

Проектирование и построение индексов

На рисунке 1 изображен код таблицы, созданной для анализа работы с индексами.

```
select top(0) *
into CopyBusInformation
from BusInformation

declare @n int = 0;
declare @a char = 'A', @z char = 'a', @w int, @l int
SET @w = ascii(@z) - ascii(@a);
SET @l = ascii(@a);
declare @BusNumber nvarchar(6) = 'AAAAA';
declare @Brand nvarchar(20) = 'BA3';
declare @Number int;

while (@n < 100000)
begin
    set @Number = left(floor(rand() * 100000), 3);
    set @Number = cast(@Number as nvarchar);
    if (len(@Brand) > 19) set @Brand = 'BA3' else set @Brand = @Brand + char(round(rand() * @w, 0) + @l);
    set @BusNumber = upper(@BusNumber+char(round(rand() * @w, 0) + @l));
    set @BusNumber = stuff(@BusNumber, 2,3, @Number);
    insert into CopyBusInformation (Brand, BusNumber, PassengerCapacity, Mileage, FuelConsumption, VolumeOfTheTank)
    values (@Brand, @BusNumber, rand() * 100, rand() * 10000, rand() * 100, rand() * 1000)
    set @n = @n + 1
end

select * from CopyBusInformation
```

Рисунок 1 – Код рассматриваемой таблицы

На рисунке 2 изображена таблица. В ней данные заполняются случайным образом, она содержит в себе 100000 строк.

BusID	Brand	BusNumber	PassengerCapacity	Mileage	FuelConsumption	VolumeOfTheTank
57334	BA3цмЮ	A295шш	0	6852	13	741
57335	BA3цмЮш	A380шш	90	9990	89	203
57336	BA3цмЮш	A190шш	82	6990	33	26
57337	BA3цмЮшш	A963шш	67	2994	12	640
57338	BA3цмЮшшш	A815шш	70	2202	58	489
57339	BA3цмЮшшшш	A111шш	52	21	65	75
57340	BA3цмЮшшшшш	A996шш	20	6379	92	889
57341	BA3цмЮшшшшшш	A282шш	23	4523	34	767
57342	BA3цмЮшшшшшшш	A827шш	18	2352	40	847
57343	BA3цмЮшшшшшшшш	A004шш	2	1403	39	49
57344	BA3цмЮшшшшшшшшш	A363шш	4	1842	81	772
57345	BA3цмЮшшшшшшшшшш	A253шш	85	934	76	504
57346	BA3цмЮшшшшшшшшшшш	A395шш	45	422	55	556
57347	BA3цмЮшшшшшшшшшшшш	A779шш	66	4094	40	111
57348	BA3	A823шш	49	2814	31	874
57349	BA3у	A850шш	13	6691	85	567
57350	BA3ш	A683шш	30	1494	10	187

Рисунок 2 – Записи в рассматриваемой таблице

На рисунке 3 представлен запрос, в котором будет производится небольшой анализ данных. Для отображения сведений о взаимодействии SQL Server с диском во время выполнения запроса и процессорного времени выполняются следующие команды соответственно SET STATISTICS IO ON и SET STATISTICS TIME ON.



Рисунок 3 – Анализ запроса

Кластерный индекс

Перед созданием кластерного индекса необходимо оценить примерные затраты на выполнение данной операции и всех предшествующих операций в поддереве. Для анализа операции будет взят запрос, показанный на рисунке 3.

На рисунке 4 изображен план выполнения запроса к таблице, не имеющей индексов. Из рисунка можно заметить характеристику по которой будет проведен анализ. Речь идет о предполагаемой стоимости поддерева (estimated subtree cost (ESC))

