

Guía de aprendizaje

Implementando una solución integral de Base de Datos

Exp 3 – Semana 8

Programación de Bases de Datos (PRY2206)

**Facilitador disciplinar:** Alex Aguilera

**Asesor par:** Germán Barrientos

# Índice

[Introducción a la semana 3](#_Toc175582609)

[Resultado de aprendizaje 4](#_Toc175582610)

[Conceptos relevantes 4](#_Toc175582611)

[Preguntas activadoras 5](#_Toc175582612)

[Actividad 5](#_Toc175582613)

[Uso de Triggers 6](#_Toc175582614)

[Triggers a nivel de fila y de sentencia 6](#_Toc175582615)

[¿Cómo crear un trigger a nivel de sentencia? 8](#_Toc175582616)

[Predicados condicionales 12](#_Toc175582618)

[Uso de OLD y NEW en triggers a nivel de fila 14](#_Toc175582619)

[Crear un trigger a nivel de fila 15](#_Toc175582620)

[Usando restricciones en un trigger a nivel de fila 29](#_Toc175582621)

[Eliminar un trigger 31](#_Toc175582622)

[Cierre de la semana 34](#_Toc175582625)

[Referencias 35](#_Toc175582626)

[Lecturas de la semana 35](#_Toc175582627)

[Apuntes 36](#_Toc175582628)

# Introducción a la semana



Durante esta semana profundizaremos en el uso de triggers, herramientas esenciales para mantener la integridad de los datos y automatizar tareas en respuesta a eventos específicos en nuestras bases de datos.

Comenzaremos explorando los triggers a nivel de fila, los triggers a nivel de sentencia y los tipos de eventos que pueden activar un trigger. Luego, aprenderemos cómo crear un trigger a nivel de sentencia, incluyendo los pasos y las mejores prácticas para su implementación efectiva.

En el tema de Predicados Condicionales, exploraremos cómo añadir lógica condicional a nuestros triggers, y entenderemos el uso de las palabras clave OLD y NEW en los triggers a nivel de fila, para acceder a los valores anteriores y nuevos de las filas afectadas. Además, practicaremos crear un trigger a nivel de fila y a usar restricciones en estos triggers para asegurar que se cumplan ciertas condiciones antes de ejecutar la acción.

Finalmente, aprenderemos a eliminar un trigger cuando ya no sea necesario o cuando deseemos modificar su comportamiento.

# Resultado de aprendizaje

**El estudiante será capaz de:**

**RA3.** Implementa programas PL/SQL en la base de datos para la construcción de soluciones integrales de procesamiento y generación de información, además que estos mismos puedan ser de utilidad en otros procesos y/o aplicaciones.

**Indicador de logro:**

**IL9.** Implementa procedimientos y funciones almacenadas, tanto simples como complejas, considerando todos los componentes requeridos para un procesamiento de datos eficaz y optimizado.

**IL10.** Construye packages y triggers, abarcando desde implementaciones simples hasta complejas con el fin de generar, en forma integral, la información requerida en los casos planteados.

# Conceptos relevantes

| Packages | Triggers | Disparadores |
| --- | --- | --- |
| DELETE | Sentencia DML | INSERT |
| UPDATE | NEW | OLD |

# Preguntas activadoras

* ¿Qué diferencia un trigger a nivel de sentencia de un trigger a nivel de fila en PL/SQL?
* ¿Cómo se puede utilizar la cláusula WHEN en un trigger a nivel de fila para controlar su ejecución?
* ¿Cuáles son las ventajas principales de utilizar triggers para automatizar tareas en una base de datos?
* ¿Cuáles son las consideraciones clave al diseñar y gestionar triggers para mantener la integridad y seguridad de los datos?

# Actividad

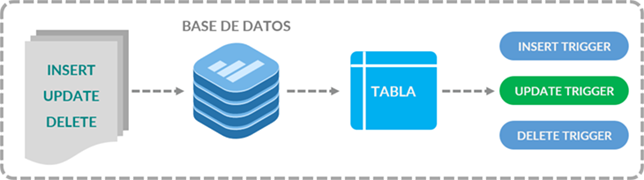
**Descripción de la actividad**

En esta octava semana realizarás una actividad sumativa individual con encargo llamada "Desarrollando programas PL/SQL en la Base de Datos", donde deberás implementar una solución integral de base de datos, incorporando procedimientos, funciones almacenadas, packages y triggers, para procesar información masiva de acuerdo al caso práctico planteado.

# Uso de Triggers

## Triggers a nivel de fila y de sentencia

La semana pasada vimos que los triggers (disparadores) son procedimientos almacenados que se ejecutan automáticamente en respuesta a ciertos eventos en una tabla o vista y pueden ser definidos para ejecutarse antes o después de una operación de inserción, actualización o eliminación.



**Existen dos tipos principales de triggers:**

### Triggers a nivel de fila (Row-level triggers)

Los triggers a nivel de fila se ejecutan una vez por cada fila afectada por la operación DML (Data Manipulation Language), como INSERT, UPDATE, o DELETE. Son particularmente útiles cuando necesitas aplicar una lógica específica a cada fila que se inserta, actualiza o elimina. Por ejemplo, si una sentencia afecta a 10 filas, el trigger se ejecutará 10 veces, una vez por cada fila.

### Triggers a nivel de sentencia (Statement-level triggers)

Por otro lado, los triggers a nivel de sentencia se ejecutan una vez por cada operación DML, sin importar cuántas filas sean afectadas. Son útiles para operaciones que necesitan ejecutarse una vez por transacción o cuando deseas realizar acciones que no dependen de la cantidad de filas afectadas. Por ejemplo, si una sentencia afecta a 10 filas, el trigger se ejecutará una sola vez.

Las diferencias clave entre triggers a nivel de fila y de sentencia son:

| **Triggers a nivel de fila** | **Triggers a nivel de sentencia** |
| --- | --- |
| **Contexto de uso** | **Frecuencia de ejecución** |
| Son más detallados y se utilizan cuando se necesita un control fino sobre cada fila afectada | Se utilizan para operaciones más generales que no dependen del número de filas afectadas. |
| Se ejecutan una vez por cada fila afectada. | Se ejecutan una vez por cada sentencia DML. |

### Tipos de eventos que pueden activar un trigger

Los triggers en bases de datos pueden ser activados por varios tipos de eventos que ocurren en las tablas o vistas de una base de datos. Los eventos más comunes que pueden activar un trigger son operaciones DML (Data Manipulation Language). Estos incluyen:

* **INSERT:** Se activa cuando se insertan nuevos registros en una tabla.
* **UPDATE**: Se activa cuando se modifican registros existentes.
* **DELETE:** Se activa cuando se eliminan registros de una tabla.

