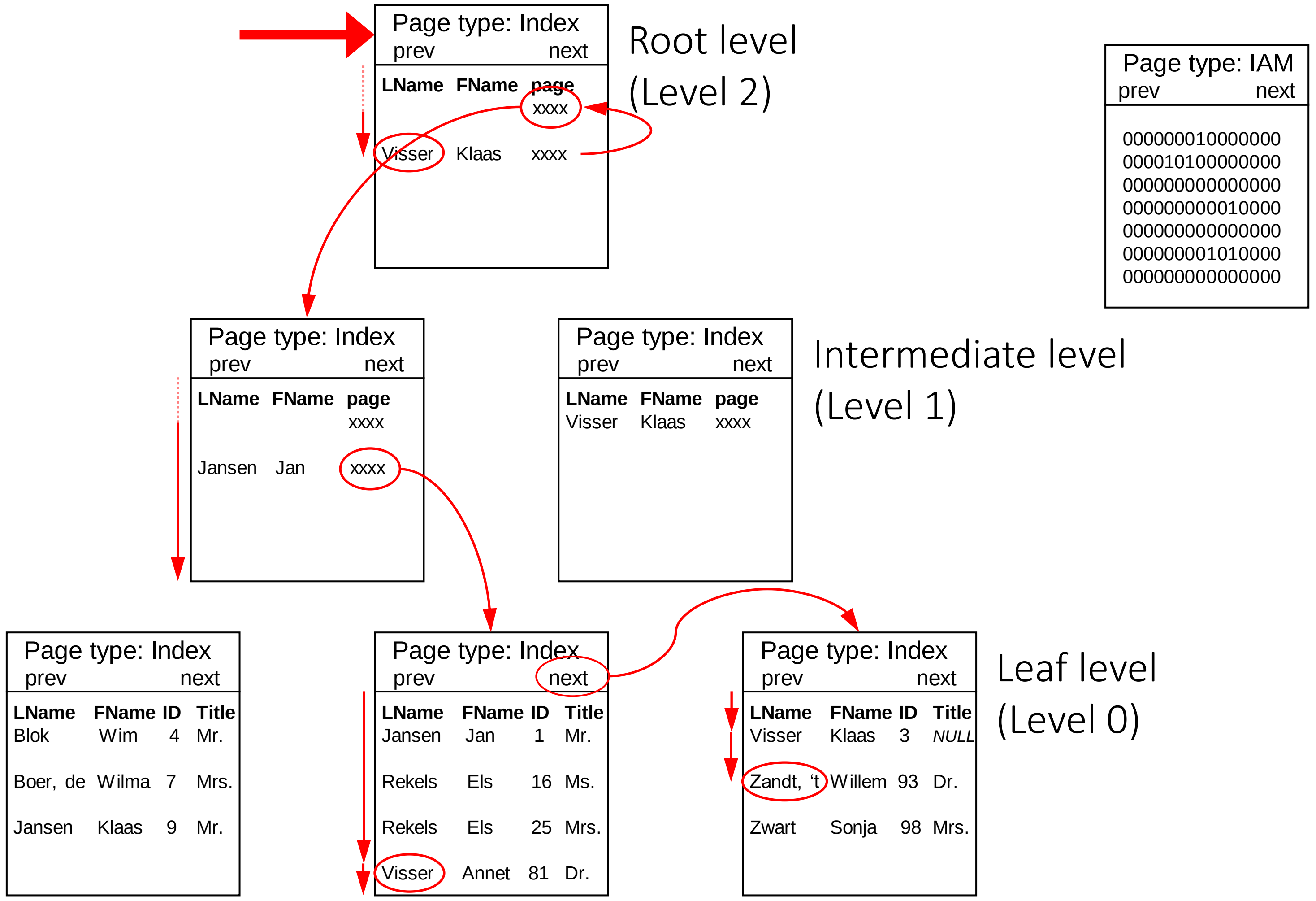


Seek operators

Index Seek

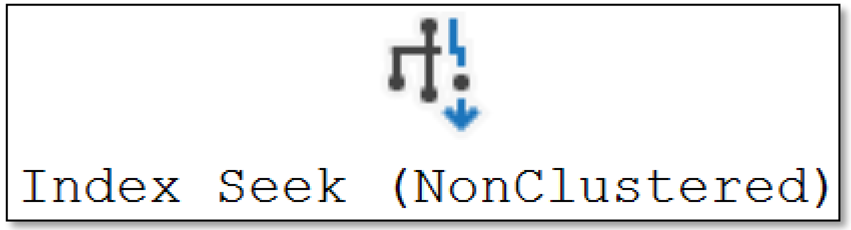
Range seek

```
SELECT Title,
       FirstName,
       LastName
FROM   dbo.PersonsCluster
WHERE  LastName = 'Visser';
```



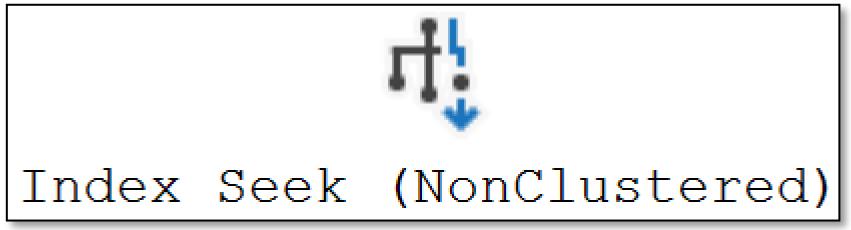
Seek operators

- Index Seek
  - Sometimes called nonclustered Index Seek for clarity
  - Reads data from a nonclustered index
  - Uses structure of the index to find specific information
  - Two behaviors
  - Requires “sargability”
    - SARG = Search ARGument
    - Column on its own on one side of the comparison
    - Not sargable when column included in expression
    - Optimizer can sometimes rewrite



