**CREACION DE LOS TRIGGERS (CÓDIGO)**

--Trigger de control de la tabla PRODUCTO—

Este trigger muestra las operaciones que se realizan, guardadas en la tabla auditar\_empresa, al introducir, modificar o borrar un producto de la tabla PRODUCTO

CREATE OR REPLACE TRIGGER control\_producto

    BEFORE INSERT OR UPDATE OR DELETE ON PRODUCTO

    FOR EACH ROW

    BEGIN

        IF INSERTING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'TABLA PRODUCTO','Producto nuevo registrado con nº ' || :new.C\_PRODUCTO);

        ELSIF UPDATING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'TABLA PRODUCTO','Producto nº ' || :old.C\_PRODUCTO || ' modificado');

        ELSIF DELETING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'TABLA PRODUCTO','Producto nº ' || :old.C\_PRODUCTO || ' borrado');

        END IF;

    END;

/

--Trigger de control de la tabla CLIENTE—

Este trigger muestra las operaciones que se realizan, guardadas en la tabla auditar\_empresa, al introducir, modificar o borrar un cliente de la tabla CLIENTE

CREATE OR REPLACE TRIGGER control\_cliente

    BEFORE INSERT OR UPDATE OR DELETE ON CLIENTE

    FOR EACH ROW

    BEGIN

        IF INSERTING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'CLIENTE', 'Nuevo cliente registrado: ' || :new.NOMBRE || ' con DNI ' || :new.DNI);

        ELSIF UPDATING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'CLIENTE', ' Datos cliente modificados');

        ELSIF DELETING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'CLIENTE', 'El cliente ' || :old.NOMBRE || ' con DNI ' || :old.DNI || ' ha sido borrado' );

        END IF;

    END;

/

--Trigger de control de la tabla PEDIDO—

Este trigger muestra las operaciones que se realizan, guardadas en la tabla auditar\_empresa, al introducir, modificar o borrar un pedido de la tabla REGISTRO\_PEDIDO

CREATE OR REPLACE TRIGGER control\_pedido

    BEFORE INSERT OR UPDATE OR DELETE ON REGISTRO\_PEDIDO

    FOR EACH ROW

    BEGIN

        IF INSERTING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'REGISTRO\_PEDIDO', 'Pedido registrado con nº: ' || :new.N\_PEDIDO || ' para el DNI ' || :new.DNI);

        ELSIF UPDATING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'REGISTRO\_PEDIDO', 'Modificado pedido nº: ' || :new.N\_PEDIDO) ;

        ELSIF DELETING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'REGISTRO\_PEDIDO', 'Pedido nº ' || :old.N\_PEDIDO || ' con DNI ' || :old.DNI || ' ha sido borrado' );

        END IF;

    END;

/

--Trigger de control de la tabla ALBARAN—

Este trigger muestra las operaciones que se realizan, guardadas en la tabla auditar\_empresa, al introducir, modificar o borrar un albaran de la tabla REGISTRO\_ALBARAN

CREATE OR REPLACE TRIGGER control\_albaran

    BEFORE INSERT OR UPDATE OR DELETE ON REGISTRO\_ALBARAN

    FOR EACH ROW

    BEGIN

        IF INSERTING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'REGISTRO\_ALBARAN', 'Albaran registrado con nº: ' || :new.N\_ALBARAN || ' para el PEDIDO ' || :new.N\_PEDIDO);

        ELSIF UPDATING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'REGISTRO\_ALBARAN', 'Modificado pedido nº: ' || :new.N\_ALBARAN) ;

        ELSIF DELETING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'REGISTRO\_ALBARAN', 'Albaran nº ' || :old.N\_ALBARAN || ' asociado al pedido ' || :old.N\_PEDIDO || ' ha sido borrado');

        END IF;

    END;

/

--Trigger de control de la tabla FACTURA—

Este trigger muestra las operaciones que se realizan, guardadas en la tabla auditar\_empresa, al introducir, modificar o borrar un factura de la tabla REGISTRO\_FACTURA

CREATE OR REPLACE TRIGGER control\_factura

    BEFORE INSERT OR UPDATE OR DELETE ON REGISTRO\_FACTURA

    FOR EACH ROW

    BEGIN

        IF INSERTING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'REGISTRO\_FACTURA', 'Factura generada con nº: ' || :new.N\_FACTURA);

        ELSIF UPDATING THEN

            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'REGISTRO\_FACTURA', 'Modificada factura nº: ' || :new.N\_FACTURA) ;