# ¿Cómo crear un trigger a nivel de sentencia?

Existen dos opciones para crear un trigger a nivel de sentencia usando SQL Developer:

**Opción 1: Abriendo una hoja de trabajo de SQL y escribir el código**

Para crear un trigger a nivel de sentencia, puedes abrir una hoja de trabajo en SQL Developer y escribir el código del trigger. Una vez que hayas terminado de escribir el código, debes elegir la opción "Ejecutar Script".

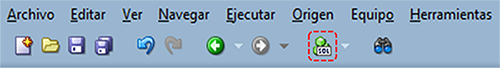
Importante

Recuerda que en la base de datos solo se guarda el código fuente del trigger, por lo tanto, si el código no posee errores el trigger quedará almacenado en la base de datos, mientras que, si el código tiene errores, el trigger no se puede crear.

Figura 1

*Creación de un trigger a través de una hoja de trabajo*

**Paso 1**

**

**Paso 2**

*Nota.* Creación de Trigger a Nivel de Sentencia usando SQL Developer. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

En este ejemplo, el trigger TRG\_SEGURIDAD\_EMP se ejecuta una vez antes de cada operación de INSERT en la tabla ‘employees’. Verifica que la operación de inserción se realice solo durante las horas laborales (entre las 08:00 y las 18:00). Si la inserción se intenta fuera de este rango de tiempo, se genera un error con el mensaje "Se debe insertar en tabla EMPLOYEES sólo durante horas de trabajo”.

**Opción 2: Usando la interfaz gráfica de SQL Developer**

Con el botón derecho del mouse selecciona “Nuevo Disparador” (paso 1). Se abrirá una ventana en donde se le puede dar el nombre al trigger, elegir a qué tabla estará asociado, el tiempo en que se activará, la sentencia DML que los activará y definir que es un trigger a nivel de sentencia (paso 2). Posteriormente al dar clic en “Aceptar”, se abrirá el área de trabajo para escribir el código del trigger.

Una vez que se ha finalizado, se debe elegir la opción “Compilar” (paso 3) para que el trigger quede almacenado en la base de datos. Si el trigger no tiene errores, quedará almacenado en la base de datos el código fuente del trigger. Si existen errores, no se podrá crear.

Figura 2

*Creación de un trigger con disparadores*

**Paso 1**

**Paso 2**

**Paso 3**

*Nota.* Creación de Trigger a Nivel de Sentencia usando SQL Developer. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

En el siguiente ejemplo, se crea el trigger TRG\_SEGURIDAD\_EMP a nivel de una sentencia INSERT que se realice sobre la EMPLEADOS.

Figura 3

*Cómo crear un Trigger a Nivel de Sentencia (código del trigger y sentencia INSERT)*

CREATE OR REPLACE TRIGGER TRG\_SEGURIDAD\_EMP

BEFORE INSERT ON employees

BEGIN

IF (TO\_CHAR(SYSDATE,'HH24:MI') NOT BETWEEN '08:00' AND '18:00') THEN

RAISE\_APPLICATION\_ERROR(-20500, 'Se debe insertar en tabla EMPLOYEES sólo durante horas de trabajo.');

END IF;

END;

INSERT INTO empleados (employee\_id, last\_name, first\_name, email, hire\_date, job\_id, salary, department\_id)

VALUES (300, 'Smith', 'Rob', 'RSMITH', SYSDATE, 'IT\_PROG', 4500, 60);

*Nota*. Creación de trigger a Nivel de sentencia usando SQL Developer. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

El trigger TRG\_SEGURIDAD\_EMP validará, antes de que se inserte una fila en la tabla, en qué horario se está efectuando la operación. Si la hora en que se está insertado la nueva fila no está entre las 09:00 y 18:00 entonces se mostrará el mensaje definido en la excepción.

Por lo tanto, primero se ejecuta la acción del trigger TRG\_SEGURIDAD\_EMP y posteriormente, si no hay error, se ejecuta la sentencia INSERT sobre la tabla EMPLEADOS.

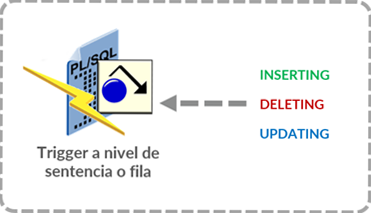
# Video

En el siguiente video, podrás observar un ejemplo práctico de cómo se crea un trigger:

<https://videosduoc.duoc.cl/media/t/1_950nc26w>

## Predicados condicionales

En los triggers a nivel de sentencia o fila, si hay varias operaciones DML que pueden activar el trigger, se pueden utilizar los predicados condicionales ‘INSERTING’, ‘DELETING’ y ‘UPDATING’ en el cuerpo del trigger para determinar qué tipo de declaración ha desencadenado su ejecución.



* **INSERTING:** retorna TRUE si la orden es INSERT.
* **DELETING:** retorna TRUE si la orden es DELETE.
* **UPDATING:** retorna TRUE si la orden es UPDATE.

En el siguiente ejemplo, se crea el trigger TRG\_VALIDA\_DML\_EMP para restringir el día y hora en que se producen todos los eventos de manipulación de datos en la tabla EMPLEADOS. El trigger efectuará la validación del día y hora que se realiza cualquier sentencia ‘INSERT’, ‘UPDATE’ o ‘DELETE’ antes de que se haga efectiva en la tabla.

**Figura 4**

*Ejemplo de predicados condicionales*

CREATE OR REPLACE TRIGGER TRG\_VALIDA\_DML\_EMP

BEFORE INSERT OR UPDATE OR DELETE ON employees

BEGIN

IF TO\_CHAR(SYSDATE,'HH24') NOT BETWEEN '08' AND '18' THEN

IF DELETING THEN

RAISE\_APPLICATION\_ERROR(-20502, 'Se debe eliminar desde tabla EMPLEADOS sólo durante horas de trabajo.');

ELSIF INSERTING THEN

RAISE\_APPLICATION\_ERROR(-20500, 'Se debe insertar en tabla EMPLEADOS solo durante horas de trabajo.');

ELSIF UPDATING('SALARY') THEN

RAISE\_APPLICATION\_ERROR(-20503, 'Se debe actualizar salario sólo durante horas de trabajo.');

END IF;

END IF;

END;

*Nota.* Ejemplo de predicados condicionales. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

Importante

En el caso de efectuarse una operación de ‘UPDATE’, se valida sólo cuando se produzca una actualización sobre la columna ‘salary’ de la tabla.

