ĐẠI NAM
UNIVERSITY

CHƯƠNG 6

TRIGGER

NỘI DUNG

- Giới thiệu Trigger
- Định nghĩa Trigger
- Trigger AFTER
- Trigger INSTEAD OF
- Bảng “ảo” INSERTED, DELETED
- Ứng dụng Trigger

Giới thiệu Trigger

- Trigger là một dạng đặc biệt của thủ tục lưu trữ, chứa các lệnh T-SQL nhằm thực hiện một số hành động nào đó do người lập trình viết
- Khác với thủ tục lưu:
  - Trigger không có tham số và giá trị trả về
  - Không gọi bằng lệnh EXEC **mà tự động** kích hoạt khi dữ liệu trên bảng có liên quan đến trigger được cập nhật

Công dụng trigger

- Kiểm tra rang buộc toàn vẹn dữ liệu phức tạp
- THực hiện các xử lý thiết kế thi hành tại server (trong mô hình client/Server). Các xử lý sẽ tự động thực hiện khi có thao tác INSERT, UPDATE, DELETE xảy ra.
- Trigger dung thay thế các constrant khi muốn kiểm tra ràng buộc dữ liệu kèm theo các câu thông báo thích hợp theo ý người dùng

Ràng buộc dữ liệu toàn vẹn với Trigger

- Để đảm bảo dữ liệu nhất quán và đúng đắn, ta cần kiểm tra thực hiện 3 thao tác: INSERT, UPDATE, DELETE
- Có 2 cách kiểm tra:

- Kiểm tra mức giao diện: là công việc lập trình trên các màn hình giao diện
- Kiểm tra mức CASL: thực hiện bởi các đối tượng constraint hoặc trigger

Ràng buộc dữ liệu toàn vẹn với Trigger

- ❑ Đối tượng constraint kiểm tra
  - ✓ Kiểm tra miền giá trị
  - ✓ Kiểm tra các ràng buộc giữa các thuộc tính trên 1 bảng dữ liệu
- ❑ Đối tượng trigger
  - ✓ - Kiểm tra tính toàn vien dữ liệu trên nhiều cột hoặc nhiều dòng của các bảng khác nhau

