



# Laboratorio 5

TRIGGERS

ACTIVIDAD DE INVESTIGACIÓN (Grupo 2)

## I. PREGUNTAS DE INVESTIGACIÓN

1. ¿Relacionando a PostgreSQL, explique que es un Trigger y cuál es el propósito principal en una base de datos relacional?
2. Describa la sintaxis básica para crear un Trigger en PostgreSQL, incluyendo las cláusulas clave como BEFORE, AFTER, FOR EACH ROW/FOR EACH STATEMENT, y WHEN.
3. ¿Cuál es la diferencia fundamental entre un Trigger BEFORE y un Trigger AFTER? Proporcione un ejemplo práctico de cuándo usaría cada uno.
4. Explique la diferencia entre un Trigger a nivel de fila (FOR EACH ROW) y un Trigger a nivel de sentencia (FOR EACH STATEMENT). ¿Cuándo sería apropiado usar cada tipo?
5. ¿Cómo se puede acceder a los valores antiguos y nuevos de una fila dentro de una función de Trigger en PostgreSQL? Mencione las variables especiales utilizadas para este propósito.
6. ¿Es posible que un Trigger falle y, si es así, cómo se manejan los errores dentro de una función de Trigger para evitar que la operación que lo disparó se complete?
7. ¿Cómo se puede deshabilitar y habilitar un Trigger existente en PostgreSQL?



8. Mencione al menos tres posibles casos de uso prácticos para los Triggers en PostgreSQL en una aplicación real.
9. Con todos los elementos investigados, construya un crucigrama con las teorías investigados con un mínimo de 15 palabras.

## II. ANÁLISIS DE ALGORITMOS

1. Codifique y reconozca la funcionalidad del código:

```
1 -----
2  ✓ CREATE TABLE productos (
3     id SERIAL PRIMARY KEY,
4     nombre VARCHAR(100),
5     precio DECIMAL(10, 2)
6 );
7
8 -----
9  ✓ CREATE TABLE productos_auditoria (
10     id SERIAL PRIMARY PRIMARY KEY,
11     id_producto INT,
12     operacion VARCHAR(10),
13     usuario VARCHAR(50),
14     fecha_hora TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
15     valor_antiguo TEXT,
16     valor_nuevo TEXT
17 );
18 -----
19  ✓ CREATE OR REPLACE FUNCTION auditar_productos()
20 RETURNS TRIGGER AS $$
21 BEGIN
22     IF TG_OP = 'INSERT' THEN
```



```
23      INSERT INTO productos_auditoria (id_producto, operacion, usuario, valor_nuevo)
24      VALUES (NEW.id, 'INSERT', CURRENT_USER, ROW_TO_JSON(NEW)::TEXT);
25  ✓   ELSEIF TG_OP = 'UPDATE' THEN
26      INSERT INTO productos_auditoria (id_producto, operacion, usuario, valor_antiguo, valor_nuevo)
27      VALUES (OLD.id, 'UPDATE', CURRENT_USER, ROW_TO_JSON(OLD)::TEXT, ROW_TO_JSON(NEW)::TEXT);
28  ✓   ELSEIF TG_OP = 'DELETE' THEN
29      INSERT INTO productos_auditoria (id_producto, operacion, usuario, valor_antiguo)
30      VALUES (OLD.id, 'DELETE', CURRENT_USER, ROW_TO_JSON(OLD)::TEXT);
31      END IF;
32      RETURN NULL;
33  END;
34  $$ LANGUAGE plpgsql;
35
36  ✓   CREATE TRIGGER tr_auditar_productos
37  AFTER INSERT OR UPDATE OR DELETE ON productos
38  FOR EACH ROW
39  EXECUTE FUNCTION auditar_productos();
```