

# BÀI TẬP THỰC HÀNH CƠ SỞ DỮ LIỆU

## =====Tuần 9=====

**Giới thiệu:** Trigger là một trường hợp đặc biệt của store procedure, nó sẽ có hiệu lực khi chúng ta thay đổi dữ liệu trên một bảng dữ liệu cụ thể, hoặc các xử lý làm thay đổi dữ liệu của các lệnh: insert, update, delete. Trigger có thể chứa các lệnh truy vấn từ các bảng khác hoặc bao gồm những lệnh SQL phức tạp. Chúng ta sẽ xây dựng một ví dụ đơn giản đầu tiên với một bảng và ví dụ liên quan đến lưu trữ, lấy dữ liệu, cập nhật và xóa các mẫu tin.

### Lệnh tạo Trigger

**CREATE TRIGGER** <tên trigger>

**ON** <tên bảng>|<tên view>

**[WITH ENCRYPTION]**

**{**

**{FOR | AFTER| INSTEAD OF}**

**<INSERT [, UPDATE] [, DELETE]>**

**}**

**AS**

**<câu lệnh SQL>**

### Tạo INSERT trigger.

INSERT trigger đảm bảo dữ liệu nhập vào bảng được đúng đắn.

Xem xét ví dụ:

Tạo INSERT trigger để đảm bảo không có vé nào được đặt vào một ngày trong quá khứ.

```
CREATE TRIGGER insert_trigg
ON Reservation
FOR INSERT
AS
IF((Select journey_date From Inserted)<getdate())
BEGIN
PRINT 'journey_date không thể nhỏ hơn ngày hiện tại'
ROLLBACK TRAN
END
```

3. Sau đó, hãy thử thực hiện thêm một bản ghi có journey\_date < ngày hiện tại của hệ thống.

## Tạo DELETE Trigger

DELETE trigger ngăn cản việc xóa đi những dữ liệu quan trọng trong bảng.

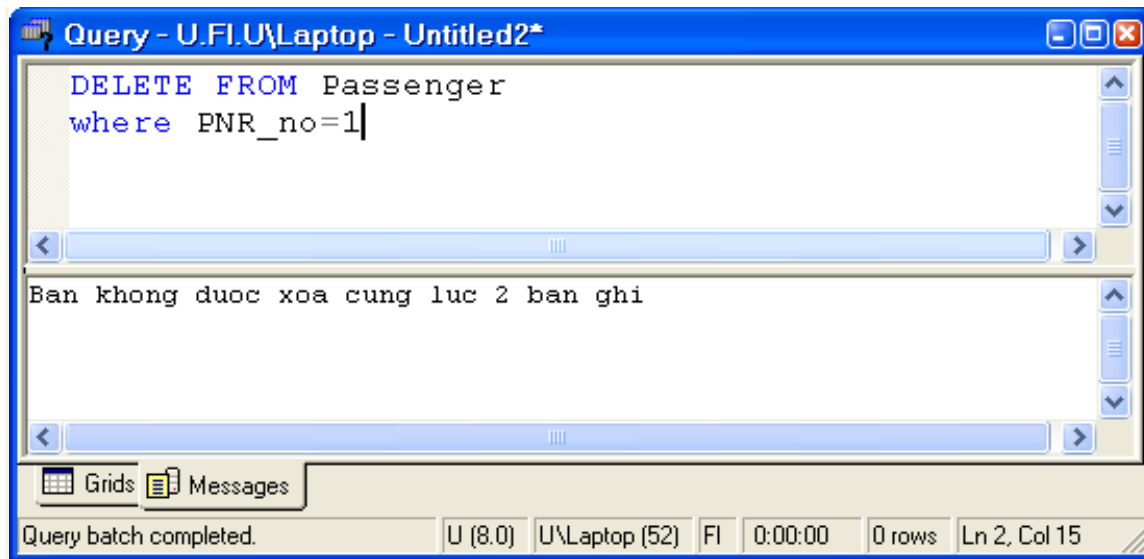
Xem xét ví dụ sau:

Tạo trigger để tránh xóa 2 bản ghi trong bảng Passenger đồng thời.

```
CREATE TRIGGER delete_trigg
ON Passenger
FOR Delete
AS
IF((Select count(*) From deleted)>2)
BEGIN
raiserror('Bạn không được xóa cùng lúc 2 bản ghi', 16,1)
ROLLBACK TRAN
END
```

2. Thực hiện câu lệnh xóa nhiều hơn 2 bản ghi từ bảng Passenger, giả sử như sau:

Kết quả:



## Tạo UPDATE Trigger.

### Tạo Table Level UPDATE Trigger.

Trigger UPDATE sẽ được thực hiện bất cứ khi nào dữ liệu trong bảng được cập nhật.

