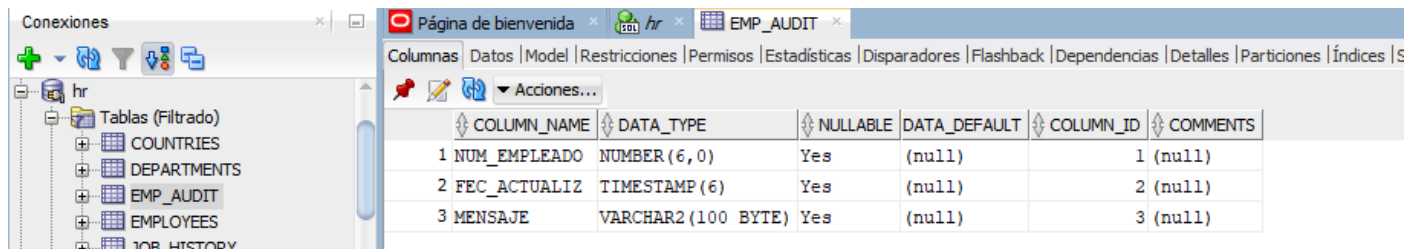


Actividad 12. Tarea Individual. Trigger

La actividad la comenzaremos implementando sentencia SQL que nos construye la tabla donde vamos a guardar los datos cuando se dispare el trigger. El enunciado nos dice que tiene tres campos, el identificador del empleado(employee_id), el momento en el cual se hace la actualización y otro campo que contiene un mensaje con el salario anterior y el salario nuevo. Para ello construimos la siguiente sentencia:

```
CREATE TABLE EMP_AUDIT (  
    num_empleado      NUMBER(6),  
    fec_actualiz       TIMESTAMP, /*El Formato de esta fecha de actualizacion lo queremos con toda  
    la informacion posible, dia,mes,año,hora,segundos.*/  
    mensaje            VARCHAR2(100) /*A esta columna que almacenará un mensaje le damos suficiente longitud  
    para que el mensaje nos entre en ella*/  
);
```



The screenshot shows the Oracle SQL Developer interface. On the left, the 'Conexiones' pane shows a tree view with 'hr' selected. The main pane shows the 'EMP_AUDIT' table structure. The table has three columns: NUM_EMPLEADO (NUMBER(6,0)), FEC_ACTUALIZ (TIMESTAMP(6)), and MENSAJE (VARCHAR2(100 BYTE)).

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 NUM_EMPLEADO	NUMBER(6,0)	Yes	(null)	1 (null)	
2 FEC_ACTUALIZ	TIMESTAMP(6)	Yes	(null)	2 (null)	
3 MENSAJE	VARCHAR2(100 BYTE)	Yes	(null)	3 (null)	

Cuando ya tenemos nuestra tabla construida donde meteremos los datos que queremos cuando se produzca una actualización del salario en productos y se dispare el trigger, pues construimos nuestro disparador.

```
-- DDL for Trigger EMPLOYEES_BU
```

```
CREATE OR REPLACE EDITIONABLE TRIGGER "HR"."EMPLOYEES_BU" BEFORE /*En esta primera linea,  
definimos el nombre del trigger, por buenas practicas  
y para que el usuario que lo utilice sepa que funcion hacel el trigger, lo llamamos como la  
tabla que se va a modificar seguido de  
dos letras, la primera indica cuando se va a desencadenar las acciones de este disparador y la  
segunda, con que sentencia se va a llevar a cabo.  
Por lo tanto, sabemos que este disparador se tiene que disparar antes de(BEFORE(B)) la  
actualizacion de los datos del salario del empleado(UPDATE(U)).  
Despues del nombre del trigger irán precisamente estas dos palabras, cuando se dispara AFTER o  
BEFORE y con que sentencia lo hace(UPDATE,INSERT,DELETE).*/
```

```
UPDATE ON employees --Nosotros tenemos que dispararlo antes de que se realice la  
actualizacion de los registros.
```

```
FOR EACH ROW --Esta sentencia significa que el trigger se dispara una vez por cada  
registro actualizado en la tabla employees
```

```
BEGIN --Aqui empiezan las condiciones que deben de cumplirse para que nuestro trigger se  
dispare.  
/*Tal como dice el enunciado, solo se disparará y copiara la informacion que queremos a la nueva  
tabla si el salario introducido es distinto  
que el salario en ese momento del empleado. Por lo tanto aqui podemos hacer un condicional if  
comparando de la siguiente manera el salario  
nuevo con el salario antiguo. El salario nuevo lo pilla de los datos que tu metes en el  
update.*/
```

