**פתרון תרגילים מחוברת תרגילים**

**פרק 19 - Indexes and Statistics**

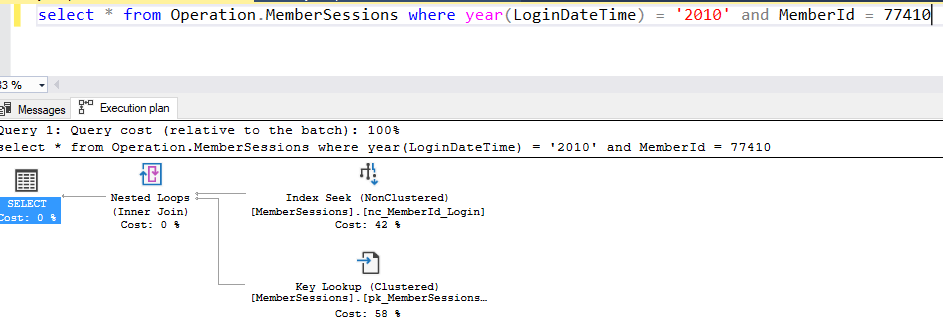
1.

CREATE NONCLUSTERED INDEX [nc\_MemberId\_Login]

ON [Operation].[MemberSessions] ([MemberId] ASC, [LoginDateTime] ASC)

WITH (FILLFACTOR = 80) ON [PRIMARY]

2.



3.

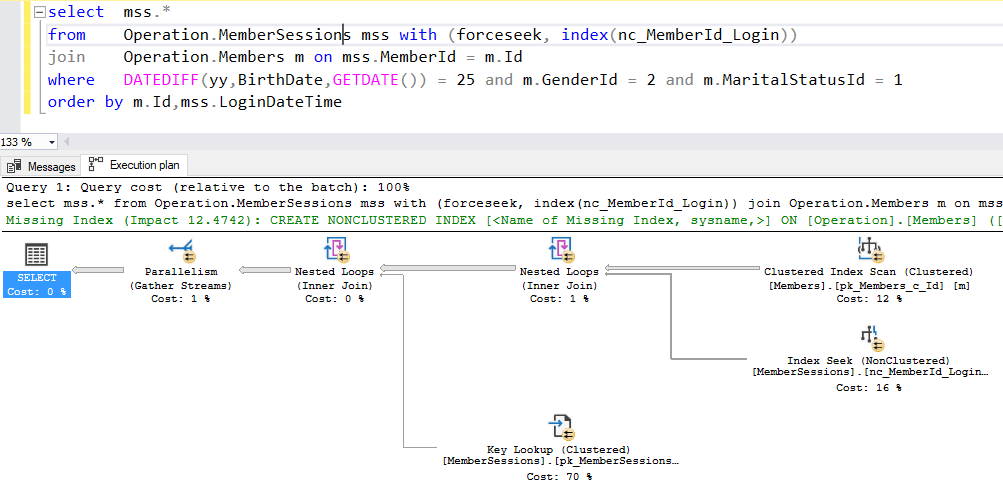
select mss.\*

from Operation.MemberSessions mss with (forceseek, index(nc\_MemberId\_Login))

join Operation.Members m on mss.MemberId = m.Id

where DATEDIFF(yy,BirthDate,GETDATE()) = 25 and m.GenderId = 2 and m.MaritalStatusId = 1

order by m.Id,mss.LoginDateTime



4.

CREATE UNIQUE NONCLUSTERED INDEX [nc\_usn\_pwd] ON [Operation].[Members]

(

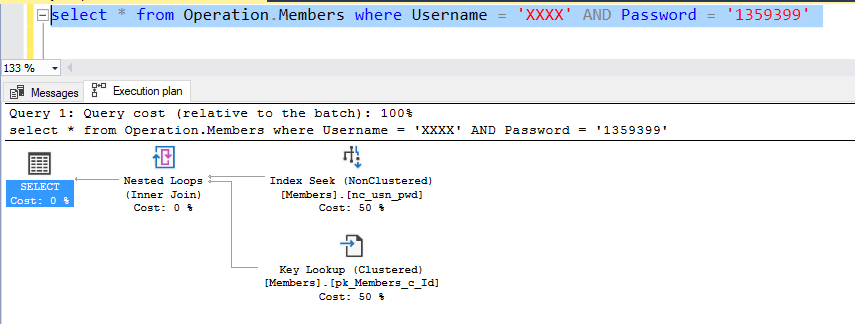
[Username] ASC,

[Password] ASC

)