Xem xét ví dụ:

Tạo UPDATE trigger đảm bảo rằng cột No\_of\_seats trong bảng Reservation không được cập nhật giá trị lớn hơn 5 và journey\_date không nhỏ hơn ngày hiện tại.

1. Thực hiện như sau trong QA.

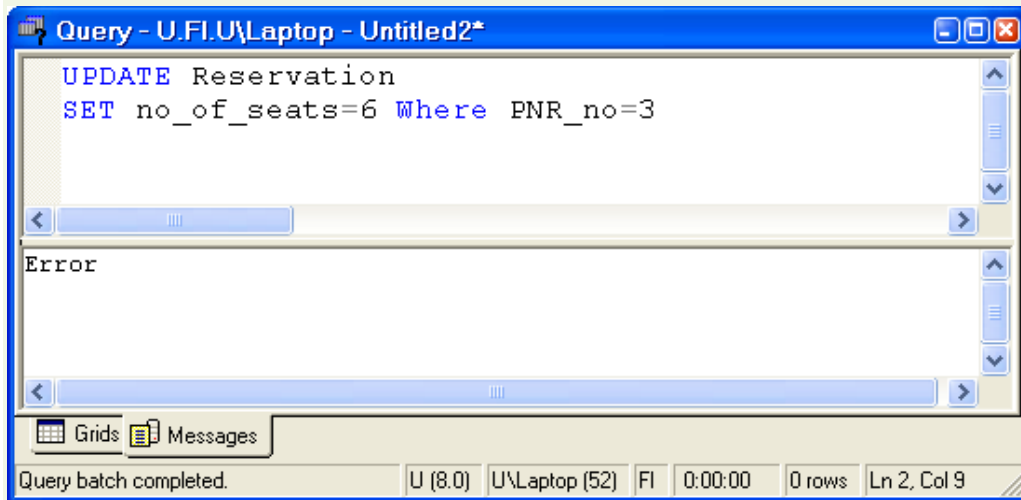
```
CREATE TRIGGER CheckingUpdate
ON Reservation
FOR UPDATE
AS
IF((Select no_of_seats From inserted)>5)
OR ((Select journey_date From Inserted)<getdate())
BEGIN
PRINT 'Error'
ROLLBACK TRAN
END
```

2. Thực hiện truy vấn sau để kiểm tra Trigger:

```
UPDATE Reservation
```

```
SET no_of_seats=6 Where PNR_no=3
```

Kết quả:



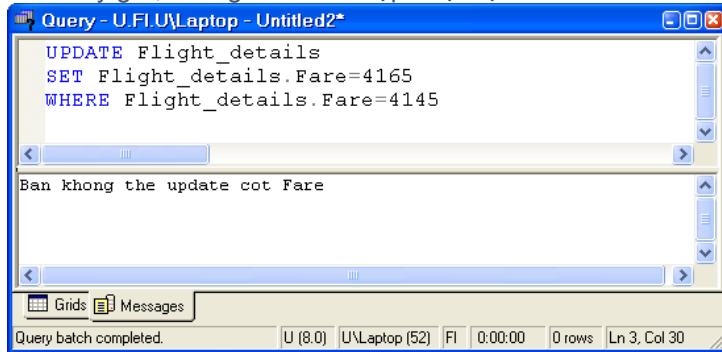
## Tạo Column Level Update Trigger

Loại Trigger được thực hiện khi dữ liệu trong cột nào đó được cập nhật.

- 1. Thực hiện như sau trong QA:

```
CREATE TRIGGER Col_Update_trig  
ON Flight_details  
FOR UPDATE  
AS  
IF UPDATE(Fare)  
BEGIN  
PRINT 'Ban khong the update cot Fare'  
ROLLBACK TRAN  
END
```

- 2. Bây giờ, chúng ta sẽ thử cập nhật cột Fare.