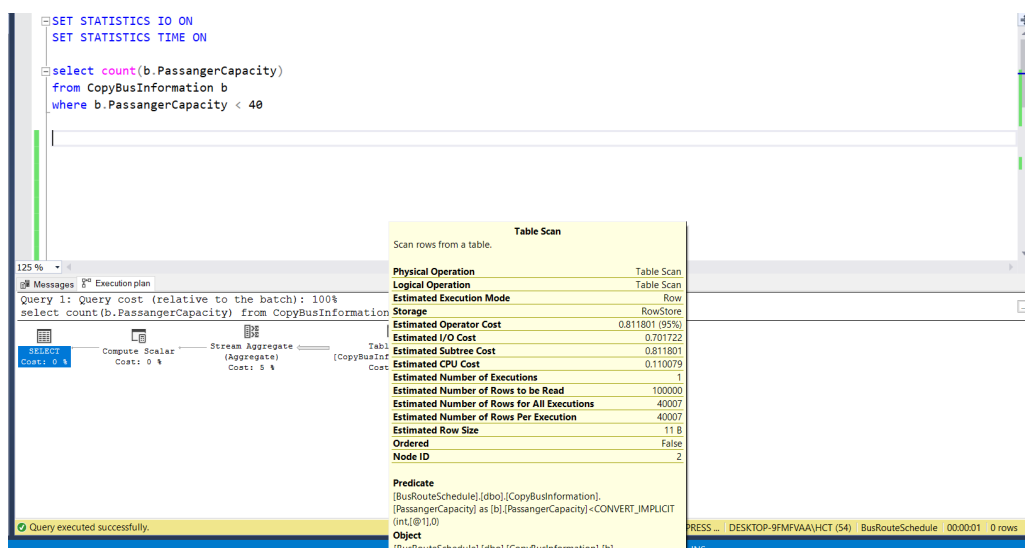


Рисунок 4 – План выполнения запроса к таблице, не имеющей индексов

Далее, на рисунке 5 изображены команда по созданию кластерного индекса и данные об ESC.

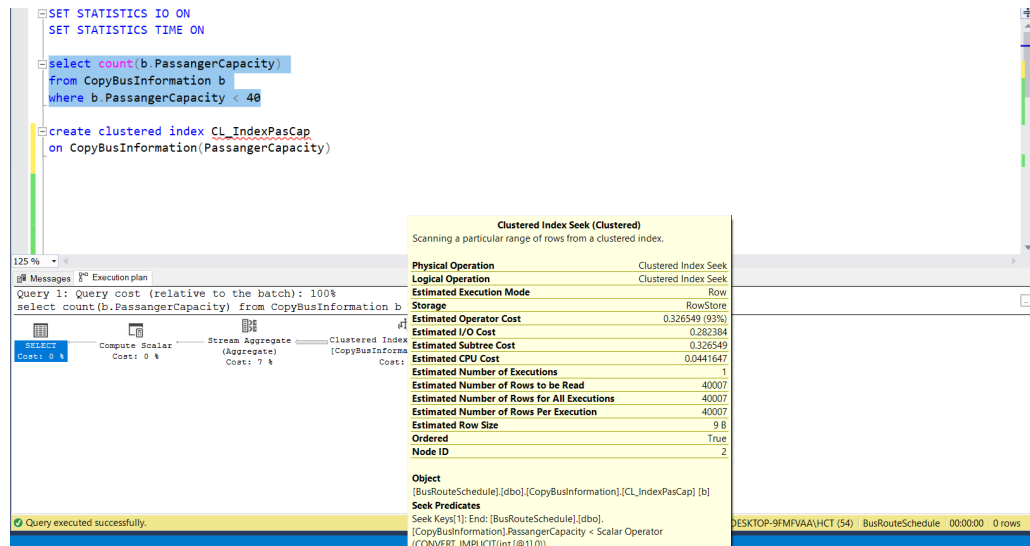


Рисунок 5 – План выполнения запроса после создания кластерного индекса

Из рисунка видно, что после создания кластерного индекса стоимость запроса значительно уменьшилось.

Некластерный индекс

На рисунке 6 создается некластерный индекс. Выполнив исходный запрос, можно заметить, что значение ESC отличается от предыдущего плана выполнения текущего запроса.