## Uso de OLD y NEW en triggers a nivel de fila

Siempre que se activa un trigger a nivel de fila, Oracle crea y completa en tiempo de ejecución dos estructuras de datos PL/SQL que funcionan como registros: los pseudo-registros **‘NEW’** y **‘OLD’** (denominados "pseudo" porque no comparten todas las propiedades de los registros PL/SQL reales) para la misma fila que se está procesando.

Tabla 1

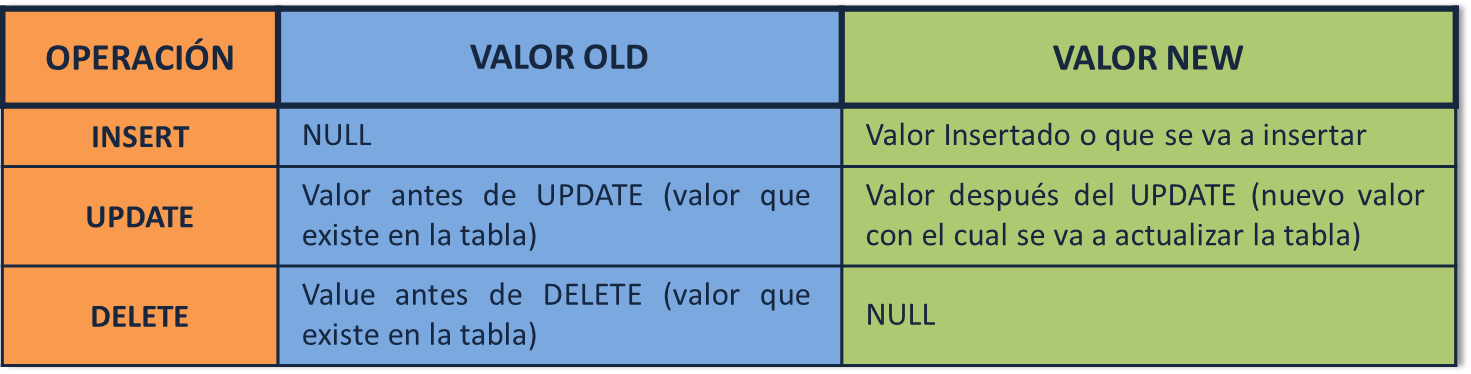
*Definición y características de OLD y NEW*

| ‘OLD’ | ‘NEW’ |
| --- | --- |
| Almacena en el registro los valores originales (de las columnas) de la fila que está procesando el trigger | Contiene los valores originales de las columnas que no se modifican en la sentencia ‘DML’ y los nuevos valores de las columnas que se modifican de la fila |
| Características | |
| * ‘OLD’ y ‘NEW’ tienen la misma estructura que un registro que se declara usando ‘%ROWTYPE’ con la tabla asociada al trigger. * ‘OLD’ y ‘NEW’ se pueden usar como prefijos para referenciar el valor de una columna antes y después del cambio de dato. * ‘OLD’ y ‘NEW’ se pueden utilizar sólo en trigger a Nivel de Filas. * Deben ir precedidos por “**:”** (dos puntos) en cada sentencia PL/SQL o SQL. * No se deben utilizar “**:**” (dos puntos) como prefijo si son referenciados en la condición de restricción WHEN. * ‘:NEW’ no se puede modificar en un trigger ‘AFTER’ a nivel de fila. * :OLD’ nunca se puede modificar, sólo se puede leer. | |

A continuación, se detallan los valores OLD y NEW para cada tipo de operación:

Tabla 2

*Valores OLD y NEW*



# Crear un trigger a nivel de fila

Existen dos opciones para crear un trigger a nivel de fila usando SQL Developer:

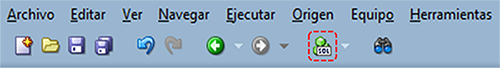
**Opción 1: usando una hoja de trabajo de SQL**

Debes abrir una hoja de trabajo de SQL y escribir el código; una vez finalizado el código del trigger, se debe elegir la opción “Ejecutar Script”.

Figura 5

*Creación de un* *Trigger a nivel de fila usando una hoja de trabajo*

**Paso 1**

**

**Paso 2**

*Nota.* Ejemplo de cómo crear un trigger a nivel de fila usando SQL Developer. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

Importante

En la base de datos solo se guarda el código fuente del trigger, por lo tanto, si el código no posee errores, el trigger quedará almacenado en la base de datos, pero si el código tiene errores, entonces el trigger no se puede crear.

**Opción 2:** **usando la interfaz gráfica de SQL Developer**

Selecciona “Disparadores” y con el botón derecho del mouse selecciona “Nuevo Disparador”. Se abrirá una ventana en donde se le puede dar el nombre al trigger, elegir a qué tabla estará asociado, el tiempo en que se activará, la sentencia DML que lo activará y desmarcar la opción “Nivel de Sentencia”.

Posteriormente al dar clic en “Aceptar” se abrirá el área de trabajo para escribir el código del trigger. Una vez que se ha finalizado se debe elegir la opción “Compilar” para que el trigger quede almacenado en la base de datos. Si el trigger no tiene errores, quedará almacenado en la base de datos el código fuente del trigger. Si existen errores, no se podrá crear.

Figura 6

*Creación de un trigger a nivel de fila usando la interfaz gráfica*

**Paso 1**

**Paso 2**

**Paso 3**

**Paso 4**

*Nota.* Ejemplo de creación de trigger a nivel de fila usando SQL Developer. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

En el siguiente ejemplo, el trigger TRG\_AUDIT\_EMP se activará después de que una sentencia de ‘DELETE’, ‘INSERT’ o ‘UPDATE’ se ejecute sobre la tabla EMPLEADOS. Cuando alguna de las operaciones afecte a la tabla, se insertará en la tabla AUDIT\_EMP los valores que existían en la tabla EMPLEADOS (:OLD) y los nuevos valores que se desean insertar, actualizar o eliminar en la tabla (:NEW).