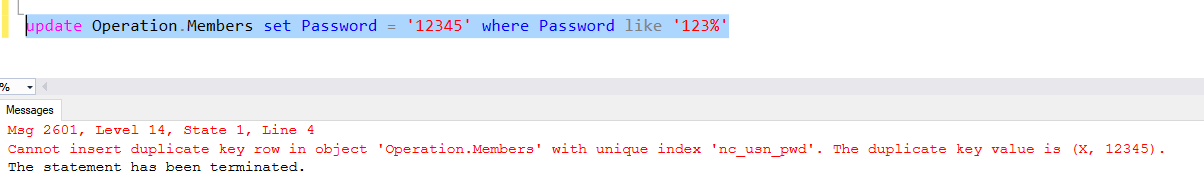
5.

select \* from Operation.Members where Username = 'XXXX' AND Password = '1359399'

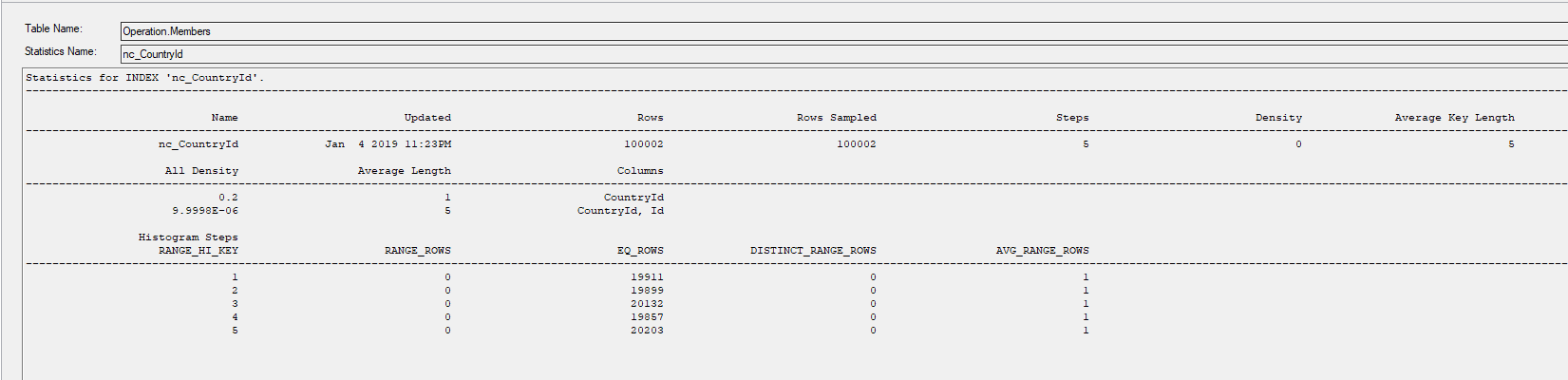


6.

The index above prevents insertion/update of same user and password



7.



8.

select \*

from Operation.Invitations i

join Operation.MemberSessions ms on i.RequestingSessionId = ms.Id

join Operation.Members m on ms.MemberId = m.Id and FirstName = 'Paul' and LastName = 'Simon'

Adding index on First and Last names:

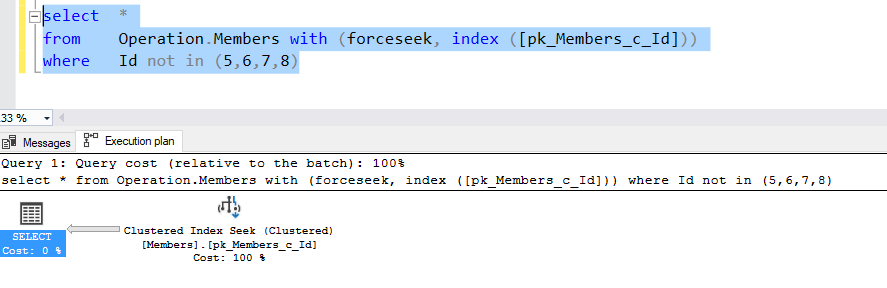
create nonclustered index nc\_fn\_ln on [Operation].[Members] (FirstName asc,LastName asc) ON [PRIMARY]

9.

select \*

from Operation.Members with (forceseek, index ([pk\_Members\_c\_Id]))

where Id not in (5,6,7,8)



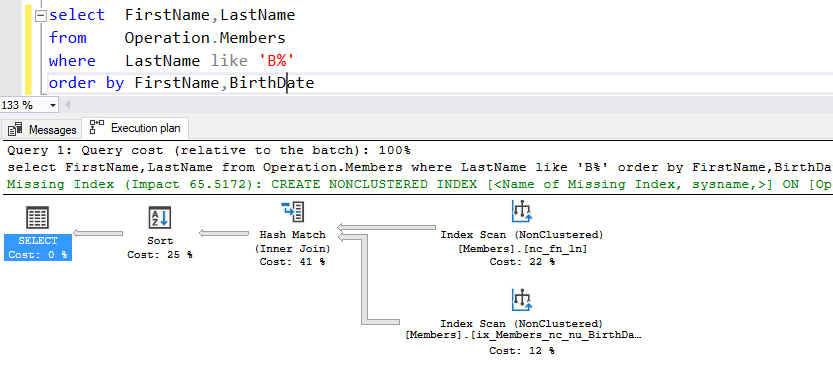
10.