        ELSIF DELETING THEN

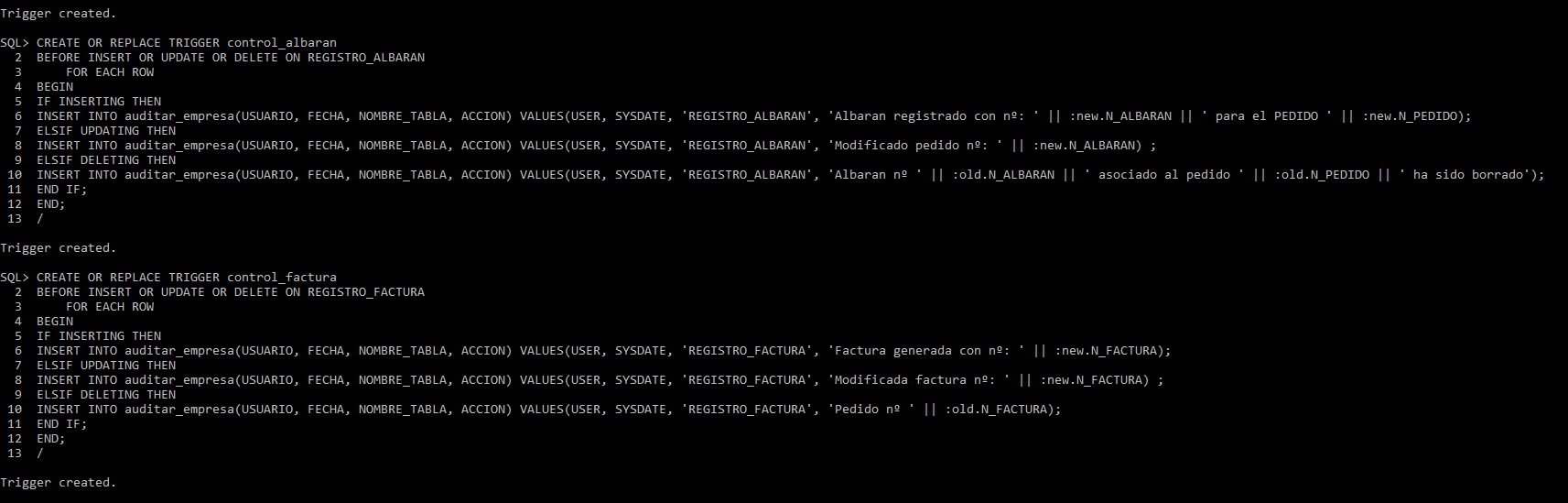
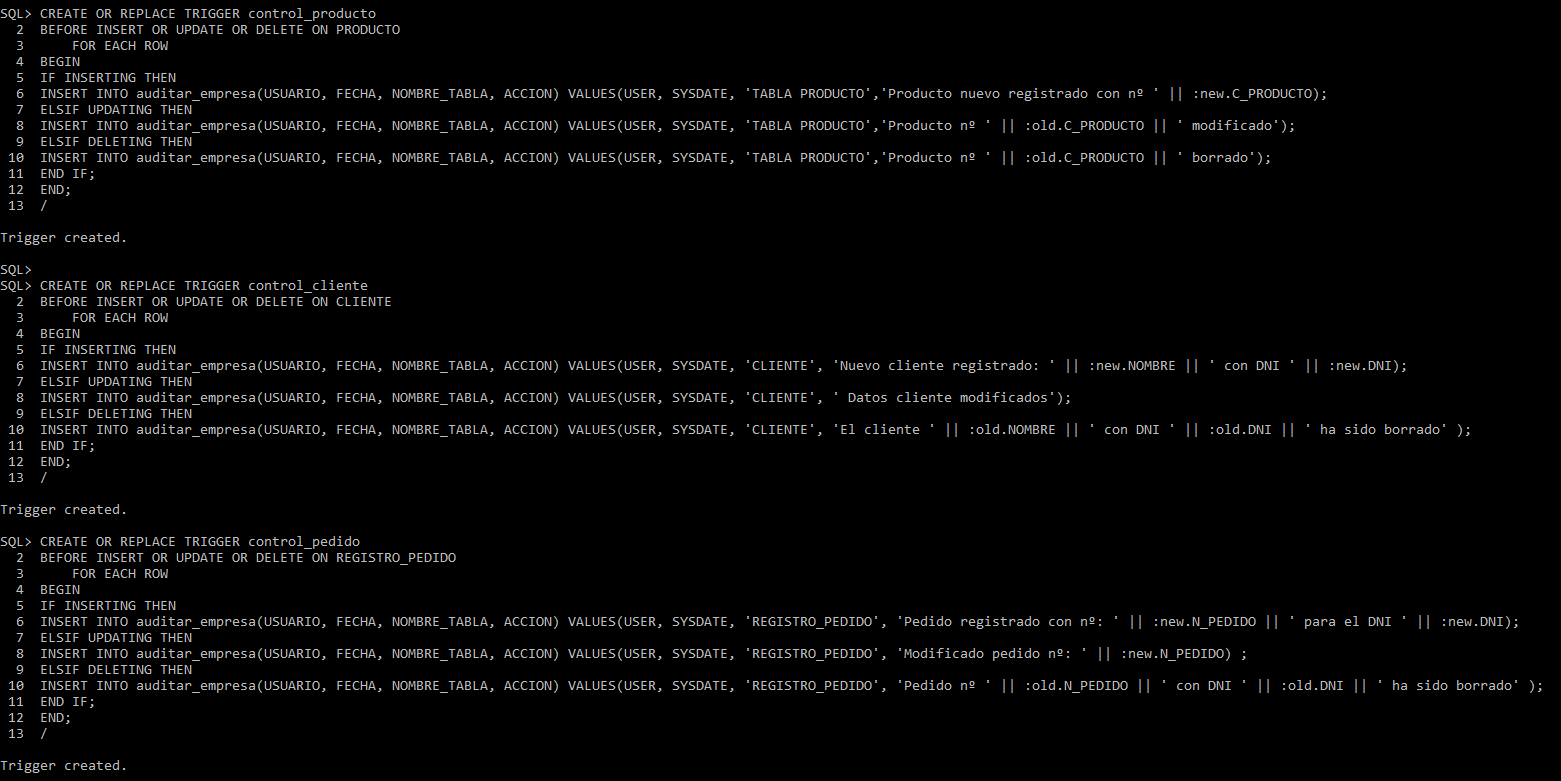
            INSERT INTO auditar\_empresa(USUARIO, FECHA, NOMBRE\_TABLA, ACCION) VALUES(USER, SYSDATE, 'REGISTRO\_FACTURA', 'Pedido nº ' || :old.N\_FACTURA);

