EightKB
June 17th, 2020

Demystifying Database Statistics in SQL Server

Erin Stellato

Erin@SQLskills.com

SQLskills
immerse yourself in sql server

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

- Team of SQL Server consultants:
  - Kimberly Tripp (@KimberlyLTripp)
  - Jonathan Kehayias (@SQLPoolBoy)
  - Tim Radney (@Tim Radney)
  - Paul Randal (@PaulRandal)
  - Erin Stellato (@ErinStellato)
- Consulting: Health checks, design, performance, upgrades, Azure
  - www.sqlskills.com/services
- Instructor-led training: Immersion Events and onsite
  - www.sqlskills.com/training
- Online training through Pluralsight (www.pluralsight.com)
- Conferences: PASS Summit, SQLintersection, SQLBits
- Get our newsletter: https://www.sqlskills.com/Insider

Slides, Scripts, and Recordings

- Scripts for this session, along with a PDF of the slides, can be downloaded from:

https://github.com/eightkb/

- This session will be recorded, and will be available via the EightKB YouTube channel

- Please enter questions in the Q&A! (chat is not monitored)

Abstract

Troubleshooting performance issues and tuning queries can be easier with some level of understanding about statistics. Something you may hear people talk about as part of query tuning are "bad cardinality estimates". If you’ve wondered what that means, and you want to understand how the optimizer uses statistics when creating a query plan, then join me for this session. We’ll set the stage with a review of what statistics are, then see how the optimizer uses them and where to find that “cardinality estimate” information in a plan. We’ll also cover the various ways statistics can be updated, explore what happens to plans when those updates occur, and ultimately you will see why understanding how it all works is critical when you’re troubleshooting performance problems. Expect lots of demos so you can truly understand what statistics are, how they're used, and why their management is so essential for good performance.

Overview

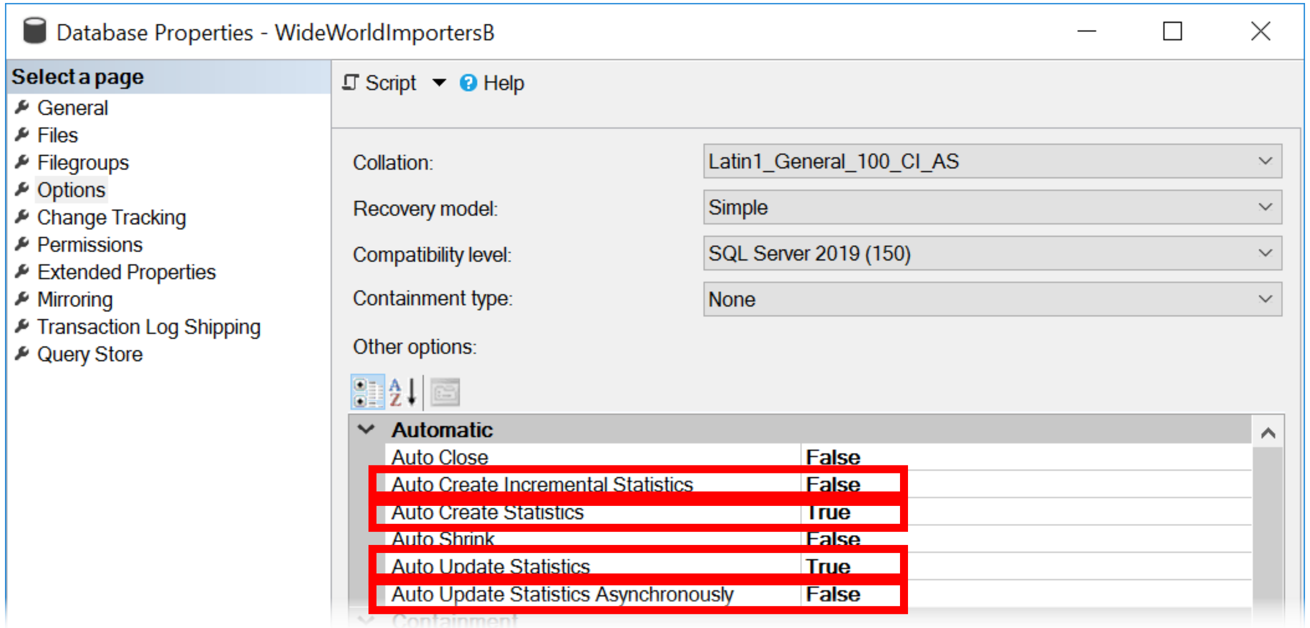
- Statistics review
- Viewing statistics
- When statistics go bad
- Updating statistics