select FirstName,LastName

from Operation.Members

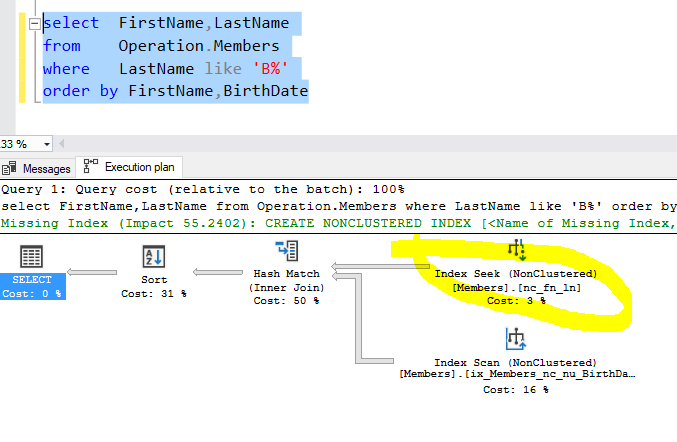
where LastName like 'B%'

order by FirstName,BirthDate

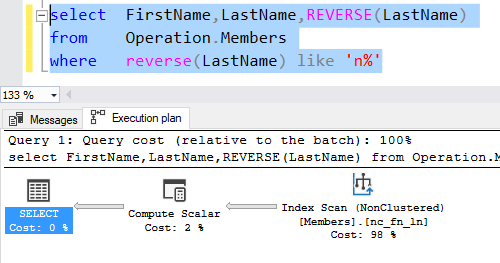


There is no seek index, because it is built on the first name and not on the last name, so sql still need to scan all the rows.

If I change my existing index and switch the last name and first name, it will search with index seek:



11. In order to improve search we can reverse the Last Name:



12. The only way to use index on substring is to build a separate calculated column and build index on it:

select FirstName,LastName,EmailAddress,

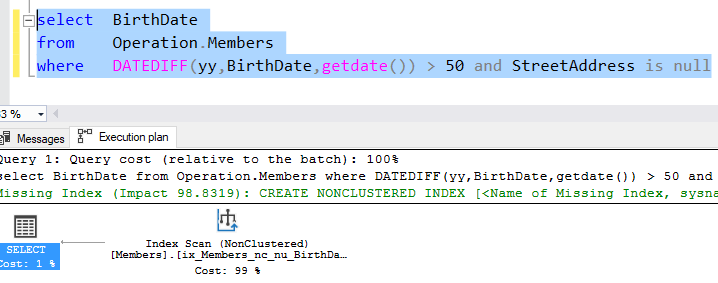
SUBSTRING(EmailAddress,

CHARINDEX('@',EmailAddress)+1,

LEN(EmailAddress)-CHARINDEX('@',EmailAddress)+1)

from Operation.Members

13. In order to improve the query I added the column StreetAdress to the exisitng index [ix\_Members\_nc\_nu\_BirthDate#GenderId#SexualPreferenceId]:



14.

ALTER TABLE [Operation].[Members] ADD CONSTRAINT [pk\_Members\_c\_Id] PRIMARY KEY CLUSTERED ([Id] ASC)

WITH (FILLFACTOR = 100) ON [PRIMARY]

15.

CREATE NONCLUSTERED INDEX [NC\_EndDateTime\_EndReasonId\_MemberId] ON [Operation].[MemberSessions]

