TRIGGER MySQL & Oracle

¿Qué es esto y qué hace el trigger en MySQL?

```
DELIMITER $$

CREATE TRIGGER trg_auditoria_nombre_departamento
AFTER UPDATE ON departamentos
FOR EACH ROW
BEGIN
   IF OLD.nombre <> NEW.nombre THEN
      INSERT INTO auditoria_departamentos (
         departamento_id,
         nombre_antiguo,
         nombre_nuevo,
         usuario
      ) VALUES (
         OLD.id,
         OLD.nombre,
         NEW.nombre,
         CURRENT_USER()
      );
   END IF;
END$$

DELIMITER ;
```

¿Qué es un trigger?

Un **trigger** es un bloque de código que se ejecuta **automáticamente** cuando ocurre un evento en una tabla (INSERT, UPDATE o DELETE).

No lo llamas tú, lo llama MySQL cuando se cumple la condición.

- Es un trigger llamado **trg_auditoria_nombre_departamento**
- Tipo: AFTER UPDATE : se ejecuta **después** de un UPDATE sobre la tabla departamentos

- FOR EACH ROW : se ejecuta **una vez por cada fila** que haya sido actualizada.

¿Qué hace exactamente?

Hace una **auditoría de cambios de nombre de departamento**.
Paso a paso:

1. Cada vez que se hace un UPDATE sobre la tabla departamentos, el trigger se activa.
2. Dentro del trigger usas OLD y NEW:
    - OLD.campo : valor antes del UPDATE
    - NEW.campo : valor después del UPDATE
3. Comprueba esta condición:
4. IF OLD.nombre <> NEW.nombre THEN

Es decir: **solo actúa si el nombre del departamento ha cambiado**.

5. Si ha cambiado, entonces inserta un registro en auditoria_departamentos con:

```
INSERT INTO auditoria_departamentos (
  departamento_id,
  nombre_antiguo,
  nombre_nuevo,
  usuario
) VALUES (
  OLD.id,        -- ID del departamento
  OLD.nombre,    -- Nombre anterior
  NEW.nombre,    -- Nombre nuevo
  CURRENT_USER()  -- Usuario que hizo el cambio
);
```

Cada vez que alguien cambia el nombre de un departamento, se guarda un registro en la tabla auditoria_departamentos con el id, el nombre antiguo, el nuevo nombre y el usuario que hizo el cambio. Es un historial de cambios.

¿Por qué usa DELIMITER $$ y luego DELIMITER ;?

Por defecto, en MySQL el delimitador de sentencias es **;**.
El problema: dentro del trigger tienes varios ;:

```
IF ... THEN
   INSERT ...;
END IF;
END$$
```

Si no cambias el delimitador:

- El cliente de MySQL pensaría que la sentencia termina en el **primer** ; que encuentre (por ejemplo, después del INSERT) y no entendería el bloque completo.

¿Qué hace DELIMITER $$?

Le dices al cliente de MySQL:
"A partir de ahora, la sentencia *termina* cuando veas $$, no cuando veas ;"
Entonces puedes escribir tranquilamente:

```
CREATE TRIGGER ...
BEGIN
  ...
END$$
```

Y MySQL entiende que **todo eso es una única sentencia** (la creación del trigger).

¿Y DELIMITER ; al final?

Cuando terminas, devuelves el delimitador al valor normal:

```
DELIMITER ;
```

Para que el resto de sentencias SQL (SELECT, UPDATE, etc.) vuelvan a funcionar normalmente acabando en ;.

Resumen

- **Trigger**: código que se ejecuta automáticamente cuando se hace un UPDATE en departamentos.
- **Función**: Si cambia el nombre del departamento, guarda en auditoria_departamentos el id, nombre antiguo, nombre nuevo y usuario.
- **OLD / NEW**: valores antes y después del UPDATE.
- **DELIMITER $$**: se cambia el final de la sentencia para poder usar ; dentro del trigger sin que el cliente se confunda.
- **DELIMITER ;**: se vuelve al delimitador normal después de crear el trigger.

Tu trigger en Oracle será este:

```
CREATE OR REPLACE TRIGGER trg_auditoria_nombre_departamento
AFTER UPDATE OF nombre ON departamentos
FOR EACH ROW
BEGIN
   -- Solo registramos si ha cambiado el nombre
   IF :OLD.nombre <> :NEW.nombre THEN
      INSERT INTO auditoria_departamentos (
         departamento_id,
         nombre_antiguo,
         nombre_nuevo,
         usuario
      ) VALUES (
         :OLD.id,
         :OLD.nombre,
         :NEW.nombre,
         USER
      );
   END IF;
END;
/
```

En Oracle:

- No se usa DELIMITER
- Se usan :OLD y :NEW
- Se suele usar USER para saber el usuario actual
- Se termina el bloque con / al final para compilarlo en herramientas como SQL*Plus o SQL Developer

Diferencias clave

- AFTER UPDATE OF nombre ON departamentos
  - En Oracle puedes especificar el campo que dispara el trigger (opcional, pero elegante).
- :OLD / :NEW en lugar de OLD / NEW
  - En Oracle siempre van con dos puntos delante.
- USER en lugar de CURRENT_USER()
  - USER devuelve el nombre del usuario de la sesión actual.
- El / final
  - Indica a Oracle que compile el bloque PL/SQL.