# PRACTICA FINAL PL/SQL

La practica busca desarrollar los siguientes objetos:

- Paquete para gestionar a los Empleados.
- Trigger para llevar control de lo que sucede en la BD.

Paquete Empleados Funciones:

1. EXISTE_EMPLEADO(FiRST_NAME,LAST_NAME)

- Devuelve true si el empleado ya existe de lo contrario devuelve false.

2. EXISTE_DEPARTMENT_ID(Department_id)

- Devuelve true si el departamento existe de lo contrario devuelve false.

Procedimientos:

1. ALTA_EMPLEADO(valores necesarios para dar de alta al empleado)
    - Debe dar de alta un empleado con los valores indicados en los parámetros
    - Debe comprobar que no se duplica
    - Debe comprobar que el departamento existe antes de darlo de alta
2. BAJA_EMPLEADO(employe_id)
    - Debe borrar el registro con el id de empleado que se pasó por parámetro

- Debe validar si el empleado existe antes de ejecutar un DELETE

3. MOD_EMPLEADO(employe_id,nuevo_Dato,indicador_De_campo)
    - Debe modificar un campo en especifico del empleado que se pasó por parámetro
    - Para saber que campo debe modificar debe enviar en el parámetro indicador_De_campo los valores del 1 al 11, tomando de referencia la siguiente tabla.

| | |
|---|---|
| EMPLOYEE_ID | 1 |
| FIRST_NAME | 2 |
| LAST_NAME | 3 |
| EMAIL | 4 |
| PHONE_NUMBER | 5 |
| HIRE_DATE | 6 |
| JOB_ID | 7 |
| SALARY | 8 |
| COMMISSION_PCT | 9 |
| MANAGER_ID | 10 |
| DEPARTMENT_ID | 11 |

Ejemplo: se envía el employe_id = 100,
Nuevo_valor = ‘Pablo’, indicador_De_campo = 2

Tendrá que ir al employe_id 100 a modificar el FIRST_NAME con el valor de ‘Pablo’

TRIGGER CONTROL

Cree una tabla llamada CONTROL_LOG con los siguientes campos:

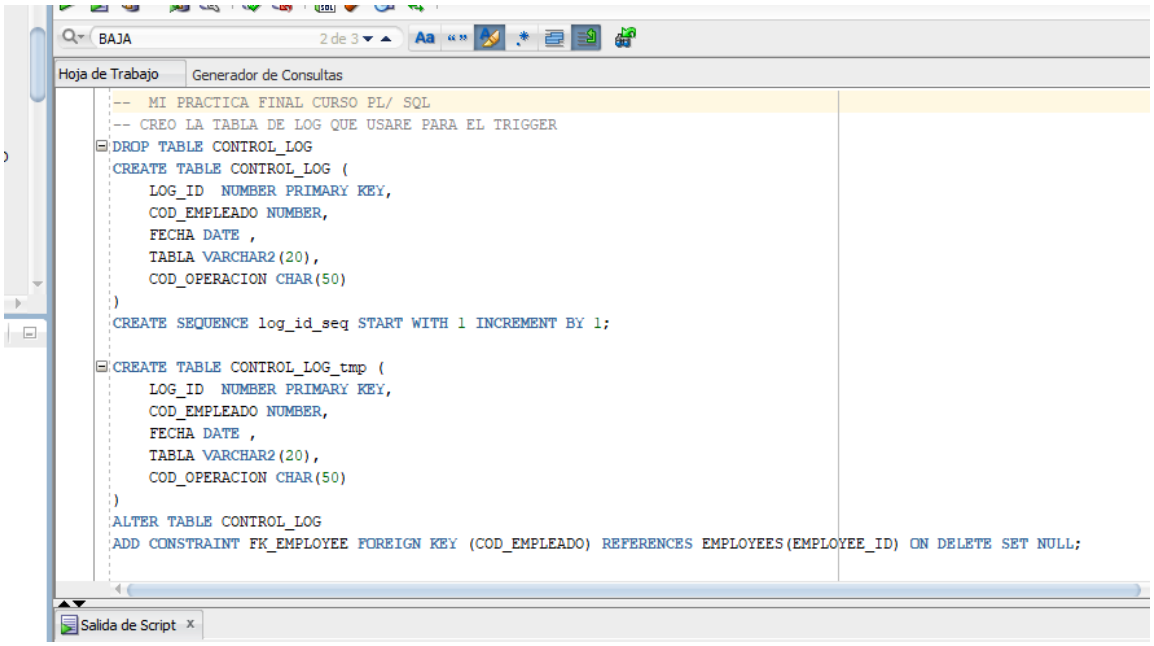
| | COLUMN_NAME | DATA_TYPE |
|---|---|---|
| 1 | COD_EMPLEADO | NUMBER |
| 2 | FECHA | DATE |
| 3 | TABLA | VARCHAR2(20 BYTE) |
| 4 | COD_OPERACION | CHAR(1 BYTE) |

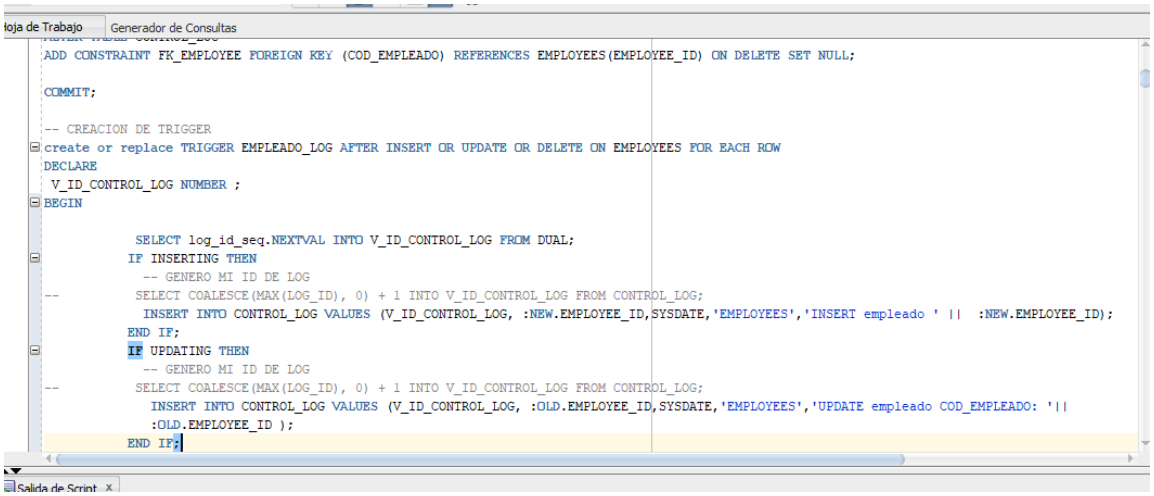
Esta tabla debe tener un campo llamado LOG_ID que sea el identificador único de la tabla, tener en cuenta que el campo COD_EMPLEADO debe ser una llave FORANEA(FK) que representara al employee_id de la tabla EMPLOYEES.

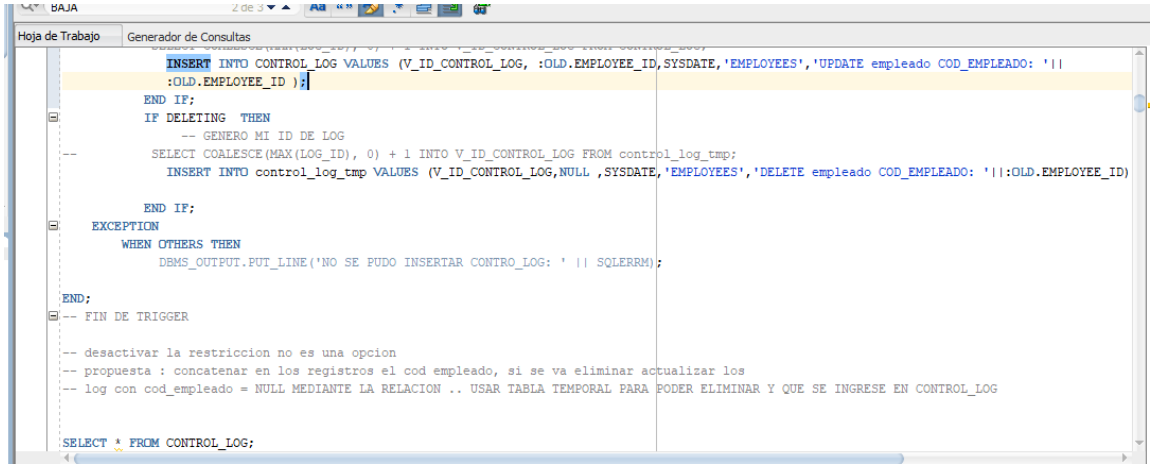
1. Cada vez que se ejecute un movimiento de INSERT, UPDATE o DELETE en la tabla EMPLOYEES debe reportarlos en la tabla CONTROL_LOG de la siguiente manera:
    - INSERT ○ COD_EMPLEADO: tendrá el nuevo código de empleado que se ingreso
        - ○ FECHA: tendrá la fecha en la que se hizo el movimiento
            - ○ TABLA: la tabla afectada ○ COD_OPERACION: ‘INSERT empleado’

