DISPARADORES EN SQL

Modelos Avanzados de Bases de Datos

DISPARADORES EN SQL:1999

- Un disparador (trigger) es un procedimiento especial que se ejecuta en respuesta a un evento específico.
  - Ej. Al aumentar el sueldo de un empleado, que se aumente automáticamente el total de gastos de la empresa.
- Un disparador tiene 3 partes:
  - Evento: cabecera con la que se crea el trigger
  - Condición: que tiene que suceder para que se dispare el trigger
  - Acción: que actividades se tienen que llevar a cabo

SINTAXIS GENERAL DE UN DISPARADOR EN SQL:1999

CREAR O REEMPLAZAR UN DISPARADOR

```
CREATE [OR REPLACE] TRIGGER nombre
    [temporalidad del evento]
    [granularidad del evento]          } EVENTO
    [WHEN condición]                   } CONDICIÓN
    BEGIN
       cuerpo del trigger              } ACCIÓN
    END;
```

SINTAXIS GENERAL DE UN DISPARADOR EN SQL:1999

- Temporalidad del evento
  - BEFORE Operación → El cuerpo del disparador debe ejecutarse antes del evento que causa la activación del disparador
  - AFTER Operación → El cuerpo del disparador debe ejecutarse después del evento que causa la activación del disparador

Operación: INSERT, DELETE O UPDATE

Ej. AFTER DELETE ON nombre_tabla
AFTER DELETE OF nombre_columna ON nombre_tabla

SINTAXIS GENERAL DE UN DISPARADOR EN SQL:1999

Ej:

1) CREATE OR REPLACE TRIGGER emp
   BEFORE INSERT ON empleado
   ……..
2) CREATE OR REPLACE TRIGGER salar
   AFTER DELETE OF salario ON empleado

SINTAXIS GENERAL DE UN DISPARADOR EN SQL:1999

- Granularidad del evento
  - FOR EACH ROW

    El disparador es a nivel de fila. El cuerpo del disparador se debe aplicar fila a fila a la tabla afectada.
  - FOR EACH STATEMENT

    El disparador es a nivel de orden. El cuerpo del disparador se debe aplicar a toda la tabla a la vez.

SINTAXIS GENERAL DE UN DISPARADOR EN SQL:1999

- WHEN condición (sólo para disparadores a nivel de fila)
  - Operadores relacionales:

    < <= > >= = <>
  - Operadores lógicos:

    AND, OR, NOT

SINTAXIS GENERAL DE UN DISPARADOR EN SQL:1999

WHEN condición

Para referirnos al valor de la fila

- OLD
- NEW

Si están en el cuerpo del disparador se referencian como :OLD ó :NEW

SINTAXIS GENERAL DE UN DISPARADOR EN SQL:1999

- Con OLD.nombre_columna referenciamos:
  - al valor que tenía la columna antes del cambio debido a una modificación (UPDATE)
  - al valor de una columna antes de una operación de borrado sobre la misma (DELETE)
  - al valor NULL para operaciones de inserción (INSERT)
- Con NEW.nombre_columna referenciamos:
  - Al valor de una nueva columna después de una operación de inserción (INSERT)
  - Al valor de una columna después de modificarla mediante una sentencia de modificación (UPDATE)
  - Al valor NULL para una operación de borrado (DELETE)

EJEMPLO

```
CREATE TRIGGER Ejemplo-fila
AFTER DELETE OF codigo ON tabla1
FOR EACH ROW
WHEN ((OLD.nombre='pepe') OR (OLD.edad > 35))
BEGIN
   DELETE FROM tabla2 WHERE tabla2.cod=:OLD.cod;
END Ejemplo-columna;
/
```

EJEMPLO

```
CREATE TRIGGER Ejemplo_sentencia
AFTER DELETE ON tabla1
REFERENCING OLD AS anterior
BEGIN
   DELETE FROM tabla2 WHERE
   tabla2.cod=anterior.cod;
END Ejemplo_sentencia;
/
```

ACTIVAR/DESACTIVAR DE DISPARADORES

- Todos los disparadores asociados a una tabla:
  - ALTER TABLE nombre_tabla ENABLE ALL TRIGGERS
  - ALTER TABLE nombre_tabla DISABLE ALL TRIGGERS
- Un disparador específico:
  - ALTER TRIGGER nombre_disparador ENABLE
  - ALTER TRIGGER nombre_disparador DISABLE

OTRAS FUNCIONES

- Eliminar un disparador
  - DROP TRIGGER nombre_disparador
- Ver todos los disparadores y su estado
  - SELECT TRIGGER_NAME , STATUS FROM USER_TRIGGERS;

OTRAS FUNCIONES

- Ver el cuerpo de un disparador
  - SELECT TRIGGER_BODY FROM USER_TRIGGERS WHERE TRIGGER_NAME='nombre_disparador';
- Ver la descripción de un disparador
  - SELECT DESCRIPTION FROM USER_TRIGGERS WHERE TRIGGER_NAME= 'nombre_disparador';

Funciones del cuerpo del disparador

- En el cuerpo del disparador se pueden incluir:
  - Sentencias SQL
  - Subprogramas escritos en PL/SQL
- Funciones: Inserting, Deleting, Updating
  - Utilizar cuando el disparador se tiene que activar ante diferentes operaciones pero deben hacer cosas diferentes para cada evento.

