

Lab 09

Trigger

Phần I – Hướng dẫn thực hành từng bước

1. Mục tiêu

- Hiểu khái niệm Trigger
- Hiểu các loại khác nhau của triggers
- Hiểu các phương pháp để tạo DML trigger
- Hiểu các phương pháp để thay đổi DML trigger
- Mô tả trigger lồng nhau
- Hiểu việc xử lý nhiều hàng trong một phiên
- Hiểu ý nghĩa hiệu suất của triggers.

2. Thực hiện

Trigger là một thủ tục lưu trữ được thực hiện khi có sự thay đổi dữ liệu trong bảng được bảo vệ bằng Trigger. Tất nhiên Trigger không giống thủ tục lưu trữ tiêu chuẩn, Trigger không thể thực thi bằng lệnh gọi trực tiếp, truyền hoặc nhận tham số. Trigger được thực thi một cách tự động khi xuất hiện những hành động như INSERT, UPDATE, DELETE....

Trigger có thể chứa logic xử lý phức tạp ở mức nghiệp vụ nhằm duy trì tính toàn vẹn dữ liệu và tính đúng đắn của quy trình nghiệp vụ của sản phẩm phần mềm.

Trigger có 3 loại cơ bản:

DML Trigger: là các trigger thực thi khi dữ liệu được chèn, sửa đổi hoặc xóa trong một bảng hoặc View sử dụng câu lệnh INSERT, UPDATE, DELETE.

DDL Trigger: là các trigger thực thi khi một bảng hoặc View được tạo ra, chỉnh sửa, hoặc xóa bằng cách sử dụng câu lệnh CREATE, ALTER, DELETE.

Logon Trigger: các trigger đăng nhập, thực thi các thủ tục lưu trữ khi một phiên giao dịch được thiết lập với sự kiện Logon. Các trigger loại này kiểm soát phiên làm việc của máy chủ nhằm hạn chế số lần đăng nhập không hợp lệ hoặc hạn chế số lượt phiên.

So sánh 2 loại Trigger

DDL Trigger	DML Trigger
Các trigger DDL thực thi các thủ tục lưu trữ trên câu lệnh CREATE, ALTER, và DROP.	Các trigger DML thực thi trên các câu lệnh INSERT, UPDATE, và DELETE.
Các trigger DDL được sử dụng để kiểm tra và kiểm soát các hoạt động của cơ sở dữ liệu.	Các trigger DML được sử dụng để thực thi các quy tắc nghiệp vụ khi dữ liệu được sửa đổi trong các bảng hoặc view.
Các trigger DDL chỉ hoạt động sau khi bảng hoặc view được sửa đổi.	Các trigger DML thực thi trong khi sửa đổi dữ liệu hoặc sau khi dữ liệu được sửa đổi.
Các trigger DDL được định nghĩa ở mức cơ sở dữ liệu hoặc máy chủ.	Các trigger DML được định nghĩa ở mức cơ sở dữ liệu.

Bài thực hành 1: Tạo trigger cho bảng bắt sự kiện khi thêm dữ liệu vào không được nằm ngoài khoảng quy định.

Bước 1: Viết lệnh SQL tạo csdl và bảng như sau, thêm dữ liệu mẫu:

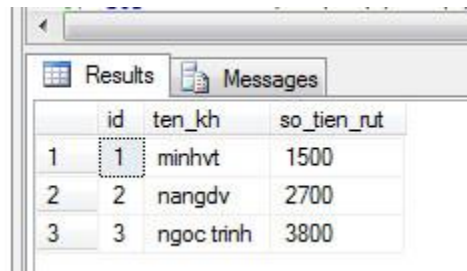
```
-- Bài thực hành Lab 09
CREATE DATABASE BKShop
GO
USE BKShop
GO

-- Thực hiện tạo bảng giao dịch rút tiền
CREATE TABLE giao_dich_khachhang(
    id int IDENTITY(1,1) PRIMARY KEY,
    ten_kh nvarchar(64),
    so_tien_rut int
)

-- Thêm dữ liệu
INSERT INTO giao_dich_khachhang(ten_kh, so_tien_rut) VALUES
('minhvt', 1500),
('nangdv', 2700),
('ngoc trịnh', 3800)

-- Truy vấn
```

```
SELECT * FROM giao_dich_khachhang
```



	id	ten_kh	so_tien_rut
1	1	minhvt	1500
2	2	nangdv	2700
3	3	ngoc trinh	3800

Bước 2: Tạo trigger kiểm soát việc nhập dữ liệu (INSERT) vào bảng

GIAO_DICH_KHACH_HANG không vượt quá 5000 đồng và thông báo:

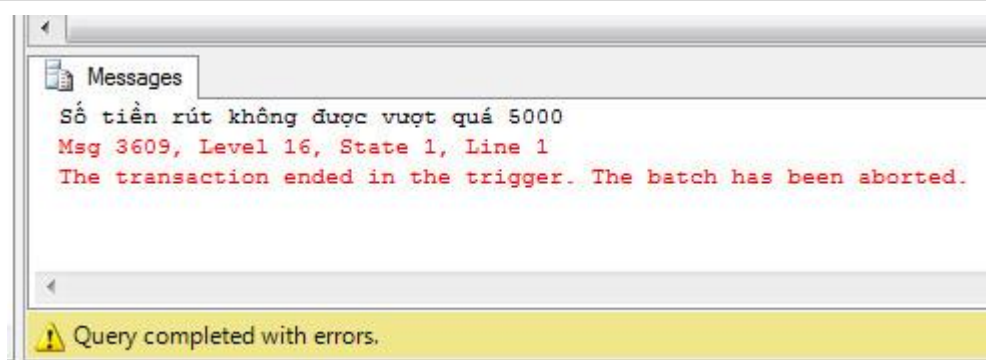
```
-- Tạo 1 trigger: yêu cầu khi thực hiện rút tiền không vượt quá 5000
CREATE TRIGGER KiemTra_SoTienRut
ON giao_dich_khachhang
FOR INSERT -- Có các hành động: INSERT | UPDATE | DELETE
AS -- định nghĩa thân của trigger
-- IF - Kiểm tra những điều kiện ràng buộc nghiệp vụ để đảm bảo tính đúng đắn của dữ liệu
IF (SELECT so_tien_rut FROM inserted) > 5000
BEGIN -- bắt đầu nhóm lệnh nghiệp vụ
    PRINT N'Số tiền rút không được vượt quá 5000'
    ROLLBACK TRANSACTION
END -- kết thúc các lệnh nghiệp vụ
```

Bước 3: Thực thi việc insert dữ liệu vào bảng với dữ liệu số tiền rút nhỏ hơn 5000:

```
-- Thực thi thêm dữ liệu sau khi đã có TRIGGER
INSERT INTO giao_dich_khachhang(ten_kh, so_tien_rut) VALUES
('minhvt', 3333)
```



```
INSERT INTO giao_dich_khachhang(ten_kh, so_tien_rut) VALUES
('minhvt', 5555)
```



Câu lệnh bị từ chối do trigger chặn.

```
-- Tạo 1 trigger: yêu cầu khi thực hiện rút tiền không vượt quá 5000
CREATE TRIGGER KiemTra_SoTienRut
ON giao_dich_khachhang
FOR INSERT -- Có các hành động: INSERT | UPDATE | DELETE
AS -- định nghĩa thân của trigger
-- IF - Kiểm tra những điều kiện ràng buộc nghiệp vụ để đảm bảo tính đúng đắn của dữ liệu
IF (SELECT so_tien_rut FROM inserted) > 5000
BEGIN -- bắt đầu nhóm lệnh nghiệp vụ
    PRINT N'Số tiền rút không được vượt quá 5000'
    ROLLBACK TRANSACTION
END -- kết thúc các lệnh nghiệp vụ
```

Trong cú pháp tạo lệnh Trigger, có lệnh SELECT từ bảng inserted, đây là một bảng được tạo bởi MS SQL Server cho lệnh DML của trigger. Bảng inserted là một trong 2 bảng (inserted và deleted) đặc biệt vì về khía cạnh vật lý nó không hiện diện trong CSDL, nó được tạo và xóa khi sự kiện kích hoạt xảy ra. Tóm lại nó là một bảng tạm dùng trong trigger.

Từ khóa **FOR**, đi kèm với **INSERT** (hoặc **UPDATE**, **DELETE**) nhằm ngụ ý trigger DML được thực thi ngay sau khi hoạt động sửa đổi được hoàn tất. Chính vì vậy, khi kiểm tra nếu không hợp lệ thì trong mã lệnh trigger có thể chèn từ khóa **ROLLBACK TRANSACTION** để hoàn tác vụ về trạng thái trước khi lệnh sửa đổi thực hiện.

Bài thực hành 2: Tạo trigger kiểm soát ngày rút tiền trong bảng trên không được là ngày trong tương lai.