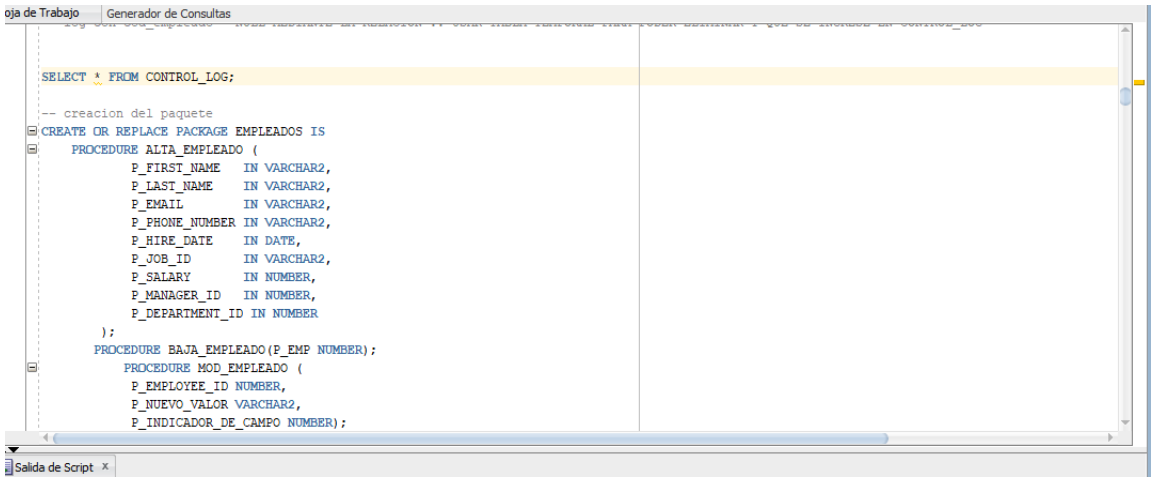
- UPDATE ○ COD_EMPLEADO: tendrá el código de empleado que será afectado
    - ○ FECHA: tendrá la fecha en la que se hizo el movimiento ○ TABLA: la tabla afectada ○ COD_OPERACION: ‘UPDATE empleado COD_EMPLEADO’
- DELETE ○ COD_EMPLEADO: tendrá el código de empleado que será afectado
    - ○ FECHA: tendrá la fecha en la que se hizo el movimiento
    - ○ TABLA: la tabla afectada
    - ○ COD_OPERACION: ‘DELETE empleado COD_EMPLEADO’

DESARROLLO


BAJA
2 de 3
Hoja de Trabajo
Generador de Consultas
-- MI PRACTICA FINAL CURSO PL/ SQL
-- CREO LA TABLA DE LOG QUE USARE PARA EL TRIGGER
DROP TABLE CONTROL_LOG
CREATE TABLE CONTROL_LOG (
LOG_ID NUMBER PRIMARY KEY,
COD_EMPLEADO NUMBER,
FECHA DATE ,
TABLA VARCHAR2(20),
COD_OPERACION CHAR(50)
)
CREATE SEQUENCE log_id_seq START WITH 1 INCREMENT BY 1;
CREATE TABLE CONTROL_LOG_tmp (
LOG_ID NUMBER PRIMARY KEY,
COD_EMPLEADO NUMBER,
FECHA DATE ,
TABLA VARCHAR2(20),
COD_OPERACION CHAR(50)
)
ALTER TABLE CONTROL_LOG
ADD CONSTRAINT FK_EMPLOYEE FOREIGN KEY (COD_EMPLEADO) REFERENCES EMPLOYEES(EMPLOYEE_ID) ON DELETE SET NULL;
Salida de Script

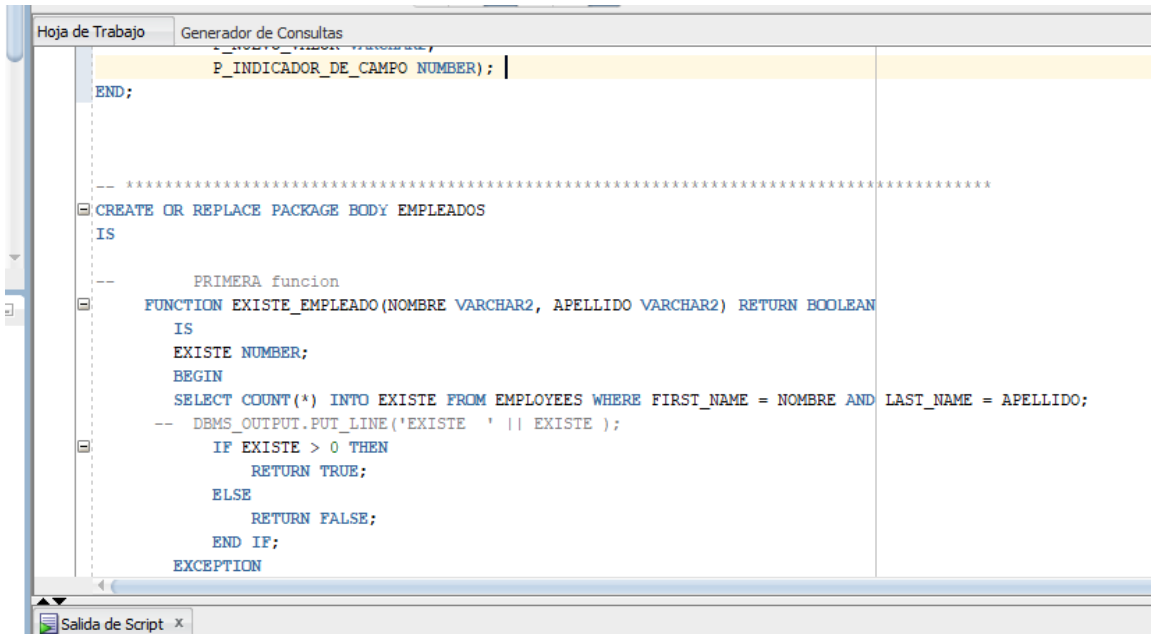

Hoja de Trabajo
Generador de Consultas
ADD CONSTRAINT FK_EMPLOYEE FOREIGN KEY (COD_EMPLEADO) REFERENCES EMPLOYEES(EMPLOYEE_ID) ON DELETE SET NULL;
COMMIT;
-- CREACION DE TRIGGER
create or replace TRIGGER EMPLEADO_LOG AFTER INSERT OR UPDATE OR DELETE ON EMPLOYEES FOR EACH ROW
DECLARE
V_ID_CONTROL_LOG NUMBER ;
BEGIN
SELECT log_id_seq.NEXTVAL INTO V_ID_CONTROL_LOG FROM DUAL;
IF INSERTING THEN
-- GENERO MI ID DE LOG
-- SELECT COALESCE(MAX(LOG_ID), 0) + 1 INTO V_ID_CONTROL_LOG FROM CONTROL_LOG;
INSERT INTO CONTROL_LOG VALUES (V_ID_CONTROL_LOG, :NEW.EMPLOYEE_ID,SYSDATE,'EMPLOYEES','INSERT empleado ' || :NEW.EMPLOYEE_ID);
END IF;
IF UPDATING THEN
-- GENERO MI ID DE LOG
-- SELECT COALESCE(MAX(LOG_ID), 0) + 1 INTO V_ID_CONTROL_LOG FROM CONTROL_LOG;
INSERT INTO CONTROL_LOG VALUES (V_ID_CONTROL_LOG, :OLD.EMPLOYEE_ID,SYSDATE,'EMPLOYEES','UPDATE empleado COD_EMPLEADO: '||
:OLD.EMPLOYEE_ID );
END IF;
Salida de Script


