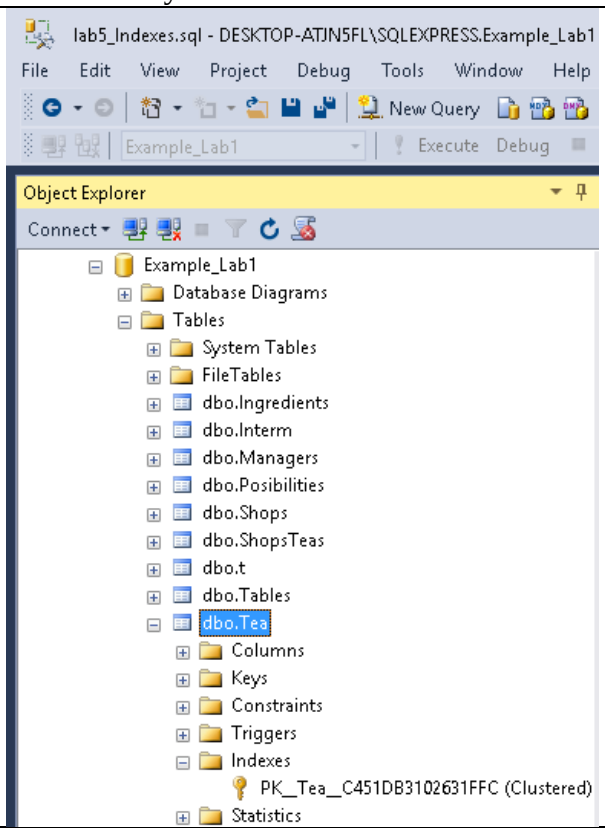
Indexes – example

```
use Example_Lab1
go

create table Tea(
Tid int primary key identity,
TName varchar(50),
Price int)

insert into Tea values ('Mint', 10),
('Ginger', 12), ('Fruits', 9), ('Rose', 8)

select * from Tea
```

Results | Messages

	Tid	TName	Price
1	1	Mint	10
2	2	Ginger	12
3	3	Fruits	9
4	4	Rose	8

Automatically a clustered index is created on the primary key (when this one is created). On a table one can have only one clustered index.

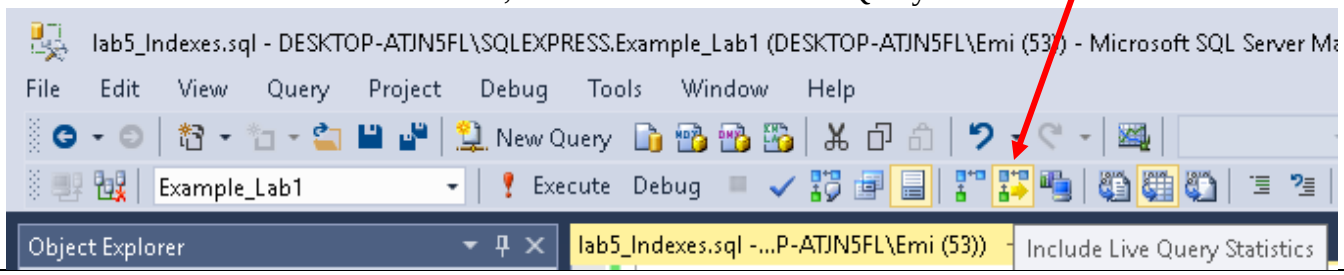



```
-- 1 clustered index was created on the primary key
-- PK__Tea__C451DB3102631FFC

select * from Tea
order by Tid
```

