SQL SERVER Performance Tuning Notes

http://msbiskills.com/

What controls an Index Scan to read all pages from leaf level? Or An Index scan operator always reads all pages from leaf level?

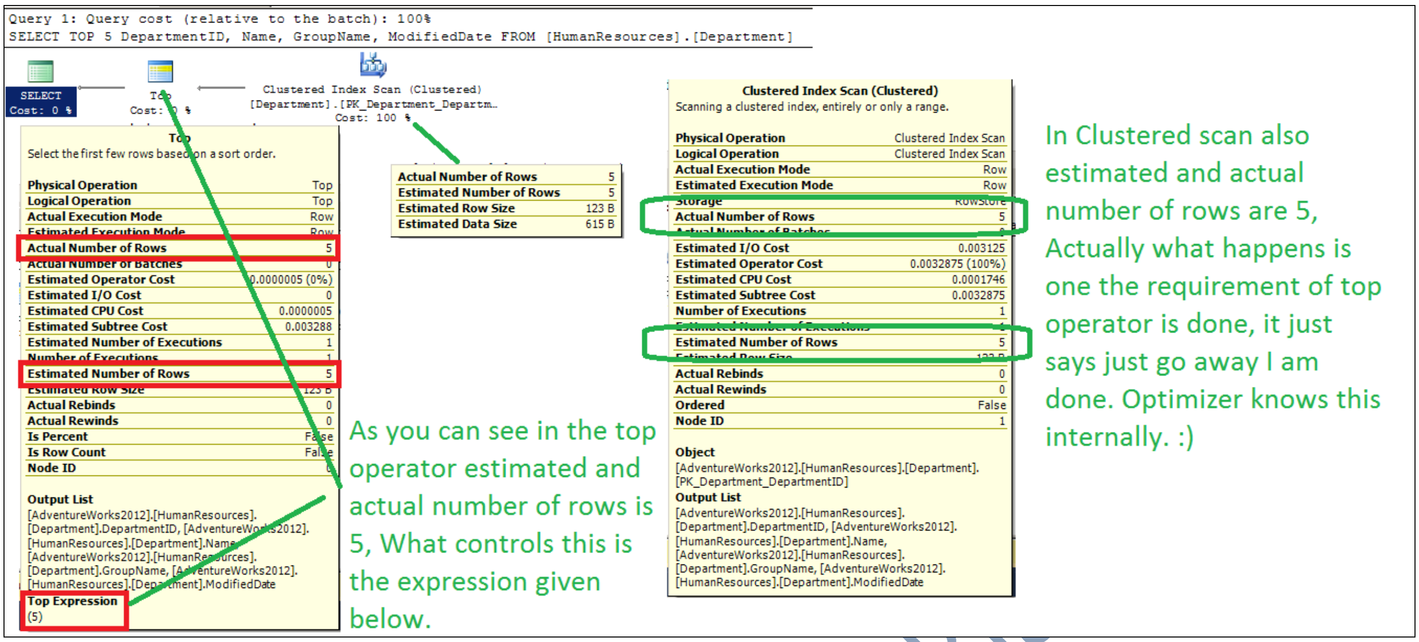
In continuation in the performance series, let's check whether an index scan operator always reads all the pages from the leaf level or not. Well, not always. It depends on the query you have written. If you are using TOP, MAX or Min in your query then your query will not read all the pages. These operators control the index scan operator and once their requirement is fulfilled, then they just says that the index scan just go away I am done. This is like this because your logic flows from left to right and your data flow from right to left. Let's go through some of the examples and try to understand what's going on behind the scenes.

Let's check out a query with TOP Operator

```
USE AdventureWorks2012
GO

SELECT TOP 5 DepartmentID, Name, GroupName, ModifiedDate
     FROM
          [HumanResources].[Department]
```

Now add the actual execution plan and execute the query. Below is what we got. By the ways in this table ([HumanResources].[Department]) we have 16 rows.



```
Query 1: Query cost (relative to the batch): 100%
SELECT TOP 5 DepartmentID, Name, GroupName, ModifiedDate FROM [HumanResources].[Department]
```

SELECT
Cost: 0 %
Top
Cost: 0 %
Clustered Index Scan (Clustered)
[Department].[PK_Department_Departm…
Cost: 100 %

Top
Select the first few rows based on a sort order.

Physical Operation	Top
Logical Operation	Top
Actual Execution Mode	Row
Estimated Execution Mode	Row
Actual Number of Rows	5
Actual Number of Batches	0
Estimated Operator Cost	0.0000005 (0%)
Estimated I/O Cost	0
Estimated CPU Cost	0.0000005
Estimated Subtree Cost	0.003288
Estimated Number of Executions	1
Number of Executions	1
Estimated Number of Rows	5
Estimated Row Size	123 B
Actual Rebinds	0
Actual Rewinds	0
Is Percent	False
Is Row Count	False
Node ID	0

Output List
[AdventureWorks2012].[HumanResources].[Department].DepartmentID, [AdventureWorks2012].[HumanResources].[Department].Name, [AdventureWorks2012].[HumanResources].[Department].GroupName, [AdventureWorks2012].[HumanResources].[Department].ModifiedDate
Top Expression
(5)

Actual Number of Rows	5
Estimated Number of Rows	5
Estimated Row Size	123 B
Estimated Data Size	615 B

As you can see in the top operator estimated and actual number of rows is 5, What controls this is the expression given below.

Clustered Index Scan (Clustered)
Scanning a clustered index, entirely or only a range.