BAJA
Hoja de Trabajo
Generador de Consultas
INSERT INTO CONTROL_LOG VALUES (V_ID_CONTROL_LOG, :OLD.EMPLOYEE_ID,SYSDATE,'EMPLOYEES','UPDATE empleado COD_EMPLEADO: '||
:OLD.EMPLOYEE_ID );
END IF;
IF DELETING THEN
-- GENERO MI ID DE LOG
-- SELECT COALESCE(MAX(LOG_ID), 0) + 1 INTO V_ID_CONTROL_LOG FROM control_log_tmp;
INSERT INTO control_log_tmp VALUES (V_ID_CONTROL_LOG,NULL ,SYSDATE,'EMPLOYEES','DELETE empleado COD_EMPLEADO: '||:OLD.EMPLOYEE_ID)
END IF;
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE('NO SE PUDO INSERTAR CONTRO_LOG: ' || SQLERRM);
END;
-- FIN DE TRIGGER
-- desactivar la restriccion no es una opcion
-- propuesta : concatenar en los registros el cod empleado, si se va eliminar actualizar los
-- log con cod_empleado = NULL MEDIANTE LA RELACION .. USAR TABLA TEMPORAL PARA PODER ELIMINAR Y QUE SE INGRESE EN CONTROL_LOG
SELECT * FROM CONTROL_LOG;



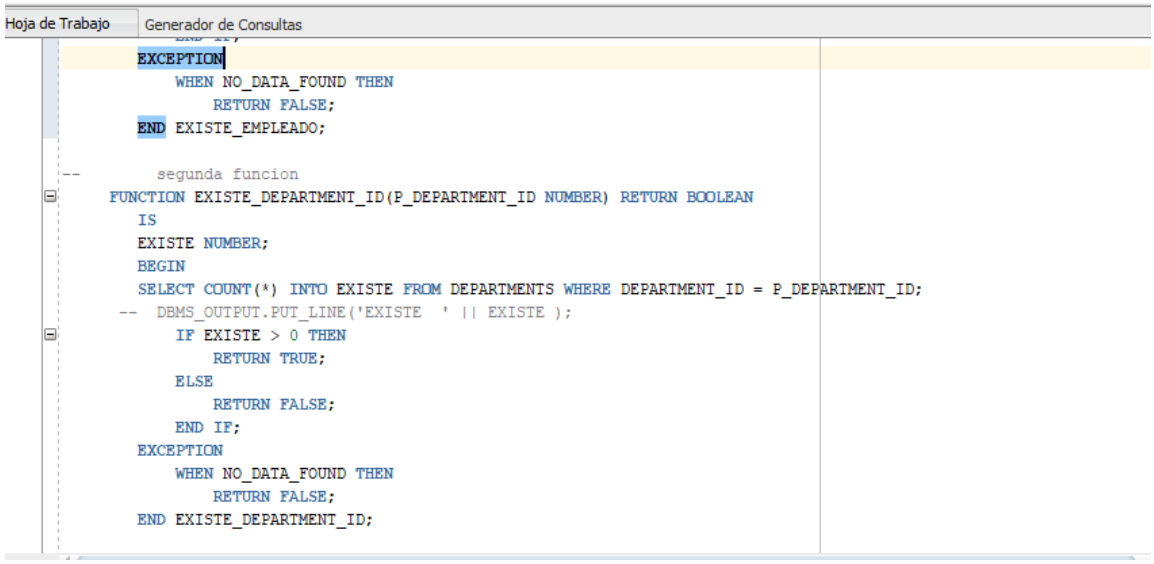
```
SELECT * FROM CONTROL_LOG;

-- creacion del paquete
CREATE OR REPLACE PACKAGE EMPLEADOS IS
    PROCEDURE ALTA_EMPLEADO (
            P_FIRST_NAME    IN VARCHAR2,
            P_LAST_NAME     IN VARCHAR2,
            P_EMAIL         IN VARCHAR2,
            P_PHONE_NUMBER IN VARCHAR2,
            P_HIRE_DATE     IN DATE,
            P_JOB_ID        IN VARCHAR2,
            P_SALARY        IN NUMBER,
            P_MANAGER_ID    IN NUMBER,
            P_DEPARTMENT_ID IN NUMBER
        );
       PROCEDURE BAJA_EMPLEADO(P_EMP NUMBER);
          PROCEDURE MOD_EMPLEADO (
           P_EMPLOYEE_ID NUMBER,
           P_NUEVO_VALOR VARCHAR2,
           P_INDICADOR_DE_CAMPO NUMBER);
```

Salida de Script



```
            P_INDICADOR_DE_CAMPO NUMBER);
END;



-- ********************************************************************************************
CREATE OR REPLACE PACKAGE BODY EMPLEADOS
IS

--        PRIMERA funcion
     FUNCTION EXISTE_EMPLEADO(NOMBRE VARCHAR2, APELLIDO VARCHAR2) RETURN BOOLEAN
        IS
        EXISTE NUMBER;
        BEGIN
        SELECT COUNT(*) INTO EXISTE FROM EMPLOYEES WHERE FIRST_NAME = NOMBRE AND LAST_NAME = APELLIDO;
       --  DBMS_OUTPUT.PUT_LINE('EXISTE  ' || EXISTE );
            IF EXISTE > 0 THEN
                RETURN TRUE;
            ELSE
                RETURN FALSE;
            END IF;
        EXCEPTION
```

Salida de Script



```
        EXCEPTION
            WHEN NO_DATA_FOUND THEN
                RETURN FALSE;
        END EXISTE_EMPLEADO;

--        segunda funcion
     FUNCTION EXISTE_DEPARTMENT_ID(P_DEPARTMENT_ID NUMBER) RETURN BOOLEAN
        IS
        EXISTE NUMBER;
        BEGIN
        SELECT COUNT(*) INTO EXISTE FROM DEPARTMENTS WHERE DEPARTMENT_ID = P_DEPARTMENT_ID;
       --  DBMS_OUTPUT.PUT_LINE('EXISTE  ' || EXISTE );
            IF EXISTE > 0 THEN
                RETURN TRUE;
            ELSE
                RETURN FALSE;
            END IF;
        EXCEPTION
            WHEN NO_DATA_FOUND THEN
                RETURN FALSE;
        END EXISTE_DEPARTMENT_ID;
```

