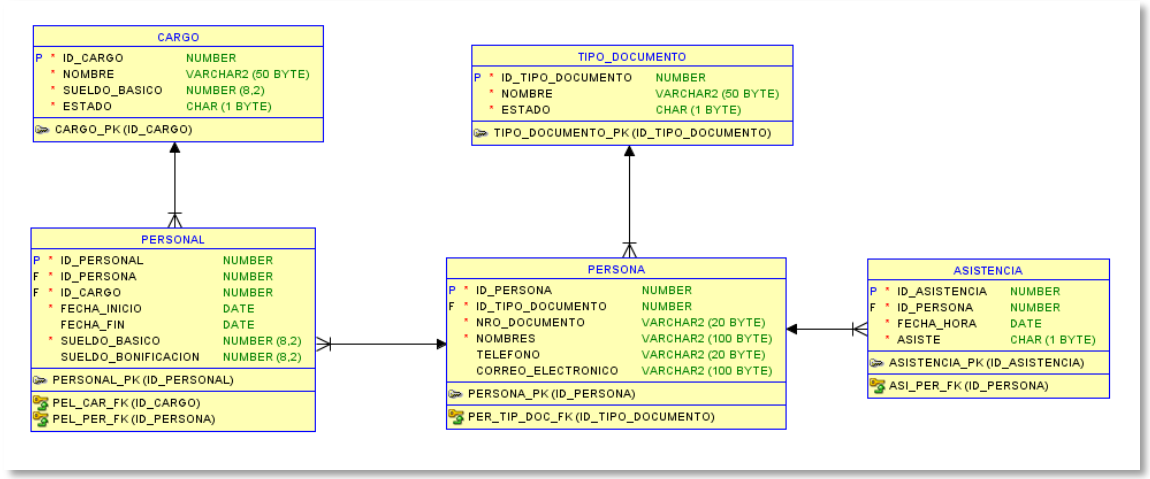
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

Bases de Datos
5ta. Práctica Dirigida
(Segundo Semestre 2021)

CURSORES – TRIGGERS

Tema propuesto:

La empresa Rent SAC es una empresa emergente (startup) recién formada que ha conseguido financiamiento y colaboración de personas de distintas partes del mundo. Desea mantener el registro de su personal en un Sistema de Recursos Humanos elaborado por su propio personal. Para este fin, se elaboró una base de datos cuyo modelo se muestra a continuación:



Ejecute el contenido del archivo **INF246_2021-2_LAB5_00_drops.sql** para limpiar las tablas, si alguna sentencia del drop falla, no hay problema, eso se debe a que no existe la tabla a borrar.

A continuación, ejecute el contenido del archivo **INF246_2021-2_LAB5_01_ddl.sql** para crear las tablas de la figura anterior. Luego ejecutar el contenido del archivo **INF246_2021-2_LAB5_02_dml.sql** para poblar las tablas.

Ejercicios cursores:

1.- Se pide realizar un procedimiento con nombre **SP_LISTAR_EMPLEADOS**, que usando cursores liste los tipos documentos, personas y cargo en el siguiente formato:

```
TIPO DOCUMENTO: 1 - DNI
   PERSONA: 1 - RODRIGUEZ ORTEGA, DANIEL MANUEL
      CARGO: 1 - Gerente General
   PERSONA: 2 - CHAVEZ TAVERA, BRUNO MARTIN
      CARGO: 2 - Gerente de Operaciones
   PERSONA: 3 - SANCHEZ CARLOS, JULIO ROBERTO
      CARGO: 3 - Administrador
```

```
   PERSONA: 6 - PISCONTE DE LA CRUZ, SANDRA VIVIANA
      CARGO: 7 - Jefe de Proyectos
….
```