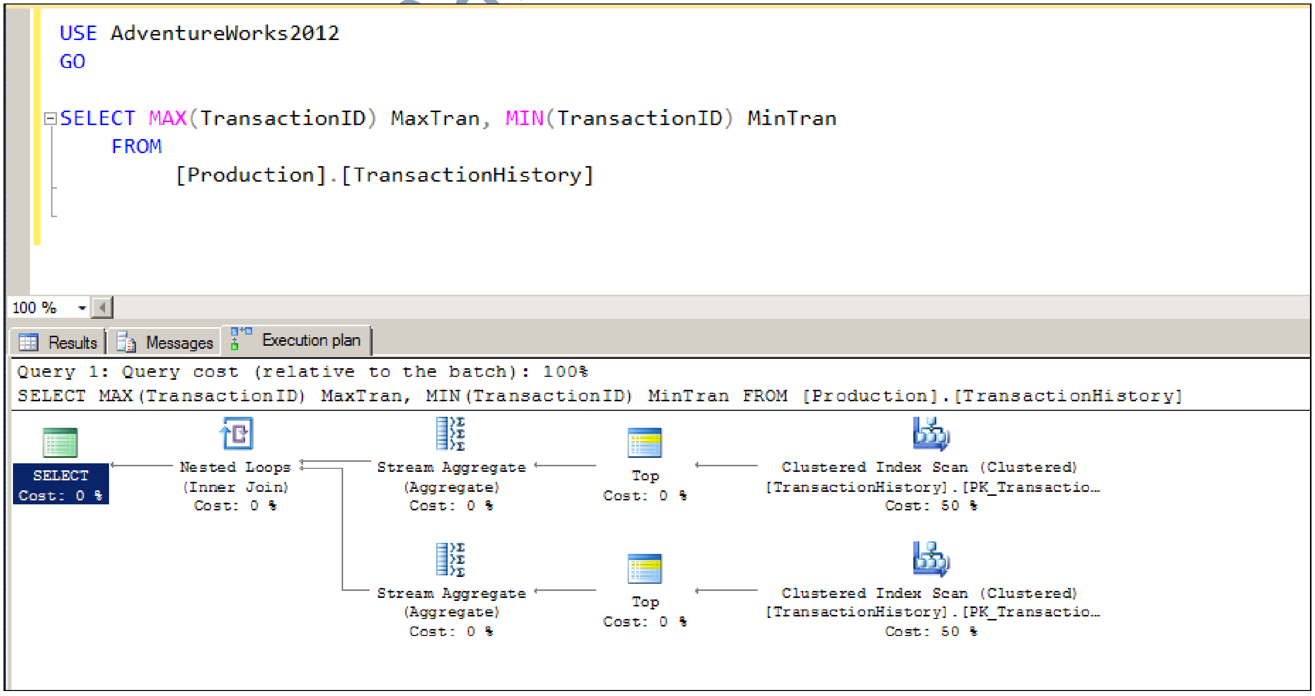
Physical Operation	Clustered Index Scan
Logical Operation	Clustered Index Scan
Actual Execution Mode	Row
Estimated Execution Mode	Row
Storage	RowStore
Actual Number of Rows	5
Actual Number of Batches	0
Estimated I/O Cost	0.003125
Estimated Operator Cost	0.0032875 (100%)
Estimated CPU Cost	0.0001746
Estimated Subtree Cost	0.0032875
Number of Executions	1
Estimated Number of Executions	1
Estimated Number of Rows	5
Estimated Row Size	123 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	False
Node ID	1

Object
[AdventureWorks2012].[HumanResources].[Department].[PK_Department_DepartmentID]
Output List
[AdventureWorks2012].[HumanResources].[Department].DepartmentID, [AdventureWorks2012].[HumanResources].[Department].Name, [AdventureWorks2012].[HumanResources].[Department].GroupName, [AdventureWorks2012].[HumanResources].[Department].ModifiedDate

In Clustered scan also estimated and actual number of rows are 5, Actually what happens is one the requirement of top operator is done, it just says just go away I am done. Optimizer knows this internally. :)

Now what happens internally here Select operator asks do you have any row for me, Top says that I don't have let me ask scan operator, Hey scan operator do you have any rows for me. Now the scan operator says yes I do have row for me. Now in our query top expression is 5, now when the top operator consumes 5 rows it says hey scan I don't need any more rows, just go away and breaks the execution and doesn't return any more rows to the select operator. Now this means our query execution is done and we got 5 rows as output.

Okay one more example let's check out a query with MIN & MAX Operator-



```
USE AdventureWorks2012
GO

SELECT MAX(TransactionID) MaxTran, MIN(TransactionID) MinTran
    FROM
        [Production].[TransactionHistory]
```

```
Query 1: Query cost (relative to the batch): 100%
SELECT MAX(TransactionID) MaxTran, MIN(TransactionID) MinTran FROM [Production].[TransactionHistory]
```

This second query returns the minimum and the maximum of the column TransactionID. TransactionID column has a Clustered Key column on it. Let's examine the execution plan. Here we have clustered index scan 2 times only to retrieve the maximum and minimum transaction id from transaction history table.

The explanation given for the top query applies here also. Top operator here consumes first row from the forward clustered index scan in case of minimum transaction id and first rows from a backward clustered index scan for maximum value, After that we are using nested loop join to join these values and then a stream aggregate. The stream aggregate is used to group rows by one or more columns and used to calculate aggregation expression.

Summary

Scan is not really always a complete scan in the execution plan. Operators like Top, Min and Max can restrict the full scan of the table.

That's all folks; I hope you've enjoyed learning this article, and I'll see you soon with more "Performance Tuning" articles.

Thanks!

Pawan Kumar Khowal

MSBISKills.com