Figura 7

*Ejemplo de trigger a nivel de fila*

CREATE OR REPLACE TRIGGER TRG\_AUDIT\_EMP

AFTER INSERT OR UPDATE OR DELETE ON employees

FOR EACH ROW

BEGIN

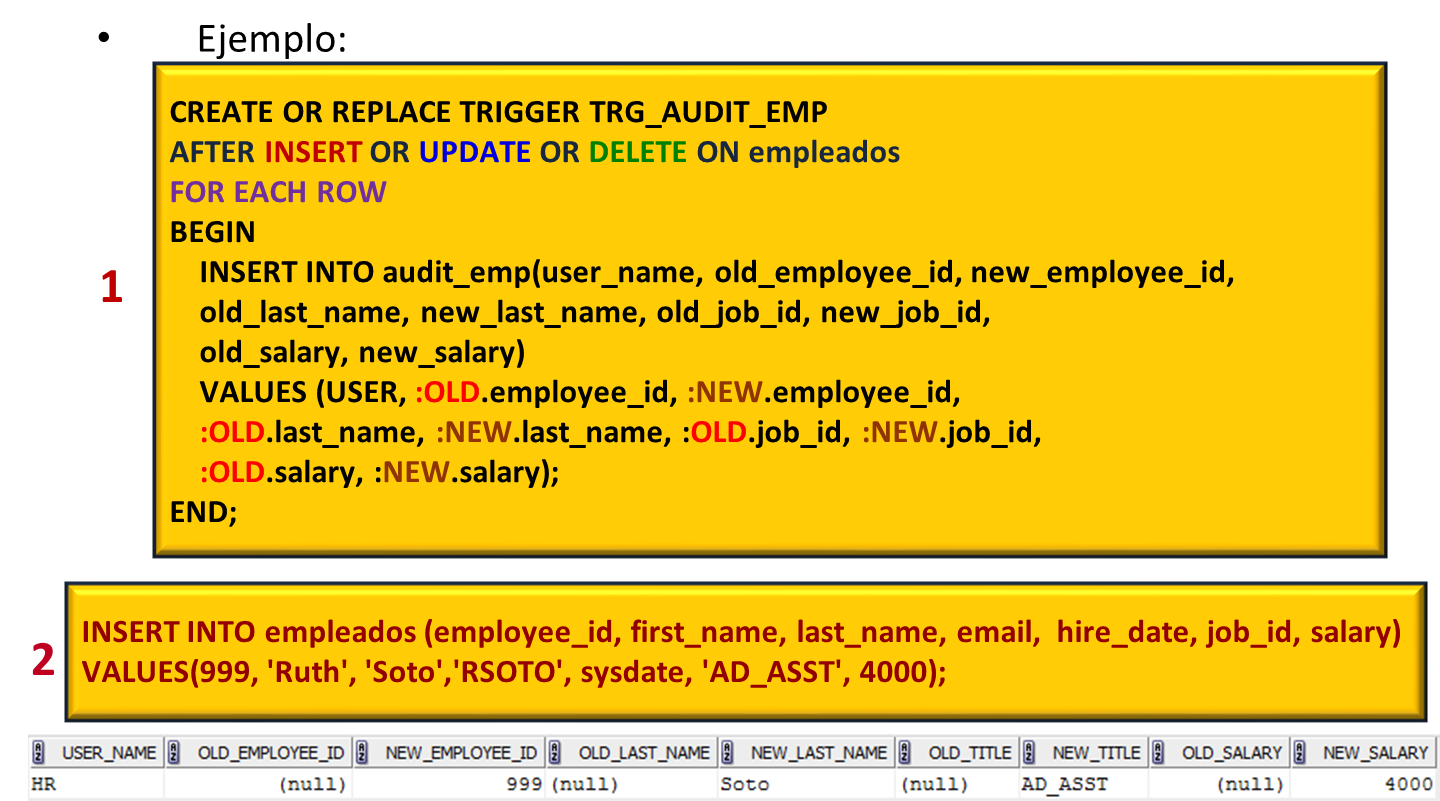
INSERT INTO audit\_emp(user\_name, old\_employee\_id, new\_employee\_id, old\_last\_name, new\_last\_name, old\_job\_id, new\_job\_id, old\_salary, new\_salary)

VALUES (USER, OLD.employee\_id, : NEW.employee\_id, :OLD.last\_name, : NEW.last\_name, : OLD.job\_id, : NEW.job\_id, :OLD.salary, : NEW.salary);

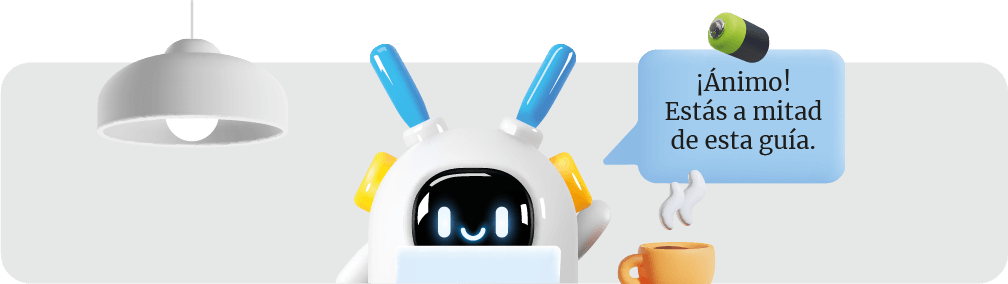
END;

INSERT INTO empleados (employee\_id, first\_name, last\_name, email, hire\_date, job\_id, salary)

VALUES (999, 'Ruth', 'Soto', 'RSOTO', SYSDATE, 'AD\_ASST', 4000);



*Nota.* Ejemplo de trigger a nivel de fila. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)



A continuación, se inserta un nuevo empleado en la tabla EMPLEADOS y el trigger TRG\_AUDIT\_EMP se activa insertando los valores que existían en la tabla (OLD) y los nuevos valores (NEW) que se insertaron. Como en este caso es una nueva fila la que se inserta, todos los OLD son nulos porque la fila no existe en la tabla EMPLEADOS.