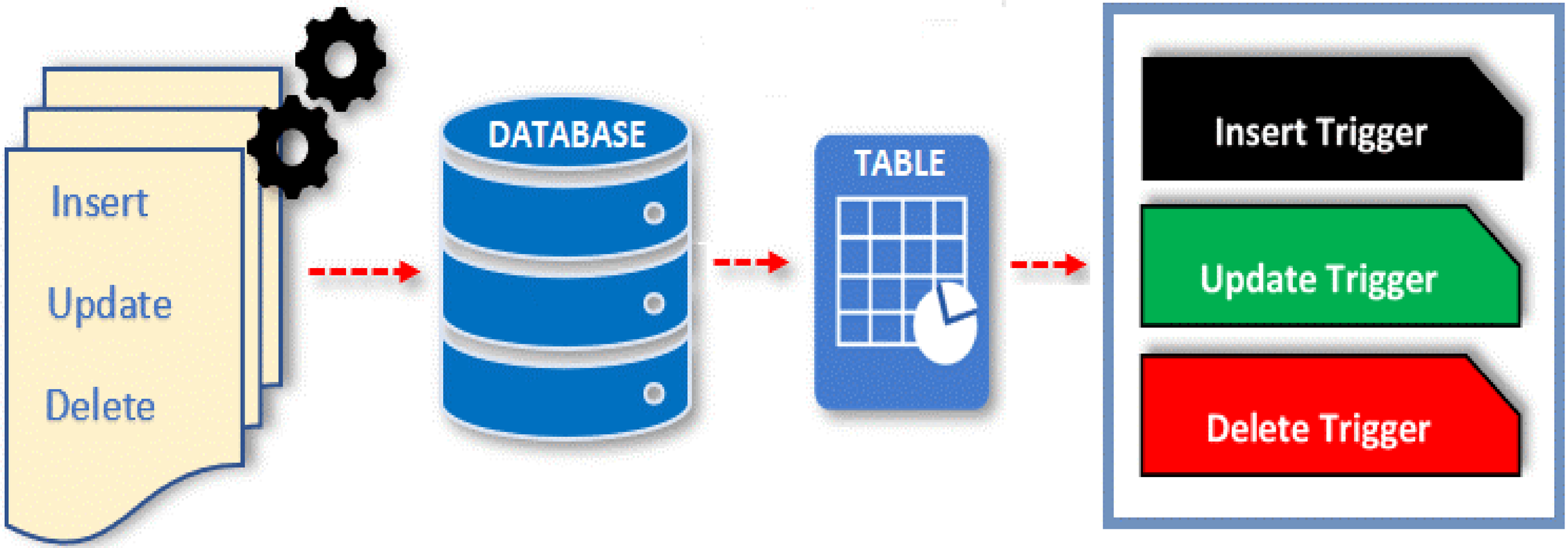
KHẢ NĂNG CỦA TRIGGER

- Một trigger có thể **nhận biết**, **ngăn chặn** và **huỷ bỏ** những thao tác làm thay đổi trái phép dữ liệu trong cơ sở dữ liệu.
- Các thao tác trên dữ liệu (xoá, cập nhật và bổ sung) có thể được trigger **phát hiện ra** và **tự động** thực hiện một loạt các thao tác khác trên cơ sở dữ liệu nhằm đảm bảo tính hợp lệ của dữ liệu.
- Có **thể tạo** và **kiểm tra** được những mối quan hệ phức tạp hơn giữa các bảng trong cơ sở dữ liệu mà bản thân các ràng buộc không thể thực hiện được.

ĐẶC ĐIỂM CỦA TRIGGER

- Một trigger có thể thực hiện nhiều công việc (theo kịch bản), có thể nhiều sự kiện kích hoạt thực thi trigger, có thể tách rời các sự kiện trong một trigger.
- Trigger không được tạo trên bảng **temprate** hay **system**.
- Trigger chỉ thực thi tự động thông qua các sự kiện mà không thực hiện bằng tay.
- Trigger sử dụng được với khung nhìn.

Hoạt động của Trigger



Trigger AFTER và Trigger INSTEAD OF

After (for) Trigger	Instead of Trigger
•Dùng cho việc cập nhật bảng •Chạy sau các hành động kiểm tra dữ liệu của các Constraint •Dữ liệu đã bị tạm thời thay đổi trong bảng. • Chỉ áp dụng cho bảng	•Dùng cho việc cập nhật bảng hoặc bảng ảo •Chạy trước các hành động kiểm tra dữ liệu •Dữ liệu chưa bị thay đổi •Có thể thay thế hành động cập nhật dữ liệu bằng các hành động khác •Có thể áp dụng cho bảng hoặc bảng ảo •Thường dùng cho việc cập nhật bảng ảo

BẢNG ẢO INSERTED VÀ DELETED

- Khi xảy ra một sự kiện thao tác dữ liệu, một bản ghi trong CSDL sẽ lưu ra một bản ghi trong Trigger có tên là **inserted** hoặc **Deleted**
- ***Bảng Inseted và Deleted có cấu trúc giống hệt bảng đang xử lý***

BẢNG ẢO INSERTED VÀ DELETED

Bảng Inserted sử dụng cho lệnh Insert	Bảng Deleted sử dụng cho lệnh Delete	Bảng Inserted và Deleted sử dụng cho lệnh Update
•Chứa dữ liệu được thêm mới trong hành động Insert/Delete •Chỉ có tại thời điểm xảy ra Trigger •Cấu trúc bảng giống với bảng của Trigger	•Chứa dữ liệu bị xoá trong hành động Delete/ Update •Chỉ có tại thời điểm xảy ra Trigger •Cấu trúc bảng giống với bảng của Trigger	•Bảng Inserted chứa các dữ liệu mới (đã được cập nhật) •Bảng Deleted chứa các dữ liệu cũ (trước khi cập nhật) •Update = Insert mới và Delete cũ

TẠO TRIGGER

❑ **Cú pháp:**

Create Trigger <Ten_trigger>
ON Ten_Bang | View
[**WITH ENCRYPTION**]
{For | After | Instead of } {[**INSERT**] [, **UPDATE**] [, **DELETE**]}
AS
Begin
Các-câu-lệnh-của-trigger
End