        END IF;

    END;

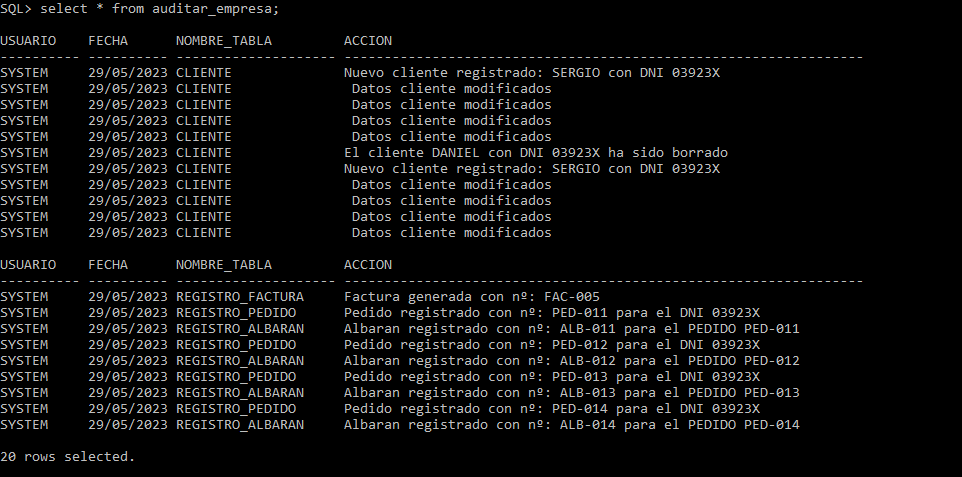
/

**CARGADO DE LOS TRIGGER EN LA MÁQUINA VIRTUAL**



**TRIGGERS DE LAS TODAS LAS OPERACIONES GESTIONADAS**

Al realizarse las operaciones, realizamos una consulta a nuestra tabla auditar\_empresa:



**CREACION DE PROCEDIMIENTO DE GESTIÓN DE CLIENTE (CÓDIGO)**

Este procedimiento nos permitirá, en función de las opciones facilitadas y el DNI del cliente, crear un cliente nuevo, modificar datos del cliente o eliminar un cliente.

Las opciones serían:

Opción1 🡪 1 (Crear un cliente nuevo)

Opción1 🡪 2 (Modificar un cliente existente)

Opción2 🡪 1 (Modificar nombre)

Opción2 🡪 2 (Modificar apellido)

Opción2 🡪 3 (Modificar direccion)

Opción2 🡪 4 (Modificar telefono)

Opción1 🡪 3 (Borrar un cliente)

Las tabla de CLIENTE se autocompletará gracias a una serie de funciones y bloques anónimos declarados en el interior del mismo procedimiento.

CREATE OR REPLACE PROCEDURE gestion\_clientes (opcion NUMBER, opcion2 NUMBER, dni\_emple CLIENTE.DNI%TYPE) IS

    FUNCTION ins\_emple (dni\_emple CLIENTE.DNI%TYPE, nombre CLIENTE.NOMBRE%TYPE, apellido CLIENTE.APELLIDO%TYPE, dir CLIENTE.DIRECCION%TYPE, tel CLIENTE.TEL%TYPE) RETURN NUMBER IS

    BEGIN

        INSERT INTO CLIENTE VALUES (dni\_emple , nombre , apellido , dir , tel);

        RETURN 1;

    END;

    FUNCTION del\_emple (dni\_emple CLIENTE.DNI%TYPE) RETURN NUMBER IS

    BEGIN

        DELETE  CLIENTE WHERE DNI = dni\_emple;

        RETURN 1;

    END;

    PROCEDURE my\_procedure (opcion NUMBER) IS

        resultado NUMBER;

    BEGIN

        CASE opcion

            WHEN 1 THEN

                DECLARE

                    nombre CLIENTE.NOMBRE%TYPE;

                    apellido CLIENTE.APELLIDO%TYPE;

                    dir CLIENTE.DIRECCION%TYPE;

                    tel CLIENTE.TEL%TYPE;

                BEGIN

                    resultado := ins\_emple(dni\_emple , 'SERGIO' , 'CACERES' , 'C/ MARIA USERA, 18 3-B' , '663432320');

                    DBMS\_OUTPUT.PUT\_LINE('Cliente creado con DNI --> ' || dni\_emple );

                END;

            WHEN 2 THEN

                DECLARE

                    nombre CLIENTE.NOMBRE%TYPE;

                    apellido CLIENTE.APELLIDO%TYPE;

                    dir CLIENTE.DIRECCION%TYPE;

                    tel CLIENTE.TEL%TYPE;

                BEGIN

                    CASE opcion2

                        WHEN 1 THEN

                            UPDATE CLIENTE SET NOMBRE = 'DANIEL' WHERE DNI = dni\_emple;

                        WHEN 2 THEN

                            UPDATE CLIENTE SET APELLIDO = 'MONTESINOS' WHERE DNI = dni\_emple;

                        WHEN 3 THEN

                            UPDATE CLIENTE SET DIRECCION = 'C/ Jovellanos, 123' WHERE DNI = dni\_emple;

