SQL Server Cardinality - Quick Notes, Part 3

Hi Friends,

Read Part 1 & Part 2.

SQL Server Cardinality Estimation is a complicated topic until you understand the internals of SQL Server Statistics. Here are some quick notes that will answer some common questions that you generally have:

How does SQL Server Query Optimizer perform cardinality estimation in case of multiple columns?

Let’s see and example (turn ON Actual Execution Plan or press Ctrl + M)

In the below query I use a predicate/filer on Color column.

useAdventureWorks2012

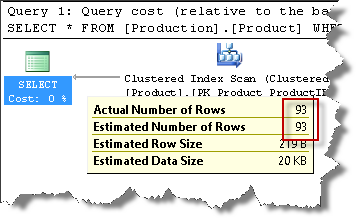
go

-- Turn On Actual Execution Plan

select\*fromProduction.Product

whereColor='Black'

go



Perfect Estimation of 93.

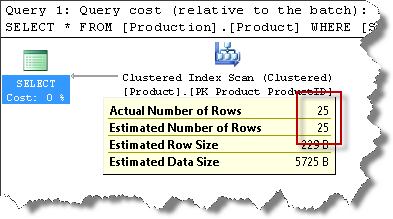
In another query I use a predicate/filer on SafetyStockLevel column.

-- Turn On Actual Execution Plan

select\*fromProduction.Product

whereSafetyStockLevel=800

go



Perfect Estimation of 25.

In both the above cases, SQL Optimizer uses Statistics. It automatically creates statistics for the columns, in case one does not exist. But note that SQL Server can only create single column statistics when it creates them automatically.

Now, I combine both of the predicates:

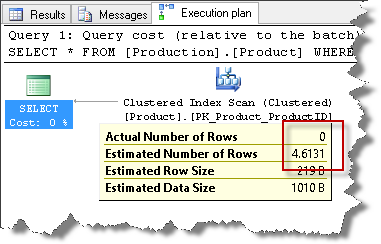
-- Turn On Actual Execution Plan

select\*fromProduction.Product

whereColor='Black'

ANDSafetyStockLevel=800

go



This time we get an estimation of 4.6131.

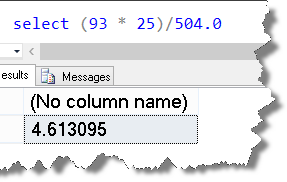
Why and where does this value come from?

SQL Server already has single column statistics on Color & SafteyStockLevel column. However, it does not have multi-column statistics to give us a good estimation in case we are combining multiple column filters in our predicate. But it uses the single column estimations to give us an approximate estimation as follows:

((Estimated Number of Rows for Color =50) \* (Estimated Number of Rows for SafteyStockLevel=800))/Total Number of Rows in the table

which is:

(93 \* 25)/504 = 4.6131



That’s how the default estimation was computed in this case. Hope the quick note was helpful.