(

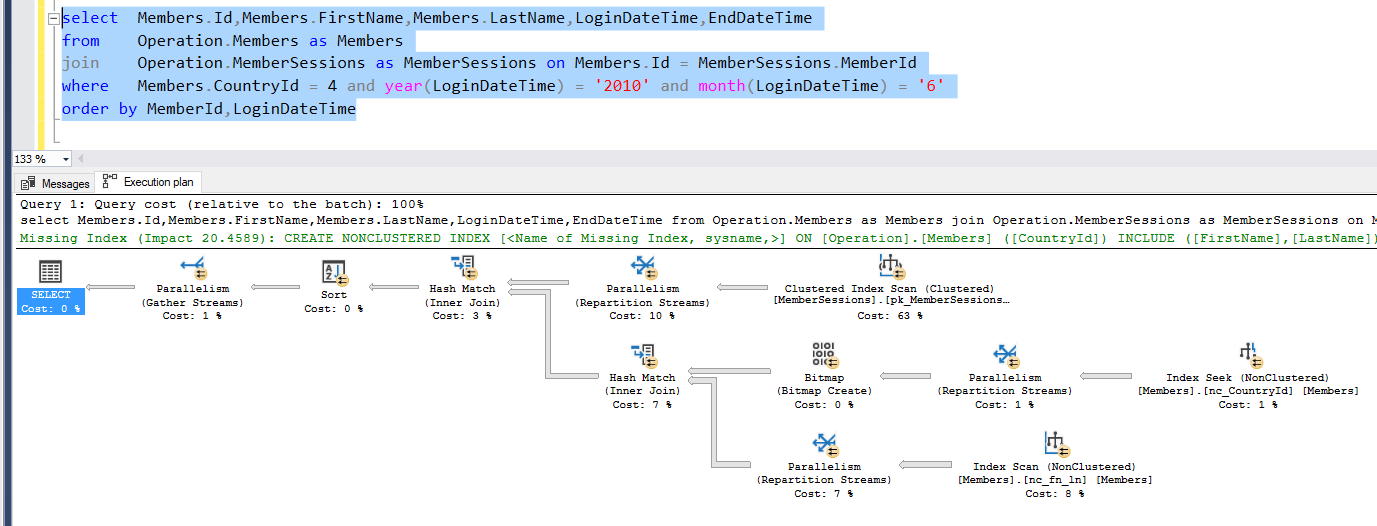
[EndDateTime] ASC,

[EndReasonId] ASC

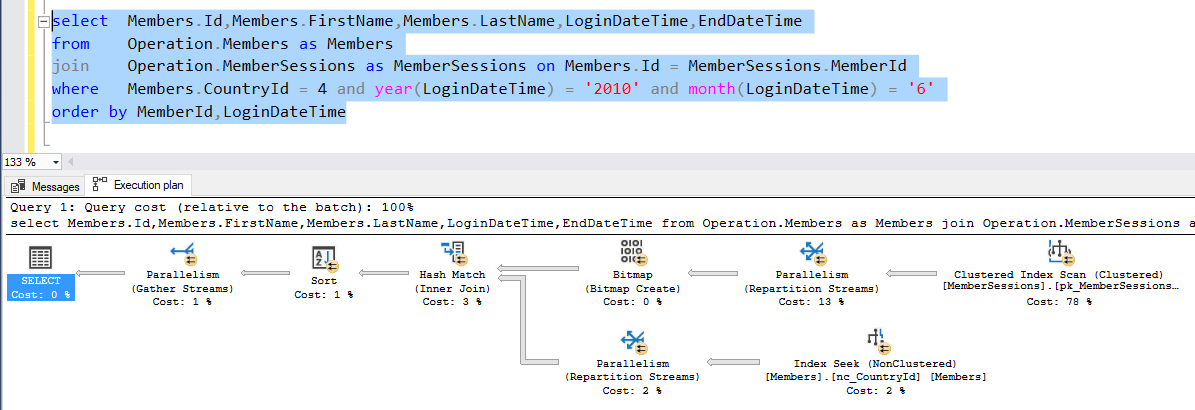
)

INCLUDE ([MemberId]) WITH (FILLFACTOR = 80) ON [PRIMARY]

16.



17. In order to improve I've added included columns First and Last Names to the existing index on CountryId:



18.

select s.name as SchemaName,t.name as TableName,i.name as IndexName,

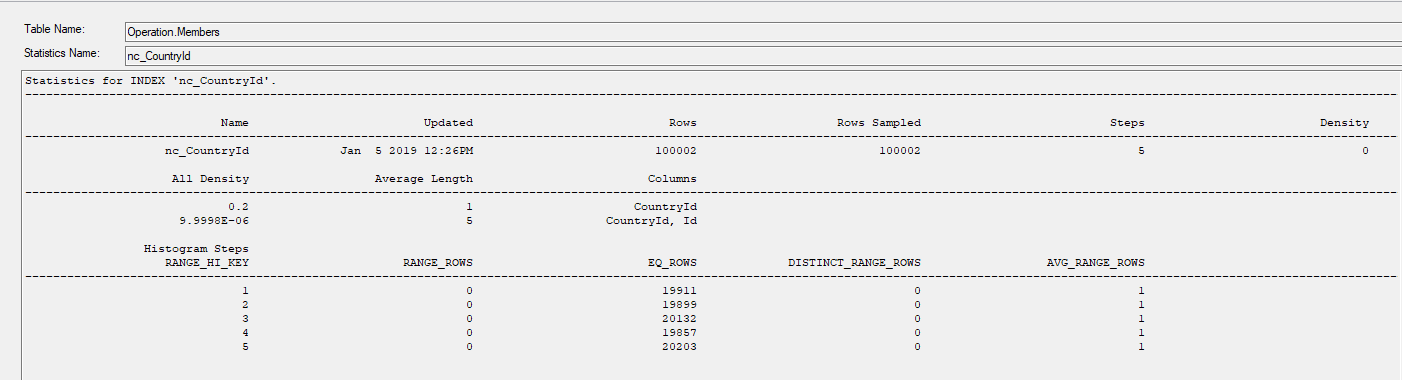
i.type\_desc as IndexType,i.is\_unique as IsUnique,i.fill\_factor as FillFactorValue

from sys.indexes i

join sys.tables t on i.object\_id = t.object\_id

join sys.schemas s on t.schema\_id = s.schema\_id

19.