                        WHEN 4 THEN

                            UPDATE CLIENTE SET TEL = '925270510' WHERE DNI = dni\_emple;

                        ELSE

                            DBMS\_OUTPUT.PUT\_LINE('No hay mas campos que modificar');

                    END CASE;

                END;

            WHEN 3 THEN

                resultado := del\_emple(dni\_emple);

        END CASE;

    END;

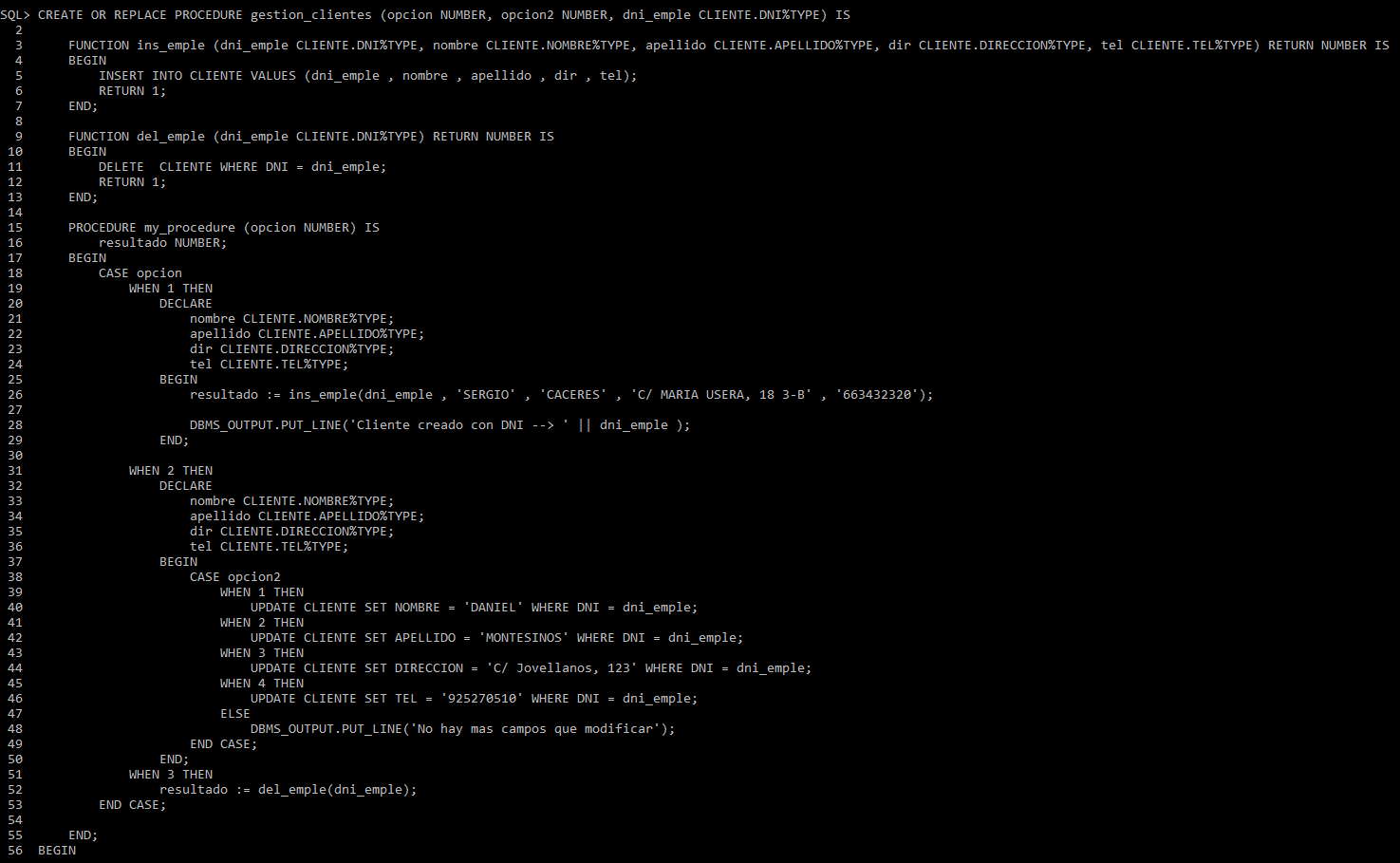
BEGIN

    my\_procedure(opcion);

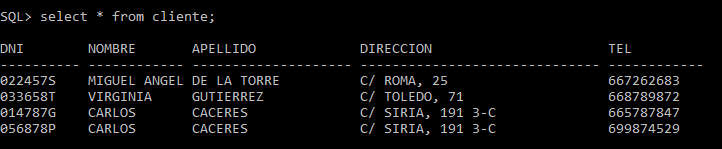
END;

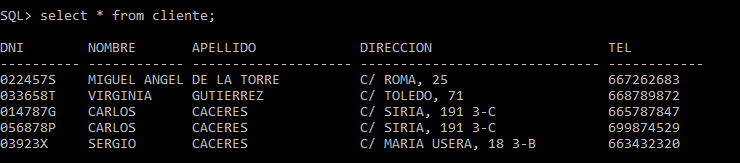
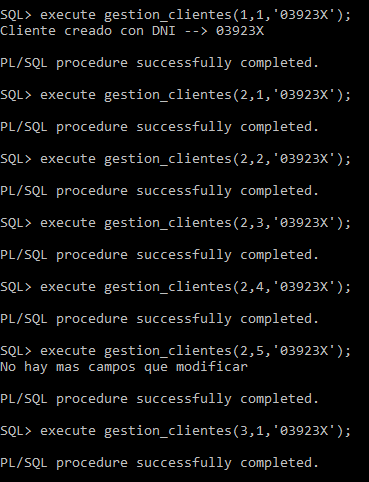
/