## Tạo Trigger có lựa chọn Encryption

Encryption (mã hoá) là phương pháp giữ bí mật cho Trigger. Nội dung của Trigger sau khi được mã hoá sẽ không đọc được.

1. Thực hiện câu lệnh ALTER TRIGGER để sửa insert\_trigg:

```
ALTER TRIGGER insert_trigg
ON Reservation
WITH ENCRYPTION
FOR INSERT
AS
IF((Select journey_date From Inserted)<getdate())
BEGIN
PRINT 'journey_date khong the nho hon ngay hien tai'
ROLLBACK TRAN
END
```

2. Thực hiện câu lệnh sau:

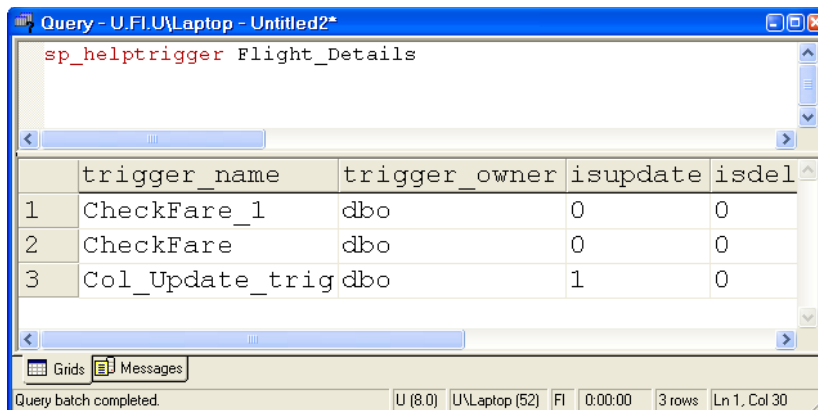


## Hiển thị danh sách các trigger trong Database

Sử dụng thủ tục hệ thống sp\_helptrigger để hiển thị danh sách các trigger trong cơ sở dữ liệu hiện tại.

```
sp_helptrigger Flight_Details
```

Kết quả:



## Sử dụng Triggers để tạo ràng buộc tham chiếu (Enforce Referential Intergrity)

Xem xét ví dụ sau:

Tạo Trigger để kiểm tra dữ liệu nhập vào cột Meal Pref của bảng Passenger phải là dữ liệu đã tồn tại trong cột Meal codes của bảng Meal.

```
CREATE TRIGGER ins_trig
ON Passenger
```

```

FOR INSERT
AS
IF (Select [Meal Pref]FROM INSERTED)
NOT IN (Select meal_code FROM Meal)
BEGIN
Print 'Ban khong the insert gia tri nay'
ROLLBACK TRAN
END

```

2. Bạn hãy thử kiểm tra hoạt động của Trigger trên.

## Cascade Delete sử dụng Nested trigger.

Trong Nested trigger, một trigger có thể được thực hiện lồng trong trigger khác. Chúng ta có thể lồng trigger tối đa 32 mức. Nested trigger cho phép cascade update và cascade delete.

Thực hiện câu lệnh để kích hoạt Nested trigger:

**sp\_configure 'nested trigger', 1**

Ngược lại:

**sp\_configure 'nested trigger', 0**

Tạo Cascade delete trigger để thực hiện công việc sau: Nếu xoá một chuyến bay trong bảng Flight, thì tất cả các thông tin liên quan trong bảng Flight\_Details sẽ bị xoá.

```

CREATE TRIGGER Casc_del
ON Flight
FOR DELETE
AS
DELETE Flight_details FROM Flight_details,
DELETED
WHERE Flight_details.aircraft_code=DELETED.aircraft_code

```

- 2.Tạo Delete Trigger khác trên bảng Flight. Trigger này sẽ thực hiện khi trigger Casc\_del thực hiện.

```

CREATE TRIGGER del_aircraftcode
ON Flight_details
FOR DELETE

