1. **PRÁCTICA PL/SQL – 7**
2. Codificar un procedimiento que permita borrar un empleado cuyo número se pasa en la llamada.

**CREATE OR REPLACE PROCEDURE BORRAR\_EMPLEADO**

**(V\_EMPNO EMP.EMPNO%TYPE)**

**AS**

**BEGIN**

**DELETE FROM EMP WHERE EMPNO = V\_EMPNO;**

**END BORRAR\_EMPLEADO;**

**/**

**EXECUTE BORRAR\_EMPLEADO(9998);**

**SELECT \* FROM EMP WHERE EMPNO=9998;**

1. Escribir un procedimiento que modifique la localidad de un departamento. El procedimiento recibirá como parámetros el número de departamento y la nueva localidad.

**CREATE OR REPLACE PROCEDURE MODIFICAR\_LOCALIDAD(**

**V\_DEPTNO NUMBER,**

**LOCALIDAD VARCHAR2)**

**AS**

**BEGIN**

**UPDATE DEPT SET LOC = LOCALIDAD**

**WHERE DEPTNO = V\_DEPTNO;**

**END MODIFICAR\_LOCALIDAD;**

**/**

**EXECUTE MODIFICAR\_LOCALIDAD(10, 'LOCNUEVA');**

**SELECT \* FROM DEPT WHERE DEPTNO=10;**

1. Crear un procedimiento llamado VEREMPLEADO que introduciendo el código de un departamento por teclado permita ver el nombre del departamento y la información de todos sus empleados. Este procedimiento constará de un cursor.

**CREATE OR REPLACE PROCEDURE** VEREMPLEADO

**(V\_DEPTNO DEPT.DEPTNO%TYPE)**

**IS**

**V\_DEPTNOMBRE DEPT.DNAME%TYPE;**

**V\_NUM\_EMP NUMBER;**

**CURSOR C IS**

**SELECT d.DNAME, e.EMPNO, e.ENAME, e.JOB, e.SAL, e.DEPNO**

**FROM TEMPLE e, DEPT d**

**WHERE d.DEPTNO = e.DEPTNO(+) AND e.DEPTNO=V\_DEPTNO**

**GROUP BY d.DNAME;**

**C\_TEMPLE C%ROWTYPE;**

**BEGIN**

**OPEN C;**

**FETCH C INTO C\_TEMPLE;**

**WHILE C%FOUND LOOP**

**DBMS\_OUTPUT.PUT\_LINE(C\_TEMPLE);**

**FETCH C INTO C\_TEMPLE;**

**END LOOP;**

**CLOSE C;**

**END VEREMPLEADO;**

**/**

**EXECUTE** VEREMPLEADO **(10);**

**SELECT \* FROM DEPT WHERE DEPTNO=10;**

1. Escribir un procedimiento que permita ver en la tabla TEMPLE, mediante un cursor, si el año de ingreso de los empleados fue bisiesto o no. ¿Podríamos hacer un procedimiento genérico que sirviera tanto para la tabla TEMPLE como para la tabla EMP?

-- Si a este procedimiento le pasamos un año nos devuelve SI es bisiesto o NO.

CREATE OR REPLACE PROCEDURE ANNO(FECHA IN DATE, BISIESTO OUT CHAR(2))

IS

V\_YEAR NUMBER;

BEGIN

V\_YEAR:= TO\_NUMBER(TO\_CHAR(FECHA, 'YYYY'));

IF (MOD(V\_YEAR,4)=0 AND MOD(V\_YEAR,100)<>0) OR

MOD(V\_YEAR,400)=0 THEN

BISIESTO:=’SI’;

ELSE

BISIESTO:=’NO’;

END IF;

END ANNO;

1. Escribir una función que para cada empleado de la tabla TEMPLE, devuelva los trienios que hay entre su fecha de nacimiento y su fecha de incorporación. (Un trienio son tres años completos).

**CREATE OR REPLACE FUNCTION TRIENIOS\_TEMPLE**

**(V\_DEPTNO DEPT.DEPTNO%TYPE)**

**RETURN VARCHAR2**

**IS**

**V\_TRIENIOS NUMBER(6);**

**CURSOR C IS**

**SELECT \* FROM TEMPLE;**

**RC C%ROWTYPE;**

**BEGIN**

**FETCH C INTO RC;**

**WHILE C%FOUND LOOP**

**V\_TRIENIOS := ABS(TRUNC(MONTHS\_BETWEEN(RC.FECNA,RC.FECIN) / 3));**

**DBMS\_OUTPUT.PUT\_LINE(RC.NOMEM || RC.FECNA || RC.FECIN || V\_TRIENIOS);**

**END LOOP;**

**END TRIENIOS\_TEMPLE;**

**/**

**DECLARE**

**BEGIN**

**TRIENIOS\_TEMPLE (10);**

**END;**

**/**

1. Escribir un procedimiento que reciba una cadena y visualice el apellido y el número de empleado de todos los empleados cuyo apellido contenga la cadena especificada. Al finalizar visualizar el número de empleados mostrados.

