#### 1.2. How to read an execution plan.

There are two types of execution plans: estimated and actual. In the 99 per cent time estimated is the same as actual will be.

Plans should be read from right to left and top to bottom.

The size of arrows is indicating the amount of data flowing to the next operator.

SET STATISTICS IO, TIME ON --> is gonna give us more information about io operations, it will breaks out to logical reads from memory and logical reads from disk. TIME ON will give us the actual execution time and the cpu time.

#### 1.4. Key lookup: when it is good and when it is bad

A key lookup is a lookup against the clustered index. The lookup will execute once for each row, which is a very expensive operation.

INCLUDE columns are not key columns in the index, so they are not ordered. This means it isn't really useful for predicates, sorting etc as I mentioned above. However, it may be useful if you have a residual lookup in a few rows from the key column(s)

#### 1.5. Nested loop vs hash join

SQL Server employs three types of joins to bring together data: **Merge**, **hash** and **nested loop** joins:

* Hash joins are useful for large, unsorted, and non-indexed data
* Merge joins are used for large but sorted and indexed data
* Nested loop joins are good for small sets of data

You can hint your join type in query, sometime optimizer is making wrong decision

SQL Server 2017 introduces deferred join selection within plans that execute in batch execution mode. This deferred join selection will switch from nested loops to hash and vice versa, based on statistics generated as the query executes.

#### 1.6. Query Store

Introduced in SQL Server 2016. Personal favorite feature for author of this course. It is not enabled by default.

#### 1.8. Missing index warnings

SELECT s.last\_user\_seek ,

d.object\_id ,

d.equality\_columns ,

d.inequality\_columns ,

d.included\_columns ,

d.statement ,

s.avg\_user\_impact

FROM sys.dm\_db\_missing\_index\_group\_stats AS s

INNER JOIN sys.dm\_db\_missing\_index\_groups AS g ON ( s.group\_handle = g.index\_group\_handle )

INNER JOIN sys.dm\_db\_missing\_index\_details AS d ON ( g.index\_handle = d.index\_handle );