| GUÍA 2.3.5:
Triggers a nivel de filas

Sigla	Asignatura	Experiencia de Aprendizaje
BDY1103	Taller de Base de Datos	EA2: Implementa programas PL/SQL en la base de datos para construir una solución integral de procesamiento y generación de información o ser usados en otros procesos y/o aplicaciones.
Tiempo	**Modalidad de Trabajo**	**Indicadores de logro**
4h	Individual	IL2.3

Antecedentes generales

En esta guía encontrarás los contenidos asociados a la construcción de triggers para el control de acciones a nivel de fila, junto con ejemplos y actividades prácticas a desarrollar.

Requerimientos para esta actividad

En esta actividad, los y las estudiantes deberán utilizar SQL Developer y seguir las instrucciones indicadas por el/la docente.

Sesión 2: Triggers a nivel de fila

Objetivos de Aprendizaje

1. Comprender qué son los triggers y sus tipos.
2. Aprender a crear triggers a nivel de fila.
3. Aplicar triggers a nivel de sentencia en un escenario de negocio.

1. Introducción a los Triggers a Nivel de Filas

1.1 Concepto

- **Triggers a Nivel de Filas**: Se ejecutan una vez por cada fila afectada por una sentencia DML. Son útiles para validaciones y modificaciones fila por fila.

1.2 Sintaxis Básica

```
CREATE OR REPLACE TRIGGER trigger_name
{BEFORE | AFTER} {INSERT | UPDATE | DELETE} ON table_name
FOR EACH ROW
BEGIN
   -- Código PL/SQL
END;
/
```

2. Triggers Before y After a Nivel de Filas

2.1 Before Triggers

- **Concepto**: Son ejecutados antes de que la operación DML afecte la fila.
- **Ejemplo Práctico**: Validar el salario antes de insertar un nuevo empleado.

```
CREATE OR REPLACE TRIGGER validate_salary_before_insert
BEFORE INSERT ON emp
FOR EACH ROW
BEGIN
   IF :NEW.sal < 0 THEN
      RAISE_APPLICATION_ERROR(-20001, 'Salary cannot be negative');
   END IF;
END;
/
```

2.2 After Triggers

- **Concepto**: Son ejecutados después de que la operación DML afecte la fila.
- **Ejemplo Práctico**: Registrar cambios en el salario de un empleado.

```
-- Crear tabla de auditoría
CREATE TABLE emp_salary_audit (
   audit_id NUMBER PRIMARY KEY,
   empno NUMBER,
   old_salary NUMBER,
   new_salary NUMBER,
   change_date DATE
);

-- Crear secuencia para IDs únicos
CREATE SEQUENCE emp_salary_audit_seq START WITH 1 INCREMENT BY 1;

-- Crear trigger a nivel de fila
CREATE OR REPLACE TRIGGER audit_salary_changes
AFTER UPDATE OF sal ON emp
FOR EACH ROW
BEGIN
   INSERT INTO emp_salary_audit (audit_id, empno, old_salary, new_salary,
change_date)
   VALUES (emp_salary_audit_seq.NEXTVAL, :OLD.empno, :OLD.sal, :NEW.sal,
SYSDATE);
END;
/
```

3. Ejemplos Prácticos

Ejemplo 1: Trigger Before en la Tabla ORDERS

Objetivo: Validar la fecha de entrega antes de insertar una nueva orden.

```
CREATE OR REPLACE TRIGGER validate_order_date
BEFORE INSERT ON orders
FOR EACH ROW
BEGIN
   IF :NEW.order_datetime > SYSDATE THEN
      RAISE_APPLICATION_ERROR(-20002, 'Order date cannot be in the fu-
ture');
   END IF;
END;
/
```

Ejemplo 2: Trigger After en la Tabla PRODUCTS

Objetivo: Registrar cambios en el precio de un producto.

```
-- Crear tabla de auditoría
CREATE TABLE product_price_audit (
   audit_id NUMBER PRIMARY KEY,
   product_id NUMBER,
   old_price NUMBER,
   new_price NUMBER,
   change_date DATE
```

```
);

-- Crear secuencia para IDs únicos
CREATE SEQUENCE product_price_audit_seq START WITH 1 INCREMENT BY 1;

-- Crear trigger a nivel de fila
CREATE OR REPLACE TRIGGER audit_price_changes
AFTER UPDATE OF unit_price ON products
FOR EACH ROW
BEGIN
   INSERT INTO product_price_audit (audit_id, product_id, old_price,
new_price, change_date)
   VALUES (product_price_audit_seq.NEXTVAL, :OLD.product_id,
:OLD.unit_price, :NEW.unit_price, SYSDATE);
END;
/
```

4. Actividades Prácticas

Actividad 1: Crear un Trigger Before en la Tabla CUSTOMERS

1. **Objetivo**: Validar el email antes de insertar un nuevo cliente.
2. **Pasos**:
    - Crear el trigger `validate_customer_email`.

Actividad 2: Crear un Trigger After en la Tabla STORES

1. **Objetivo**: Registrar cambios en el nombre de la tienda.
2. **Pasos**:
    - Crear la tabla `store_name_audit`.
    - Crear la secuencia `store_name_audit_seq`.
    - Crear el trigger `audit_store_name_changes`.

Conclusión de la Sesión 2

- **Resumen**: Los triggers a nivel de fila son útiles para realizar validaciones y auditorías detalladas de cambios en las filas.
- **Actividades**: Prácticas que ayudan a entender la creación y aplicación de triggers a nivel de filas para las tablas CUSTOMERS y STORES.

Evaluación y Mejores Prácticas

1. **Validación**: Siempre validar los datos en los triggers BEFORE.
2. **Auditoría**: Usar triggers AFTER para auditar cambios importantes.
3. **Pruebas**: Realizar pruebas exhaustivas para asegurar el funcionamiento correcto de los triggers en diferentes escenarios.

Recursos Adicionales

Candel, C. J. F., Molina, J., Ruiz, F. J. B., Barceló, J. R. H., Ruiz, D. S., & Viera, B. J. C. (2019). Developing a model-driven reengineering approach for migrating PL/SQL triggers to Java: A practical experience. *The Journal of systems and software*, *151*, 38–64. https://doi.org/10.1016/J.JSS.2019.01.068

INFORMATICONFIG [@informaticonfig333]. (2021, enero 14). *Curso de Oracle PLSQL en español desde cero | TRIGGERS, FOR EACH ROW / BEFORE INSERT video(17)*. Youtube. https://www.youtube.com/watch?v=xkhGEFEH_Ec