TẠO TRIGGER

- Trong đó:
  - **Tên bảng/view**: là bảng/view mà trigger được tạo
    - Từ khóa AFTER không hỗ trợ VIEW
  - **With Encryption:** ngăn ngừa việc sửa đổi nội dung Trigger. Khi sử dụng Alter trigger thì with Encryption không hỗ trợ
  - **Insert, Update, Delete**: là tên các biến cố xảy ra khi thực hiện thao tác tương ứng trên bảng
  - **Instead of** thì trigger sẽ được kích hoạt trước khi dữ liệu đã cập nhật vào bảng, thường được dùng để kiểm tra dữ liệu cập nhật trên View
  - **For (After)** thì trigger sẽ được kích hoạt sau khi dữ liệu đã cập nhật vào bảng

Câu lệnh không được thực thi

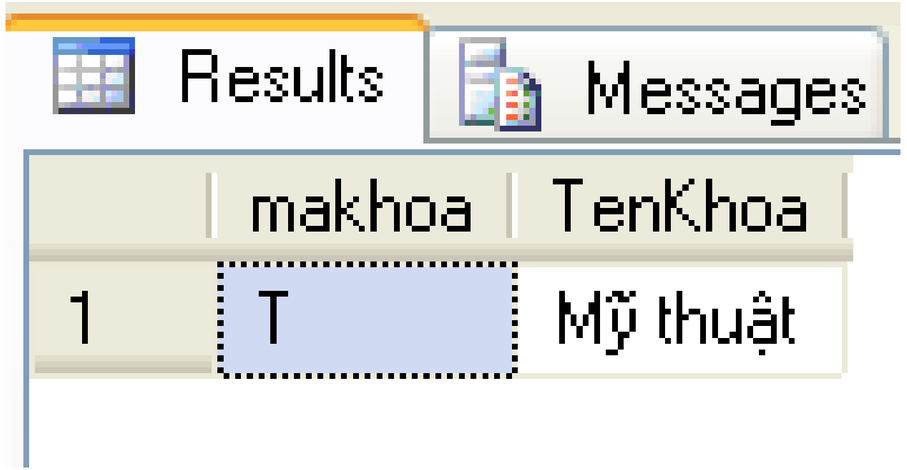
- Các câu lệnh không được thực thi trong kịch bản của câu lệnh xử lý của Trigger:

ALTER DATABASE sửa	**CREATE DATABASE**	**DISK INIT**
DISK RESIZE tăng ổ đĩa	**DROP DATABASE**	**LOAD DATABASE**
LOAD LOG	**RECONFIGURE**	**RESTORE DATABASE**
RESTORE LOG		

VÍ DỤ

```
create trigger them_khoa
on khoa
after insert
as
    select * from inserted

--áp dụng
insert into khoa(makhoa,tenkhoa) values(‘T',N'Mỹ
    thuật')
```



VÍ DỤ

Ví dụ: Tạo trigger tự động tăng sĩ số nhân viên ở bảng Phòng ban khi có một nhân viên được thêm vào phòng ban đó

Yêu cầu:

- Lấy mã phòng ban của nhân viên mới
- Kiểm tra mã phòng ban của nhân viên có tồn tại trong bảng Phòng ban hay không?
- Nếu mã phòng ban không tồn tại thì thông báo và không cho thêm
- Ngược lại sẽ tăng sĩ số của phòng ban lên 1 đơn vị

VÍ DỤ

1. Tạo trigger cho phép nhập thời gian của mỗi nhân viên tham gia dự án từ 5 đến 10
2. Trigger kiểm tra một Phòng ban có tồn tại hay không trước khi thêm cho một NV
3. Xóa một NV thì xóa các phân công
4. Xóa PB thì các ĐĐ, NV và DA xóa theo
5. Kiểm soát ngày làm cho dự án của nhân viên phải sau ngày bắt đầu dự án đó

- *NHANVIEN (MANV, HOTen, NS, GT, HSL, DC, MADV, NgayVL)*
- *PHONGBAN (MADV, TENDV , MaTP, NgayBD, **SoNV**)*
- *DIADIEM_PB (MaDV, DiaDiem)*
- *DUAN (MADA, TENDA, DIADIEM, **NgBD**, MADV)*
- *PHANCONG (MADA, MANV, SoGio, **NgLamDA**)*

Trigger sử dụng mệnh đề IF UPDATE

- Thay vì chỉ định một trigger được kích hoạt trên một bảng ta có thể chỉ định trigger được kích hoạt và thực hiện những thao tác cụ thể khi việc thay đổi dữ liệu chỉ liên quan đến **một số cột nhất định** nào đó của bảng.