Seek operators

- Index Seek
  - Sometimes called nonclustered Index Seek for clarity
  - Reads data from a nonclustered index
  - Uses structure of the index to find specific information
  - Two behaviors
  - Requires “sargability”
    - SARG = Search ARGument
    - Column on its own on one side of the comparison
    - Not sargable when column included in expression
    - Optimizer can sometimes rewrite (not guaranteed)



Seek operators

Index Seek

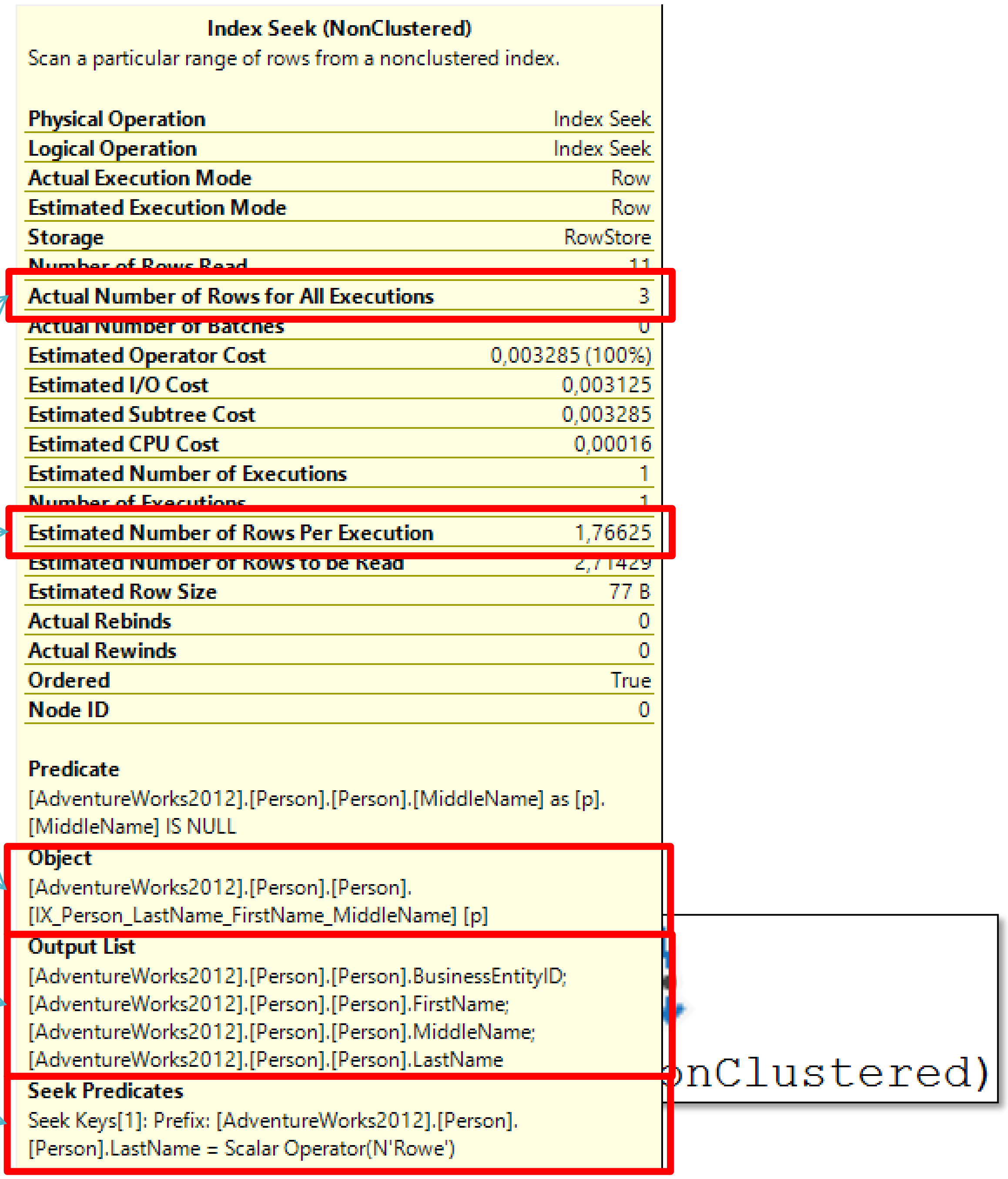
Properties (popup)

Actual and estimated row count

Database, schema, table, index
(and optionally alias)

Data returned to caller

Specification of rows to be read
(drives the index navigation process)



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
Number of Rows Read	11
Actual Number of Rows for All Executions	3
Actual Number of Batches	0
Estimated Operator Cost	0,003285 (100%)
Estimated I/O Cost	0,003125
Estimated Subtree Cost	0,003285
Estimated CPU Cost	0,00016
Estimated Number of Executions	1
Number of Executions	1
Estimated Number of Rows Per Execution	1,76625
Estimated Number of Rows to be Read	2,71429
Estimated Row Size	77 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	True
Node ID	0

Predicate
[AdventureWorks2012].[Person].[Person].[MiddleName] as [p].[MiddleName] IS NULL