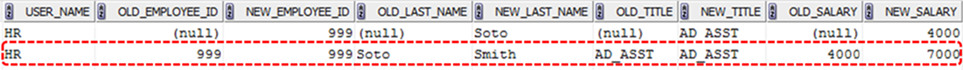
Figura 8

*Ejemplo de trigger a nivel de fila*

UPDATE empleados

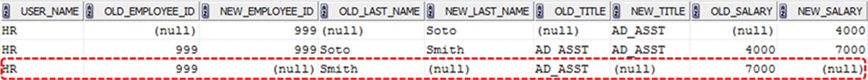
SET salary = 7000, last\_name = ‘Smith’

WHERE employee\_id = 999;



DELETE empleados

WHERE employee\_id = 999;



*Nota.* Ejemplo de trigger a nivel de fila. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

Después, se actualiza el salario y apellido del nuevo empleado y finalmente se elimina el empleado. En estas dos operaciones, el trigger TRG\_AUDIT\_EMP se ejecuta insertando los valores que existían en la tabla (OLD) y los nuevos valores (NEW) que actualizaron y eliminaron.

Figura 9

*Ejemplo de trigger a nivel de fila*

CREATE OR REPLACE TRIGGER TRG\_U\_PRESTAMO\_SOCIAL

BEFORE UPDATE ON PRESTAMO\_SOCIAL

FOR EACH ROW

BEGIN

IF :NEW.fecha\_pago > :OLD.fecha\_venc\_prestamo THEN

:NEW.multa := ROUND(:OLD.valor\_prestamo \* ((:NEW.fecha\_pago - :OLD.fecha\_venc\_prestamo) / 100));

:NEW.estado\_pago := 'El valor de la multa corresponde a ' ||

(TO\_DATE(TO\_CHAR(:NEW.fecha\_pago,'dd/mm/yyyy')) -

TO\_DATE(TO\_CHAR(:OLD.fecha\_venc\_prestamo,'dd/mm/yyyy'))) ||

' días de atraso. La multa corresponde a un ' ||

(:NEW.fecha\_pago - :OLD.fecha\_venc\_prestamo) ||

' % del valor del préstamo';

END IF;

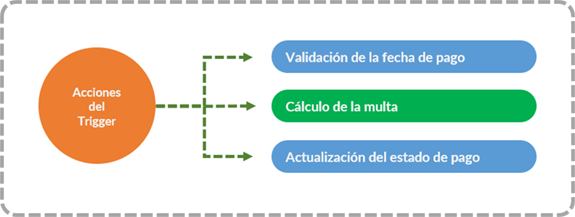
END;

*Nota.* Ejemplo de trigger a nivel de fila. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

Como pudiste ver, se crea el trigger a nivel de fila TRG\_U\_PRESTAMO\_SOCIAL, que se activará antes de que se realice un UPDATE sobre la tabla PRESTAMO\_SOCIAL. Como este es un trigger a nivel de fila, se debe usar la cláusula FOR EACH ROW.

Para hacer referencia a los nuevos valores de las columnas que se están actualizando en la sentencia UPDATE y a los nuevos valores que tendrán las columnas en la tabla, se usa ‘:NEW’, mientras que para hacer referencia a los valores que las columnas tienen en la tabla, se usa OLD.

Por lo tanto, las acciones del trigger serán las siguientes:



* **Validación de la fecha de pago:** el trigger verificará si la nueva fecha de pago (‘:NEW.fecha\_pago’) es mayor que la fecha de vencimiento del préstamo (‘:OLD.fecha\_venc\_prestamo’) almacenada en la tabla.
* **Cálculo de la multa:** si la condición anterior se cumple, se calculará el valor de la multa. Esto se hace restando la nueva fecha de pago de la fecha de vencimiento del préstamo. La cantidad de días de atraso obtenida se utilizará como el porcentaje de la multa. El nuevo valor asignado a la columna ‘multa’ será el resultado de multiplicar el valor del préstamo (‘:OLD.valor\_prestamo’) por el porcentaje calculado.
* **Actualización del estado de pago:** la columna ‘estado\_pago’ se actualizará con un mensaje que indicará el valor de la multa y los días de atraso. El mensaje seguirá el formato: "El valor de la multa corresponde a [días\_de\_atraso\_calculado] días de atraso. La multa corresponde a [días\_de\_atraso\_calculado] % del valor del préstamo".

Figura 10

*Código de actualización de préstamos*

UPDATE prestamo\_social

SET fecha\_pago = ADD\_MONTHS(fecha\_prestamo, 1)

WHERE nro\_prestamo IN (1, 2);

UPDATE prestamo\_social

SET fecha\_pago = ADD\_MONTHS(fecha\_venc\_prestamo, 1) + 1

WHERE nro\_prestamo IN (3, 4);

UPDATE prestamo\_social

SET fecha\_pago = fecha\_venc\_prestamo + 1

WHERE nro\_prestamo IN (5);

COMMIT;





*Nota.* Ejemplo de trigger a nivel de fila. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

Al realizar la actualización de las fechas de pago en la tabla PRESTAMO\_SOCIAL, se activará el trigger creado anteriormente y actualizará las columnas ‘multa’ y ‘estado\_pago’ de acuerdo a los días de atraso que calculó con las nuevas fechas de pagos que se actualizaron de los préstamos del ejemplo.

Importante

El trigger realizará el cálculo según la fecha en que se ejecute el script de poblado de la tabla PRESTAMO\_SOCIAL.

Figura 11

*Código del trigger*

CREATE OR REPLACE TRIGGER TRG\_U\_PRESTAMO\_SOCIAL

BEFORE UPDATE ON PRESTAMO\_SOCIAL

FOR EACH ROW

BEGIN

IF:NEW.fecha\_pago > : OLD.fecha\_venc\_prestamo THEN

:NEW.multa : ROUND(:OLD.valor\_prestamo \* ((:NEW.fecha\_pago: OLD.fecha\_venc\_prestamo)))

: NEW.estado\_pago:=

'El valor de la multa corresponde a ‘ ||

(TO\_DATE(TO\_CHAR(:NEW.fecha\_pago, 'dd/mm/yyyy’))

TO\_DATE(TO\_CHAR(: OLD.fecha\_venc\_prestamo, 'dd/mm/yyyy’))) ||

‘días de atraso. La multa corresponde a un’ ||

(:NEW.fecha\_pago : OLD.fecha\_venc\_prestamo) ||

‘% del valor del préstamo’;

END IF;

END;

*Nota.* Ejemplo de trigger a nivel de fila. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

En el ejemplo, se crea el trigger a nivel de fila **TRG\_ACTUALIZA\_STOCK** que se activará después de que se realice un ‘INSERT’, ‘UPDATE’ o ‘DELETE’ sobre la tabla **DETALLE\_BOLETA.**

Las acciones que realizará el trigger serán las siguientes:

* **Inserción de fila:** si se está insertando una fila en la tabla, el trigger actualizará la tabla STOCK, disminuyendo del stock del producto la cantidad que se está comprando.
* **Actualización de fila:** si se está actualizando una fila, el trigger actualizará el stock del producto con la nueva cantidad del producto que se está actualizando. Para esto, primero vuelve a sumar al stock del producto la cantidad anterior que se había comprado y luego le resta la nueva cantidad del producto que se está actualizando.
* **Eliminación de fila:** si se está eliminando una compra, el trigger actualizará el stock del producto sumándole la cantidad de la compra que se está eliminando.