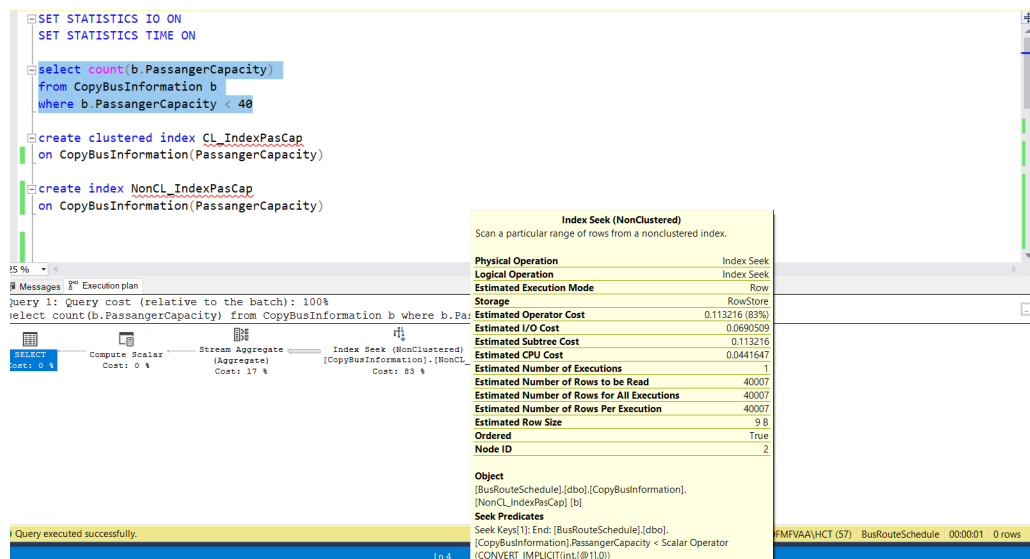


Рисунок 6 – План выполнения запроса после создания некластерного индекса

Известно, что существует несколько типов некластерного индекса, например на рисунке 6 был представлен одиночный индекс. Поэтому далее будут рассмотрены остальные его виды.

— Составной индекс

На рисунке 7 представлено выполнение нового запроса. Стоит отметить, кластерный индекс остался не изменным.

```

SET STATISTICS IO ON
SET STATISTICS TIME ON

select PassengerCapacity, VolumeOfTheTank
from CopyBusInformation
where PassengerCapacity = 30 and VolumeOfTheTank = 200

```

Clustered Index Seek (Clustered)
Scanning a particular range of rows from a clustered index.

Physical Operation	Clustered Index Seek
Logical Operation	Clustered Index Seek
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.0109414 (100%)
Estimated I/O Cost	0.0096426
Estimated Subtree Cost	0.0109414
Estimated CPU Cost	0.0012988
Estimated Number of Executions	1
Estimated Number of Rows Per Execution	9.39864
Estimated Number of Rows to be Read	1038
Estimated Number of Rows for All Executions	9.39864
Estimated Row Size	15 B
Ordered	True
Node ID	0

Predicate
[BusRouteSchedule].[dbo].[CopyBusInformation].[VolumeOfTheTank] = (200)

Object
[BusRouteSchedule].[dbo].[CopyBusInformation].[CL_IndexPasCap]

Output List
[BusRouteSchedule].[dbo].[CopyBusInformation].PassengerCapacity,
[BusRouteSchedule].[dbo].[CopyBusInformation].VolumeOfTheTank

Seek Predicates
Seek Keys[1]: Prefix: [BusRouteSchedule].[dbo].[CopyBusInformation].[VolumeOfTheTank] = 200, Scalar Operator[200]

Query 1: Query cost (relative to the select PassengerCapacity, VolumeOfTheTank from CopyBusInformation where PassengerCapacity = 30 and VolumeOfTheTank = 200) is 1.000000000. Estimated number of rows returned is 1038. (Impact 66.2074): CREATE INDEX [CL_IndexPasCap] ON [BusRouteSchedule].[dbo].[CopyBusInformation] ([PassengerCapacity], [VolumeOfTheTank]) WITH (INDEXID=1, SORT_IN_TEMPDB=OFF, STATISTICS=ON, FILLFACTOR=100, ALLOW_ROW_LOCKS=ON, ALLOW_PAGE_LOCKS=ON) [PRIMARY]

25% ▾

Messages Execution plan

SELECT [PassengerCapacity], [VolumeOfTheTank] FROM [CopyBusInformation] WHERE [PassengerCapacity] = 30 AND [VolumeOfTheTank] = 200

Clustered Index Seek (Clustered) (CopyBusInformation). [CL_IndexPasCap]

Cost: 100 %

ity = 30 and VolumeOfTheTank = 200

name,>] ON [dbo].[CopyBusInformation] ([PassengerCapacity],[VolumeOfTheTank])

Query executed successfully.

DESKTOP-9FMFVAA\SQLEXPRESS _

DESKTOP-9FMFVAA\HCT (57) | BusRouteSchedule | 00:00:00 | 0 rows

Рисунок 7 – План выполнения нового запроса

На рисунке 8 представлен код составного некластеризованного индекса, созданного для выполнения текущего запроса.

```
create index NonCL_IndexPasAndVol
on CopyBusInformation(PassangerCapacity, VolumeOfTheTank)
```

Рисунок 8 – Код составного некластеризованного индекса

На рисунке 9 представлен план выполнения запроса, вместе со значением ESC.


```

SET STATISTICS IO ON
SET STATISTICS TIME ON

select *
from CopyBusInformation
where PassengerCapacity >
and BusNumber like '1%'
and Brand like 'BA3X'
and VolumeOfTheTank >
and Mileage > 5000
and FuelConsumption >

```

Clustered Index Seek (Clustered)
Scanning a particular range of rows from a clustered index.