Actividad 12. Tarea Individual. Trigger

```
IF :old.salary = :new.salary THEN
/*Si el salario es el mismo, hacemos que salte el procedimiento llamado
raise_application_error que nos sirve para levantar errores
y definir con un mensaje el error*/
    raise_application_error(-
        20600,
        ' El nuevo salario es igual que el anterior '
        || :new.salary
        || '='
        || :old.salary);

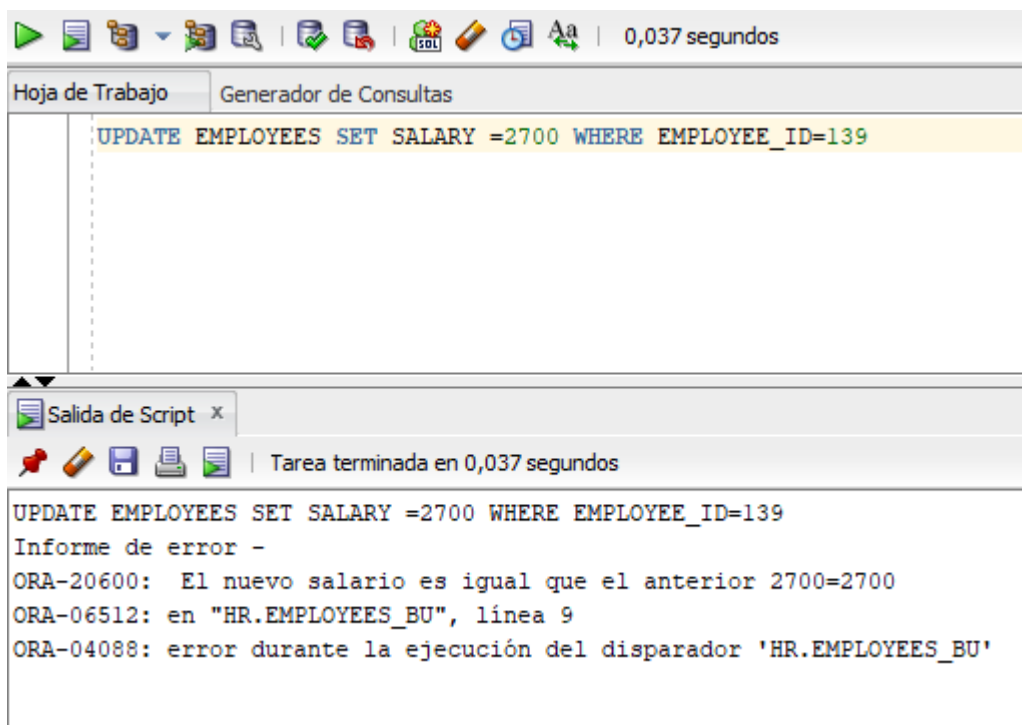
ELSE
/*Si el salario nuevo es distinto del antiguo, entonces introducimos en nuestra tabla creada
los valores que nos interesan
segun el enunciado(numero de empleado, momento de la actualizacion y un mensaje donde se
especifica el salario viejo y el nuevo*/

    INSERT INTO emp_audit VALUES (
:old.employee_id,
sysdate,
'El salario anterior era '
|| :old.salary
|| ' y el salario nuevo es '
|| :new.salary
);

END IF; --Finalizamos el condicional if
END; --finalizamos la programacion de nuestro trigger
/
ALTER TRIGGER "HR"."EMPLOYEES_BU" ENABLE; --Esta sentencia es para activar nuestro disparador
```

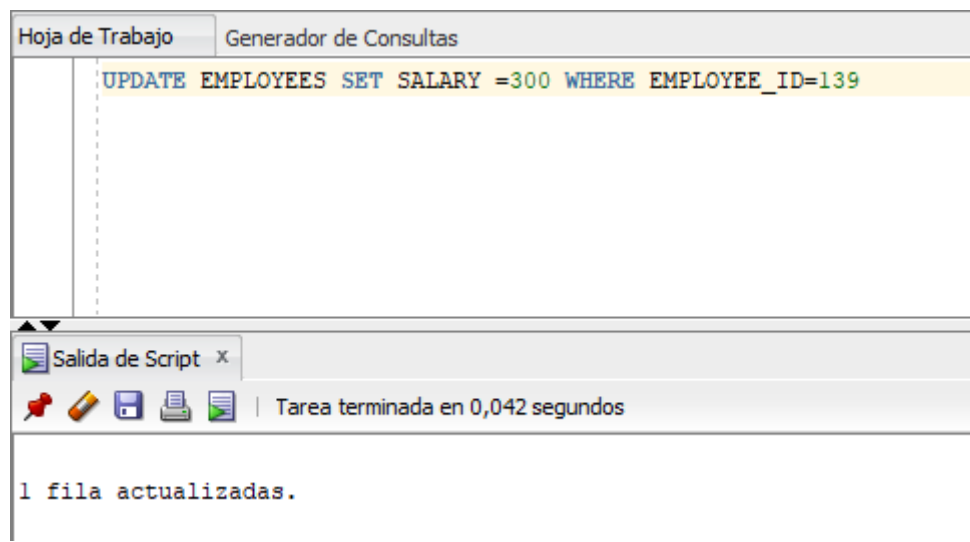
EJECUCIÓN CASOS PROBABLES







1-Actualizacion de un registro salario con el mismo valor que el que tiene el empleado en ese momento.



Actividad 12. Tarea Individual. Trigger

2- Actualizacion de un registro salario distinto valor que el que tiene el empleado en ese momento.



Columnas Datos Model Restricciones Permisos Estadísticas Disparadores Flashback Dependencias Detalles Particiones Índices SQL			
      Ordenar... Filtrar: <input type="text"/>			
	NUM_EMPLEADO	FEC_ACTUALIZ	MENSAJE
1	139	27/02/21 12:55:52,000000000	El salario anterior era 2700 y el salario nuevo es 300

En el .zip del proyecto se adjunta los archivos .sql de creación de la tabla para insertar los registros y el trigger para la comprobación en caso de ser necesario.