20.

create table CheckFillFactor(col nvarchar(100) not null)

--insert values

declare @i int = 0, @text nvarchar(100) = 'a';

while @i < 1000

begin

insert into CheckFillFactor values (CONCAT(@text,cast(@i as nvarchar)))

set @i = @i + 1

end

--test inserted values

select \* from CheckFillFactor order by col desc

--create non-clustered index with 80% fill factor

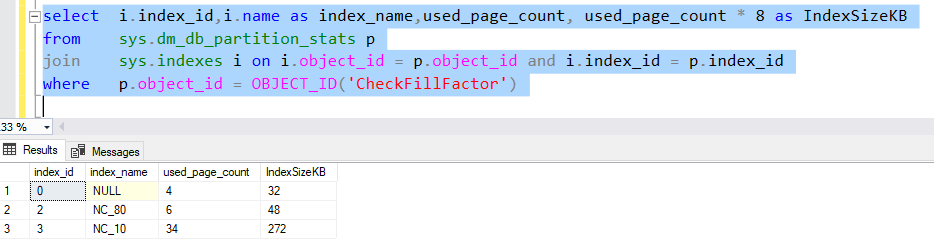
CREATE NONCLUSTERED INDEX [NC\_80] ON [dbo].[CheckFillFactor]([col] ASC)

WITH (FILLFACTOR = 80) ON [PRIMARY]

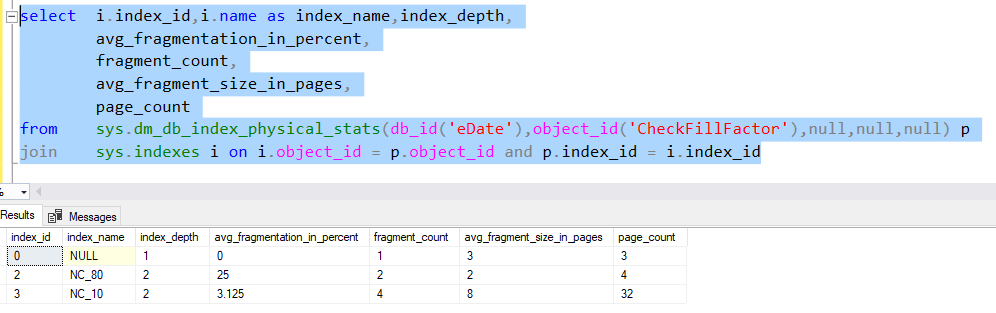
--create non-clustered index with 10% fill factor

CREATE NONCLUSTERED INDEX [NC\_10] ON [dbo].[CheckFillFactor] ([col] ASC)

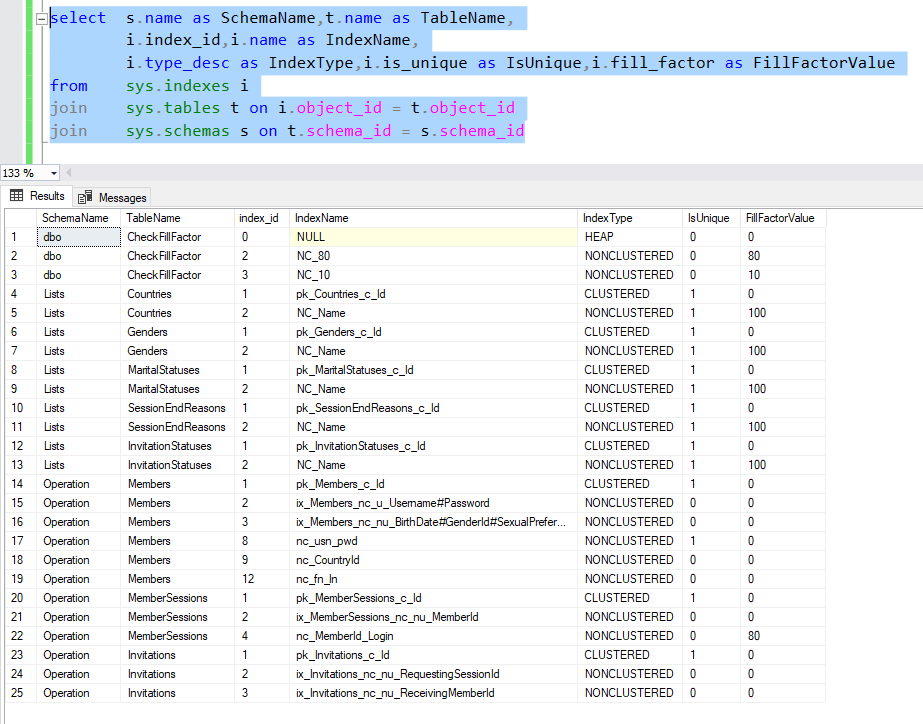
WITH (FILLFACTOR = 10) ON [PRIMARY]



21.