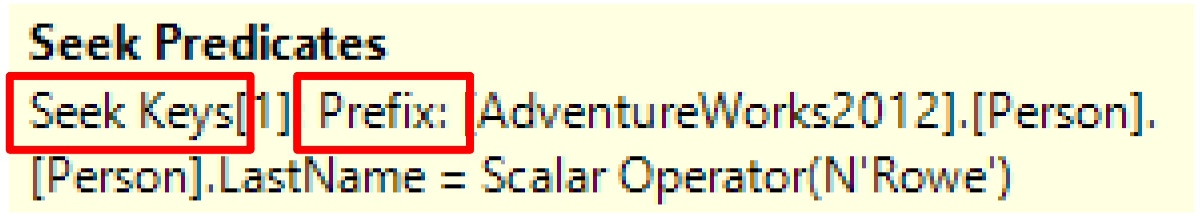
Object
[AdventureWorks2012].[Person].[Person].[IX_Person_LastName_FirstName_MiddleName] [p]

Output List
[AdventureWorks2012].[Person].[Person].BusinessEntityID; [AdventureWorks2012].[Person].[Person].FirstName; [AdventureWorks2012].[Person].[Person].MiddleName; [AdventureWorks2012].[Person].[Person].LastName

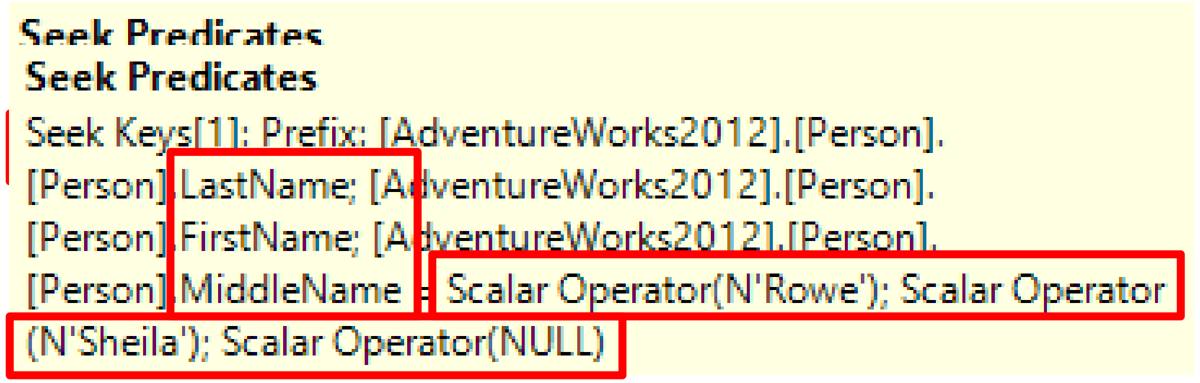
Seek Predicates
Seek Keys[1]: Prefix: [AdventureWorks2012].[Person].[Person].LastName = Scalar Operator(N'Rowe')

Seek operators

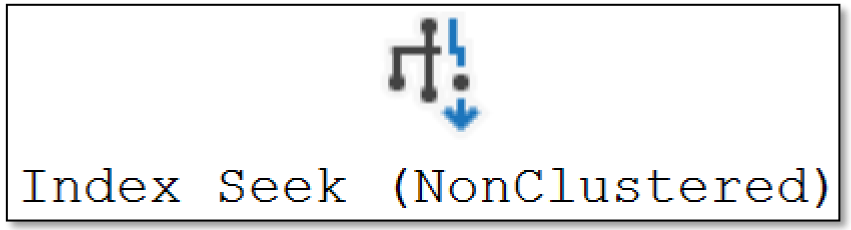
- Index Seek
  - *Seek Predicates* property
    - Contains *Seek Keys* element
      - Equality: “Prefix”
        - Can use NULL comparison
        - Multiple columns possible
          - Columns listed in index order
          - First all columns, then all values



Seek Predicates
Seek Keys[1]: Prefix: [AdventureWorks2012].[Person].
[Person].LastName = Scalar Operator(N'Rowe')

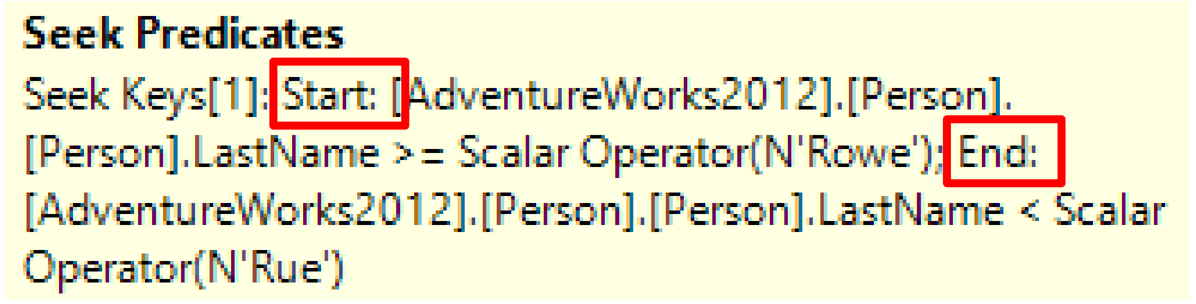


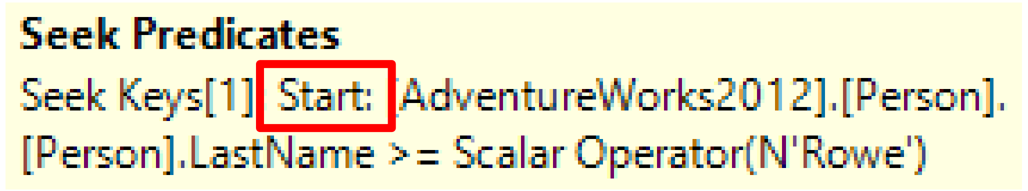
Seek Predicates
Seek Keys[1]: Prefix: [AdventureWorks2012].[Person].
[Person].LastName; [AdventureWorks2012].[Person].
[Person].FirstName; [AdventureWorks2012].[Person].
[Person].MiddleName = Scalar Operator(N'Rowe'); Scalar Operator
(N'Sheila'); Scalar Operator(NULL)

