

Assignment

Week 10

Maziar Shajari | Advanced databases| Nov 2020

In this assignment, we use the ALTER TABLE command to add and drop constraints to a table, including primary key, unique, and foreign key constraints.

Note: do not create any other table, all tables already exists except for one (products) and you will create it. Just run the code below. Use Adventureworks2017 database.

/\*

USE AdventureWorks2017

CREATE TABLE Production.Products (

productid INT NOT NULL IDENTITY,

productname NVARCHAR(40) NOT NULL,

supplierid INT NOT NULL,

categoryid INT NOT NULL,

unitprice MONEY NOT NULL

CONSTRAINT DFT\_Products\_unitprice DEFAULT(0),

discontinued BIT NOT NULL

CONSTRAINT DFT\_Products\_discontinued DEFAULT(0),

CONSTRAINT PK\_Products PRIMARY KEY(productid),

CONSTRAINT FK\_Products\_Categories FOREIGN KEY(categoryid)

REFERENCES Production.ProductCategory(Productcategoryid),

CONSTRAINT CHK\_Products\_unitprice CHECK(unitprice >= 0))

\*/

1. Test the primary key using the following. Execute the queries one at a time and after every Insert command explain the output message.

SET IDENTITY\_INSERT Production.Products ON;

GO

INSERT INTO Production.Products (productid, productname, supplierid, categoryid,

unitprice, discontinued)

VALUES (1, N'Product TEST', 1, 1, 18, 0);

GO

SET IDENTITY\_INSERT Production.Products OFF;

INSERT INTO Production.Products (productid, productname, supplierid, categoryid,

unitprice, discontinued)

VALUES (2, N'Product TEST', 1, 1, 18, 0);  
GO

SELECT productid, productname, supplierid, categoryid,

unitprice, discontinued

FROM Production.Products;

Include any messages you get and explain them.

Answer:

This is error message

Msg 544, Level 16, State 1, Line 10

Cannot insert explicit value for identity column in table 'Products' when IDENTITY\_INSERT is set to OFF.

This message occurred because IDENTITY\_INSERT is set to off.  
so identity field value wasn’t inserted.  
In this case we can use following query instead.

(productid is inserted automatically)

SET IDENTITY\_INSERT Production.Products OFF;

INSERT INTO Production.Products (productname, supplierid, categoryid,

unitprice, discontinued)

VALUES ( N'Product TEST', 1, 1, 18, 0);  
GO

2. Insert a new row that lets the Identity property assign a new productid.

INSERT INTO Production.Products (productname, supplierid, categoryid, unitprice,discontinued)

VALUES (N'Product TEST', 1, 1, 18, 0);

SELECT productid, productname, supplierid, categoryid,

unitprice, discontinued

FROM Production.Products

Copy and paste any message you get from SSMS and briefly explain it.

Answer:

There is no error message.

This query runs successfully because productid is inserted automatically.

1. Delete the test row.

DELETE FROM Production.Products

WHERE productname = N'Product TEST';

Copy and paste any message you get from SSMS.

Answer:

(1 row affected)

Completion time: 2020-11-27T16:24:06.8598389-04:00

1. Try again with an invalid categoryid = 99.

INSERT INTO Production.Products (productname, supplierid, categoryid, unitprice, discontinued)

VALUES (N'Product TEST', 1, 99, 18, 0);GO

Explain the reason the INSERT failed and Copy and paste any message you get from SSMS.

Answer:

categoryid is foreign key reference of Production.ProductCategory id.

So categoryid must exist in Production.ProductCategory id list.

But there is no.

So error occurs.

Msg 547, Level 16, State 0, Line 1

The INSERT statement conflicted with the FOREIGN KEY constraint "FK\_Products\_Categories". The conflict occurred in database "AdventureWorks2017", table "Production.ProductCategory", column 'ProductCategoryID'.

The statement has been terminated.

1. Drop the foreign key constraint on Production.Products.

Show your query and the SSMS message.

Answer:

ALTER TABLE Production.Products

DROP CONSTRAINT FK\_Products\_Categories;

1. After doing #5 above, try the insert now with the invalid categoryid = 99.

INSERT INTO Production.Products (productname, supplierid, categoryid, unitprice, discontinued)

VALUES (N'Product TEST', 1, 99, 18, 0);

Explain the reason the INSERT failed before and Copy and paste any message you get from SSMS.

Answer:

categoryid was foreign key reference of Production.ProductCategory id.

So categoryid had to exist in Production.ProductCategory id list.

But there was no.

Now foregn key constraint is removed.

So it’s success.

No error message.