Ahora, veamos el siguiente ejemplo donde se inserta una nueva compra con la boleta 105. Con esta boleta se compra el producto 2000, por lo tanto, el trigger rebajó el stock de este producto según la cantidad vendida.

Figura 12

*Tabla STOCK antes de que se active el trigger TRG\_ACTUALIZA\_STOCK*

*Tabla

Descripción generada automáticamenteNota.* Ejemplo de trigger a nivel de fila. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

INSERT INTO boleta

VALUES (105, SYSDATE, 225000, 111111);

INSERT INTO detalle\_boleta

VALUES (105, 2000, 150, 1500, 225000);

COMMIT;

UPDATE detalle\_boleta

SET cantidad = 10,

valor\_total = 18000

WHERE nro\_boleta = 100

AND cod\_producto = 3000;

COMMIT;

DELETE FROM detalle\_boleta

WHERE nro\_boleta = 101;

DELETE FROM boleta

WHERE nro boleta = 101;

COMMIT;

Después de modificar la cantidad vendida del producto 3000 de la boleta 100, el trigger actualizó el stock del producto 3000 según la nueva cantidad. Finalmente se elimina la boleta 101, por lo tanto, el trigger aumentó el stock de los productos que se compraron con la boleta eliminada.

Siguiendo con el ejemplo, se crea la función almacenada FN\_OBT\_VALOR\_COM que obtiene el porcentaje de comisión de la tabla TRAMO\_PORC\_COMISION según el salario del empleado.

Figura 13

*Trigger para asignación de porcentaje por comisión*

CREATE OR REPLACE FUNCTION FN\_OBT\_VALOR\_COM(

p\_sueldo\_base NUMBER,

p\_id\_emp NUMBER) RETURN NUMBER IS

v\_valor\_com NUMBER(4,3);

v\_mensaje\_error VARCHAR2(200);

BEGIN

SELECT tpc.porc\_com

INTO v\_valor\_com

FROM tramo\_porc\_comision tpc

WHERE p\_sueldo\_base BETWEEN tpc.rango\_inf\_pc AND tpc.rango\_sup\_pc;

RETURN v\_valor\_com;

EXCEPTION

WHEN OTHERS THEN

v\_mensaje\_error := SQLERRM;

INSERT INTO error\_proceso

VALUES (seq\_error\_proc.NEXTVAL,

'Error en la Función FN\_OBT\_VALOR\_COM. Empleado: ‘ || p\_id\_emp,

v\_mensaje\_error);

RETURN 0;

END FN\_OBT\_VALOR\_COM;

*Nota.* Ejemplo de trigger a nivel de fila. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

A continuación, se crea el trigger TRG\_IU\_ASIG\_PORC\_COM asociado a la tabla EMPLEADOS, que se activará cuando se inserte una fila o se actualice la columna ‘salary’ de la tabla. Si el trabajo del empleado es SA\_REP, el porcentaje de comisión será 30%, mientras que si el trabajo del empleado es diferente a SA\_REP, el porcentaje de comisión se obtiene a través de la función almacenada FN\_OBT\_VALOR\_COM.

Figura 14

*Código del trigger TRG\_IU\_ASIG\_PORC\_COM*

CREATE OR REPLACE TRIGGER TRG\_IU\_ASIG\_PORC\_COM

BEFORE INSERT OR UPDATE OF salary ON empleados

FOR EACH ROW

DECLARE

v\_porc\_com NUMBER (4,3);

BEGIN

IF NEW.job\_id = 'SA\_REP’ THEN

:NEW.commission\_pct := .30;

ELSE

v\_porc\_com := FN\_OBT\_VALOR\_COM(:NEW.salary, : NEW.employee\_id);

:NEW.commission\_pct := v\_porc\_com;

END IF;

END;

INSERT INTO empleados (employee\_id, first\_name, last\_name, email, hire\_date, job\_id, salary);

VALUES (997, 'Alexander', 'Biber', 'ABIBER', SYSDATE, 'SH CLERK', 5000);