Bước 1: Gõ lệnh SQL thêm cột vào bảng:

```
-- Sửa bảng Giao Dịch thêm cột ngày tháng thực hiện
ALTER TABLE giao_dich_khachhang
ADD ngay_gd DATETIME
```

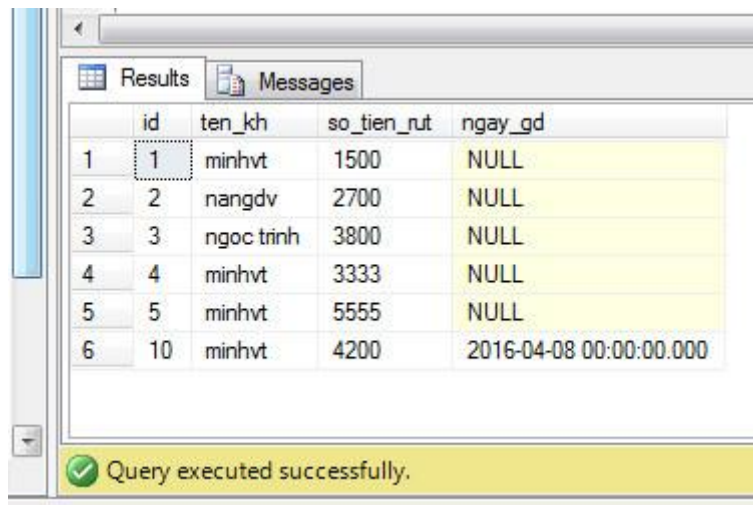
Bước 2: Viết lệnh SQL tạo trigger:

```
-- Tạo Trigger khi thực hiện cập nhật dữ liệu
CREATE TRIGGER kiểmtra_capnhat_gd
ON giao_dich_khachhang
FOR UPDATE
AS
IF (SELECT ngay_gd FROM inserted) > getDate()
BEGIN
    PRINT N'Không thể cập nhật ngày giao dịch trong tương lai'
    ROLLBACK TRANSACTION -- Không cho phép nếu là ngày trong
    tương lai
END
```

Bước 3: Thực thi thêm dữ liệu, sau đó sửa dữ liệu (UPDATE) ngày trong tương lai:

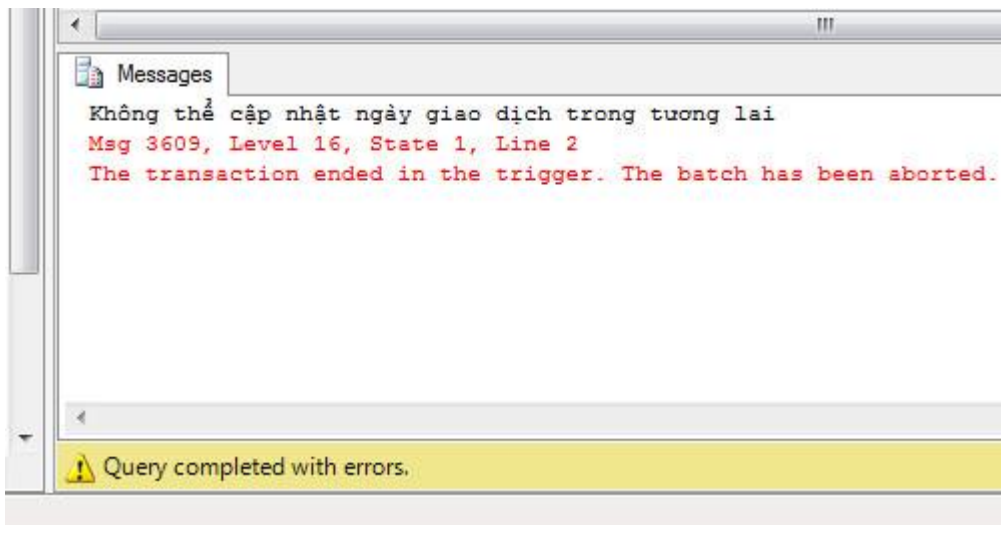
```
-- Thêm dữ liệu
INSERT INTO giao_dich_khachhang(ten_kh, so_tien_rut, ngay_gd)
VALUES ('minhvt', 4200, '2016-04-08') -- Insert vẫn được

SELECT * FROM giao_dich_khachhang
```



	id	ten_kh	so_tien_rut	ngay_gd
1	1	minhvt	1500	NULL
2	2	nangdv	2700	NULL
3	3	ngoc trinh	3800	NULL
4	4	minhvt	3333	NULL
5	5	minhvt	5555	NULL
6	10	minhvt	4200	2016-04-08 00:00:00.000

```
-- Lệnh cập nhật bị từ chối
UPDATE giao_dich_khachhang
SET giao_dich_khachhang.ngay_gd = '2016-04-10'
WHERE giao_dich_khachhang.id = 10
```