Hoja de Trabajo | Generador de Consultas

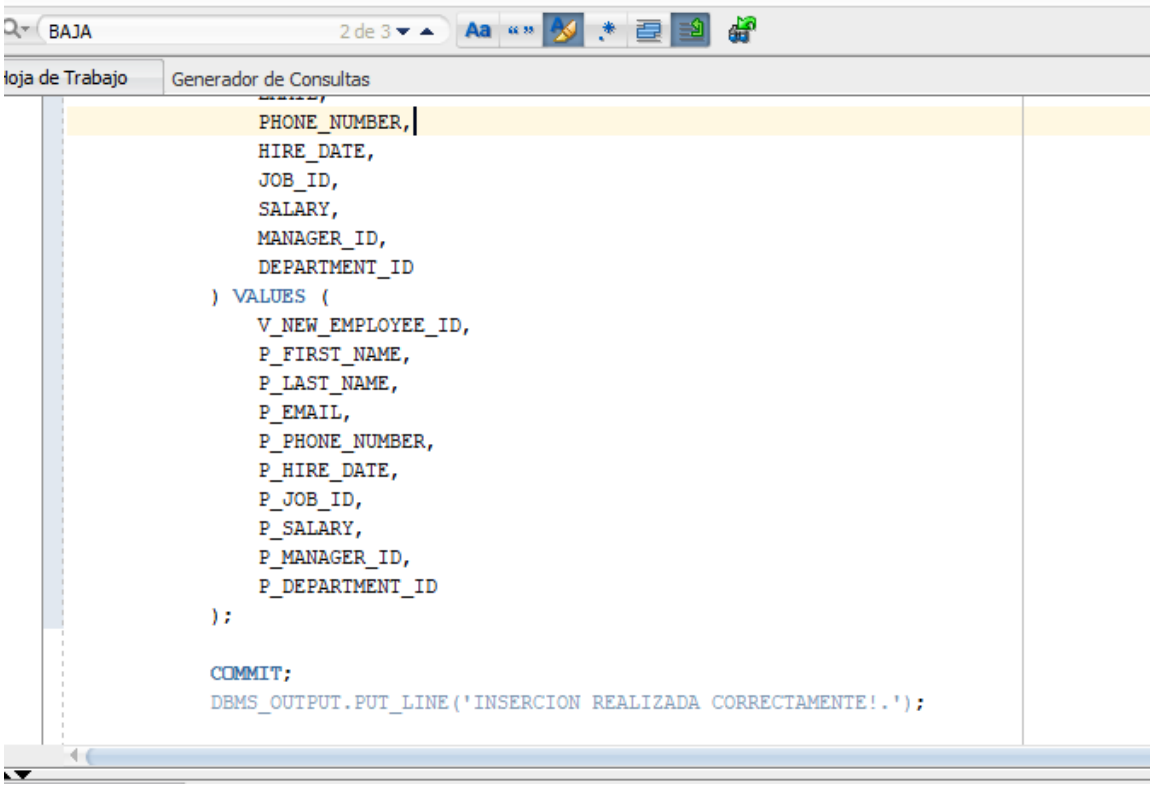
```
END EXISTE_DEPARTMENT_ID;

-- PRIMER PROC
 PROCEDURE ALTA_EMPLEADO (
    P_FIRST_NAME    IN VARCHAR2,
    P_LAST_NAME     IN VARCHAR2,
    P_EMAIL         IN VARCHAR2,
    P_PHONE_NUMBER IN VARCHAR2,
    P_HIRE_DATE     IN DATE,
    P_JOB_ID        IN VARCHAR2,
    P_SALARY        IN NUMBER,
    P_MANAGER_ID    IN NUMBER,
    P_DEPARTMENT_ID IN NUMBER
)
IS
    V_DEPARTMENT_EXISTS NUMBER;
    V_MANAGER_EXISTS NUMBER;
    V_JOB_EXISTS NUMBER;
    V_NEW_EMPLOYEE_ID NUMBER;
    V_USER VARCHAR2(50);
    V_ID_BIT NUMBER;
BEGIN
```

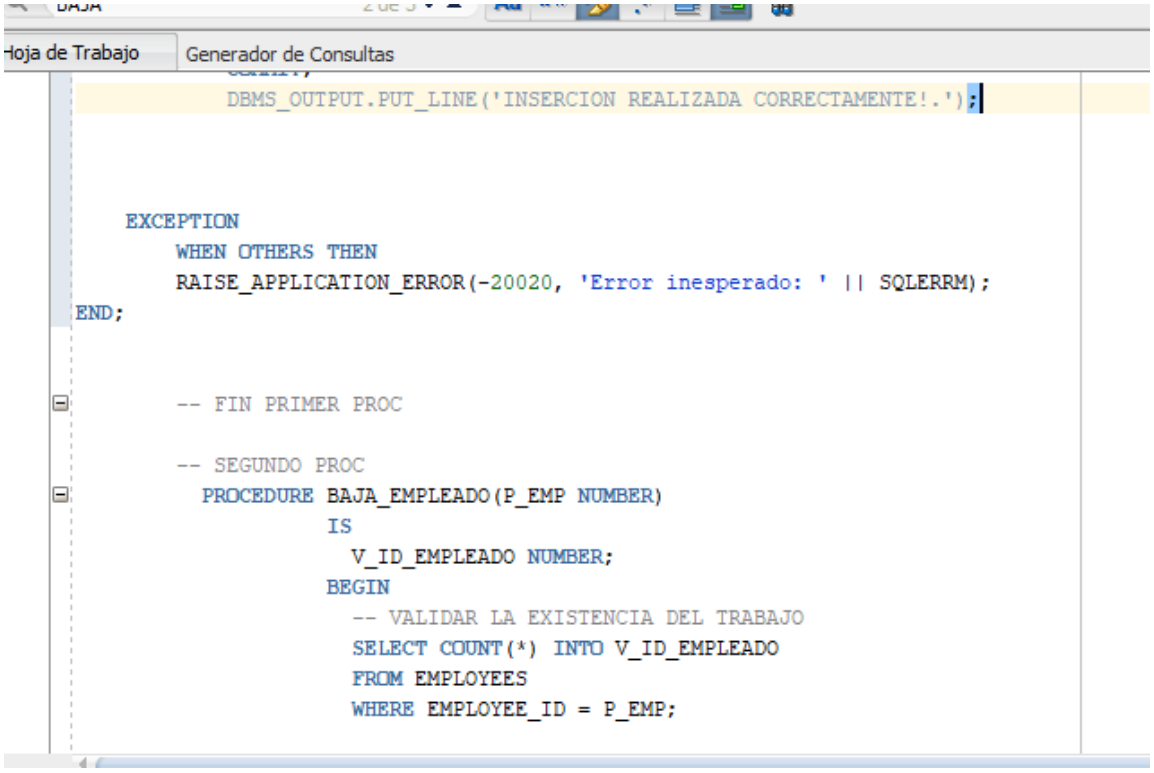
oja de Trabajo | Generador de Consultas

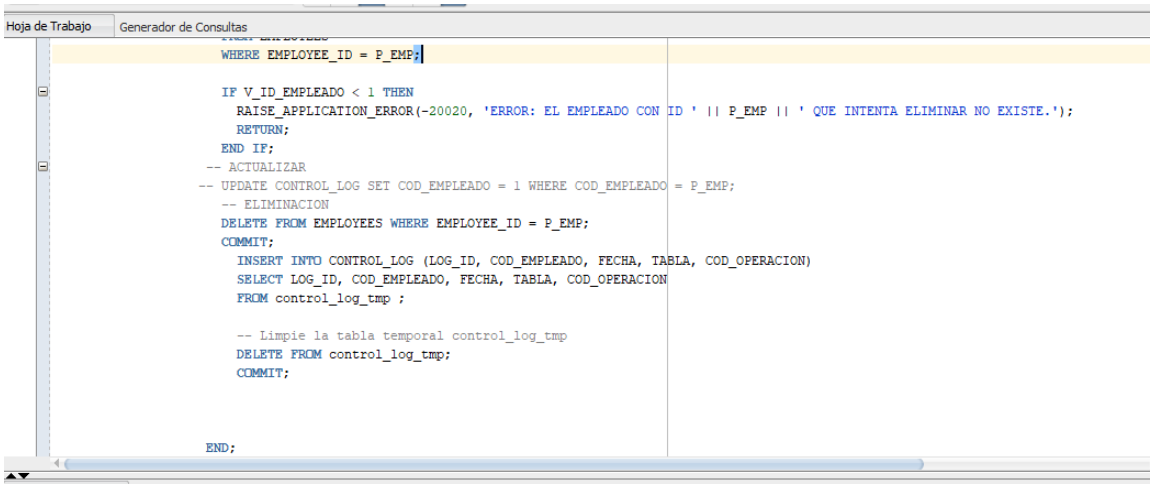
```
BEGIN
    SELECT COALESCE(MAX(EMPLOYEE_ID), 0) + 1 INTO V_NEW_EMPLOYEE_ID FROM EMPLOYEES;
  -- VALIDAR QUE NO SE DUPLIQUE
    IF EXISTE_EMPLEADO(P_FIRST_NAME, P_LAST_NAME) = TRUE THEN
        RAISE_APPLICATION_ERROR(-20020, 'ESE EMPLEADO YA EXISTE, EL NOMBRE Y APELLIDO COINCIDE CON UN REGISTRO EN LA BASE DE DATOS');
        RETURN;
    END IF;
    -- VALIDAR LA EXISTENCIA DEL DEPARTAMENTO
  IF EXISTE_DEPARTMENT_ID(P_DEPARTMENT_ID) = FALSE THEN
    RAISE_APPLICATION_ERROR(-20020, 'ERROR: DEPARTAMENTO CON ID ' || P_MANAGER_ID || ' NO EXISTE.');

        RETURN;
    END IF;

    -- VALIDAR LA EXISTENCIA DEL GERENTE
    IF P_MANAGER_ID IS NOT NULL THEN
        SELECT COUNT(*) INTO V_MANAGER_EXISTS
        FROM EMPLOYEES
        WHERE EMPLOYEE_ID = P_MANAGER_ID;

        IF V_MANAGER_EXISTS = 0 THEN
         RAISE_APPLICATION_ERROR(-20020, 'ERROR: MANAGER CON ID ' || P_MANAGER_ID || ' NO EXISTE.');
```