  Khi đó sử dụng mệnh đề IF UPDATE trong trigger

  – *Giới hạn cột nào được phép cập nhật*
- IF UPDATE không sử dụng được đối với câu lệnh DELETE

  – *Chỉ sử dụng khi INSERT hoặc UPDATE*

Trigger sử dụng mệnh đề IF UPDATE

- Cú pháp:

CREATE TRIGGER tên_trigger

ON tên_bảng

FOR { [INSERT] [,] [UPDATE] }

AS

IF UPDATE (tên_cột)

<các_câu_lệnh_của_trigger >

VD: trigger không cho phép sửa MNV

```
CREATE TRIGGER UpdateMaNV
ON NHANVIEN
For Update
As
Begin
   If update(MaNV)
   Begin
     Print N'Không thể thay đổi Mã NV'
     RollBack transaction --không lưu lại các thay đổi
   end
End
```

VD: Kiểm soát ngày vào làm phải sau ngày sinh

```
CREATE TRIGGER CHECK_NGAYVL --Tên Trigger
ON NHANVIEN
FOR UPDATE,INSERT
AS
    IF UPDATE(NGAYVL) --Kiểm tra việc cập nhật trên cột
    BEGIN
        DECLARE @NS SMALLDATETIME, @NGVL SMALLDATETIME
    SELECT @NGVL = NGAYVL FROM INSERTED
    SELECT @NS = NGAYSINH FROM INSERTED
    IF( @NS > @NGVL )
            BEGIN
                PRINT N'Ngày vào Làm phải sau ngày sinh'
                ROLLBACK TRAN
            -- Câu lệnh quay lui khi thực hiện biến cố không thành công
            END
END
```

VẬN DỤNG

- Kiểm soát ngày tham gia DA sau ngày BĐ Dự án
- Tạo Trigger Update MaPhong trên bảng PHÒNG BAN, nếu MÃ PHÒNG của bảng PHÒNG BAN bị sửa thì
  - trường MA PHONG của bảng NHAN VIEN cũng thay đổi theo.
  - Trường MÃ PHÒNG của bảng ĐỊA ĐIỂM– PHÒNG BAN cũng thay đổi theo
  - Trường MÃ PHÒNG của bảng DỰ ÁN cũng thay đổi theo

Kiểm soát ngày tham gia DA sau ngày BĐ Dự án?

```
CREATE  TRIGGER  CHECK_NGAY_THAM_GIA_DA
ON PHANCONG
AFTER UPDATE, INSERT
AS
    IF UPDATE(NgayThamGia) --Kiểm tra việc cập nhật trên cột
    BEGIN
    declare @NgTG datetime, @NgBD datetime
    SET @NgTG = (SELECT NgayThamGia FROM INSERTED)
    SET @NgBD = (SELECT NgayBD FROM DEAN A,INSERTED B
                        WHERE A.MaDA = B.MaDA)
    IF( @NgTG < @NgBD )
    BEGIN
       raiserror(N'Ngày Tham gia phải sau ngày Bắt đầu',16,1)
       ROLLBACK TRAN
              -- Câu lệnh quay lui khi thực hiện biến cố không thành công
    END
END
```

VD: Cập nhật mã phòng ban

```
CREATE TRIGGER UpdateMaPhong
ON PHONGBAN
FOR UPDATE
AS
BEGIN
   IF UPDATE( MaPhong ) --Nếu cột mã phòng sửa
   BEGIN
     DECLARE @MaPhgCu nvarchar(9), @MaPhgMoi nvarchar(9)
     SELECT @MaPhgCu =  MaPhong FROM DELETED
     SELECT @MaPhgMoi =   MaPhong FROM INSERTED
     If ( EXISTS (select MaPhong from NHANVIEN Where MaPhong = @MaPhgCu))
       UPDATE NHANVIEN SET MaPhong = @MaPhgMoi
       WHERE MaPhong = @MaPhgCu
     If ( EXISTS ( select MaPhong from DiaDiem_Phong Where MaPhong = @MaPhgCu))
       UPDATE DiaDiem_Phong set MaPhong = @MaPhgMoi
       WHERE MaPhong = @MaPhgCu
    END
END
```

SỬA, XÓA TRIGGER

- Sửa:

  ALTER TRIGGER <tên_trigger>
- Xóa:

  DROP TRIGGER <tên_trigger>
- Làm mất hiệu lực của trigger:

ALTER TABLE <tên_bảng> DISABLE TRIGGER <tên_trigger>

- Làm trigger có hiệu lực:

ALTER TABLE <tên_bảng> ENABLE TRIGGER
<tên_trigger>/[ALL]

SỬA, XÓA TRIGGER

Khi thêm mới mẫu tin :

- Trigger của sự kiện này sẽ tự động kích hoạt khi có một bản ghi được thêm vào bảng dữ liệu. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như:
  - Khóa ngoại.
  - Miền giá trị.
  - Các thuộc tính liên quan trong cùng một bảng.
  - Các thuộc tính liên quan của nhiều bảng khác nhau.