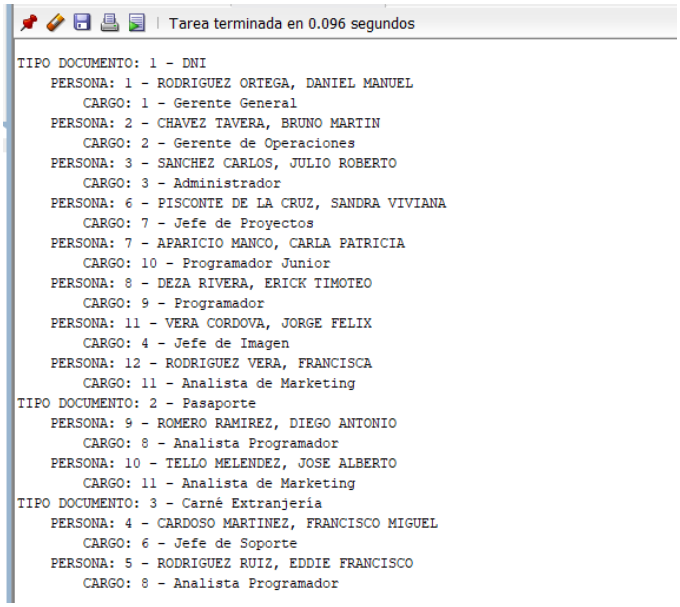
Notas:

- Utilizar **un cursor** para listar los tipos documentos, **un segundo cursor** para las personas (que recibe como parámetro el **id tipo documento**), y **un tercer cursor** para los cargos del personal (que recibe como parámetro el **id de la persona**).

```
CREATE OR REPLACE PROCEDURE SP_LISTAR_EMPLEADOS IS

  CURSOR C_TIPODOCUMENTO IS SELECT ID_TIPO_DOCUMENTO, NOMBRE FROM
TIPO_DOCUMENTO ORDER BY 1;
  CURSOR C_PERSONA (P_ID_TIPO_DOCUMENTO NUMBER) IS SELECT ID_PERSONA,
NOMBRES FROM PERSONA WHERE ID_TIPO_DOCUMENTO = P_ID_TIPO_DOCUMENTO ORDER
BY 1;
  CURSOR C_PERSONAL (P_ID_PPERSONA NUMBER) IS SELECT C.ID_CARGO, C.NOMBRE
FROM PERSONAL P,CARGO C WHERE P.ID_CARGO = C.ID_CARGO AND P_ID_PPERSONA =
P.ID_PERSONA ORDER BY 1;

  V_ID_TIPO_DOCUMENTO TIPO_DOCUMENTO.ID_TIPO_DOCUMENTO%TYPE;
  V_NOMBRE_TIPODOCUMENTO TIPO_DOCUMENTO.NOMBRE%TYPE;

  V_ID_PERSONA PERSONA.ID_PERSONA%TYPE;
  V_NOMBRES_PERSONA PERSONA.NOMBRES%TYPE;

  V_ID_CARGO CARGO.ID_CARGO%TYPE;
  V_NOMBRE_CARGO CARGO.NOMBRE%TYPE;
BEGIN

  OPEN C_TIPODOCUMENTO;
  LOOP
    FETCH C_TIPODOCUMENTO INTO V_ID_TIPO_DOCUMENTO, V_NOMBRE_TIPODOCUMENTO;
    EXIT WHEN C_TIPODOCUMENTO%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('TIPO DOCUMENTO: ' || V_ID_TIPO_DOCUMENTO || ' - ' ||
V_NOMBRE_TIPODOCUMENTO);

    OPEN C_PERSONA(V_ID_TIPO_DOCUMENTO);
    LOOP
      FETCH C_PERSONA INTO V_ID_PERSONA, V_NOMBRES_PERSONA;
      EXIT WHEN C_PERSONA%NOTFOUND;
      DBMS_OUTPUT.PUT_LINE('    PERSONA: ' || V_ID_PERSONA || ' - ' || V_NOMBRES_PERSONA);

      OPEN C_PERSONAL(V_ID_PERSONA);
      LOOP
        FETCH C_PERSONAL INTO V_ID_CARGO, V_NOMBRE_CARGO;
        EXIT WHEN C_PERSONAL%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('       CARGO: ' || V_ID_CARGO || ' - ' || V_NOMBRE_CARGO);
      END LOOP;
      CLOSE C_PERSONAL;

    END LOOP;
    CLOSE C_PERSONA;

  END LOOP;
  CLOSE C_TIPODOCUMENTO;

END;
```