Statistics: Why we care

- Information about the distribution of data in a column or set of columns in a table or indexed view
  - Stored as BLOBs
- The Query Optimizer uses statistics to estimate how many rows will be returned…this is *cardinality*
  - In turn, cardinality influences:
    - What index to use
    - Whether to seek or scan an index
    - How much memory is needed
- **Accurate** cardinality estimates help the optimizer generate a good plan

Statistics-related Database Options

- Auto Create Statistics
  - Enabled by default
- Auto Update Statistics
  - Enabled by default
- Auto Update Statistics Asynchronously
- Auto Create Incremental Statistics



Collation: Latin1_General_100_CI_AS

Recovery model: Simple

Compatibility level: SQL Server 2019 (150)

Containment type: None

Automatic	
Auto Close	False
Auto Create Incremental Statistics	False
Auto Create Statistics	True
Auto Shrink	False
Auto Update Statistics	True
Auto Update Statistics Asynchronously	False

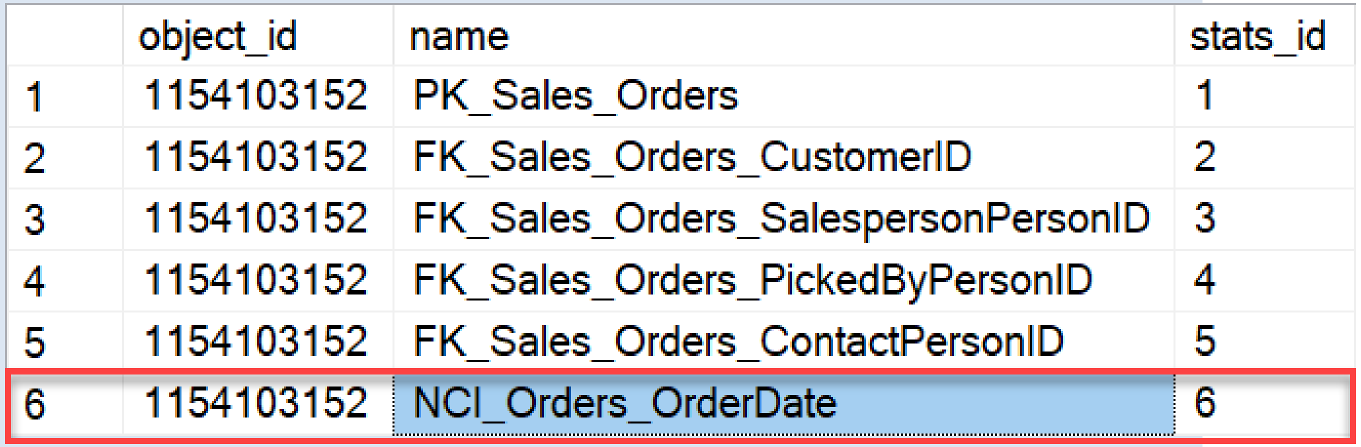
How are statistics created?