22.



23.

create table TestTable(col\_a char(10) not null)

--insert values

declare @i int = 0;

while @i < 10000

begin

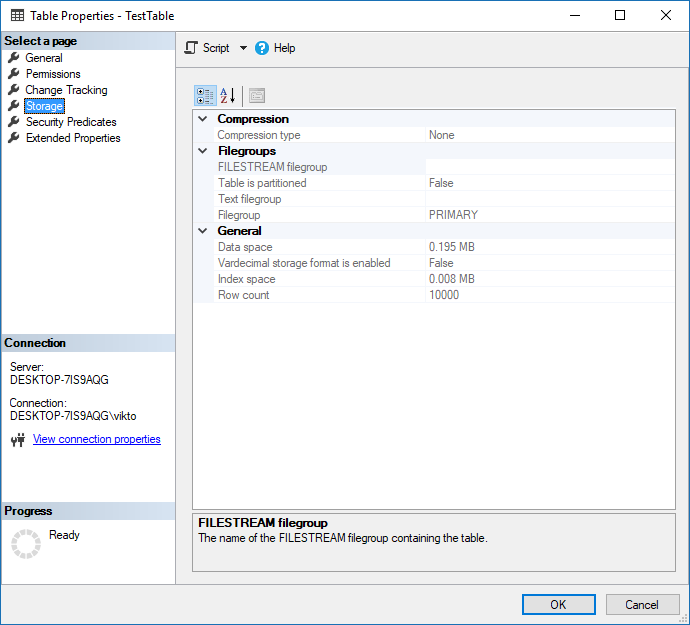
insert into TestTable (col\_a) values ('A')

set @i = @i + 1

end

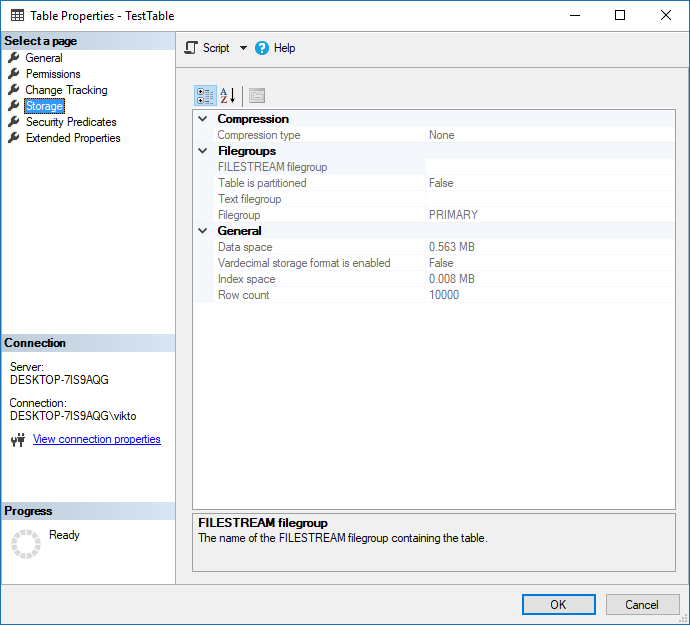
--add new calculated column

alter table TestTable add calculated\_a as col\_a+col\_a



--add new persisted calculated column

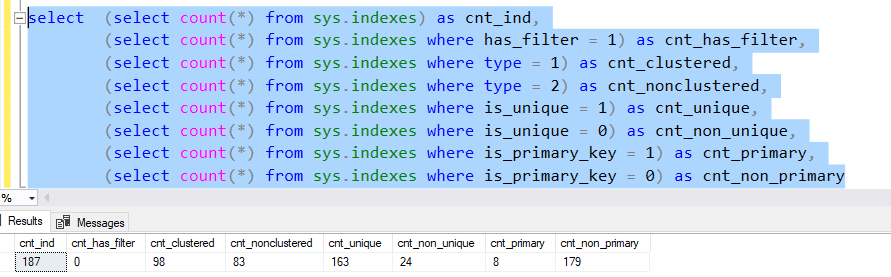
alter table TestTable add persisted\_a as col\_a+col\_a persisted