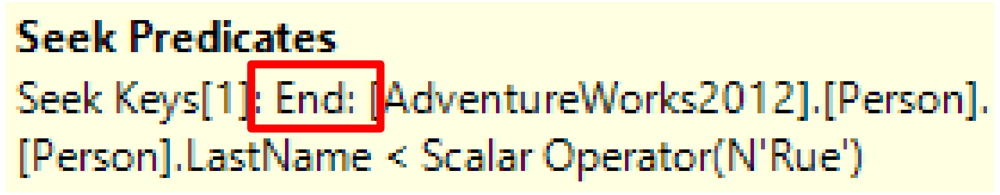


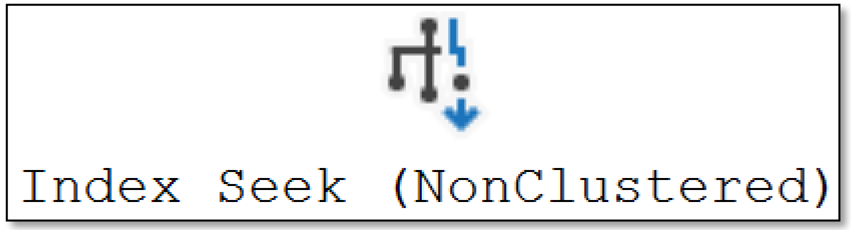
Seek operators

- Index Seek
  - *Seek Predicates* property
    - Contains *Seek Keys* element
      - Equality: “Prefix”
      - Range: “Start” and “End”
        - Both present (range from X to Y)
        - Only Start present (from X to end of table)
        - Only End present (from start of table to Y)


Seek Predicates
Seek Keys[1]: Start: [AdventureWorks2012].[Person].[Person].LastName >= Scalar Operator(N'Rowe'); End: [AdventureWorks2012].[Person].[Person].LastName < Scalar Operator(N'Rue')

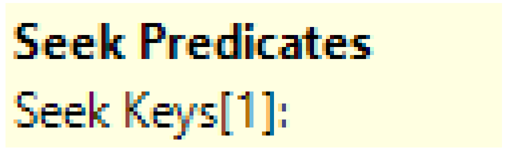

Seek Predicates
Seek Keys[1]: Start: [AdventureWorks2012].[Person].[Person].LastName >= Scalar Operator(N'Rowe')


Seek Predicates
Seek Keys[1]: End: [AdventureWorks2012].[Person].[Person].LastName < Scalar Operator(N'Rue')

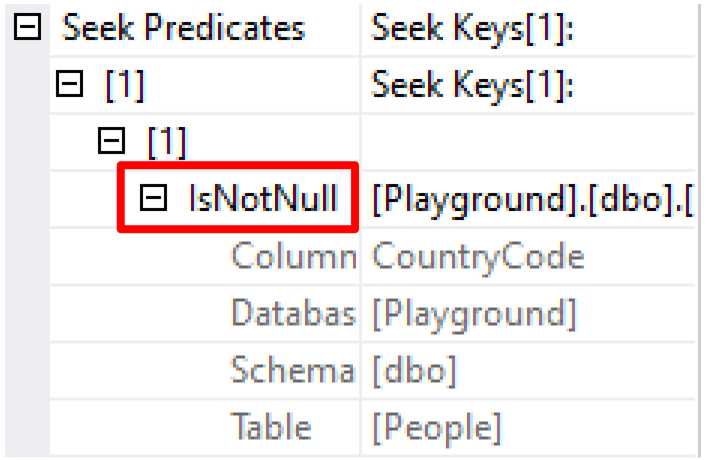


Seek operators

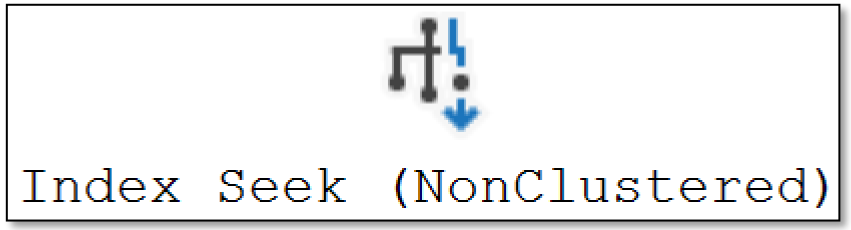
- Index Seek
  - *Seek Predicates* property
    - Contains *Seek Keys* element
      - Equality: “Prefix”
      - Range: “Start” and “End”
      - Special range: “IsNotNull”
        - Starts at first non-null value
        - No end (until end of table)



Seek Predicates
Seek Keys[1]:

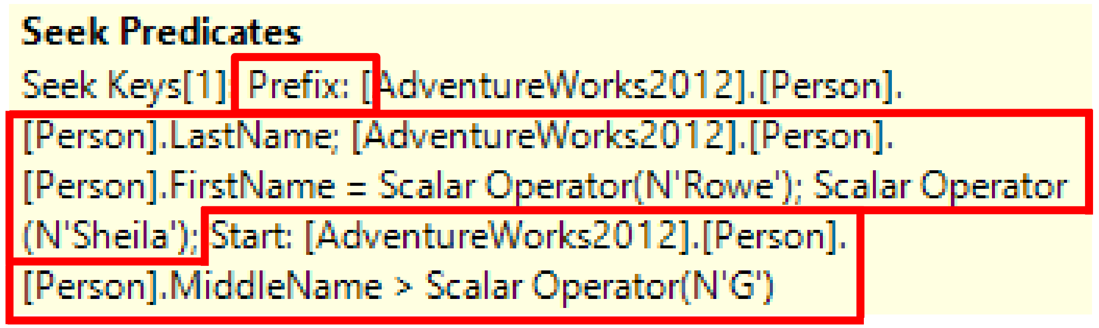


⊟ Seek Predicates	Seek Keys[1]:
⊟ [1]	Seek Keys[1]:
⊟ [1]	
⊟ IsNotNull	[Playground].[dbo].[
Column	CountryCode
Databas	[Playground]
Schema	[dbo]
Table	[People]

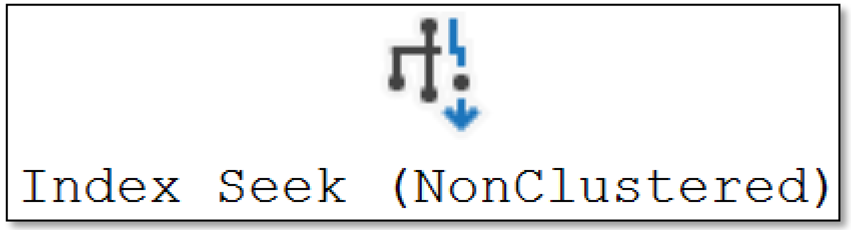


Seek operators

- Index Seek
  - *Seek Predicates* property
    - Contains *Seek Keys* element
      - Equality: “Prefix”
      - Range: “Start” and “End”
      - Special range: “IsNotNull”
      - Combination
        - Equality on one or more leading columns
        - Range on the first column not in the equality

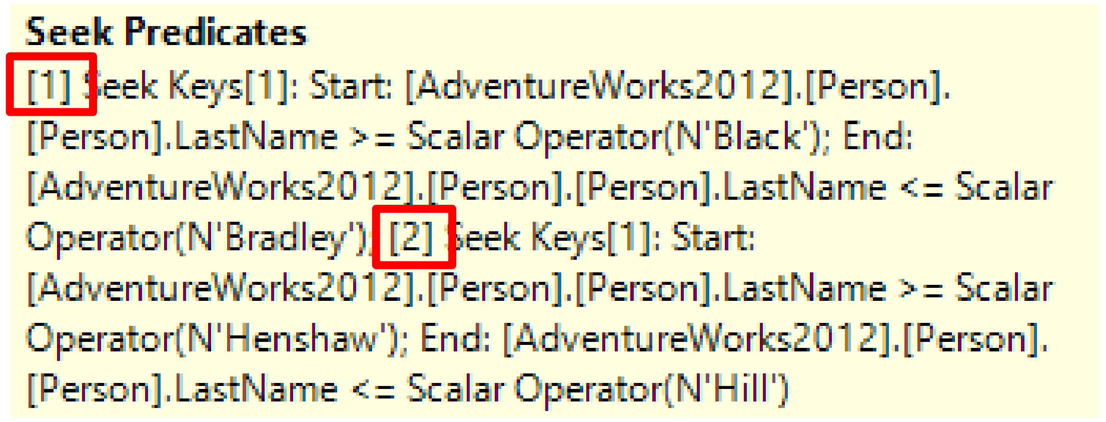


Seek Predicates
Seek Keys[1]: Prefix: [AdventureWorks2012].[Person].[Person].LastName; [AdventureWorks2012].[Person].[Person].FirstName = Scalar Operator(N'Rowe'); Scalar Operator(N'Sheila'); Start: [AdventureWorks2012].[Person].[Person].MiddleName > Scalar Operator(N'G')



Seek operators

- Index Seek
  - *Seek Predicates* property
    - Contains *Seek Keys* element
      - Equality: “Prefix”
      - Range: “Start” and “End”
      - Special range: “IsNotNull”
      - Combination
    - May contain multiple *Seek Keys* elements
      - Numbered
      - Processed in order



Seek Predicates

[1] Seek Keys[1]: Start: [AdventureWorks2012].[Person].[Person].LastName >= Scalar Operator(N'Black'); End: [AdventureWorks2012].[Person].[Person].LastName <= Scalar Operator(N'Bradley'), [2] Seek Keys[1]: Start: [AdventureWorks2012].[Person].[Person].LastName >= Scalar Operator(N'Henshaw'); End: [AdventureWorks2012].[Person].[Person].LastName <= Scalar Operator(N'Hill')

Index Seek (NonClustered)