```

AS

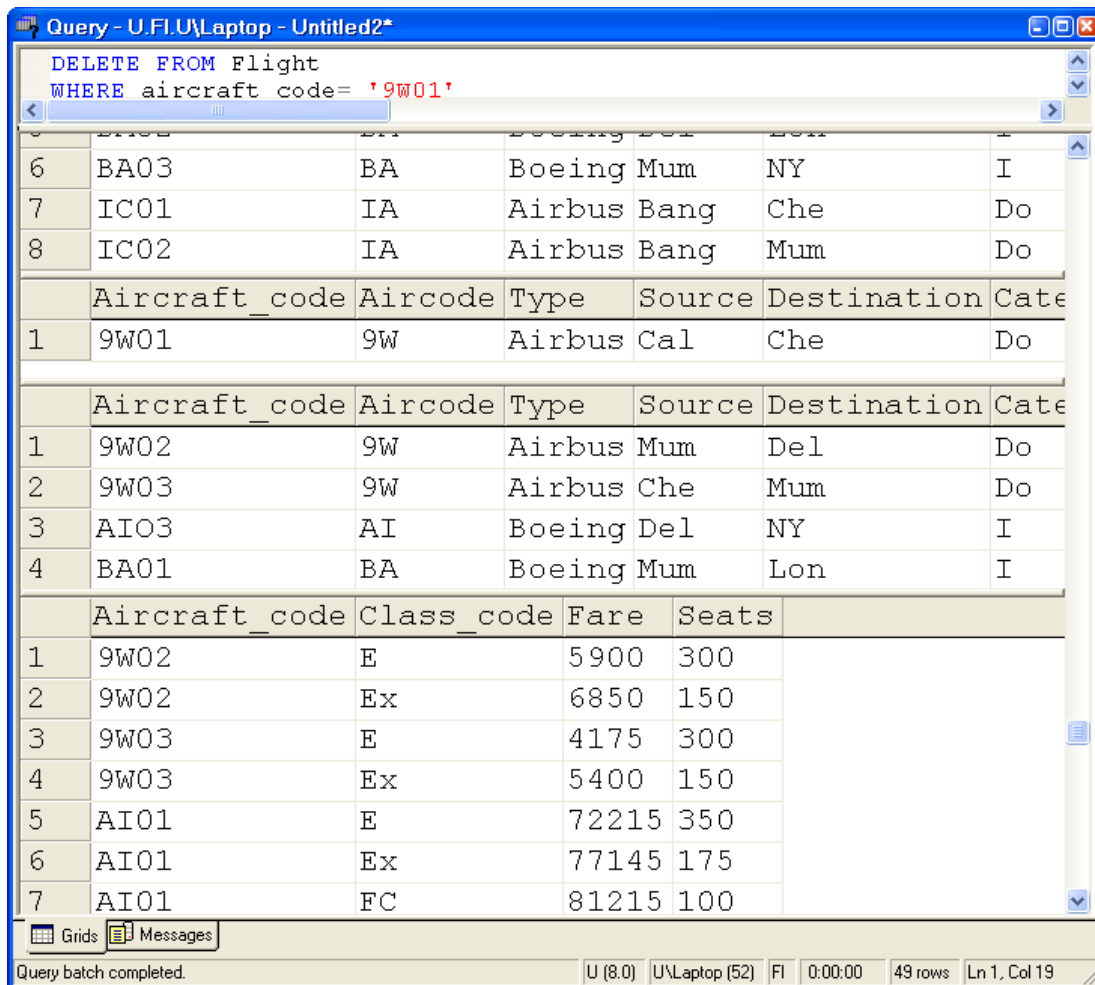
```
SELECT * FROM Flight
```

```
SELECT * FROM Flight_Details
```

- 3.Thực hiện câu lệnh sau:

```
DELETE FROM Flight
```

```
WHERE aircraft_code= '9W01'
```



The screenshot shows a SQL query window titled "Query - U.FI.U\Laptop - Untitled2\*". The query entered is:

```
DELETE FROM Flight
WHERE aircraft code= '9W01'
```

The results are displayed in two tables. The first table has 7 columns: Aircraft\_code, Aircode, Type, Source, Destination, and Cate. The second table has 5 columns: Aircraft\_code, Class\_code, Fare, Seats, and an empty column.