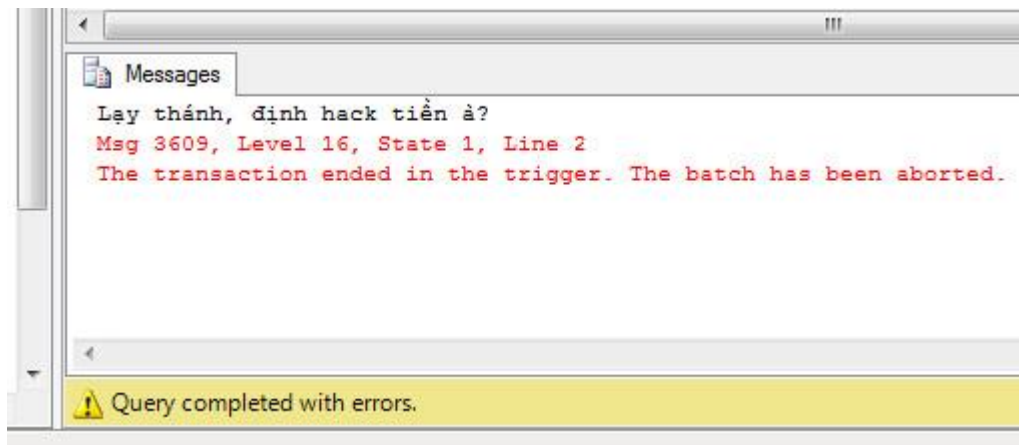
Bài thực hành 3: Tạo trigger cấm cập nhật lên cột số tiền.

Bước 1: Gõ lệnh SQL:

```
-- Yêu cầu: cấm cập nhật lên cột số tiền
CREATE TRIGGER kiểmtra_capnhat_sotien
ON giao_dich_khachhang
FOR UPDATE
AS
IF UPDATE(so_tien_rut)
BEGIN
    PRINT N'Lạy thánh, định hack tiền à?'
    ROLLBACK TRANSACTION
END
```

Bước 2: Thực thi lệnh SQL cập nhật số tiền:

```
-- Hack tiền
UPDATE giao_dich_khachhang
SET so_tien_rut = 200
WHERE id = 6
```



Bài thực hành 4: Tạo trigger cấm xóa giao dịch có ID bằng 10.

Bước 1: Gõ lệnh SQL:

```
-- Trigger cho hành động xóa
CREATE TRIGGER kiểmtra_xoa
ON giao_dich_khachhang
FOR DELETE
AS
IF 10 IN (SELECT id FROM deleted)
BEGIN
    PRINT N'Không thể xóa giao dịch số 10'
    ROLLBACK TRANSACTION
END
```

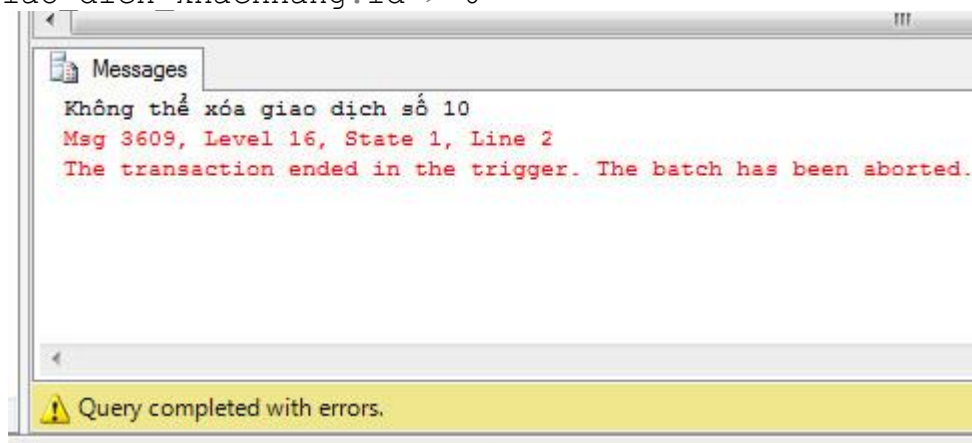
Bước 2: Thực thi lệnh SQL xóa bản ghi:

```
SELECT * FROM giao_dich_khachhang
```

	id	ten_kh	so_tien_rut	ngay_gd
1	1	minhvt	1500	NULL
2	2	nangdv	2700	NULL
3	3	ngoc trinh	3800	NULL
4	4	minhvt	3333	NULL
5	5	minhvt	5555	NULL
6	10	minhvt	4200	2016-04-08 00:00:00.000

```
-- Lệnh xóa
```

```
DELETE giao_dich_khachhang  
WHERE giao_dich_khachhang.id > 0
```



Kết quả là nhận được thông báo lỗi.

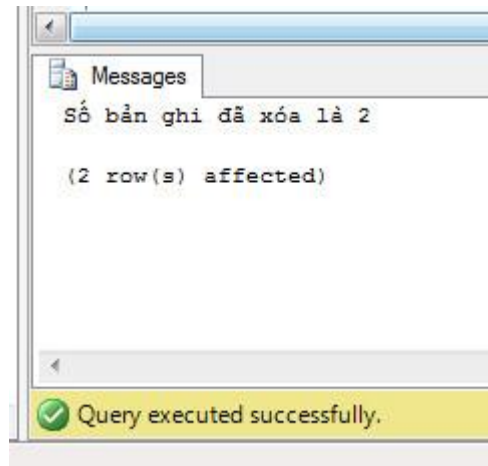
Bài thực hành 5: Tạo trigger khi thực hiện xóa bản ghi thì tính tổng số bản ghi đã xóa và hiển thị ra màn hình.