Results | Messages

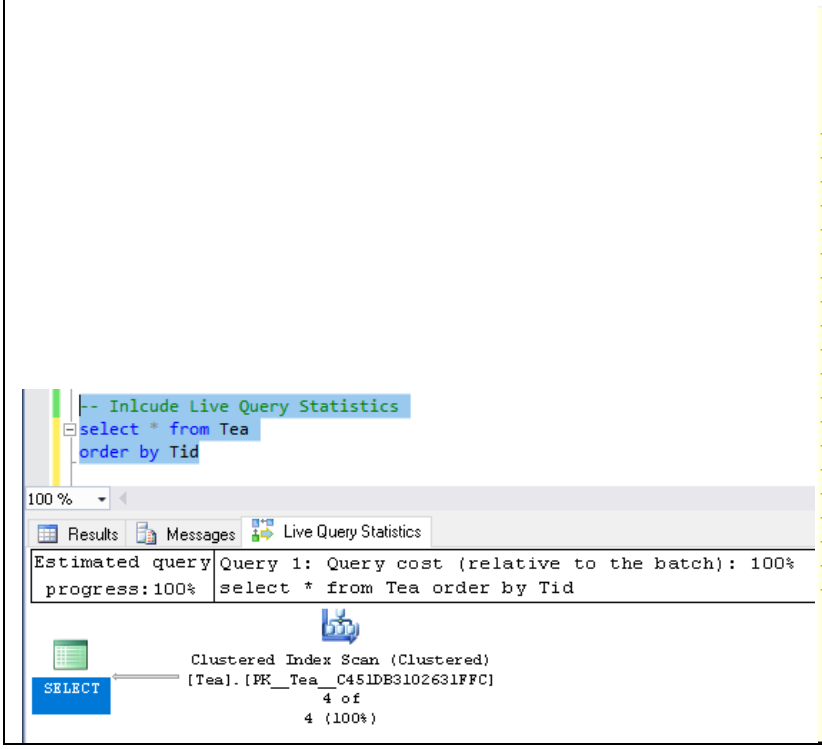
	Tid	TName	Price
1	1	Mint	10
2	2	Ginger	12
3	3	Fruits	9
4	4	Rose	8

To check the indexes and how are used, one can use Include Live Query Statistics.




```
-- Inlcude Live Query Statistics
```

```
select * from Tea
order by Tid
```



Clustered Index Scan (Clustered)

Scanning a clustered index, entirely or only a range.

Estimated operator progress: 100%

Physical Operation	Clustered Index Scan
Logical Operation	Clustered Index Scan
Actual Execution Mode	Row
Estimated Execution Mode	Row
Storage	RowStore
Number of Rows Read	4
Actual Number of Rows	4
Actual Number of Batches	0
Estimated I/O Cost	0.003125
Estimated Operator Cost	0.0032864 (100%)
Estimated Subtree Cost	0.0032864
Estimated CPU Cost	0.0001614
Estimated Number of Executions	1
Number of Executions	1
Estimated Number of Rows	4
Estimated Row Size	44 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	True
Node ID	0

Object

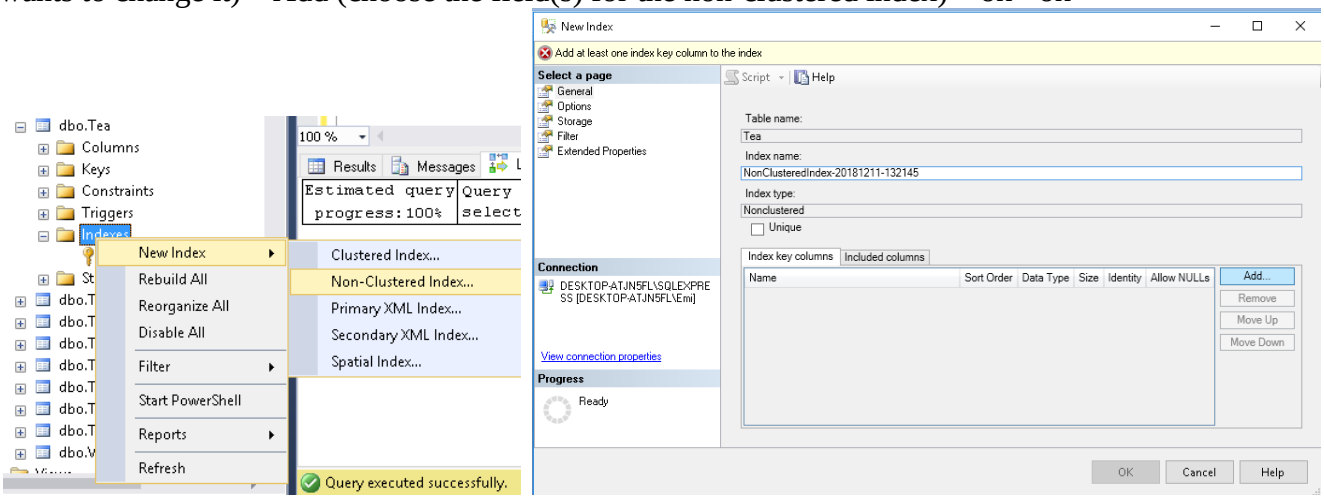
[Example_Lab1].[dbo].[Tea].[PK__Tea__C451DB3102631FFC]