Funciones del cuerpo del disparador

```
CREATE OR REPLACE TRIGGER ejemplo
BEFORE INSERT OR UPDATE OR DELETE ON tabla
BEGIN
        IF DELETING THEN
                Acciones asociadas al borrado
        ELSIF INSERTING THEN
                Acciones asociadas a la inserción
        ELSE
                Acciones asociadas a la modificación
        END IF;
END ejemplo;
/
```

Funciones del cuerpo del disparador

- RAISE_ERROR_APPLICATION (nro_error, mensaje); [-20000 y -20999]

Esta función devuelve mensajes de error al usuario cuando éste va a realizar una operación no permitida y que interrumpe la ejecución del disparador

```
CREATE OR REPLACE TRIGGER ejemplo
BEFORE DELETE ON tabla
FOR EACH ROW
BEGIN
        IF tabla.columna= valor_no_borrable THEN
            RAISE_APPLICATION_ERROR(-20000,'La     fila
              no se puede borrar');
        END IF;
        ...
END ejemplo;
```

Declaración de variables

Dentro de un disparador se pueden declarar variables que se utilizarán dentro del cuerpo del mismo.

```
CREATE...
BEFORE...
[FOR EACH ROW ...]
DECLARE
    Declaración de variables
BEGIN
...
Nombre CONSTANT NUMBER:=valor;
nombre TIPO;
nombre nombretabla.nombrecolumna%TYPE;
nombre nombretabla%ROWTYPE;
```

Crea una variable nombre del mismo tipo que el de la columna nombrecolumna de la tabla nombretabla

Crea una variable nombre del tipo de la tabla. Para asignar el valor de una columna se pone: nombre.nombrecolumna

Restricciones de los disparadores

- Tabla mutante. La que está siendo modificada por una operación DML o una tabla que se verá afectada por los efectos de un DELETE CASCADE debido a la integridad referencial
- Tabla de restricción. Tabla de la que un disparador puede necesitar leer debido a una restricción de integridad referencial

Restricciones de los disparadores

- Las órdenes del cuerpo de un disparador no pueden:
  - Leer o modificar una tabla mutante
  - Leer o modificar claves primarias o ajenas de una tabla de restricción
- Para evitar estos problemas es necesaria la utilización de paquetes

Restricciones de los disparadores

```
CREATE OR REPLACE TRIGGER chequear_salario
    BEFORE INSERT OR UPDATE ON empleado
    FOR EACH ROW
    WHEN (new.trabajo<>'presidente')
    DECLARE
          v_salariomin empleado.salario%TYPE;
          v_salariomax empleado.salario%TYPE;
    BEGIN
          SELECT MAX(salario), MIN(salario) FROM empleado
                    INTO v_salariomin, v_salariomax
          FROM empleado
          WHERE trabajo=:new.trabajo;
          IF :new.salario<v_salariomin OR :new.salario> v_salariomax THEN
             RAISE_APPLICATION_ERROR(-20001, 'Sueldo fuera de rango');
          END IF;
    END chequear_salario;
    /
```

UPDATE empleado SET salario=1500 WHERE nom_emp='Cortecero';

Restricciones de los disparadores

Dos tablas: empleado y departamento. En la tabla empleado tenemos el número de departamento como clave ajena.

```
CREATE TRIGGER ejemplo
AFTER UPDATE ON nrodep OF departamento
FOR EACH ROW
BEGIN
      UPDATE empleado
SET empleado.dep = :NEW.nrodep
      WHERE empleado.dep= :OLD.nrodep;
END ejemplo;
/
```

UPDATE departamento
SET nrodep= 1
WHERE nrodep=7;

Disparadores de sustitución

- INSTEAD OF. Sólo para vistas

```
CREATE VIEW vista AS
SELECT edificio, sum(numero_asientos) FROM habitaciones
GROUP BY edificio;
```

Es ilegal hace una operación de borrado directamente en la vista:

```
DELETE FROM vista WHERE edificio='edificio 7';
```

Se crea un disparador de sustitución que efectúe el borrado equivalente pero sobre la tabla habitaciones

Disparadores de sustitución

```
CREATE TRIGGER borra_en_vista
    INSTEAD OF DELETE ON vista
    FOR EACH ROW
    BEGIN
          DELETE FROM habitaciones
          WHERE edificio = :OLD.edificio;
    END borra_en_vista;
/
```