Bước 1: Gõ lệnh SQL:

```
-- Tạo Trigger After: là những trigger được thực hiện sau khi  
CHUYỂN ĐÃ RỒI  
CREATE TRIGGER hienthi_thongbao_xoa  
ON giao_dich_khachhang  
AFTER DELETE  
AS  
BEGIN  
    DECLARE @sobanghi int;  
    SELECT @sobanghi = COUNT(*) FROM deleted -- Lấy tất cả bản  
ghi trong bảng deleted của trigger  
    PRINT N'Số bản ghi đã xóa là ' + CONVERT(varchar, @sobanghi)  
END
```

Bước 2: Thực thi lệnh SQL xóa bản ghi:

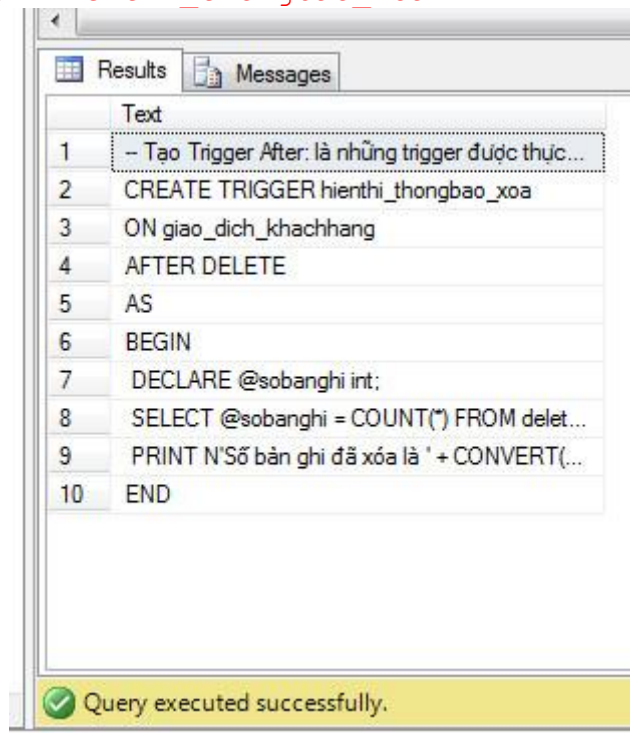
```
-- Lệnh xóa  
DELETE giao_dich_khachhang  
WHERE giao_dich_khachhang.id > 0 AND giao_dich_khachhang.id < 3
```

Bài thực hành 6: Xem cú pháp trigger.

Gõ lệnh SQL:

```
-- Xem cú pháp tạo trigger  
EXEC sp_helptext 'hienthi_thongbao_xoa'
```



Bài thực hành 7: Sửa trigger khi thực hiện xóa bản ghi thì tính tổng số bản ghi đã xóa và hiển thị ra màn hình.

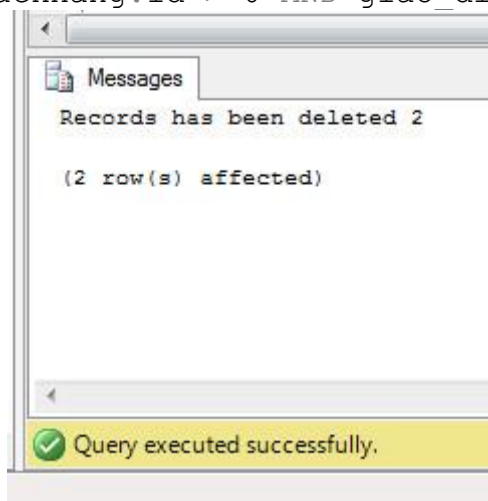
Bước 1: Gõ lệnh SQL:

```
-- Sửa TRIGGER  
ALTER TRIGGER hienthi_thongbao_xoa  
ON giao_dich_khachhang  
AFTER DELETE  
AS
```

```
BEGIN
    DECLARE @sobanghi int;
    SELECT @sobanghi = COUNT(*) FROM deleted -- Lấy tất cả bản ghi trong bảng deleted của trigger
    PRINT N'Records has been deleted ' + CONVERT(varchar, @sobanghi)
END
```

Bước 2: Thực thi lệnh SQL xóa bản ghi:

```
-- Lệnh xóa
DELETE giao_dich_khachhang
WHERE giao_dich_khachhang.id > 0 AND giao_dich_khachhang.id < 5
```