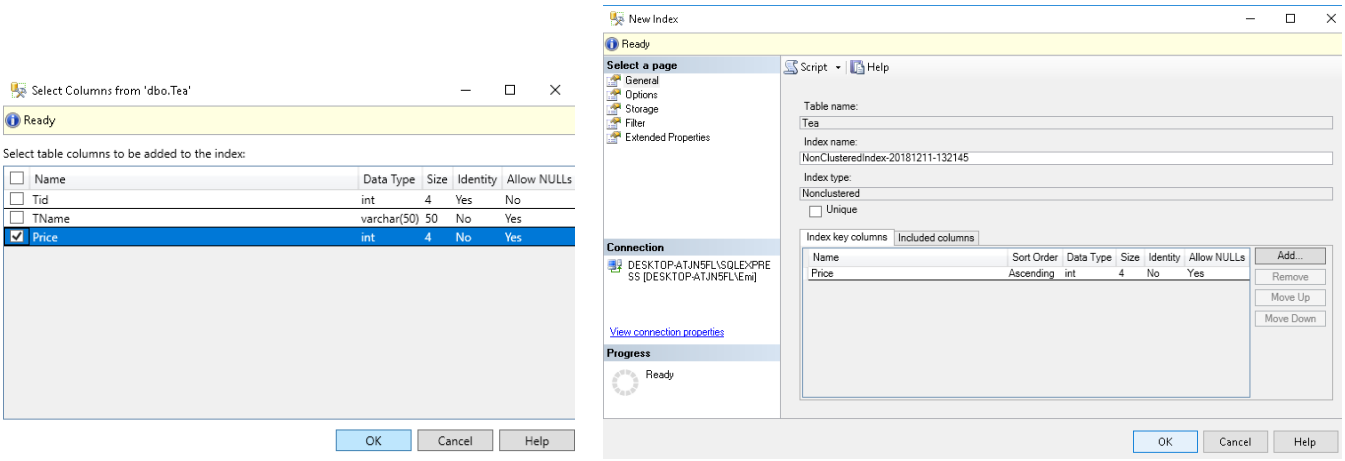
Output List

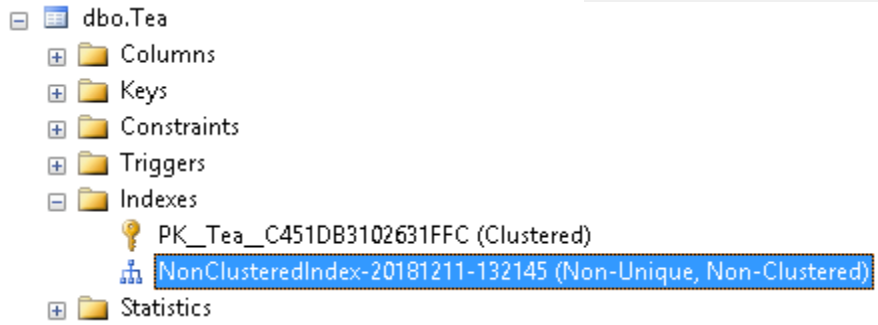
[Example_Lab1].[dbo].[Tea].Tid, [Example_Lab1].[dbo].[Tea].TName, [Example_Lab1].[dbo].[Tea].Price

<u>Create Non-Clustered Indexes by Design View</u>

- in the table tabs – choose Indexes – right click – new Index – Non-Clustered Index – Name (if one wants to change it) – Add (choose the field(s) for the non-clustered index) – ok - ok








```
-- create non-clustered index - by Design View
-- NonClusteredIndex-20181211-132145
```

Create Non-Clustered Indexes by Code

```
-- all the indexes from the
current database
select * from sys.indexes
```