Physical Operation	Clustered Index Seek
Logical Operation	Clustered Index Seek
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.561439 (100%)
Estimated I/O Cost	0.485347
Estimated Subtree Cost	0.561439
Estimated CPU Cost	0.0760922
Estimated Number of Executions	1
Estimated Number of Rows Per Execution	40.6014
Estimated Number of Rows to be Read	69032
Estimated Number of Rows for All Executions	40.6014
Estimated Row Size	61 B
Ordered	True
Node ID	0

Predicate
[BusRouteSchedule].[dbo].[CopyBusInformation].[VolumeOfTheTank] < (300) AND [BusRouteSchedule].[dbo].[CopyBusInformation].[Mileage] > (5000) AND [BusRouteSchedule].[dbo].[CopyBusInformation].[FuelConsumption] > (10) AND [BusRouteSchedule].[dbo].[CopyBusInformation].[BusNumber] like N'1%' AND [BusRouteSchedule].[dbo].[CopyBusInformation].[Brand] like N'BA3X'

Object
[BusRouteSchedule].[dbo].[CopyBusInformation].[CL_IndexPasCap]

Output List
[BusRouteSchedule].[dbo].[CopyBusInformation].BusID,
[BusRouteSchedule].[dbo].[CopyBusInformation].Brand,
[BusRouteSchedule].[dbo].[CopyBusInformation].BusNumber,
[BusRouteSchedule].[dbo].[CopyBusInformation].PassangerCapacity,
[BusRouteSchedule].[dbo].[CopyBusInformation].Mileage,
[BusRouteSchedule].[dbo].[CopyBusInformation].FuelConsumption,
[BusRouteSchedule].[dbo].[CopyBusInformation].VolumeOfTheTank

Seek Predicates
Seek Keys[1]: Start: [BusRouteSchedule].[dbo].[CopyBusInformation].PassangerCapacity > Scalar Operator((30))

Query 1: Query cost (relative to select * from CopyBusInformation)

SELECT Cost: 0 %

Clustered Index Seek (Clus [CopyBusInformation].[CL_ Cost: 100 %

Query executed successfully.

Рисунок 11 – План выполнения нового запроса

На рисунке 12 представлен код покрывающего некластеризованного индекса, созданного для выполнения текущего запроса.

```

create index NonCL_Index_AllColumns
on CopyBusInformation(PassangerCapacity)
include (BusID, Brand, BusNumber, Mileage, FuelConsumption, VolumeOfTheTank)

```

Рисунок 12 – Код покрывающего некластеризованного индекса

На рисунке 13 представлен план выполнения запроса, вместе со значением ESC.

