ACTIVIDADES TEMA 6 PARTE III PL/SQL CURSORES - SOLUCIONES

1. Desarrollar un procedimiento que visualice el apellido y la fecha de alta de todos los empleados ordenados por apellidos

```
CREATE OR REPLACE PROCEDURE ver_t_emple
AS
      CURSOR c_emple IS
            SELECT APELLIDO, FECHA_ALT
            FROM EMPLE
            ORDER BY APELLIDO;
      v_apellido VARCHAR2(10);
      v_fecha DATE;
BEGIN
      OPEN c_emple;
      FETCH c_emple into v_apellido, v_fecha;
      WHILE c_emple%FOUND LOOP
        DBMS_OUTPUT.PUT_LINE( v_apellido||' * '||v_fecha);
        FETCH c_emple into v_apellido,v_fecha;
      END LOOP;
      CLOSE c_emple;
END ver_t_emple;
```

Recordar: Si usamos el FETCH es necesario previamente abrirlo (OPEN) y una vez finalizado será necesario cerrarlo (CLOSE).

NOTA: En muchos de ellos podemos utilizar el FOR..LOOP para cursores simplificándose mucho los ejercicios
Recordar: Si usamos el FOR con un cursor automáticamente se abre y una vez finalizado el bucle se cierra automáticamente. Así mismo la variable (v_reg en este caso) no será necesario declararla

Otra forma quizás más sencilla y clara de realizar el ejercicio

```
CREATE OR REPLACE PROCEDURE ver_t_emple
AS
      CURSOR c_emple IS SELECT APELLIDO, FECHA_ALT
                        FROM EMPLE
                        ORDER BY APELLIDO;
BEGIN
      for v_reg in c_emple LOOP
        DBMS_OUTPUT.PUT_LINE( v_reg.apellido ||' * '||v_reg.fecha_alt);
      END LOOP;
END ver_t_emple;
```

Recordar que cuando usamos el FOR … LOOP con un cursor la variable de control (V_REG) es un registro que contiene todos los campos del registro de la tabla sobre la que actual el cursor, en nuestro caso EMPLE. Cada vez que realizamos una pasada por el bucle se cargan automáticamente en cada uno de los campos de dicho registro los valores de la fila a la que estamos accediendo y para acceder a cada campo solo debemos de indicar V_REG.nom_campo

<---------- ***V_REG*** ---------->

Emp_no	***apellido***	***Oficio***	***Dir***	***Fecha_alt***	***Salario***	***Comision***	***Dept_no***

2. Procedimiento que muestre el nombre de cada departamento y el número de empleados que tiene, incluidos los departamentos que no tienen empleados.

```
CREATE OR REPLACE PROCEDURE ver_emple_depart
AS
      CURSOR c_emple IS
            SELECT dnombre, COUNT(emp_no)
            FROM emple e, depart d
            WHERE d.dept_no = e.dept_no(+)
            GROUP BY dnombre;
      v_dnombre depart.dnombre%TYPE;
      v_num_emple BINARY_INTEGER;
BEGIN
      OPEN c_emple;
      FETCH c_emple into v_dnombre, v_num_emple;
      WHILE c_emple%FOUND LOOP
        DBMS_OUTPUT.PUT_LINE(v_dnombre||' * '||v_num_emple);
        FETCH c_emple into v_dnombre,v_num_emple;
      END LOOP;
      CLOSE c_emple;
END ver_emple_depart;
```

Usando el FOR…LOOP

```
CREATE OR REPLACE PROCEDURE ver_emple_depart
AS
      CURSOR c_empleDepart IS SELECT dnombre, COUNT(emp_no) total FROM emple e, depart d
                                                        WHERE d.dept_no = e.dept_no(+)
                                                        GROUP BY dnombre;
      nomDept           depart.dnombre%TYPE;
      num_emple         number;
BEGIN
      FOR v_reg in  c_empleDepart LOOP
        nomDept         := v_reg.dnombre;
        num_emple       := v_reg.total;
        DBMS_OUTPUT.PUT_LINE('Departamento: '||nomDept||' Total emple: '||num_emple);
      END LOOP;
END ver_emple_depart;
```

3. Modificar el bloque anterior para que al final muestre también el número de empleados que tiene la empresa. Indicar cual es el cursor explícito y cual el implícito y que diferencia hay entre uno y otro

```
CREATE OR REPLACE PROCEDURE ver_emple_depart
AS
      CURSOR c_empleDepart IS SELECT dnombre, COUNT(emp_no) total FROM emple e, depart d
                                                    WHERE d.dept_no = e.dept_no(+)
                                                    GROUP BY dnombre;
      nomDept           depart.dnombre%TYPE;
      num_emple         number;
      total_emple       number default 0;
BEGIN
      FOR v_reg in  c_empleDepart LOOP
        nomDept         := v_reg.dnombre;
        num_emple       := v_reg.total;
        DBMS_OUTPUT.PUT_LINE('Departamento: '||nomDept||' Total emple: '||num_emple);
      END LOOP;
      Select count(*) into total_emple from emple;
      DBMS_OUTPUT.PUT_LINE('Total empleados en la empresa: '||total_emple);

END ver_emple_depart;
```