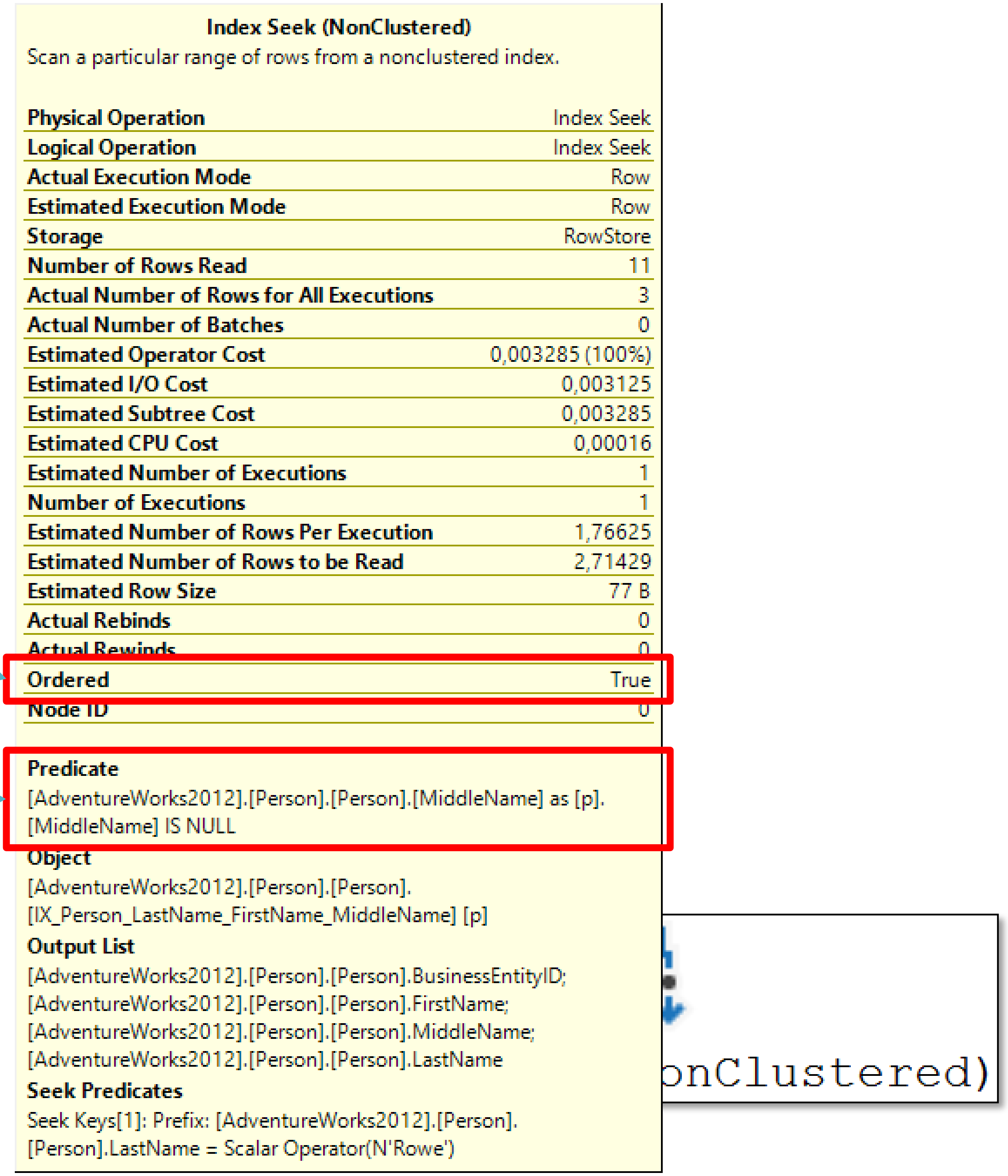
Seek operators

Index Seek

Properties (popup)

Index Seek is always ordered

“Pushed down” additional filter
(does not use the index structure)



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
Number of Rows Read	11
Actual Number of Rows for All Executions	3
Actual Number of Batches	0
Estimated Operator Cost	0,003285 (100%)
Estimated I/O Cost	0,003125
Estimated Subtree Cost	0,003285
Estimated CPU Cost	0,00016
Estimated Number of Executions	1
Number of Executions	1
Estimated Number of Rows Per Execution	1,76625
Estimated Number of Rows to be Read	2,71429
Estimated Row Size	77 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	True
Node ID	0

Predicate
[AdventureWorks2012].[Person].[Person].[MiddleName] as [p].[MiddleName] IS NULL

Object
[AdventureWorks2012].[Person].[Person].[IX_Person_LastName_FirstName_MiddleName] [p]

Output List
[AdventureWorks2012].[Person].[Person].BusinessEntityID; [AdventureWorks2012].[Person].[Person].FirstName; [AdventureWorks2012].[Person].[Person].MiddleName; [AdventureWorks2012].[Person].[Person].LastName

Seek Predicates
Seek Keys[1]: Prefix: [AdventureWorks2012].[Person].[Person].LastName = Scalar Operator(N'Rowe')