- Automatically
  - Every index, filtered index, indexed view (row store)

Create an index, get a statistic!

```
CREATE NONCLUSTERED INDEX NCI_Orders_OrderDate
      ON Sales.Orders (OrderDate);


SELECT *
FROM sys.stats
WHERE object_id =
      OBJECT_ID(N'Sales.Orders')
```



	object_id	name	stats_id
1	1154103152	PK_Sales_Orders	1
2	1154103152	FK_Sales_Orders_CustomerID	2
3	1154103152	FK_Sales_Orders_SalespersonPersonID	3
4	1154103152	FK_Sales_Orders_PickedByPersonID	4
5	1154103152	FK_Sales_Orders_ContactPersonID	5
6	1154103152	NCI_Orders_OrderDate	6

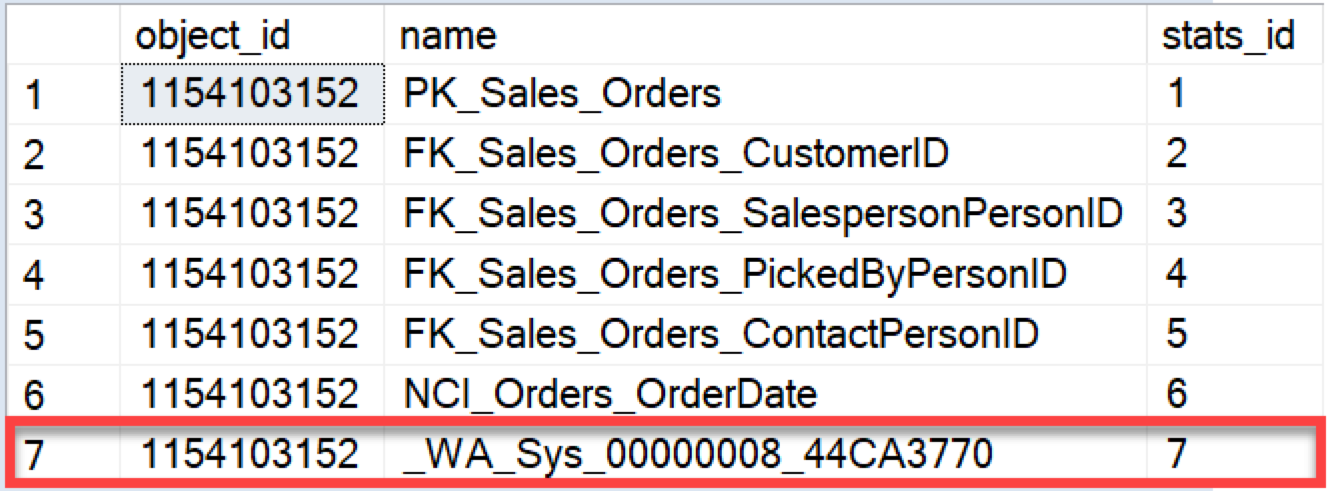
How are statistics created?

- Automatically
  - Every index, filtered index, indexed view (row store)
  - Individual columns

Query or join on a column, get a statistic*!

```
SELECT CustomerID, OrderDate
FROM Sales.Orders
WHERE ExpectedDeliveryDate = GETDATE();

SELECT *
FROM sys.stats
WHERE object_id =
      OBJECT_ID(N'Sales.Orders')
```