SỬA, XÓA TRIGGER

Khi hủy bỏ mẫu tin:

- Trigger của sự kiện này sẽ tự động kích hoạt khi dữ liệu trong bảng bị xóa. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như là :
  - Khóa ngoại để xóa tự động các dữ liệu bên bảng nhiều có liên quan hoặc thông báo cho người dùng

SỬA, XÓA TRIGGER

Khi sửa mẫu tin :

- Trigger của sự kiện này sẽ tự động kích hoạt khi dữ liệu trong bảng bị sửa đổi. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như là :
  - Khóa ngoại.
  - Miền giá trị.
  - Các thuộc tính liên quan trong cùng một bảng.
  - Các thuộc tính liên quan của nhiều bảng khác nhau.
  - Thường chỉ cho phép sửa đổi trên một số cột nhất định.

Bài tập áp dụng

- Tạo trigger cho phép giảm sĩ số của lớp khi có một sinh viên bị xóa.
- Viết trigger để kiểm tra mã sv có tồn tại trong bảng Sinh viên khi thực hiện thêm dữ liệu vào bảng Điểm.
- Viết trigger kiểm soát việc thêm dữ liệu vào bảng Diem với trường Kết quả có thỏa mãn nằm trong khoảng [0,10]
- Viết trigger xóa lớp: nếu lớp không có sinh viên thì cho phép xóa, ngược lại thông báo không được xóa.

TRANSACTION – GIAO TÁC

- Giao tác là các hành động cập nhật dữ liệu trên nhiều bảng khác nhau được thực hiện trong cùng một khối.
- *Là tập hợp các lệnh sẽ được thực hiện nếu tất cả đều thành công, nếu có một lệnh thất bại ,thì sẽ không có lệnh nào được thực hiện*
- Ví dụ: Một khách hàng có cùng lúc 2 tài khoản trong ngân hàng: *thanh toán và tiết kiệm*.
  - Tài khoản thanh toán dùng để thực hiện các giao dịch nộp hoặc chuyển khoản của khách hàng với đối tác
  - Tài khoản tiết kiệm dùng để gửi tiền tiết kiệm lấy lãi cuối kì theo kì hạn 3 tháng

Tại sao phải dùng Transaction?

- Dùng khái niệm giao tác khi xử lý các vấn đề liên quan đến truy xuất dữ liệu đồng thời
- Có những xử lý trên CSDL được thực hiện bằng nhiều thao tác liên tiếp nhau, tập hợp các thao tác này phải được xem là một thao tác nguyên tố để đảm bảo tính nhất quán của dữ liệu sau khi thực hiện, nghĩa là, hoặc tất cả được thực hiện thành công, hoặc không có thao tác nào được thực hiện --> tập hợp các thao tác này được viết thành một transaction.

VÍ DỤ

- Stored procedure thực hiện việc thêm một sinh viên vào lớp

--Bước 1

insert into sinhvien (masv, hodem, ten, malop)

values (‘S001’, N’Nguyễn Văn’, N’Sơn’,’L01’)

--Bước 2

Update Lop

Set SiSo = SiSo +1

Nếu bước 2 của stored proc thực hiện không thành công thì dữ liệu trong CSDL có còn nhất quán không?

CÁC THAO TÁC THỰC HIỆN

- **BEGIN TRANSACTION** <tên giao tác>: Bắt đầu một giao tác
- **SAVE TRANSACTION** <tên điểm đánh dấu>: Đánh dấu một vị trí trong giao tác (gọi là điểm đánh dấu).
- **ROLLBACK [TRANSACTION** <tên giao tác>]: Quay lui trở lại đầu giao tác hoặc một điểm đánh dấu nào đó trong giao tác.
- **COMMIT [TRANSACTION** <tên giao tác>] : Đánh dấu điểm kết thúc một giao tác.

