# BẪY LỖI (TRIGGER)

Sau khi học xong bài này, sinh viên có thể:

* *Nắm được cơ chế hoạt động của một Trigger.*
* *Các biến cố tác động lên một table. Liên kết hoạt động giữa 2 hay nhiều table có mối kết hợp với nhau.*
* *Hiểu rõ tính ràng buộc toàn vẹn của CSDL, từ đó có thể xây dựng và vận hành các trigger nhằm đảm bảo sự toàn vẹn dữ liệu và an toàn cho hệ thống.*

## KHÁI NIỆM

Trigger là một Stored procedure đặc biệt tự động được chạy mỗi khi có một hành động nào đó xảy ra có liên quan đến nó. Các hoạt động xảy ra giúp trigger hoạt động: Insert, Delete, Update (DML). TRIGGER không được thực thi một cách tường minh nên cần thận trọng khi dùng trigger.

Trigger hoạt động gần giống các hàm bắt sự kiện trong javascript. Nó chỉ được kích hoạt khi xảy ra một hành động mà cụ thể là các thao tác Insert, Delete, Update trên một table. Nó hoạt động một các thụ động nên chúng ta không thể biết trigger thực thi khi nào.

## SỬ DỤNG TRIGGER

Sử dụng trigger nhằm mục đích:

* Đảm bảo tính ràng buộc toàn vẹn cho CSDL.
* Kiểm soát dữ liệu hiện có trong CSDL khi có thay đổi giá trị của một mẩu tin trong bảng.
* Kiểm tra dữ liệu mới nhập vào có thỏa mãn điều kiện không.
* Kiểm chứng khi xóa mẩu tin trong bảng.
* Tự động cập nhật dữ liệu cho bảng B khi dữ liệu bảng A thay đổi (khi 2 bảng có mối quan hệ với nhau).

## NGUYÊN TẮC HOẠT ĐỘNG

* Triggers được thực hiện tự động sau khi lệnh **INSERT, UPDATE, hoặc DELETE** được thực hiện trên một table mà trigger đó được định nghĩa. Còn các **constraints** và INSTEAD OF trigger sẽ được kiểm tra trước khi lệnh INSERT, UPDATE, hoặc DELETE thực hiện.
* Constraints sẽ được kiểm tra trước trigger.
* Một table có thể có nhiều Triggers cho một action. Một trigger có thể được định nghĩa cho nhiều action.
* Thứ tự thi hành sẽ là: trigger INSTEAD OF, các constraint, và sau cùng là trigger AFTER. Khi có nhiều trigger trong một table, thì table owner có thể dùng procedure hệ thống sp\_settriggerorder để chỉ định trigger đầu và trigger cuối để thực thi. Thứ tự của các trigger còn lại không thể sắp xếp được.
* User phải có quyền để thực hiện tất cả các lệnh mà được định nghĩa trong Triggers
* Table Owners không thể tạo ra các Triggers trên Views hoặc Temporary Tables nhưng có thể tham chiếu đến view và temporary.
* Khi có nhiều trigger trong một table, thì table owner có thể dùng procedure hệ thống **sp\_settriggerorder** để chỉ định trigger đầu và trigger cuối để thực thi. Thứ tự của các trigger còn lại không thể sắp xếp được.
* User phải có quyền để thực hiện tất cả các lệnh mà được định nghĩa trong Triggers.
* Table Owners không thể tạo ra các Triggers trên Views hoặc Temporary Tables nhưng có thể tham chiếu đến view và temporary.
* Triggers không trả kết quả về.
* Triggers có thể điều khiển multi-row actions: một hành động INSERT, UPDATE, hoặc DELETE gọi một trigger có thể ảnh hưởng lên nhiều dòng dữ liệu, Ta có thể chọn:
* Xử lý tất cả các dòng cùng với nhau trong trường hợp các dòng ảnh hưởng phải thỏa điều kiện của trigger.
* Xử lý từng dòng thỏa điều kiện.

## PHÂN LOẠI

Có 4 loại trigger sau:

1. **Insert trigger:**

Được thực hiện mỗi khi mẩu tin mới được chèn vào bảng. Một bảng tạm Inserted sẽ được sinh ra để chứa mẩu tin cần chèn.

1. **Delete trigger:**

Được thực hiện mỗi khi các mẩu tin trong bảng bị xóa. Một bảng tạm Deleted được sinh ra để lưu các mẩu tin bị xóa.

1. **Update trigger:**

Được thực hiện khi các bản các mẩu tin của bảng được cập nhật. Hai bảng Inserted và Deleted sẽ được sinh ra. Bảng Inserted sẽ lưu thông tin các mẩu tin mới được sửa, bảng Deleted sẽ lưu thông tin các mẩu tin cũ.

1. **Instead of trigger:**

Trigger cho phép cập nhật dữ liệu các bảng thông qua view có liên kết nhiều bảng, hoặc bẫy trước khi các constraint có tác dụng.

**Chú ý:**

* Một bảng có nhiều trigger.
* Mỗi một trigger có tên duy nhất.
* Trong trigger thường dùng mệnh đề **IF EXISTS.**
* Sử dụng trigger trong toàn vẹn dữ liệu.
* Sử dụng trigger trong ràng buộc tham chiếu.

## TẠO TRIGGER

Cú pháp:

CREATE TRIGGER <Tên\_ Trigger> ON <Tên\_bảng>

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

AS

<Các\_câu\_lệnh\_SQL>

Chú thích:

<Tên\_ Trigger>: tên Trigger.

<Tên\_bảng>: tên bảng cần định nghĩa Trigger.