Results | Messages | Live Query Statistics

	object_id	name	index_id	type	type_desc	is_unique	data_space_id	ignore_dup_key	is_primary_key	is_unique_constraint	fill_factor	is_padded
1	3	clst	1	1	CLUSTERED	1	1	0	0	0	0	0
2	5	clust	1	1	CLUSTERED	1	1	0	0	0	0	0
3	6	clst	1	1	CLUSTERED	1	1	0	0	0	0	0
4	7	clust	1	1	CLUSTERED	1	1	0	0	0	0	0
5	7	nc	2	2	NONCLUSTERED	1	1	0	0	0	0	0
6	8	NULL	0	0	HEAP	0	1	0	0	0	0	0
7	9	clst	1	1	CLUSTERED	1	1	0	0	0	0	0
8	17	cl	1	1	CLUSTERED	1	1	0	0	0	0	0
9	17	nc	2	2	NONCLUSTERED	1	1	0	0	0	0	0
10	17	nc2	3	2	NONCLUSTERED	1	1	0	0	0	0	0

Query executed successfully. | DESKTOP-ATJN5FL\SQLEXPRESS ... | DESKTOP-ATJN5FL\Emi (53) | Example_Lab1 | 00:00:03 | 176 rows

```
select name from sys.indexes
```

Results

	name
1	clst
2	clust
3	clst
4	clust
5	nc
6	NULL
7	clst
8	cl
9	nc
10	nc2
11	cl

```
select * from Tea
Order by TName
-- only the clustered index is used
```

Results | Messages | Live Query Statistics

Estimated query progress: 100%

Query 1: Query cost (relative to the batch): 100%
select * from Tea Order by TName

Clustered Index Scan (Clustered)

Scanning a clustered index, entirely or only a range.

Estimated operator progress: 100%

Physical Operation	Clustered Index Scan
Logical Operation	Clustered Index Scan
Actual Execution Mode	Row
Estimated Execution Mode	Row
Storage	RowStore
Number of Rows Read	4
Actual Number of Rows	4
Actual Number of Batches	0
Estimated I/O Cost	0.003125
Estimated Operator Cost	0.0032864 (22%)
Estimated CPU Cost	0.0001614
Estimated Subtree Cost	0.0032864
Estimated Number of Executions	1
Number of Executions	1
Estimated Number of Rows	4
Estimated Row Size	44 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	False
Node ID	1

Object

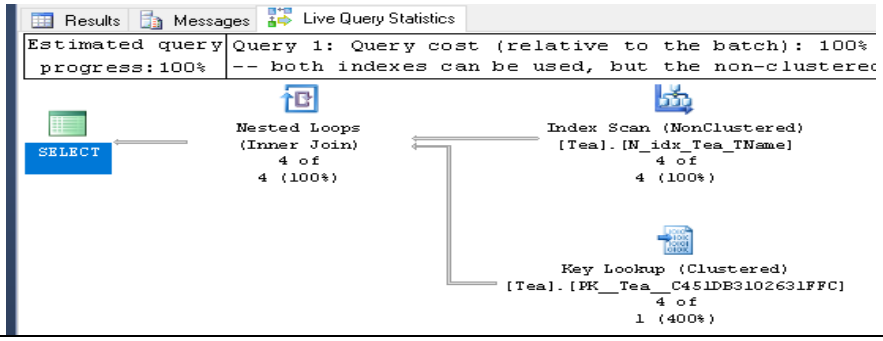
[Example_Lab1].[dbo].[Tea].[PK__Tea__C451DB3102631FFC]

Output List

[Example_Lab1].[dbo].[Tea].Tid, [Example_Lab1].[dbo].[Tea].TName, [Example_Lab1].[dbo].[Tea].Price

```
--create index non-clustered on the TName field
IF EXISTS (SELECT name FROM sys.indexes WHERE name = N'N_idx_Tea_TName')
   DROP INDEX N_idx_Tea_TName ON Tea;
GO

CREATE NONCLUSTERED INDEX N_idx_Tea_TName ON Tea(TName);
GO
```

```
-- both indexes can be used, but the non-clustered one is more efficient
select * from Tea
Order by TName
```



Index Scan (NonClustered)

Scan a nonclustered index, entirely or only a range.