Cú pháp xây dựng Transaction

```
Begin Tran
    < tập các lệnh>
    If  @@error<>0
        begin
          print ‘giao tac that bai’
          Rollback  tran
        end
    Else
       Commit tran
```

VÍ DỤ

- **Ví dụ 1:**

BEGIN TRANSACTION giaotac1

UPDATE monhoc SET sodvht = 4 WHERE sodvht = 3

UPDATE lop SET malop = 2 WHERE tenlop = ‘07B1’

COMMIT TRANSACTION giaotac1

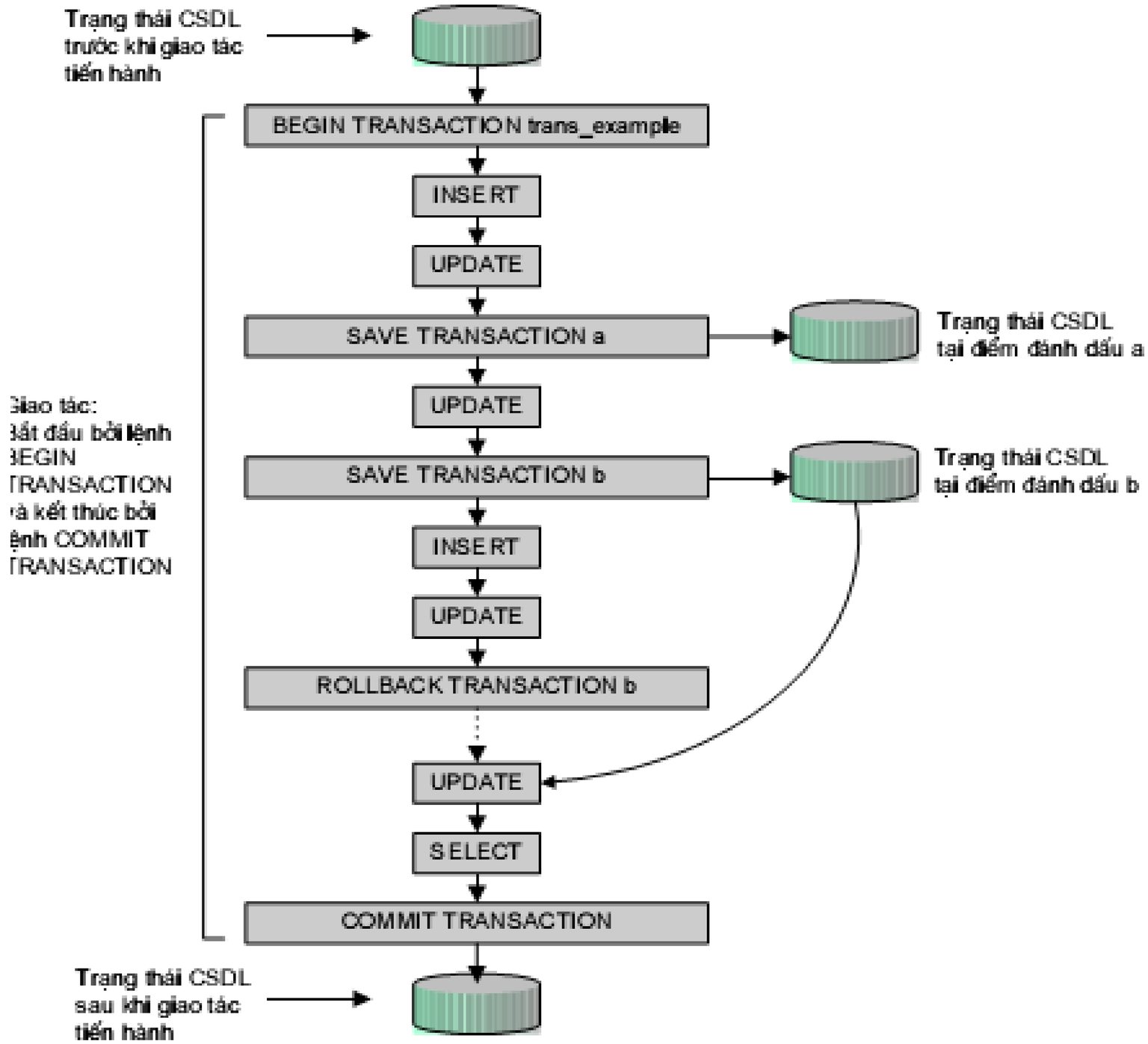
- **Ví dụ 2:**

BEGIN TRANSACTION giaotac1

UPDATE monhoc SET sodvht = 4 WHERE sodvht = 3

UPDATE lop SET malop = 2 WHERE tenlop = ‘07B1’