Pruebas:

```
-- Ejecución
SET SERVEROUTPUT ON
EXEC SP_LISTAR_EMPLEADOS;
```



```
Tarea terminada en 0.096 segundos

TIPO DOCUMENTO: 1 - DNI
    PERSONA: 1 - RODRIGUEZ ORTEGA, DANIEL MANUEL
        CARGO: 1 - Gerente General
    PERSONA: 2 - CHAVEZ TAVERA, BRUNO MARTIN
        CARGO: 2 - Gerente de Operaciones
    PERSONA: 3 - SANCHEZ CARLOS, JULIO ROBERTO
        CARGO: 3 - Administrador
    PERSONA: 6 - PISCONTE DE LA CRUZ, SANDRA VIVIANA
        CARGO: 7 - Jefe de Proyectos
    PERSONA: 7 - APARICIO MANCO, CARLA PATRICIA
        CARGO: 10 - Programador Junior
    PERSONA: 8 - DEZA RIVERA, ERICK TIMOTEO
        CARGO: 9 - Programador
    PERSONA: 11 - VERA CORDOVA, JORGE FELIX
        CARGO: 4 - Jefe de Imagen
    PERSONA: 12 - RODRIGUEZ VERA, FRANCISCA
        CARGO: 11 - Analista de Marketing
TIPO DOCUMENTO: 2 - Pasaporte
    PERSONA: 9 - ROMERO RAMIREZ, DIEGO ANTONIO
        CARGO: 8 - Analista Programador
    PERSONA: 10 - TELLO MELENDEZ, JOSE ALBERTO
        CARGO: 11 - Analista de Marketing
TIPO DOCUMENTO: 3 - Carné Extranjería
    PERSONA: 4 - CARDOSO MARTINEZ, FRANCISCO MIGUEL
        CARGO: 6 - Jefe de Soporte
    PERSONA: 5 - RODRIGUEZ RUIZ, EDDIE FRANCISCO
        CARGO: 8 - Analista Programador
```

2.- Se pide realizar una función que calcule la bonificación sobre el sueldo básico de un empleado, basado en la siguiente tabla (para la función puede usar directamente los id_cargo):

Cargo	**% aumento**
Gerente General Gerente de Operaciones Administrador Jefe de Imagen	20%
Analista Comercial Jefe de Soporte Jefe de Proyectos Analista Programador	15%
Programador Programador Junior Analista de Marketing	10%

```
CREATE OR REPLACE FUNCTION FN_CALCULAR_BONIF(P_SUELDO NUMBER, P_ID_CARGO
NUMBER) RETURN NUMBER
IS
BEGIN
   IF P_ID_CARGO IN (1, 2, 3, 4) THEN
     RETURN P_SUELDO * 1.2;
   END IF;
   IF P_ID_CARGO IN (5, 6, 7, 8) THEN
     RETURN P_SUELDO * 1.15;
   END IF;
   IF P_ID_CARGO IN (9, 10, 11) THEN
     RETURN P_SUELDO * 1.1;
   END IF;
   RETURN 0; -- SI INGRESA UN CARGO QUE NO EXISTE
```

```
END;
```

Pruebas:

```
SELECT FN_CALCULAR_BONIF(25000,1) AS BONIFICACION FROM DUAL;
SELECT FN_CALCULAR_BONIF(7000,7) AS BONIFICACION FROM DUAL;
SELECT FN_CALCULAR_BONIF(2500,11) AS BONIFICACION FROM DUAL;
```

```
BONIFICACION
--------------------
             30000

BONIFICACION
--------------------
              8050

BONIFICACION
--------------------
              2750
```