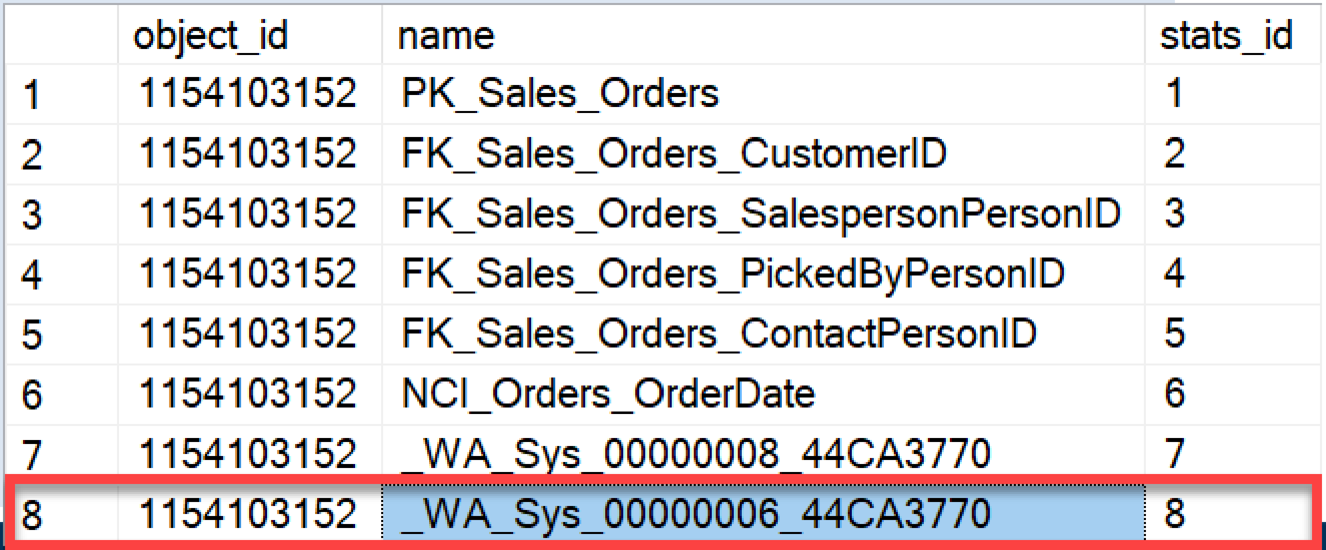


	object_id	name	stats_id
1	1154103152	PK_Sales_Orders	1
2	1154103152	FK_Sales_Orders_CustomerID	2
3	1154103152	FK_Sales_Orders_SalespersonPersonID	3
4	1154103152	FK_Sales_Orders_PickedByPersonID	4
5	1154103152	FK_Sales_Orders_ContactPersonID	5
6	1154103152	NCI_Orders_OrderDate	6
7	1154103152	_WA_Sys_00000008_44CA3770	7

Query or join on a column, get a statistic*!

```
SELECT b.CustomerID, b.OrderDate
FROM Sales.Orders b
JOIN Sales.Orders o
      ON b.BackorderOrderID = b.OrderID
WHERE b.ExpectedDeliveryDate = GETDATE();


SELECT *
FROM sys.stats
WHERE object_id =
      OBJECT_ID(N'Sales.Orders')
```



	object_id	name	stats_id
1	1154103152	PK_Sales_Orders	1
2	1154103152	FK_Sales_Orders_CustomerID	2
3	1154103152	FK_Sales_Orders_SalespersonPersonID	3
4	1154103152	FK_Sales_Orders_PickedByPersonID	4
5	1154103152	FK_Sales_Orders_ContactPersonID	5
6	1154103152	NCI_Orders_OrderDate	6
7	1154103152	_WA_Sys_00000008_44CA3770	7
8	1154103152	_WA_Sys_00000006_44CA3770	8

How are statistics created?

- Automatically
  - Every index, filtered index, indexed view (row store)
  - Individual columns
- Manually
  - Individual columns
  - Multiple columns
  - Filtered values in a column