Estimated operator progress: 100%

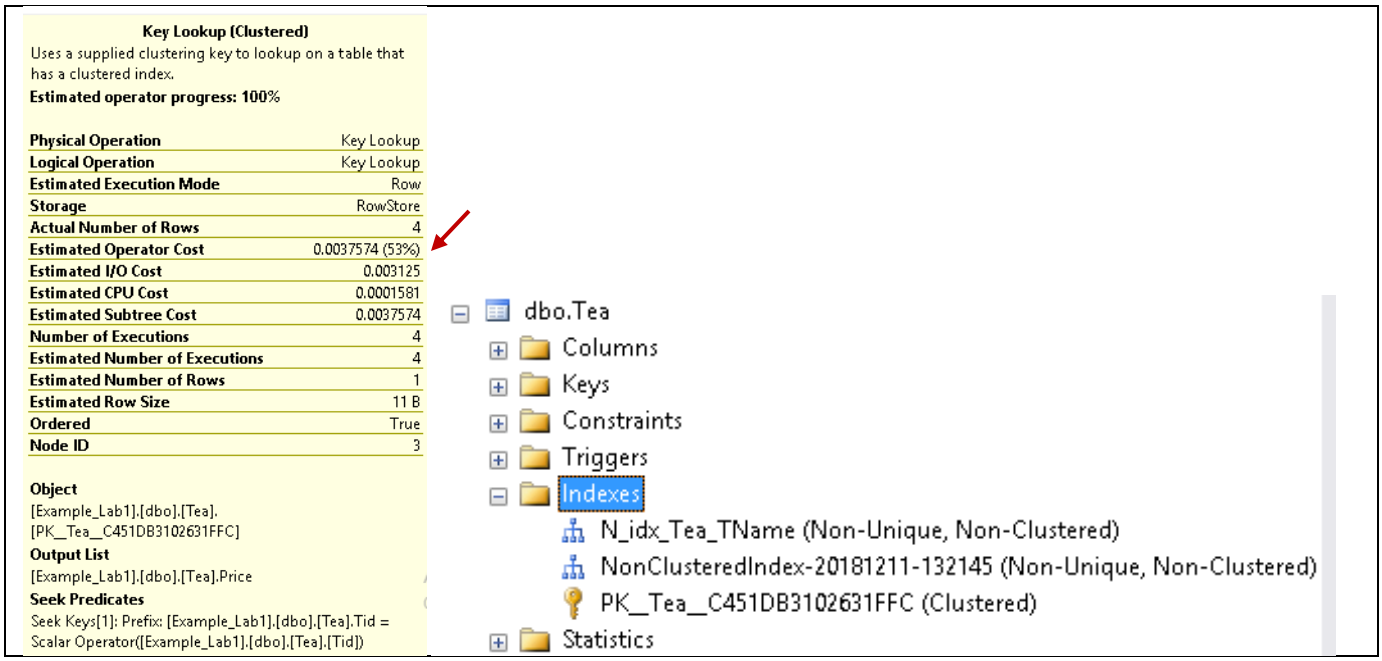
Physical Operation	Index Scan
Logical Operation	Index Scan
Estimated Execution Mode	Row
Storage	RowStore
Actual Number of Rows	4
Estimated Operator Cost	0.0032864 (47%)
Estimated I/O Cost	0.003125
Estimated CPU Cost	0.0001614
Estimated Subtree Cost	0.0032864
Number of Executions	1
Estimated Number of Executions	1
Estimated Number of Rows	4
Estimated Row Size	40 B
Ordered	True
Node ID	1

Object

[Example_Lab1].[dbo].[Tea].[N_idx_Tea_TName]

Output List

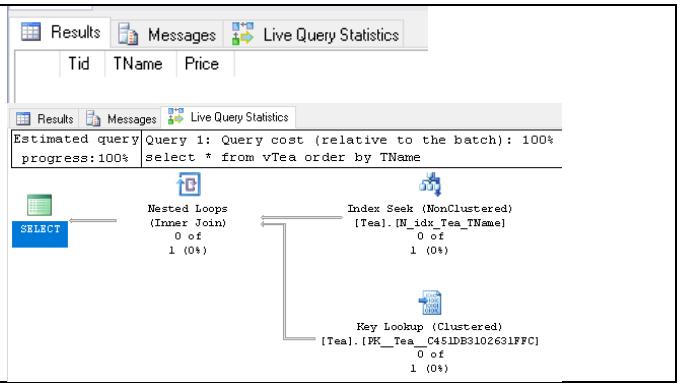
[Example_Lab1].[dbo].[Tea].Tid, [Example_Lab1].[dbo].[Tea].TName



The Non-clustered index should be created on the fields involved in ORDER BY clauses, WHERE clause, JOIN clauses, to increase the efficiency and decrease the execution time.