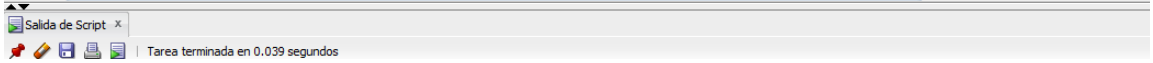
oja de Trabajo | Generador de Consultas

```
            RAISE_APPLICATION_ERROR(-20020, 'ERROR: MANAGER CON ID ' || P_MANAGER_ID || ' NO EXISTE.');
               RETURN;
            END IF;
        END IF;

        -- VALIDAR LA EXISTENCIA DEL TRABAJO
        SELECT COUNT(*) INTO V_JOB_EXISTS
        FROM JOBS
        WHERE JOB_ID = P_JOB_ID;

        IF V_JOB_EXISTS = 0 THEN
             RAISE_APPLICATION_ERROR(-20020, 'ERROR: JOB CON ID ' || P_JOB_ID || ' NO EXISTE.');
            RETURN;
        END IF;

        -- INSERTAR EL NUEVO EMPLEADO
        INSERT INTO EMPLOYEES (
            EMPLOYEE_ID,
            FIRST_NAME,
            LAST_NAME,
            EMAIL,
            PHONE_NUMBER,
```

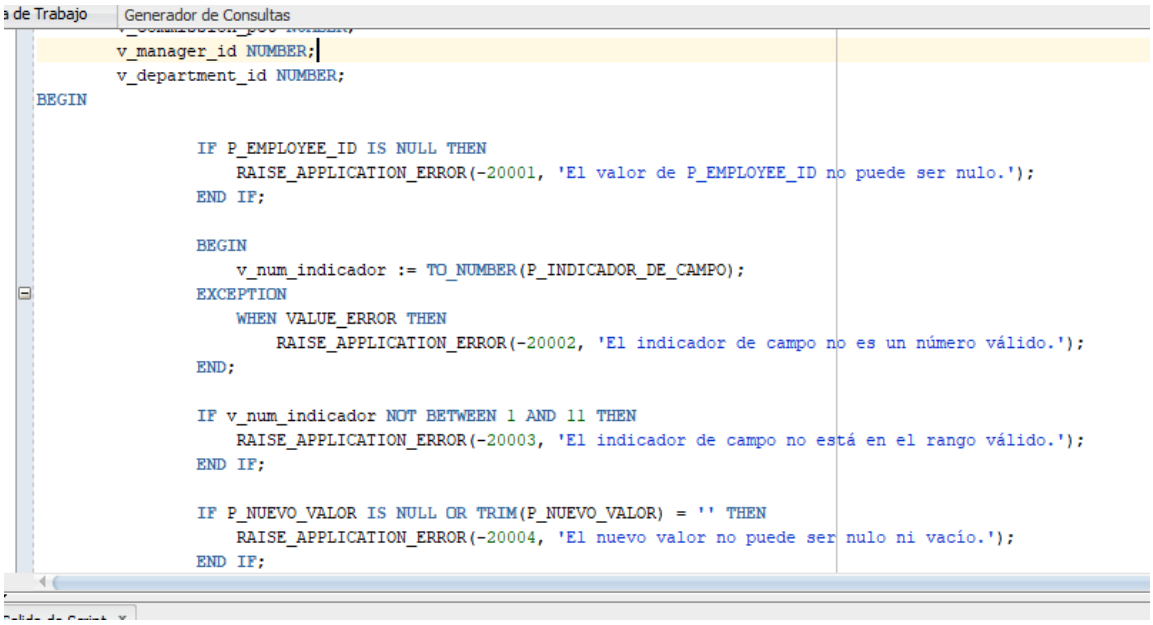

BAJA
2 de 3
Hoja de Trabajo
Generador de Consultas
PHONE_NUMBER,
HIRE_DATE,
JOB_ID,
SALARY,
MANAGER_ID,
DEPARTMENT_ID
) VALUES (
V_NEW_EMPLOYEE_ID,
P_FIRST_NAME,
P_LAST_NAME,
P_EMAIL,
P_PHONE_NUMBER,
P_HIRE_DATE,
P_JOB_ID,
P_SALARY,
P_MANAGER_ID,
P_DEPARTMENT_ID
);
COMMIT;
DBMS_OUTPUT.PUT_LINE('INSERCION REALIZADA CORRECTAMENTE!.');


Hoja de Trabajo
Generador de Consultas
DBMS_OUTPUT.PUT_LINE('INSERCION REALIZADA CORRECTAMENTE!.');
EXCEPTION
WHEN OTHERS THEN
RAISE_APPLICATION_ERROR(-20020, 'Error inesperado: ' || SQLERRM);
END;
-- FIN PRIMER PROC
-- SEGUNDO PROC
PROCEDURE BAJA_EMPLEADO(P_EMP NUMBER)
IS
V_ID_EMPLEADO NUMBER;
BEGIN
-- VALIDAR LA EXISTENCIA DEL TRABAJO
SELECT COUNT(*) INTO V_ID_EMPLEADO
FROM EMPLOYEES
WHERE EMPLOYEE_ID = P_EMP;

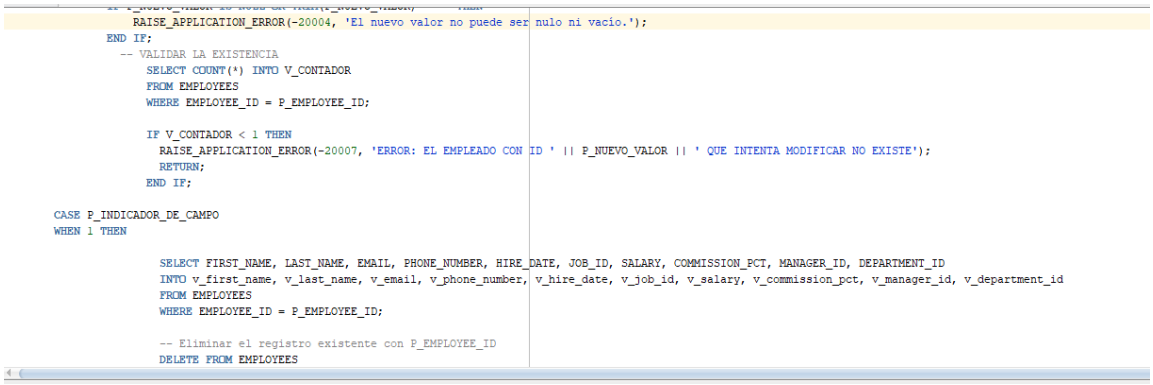

Hoja de Trabajo
Generador de Consultas
WHERE EMPLOYEE_ID = P_EMP;
IF V_ID_EMPLEADO < 1 THEN
RAISE_APPLICATION_ERROR(-20020, 'ERROR: EL EMPLEADO CON ID ' || P_EMP || ' QUE INTENTA ELIMINAR NO EXISTE.');
RETURN;
END IF;
-- ACTUALIZAR
-- UPDATE CONTROL_LOG SET COD_EMPLEADO = 1 WHERE COD_EMPLEADO = P_EMP;
-- ELIMINACION
DELETE FROM EMPLOYEES WHERE EMPLOYEE_ID = P_EMP;
COMMIT;
INSERT INTO CONTROL_LOG (LOG_ID, COD_EMPLEADO, FECHA, TABLA, COD_OPERACION)
SELECT LOG_ID, COD_EMPLEADO, FECHA, TABLA, COD_OPERACION
FROM control_log_tmp ;
-- Limpie la tabla temporal control_log_tmp
DELETE FROM control_log_tmp;
COMMIT;
END;