Side note: Columnstore and In-Memory Tables

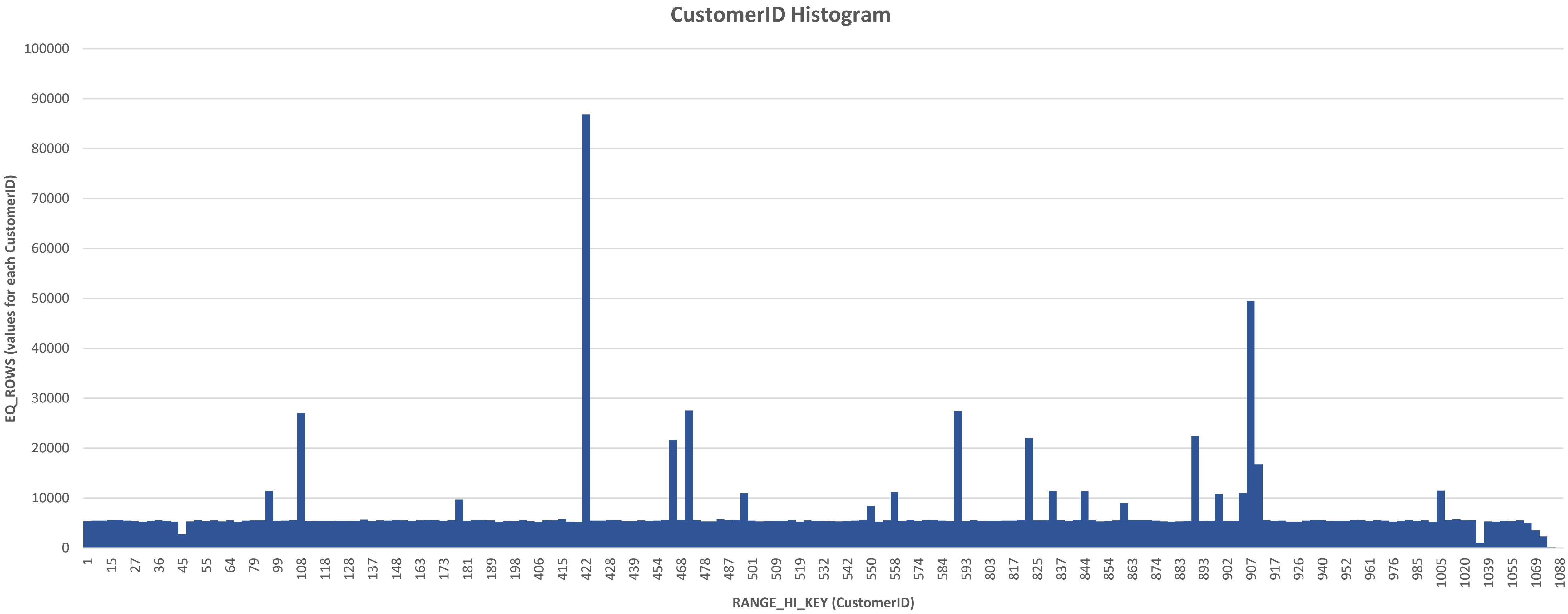
- Columnstore indexes have information about row groups and segments, but do not have the same type of detail as a row-store statistic
- In-Memory tables will have index and column stats
  - Filtered statistics are not supported
  - Automatic update is supported in SQL Server 2016+ and Azure SQL with compatibility mode 130
    - The execution plans for natively compiled stored procedures are based on the statistics available during initial compilation; manually recompile using sp_recompile