(1 row affected)

1. Try to add the foreign key constraint back in using WITH CHECK.

ALTER TABLE Production.Products WITH CHECK

ADD CONSTRAINT FK\_Products\_Categories FOREIGN KEY(categoryid)

REFERENCES Production.ProductCategory (ProductCategoryID);

GO

What happened and why? Include the message.

Answer:

Error occurred.

It’s because 99 value is in table.

So when check, error occurs

Message:

Msg 547, Level 16, State 0, Line 1

The ALTER TABLE statement conflicted with the FOREIGN KEY constraint "FK\_Products\_Categories". The conflict occurred in database "AdventureWorks2017", table "Production.ProductCategory", column 'ProductCategoryID'.

1. Update the row so that it has a valid categoryid.

UPDATE Production.Products

SET categoryid = 1

WHERE productname = N'Product TEST';

GO

Include the message and execute the select query below:

select \* from Production.Products

Answer:

This is message.

(1 row affected)

9. Now try to add the foreign key constraint back to the table.

ALTER TABLE Production.Products WITH CHECK

ADD CONSTRAINT FK\_Products\_Categories FOREIGN KEY(categoryid)

REFERENCES Production.ProductCategory (ProductCategoryID);

GO

This time what happened? Can you explain why?

Answer:

This is success.

Message:

Commands completed successfully.

This table is updated.

So now, categoryid value is in Production.ProductCategory ids.

Therefore when check, everything is okay.

So This command completed successfully.

10. Drop the test row from the table.

DELETE FROM Production.Products

WHERE productname = N'Product TEST';

Answer:

Success.

11- Do some research to find out how to list all the constraints using a T-SQL command.

Run the command and list:

All the Primary Key, Foreign Key and Unique constraints in the AdventureWorks table. There are several ways to reach to a solution but try to find the simple one.

Answer:

select table\_view,

object\_type,

constraint\_type,

constraint\_name,

details

from (

select schema\_name(t.schema\_id) + '.' + t.[name] as table\_view,

case when t.[type] = 'U' then 'Table'

when t.[type] = 'V' then 'View'

end as [object\_type],

case when c.[type] = 'PK' then 'Primary key'

when c.[type] = 'UQ' then 'Unique constraint'

when i.[type] = 1 then 'Unique clustered index'

when i.type = 2 then 'Unique index'

end as constraint\_type,

isnull(c.[name], i.[name]) as constraint\_name,

substring(column\_names, 1, len(column\_names)-1) as [details]

from sys.objects t

left outer join sys.indexes i

on t.object\_id = i.object\_id

left outer join sys.key\_constraints c

on i.object\_id = c.parent\_object\_id

and i.index\_id = c.unique\_index\_id

cross apply (select col.[name] + ', '

from sys.index\_columns ic

inner join sys.columns col

on ic.object\_id = col.object\_id

and ic.column\_id = col.column\_id

where ic.object\_id = t.object\_id

and ic.index\_id = i.index\_id

order by col.column\_id

for xml path ('') ) D (column\_names)

where is\_unique = 1

and t.is\_ms\_shipped <> 1

union all

select schema\_name(fk\_tab.schema\_id) + '.' + fk\_tab.name as foreign\_table,

'Table',

'Foreign key',

fk.name as fk\_constraint\_name,

schema\_name(pk\_tab.schema\_id) + '.' + pk\_tab.name

from sys.foreign\_keys fk

inner join sys.tables fk\_tab

on fk\_tab.object\_id = fk.parent\_object\_id

inner join sys.tables pk\_tab

on pk\_tab.object\_id = fk.referenced\_object\_id

inner join sys.foreign\_key\_columns fk\_cols

on fk\_cols.constraint\_object\_id = fk.object\_id

union all

select schema\_name(t.schema\_id) + '.' + t.[name],

'Table',

'Check constraint',

con.[name] as constraint\_name,

con.[definition]

from sys.check\_constraints con

left outer join sys.objects t

on con.parent\_object\_id = t.object\_id

left outer join sys.all\_columns col

on con.parent\_column\_id = col.column\_id

and con.parent\_object\_id = col.object\_id

union all

select schema\_name(t.schema\_id) + '.' + t.[name],

'Table',

'Default constraint',

con.[name],

col.[name] + ' = ' + con.[definition]

from sys.default\_constraints con

left outer join sys.objects t

on con.parent\_object\_id = t.object\_id

left outer join sys.all\_columns col

on con.parent\_column\_id = col.column\_id

and con.parent\_object\_id = col.object\_id) a

order by table\_view, constraint\_type, constraint\_name