I.E.S. Jacarandá
Gestión de Base de Datos
PL/SQL
Relación de ejercicios

1. Realiza un procedimiento que reciba un número de departamento y muestra por pantalla su nombre y localidad.

```
CREATE OR REPLACE PROCEDURE EJER1(numero NUMBER)
IS

nombre DEPT.DNAME%TYPE;
localidad DEPT.LOC%TYPE;

BEGIN

SELECT DNAME, LOC INTO nombre, localidad FROM DEPT WHERE DEPTNO = numero;

DBMS_OUTPUT.PUT_LINE(nombre || ' ' || localidad);

END;
/

SET SERVEROUTPUT ON
BEGIN
EJER1(10);
END;
/
```

2. Realiza una función devolver_sal que reciba un nombre de departamento y devuelva la suma de sus salarios.

```
CREATE OR REPLACE FUNCTION devolver_sal(nombre DEPT.DNAME%TYPE) RETURN NUMBER
IS

sumando NUMBER(5);

BEGIN

SELECT SUM(sal) INTO sumando FROM DEPT JOIN EMP ON DEPT.DNAME = nombre WHERE DEPT.DEPTNO = EMP.DEPTNO;
RETURN sumando;

END;
/

BEGIN
DBMS_OUTPUT.PUT_LINE(devolver_sal('SALES'));
END;
/
```

3. Realiza un procedimiento que reciba un número de departamento y muestre una lista de sus empleados.

```
CREATE OR REPLACE PROCEDURE EJER3(numero NUMBER)
IS

CURSOR nombreE IS SELECT ENAME FROM EMP WHERE DEPTNO = numero;

BEGIN

   FOR v_nombreE IN nombreE LOOP
      DBMS_OUTPUT.PUT_LINE('EMPLEADO: ' || v_nombreE.ENAME);
   END LOOP;
```

```
END;
/

SET SERVEROUTPUT ON
BEGIN
EJER3(10);
END;
/
```

4. Realiza un procedimiento MostrarJefes que reciba el nombre de un departamento y muestre los nombres de los empleados de ese departamento que son jefes de otros empleados. Trata las excepciones que consideres necesarias.

```
CREATE OR REPLACE PROCEDURE MostrarJefes(nombre DEPT.DNAME%TYPE)
IS

    CURSOR nombreJ IS SELECT DISTINCT E1.ENAME FROM EMP E1, EMP E2, DEPT WHERE E1.EMPNO = E2.MGR AND
E1.DEPTNO = DEPT.DEPTNO AND DEPT.DNAME = nombre;

BEGIN

  FOR v_nombresJ IN nombreJ LOOP
    DBMS_OUTPUT.PUT_LINE('ESTE HOMBRE ES JEFE: ' ||  ' ' || v_nombresJ.ENAME);
  END LOOP;

END;
/

SET SERVEROUTPUT ON
BEGIN
  MostrarJefes('RESEARCH');
END;
/
```

5. Hacer un procedimiento que muestre el nombre y el salario del empleado cuyo código es 7082

```
CREATE OR REPLACE PROCEDURE EJER5
IS

nombre EMP.ENAME%TYPE;
salario EMP.SAL%TYPE;

BEGIN

SELECT EMP.ENAME, EMP.SAL INTO nombre, salario FROM EMP WHERE EMP.EMPNO = 7082;

EXCEPTION
WHEN no_data_found THEN
DBMS_OUTPUT.PUT_LINE('NO EXISTE');

END;
/

SET SERVEROUTPUT ON
BEGIN
EJER5();
END;
/
```

6. Realiza un procedimiento llamado HallarNumEmp que recibiendo un nombre de departamento, muestre en pantalla el número de empleados de dicho departamento
   Si el departamento no tiene empleados deberá mostrar un mensaje informando de ello. Si el departamento no existe se tratará la excepción correspondiente.

```
CREATE OR REPLACE PROCEDURE HallarNumEmp(nombreD DEPT.DNAME%TYPE)
IS

numEmp NUMBER(2);

BEGIN
```

```
SELECT COUNT(EMP.ENAME) INTO numEmp FROM DEPT, EMP WHERE DEPT.DNAME = nombreD;
DBMS_OUTPUT.PUT_LINE('EMPLEADOS: ' || numEmp);

EXCEPTION
WHEN no_data_found THEN
DBMS_OUTPUT.PUT_LINE('NO HAY DATOS');

END;
/

SET SERVEROUTPUT ON
BEGIN
HallarNumEmp('RESEARCH');
END;
/
```

7. ~~Hacer un procedimiento que reciba como parámetro un código de empleado y devuelva su nombre, junto con los códigos y las fechas de todas sus facturas.~~

8. Codificar un procedimiento que permita borrar un empleado cuyo número se pasará en la llamada. Si no existiera dar el correspondiente mensaje de error.

```
CREATE OR REPLACE PROCEDURE EJER8(numEmp EMP.EMPNO%TYPE)
IS
BEGIN

DELETE FROM EMP
WHERE EMP.EMPNO = numEmp;

EXCEPTION
WHEN no_data_found THEN
DBMS_OUTPUT.PUT_LINE('NO EXISTE DATO');

END;
/

BEGIN
EJER8(7369);
END;
/
```