Create view

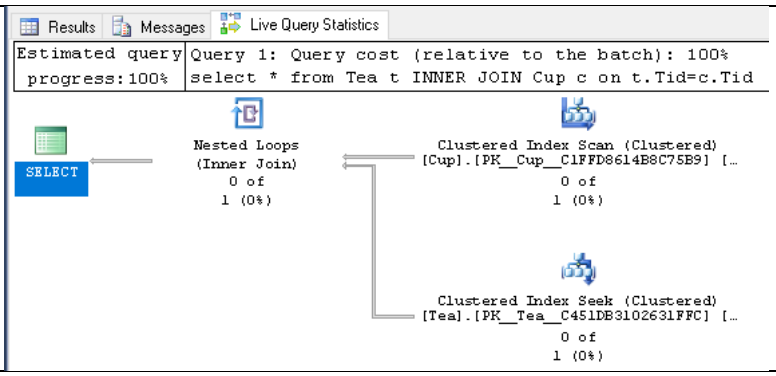
```
-- create view
create view vTea
as
  select * from Tea
  where TName LIKe 'a%'
go

-- execute
select * from vTea
order by TName
```



Example with JOIN's

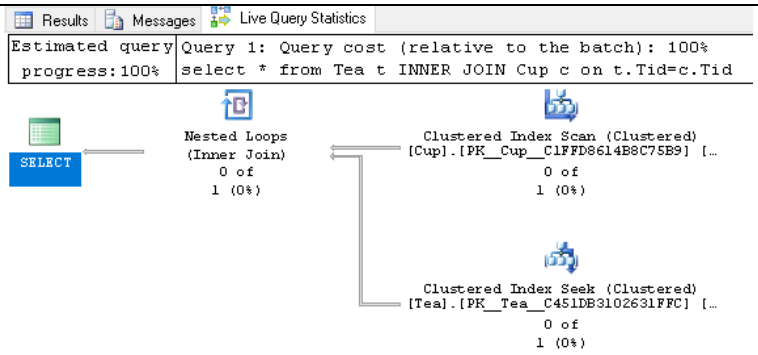
```
-- with Join's
create table Cup(
Cid int primary key identity,
CName varchar(50),
Color varchar(20),
Size varchar(20),
Tid int foreign key references Tea(Tid))
```

```sql
-- with include live query statistics
select *
from Tea t INNER JOIN Cup c on t.Tid=c.Tid
-- Clustered Index Scan + Clustered Index Seek
```

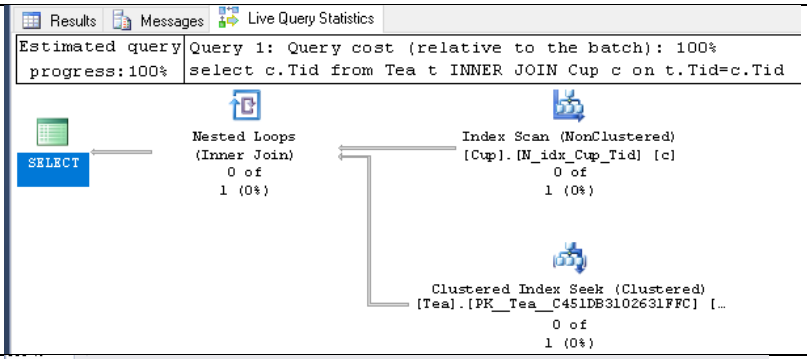



```sql
-- create nonclustered index on the foreign key :) it is very good
--create index non-clustered on the Tid field on the table Cup
IF EXISTS (SELECT name FROM sys.indexes WHERE name = N'N_idx_Cup_Tid')
   DROP INDEX N_idx_Cup_Tid ON Cup;
GO
CREATE NONCLUSTERED INDEX N_idx_Cup_Tid ON Cup(Tid);
GO
```

