Triggers

Indice

- Definición de Trigger
- Ejecución de Triggers
- Variables especiales
- Nivel-Sentencia (Level-statement)
- Nivel-Fila (Level-Row)

Triggers

- **En PostgreSQL, un trigger es una función que se ejecuta de manera automática cuando se produce un determinado evento en la base de datos o los datos cambian.**

- **Cuando se ejecuta:**
  - **Antes de una operación:** se comprueban algunas restricciones antes de un insert, update o delete.
  - **Después de una operación completa:** se comprueban algunas restricciones después de un insert, update o delete.

- **Sintaxis básica**:

create trigger nombre_trigger { **before** | **after** }
{ **insert** | **update** | **delete** [o …] }
on nombre_tabla
[**for** [**each**] {fila | sentencia}]
[**when** (condicion)]
execute procedure nombre_función (argumentos)

Ejecución de un Trigger

- Trabaja sobre una tabla.
- Se **puede ejecutar a dos niveles:**
  - **Nivel-Sentencia:** el trigger es llamado una vez por cada operación, con independencia del número de filas que modifica.
  - **Nivel-Fila:** el trigger es llamado para cada fila que la operación modifica.
- **Un procedimiento asociado a un trigger:**
  - Tiene que ser creado e instalado antes de la definición del trigger.
  - Puede ser usado por diferentes triggers y tablas.
  - Si varios triggers están asociados al mismo evento, entonces se ejecutarán en orden alfabético.
  - Puede llamar a otro trigger, pero tendrá un impacto en el rendimiento.

Variables especiales

Cuando una función es llamada como un trigger, entonces algunas variables son creadas automáticamente y proporcionan la siguiente información:

- **NEW** (tupla): contiene la nueva tupla para ser insertada/actualizada a nivel de fila.
- **OLD** (tupla): contiene la tupla antigüa para ser insertada/actualizada a nivel de fila.
- **TG_<x>:** describe la condición que dispara la llamada al trigger, tales como:
  - **TG_WHEN** (varchar): Retorna 'BEFORE' o 'AFTER' dependiendo de si el trigger se ejecuta antes o después de la operación.
  - **TG_LEVEL** (varchar): Retorna 'ROW' o 'STATEMENT'.
  - **TG_OP** (varchar): Retorna 'INSERT', 'UPDATE', 'DELETE' o 'TRUNCATE' dependiendo de la operación que el trigger haya ejecutado.
  - **TG_TABLE_NAME** (varchar): Retorna el nombre de la tabla que dispara la operación.

Trigger a Nivel-Sentencia

- Siempre retornan NULL.
- No accede a los datos vinculados a la instrucción.
- Útil para disparar otras acciones.
- Si son ejecutados antes (BEFORE) de las instrucciones, entonces pueden abortar operaciones mediante el lanzamiento de una excepción.

Ejemplo: Trigger a Nivel-Sentencia

```
CREATE OR REPLACE FUNCTION check_cliente() RETURNS
trigger AS $$

BEGIN
        if extract(dow from current_date) in (6,7) then RAISE
        EXCEPTION 'Hoy no es un día laborable';
END IF;
RETURN null;
END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER tg_check_cliente BEFORE INSERT
ON clientes FOR EACH STATEMENT
EXECUTE PROCEDURE check_cliente();
```

Trigger a Nivel-Fila

- Si se ejecuta antes (**BEFORE)**:
  - Si retorna null, entonces la operación no se ejecuta.
  - En caso contrario, la operación se ejecuta como:
    - Modifica la nueva tupla (new) con otros valores y return new. Ejemplo: new.nombre = <nuevo valor>.
    - Retorna un valor diferente de new, entonces las filas cambiarán en función de dicho valor.
    - Retorna new sin modificación, entonces la fila se modifica de acuerdo a los valores proporcionados en la instrucción.
  - En operaciones de DELETE, el valor retornado no es relevante, pero si se retorna new, entonces el valor será null.

Trigger a Nivel-Fila

- Si se ejecuta después (**AFTER)**:
  - El valor retornado no es relevante.
  - No se puede modificar datos en la tabla porque ya han sido modificados.
  - Se pueden usar las variables NEW y OLD para modificar otros datos.

Ejemplo: Trigger a Nivel-Fila

```
CREATE OR REPLACE FUNCTION modifica_fecha()
RETURNS trigger AS $$

BEGIN
        new.fecha = current_date;
        return new;
END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER tg_modifica_fecha
BEFORE INSERT ON clientes
FOR EACH ROW
EXECUTE PROCEDURE modifica_fecha();
```

Triggers