**Bài thực hành 8: Xóa bỏ trigger.****Gỡ lệnh SQL:**

```
-- Xóa bỏ TRIGGER
DROP TRIGGER hienthi_thongbao_xoa
```

Bài thực hành 9: Tạo INSTEAD OF Trigger khi thực hiện xóa bản ghi thì kiểm tra ràng buộc và thực hiện các lệnh cần thiết để đảm bảo ràng buộc.**Bước 1: Gỡ lệnh SQL tạo bảng và thêm dữ liệu:**

```
-- Tạo bảng Phòng Ban
CREATE TABLE PhongBan(
    id_pb int PRIMARY KEY,
    ten_pb nvarchar(128)
)
GO

-- Tạo bảng Nhân Viên
CREATE TABLE NhanVien(
    id_nv int PRIMARY KEY,
    id_pb int FOREIGN KEY REFERENCES PhongBan(id_pb),
```

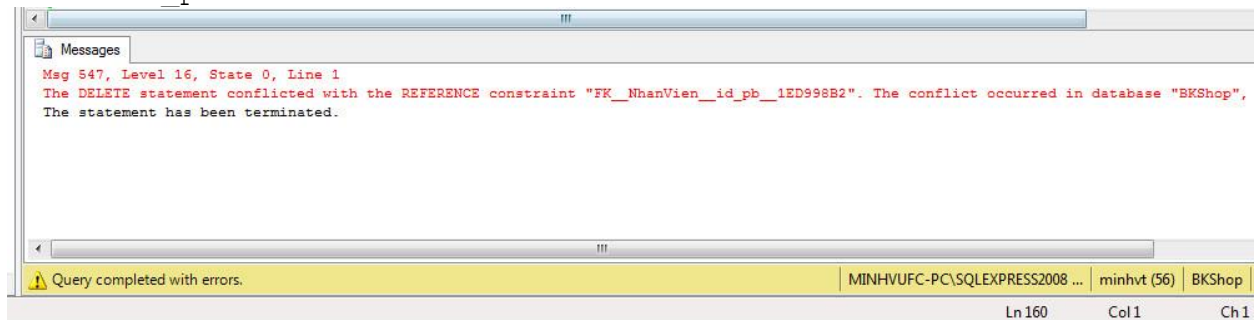
```
ten_nv nvarchar(128)
)
GO

INSERT INTO PhongBan VALUES
(1, 'Phong ke toan'),
(2, 'Hanh chinh TH'),
(3, 'Phong ky thuat')
GO
```

```
INSERT INTO NhanVien VALUES
(1, 1, 'Vu Tuan Minh'),
(2, 1, 'Nguyen Cong Phuong'),
(3, 3, 'Pham Van Mach')
GO
```

-- Thực hiện xóa dữ liệu có ràng buộc

```
DELETE PhongBan
WHERE id_pb = 1
```



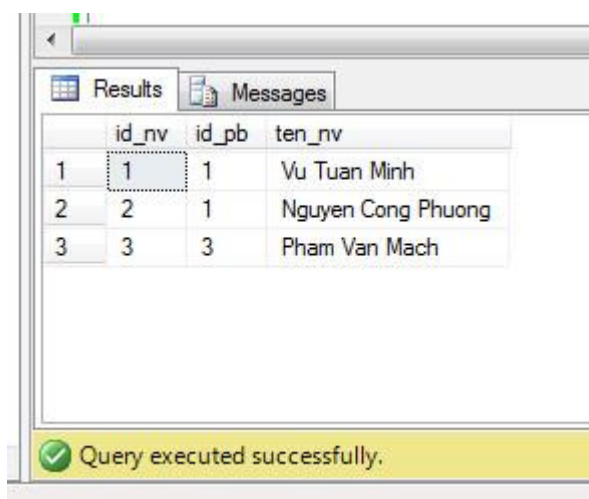
Không thể xóa do có ràng buộc

Bước 2: Viết trigger để thực hiện xóa dữ liệu ở bảng NhanVien khi xóa PhongBan:

```
-- Tạo instead of trigger, nó sẽ được thực thi trước khi các ràng
bộc được kiểm tra
CREATE TRIGGER trg_delete_pb
ON PhongBan
INSTEAD OF DELETE
AS
BEGIN
    DELETE NhanVien WHERE NhanVien.id_pb IN (SELECT id_pb FROM
deleted)
END

-- Thực hiện xóa dữ liệu có ràng buộc
DELETE PhongBan
WHERE id_pb = 1

SELECT * FROM NhanVien
```