Viewing Statistics

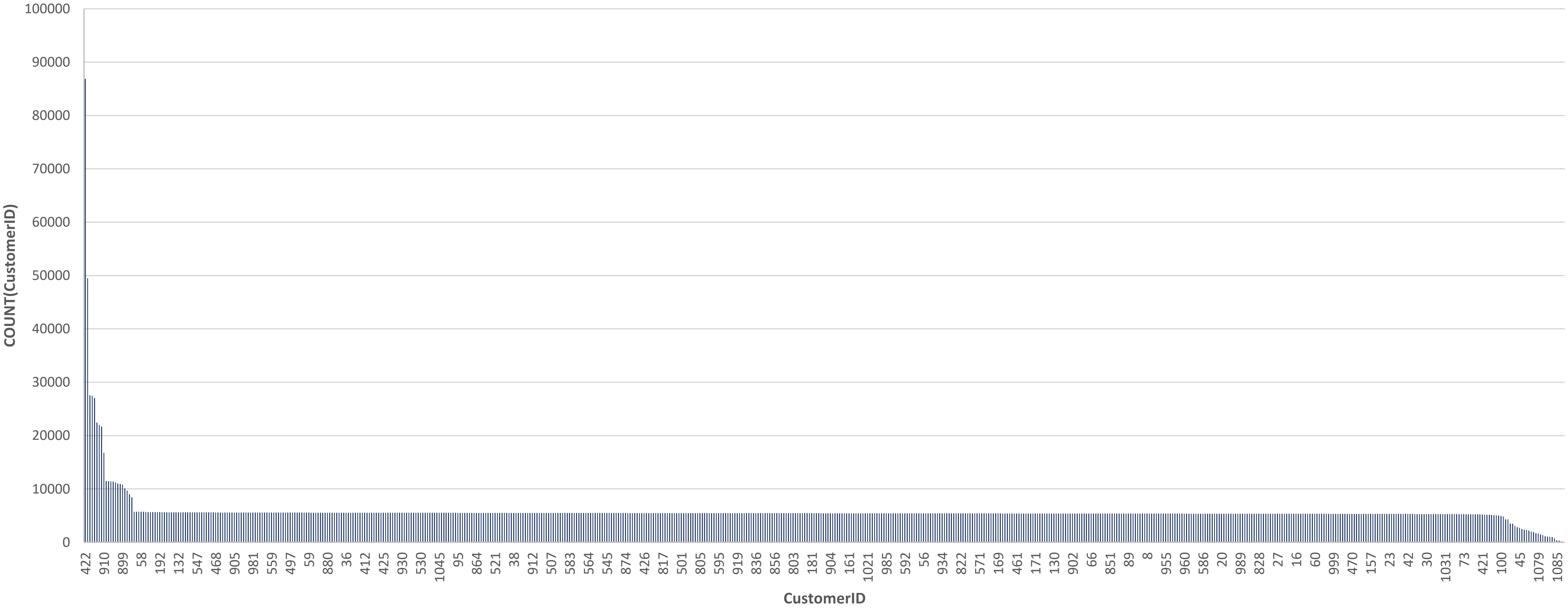
- Use sys.stats to view all statistics for a table
  - The STATS_DATE function returns when a statistic was updated
- DBCC SHOW_STATISTICS
  - Header
  - Density Vector
  - Histogram
- sys.dm_db_stats_properties
- sys.dm_db_stats_histogram

Demo: Viewing and Reading Statistics

Another way to think of a histogram



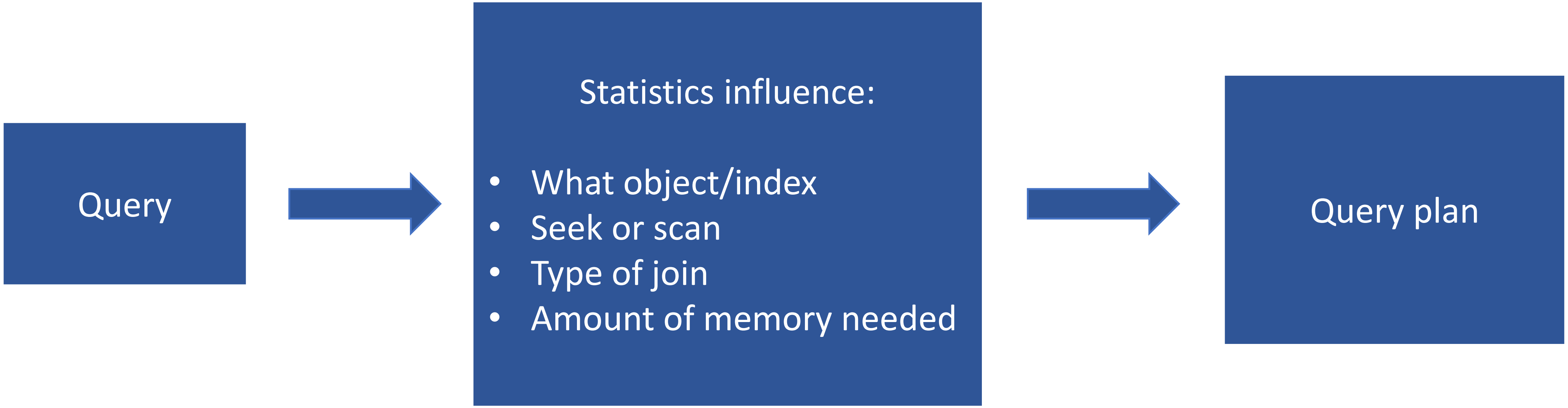
CustomerID ordered by COUNT



SELECT CustomerID, Count(CustomerID) FROM Sales.Orders GROUP BY CustomerID ORDER BY COUNT(CustomerID) DESC

How does the Query Optimizer use statistics?