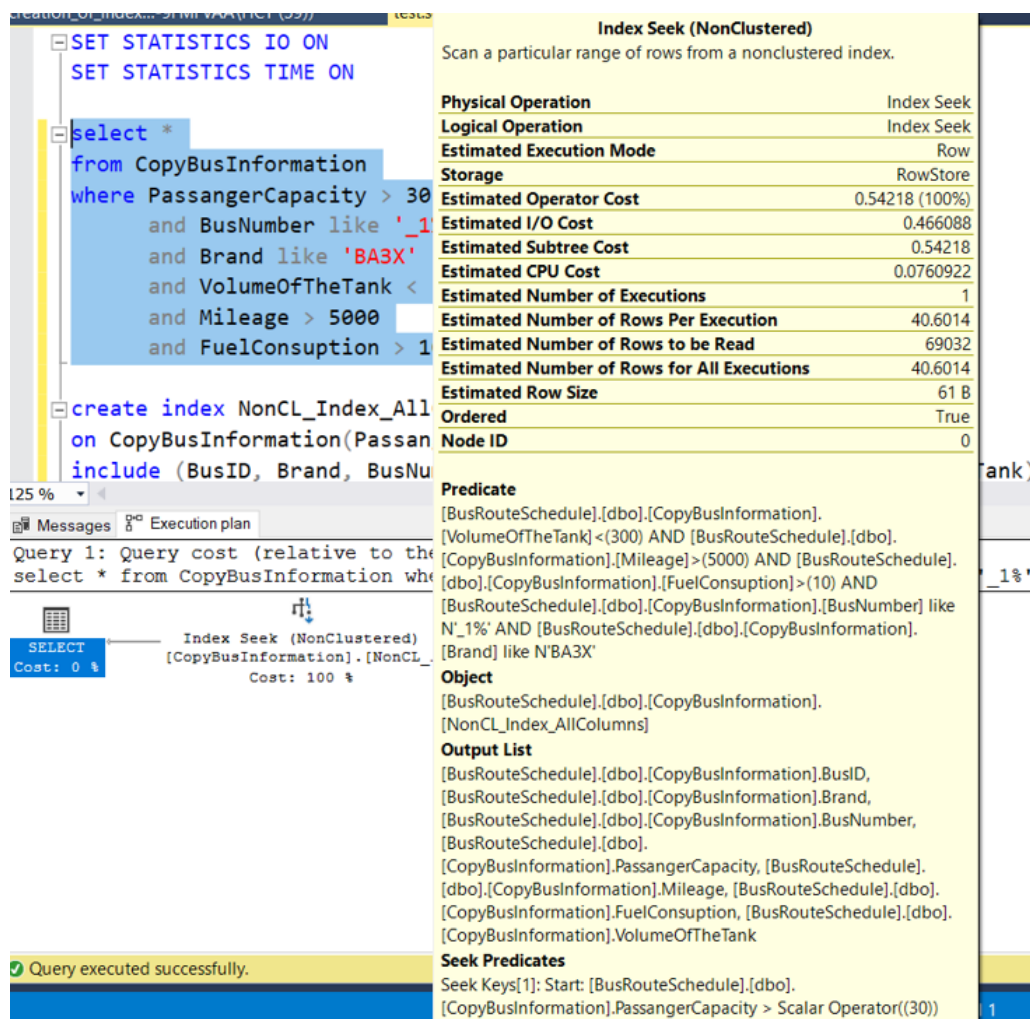


Рисунок 13 – План выполнения нового запроса после создания покрывающего некластерного индекса

— Уникальный индекс

На рисунке 14 представлен запрос и план выполнения данного запроса.


```

SET STATISTICS IO ON
SET STATISTICS TIME ON

select BusID
from CopyBusInformation
where BusID % 5 = 0

create unique index Unique_Index_BusId
on CopyBusInformation (BusID)

drop index Unique_Index_BusId

```

Query 1: Query cost (relative to the select BusID from CopyBusInformation)

SELECT
Cost: 0 %

Clustered Index Scan (Clustered)
[CopyBusInformation].[CL_IndexPasCap]
Cost: 100 %

Clustered Index Scan (Clustered)	
Scanning a clustered index, entirely or only a range.	
Physical Operation	Clustered Index Scan
Logical Operation	Clustered Index Scan
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.811801 (100%)
Estimated I/O Cost	0.701644
Estimated Subtree Cost	0.811801
Estimated CPU Cost	0.110157
Estimated Number of Executions	1
Estimated Number of Rows to be Read	100000
Estimated Number of Rows for All Executions	20000
Estimated Number of Rows Per Execution	20000
Estimated Row Size	11 B
Ordered	False
Node ID	0
Predicate [BusRouteSchedule].[dbo].[CopyBusInformation].[BusID]%[@1] =CONVERT_IMPLICIT(int,[@2],0)	
Object [BusRouteSchedule].[dbo].[CopyBusInformation].[CL_IndexPasCap]	
Output List [BusRouteSchedule].[dbo].[CopyBusInformation].BusID	

Query executed successfully.

DESKTOP-9FMFVAJ

Ch 1

Рисунок 14 – Запрос

На рисунке 15 представлен код покрывающего некластеризованного индекса, созданного для выполнения текущего запроса.

```

create unique index Unique_Index_BusId
on CopyBusInformation (BusID)

```

Рисунок 15 – Код уникального некластеризованного индекса

Уникальный индекс добавляет ограничение, что все записи должны иметь разное значение для этого столбца или комбинации столбцов. Если вставить запись с той же комбинацией, то возникнет ошибка, которая предотвратит вставку (или обновление).

Использование уникального индекса предполагает то, что пользователь будет полностью уверен, что в данном столбце будет не более одной записи для какого-либо ключа.

На рисунке 16 представлен план выполнения запроса, вместе со значением ESC.

```

SET STATISTICS IO ON
SET STATISTICS TIME ON

select BusID
from CopyBusInformation
where BusID % 5 = 0

create unique index Unique_Index_BusId
on CopyBusInformation (BusID)

drop index Unique_Index_BusId

```

125 %

Messages Execution plan

Query 1: Query cost (relative to select BusID from CopyBusInformation)

SELECT Index Scan (NonClustered) [CopyBusInformation].[Unique_Index_BusId] Cost: 0 %

Query executed successfully.