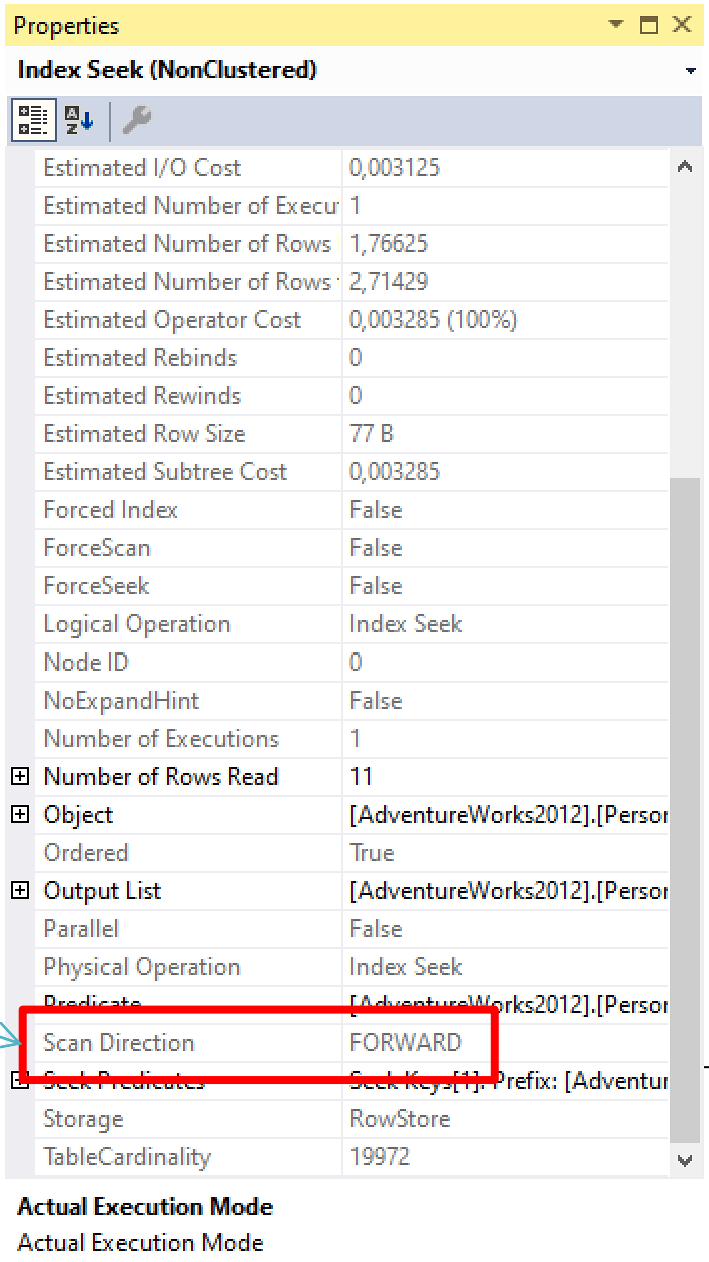
Seek operators

Index Seek

Properties (full list)

Scan direction: Usually FORWARD
When BACKWARD:

- Range scan starts at "End", moves back
- Multiple Seek Keys process in reverse order

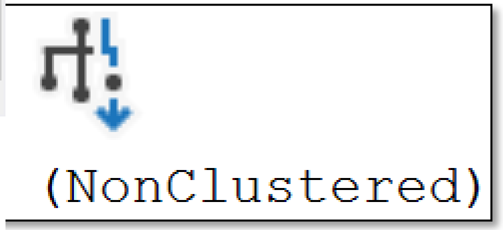


Properties

Index Seek (NonClustered)

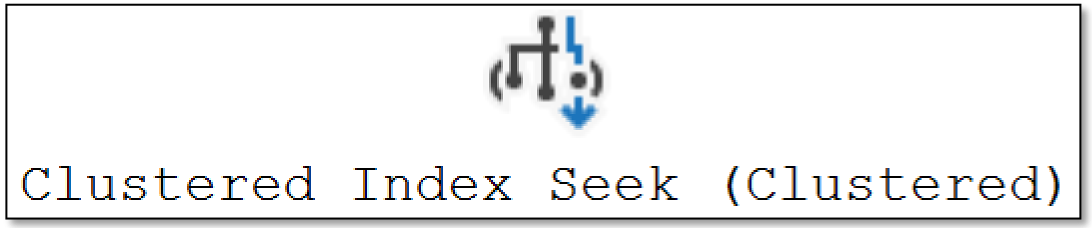
Estimated I/O Cost	0,003125
Estimated Number of Execu	1
Estimated Number of Rows	1,76625
Estimated Number of Rows	2,71429
Estimated Operator Cost	0,003285 (100%)
Estimated Rebinds	0
Estimated Rewinds	0
Estimated Row Size	77 B
Estimated Subtree Cost	0,003285
Forced Index	False
ForceScan	False
ForceSeek	False
Logical Operation	Index Seek
Node ID	0
NoExpandHint	False
Number of Executions	1
⊞ Number of Rows Read	11
⊞ Object	[AdventureWorks2012].[Persor
Ordered	True
⊞ Output List	[AdventureWorks2012].[Persor
Parallel	False
Physical Operation	Index Seek
Predicate	[AdventureWorks2012].[Persor
Scan Direction	FORWARD
⊞ Seek Predicates	Seek Keys[1]: Prefix: [Adventur
Storage	RowStore
TableCardinality	19972

Actual Execution Mode
Actual Execution Mode



Seek operators

- Clustered Index Seek
  - Behaves the same as (nonclustered) Index Seek
  - Reads data from a ***clustered*** index
    - On-disk rowstore
  - Two behaviors
    - Singleton lookup
    - Range seek
  - Uses structure of the index to find specific information
  - Requires “sargability”