9. **Realiza un procedimiento MostrarCostesSalariales que muestre los nombres de todos los departamentos y el coste salarial de cada uno de ellos**

```
CREATE OR REPLACE PROCEDURE MostrarCostesSalariales
IS

CURSOR nombreD IS SELECT * FROM DEPT;
aux DEPT.DEPTNO%TYPE;
CURSOR salarioD IS SELECT SUM(EMP.SAL) FROM EMP WHERE aux = DEPTNO;

BEGIN

FOR v_nombreD IN nombreD LOOP
DBMS_OUTPUT.PUT_LINE(v_nombreD.DNAME);
aux := v_nombreD.DEPTNO;
   FOR v_salarioD IN salarioD LOOP
   DBMS_OUTPUT.PUT_LINE(v_salarioD.aux);
   END LOOP;
END LOOP;

END;
/

BEGIN
MostrarCostesSalariales();
END;
/
```

10. Escribir un procedimiento que modifique la localidad de un departamento. El procedimiento

recibirá como parámetros el número del departamento y la localidad nueva.

```
CREATE OR REPLACE PROCEDURE EJER10(locD DEPT.LOC%TYPE, numeroD DEPT.DEPTNO%TYPE)
IS

BEGIN

UPDATE DEPT
SET LOC = locD
WHERE DEPT.DEPTNO = numeroD;

END;
/

BEGIN
EJER10('SEVILLA', 30);
END;
/
```

11. Realiza un procedimiento MostrarMasAntiguos que muestre el nombre del empleado más antiguo de cada departamento junto con el nombre del departamento.

```
CREATE OR REPLACE PROCEDURE MostrarMasAntiguos
IS

aux DEPT.DEPTNO%TYPE;
aux2 DEPT.DNAME%TYPE;

CURSOR nombreD IS SELECT DEPT.DNAME, DEPT.DEPTNO FROM DEPT;

CURSOR nombreE IS SELECT EMP.ENAME FROM EMP, DEPT WHERE EMP.DEPTNO = aux AND ROWNUM = 1 ORDER BY
    EMP.HIREDATE;

BEGIN

FOR v_nombreD IN nombreD LOOP
DBMS_OUTPUT.PUT_LINE(v_nombreD.DNAME);
aux := v_nombreD.DEPTNO;
  FOR v_nombreE IN nombreE LOOP
  DBMS_OUTPUT.PUT_LINE('** ' || v_nombreE.ENAME);
  END LOOP;
END LOOP;

END;
/

SET SERVEROUTPUT ON
BEGIN
MostrarMasAntiguos();
END;
/
```

12. Crear un procedimiento que en la tabla emp incrementar el salario el 10% a los empleados que tengan una comisión superior al 5% del salario.

```
CREATE OR REPLACE PROCEDURE EJER12
IS

BEGIN

UPDATE EMP
SET SAL = SAL*1.1
WHERE EMP.COMM > EMP.SAL*0.05;

END;
/

SET SERVEROUTPUT ON
BEGIN
EJER12();
END;
/
```

13. Realiza un procedimiento MostrarMejoresVendedores que muestre los nombres de los dos vendedores con más comisiones.

```
CREATE OR REPLACE PROCEDURE MostrarMejoresVendedores
IS

CURSOR nombreE IS SELECT EMP.ENAME FROM EMP WHERE ROWNUM <= 2 ORDER BY EMP.COMM;

BEGIN
FOR v_nombreE IN nombreE LOOP
DBMS_OUTPUT.PUT_LINE(v_nombreE.ENAME);
END LOOP;
END;
/

SET SERVEROUTPUT ON
BEGIN
MostrarMejoresVendedores();
END;
/
```

14. Realiza un procedimiento MostrarsodaelpmE que reciba el nombre de un departamento al revés y muestre los nombres de los empleados de ese departamento.

```
CREATE OR REPLACE PROCEDURE MostrarsodaelpmE(nombreD DEPT.DNAME%TYPE)
IS

CURSOR nombresE IS SELECT EMP.ENAME FROM DEPT, EMP WHERE REVERSE(DEPT.DNAME) = nombreD AND
    EMP.DEPTNO = EMP.DEPTNO;

BEGIN

FOR v_nombresE IN nombresE LOOP
DBMS_OUTPUT.PUT_LINE(v_nombresE.ENAME);
END LOOP;

END;
/

SET SERVEROUTPUT ON
BEGIN
MostrarsodaelpmE('SELAS');
END;
/
```

15. **Realiza un procedimiento RecortarSueldos que recorte el sueldo un 20% a los empleados cuyo nombre empiece por la letra que recibe como parámetro.Trata las excepciones que consideres necesarias.**

```
CREATE OR REPLACE PROCEDURE RecortarSueldos(letra VARCHAR2)
IS

BEGIN

UPDATE EMP
SET SAL = SAL - (SAL * 0.2)
WHERE EMP.ENAME LIKE letra || '%';

EXCEPTION
WHEN no_data_found THEN
DBMS_OUTPUT.PUT_LINE('ERROR');

END;
/

BEGIN
RecortarSueldos('S');
END;
/
```

16. Realiza un procedimiento BorrarBecarios que borre a los dos empleados más nuevos de cada

departamento

```
CREATE OR REPLACE PROCEDURE BorrarBecarios

IS

aux DEPT.DEPTNO%TYPE;
aux2 DEPT.DNAME%TYPE;

CURSOR numeroD IS SELECT DEPT.DEPTNO INTO aux FROM DEPT;

CURSOR nombreE IS SELECT * FROM (SELECT ENAME FROM EMP WHERE ROWNUM >= 1  AND EMP.DEPTNO = aux ORDER BY
      HIREDATE DESC) WHERE rownum <= 2;

BEGIN

FOR v_numeroD IN numeroD LOOP
aux := v_numeroD.DEPTNO;
   FOR v_nombreE IN nombreE LOOP
   DELETE FROM EMP WHERE EMP.ENAME =  v_nombreE.ENAME;
   END LOOP;
END LOOP;

END;
/

SET SERVEROUTPUT ON
BEGIN
BorrarBecarios();
END;
/
```