Index Scan (NonClustered)	
Scan a nonclustered index, entirely or only a range.	
Physical Operation	Index Scan
Logical Operation	Index Scan
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.314023 (100%)
Estimated I/O Cost	0.203866
Estimated Subtree Cost	0.314023
Estimated CPU Cost	0.110157
Estimated Number of Executions	1
Estimated Number of Rows to be Read	100000
Estimated Number of Rows for All Executions	20000
Estimated Number of Rows Per Execution	20000
Estimated Row Size	11 B
Ordered	False
Node ID	0
Predicate [BusRouteSchedule].[dbo].[CopyBusInformation].[BusID]%(1) = CONVERT_IMPLICIT(int,(2),0)	
Object [BusRouteSchedule].[dbo].[CopyBusInformation].[Unique_Index_BusId]	
Output List [BusRouteSchedule].[dbo].[CopyBusInformation].BusID	

col 1

Рисунок 16 – План выполнения нового запроса после создания уникального некластерного индекса

— Индекс с включенными столбцами

На рисунке 17 представлен запрос, а на рисунке 18 – план выполнения данного запроса.

```

select Brand, BusNumber, PassengerCapacity, VolumeOfTheTank
from CopyBusInformation
where PassengerCapacity > 30
and BusNumber like '%bш'
and Brand like 'BA3X'
and VolumeOfTheTank < 300

```

Рисунок 17 – Запрос

SET STATISTICS IO ON
SET STATISTICS TIME ON

```
select Brand, BusNumber, PassangerCapacity, VolumeOfTheTank
from CopyBusInformation
where PassangerCapacity > 30
and BusNumber like 'BA3X'
and Brand like 'BA3X'
and VolumeOfTheTank > 30
```

```
create index NonCL_IndexPas_BusNum_Brand_Vol
on CopyBusInformation(PassangerCapacity)
include (BusNumber, Brand, VolumeOfTheTank)

drop index NonCL_IndexPas_BusNum_Brand_Vol
```

125 %

Messages Execution plan

Query 1: Query cost (relative to query plan)

SELECT [CopyBusInformation].[Brand], [CopyBusInformation].[BusNumber], [CopyBusInformation].[PassangerCapacity], [CopyBusInformation].[VolumeOfTheTank] FROM [CopyBusInformation] WHERE ([CopyBusInformation].[PassangerCapacity] > 30) AND ([CopyBusInformation].[BusNumber] like N'BA3X') AND ([CopyBusInformation].[Brand] like N'BA3X')

Clustered Index Seek (Clustered)

Cost: 0 %

Cost: 100 %

Clustered Index Seek (Clustered)

Scanning a particular range of rows from a clustered index.

Physical Operation	Clustered Index Seek
Logical Operation	Clustered Index Seek
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.561439 (100%)
Estimated I/O Cost	0.485347
Estimated Subtree Cost	0.561439
Estimated CPU Cost	0.0760922
Estimated Number of Executions	1
Estimated Number of Rows Per Execution	13.0878
Estimated Number of Rows to be Read	69032
Estimated Number of Rows for All Executions	13.0878
Estimated Row Size	49 B
Ordered	True
Node ID	0

Predicate

[BusRouteSchedule].[dbo].[CopyBusInformation].[VolumeOfTheTank] < (300) AND [BusRouteSchedule].[dbo].[CopyBusInformation].[BusNumber] like N'%b1W' AND [BusRouteSchedule].[dbo].[CopyBusInformation].[Brand] like N'BA3X'

Object

[BusRouteSchedule].[dbo].[CopyBusInformation].[CL_IndexPasCap]

Output List

[BusRouteSchedule].[dbo].[CopyBusInformation].[Brand], [BusRouteSchedule].[dbo].[CopyBusInformation].[BusNumber], [BusRouteSchedule].[dbo].[CopyBusInformation].[PassangerCapacity], [BusRouteSchedule].[dbo].[CopyBusInformation].[VolumeOfTheTank]

Seek Predicates

Seek Keys[1]: Start: [BusRouteSchedule].[dbo].[CopyBusInformation].[PassangerCapacity] > Scalar Operator((30))

Query executed successfully.

Col 1

Рисунок 18 – План выполнения нового запроса

На рисунке 19 представлен код покрывающего некластеризованного индекса, созданного для выполнения текущего запроса.

```
create index NonCL_IndexPas_BusNum_Brand_Vol
on CopyBusInformation(PassangerCapacity)
include (BusNumber, Brand, VolumeOfTheTank)
```

Рисунок 19 – Код некластеризованного индекса с включенными столбцами

Если сравнить этот код с кодом покрывающего индекса, то он практически идентичен данному. Тогда в чём же отличие покрывающего индекса от индекса с включенными столбцами? Ответ: Если индекс содержит все столбцы, ссылаемые в запросе, то это определено покрывающий индекс. На рисунке 20 представлен план выполнения запроса, вместе со значением ESC.