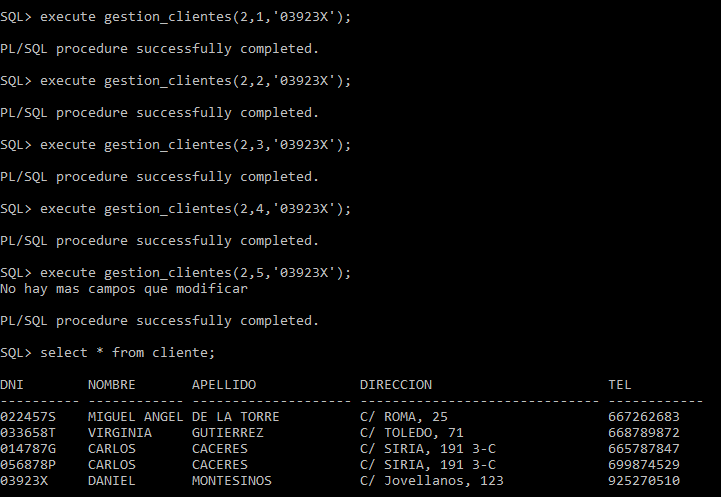
**CARGADO DEL PROCEDIMIENTO EN LA MÁQUINA VIRTUAL**

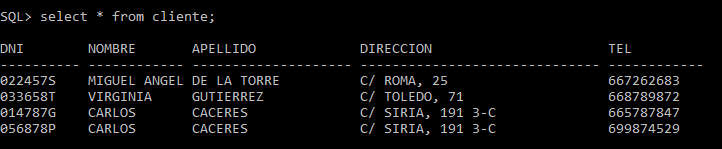
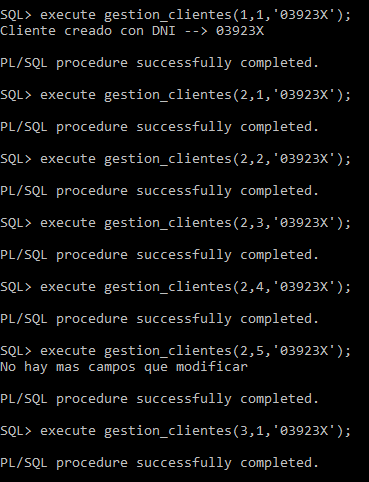


**EJECUCIÓN DEL PROCEDIMIENTO EN LA MÁQUINA VIRTUAL**









**CREACION DE PROCEDIMIENTO DE GESTIÓN DE PEDIDOS (CÓDIGO)**

Este procedimiento nos permitirá, al igual que el anterior, en función de la opción facilitada, el DNI del cliente y la cantidad, crear una factura, una serie de pedidos nuevos en función del valor dado o eliminar todo (no completado).

Las opciones serían:

Opción1 🡪 1 (Crear un cliente nuevo)

Opción1 🡪 2 (Borrar)

Las líneas de la tabla REGISTRO\_PEDIDO, REGISTRO\_ALBARAN y REGISTRO\_FACTURA se autocompletará llamando a una serie de funciones y bloques anónimos declarados dentro del mismo procedimiento.

CREATE OR REPLACE PROCEDURE gestion\_pedidos (opcion NUMBER, dni\_emple CLIENTE.DNI%TYPE, cant\_pedidos NUMBER) IS

    FUNCTION ins\_pedido (num\_pedido REGISTRO\_PEDIDO.N\_PEDIDO%TYPE, dni\_emple REGISTRO\_PEDIDO.DNI%TYPE, fecha\_ped REGISTRO\_PEDIDO.FECHA%TYPE) RETURN NUMBER IS

        BEGIN

            INSERT INTO REGISTRO\_PEDIDO VALUES (num\_pedido , dni\_emple , fecha\_ped);

            RETURN 1;

        END;

    FUNCTION ins\_factura (num\_factura REGISTRO\_FACTURA.N\_FACTURA%TYPE, fecha\_fac REGISTRO\_FACTURA.FECHA%TYPE) RETURN NUMBER IS

        BEGIN

            INSERT INTO REGISTRO\_FACTURA VALUES (num\_factura, fecha\_fac);

            RETURN 1;

        END;

        FUNCTION ins\_albaran (num\_albaran REGISTRO\_ALBARAN.N\_ALBARAN%TYPE, fecha\_alb REGISTRO\_ALBARAN.FECHA%TYPE, num\_factura REGISTRO\_FACTURA.N\_FACTURA%TYPE, num\_pedido REGISTRO\_PEDIDO.N\_PEDIDO%TYPE) RETURN NUMBER IS

        BEGIN

            INSERT INTO REGISTRO\_ALBARAN VALUES (num\_albaran,fecha\_alb,num\_factura,num\_pedido);

            RETURN 1;

        END;

    PROCEDURE my\_procedure (opcion NUMBER) IS

        resultado NUMBER;