	Aircraft_code	Aircode	Type	Source	Destination	Cate
6	BA03	BA	Boeing	Mum	NY	I
7	IC01	IA	Airbus	Bang	Che	Do
8	IC02	IA	Airbus	Bang	Mum	Do

	Aircraft_code	Aircode	Type	Source	Destination	Cate
1	9W01	9W	Airbus	Cal	Che	Do

	Aircraft_code	Class_code	Fare	Seats	
1	9W02	E	5900	300	
2	9W02	Ex	6850	150	
3	9W03	E	4175	300	
4	9W03	Ex	5400	150	
5	AI01	E	72215	350	
6	AI01	Ex	77145	175	
7	AI01	FC	81215	100	

The status bar at the bottom indicates "Query batch completed." and "U (8.0) U\Laptop (52) FI 0:00:00 49 rows Ln 1, Col 19".

## Tạo INSTEAD OF Trigger

Chúng ta có thể thực hiện INSTEAD OF trigger trên bảng, nó thay thế cho câu lệnh INSERT, UPDATE, DELETE nguyên thủy.

- 1.Thực hiện như sau:

```
CREATE TRIGGER instead_trigg
```

```
ON Service
```



```
INSTEAD OF INSERT
```

```
AS
```

```
BEGIN
```

```
Select Service_code AS 'Inserted columns' From Inserted
```

```
Select Service_code AS 'Deleted columns' From Deleted
```

```
Select Service_code AS 'Table contents' From Service
```

```
END
```

- 2.Thực hiện câu lệnh sau:

```
INSERT INTO Service
```

```
Values('FA', 'First Aid')
```

Kết quả:

The screenshot shows a SQL query window titled "Query - U.FI.U\Laptop - Untitled1\*". The query entered is:

```
INSERT INTO Service  
Values('FA', 'First Aid')
```

The results are displayed in three sections:

- Inserted columns:** A table with one row containing the value "FA".
- Deleted columns:** An empty table.
- Table contents:** A table with four rows: "AA", "CC", "FA", and "WC".

The status bar at the bottom indicates "Query batch completed." and shows the current position as "Ln 1, Col 1".

## BÀI TẬP

### Bài tập 1:

Giả sử cần quản lý một cửa hàng bán sách. Mỗi cuốn sách phân biệt với nhau thông qua mã sách, mã sách xác định các thông tin: tên sách, tác giả, đơn giá, số lượng tồn. Mỗi cuốn sách chỉ thuộc về một nhóm sách. Một nhóm sách có thể có nhiều cuốn sách hoặc không có cuốn nào. Thông tin về nhóm sách: mã nhóm để phân biệt nhóm sách này với nhóm sách khác, tên nhóm. Khi có sách được bán, nhân viên lập hóa đơn để lưu trữ thông tin bán sách. Mỗi hóa đơn có một số hóa đơn duy nhất xác định nhân viên lập hóa đơn và ngày lập hóa đơn. Mỗi nhân viên có một mã nhân viên duy nhất xác định các thông tin như họ lót, tên, phái, ngày sinh, địa chỉ. Mỗi hóa đơn có thể có một hoặc nhiều cuốn sách, mỗi cuốn sách có thể mua với số lượng bất kỳ.

+ `NhomSach(MaNhom char(5), TenNhom nvarchar(25))`

+ `NhanVien(MaNV char(5), HoLot nvarchar(25), TenNV nvarchar(10), Phai nvarchar(3), NgaySinh Smalldatetime, DiaChi nvarchar(40))`

+ `DanhMucSach(MaSach char(5), TenSach nvarchar(40), TacGia nvarchar(20), MaNhom char(5), DonGia Numeric(5), SLTon numeric(5))`