3.- Se pide crear un procedimiento que permita recorrer a partir de la tabla de **CARGOS** todos los registros de la tabla **PERSONAL** para poder calcular y actualizar el valor del campo **SUELDO_BONIFICACION** usando la función anterior. Asimismo, si el sueldo básico es menor o igual a 5500 agregar una bonificación de 300.

```
CREATE OR REPLACE PROCEDURE PR_CALCULAR_BONIF IS
  CURSOR C_CARGOS IS
    SELECT ID_CARGO, SUELDO_BASICO FROM CARGO WHERE ESTADO = 'A';

  V_ID_CARGO CARGO.ID_CARGO%TYPE;
  V_SUELDO_BASICO CARGO.SUELDO_BASICO%TYPE;


BEGIN
  OPEN C_CARGOS;
  LOOP
    FETCH C_CARGOS INTO V_ID_CARGO,V_SUELDO_BASICO;
    EXIT WHEN C_CARGOS%NOTFOUND;

    IF V_SUELDO_BASICO <= 5500 THEN
       UPDATE PERSONAL SET SUELDO_BONIFICACION =
FN_CALCULAR_BONIF(SUELDO_BASICO, V_ID_CARGO) + 300
       WHERE ID_CARGO = V_ID_CARGO;
    ELSE
       UPDATE PERSONAL SET SUELDO_BONIFICACION =
FN_CALCULAR_BONIF(SUELDO_BASICO, V_ID_CARGO)
       WHERE ID_CARGO = V_ID_CARGO;
    END IF;



  END LOOP;
  CLOSE C_CARGOS;
END;
```

Prueba:

```
EXEC PR_CALCULAR_BONIF;

SELECT * FROM PERSONAL;
```

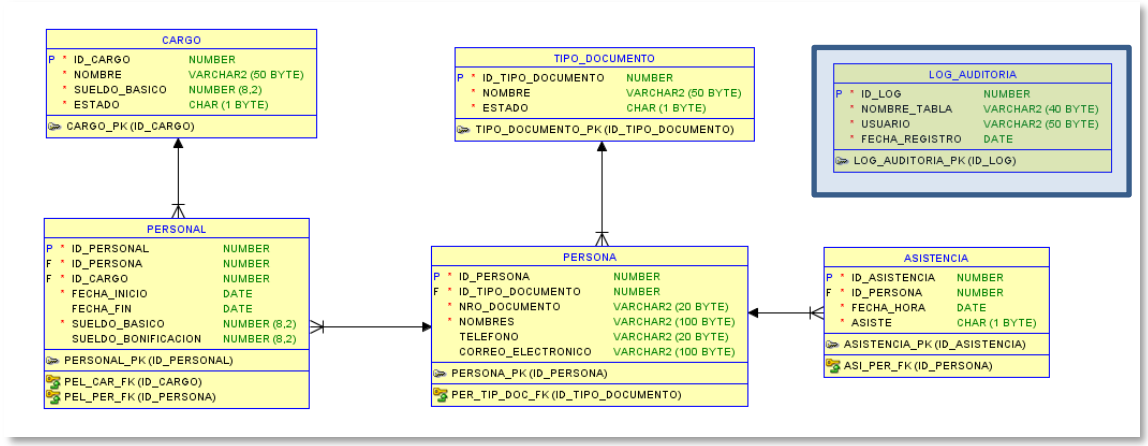
	ID_PERSONAL	ID_PERSONA	ID_CARGO	FECHA_INICIO	FECHA_FIN	SUELDO_BASICO	SUELDO_BONIFICACION
1	1	1	1	02/02/16	02/02/21	2500	30000
2	2	2	2	02/02/16	02/02/21	1500	18000
3	3	3	3	02/02/16	02/02/21	1600	19200
4	4	4	6	02/02/16	(null)	1000	11500
5	5	5	8	02/02/16	(null)	550	6625
6	6	6	7	01/04/16	(null)	700	8050
7	7	7	10	01/04/16	(null)	200	2500
8	8	8	9	01/04/16	(null)	400	4700
9	9	9	8	05/05/16	(null)	550	6625
10	10	10	11	16/04/17	08/08/18	250	3050
11	11	11	4	01/01/20	(null)	1200	14400
12	12	12	11	01/07/20	01/08/20	250	3050

Ejercicios triggers:

1.- Se pide llevar la trazabilidad de qué usuarios de base de datos realizan inserciones sobre la tabla **PERSONA**, a manera de auditoría. Esta trazabilidad se llevará sobre una nueva tabla llamada **LOG_AUDITORIA**. El trigger debe registrar el *nombre de la tabla*, el *usuario* que realizó el registro (inserción) y la *fecha* de ese insert, para esos dos últimos atributos se usarán las funciones **USER** y **SYSDATE**.