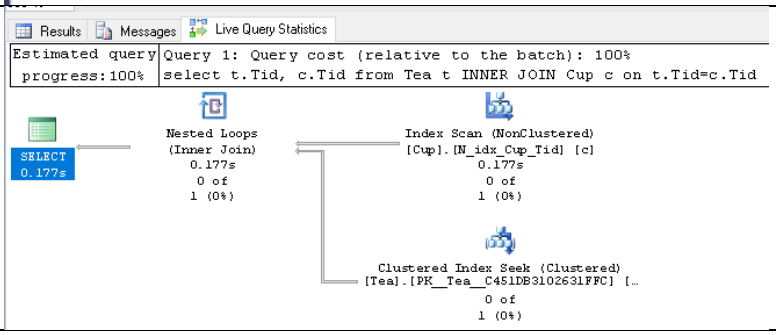
```sql
-- with include live query statistics
select *
from Tea t INNER JOIN Cup c on t.Tid=c.Tid
-- Clustered Index Scan + Clustered Index Seek
```



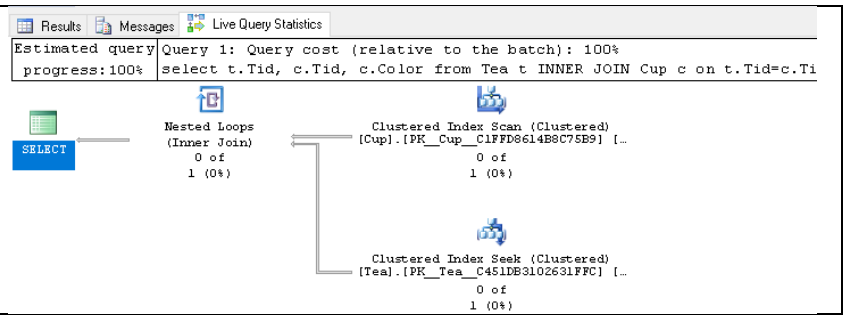

```sql
-- with include live query statistics
select  c.Tid
from Tea t INNER JOIN Cup c on t.Tid=c.Tid
-- Non-Clustered Index Scan + Clustered Index
Seek
```




```sql
select  t.Tid, c.Tid
from Tea t INNER JOIN Cup c on t.Tid=c.Tid
-- Non-Clustered Index Scan + Clustered Index
Seek
```



```
select  t.Tid, c.Tid, c.Color
from Tea t INNER JOIN Cup c on t.Tid=c.Tid
-- Clustered Index Scan + Clustered Index Seek
```



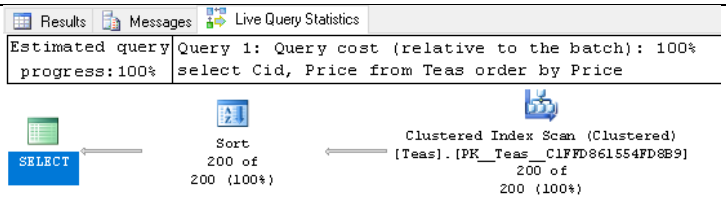
At WHERE and JOIN's is(are) very important the field(s) used in SELECT clause (this(these) field(s) decide which of the indexes are used).
At ORDER BY is (are) very important the field(s) used in ORDER BY (this(these) one(s) decide the index(es) that is(are) used).

Observation:

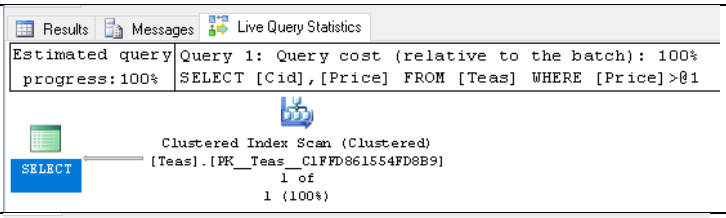
Creating and using a non-clustered index on a field that is not used in the Select, Where, Join's, Order by clauses, it is useless and should be avoided.

```
-- creating a non-clustered index on a field that will not be used, it is useless

-- we have a non-clustered index on Teas in the field TName, but we do not use the field TName in our queries
```

```
select Cid, Price from Teas order by Price
-- clustered index
```




```
select Cid, Price from Teas where Price>10
-- clustered index
```




```
select C.Cid, Price from Teas T inner join Cup C on T.Cid=C.Cid -- clustered index
-- the non-clustered index used is on the Cup.Cid field created
```