+ `HoaDon(MaHD char(5), NgayBan SmallDatetime, MaNV char(5))`

+ `ChiTietHoaDon(MaHD char(5), MaSach char(5), SoLuong numeric(5))`

1. Viết trigger cho thao tác Insert của bảng NHOMSACH. Khi có thao tác chèn vào bảng nhóm sách thì đưa ra một thông báo là 'Có <n> mẫu tin được chèn'.
2. Viết trigger cho thao tác Insert trên bảng HOADON. Sau khi có mẫu tin được chèn vào bảng HOADON thì mẫu tin đó cũng được chèn vào bảng HOADON\_Luu. Lưu ý: nếu chưa có bảng HOADON\_Luu thì tạo HOADON\_Luu (có cấu trúc hoàn toàn giống như bảng HOADON) trước khi kiểm chứng trigger.
3. Viết trigger cho thao tác Insert, Update, Delete trên bảng CHITIETHOADON. Khi có mẫu tin được chèn vào hoặc hiệu chỉnh hoặc xóa thì cập nhật lại cột TongTriGia trong bảng HOADON với  $TongTriGia = Tổng tiền của Số lượng * Đơn giá$ . Lưu ý: nếu bảng HOADON chưa có cột TongTriGia thì bổ sung vào trước khi kiểm chứng trigger.

4. Viết trigger cho thao tác Insert, Update để kiểm tra ràng buộc liên thuộc tính liên quan hệ giữa GIABAN trong CHITIETHOADON và DONGIA trong bảng mặt hàng như sau: GIABAN trong CHITIETHOADON luôn luôn bằng DONGIA trong DANHMUCSACH, nếu vi phạm thì thông báo và không cho phép Insert hay Update.
5. Có ràng buộc liên thuộc tính là ngày bán của một HOADON thì luôn luôn lớn hơn hay bằng ngày lập hóa đơn . Hãy bắt ràng buộc trên khi cần thiết.

## Bài tập 2: Cho CSDL QL Thư viện

**DocGia** (ma\_DocGia, ho, tenlot, ten, ngaysinh)

**Nguoilon** (ma\_DocGia, sonha, duong, quan, dienthoai, han\_sd)

**Treem** (ma\_DocGia, ma\_DocGia\_nguoilon)

**Tuasach** (ma\_tuasach, tuasach, tacgia, tomtat)

**Dausach** (isbn, ma\_tuasach, ngonngu, bìa, trangthai)

**Cuonsach** (isbn, ma\_cuonsach, tinhtrang)

Tạo một số Trigger như sau trong CSDL **Thư viện**:

### 1. tg\_delMuon:

Nội dung: Cập nhật tình trạng của cuốn sách là yes.

### 2. tg\_insMuon:

Nội dung: Cập nhật tình trạng của cuốn sách là no.

### 3. tg\_updCuonSach:

Nội dung: Khi thuộc tính tình trạng trên bảng cuốn sách được cập nhật thì trạng thái của đầu sách cũng được cập nhật theo. Cài đặt các thủ tục sau cho CSDL Quản lý thư viện.

### 4. tg\_InfThongBao

Nội dung: Viết trigger khi thêm mới, sửa tên tác giả, thêm/sửa một tựa sách thì in ra câu thông báo bằng Tiếng Việt ‘*Đã thêm mới tựa sách*’.

Gợi ý :

Kiểm tra trigger đã tạo bằng khối lệnh để dữ liệu không bị thay đổi :

*begin tran*

*--khối lệnh thêm,xóa,sửa*

*rollback*

### 5. tg\_SuraSach

Nội dung: Viết trigger khi sửa tên tác giả cho một (hoặc nhiều) tựa sách thì in ra:

- Danh sách mã các tựa sách vừa được sửa.

- Danh sách mã tựa sách vừa được sửa và tên tác giả mới.
- Danh sách mã tựa sách vừa được sửa và tên tác giả cũ.
- Danh sách mã tựa sách vừa được sửa cùng tên tác giả cũ và tác giả mới.

Gợi ý :

- + Câu lệnh insert into T select... from ... cho phép insert cùng lúc nhiều dòng.
- + Dùng bảng Inserted (hoặc/và Deleted).
- Câu thông báo bằng Tiếng Việt '*Vừa sửa thông tin của tựa sách có mã số xx*' với xx là mã tựa sách vừa được sửa.

Gợi ý : Sách được sửa trong bảng Inserted (hoặc Deleted).

## **6. tg\_KiemTraTrung**

Nội dung: Viết trigger khi Khi thêm mới một tựa sách thì kiểm tra xem đã có tựa sách trùng tên với tựa sách vừa được thêm hay không. Xử lý 2 trường hợp :

- Trường hợp xử lý 1: chỉ thông báo vẫn cho insert
- Trường hợp xử lý 2: thông báo và không cho insert