Cursor implícito: Cuando la sentencia SELECT devuelve solo un valor y se asigna a una variable → total_emple . También podemos devolver con el *SELECT* varios valores pero en una sola fila en cuyo caso tendremos que definir tantas variables como valores. Si el resultado de la SELECT devuelve más de una fila/registro no debemos utilizar un cursos implícito

Cursor explícito: cuando la sentencia SELECT devuelve una o mas filas y para procesarlo necesitamos recorrer dicho juego de resultados. → c_empleDepart

4. Procedimiento que visualice el apellido y el salario de los cinco empleados que tienen el salario más alto

```
CREATE OR REPLACE PROCEDURE emp_5maxsal
AS
      CURSOR c_emp IS
            SELECT apellido, salario FROM emple ORDER BY salario DESC;
      vr_emp c_emp%ROWTYPE;
      i NUMBER;
BEGIN
      i:=1;
      OPEN c_emp;
      FETCH c_emp INTO vr_emp;
      WHILE c_emp%FOUND AND i<=5 LOOP
        DBMS_OUTPUT.PUT_LINE(vr_emp.apellido ||' * '|| vr_emp.salario);
        FETCH c_emp INTO vr_emp;
        i:=I+1;
      END LOOP;
      CLOSE c_emp;
END emp_5maxsal;
```

5. Codificar un programa que visualice los dos empleados que ganan menos de cada oficio

Solución Utilizando RUPTURA DE CONTROL

```
CREATE OR REPLACE PROCEDURE emp_2minsal AS
      CURSOR c_emp IS
      SELECT apellido, oficio, salario FROM emple ORDER BY oficio, salario;
      vr_emp c_emp%ROWTYPE;
      oficio_ant EMPLE.OFICIO%TYPE;
      i NUMBER;
BEGIN
      OPEN c_emp;
      oficio_ant:='*';
      FETCH c_emp INTO vr_emp;
      WHILE c_emp%FOUND LOOP
        IF oficio_ant <> vr_emp.oficio THEN
          oficio_ant := vr_emp.oficio;
          i := 1;
        END IF;
        IF i <= 2 THEN
          DBMS_OUTPUT.PUT_LINE(vr_emp.oficio||' * '||vr_emp.apellido||' * '
                  ||vr_emp.salario);
        END IF;
        FETCH c_emp INTO vr_emp;
        i:=I+1;
      END LOOP;
      CLOSE c_emp;
END emp_2minsal;
```

Solución utilizando cursores, uno de ellos con paso de parámetros

```
CREATE OR REPLACE PROCEDURE emp_ganan_menos_por_oficio IS
   cursor c_oficio is select distinct oficio from u3.emple;
   cursor c_emp_por_oficio (p_oficio varchar2) is select emp_no, apellido, salario
                                                  from emple
                                                  where oficio = p_oficio order by salario;
   reg_c_oficio              c_oficio%rowtype;
   reg_c_emp_por_oficio      c_emp_por_oficio%rowtype;
   cont number;
BEGIN
 for reg_c_oficio IN c_oficio loop
      cont:=0;
      dbms_output.put_line('Empleados del oficio : '||reg_c_oficio.oficio);

      open c_emp_por_oficio(reg_c_oficio.oficio);
      fetch c_emp_por_oficio into reg_c_emp_por_oficio;
      while c_emp_por_oficio%found and cont<2loop
              cont:= cont+1;
              dbms_output.put_line('Numero empleado '||reg_c_emp_por_oficio.emp_no
                  ||'  Nombre empleado '||reg_c_emp_por_oficio.apellido ||
                    '  Salario '||reg_c_emp_por_oficio.salario );
                       fetch c_emp_por_oficio into reg_c_emp_por_oficio;
      end loop;
      close c_emp_por_oficio;
      dbms_output.put_line(' ');
 end loop;

END emp_ganan_menos_por_oficio;
```

6. Realizar un procedimiento que pasado un departamento como parámetro, muestre los empleados que tiene ordenado por apellidos.

```
CREATE OR REPLACE PROCEDURE p_emple_depart(dep number) IS
         cursor c(dep number)is select * from emple where dept_no = dep;

BEGIN
   for v_reg in c(dep) loop
         dbms_output.put_line(v_reg.emp_no||' ' ||v_Reg.apellido||' '||v_reg.oficio||' '
                              ||v_reg.dept_no);
   end loop;
END p_emple_depart;

--

Bloque que comprueba el funcionamiento
begin
       p_emple_depart(20);
end;
```