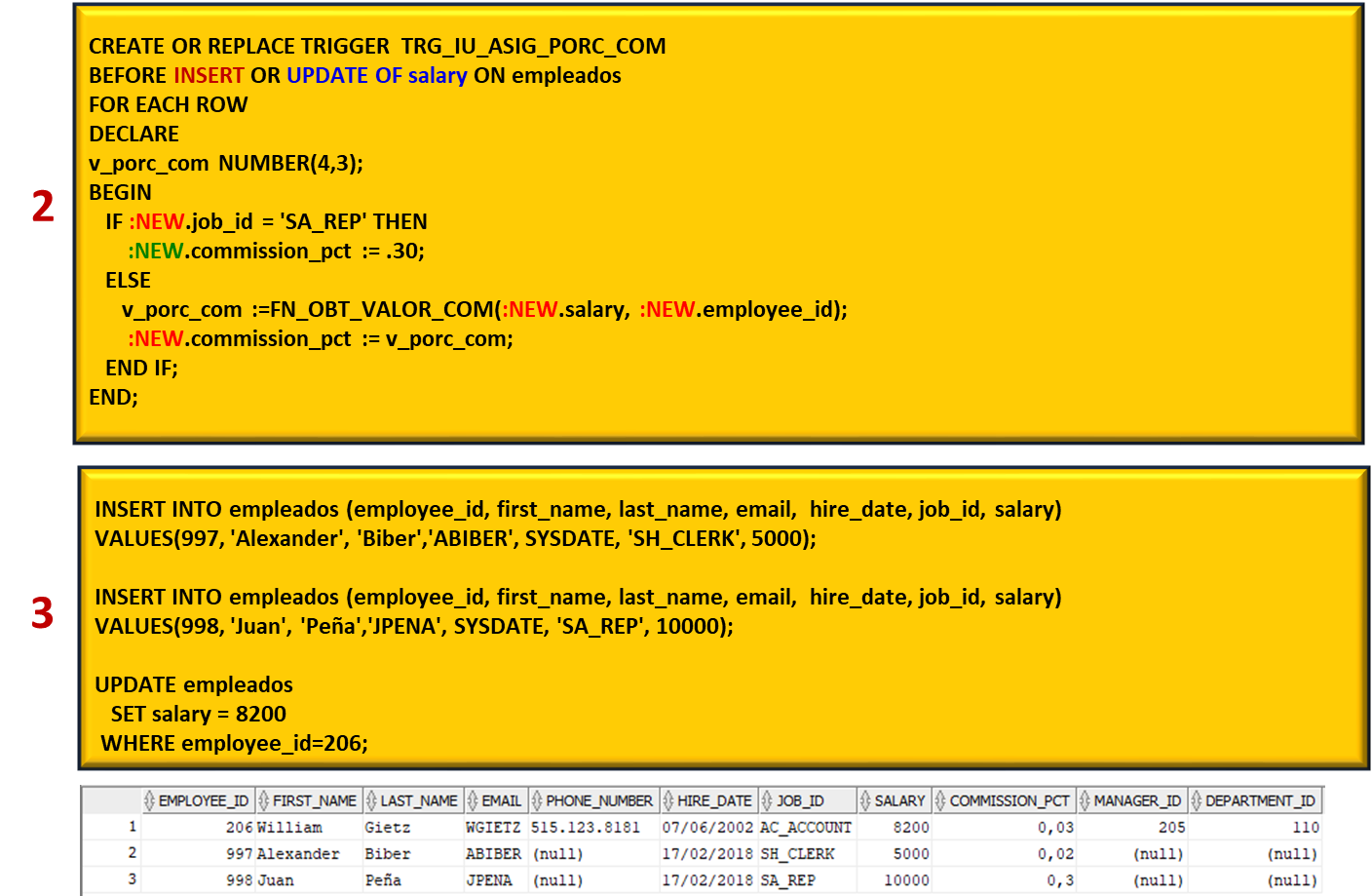
INSERT INTO empleados (employee\_id, first\_name, last\_name, email, hire\_date, job\_id, salary);

VALUES (998, 'Juan', 'Peña', 'JPENA', SYSDATE, 'SA REP', 10000);

UPDATE empleados

SET salary = 8200

WHERE employee\_id = 206;



*Nota.* Ejemplo de trigger a nivel de fila. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

## Usando restricciones en un trigger a nivel de fila

Opcionalmente, se puede incluir una restricción en la definición de un trigger a nivel de fila especificando una expresión SQL booleana en una cláusula ‘WHEN’. Si se incluye una cláusula ‘WHEN’ en el trigger, entonces la expresión en la cláusula ‘WHEN’ se evalúa para cada fila afectada por el trigger.

Si la expresión se evalúa como ‘TRUE’ para una fila, entonces el cuerpo del trigger se ejecuta para esa fila. Sin embargo, si la expresión se evalúa como FALSO o NO ES VERDADERA para una fila, el cuerpo del trigger no se ejecuta para esa fila.

La evaluación de la cláusula ‘WHEN’ no tiene efecto sobre la ejecución de la sentencia SQL que activó el trigger, es decir, que esta sentencia no se revierte si la expresión en una cláusula ‘WHEN’ se evalúa como FALSO.

Una cláusula ‘WHEN’ no se puede incluir en la definición de un trigger a nivel de sentencia.

En el siguiente ejemplo, el trigger TRG\_IU\_ASIG\_COM\_JOB\_SA\_REP se ejecutará antes de insertar una nueva fila o de actualizar la columna ‘salary’ en la tabla EMPLEADOS solo si el trabajo es igual a SA\_REP. Si la operación realizada es la inserción de una fila en la tabla EMPLEADOS, entonces se le asignará al porcentaje de comisión el valor cero, y si es una actualización, el porcentaje de comisión será igual al porcentaje actual: más 0.05.

Figura 15

*Restricción de trigger TRG\_IU\_ASIG\_COM\_JOB\_SA\_REP*

CREATE OR REPLACE TRIGGER TRG IU ASIG COM JOB SA REP

BEFORE INSERT OR UPDATE OF salary ON empleados

FOR EACH ROW

WHEN (NEW.job\_id = 'SA\_REP’)

BEGIN

IF INSERTING THEN

:NEW.commission\_pct := 0;

ELSE

:NEW.commission\_pct := NVL(:OLD.commission\_pct, 0) + 0.05;

END IF;

END;

* **Inserción de un nuevo empleado:**

INSERT INTO empleados

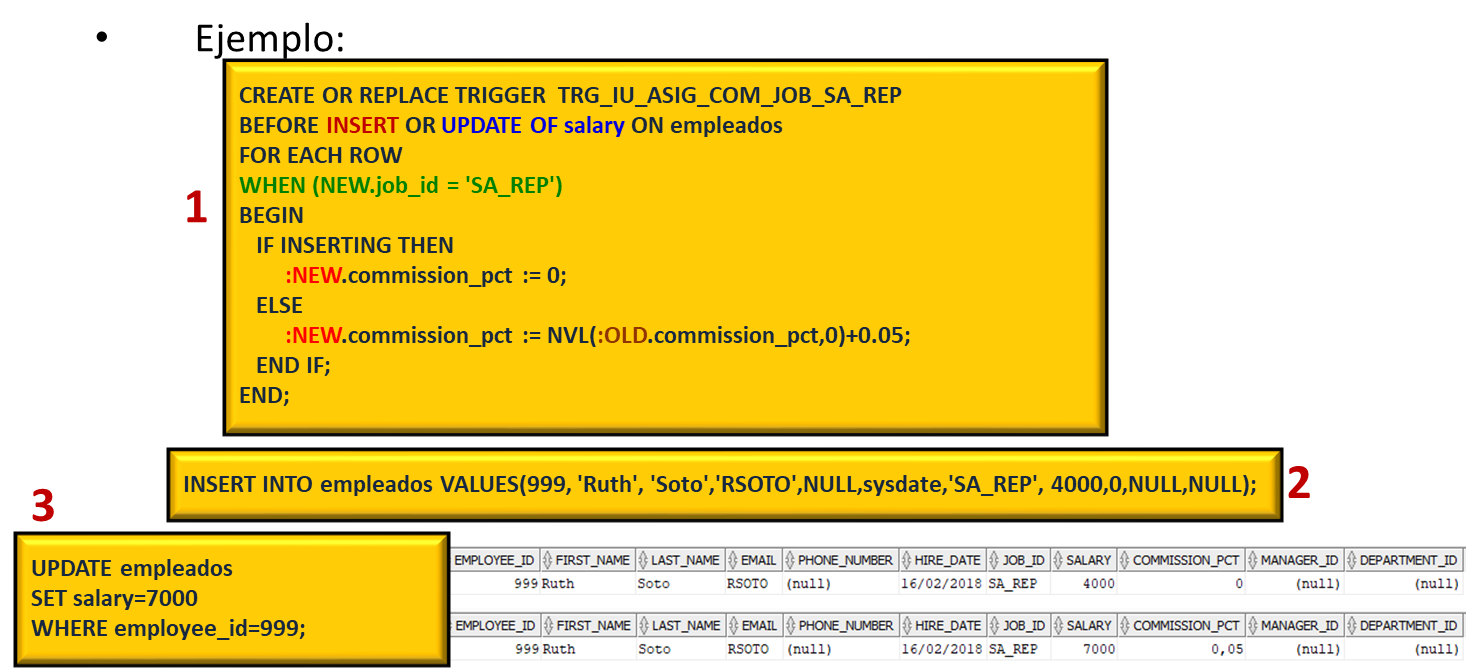
VALUES (999, 'Ruth', 'Soto', 'RSOTO', NULL, SYSDATE, 'SA\_REP', 4000, 0, NULL, NULL);