- The optimizer uses statistics to estimate cardinality (the number of rows), and then choose the most efficient plan for retrieving or updating data



Query

Statistics influence:

- What object/index
- Seek or scan
- Type of join
- Amount of memory needed

Query plan

Demo: When stats go bad…

Automatic updates

- Occur if AUTO_UPDATE_STATISTICS is enabled
- When the number of modifications exceeds a threshold

Old threshold	20% + 500 rows changed
New threshold	SQRT(1000*Number of rows)

- Default behavior in SQL Server 2016+ with compatibility mode 130 (or higher)
- Can be enabled on earlier versions (or with older compatibility modes) using TF 2371 (down to SQL Server 2008 R2 SP1)

Demo: Updating and plans

Updates to Statistics and Plan Invalidation

- SQL Server 2012+
  - When statistics are updated, if NO data has changed, plans will NOT invalidate
- Before SQL Server 2012…
  - If you updated statistics with AUTO_UPDATE_STATS *enabled*, plans were invalidated
  - If you updated stats with AUTO_UPDATE_STATS *disabled*, plans were NOT invalidated

Updating statistics

- Automatically if auto update statistics option enabled
  - Can be asynchronous
  - New wait type in SQL Server 2019 when waiting on *synchronous* updates: WAIT_ON_SYNC_STATISTICS_REFRESH

Threshold Comparison

Rows	Old Threshold	New Threshold
1,000,000	200,500	31,623
5,000,000	1,000,500	70,711
10,000,000	2,000,500	100,000
50,000,000	10,000,500	223,607
100,000,000	20,000,500	316,228
1,000,000,000	200,000,500	1,000,000

Updating statistics

- Automatically if auto update statistics option enabled
  - Can be asynchronous
  - New wait type in SQL Server 2019 when waiting on *synchronous* updates: WAIT_ON_SYNC_STATISTICS_REFRESH
- Manually
  - Should be part of regular maintenance
- Useful options
  - NORECOMPUTE
  - PERSIST_SAMPLE_PERCENT (SQL 2016 SP1 CU4, SQL 2017 CU1)

Estimates in plans are not always “correct”

- Statistics haven’t been updated recently
- Uneven distribution of data
- Sample rate is too low
- Ascending keys
- Table variables
- Local variables/modifying in-flight
- Linked servers
- Partitioned tables
- In-Memory OLTP

Overview

- Statistics review
- Viewing statistics
- When statistics go bad
- Updating statistics

Resources

- Blog Posts
  - UPDATEs to Statistics
    - https://sqlperformance.com/2017/10/sql-statistics/updates-to-statistics
  - What caused that plan to go horribly wrong – should you update statistics?
    - https://www.sqlskills.com/blogs/kimberly/what-caused-that-plan-to-go-horribly-wrong-should-you-update-statistics/

Resources

- References
  - Statistical maintenance functionality (AutoStats) in SQL Server
    - https://support.microsoft.com/en-us/help/195565/statistical-maintenance-functionality-autostats-in-sql-server
  - Controlling Autostat (AUTO_UPDATE_STATISTICS) behavior in SQL Server
    - https://support.microsoft.com/en-us/help/2754171/controlling-autostat-auto-update-statistics-behavior-in-sql-server

Session and Event Feedback + Scripts

- Please visit the following page to provide feedback:

  https://eightkb.online/feedback/

- Scripts for this session, along with a PDF of the slides, can be downloaded from:

  https://github.com/eightkb/

Thank you!

Erin@SQLskills.com

@erinstellato

www.sqlskills.com/blogs/erin

SQLskills

immerse yourself in sql server