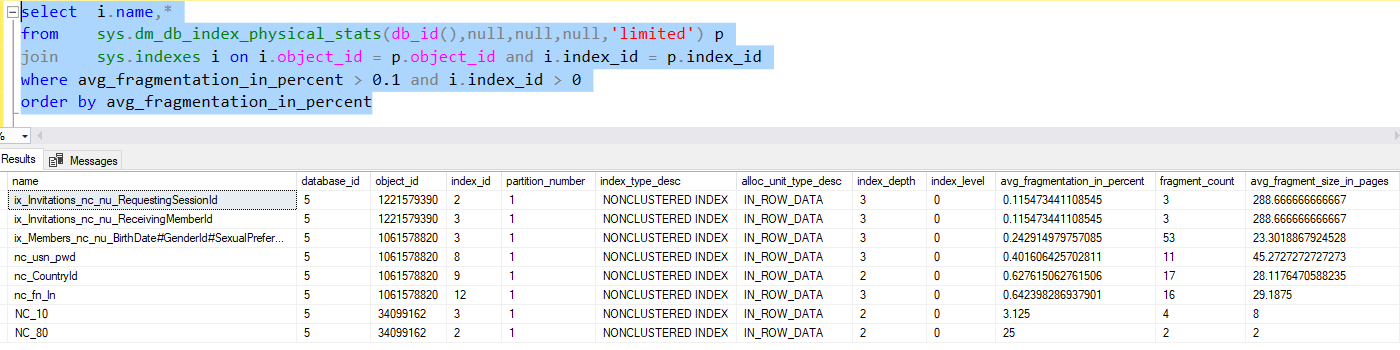
24. Birthday is not null - correct for filtered index

Birthday like '2003' - possible but not useful, filter index has benefits when it uses simple where clauses and can filter most of the data from table or when it retrieves the most common queries.

25.



26.



DECLARE @index\_name nvarchar(100),@table\_name nvarchar(100),@avg float

DECLARE cur CURSOR FOR

select i.name as index\_name,s.name+'.'+t.name as table\_name,avg\_fragmentation\_in\_percent

from sys.dm\_db\_index\_physical\_stats(db\_id(),null,null,null,'limited') p

join sys.indexes i on i.object\_id = p.object\_id and i.index\_id = p.index\_id

join sys.tables t on i.object\_id = t.object\_id

join sys.schemas s on t.schema\_id = s.schema\_id

where avg\_fragmentation\_in\_percent > 0.1 and i.index\_id > 0

OPEN cur

FETCH NEXT FROM cur INTO @index\_name,@table\_name,@avg

WHILE @@FETCH\_STATUS = 0

BEGIN

if @avg >= 0.3

BEGIN

PRINT 'avg\_fragmentation\_in\_percent = ' + cast(@avg as nvarchar);

PRINT 'alter index' + @index\_name + ' on ' + @table\_name + ' rebuild';

END

else

BEGIN

PRINT 'avg\_fragmentation\_in\_percent = ' + cast(@avg as nvarchar);

PRINT 'alter index' + @index\_name + ' on ' + @table\_name + ' reorganize';

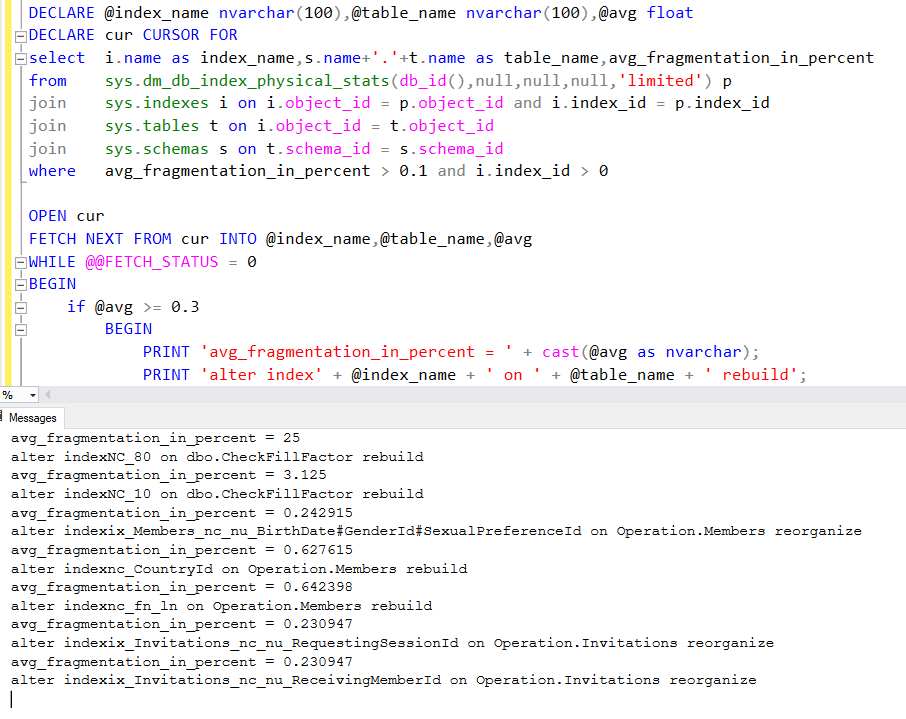
END

FETCH NEXT FROM cur INTO @index\_name,@table\_name,@avg;

