TRIGGERS

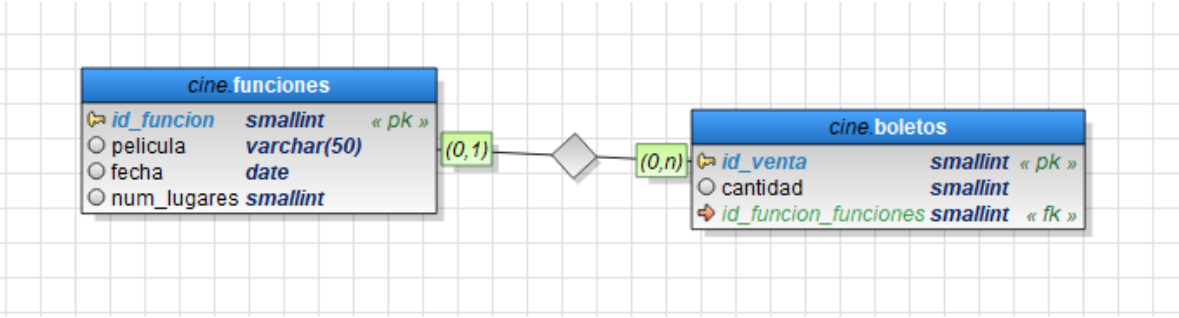
ORLANDO ULISES AGUILAR ROJAS - 173118

Introducción

Los triggers son una funcionalidad de la base de datos que se ejecuta de forma automática en operaciones como tipo insert, update o delete sobre una tabla o una vista

Desarrollo

Creación de las tablas en pgmodeler



Creamos las tablas en pgAdmin

```sql
--orlando aguilar
CREATE TABLE cine.funciones(
    id_funcion smallint,
    pelicula varchar(50),
    fecha date,
    num_lugares smallint,
    CONSTRAINT id_funcion PRIMARY KEY (id_funcion)

);
-- ddl-end --
```

```sql
--orlando aguilar
CREATE TABLE cine.boletos(
    id_venta smallint,
    cantidad smallint,
    id_funcion_funciones smallint,
    CONSTRAINT id_venta PRIMARY KEY (id_venta)

);
-- ddl-end --
```

```
21
22  --orlando aguilar
23  ALTER TABLE cine.boletos ADD CONSTRAINT funciones_fk FOREIGN KEY (i
24  REFERENCES cine.funciones (id_funcion) MATCH FULL
25  ON DELETE SET NULL ON UPDATE CASCADE;
26  -- ddl-end --
27
28
```

Insertamos un valor

```
29  --orlando aguilar
30  insert into cine.funciones (id_funcion, pelicula, fecha, num_lugares)
31  values (3, 'smile', now(), 20);
32
33
34
```

Query Editor | Query History | Notifications

Data Output | Explain | Messages

```
INSERT 0 1

Query returned successfully in 51 msec.
```

Creamos el trigger donde se verifica si va haber una inserción, verifica si hay boletos disponibles, en caso de que si, hace la transacción y actualiza la tabla de los boletos restantes

Query Editor

```
--orlando aguilar
CREATE OR REPLACE FUNCTION cine.revisar()

    RETURNS trigger
    LANGUAGE 'plpgsql'
     AS $BODY$
     DECLARE
        canti smallint := 0;
        restantes smallint := 0;
        BEGIN
      IF (TG_OP = 'INSERT') THEN

        SELECT num_lugares into canti
        FROM cine.funciones WHERE id_funcion = NEW.id_funcion ;
        IF canti >= NEW.cantidad THEN
            restantes = canti - NEW.cantidad;
            RAISE NOTICE 'Quedan % boletos',restantes;
            UPDATE cine.funcion
            SET  num_lugares = restantes
            WHERE id_funcion = NEW.id_funcion;
        RETURN NEW;
        ELSE
            RAISE NOTICE 'no hay boletos disponibles';
        END IF;
        END IF;
      RETURN NULL;
END;$BODY$;
```

```
--orlando aguilar
CREATE  TRIGGER checador
    BEFORE INSERT
    ON cine.boletos
    FOR EACH ROW
    EXECUTE PROCEDURE cine.revisar();
```

Insertamos una venta con 10 boletos comprados

```
68
69  --orlando aguilar
70  insert into cine.boletos(id_venta, cantidad, id_funcion)
71  values (9, 10, 3);
72
```

Query Editor Query History Notifications

Data Output Explain Messages

```
NOTICE:  Quedan 10 boletos
INSERT 0 1

Query returned successfully in 40 msec.
```

Otra con otros 10 boletos comprados

```
68
69  --orlando aguilar
70  insert into cine.boletos(id_venta, cantidad, id_funcion)
71  values (10, 10, 3);
72
```

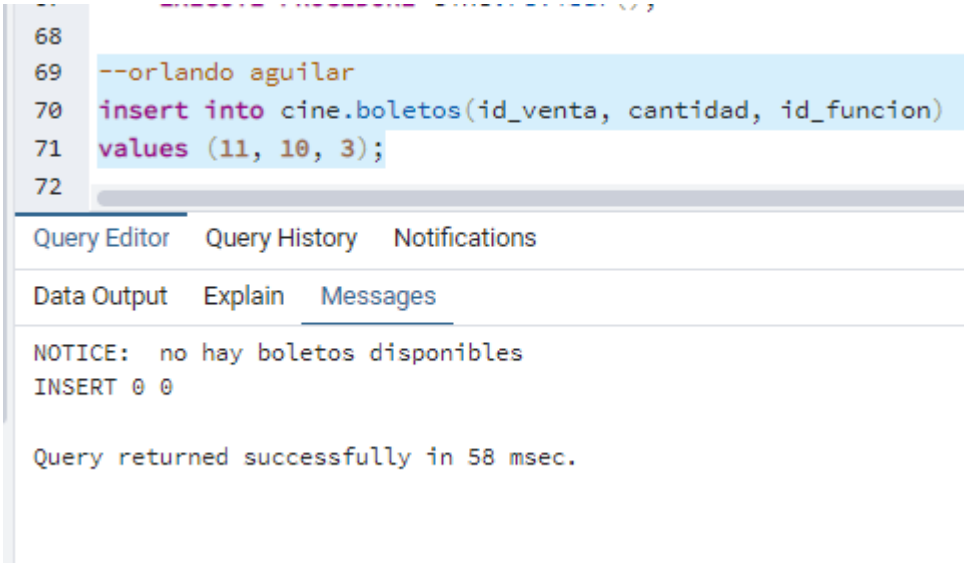
Query Editor Query History Notifications

Data Output Explain Messages

```
NOTICE:  Quedan 0 boletos
INSERT 0 1

Query returned successfully in 37 msec.
```

Y si queremos comprar una vez mas nos sale este mensaje



Conclusión

Los triggers tienen una buena función, pero no los usaría para controlar todo el backend de una aplicación, pues siento que toma muchos recursos de la base de datos y puede que sea mas gasto al usar el cpu para este tipo de filtrado de datos. Es interesante saber este tipo de controles que se puede hacer con el propio lenguaje de pg.