SQL Query:

```

select Brand, BusNumber, PassangerCapacity, VolumeOfTheTank
from CopyBusInformation
where PassangerCapacity > 300
and BusNumber like 'BA3X'
and Brand like 'BA3X'
and VolumeOfTheTank < 300

create index NonCL_IndexPas_E
on CopyBusInformation(PassangerCapacity)
include (BusNumber, Brand, VolumeOfTheTank)

drop index NonCL_IndexPas_E

```

Execution Plan:

Index Seek (NonClustered)

Scan a particular range of rows from a nonclustered index.

Physical Operation	Index Seek
Logical Operation	Index Seek
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.465884 (100%)
Estimated I/O Cost	0.389792
Estimated Subtree Cost	0.465884
Estimated CPU Cost	0.0760922
Estimated Number of Executions	1
Estimated Number of Rows Per Execution	13.0878
Estimated Number of Rows to be Read	69032
Estimated Number of Rows for All Executions	13.0878
Estimated Row Size	49 B
Ordered	True
Node ID	0

Predicate

[BusRouteSchedule].[dbo].[CopyBusInformation].[VolumeOfTheTank] < (300) AND [BusRouteSchedule].[dbo].[CopyBusInformation].[BusNumber] like N'%blll' AND [BusRouteSchedule].[dbo].[CopyBusInformation].[Brand] like N'BA3X'

Object

[BusRouteSchedule].[dbo].[CopyBusInformation].[NonCL_IndexPas_BusNum_Brand_Vol]

Output List

[BusRouteSchedule].[dbo].[CopyBusInformation].Brand, [BusRouteSchedule].[dbo].[CopyBusInformation].BusNumber, [BusRouteSchedule].[dbo].[CopyBusInformation].PassangerCapacity, [BusRouteSchedule].[dbo].[CopyBusInformation].VolumeOfTheTank

Seek Predicates

Seek Keys[1]: Start: [BusRouteSchedule].[dbo].[CopyBusInformation].PassangerCapacity > Scalar Operator((300))

Query executed successfully.

Рисунок 20 – План выполнения нового запроса после создания индекса с включенными столбцами

— Отфильтрованный индекс

На рисунке 21 представлен запрос и план выполнения данного запроса.

SQL Query:

```

SET STATISTICS IO ON
SET STATISTICS TIME ON

select count(PassangerCapacity)
from CopyBusInformation
where PassangerCapacity > 20
and PassangerCapacity < 50

create index NonCL_Index_PasCap
on CopyBusInformation (PassangerCapacity)
where PassangerCapacity = 24

drop index NonCL_Index_PasCap on CopyBusInformation

```

Execution Plan:

Clustered Index Seek (Clustered)

Scanning a particular range of rows from a clustered index.

Physical Operation	Clustered Index Seek
Logical Operation	Clustered Index Seek
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.23995 (93%)
Estimated I/O Cost	0.207569
Estimated Subtree Cost	0.23995
Estimated CPU Cost	0.0323804
Estimated Number of Executions	1
Estimated Number of Rows to be Read	29294
Estimated Number of Rows for All Executions	29294
Estimated Number of Rows Per Execution	9 B
Ordered	True
Node ID	2

Object

[BusRouteSchedule].[dbo].[CopyBusInformation].[CL_IndexPasCap]

Seek Predicates

Seek Keys[1]: Start: [BusRouteSchedule].[dbo].[CopyBusInformation].PassangerCapacity > Scalar Operator (CONVERT_IMPLICIT(int,[@1],0)), End: [BusRouteSchedule].[dbo].[CopyBusInformation].PassangerCapacity < Scalar Operator (CONVERT_IMPLICIT(int,[@2],0))

Query executed successfully.