Salida de Script
Tarea terminada en 0.039 segundos

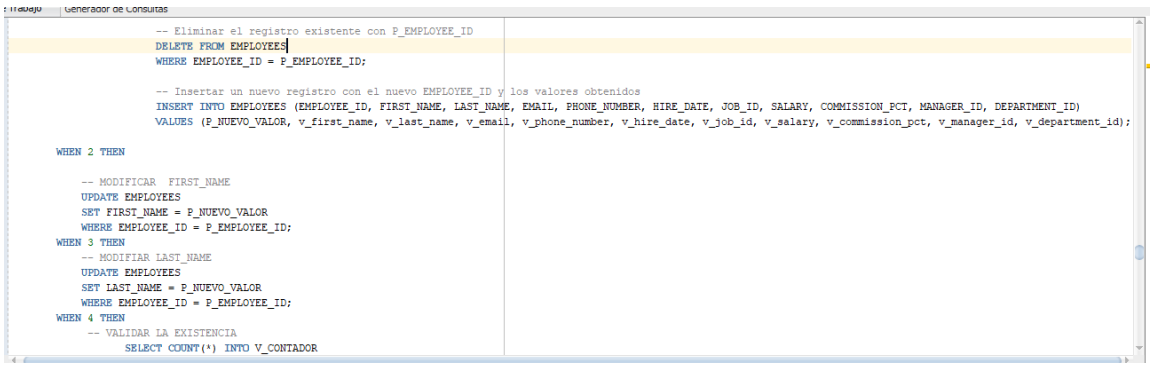

Hoja de Trabajo
Generador de Consultas
END;
-- FIN SEGUNDO PROC
-- TERCER PROC
PROCEDURE MOD_EMPLEADO (
P_EMPLOYEE_ID NUMBER,
P_NUEVO_VALOR VARCHAR2,
P_INDICADOR_DE_CAMPO NUMBER
)IS
v_num_indicador NUMBER;
V_CONTADOR NUMBER ;
v_first_name VARCHAR2(255);
v_last_name VARCHAR2(255);
v_email VARCHAR2(255);
v_phone_number VARCHAR2(255);
v_hire_date DATE;
v_job_id VARCHAR2(255);
v_salary NUMBER;
v_commission_pct NUMBER;
v_manager_id NUMBER;



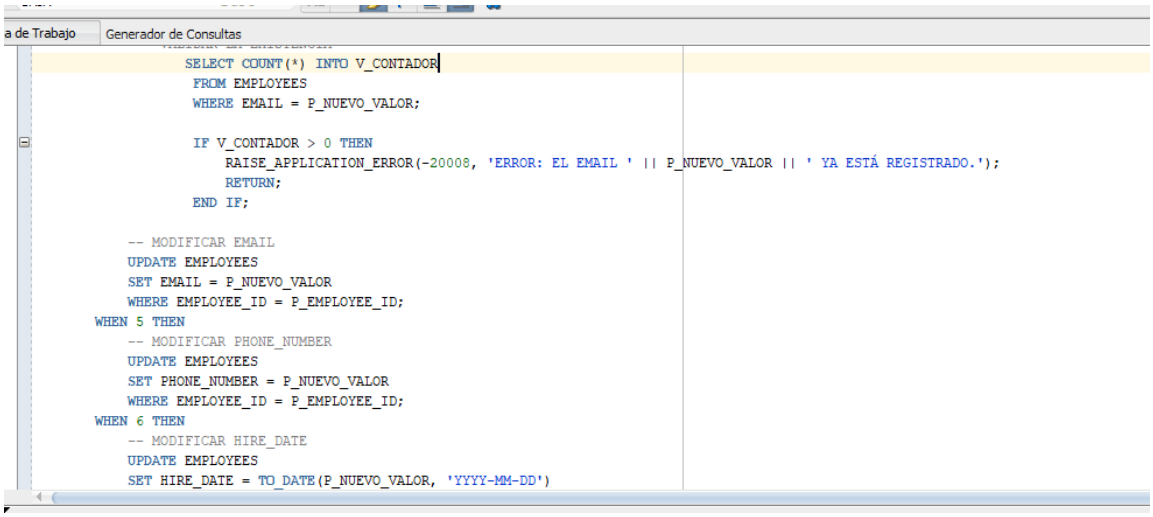

```
        v_manager_id NUMBER;
        v_department_id NUMBER;
BEGIN

                IF P_EMPLOYEE_ID IS NULL THEN
                    RAISE_APPLICATION_ERROR(-20001, 'El valor de P_EMPLOYEE_ID no puede ser nulo.');
                END IF;

                BEGIN
                    v_num_indicador := TO_NUMBER(P_INDICADOR_DE_CAMPO);
                EXCEPTION
                    WHEN VALUE_ERROR THEN
                        RAISE_APPLICATION_ERROR(-20002, 'El indicador de campo no es un número válido.');
                END;

                IF v_num_indicador NOT BETWEEN 1 AND 11 THEN
                    RAISE_APPLICATION_ERROR(-20003, 'El indicador de campo no está en el rango válido.');
                END IF;

                IF P_NUEVO_VALOR IS NULL OR TRIM(P_NUEVO_VALOR) = '' THEN
                    RAISE_APPLICATION_ERROR(-20004, 'El nuevo valor no puede ser nulo ni vacío.');
                END IF;
```



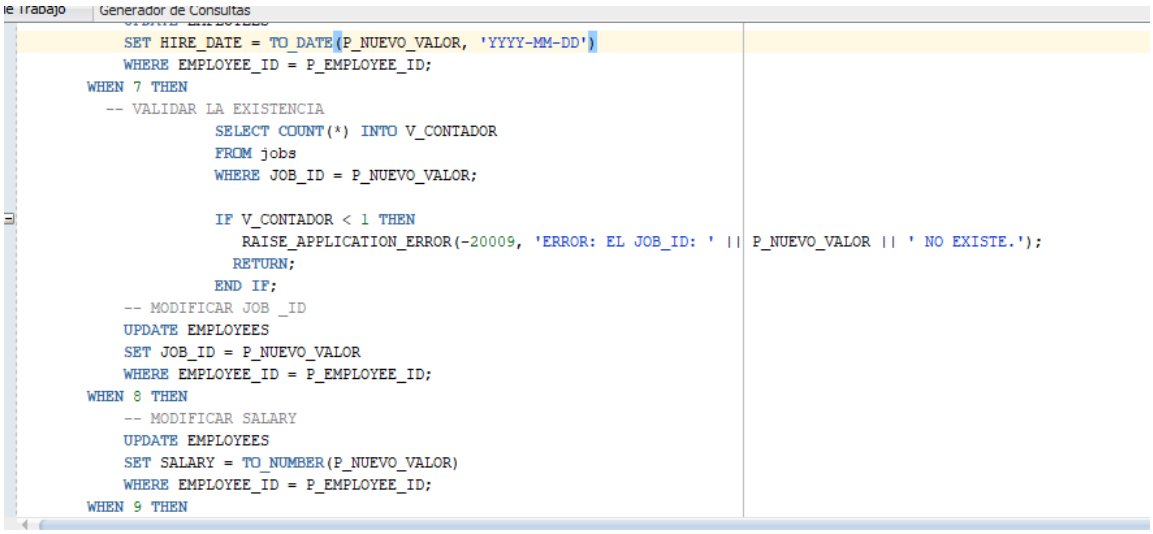
```
            RAISE_APPLICATION_ERROR(-20004, 'El nuevo valor no puede ser nulo ni vacío.');
        END IF;
          -- VALIDAR LA EXISTENCIA
            SELECT COUNT(*) INTO V_CONTADOR
            FROM EMPLOYEES
            WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;

            IF V_CONTADOR < 1 THEN
              RAISE_APPLICATION_ERROR(-20007, 'ERROR: EL EMPLEADO CON ID ' || P_NUEVO_VALOR || ' QUE INTENTA MODIFICAR NO EXISTE');
              RETURN;
            END IF;

CASE P_INDICADOR_DE_CAMPO
WHEN 1 THEN

              SELECT FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID, DEPARTMENT_ID
              INTO v_first_name, v_last_name, v_email, v_phone_number, v_hire_date, v_job_id, v_salary, v_commission_pct, v_manager_id, v_department_id
              FROM EMPLOYEES
              WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;

              -- Eliminar el registro existente con P_EMPLOYEE_ID
              DELETE FROM EMPLOYEES
```