Ejecute el contenido del archivo **INF246_2021-2_LAB5_03_logauditoria.sql**.

El modelo quedará ahora de la siguiente manera:



El trigger a crear se disparará cada vez que un usuario **inserte** datos en la tabla **PERSONA**, por lo tanto, será un trigger "**after insert**":

```
CREATE OR REPLACE TRIGGER TR_LOG_PERSONA
AFTER INSERT ON PERSONA
FOR EACH ROW
DECLARE
   V_ID_LOG NUMBER;
BEGIN
```

```
    SELECT NVL(MAX(ID_LOG), 0) INTO V_ID_LOG FROM LOG_AUDITORIA;

    INSERT INTO LOG_AUDITORIA (ID_LOG, NOMBRE_TABLA, USUARIO, FECHA_REGISTRO)
    VALUES (V_ID_LOG + 1, 'PERSONA', USER, SYSDATE);
END;
```

Realizamos la prueba de la siguiente manera:

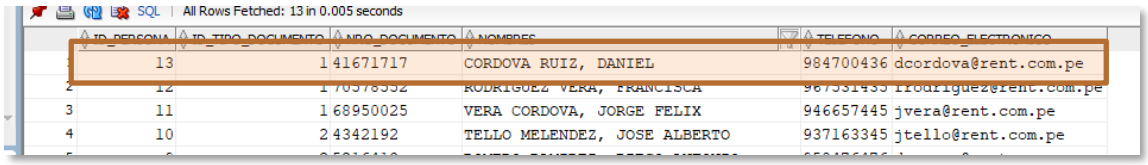
- Insertamos una persona en la tabla **PERSONA**.
- Comprobamos el registro insertado con un query en **PERSONA**.
- Comprobamos el funcionamiento correcto del trigger con un query en **LOG_AUDITORIA**.

Ya que el último ID_PERSONA es 12, usaremos el 13:

```
INSERT INTO PERSONA (ID_PERSONA, ID_TIPO_DOCUMENTO, NRO_DOCUMENTO, NOMBRES, TELEFONO, CORREO_ELECTRONICO) VALUES (13, 1, '41671717' , 'CORDOVA RUIZ, DANIEL', '984700436', 'dcordova@rent.com.pe');
```

Verificamos que se insertó en la tabla PERSONA:

```
SELECT * FROM PERSONA ORDER BY 1 DESC;
```



Revisamos la tabla afectada por el trigger:

```
SELECT * FROM LOG_AUDITORIA;
```