    BEGIN

        CASE opcion

            WHEN 1 THEN

                DECLARE

                    num\_pedido VARCHAR(10);

                    num\_factura VARCHAR(20);

                    num\_albaran VARCHAR(20);

                    max\_num\_pedido NUMBER;

                    max\_num\_factura NUMBER;

                    max\_num\_albaran NUMBER;

                    fecha\_ped REGISTRO\_PEDIDO.FECHA%TYPE;

                    fecha\_fac REGISTRO\_PEDIDO.FECHA%TYPE;

                    v\_cant\_pedidos NUMBER := cant\_pedidos;

                BEGIN

                    SELECT TO\_NUMBER(SUBSTR(MAX(N\_FACTURA), 5)) INTO max\_num\_factura FROM REGISTRO\_FACTURA;

                    max\_num\_factura := max\_num\_factura + 1;

                    num\_factura := 'FAC-' || LPAD(max\_num\_factura, 3, '0');

                    resultado := ins\_factura( num\_factura, '15/05/2023');

                    DBMS\_OUTPUT.PUT\_LINE('Factura creada con nº --> ' || num\_factura);

                    LOOP

                        SELECT TO\_NUMBER(SUBSTR(MAX(N\_PEDIDO), 5)) INTO max\_num\_pedido FROM REGISTRO\_PEDIDO;

                        max\_num\_pedido := max\_num\_pedido + 1;

                        num\_pedido := 'PED-' || LPAD(max\_num\_pedido, 3, '0');

                        resultado := ins\_pedido( num\_pedido , dni\_emple , '12/02/2024');

                        DBMS\_OUTPUT.PUT\_LINE('Pedido creado con nº --> ' || num\_pedido || ' para el DNI: ' || dni\_emple );

                        SELECT TO\_NUMBER(SUBSTR(MAX(N\_ALBARAN), 5)) INTO max\_num\_albaran FROM REGISTRO\_ALBARAN;

                        max\_num\_albaran := max\_num\_albaran + 1;

                        num\_albaran := 'ALB-' || LPAD(max\_num\_albaran, 3, '0');

                        resultado := ins\_albaran(num\_albaran,'14/07/2023',num\_factura,num\_pedido);

                        DBMS\_OUTPUT.PUT\_LINE('Albaran creado con nº --> ' || num\_albaran || ' para el DNI: ' || dni\_emple );

                        v\_cant\_pedidos := v\_cant\_pedidos - 1;

                        EXIT WHEN (v\_cant\_pedidos = 0);

                    END LOOP;

                EXCEPTION

                    WHEN DUP\_VAL\_ON\_INDEX THEN

                        DBMS\_OUTPUT.PUT\_LINE('Pedido duplicado');

                END;

            /\*WHEN 2 THEN

                DECLARE

                    num\_pedido VARCHAR(10);

                    num\_factura VARCHAR(20);

                    num\_albaran VARCHAR(20);

                BEGIN

                    DELETE REGISTRO\_ALBARAN WHERE N\_FACTURA = 'FAC-006';

                    DELETE REGISTRO\_ALBARAN WHERE N\_FACTURA = 'FAC-006';

                    DELETE REGISTRO\_FACTURA WHERE N\_FACTURA = 'FAC-006';

                    DELETE REGISTRO\_PEDIDO WHERE DNI = '03923X';

                    DELETE REGISTRO\_PEDIDO WHERE N\_PEDIDO = 'PED-016';

                    DELETE CLIENTE WHERE DNI = '03923X';

                EXCEPTION

                    WHEN NO\_DATA\_FOUND THEN

                        DBMS\_OUTPUT.PUT\_LINE('EL cliente no existe');

                END;

                \*/

            ELSE DBMS\_OUTPUT.PUT\_LINE('No hay mas opciones disponibles');

        END CASE;

    END;