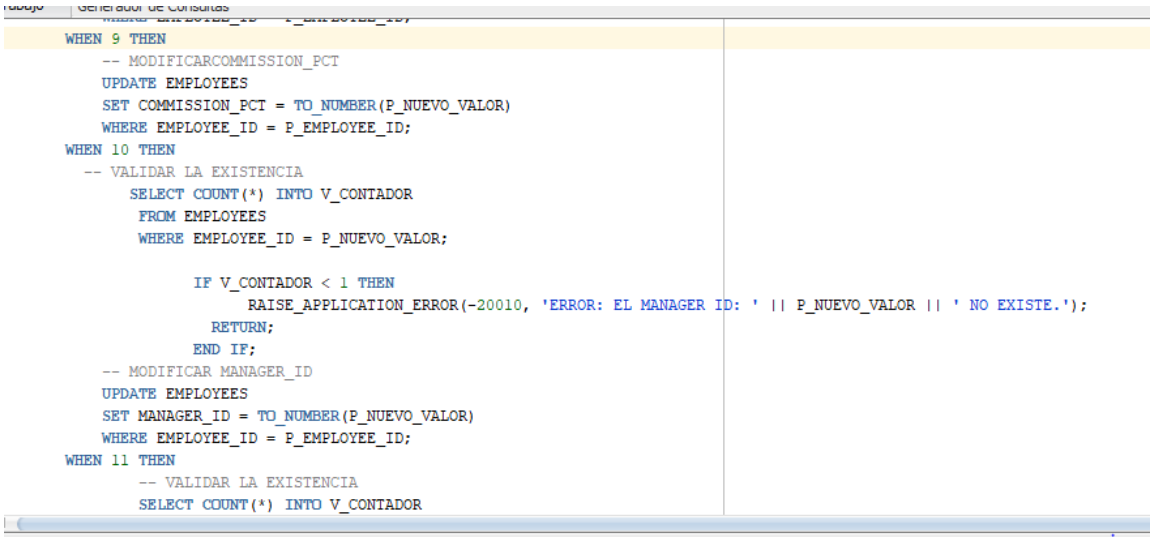
```
              -- Eliminar el registro existente con P_EMPLOYEE_ID
              DELETE FROM EMPLOYEES
              WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;

              -- Insertar un nuevo registro con el nuevo EMPLOYEE_ID y los valores obtenidos
              INSERT INTO EMPLOYEES (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID, DEPARTMENT_ID)
              VALUES (P_NUEVO_VALOR, v_first_name, v_last_name, v_email, v_phone_number, v_hire_date, v_job_id, v_salary, v_commission_pct, v_manager_id, v_department_id);

WHEN 2 THEN

    -- MODIFICAR  FIRST_NAME
    UPDATE EMPLOYEES
    SET FIRST_NAME = P_NUEVO_VALOR
    WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;
WHEN 3 THEN
    -- MODIFIAR LAST_NAME
    UPDATE EMPLOYEES
    SET LAST_NAME = P_NUEVO_VALOR
    WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;
WHEN 4 THEN
     -- VALIDAR LA EXISTENCIA
          SELECT COUNT(*) INTO V_CONTADOR
```



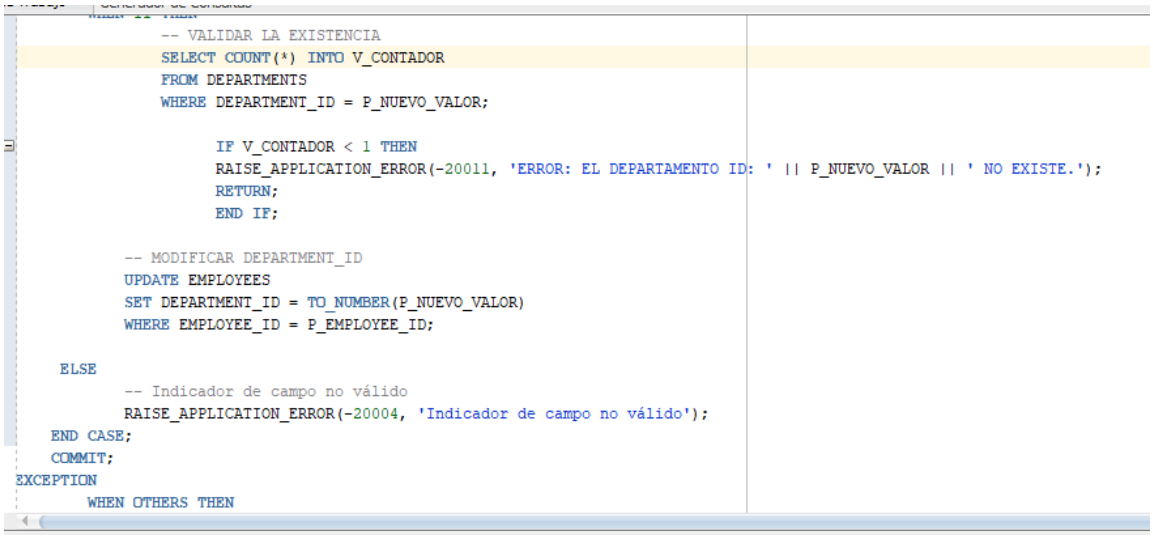
```
a de Trabajo | Generador de Consultas

                SELECT COUNT(*) INTO V_CONTADOR
                 FROM EMPLOYEES
                 WHERE EMAIL = P_NUEVO_VALOR;

                 IF V_CONTADOR > 0 THEN
                     RAISE_APPLICATION_ERROR(-20008, 'ERROR: EL EMAIL ' || P_NUEVO_VALOR || ' YA ESTÁ REGISTRADO.');
                     RETURN;
                 END IF;

        -- MODIFICAR EMAIL
        UPDATE EMPLOYEES
        SET EMAIL = P_NUEVO_VALOR
        WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;
    WHEN 5 THEN
        -- MODIFICAR PHONE_NUMBER
        UPDATE EMPLOYEES
        SET PHONE_NUMBER = P_NUEVO_VALOR
        WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;
    WHEN 6 THEN
        -- MODIFICAR HIRE_DATE
        UPDATE EMPLOYEES
        SET HIRE_DATE = TO_DATE(P_NUEVO_VALOR, 'YYYY-MM-DD')
```



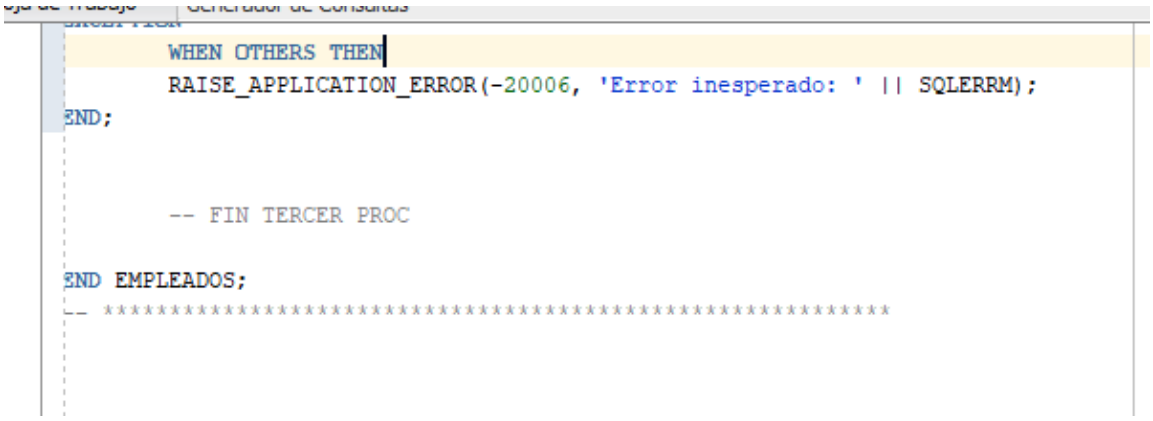
```
le Trabajo | Generador de Consultas

        SET HIRE_DATE = TO_DATE(P_NUEVO_VALOR, 'YYYY-MM-DD')
        WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;
    WHEN 7 THEN
      -- VALIDAR LA EXISTENCIA
                SELECT COUNT(*) INTO V_CONTADOR
                FROM jobs
                WHERE JOB_ID = P_NUEVO_VALOR;

                IF V_CONTADOR < 1 THEN
                   RAISE_APPLICATION_ERROR(-20009, 'ERROR: EL JOB_ID: ' || P_NUEVO_VALOR || ' NO EXISTE.');
                  RETURN;
                END IF;
        -- MODIFICAR JOB _ID
        UPDATE EMPLOYEES
        SET JOB_ID = P_NUEVO_VALOR
        WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;
    WHEN 8 THEN
        -- MODIFICAR SALARY
        UPDATE EMPLOYEES
        SET SALARY = TO_NUMBER(P_NUEVO_VALOR)
        WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;
    WHEN 9 THEN
```



```
    WHEN 9 THEN
        -- MODIFICARCOMMISSION_PCT
        UPDATE EMPLOYEES
        SET COMMISSION_PCT = TO_NUMBER(P_NUEVO_VALOR)
        WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;
    WHEN 10 THEN
      -- VALIDAR LA EXISTENCIA
          SELECT COUNT(*) INTO V_CONTADOR
           FROM EMPLOYEES
           WHERE EMPLOYEE_ID = P_NUEVO_VALOR;

                IF V_CONTADOR < 1 THEN
                      RAISE_APPLICATION_ERROR(-20010, 'ERROR: EL MANAGER ID: ' || P_NUEVO_VALOR || ' NO EXISTE.');
                  RETURN;
                END IF;
        -- MODIFICAR MANAGER_ID
        UPDATE EMPLOYEES
        SET MANAGER_ID = TO_NUMBER(P_NUEVO_VALOR)
        WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;
    WHEN 11 THEN
           -- VALIDAR LA EXISTENCIA
           SELECT COUNT(*) INTO V_CONTADOR
```



```
            -- VALIDAR LA EXISTENCIA
            SELECT COUNT(*) INTO V_CONTADOR
            FROM DEPARTMENTS
            WHERE DEPARTMENT_ID = P_NUEVO_VALOR;

                IF V_CONTADOR < 1 THEN
                RAISE_APPLICATION_ERROR(-20011, 'ERROR: EL DEPARTAMENTO ID: ' || P_NUEVO_VALOR || ' NO EXISTE.');
                RETURN;
                END IF;

        -- MODIFICAR DEPARTMENT_ID
        UPDATE EMPLOYEES
        SET DEPARTMENT_ID = TO_NUMBER(P_NUEVO_VALOR)
        WHERE EMPLOYEE_ID = P_EMPLOYEE_ID;

    ELSE
        -- Indicador de campo no válido
        RAISE_APPLICATION_ERROR(-20004, 'Indicador de campo no válido');
    END CASE;
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN
```



```
        WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20006, 'Error inesperado: ' || SQLERRM);
END;


        -- FIN TERCER PROC

END EMPLEADOS;
-- ******************************************************************
```