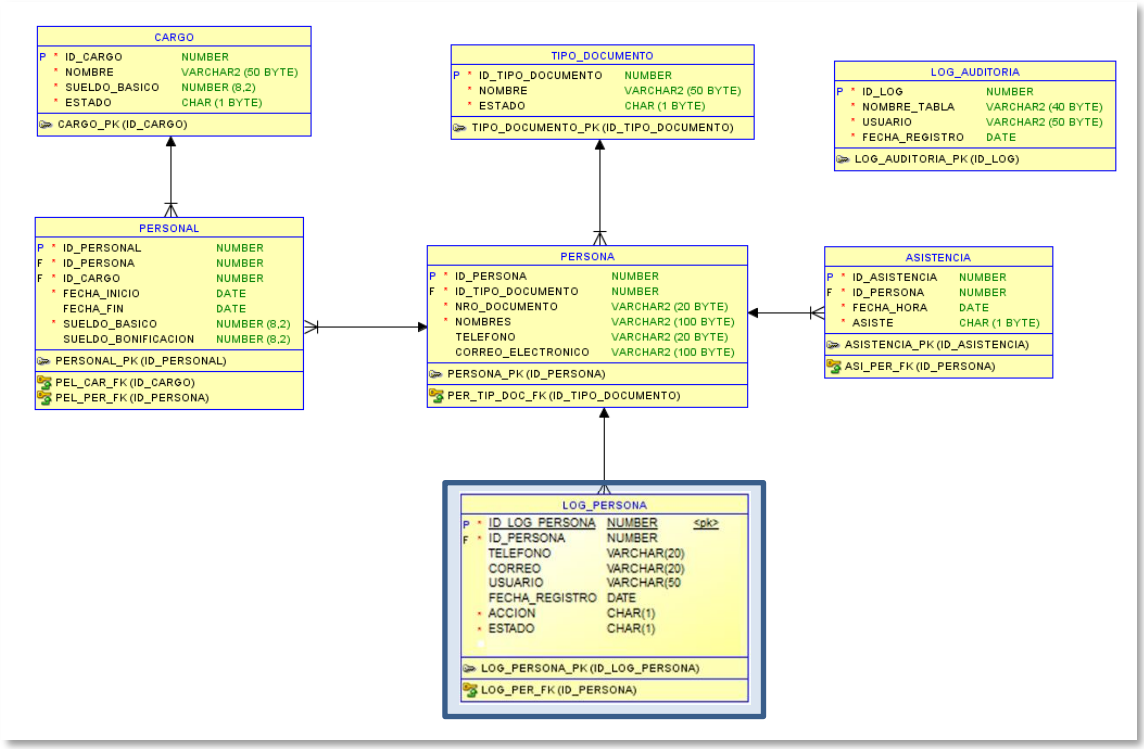
	ID_LOG	NOMBRE_TABLA	USUARIO	FECHA_REGISTRO
1	1	PERSONA	SYSTEM	27/11/21

Y efectivamente, se realizó la inserción en la tabla de auditoría. Con esto podemos llevar la trazabilidad de los usuarios del sistema que hicieron inserciones (o cualquier otra acción) sobre cualquier tabla del sistema, para este ejemplo: PERSONA.

2.- Se desea tener un mayor control sobre la auditoría de la tabla **PERSONA** y no necesariamente guardar la información de quién y cuándo se realizaron cambios, sino también se quiere llevar un control de la historia de cambios realizados a los datos de las personas, en este caso conocer si hubieron correcciones en los campos **CORREO_ELECTRONICO** y **TELEFONO** (considerar **inserción** y **modificación**). Este registro de cambios se llevará en la tabla **LOG_PERSONA**.

Ejecute el contenido del archivo **INF246_2021-2_LAB5_04_logpersona.sql**.

El modelo quedará ahora de la siguiente manera:



La tabla **LOG_PERSONA** guardará entonces los siguientes registros:

- Si es inserción: para los registros nuevos, si en los campos teléfono y correo en caso no hubiera dato poner guion '-'. Poner el campo estado en **LOG_PERSONA** como activo.
- Si es modificación: se actualiza el registro anterior poniendo su estado como desactivado en **LOG_PERSONA**, y se insertan una nueva tupla con los valores actualizados.
- El campo **ACCION** tendrá los valores de **'I'** para insert y **'U'** para update.
- En el campo **ESTADO** tendrá los valores de **'0'** para indicar estado desactivado y **'1'** para activo