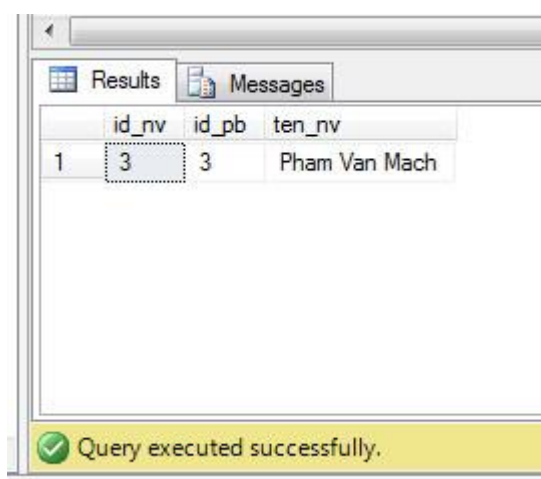


Results Messages

	id_nv	id_pb	ten_nv
1	1	1	Vu Tuan Minh
2	2	1	Nguyen Cong Phuong
3	3	3	Pham Van Mach

Query executed successfully.

Trước khi xóa và...



Results Messages

	id_nv	id_pb	ten_nv
1	3	3	Pham Van Mach

Query executed successfully.

Sau khi xóa

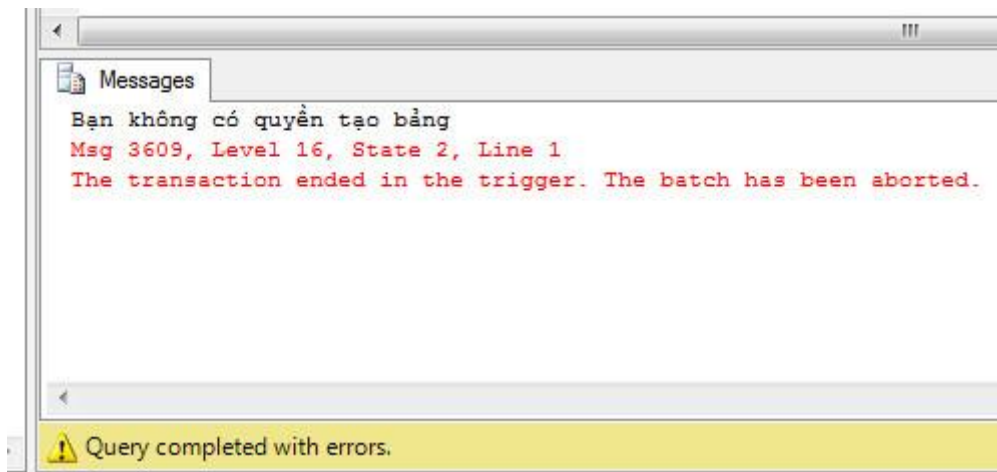
Bài thực hành 10: Viết lệnh tạo Trigger nhóm DDL cấm không được thực hiện thao tác tạo bảng trên csdl BKShop.

Bước 1: Gõ lệnh SQL:

```
-- Cú pháp tạo trigger
CREATE TRIGGER tg_createtb
ON DATABASE
FOR CREATE_TABLE, ALTER_TABLE
AS
    print N'Bạn không có quyền tạo bảng';
    ROLLBACK;
```

Bước 2: Tạo bảng tblTest:

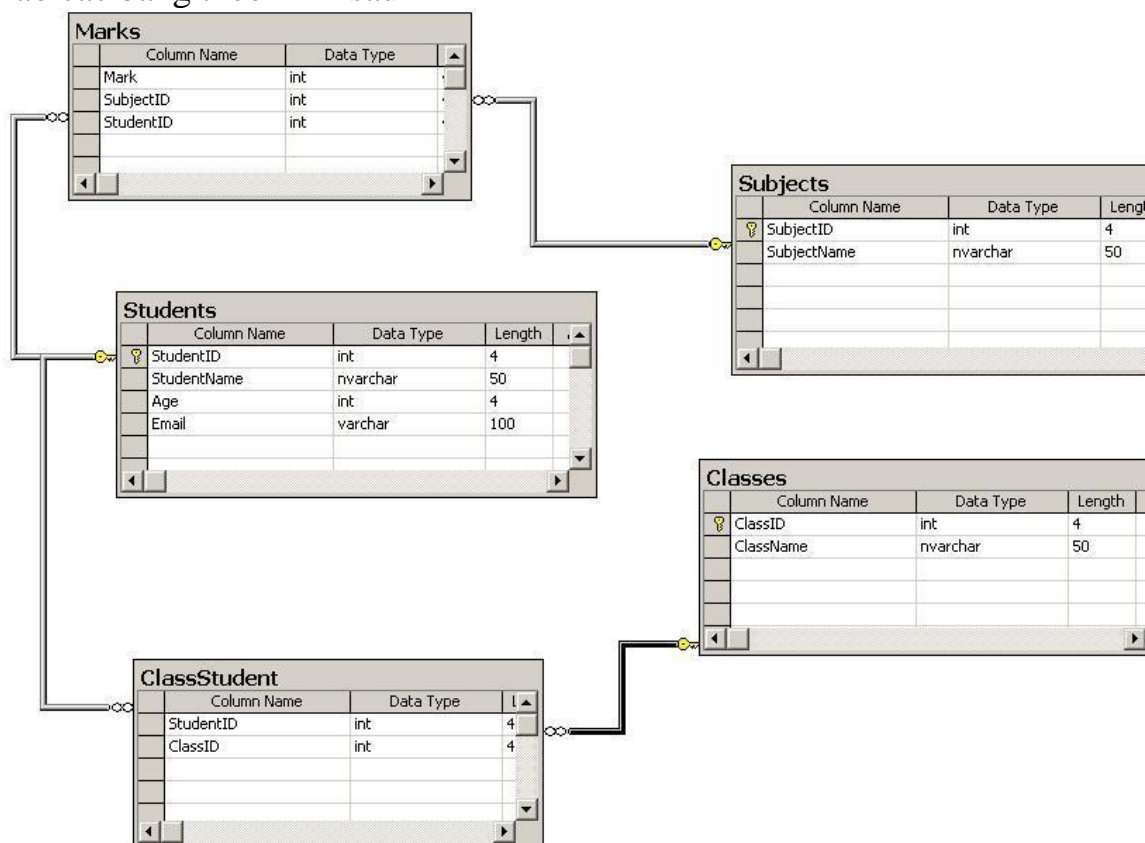
```
CREATE TABLE tblTest (
    id int,
    name varchar(128)
)
```