CREATE OR REPLACE PROCEDURE ver\_emple\_apell(

cadena VARCHAR2)

AS

cad VARCHAR2(10);

CURSOR c\_emple IS

SELECT apellido, emp\_no FROM emple

WHERE apellido LIKE cad;

vr\_emple c\_emple%ROWTYPE;

BEGIN

cad :='%'||cadena||'%';

OPEN c\_emple;

FETCH c\_emple INTO vr\_emple;

WHILE (c\_emple%FOUND) LOOP

DBMS\_OUTPUT.PUT\_LINE(vr\_emple.emp\_no||' \* '||vr\_emple.apellido);

FETCH c\_emple INTO vr\_emple;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('NUMERO DE EMPLEADOS: '|| c\_emple%ROWCOUNT);

CLOSE c\_emple;

END ver\_emple\_apell;

1. Haz un procedimiento (usando cursores) donde visualicemos los departamentos y el total de dinero destinado en salarios por cada departamento.

CREATE OR REPLACE PROCEDURE EJ17

AS

CURSOR C1 IS SELECT D.DEPT\_NO, DNOMBRE,

NVL(SUM(SALARIO), 0)+NVL(SUM(COMISION), 0)

FROM EMPLE E, DEPART D

WHERE E.DEPT\_NO (+)=D.DEPT\_NO

GROUP BY D.DEPT\_NO, DNOMBRE

ORDER BY DEPT\_NO;

v\_deptno DEPART.DEPT\_NO%TYPE;

v\_dnombre DEPART.DNOMBRE%TYPE;

v\_total NUMBER(10);

BEGIN

OPEN C1;

LOOP

FETCH C1 INTO v\_deptno, v\_dnombre, v\_total;

EXIT WHEN C1%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(rpad(v\_deptno, 5)||' '||rpad(v\_dnombre, 20)||' '||v\_total);

END LOOP;

CLOSE C1;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error '||sqlcode||' '||sqlerrm);

END;

/

1. Haz un procedimiento (usando cursores) donde visualicemos los oficios de los empleados y el total de dinero destinado en sueldos (salario + comisión) por cada oficio. Ordena el listado de forma descendente por dinero, es decir, el oficio más caro primero.

CREATE OR REPLACE PROCEDURE EJ19

AS

CURSOR C1 IS SELECT OFICIO, SUM(NVL(SALARIO,0)+NVL(COMISION,0))

FROM EMPLE GROUP BY OFICIO

ORDER BY SUM(SALARIO+NVL(COMISION,0)) DESC;

v\_oficio EMPLE.OFICIO%TYPE;

v\_total NUMBER(10);

BEGIN

OPEN C1;

LOOP

FETCH C1 INTO v\_oficio, v\_total;

EXIT WHEN C1%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(rpad(v\_oficio, 15)||v\_total);

END LOOP;

CLOSE C1;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error '||sqlcode||' '||sqlerrm);

END;

/

1. Haz un procedimiento con las mismas características que el anterior, pero muestra SOLAMENTE los dos oficios más caros.

CREATE OR REPLACE PROCEDURE EJ19

AS

CURSOR C1 IS SELECT OFICIO, SUM(NVL(SALARIO,0)+NVL(COMISION, 0))

FROM EMPLE GROUP BY OFICIO

ORDER BY SUM(SALARIO+NVL(COMISION, 0)) DESC;

v\_oficio EMPLE.OFICIO%TYPE;

v\_total NUMBER(10);

BEGIN

OPEN C1;

LOOP

FETCH C1 INTO v\_oficio, v\_total;

EXIT WHEN C1%NOTFOUND OR C1%ROWCOUNT>2;

DBMS\_OUTPUT.PUT\_LINE(rpad(v\_oficio, 15)||v\_total);

END LOOP;

CLOSE C1;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error '||sqlcode||' '||sqlerrm);

END;

/

1. Realizar un procedimiento que suba el salario a todos los empleados del departamento indicado en la llamada.
   * El cursor será diseñado con las cláusulas FOR UPDATE y WHERE CURRENT OF.
   * El cursor será diseñado con las cláusulas ROWID en lugar de FOR UPDATE.

El cursor será diseñado con las cláusulas FOR UPDATE y WHERE CURRENT OF.

create or replace procedure subir\_salario\_dpto

(Vp\_num\_dpto number,

Vp\_pct\_subida number)

as

cursor c\_emple is select oficio,salario

from emple

where dept\_no = vp\_num\_dpto

FOR UPDATE;

Vc\_reg\_emple c\_emple%rowtype;

V\_inc number(8,2);

begin

open c\_emple;

fetch c\_emple into vc\_reg\_emple;

while c\_emple%found loop

v\_inc := (vc\_reg\_emple.salario/100)\*Vp\_pct\_subida;

update emple set salario = salario + v\_inc

WHERE CURRENT OF C\_EMPLE;

FETCH C\_EMPLE INTO VC\_REG\_EMPLE;

end loop;

end subir\_salario\_dpto;

El cursor será diseñado con las cláusulas ROWID en lugar de FOR UPDATE.

create or replace procedure subir\_salario\_dpto\_B

(Vp\_num\_dpto number,

Vp\_pct\_subida number)

as

cursor c\_emple is select oficio, salario, ROWID

from emple

where dept\_no = vp\_num\_dpto;

Vc\_reg\_emple c\_emple%rowtype;

V\_inc number(8,2);

begin

open c\_emple;

fetch c\_emple into vc\_reg\_emple;

while c\_emple%found loop

v\_inc := (vc\_reg\_emple.salario/100)\*Vp\_pct\_subida;

update emple set salario = salario + v\_inc

WHERE ROWID = vc\_reg\_emple.ROWID;

FETCH C\_EMPLE INTO VC\_REG\_EMPLE;

end loop;

end subir\_salario\_dpto\_b;