USO

```
SET SERVEROUTPUT ON
BEGIN
    INGRESO_EMPLEADO.INSETAR_EMPLEADO(
        'NOMBRE',
        'APELLIDO',
        'MAIL441',
        '503-7777-0000',
         SYSDATE,
        'SA_REP',
         50000,
         100,
         50
    );
END;
```

Salida de Script

Tarea terminada en 0.053 segundos

```
INSERCION REALIZADA CORRECTAMENTE!.

Procedimiento PL/SQL terminado correctamente.
```

Hoja de Trabajo | Generador de Consultas

```
         SYSDATE,
        'SA_REP',
         50000,
         100,
         50
    );
END;

BEGIN
    EMPLEADOS.MOD_EMPLEADO(751,'EMAIL22',4);

END;
BEGIN
    EMPLEADOS.BAJA_EMPLEADO(750);
END;
```

Salida de Script

Tarea terminada en 0.056 segundos

```
END;
Informe de error -
ORA-20006: Error inesperado: ORA-20007: ERROR: EL EMPLEADO CON ID EMAIL22 QUE INTENTA MODIFICAR NO EXISTE
ORA-06512: en "HR.EMPLEADOS", línea 326
ORA-06512: en línea 2
```

```
            50
        );
END;

BEGIN
    EMPLEADOS.MOD_EMPLEADO(207,'EMAIL22',4);

END;
BEGIN
    EMPLEADOS.BAJA_EMPLEADO(750);
END;
```

Salida de Script

Tarea terminada en 0.089 segundos

```
Procedimiento PL/SQL terminado correctamente.
```

FINAL .sql | Recursos Humanos1.sql | Recursos Humanos~1 | CONTROL_LOG | EMPLOYEES

Columnas | Datos | Model | Restricciones | Permisos | Estadísticas | Disparadores | Flashback | Dependencias | Detalles | Particiones | Índices | SQL

Ordenar... | Filtrar: | Acciones...

| | EMPLOYEE_ID | FIRST_NAME | LAST_NAME | EMAIL | PHONE_NUMBER | HIRE_DATE | JOB_ID |
|---|---|---|---|---|---|---|---|
| 1 | 207 | NOMBRE | APELLIDO | EMAIL22 | 503-7777-0000 | 01/09/23 | SA_REP |
| 2 | 206 | William | Gietz | WGIETZ | 515.123.8181 | 07/06/02 | AC_ACCOU |
| 3 | 205 | Shelley | Higgins | SHIGGINS | 515.123.8080 | 07/06/02 | AC_MGR |
| 4 | 204 | Hermann | Baer | HBAER | 515.123.8888 | 07/06/02 | PR_REP |
| 5 | 203 | Susan | Mavris | SMAVRIS | 515.123.7777 | 07/06/02 | HR_REP |
| 6 | 202 | Pat | Fay | PFAY | 603.123.6666 | 17/08/05 | MK_REP |
| 7 | 201 | Michael | Hartstein | MHARTSTE | 515.123.5555 | 17/02/04 | MK_MAN |
| 8 | 200 | Jennifer | Whalen | JWHALEN | 515.123.4444 | 17/09/03 | AD_ASST |

```
END;
BEGIN
    EMPLEADOS.BAJA_EMPLEADO(750);
END;
```

Salida de Script

Tarea terminada en 0.044 segundos

```
    EMPLEADOS.BAJA_EMPLEADO(750);
END;
Informe de error -
ORA-20020: ERROR: EL EMPLEADO CON ID 750 QUE INTENTA ELIMINAR NO EXISTE.
ORA-06512: en "HR.EMPLEADOS", línea 142
ORA-06512: en línea 2
```

```
BEGIN
    EMPLEADOS.MOD_EMPLEADO(207,'EMAIL22',4);

END;
BEGIN
    EMPLEADOS.BAJA_EMPLEADO(207);
END;
```

Salida de Script

Tarea terminada en 0.035 segundos

```
END;
Informe de error -
ORA-20020: ERROR: EL EMPLEADO CON ID 750 QUE INTENTA ELIMINAR NO EXISTE.
ORA-06512: en "HR.EMPLEADOS", línea 142
ORA-06512: en línea 2

Procedimiento PL/SQL terminado correctamente.
```

Columnas | Datos | Model | Restricciones | Permisos | Estadísticas | Disparadores | Flashback | Dependencias | Detalles | Particiones | Índices | SQL

Ordenar... | Filtrar:

| | LOG_ID | COD_EMPLEADO | FECHA | TABLA | COD_OPERACION |
|---|---|---|---|---|---|
| 1 | 85 | (null) | 01/09/23 | EMPLOYEES | DELETE empleado COD_EMPLEADO: 207 |
| 2 | 83 | (null) | 01/09/23 | EMPLOYEES | INSERT empleado 207 |
| 3 | 84 | (null) | 01/09/23 | EMPLOYEES | UPDATE empleado COD_EMPLEADO: 207 |

FINAL .sql | Recursos Humanos1.sql | Recursos Humanos~1 | CONTROL_LOG | EMPLOYEES

Columnas | Datos | Model | Restricciones | Permisos | Estadísticas | Disparadores | Flashback | Dependencias | Detalles | Particiones | Índices | SQL

Ordenar... | Filtrar: | Acciones...

| | EMPLOYEE_ID | FIRST_NAME | LAST_NAME | EMAIL | PHONE_NUMBER | HIRE_DATE | JOB_ID |
|---|---|---|---|---|---|---|---|
| 1 | 206 | William | Gietz | WGIETZ | 515.123.8181 | 07/06/02 | AC_ACCOU |
| 2 | 205 | Shelley | Higgins | SHIGGINS | 515.123.8080 | 07/06/02 | AC_MGR |
| 3 | 204 | Hermann | Baer | HBAER | 515.123.8888 | 07/06/02 | PR_REP |
| 4 | 203 | Susan | Mavris | SMAVRIS | 515.123.7777 | 07/06/02 | HR_REP |
| 5 | 202 | Pat | Fay | PFAY | 603.123.6666 | 17/08/05 | MK_REP |
| 6 | 201 | Michael | Hartstein | MHARTSTE | 515.123.5555 | 17/02/04 | MK_MAN |
| 7 | 200 | Jennifer | Whalen | JWHALEN | 515.123.4444 | 17/09/03 | AD_ASST |