Phần II - Bài tập tự làm

Sử dụng câu lệnh T-SQL thực hiện các yêu cầu sau

1. Tạo CSDL Lab6_1
2. Tạo các bảng theo hình sau



3. Chèn dữ liệu theo hình sau
Students

<i>StudentID</i>	<i>StudentName</i>	<i>Age</i>	<i>Email</i>
1	Nguyen Quang An	18	an@yahoo.com
2	Nguyen Cong Vinh	20	vinh@gmail.com

3	Nguyen Van Quyen	19	quyen
4	Pham Thanh Binh	25	binh@com
5	Nguyen Van Tai Em	30	taiem@sport.vn

Classes

<i>ClassID</i>	<i>ClassName</i>
1	C1506L
2	C1603G

ClassStudent

<i>ClassID</i>	<i>StudentID</i>
1	1
1	2
2	3
2	4
2	5

Subjects

SubjectID	SubjectName
1	SQL
2	Java
3	C
4	Visual Basic

Marks

SubjectID	StudentID	Mark
1	1	8
2	1	4
1	1	9
1	3	7
1	4	3
2	5	5
3	3	8
3	5	1
2	4	3

4. Tao view:

- Hiện thị danh sách tất cả các học viên (Danh sách phải sắp xếp theo tên học viên)
- Hiện thị danh sách tất cả các môn học
- Hiện thị danh sách những học viên nào có địa chỉ email chính xác

- Hiển thị danh sách những học viên có họ là Nguyễn
- Hiển thị danh sách ác bạn học viên của lớp C0706L
- Hiển thị danh sách và điểm học viên ứng với môn học
- Hiển thị danh sách học viên chưa thi môn nào (Chưa có điểm)
- Hiển thị môn nào chưa được học viên nào thi
- Tính điểm trung bình cho các học viên
- Hiển thị môn học nào được thi nhiều nhất
- Hiển thị môn học nào có học sinh thi được điểm cao nhất
- Hiển thị môn học nào có nhiều điểm dưới trung bình nhất (<5)

5. Tạo ràng buộc

- Viết Check Constraint để kiểm tra độ tuổi nhập vào trong bảng Student yêu cầu $\text{Age} > 15$ và $\text{Age} < 50$
- Loại bỏ tất cả quan hệ giữa các bảng
- Xóa học viên có StudentID là 1
- Trong bảng Student thêm một column Status có kiểu dữ liệu là Bit và có giá trị Default là 1
- Cập nhật giá trị Status trong bảng Student là 1

6. Tạo thủ tục:

- Thủ tục nhận tham số là StudentID và hiển thị điểm từng môn học của sinh viên này
- Thủ tục nhận tham số là SubjectID, thực hiện cập nhật toàn bộ điểm của sinh viên trong môn học này về 0.
- Thủ tục nhận tham số đầu vào là StudentID, SubjectID và hiển thị điểm môn học của sinh viên này.

7. Tạo các trigger

- Trigger thực hiện khi xóa Student thì xóa toàn bộ thông tin tương ứng ở các bảng khác
- Trigger khi thêm mới Marks, nếu $\text{Mark} < 0$ thì thông báo lỗi và không cho insert