END

CLOSE cur;

DEALLOCATE cur;



27.

create table #temp\_table(table\_name nvarchar(100),column\_name nvarchar(100))

declare @table nvarchar(100),@column nvarchar(100);

declare c2 cursor for

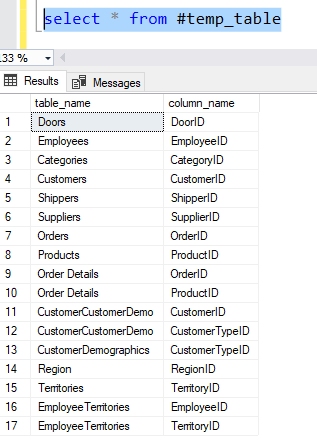
select t.name as table\_name,c.name as col\_name

from sys.tables t

join sys.indexes i on t.object\_id = i.object\_id

join sys.index\_columns ic on i.index\_id = ic.index\_id and i.object\_id = ic.object\_id

join sys.columns c on ic.column\_id = c.column\_id and c.object\_id = t.object\_id

where i.is\_primary\_key = 1

OPEN c2

FETCH NEXT FROM c2 INTO @table,@column

WHILE @@FETCH\_STATUS = 0

begin

print @table + ' ' + @column;

insert into #temp\_table values(@table,@column);

FETCH NEXT FROM c2 INTO @table,@column;

end

CLOSE c2;

DEALLOCATE c2;

28.

select t.name as TableName,

i.index\_id,i.name as IndexName,

i.type\_desc as IndexType,

case when i.is\_unique = 1 then 'Unique' else 'Non-Unique' end as IsUnique,

o.create\_date,

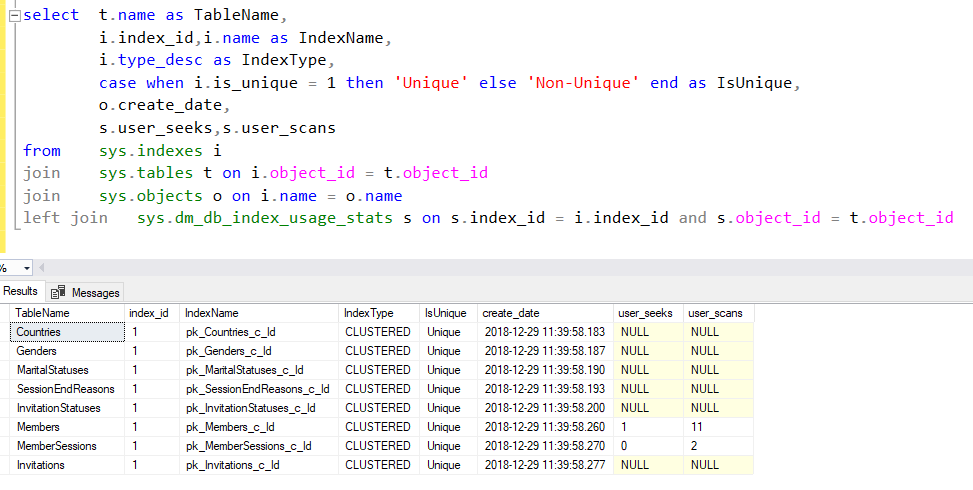
s.user\_seeks,s.user\_scans

from sys.indexes i

join sys.tables t on i.object\_id = t.object\_id

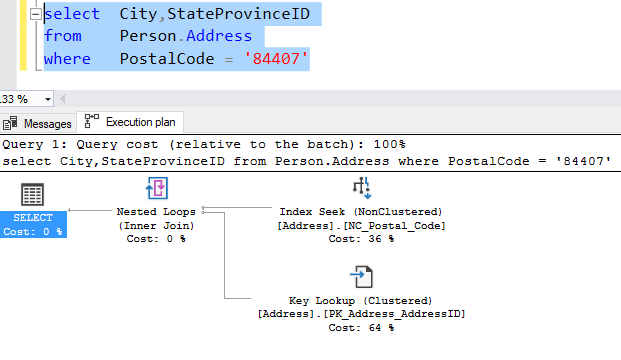
join sys.objects o on i.name = o.name

left join sys.dm\_db\_index\_usage\_stats s on s.index\_id = i.index\_id and s.object\_id = t.object\_id



29.

CREATE NONCLUSTERED INDEX [NC\_Postal\_Code] ON [Person].[Address] ([PostalCode] ASC) ON [PRIMARY]

Before adding included columns: 

After index update with adding columns: 