Рисунок 21 – Запрос

На рисунке 22 представлен код покрывающего некластеризованного индекса, созданного для выполнения текущего запроса.

```
create index NonCL_Index_PasCap
on CopyBusInformation (PassangerCapacity)
where PassangerCapacity = 24
```

Рисунок 22 – Код уникального отфильтрованного индекса

На рисунке 23 представлен план выполнения запроса, вместе со значением ESC.

```
SET STATISTICS IO ON
SET STATISTICS TIME ON

select count(PassangerCapacity)
from CopyBusInformation
where PassangerCapacity > 20
and PassangerCapacity < 50

create index NonCL_Index_PasCap
on CopyBusInformation (PassangerCapacity)
where PassangerCapacity = 24

drop index NonCL_Index_PasCap on CopyBusInformation
```

125 %

Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

select count(PassangerCapacity) from CopyBusInformation where Pa

SELECT Cost: 0 %

Compute Scalar Cost: 0 %

Stream Aggregate (Aggregate) Cost: 7 %

Clustered Index Seek [CopyBusInformation] Cost: 93 %

Clustered Index Seek (Clustered)	
Scanning a particular range of rows from a clustered index.	
Physical Operation	Clustered Index Seek
Logical Operation	Clustered Index Seek
Estimated Execution Mode	Row
Storage	RowStore
Estimated Operator Cost	0.23995 (93%)
Estimated I/O Cost	0.207569
Estimated Subtree Cost	0.23995
Estimated CPU Cost	0.0323804
Estimated Number of Executions	1
Estimated Number of Rows to be Read	29294
Estimated Number of Rows for All Executions	29294
Estimated Number of Rows Per Execution	29294
Estimated Row Size	9 B
Ordered	True
Node ID	2
Object	[BusRouteSchedule].[dbo].[CopyBusInformation].[CL_IndexPasCap]
Seek Predicates	Seek Keys[1]: Start: [BusRouteSchedule].[dbo].[CopyBusInformation].PassangerCapacity > Scalar Operator((20)), End: [BusRouteSchedule].[dbo].[CopyBusInformation].PassangerCapacity < Scalar Operator((50))

Query executed successfully.

ESKTOP-9FMFVAA

Рисунок 23 – План выполнения нового запроса после создания отфильтрованного индекса

Стоит отметить, что значение ESC не поменялось, потому что была отфильтрованный индекс не выполнил свою работу из-за некорректного запроса. Изменим условие в коде отфильтрованного индекса и, создав заново текущий индекс, выполним запрос, как показано на рисунке 24.

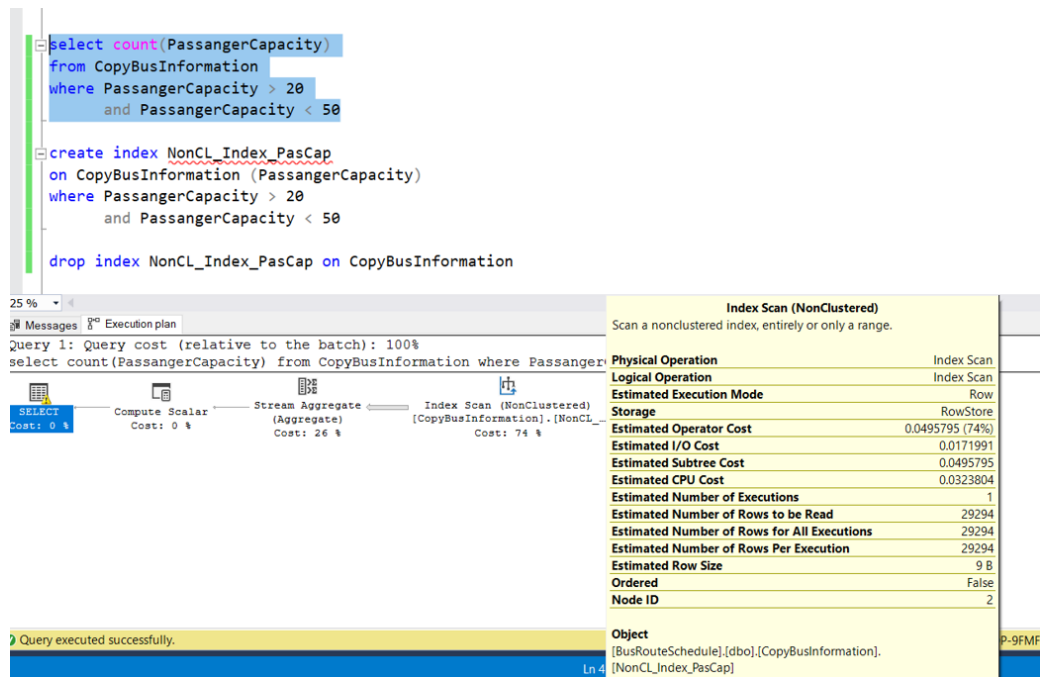


Рисунок 24 – План запроса после создания нового отфильтрованного индекса

Получается, что этот тип индекса будет использоваться только тогда, когда его предложение WHERE совпадает с предложением WHERE запроса.