INSERT, UPDATE, DELETE: định nghĩa thao tác để kích hoạt Trigger.

**Ví dụ 1: Tạo ràng buộc lương nhân viên phải là số dương.**

CREATE TRIGGER LUONG\_DUONG ON NHANVIEN

FOR INSERT, UPDATE

AS

BEGIN

IF (SELECT **COUNT(\*)** FROM INSERTED WHERE LUONG < 0) **> 0**

BEGIN

Print N'Lương phải là số dương'

Rollback Tran

END

END

**Ví dụ 2: Ràng buộc 0 <= LUONG <= 100000**

CREATE TRIGGER LUONG\_DUONG ON NHANVIEN

FOR INSERT, UPDATE

AS

BEGIN

IF (SELECT **COUNT(\*)** FROM INSERTED

WHERE LUONG < 0 OR LUONG > 100000) **> 0**

BEGIN

Print ‘0 <= LUONG <= 100000’

Rollback Tran

END

END

## HIỆU CHỈNH TRIGGER

Cú pháp:

ALTER TRIGGER <Tên\_ Trigger> ON <Tên\_bảng>

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

AS

<Các\_câu\_lệnh\_SQL>

## XÓA TRIGGER

Cú pháp:

DROP TRIGGER<Ten\_ Trigger>

Ví dụ:

DROP TRIGGER LUONG\_DUONG

## 

## MỘT VÀI VÍ DỤ

* **Ràng buộc lương nhân viên phải tăng:**

CREATE TRIGGER LUONG\_TANG ON NHANVIEN

FOR UPDATE

AS

IF UPDATE(LUONG)

BEGIN

IF (SELECT COUNT(\*) FROM INSERTED I, DELETED D

WHERE D.LUONG >= I.LUONG AND I.MANV=D.MANV) > 0

BEGIN

Print N'Lương nhân viên phải tăng'

Rollback Tran

END

END

* **Ràng buộc nhân viên phải từ 18 tuổi trở lên:**

CREATE TRIGGER TUOI\_18\_TROLEN ON NHANVIEN

FOR INSERT, UPDATE

AS

IF EXISTS (SELECT \* FROM INSERTED

WHERE DATEADD(YY, 18, NGSINH) > GETDATE())

BEGIN

Print N'Nhân viên phải từ 18 tuổi trở lên'

Rollback Tran

END

* **Ràng buộc không cho phép thêm, sửa bảng NHANVIEN nếu PHG không có trong bảng PHONGBAN:**

CREATE TRIGGER TONTAI\_PHONG ON NHANVIEN

FOR INSERT, UPDATE

AS

IF NOT EXISTS (SELECT \* FROM INSERTED, PHONGBAN

WHERE PHG = MAPHG)

BEGIN

Print N’Mã phòng chưa tồn tại’

Rollback Tran

END

* **Ràng buộc không cho sửa MAPHG trong bảng PHONGBAN:**

CREATE TRIGGER KHONG\_SUA\_KHOACHINH ON PHONGBAN

FOR UPDATE

AS

IF UPDATE(MAPHG)

BEGIN

Print N‘Không được sửa khóa chính’

Rollback Tran

END

* **Sửa MAPHG trong bảng PHONGBAN thì phải sửa luôn những mẫu tin có liên quan trong bảng NHANVIEN:**

CREATE TRIGGER SUA\_DAYCHUYEN ON PHONGBAN

FOR UPDATE

AS

IF UPDATE(MAPHG)

BEGIN

UPDATE NHANVIEN

SET PHG = I.MAPHG

FROM NHANVIEN, DELETED D, INSERTED I

WHERE PHG = D.MAPHG

END

* **Không cho xóa NHANVIEN nếu nhân viên đó có trong bảng PHANCONG:**

CREATE TRIGGER KHONG\_XOA\_NV\_CO\_PHANCONG ON NHANVIEN

FOR DELETE

AS

IF (SELECT COUNT(\*) FROM DELETED, PHANCONG

WHERE MANV = MA\_NVIEN) > 0

BEGIN

Print N’Nhân viên có trong phân công’

Rollback Tran

END

TÓM TẮT

Trigger là một loại đặc biệt của thủ tục lưu trữ, nó tự động thực thi khi một sự kiện xảy ra trong các máy chủ CSDL. DML (Data Manipulation Language – ngôn ngữ thao tác dữ liệu) gây nên khi người dùng cố gắng sửa đổi dữ liệu thông qua một ngôn ngữ thao tác dữ liệu (sự kiện INSERT, UPDATE, hoặc DELETE trên một Table hoặc View). Trigger không được thực thi một cách tường minh nên cần thận trọng khi dùng trigger.

Trigger là một thủ tục không có tham số. Một table có thể có nhiều trigger. Trigger thường dùng để kiểm tra các ràng buộc mà không thể khai báo trên table như các ràng buộc liên thuộc tính-liên quan hệ, liên bộ-liên quan hệ, ràng buộc do sự hiện diện của chu trình.

Sử dụng lệnh ROLLBACK TRAN trong trigger để bãi bỏ phát biểu cập nhật khi cần thiết.

Các loại trigger: Insert trigger, Delete trigger, Update trigger và Istead of trigger.

CÂU HỎI ÔN TẬP

1. Khái niệm về trigger? Tại sao phải dùng trigger trong một hệ thống CSDL?
2. Phân loại trigger và cho biết nguyên lý hoạt động của từng loại?
3. Ta sử dụng Check Constraint lúc tạo field cho table khi nào? Trong ngữ cảnh đó, ta dùng trigger được không? Giải thích?

Thực hiện các câu hỏi trong bài thực hành số 2 và 3 liên quan đến trigger