* **Inserción de un nuevo empleado:**

UPDATE empleados

SET salary = 7000

WHERE employee\_id = 999;



*Nota.* Ejemplo de restricciones en un trigger a nivel de fila. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. <https://www.oracle.com>

# Video

En el siguiente video, podrás revisar cómo se implementa un trigger: <https://www.youtube.com/watch?v=N0ckjGmbH80>

# Eliminar un trigger

Para eliminar un trigger en PL/SQL, existen tres opciones principales:

1. Se puede eliminar un trigger de manera explícita utilizando la sentencia **SQL DROP TRIGGER:**

DROP TRIGGER nombre\_trigger;

**En la sintaxis:**

* **nombre\_trigger**: es el nombre del trigger que se desea eliminar.

1. Puedes eliminar un trigger de forma gráfica. Para hacerlo, selecciona el trigger que deseas eliminar, haz clic derecho y elige la opción "Borrar Disparador" (o equivalente en inglés, "Drop Trigger"). Esto abrirá una ventana de confirmación donde puedes proceder a eliminar el trigger.

* Un trigger se elimina automáticamente cuando se elimina la tabla asociada al trigger.

Figura 16

*Eliminar un trigger*

DROP TRIGGER TRG\_U\_PRESTAMO\_SOCIAL ;

**Paso 1**

**Paso 2**

*Nota.* Ejemplo de cómo eliminar un Trigger. Oracle. (2023). *Oracle SQL Developer 23.1.1.* [Software]. [https://www.oracle.com](https://www.oracle.com/)

En el ejemplo se elimina el trigger TRG\_U\_PRESTAMO\_SOCIAL.

# Videos

Si deseas seguir aprendiendo y profundizando en el uso de triggers, te recomendamos ver los siguientes videos realizados por Bilbao Labs:

* TRIGGERS en PL/SQL – ORACLE:

<https://www.youtube.com/watch?v=N0ckjGmbH80>

* TRIGGERS con más opciones en PL/SQL – ORACLE:

<https://www.youtube.com/watch?v=zoUAEBeE6YE&t=91s>

# Cierre de la semana

Esta semana profundizamos en el uso de triggers, esenciales para mantener la integridad de los datos y automatizar tareas. Exploramos los triggers a nivel de fila y de sentencia, los tipos de eventos que los activan, y cómo crear un trigger a nivel de sentencia de manera efectiva. En predicados condicionales, vimos cómo añadir lógica condicional a nuestros triggers, y entendimos el uso de OLD y NEW en triggers a nivel de fila. Practicamos la creación de un trigger a nivel de fila, aplicando restricciones para asegurar condiciones específicas antes de ejecutar acciones, y aprendimos cómo eliminar un trigger cuando es necesario modificar su comportamiento.

# Referencias

* Bilbao Labs (2024) PACKAGE PL/SQL – ORACLE 1 [Video]. Youtube. <https://www.youtube.com/watch?v=g3yKCCCP-rQ>
* Bilbao Labs (2024) PACKAGE PL/SQL – ORACLE 2 [Video]. Youtube. <https://www.youtube.com/watch?v=wfyzb-cEQfQ>
* Bilbao Labs (2024) TRIGGERS en PL/SQL– ORACLE [Video]. Youtube. <https://www.youtube.com/watch?v=N0ckjGmbH80>
* ORACLE (2024) PL/SQL [enlace] <https://www.oracle.com/cl/database/technologies/appdev/plsql.html>
* ORACLE (2024) Cloud. [enlace] <https://academy.oracle.com/en/solutions-cloud.html>
* ORACLE (2024) Gimnasio para desarrolladores. [enlace] <https://www.oracle.com/cl/database/technologies/appdev/plsql.html>
* ORACLE (2024) ORACLE Apex. [enlace] <https://apex.oracle.com/pls/apex/f?p=4550:1:110831667031748>

# Lecturas de la semana

* Capítulo 10: Disparadores o Triggers

Muñoz Chaparro, A. (2014). Oracle 12c PL/SQL1: Curso práctico de formación, Alpha Editorial. <https://webezproxy.duoc.cl/login?url=http://biblioteca.duoc.cl/bdigital/elibros/a41214-Oracle%2012c%20PL%20SQL/246/> Páginas 243 a 261

# Apuntes

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



Reservados todos los derechos Fundación Instituto Profesional Duoc UC. No se permite copiar, reproducir, reeditar, descargar, publicar, emitir, difundir, de forma total o parcial la presente obra, ni su incorporación a un sistema informático, ni su transmisión en cualquier forma o por cualquier medio (electrónico, mecánico, fotocopia, grabación u otros) sin autorización previa y por escrito de Fundación Instituto Profesional Duoc UC La infracción de dichos derechos puede constituir un delito contra la propiedad intelectual.