El trigger a crear tiene que dispararse cada vez que un usuario **inserte** o **actualice** datos en la tabla **PERSONA**, por lo tanto, será un trigger "**after insert or update**":

```
CREATE OR REPLACE TRIGGER TR_TRAZA_PERSONA
AFTER INSERT OR UPDATE ON PERSONA
FOR EACH ROW
DECLARE
   V_ID_LOG NUMBER;
BEGIN
   SELECT NVL(MAX(ID_LOG_PERSONA), 0) INTO V_ID_LOG FROM LOG_PERSONA;

   IF INSERTING THEN
      INSERT INTO LOG_PERSONA (ID_LOG_PERSONA, ID_PERSONA, TELEFONO, CORREO,
USUARIO, FECHA_REGISTRO, ACCION,ESTADO)
      VALUES (V_ID_LOG + 1, :NEW.ID_PERSONA, :NEW.TELEFONO,
:NEW.CORREO_ELECTRONICO, USER, SYSDATE, 'I','1');
   END IF;
```

```
    IF UPDATING THEN
        UPDATE LOG_PERSONA SET ESTADO = '0'
        WHERE ID_PERSONA = :NEW.ID_PERSONA
        AND ESTADO = '1';

        INSERT INTO LOG_PERSONA (ID_LOG_PERSONA, ID_PERSONA, TELEFONO, CORREO,
USUARIO, FECHA_REGISTRO, ACCION, ESTADO)
        VALUES (V_ID_LOG + 1, :NEW.ID_PERSONA, :NEW.TELEFONO,
:NEW.CORREO_ELECTRONICO, USER, SYSDATE, 'U','1');
    END IF;
END;
```

Realizamos la prueba de la siguiente manera:

- Insertamos una persona en la tabla **PERSONA**.
- Comprobamos el registro insertado con un query en **PERSONA**.
- Comprobamos el funcionamiento correcto del trigger con un query en **LOG_PERSONA**.
- Actualizamos una persona en la tabla **PERSONA**.
- Comprobamos el registro actualizado con un query en **PERSONA**.
- Comprobamos el funcionamiento correcto del trigger con un query en **LOG_PERSONA**.

Cuando es insert:

```
INSERT INTO PERSONA (ID_PERSONA, ID_TIPO_DOCUMENTO, NRO_DOCUMENTO, NOMBRES,
TELEFONO, CORREO_ELECTRONICO) VALUES (14, 1, '42671717' , 'ARANA QUISPE, DAVID',
'944700499', 'darana@rent.com.pe');
```

Verificamos que se insertó en la tabla PERSONA:

```
SELECT * FROM PERSONA ORDER BY 1 DESC;
```

	ID_PERSONA	ID_TIPO_DOCUMENTO	NRO_DOCUMENTO	NOMBRES	TELEFONO	CORREO_ELECTRONICO
1	14	1	42671717	ARANA QUISPE, DAVID	944700499	darana@rent.com.pe
2	13	1	41671717	CORDOVA RUIZ, DANIEL	984700436	dcordova@rent.com.pe

Revisamos la tabla afectada por el trigger:

```
SELECT * FROM LOG_PERSONA;
```

	ID_LOG_PERSONA	ID_PERSONA	TELEFONO	CORREO	USUARIO	FECHA_REGISTRO	ACCION	ESTADO
1	1	14	944700499	darana@rent.com.pe	SYSTEM	28/11/21	I	1

Cuando es update:

```
UPDATE PERSONA SET TELEFONO = '956554779', CORREO_ELECTRONICO =
'fmcardoso@rent.com.pe' WHERE ID_PERSONA = 14;
```

Verificamos que se actualizó en la tabla PERSONA:

```
SELECT * FROM PERSONA WHERE ID_PERSONA = 14;
```

	ID_PERSONA	ID_TIPO_DOCUMENTO	NRO_DOCUMENTO	NOMBRES	TELEFONO	CORREO_ELECTRONICO
1	14	1	42671717	ARANA QUISPE, DAVID	956554779	fmcardoso@rent.com.pe

Revisamos la tabla afectada por el trigger:

```
SELECT * FROM LOG_PERSONA;
```

	ID_LOG_PERSONA	ID_PERSONA	TELEFONO	CORREO	USUARIO	FECHA_REGISTRO	ACCION	ESTADO
1	1	14	944700499	darana@rent.com.pe	SYSTEM	28/11/21	I	0
2	2	14	956554779	fmcardoso@rent.com.pe	SYSTEM	28/11/21	U	1

29 de noviembre de 2021
OTM, JPSG