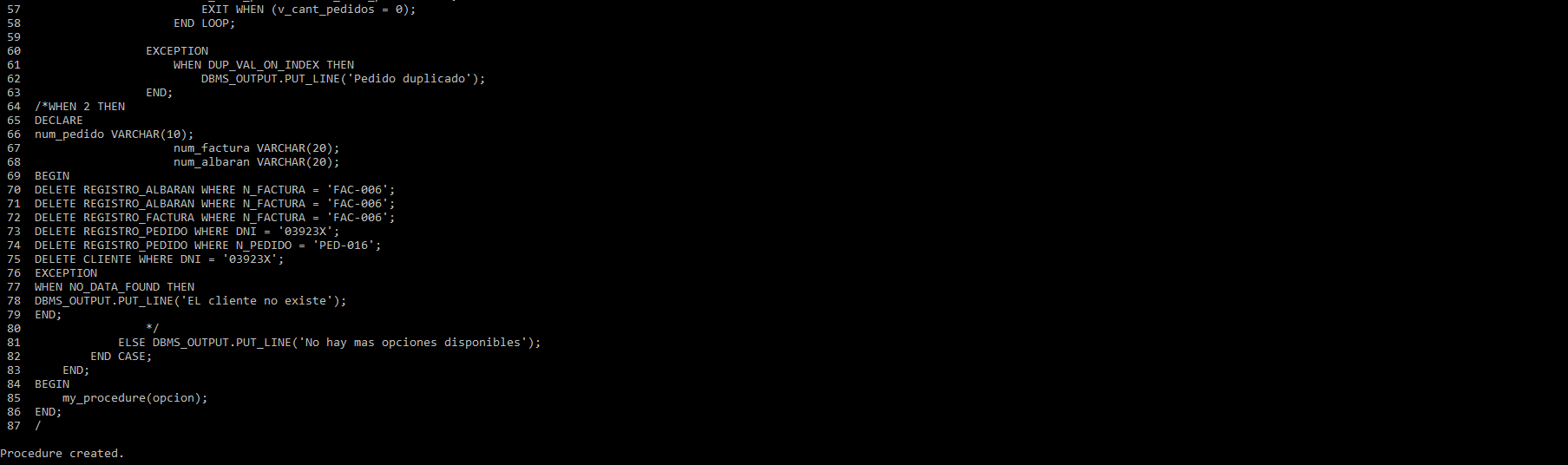
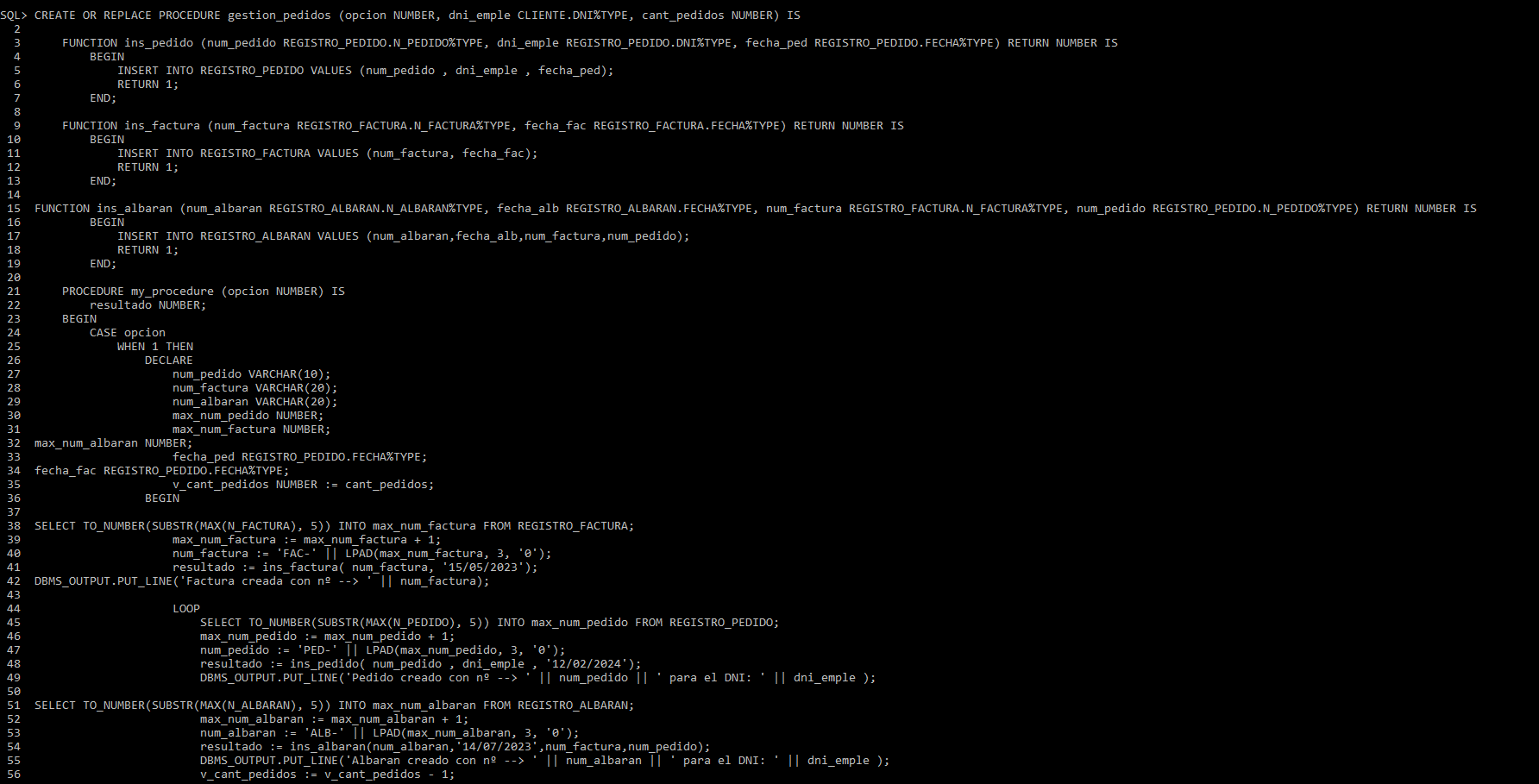
BEGIN

    my\_procedure(opcion);

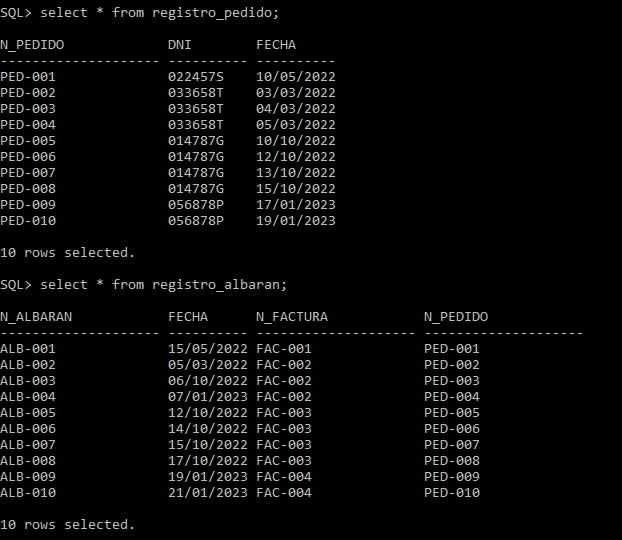
END;