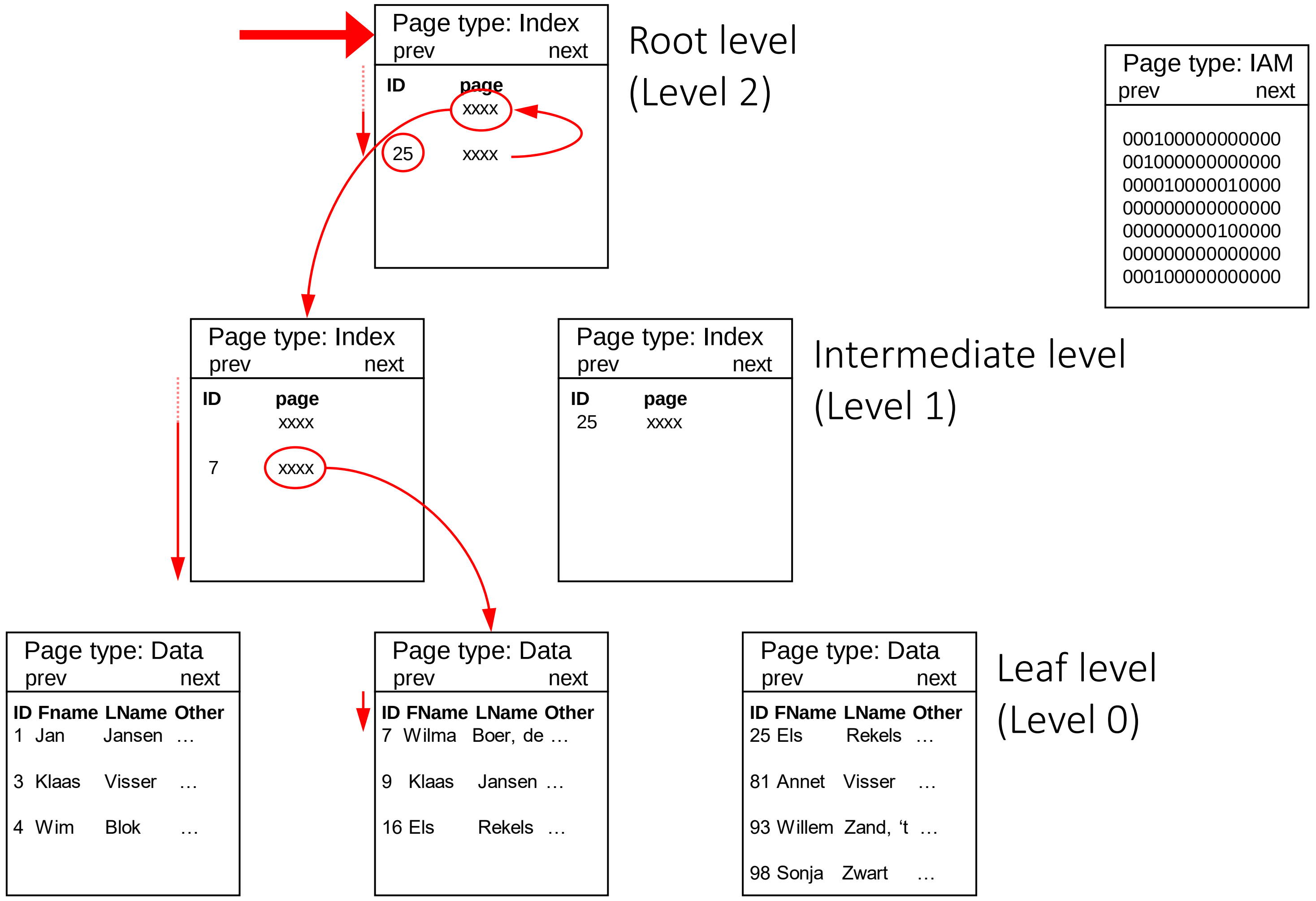


Seek operators

Clustered Index Seek

Singleton lookup

```
SELECT FirstName,
       LastName,
       Other
FROM   dbo.PersonsCluster
WHERE  PersonID = 7;
```



Seek operators

- Possible *Seek Predicates* depend on index definition
  - Equality on one or more **leading** column(s)
  - Range on *just one* **leading** column
  - Equality on one or more leading column(s), ***plus*** range on next column

Index tuning

- Possible *Seek Predicates* depend on index definition
- Adding extra indexes – yes or no?
  - Would it benefit where it matters?
    - How often is the index used?
    - Is it used in a critical process?

Index tuning

- Possible *Seek Predicates* depend on index definition
- Adding extra indexes – yes or no?
  - Would it benefit where it matters?
  - Can the query be rewritten?
    - Non-sargable filters?
    - Filters missing?
    - Too many rows returned?

Index tuning

- Possible *Seek Predicates* depend on index definition
- Adding extra indexes – yes or no?
  - Would it benefit where it matters?
  - Can the query be rewritten?
  - Is this the correct table (in the query) to focus on?
    - Is more work wasted on other tables?
    - Are the operators in the execution plan appropriate for this query and this data?

Index tuning

- Possible *Seek Predicates* depend on index definition
- Adding extra indexes – yes or no?
  - Would it benefit where it matters?
  - Can the query be rewritten?
  - Is this the correct table (in the query) to focus on?
  - Can an existing index be modified to benefit this query as well?
    - Adding extra columns (indexed / included)
    - Changing order of indexed columns
    - Changes might affect other queries!

Index tuning

- Possible *Seek Predicates* depend on index definition
- Adding extra indexes – yes or no?
- Fine tuning the new index
  - Order of indexed columns
    - For this query, but also for other queries
    - Affects statistics, which can change cardinality estimations

Index tuning

- Possible *Seek Predicates* depend on index definition
- Adding extra indexes – yes or no?
- Fine tuning the new index
  - Order of indexed columns
  - Additional indexed columns?
  - INCLUDEd columns needed?
  - Filtered index?
  - Can other queries benefit from the index too?

Index tuning

- Possible *Seek Predicates* depend on index definition
- Adding extra indexes – yes or no?
- Fine tuning the new index
- Goal:
  - Minimal set of indexes
  - Maximum benefit for overall performance
  - As little overhead as possible

Summary

- Seek operators
  - Index Seek / Clustered Index Seek
    - Singleton lookup
      - Unique index; equality on all columns
    - Range seek
      - All other cases
  - Seek Predicates
    - Defines singleton value or range to find
    - Multiple values or ranges can be combined

Next chapters

- Chapter 4: Lookup operators
  - Key Lookup
  - RID Lookup
- Chapter 5: Special scans