ROLLBACK TRANSACTION giaotac1


Trạng thái CSDL
trước khi giao tác
tiến hành
BEGIN TRANSACTION trans_example
INSERT
UPDATE
SAVE TRANSACTION a
Trạng thái CSDL
tại điểm đánh dấu a
UPDATE
SAVE TRANSACTION b
Trạng thái CSDL
tại điểm đánh dấu b
INSERT
UPDATE
ROLLBACK TRANSACTION b
UPDATE
SELECT
COMMIT TRANSACTION
Trạng thái CSDL
sau khi giao tác
tiến hành

VÍ DỤ

- **Ví dụ 3:**

BEGIN TRANSACTION giaotac3

UPDATE monhoc SET sodvht=4 WHERE sodvht=3

SAVE TRANSACTION a

UPDATE lop SET malop=2 WHERE tenlop=’07B1’

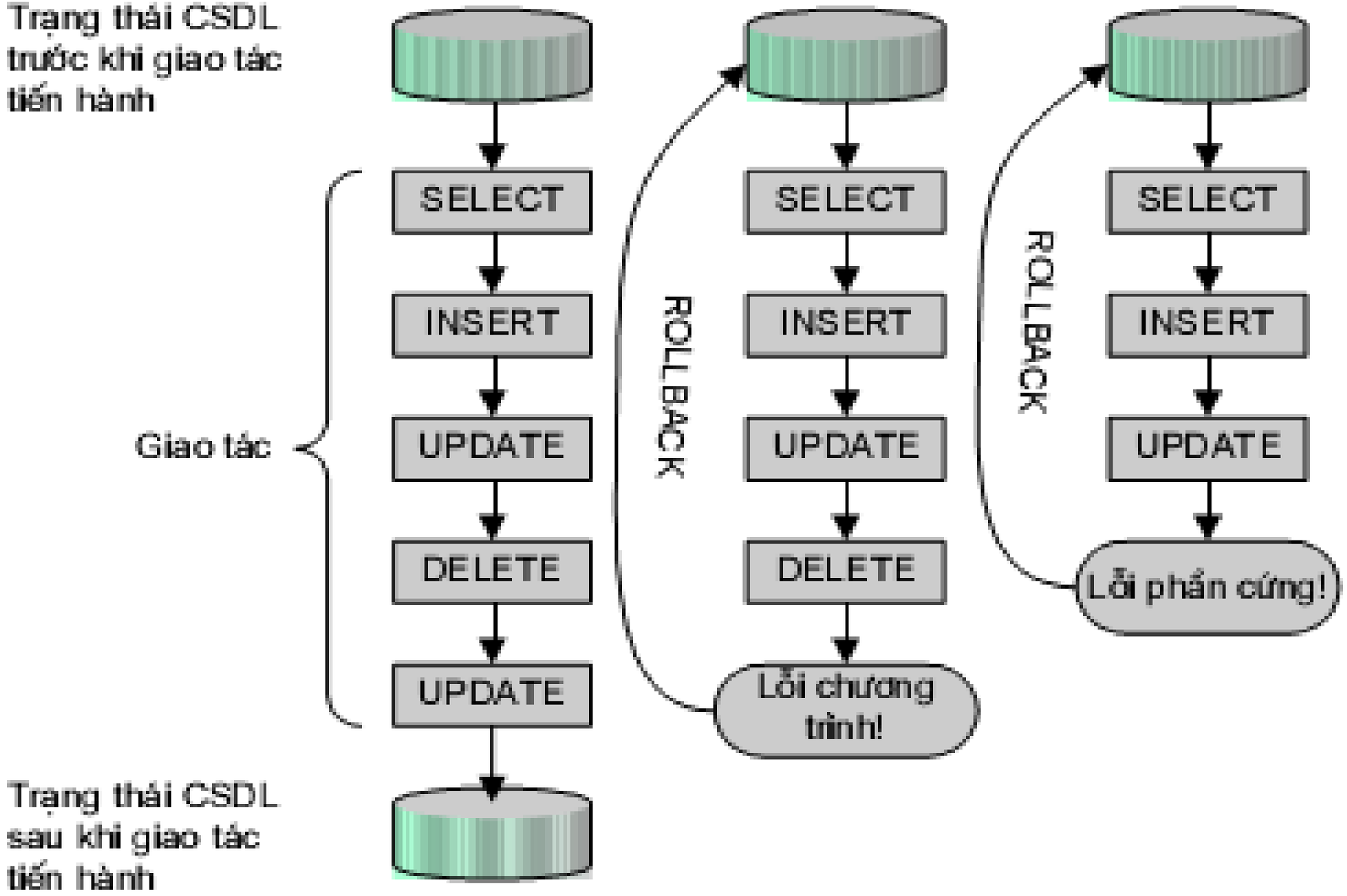
ROLLBACK TRANSACTION a

UPDATE monhoc SET sodvht=3 WHERE sodvht=2

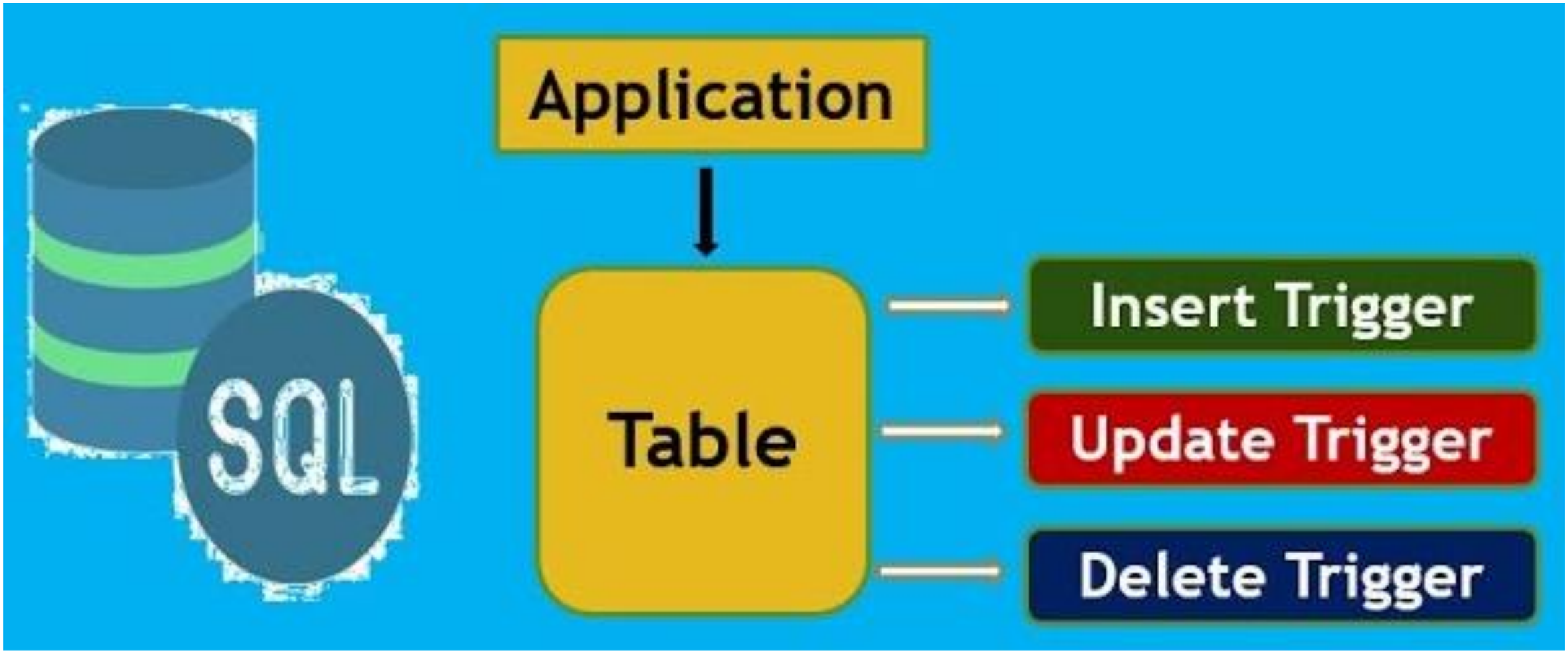
COMMIT TRANSACTION giaotac3

ROLLBACK TRANSACTION giaotac3

Sơ đồ thực hiện giao tác



ỨNG DỤNG TRIGGER



THỰC HÀNH


Q&A

BÀI TẬP VỀ NHÀ

- ✓ Câu hỏi trắc nghiệm
- ✓ Bài tập