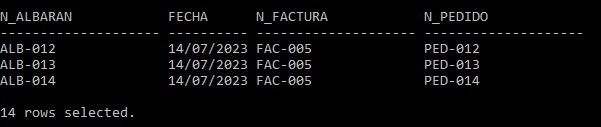
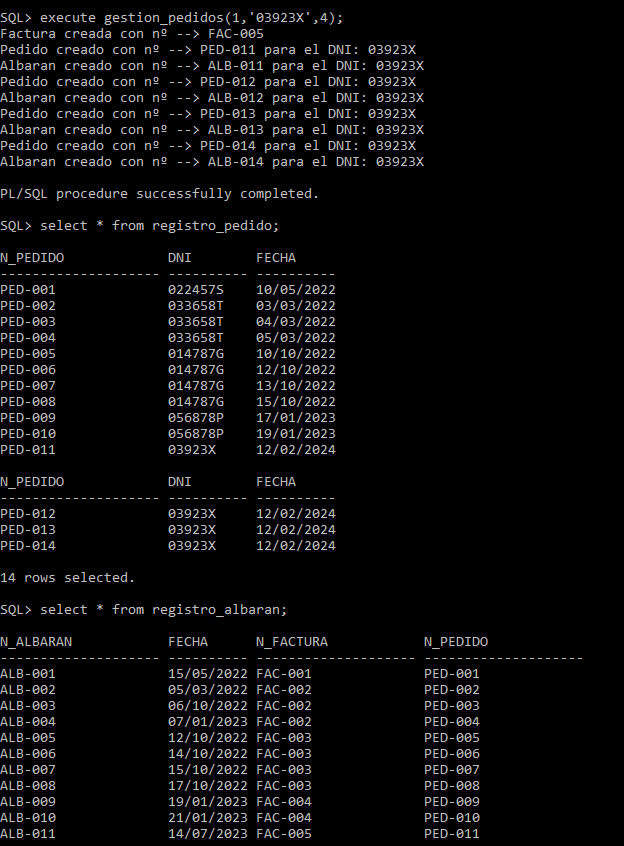
/

**CARGADO DEL PROCEDIMIENTO EN LA MÁQUINA VIRTUAL**



**EJECUCIÓN DEL PROCEDIMIENTO EN LA MÁQUINA VIRTUAL**





**CREACION DE PROCEDIMIENTO DE GESTIÓN DE LINEAS DE PEDIDO (CÓDIGO)**

Este procedimiento nos permitirá crear líneas en nuestro pedido y albarán en función de la cantidad de le estipulemos al procedimiento, además de facilitarle el número tanto de pedido como el de albarán que deseamos completar.

A través de un bucle, autocompletará tanto la tabla de LINEA\_PEDIDO como la de LINEA\_ALBARAN de manera aleatoria.

CREATE OR REPLACE PROCEDURE inser\_lineas (num\_lineas NUMBER,

                                          num\_pedido VARCHAR,

                                          num\_albaran VARCHAR

                                          ) IS

    v\_num\_lineas NUMBER := num\_lineas;

    v\_num\_pedido NUMBER := 1;

BEGIN

    LOOP

        INSERT INTO LINEA\_PEDIDO VALUES (

            'LP-' || LPAD(v\_num\_pedido, 3, '0'),

            num\_pedido,

            ROUND(DBMS\_RANDOM.VALUE(1, 100)),

            'AN-' || LPAD(TO\_CHAR(ROUND(DBMS\_RANDOM.VALUE(1, 41))), 3, '0'));

        INSERT INTO LINEA\_ALBARAN VALUES (

            'LA-' || LPAD(v\_num\_pedido, 3, '0'),

            num\_albaran,

            ROUND(DBMS\_RANDOM.VALUE(1, 100)),

            'AN-' || LPAD(TO\_CHAR(ROUND(DBMS\_RANDOM.VALUE(1, 41))), 3, '0'));

        v\_num\_pedido := v\_num\_pedido + 1;

        v\_num\_lineas := v\_num\_lineas - 1;

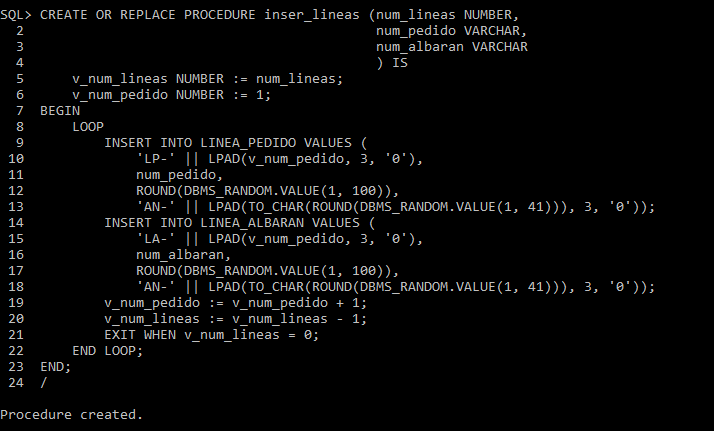
        EXIT WHEN v\_num\_lineas = 0;

    END LOOP;

END;

/

**CARGADO DEL PROCEDIMIENTO EN LA MÁQUINA VIRTUAL**



**EJECUCIÓN DEL PROCEDIMIENTO EN LA MÁQUINA VIRTUAL**