7. Realizar un bloque, utilizando el procedimiento anterior, muestre todos los empleados por cada departamento.

```
declare
      cursor c_depart is select dept_no, dnombre from depart;
begin
      for v_reg in c_depart loop
            dbms_output.put_line('NUM DEPARTAMENTO: '||v_reg.dept_no||
                  ' Nombre:'||v_Reg.dnombre);
      -- llamo al procedimiento para que me visualice los datos de los empleados del
departamento que estoy procesando
            p_emple_depart(v_reg.dept_no);
            dbms_output.put_line('   ');
      end loop;
end;
```

8. Realizar un bloque que muestre por cada departamento los empleados que tiene y dentro de un departamento ordenado por apellidos:

- ***Para cada empleado apellido y sueldo***
- ***Para cada departamento: número de empleados y suma de los salarios del departamento***
- ***Al final del listado número total de empleados y suma de todos los salarios***

```
DECLARE
      CURSOR c_depart IS SELECT dept_no,dnombre from depart;
      CURSOR c_emple(dep number) IS SELECT emp_no,apellido, salario, dept_no FROM emple
                where dept_no =dep  ORDER BY dept_no, apellido;
      sum_emple NUMBER(4) DEFAULT 0;
      sum_sal NUMBER(9) DEFAULT 0;
      tot_emple NUMBER(4) DEFAULT 0;
      tot_sal NUMBER(10) DEFAULT 0;

BEGIN
      -- recorro cada uno de los departamentos
      for v_reg_dept in c_depart loop
            --reiniciamos los contadores parciales para el departamento
            sum_emple:=0;
            sum_sal:=0;
            dbms_output.put_line('NUMERO DEPARTAMENTO: '|| v_reg_dept.dept_no||'   NOMBRE:'
                        ||v_reg_dept.dnombre);
            dbms_output.put_line('-----------------------   ----------------------');

            --linea de cabecera del listado del departamento
            dbms_output.put_line('EMP_NO NOMBRE          OFICIO         DEPARTAMENTO');

            --para cada departamento que pasamos como param, miramos los empleados que tiene
            for v_reg_emp in c_emple(v_reg_dept.dept_no) loop
                  dbms_output.put_line( v_reg_emp.emp_no||'    '||v_reg_emp.apellido||'   '||
                        v_reg_emp.salario||'   ' ||v_reg_emp.dept_no);
            end loop;

            --con un cursor calculamos los empleados de ese departamento
            select count(*) into sum_emple from emple where dept_no = v_reg_dept.dept_no;
            dbms_output.put_line ('TOTAL DE EMPLEADOS:' ||sum_emple);

            --con un cursor calculamos la suma de salarios de ese departamento
            select sum(salario) into sum_sal from emple where dept_no = v_reg_dept.dept_no;

            dbms_output.put_line ('TOTAL SALARIO:' ||sum_sal);

            dbms_output.put_line (' ');
            tot_emple:= tot_emple + sum_emple;
            tot_sal:= tot_sal + nvl(sum_sal,0); -- si no existen empleados, sum_sal=NULL y no
se puede sumar a menos que se convierta a numero 0

      end loop;
            --visualizamos las sumas totales de empleados y salarios
      dbms_output.put_line ('TOTAL EMPLEADOS:' ||tot_emple||'  TOTAL SALARIO:' ||tot_sal );

END;
```

NOTA: el total de empleados y el total de salarios se podría calcular en vez de utilizar variables acumuladores utilizando un cursor implícito cuando se fueran a imprimir

```
Select count(*) into tot_emp from emple;
Select sum(salario) into  tot_sal from emple;
```

-- otra forma de realizar el ejercicio

```
DECLARE
      CURSOR c1 IS SELECT apellido, salario, dept_no FROM emple
                  ORDER BY dept_no, apellido;
      vr_emp c1%ROWTYPE;
      dep_ant EMPLE.DEPT_NO%TYPE;
      cont_emple NUMBER(4) DEFAULT 0;
      sum_sal NUMBER(9) DEFAULT 0;
      tot_emple NUMBER(4) DEFAULT 0;
      tot_sal NUMBER(10) DEFAULT 0;
BEGIN
      OPEN c1;
      FETCH c1 INTO vr_emp;
      IF c1%FOUND THEN
            dep_ant := vr_emp.dept_no;
      END IF;
      WHILE c1%FOUND LOOP
            /* Comprobación nuevo departamento y resumen */
            IF dep_ant <> vr_emp.dept_no THEN
                  DBMS_OUTPUT.PUT_LINE('*** DEPTO: ' || dep_ant ||' NUM.
                    EMPLEADOS: '||cont_emple ||' SUM. SALARIOS:  '||sum_sal);
                  dep_ant := vr_emp.dept_no;
                  tot_emple := tot_emple + cont_emple;
                  tot_sal:= tot_sal + sum_sal;
                  cont_emple:=0;
                  sum_sal:=0;
            END IF;

            /* Líneas de detalle */
            DBMS_OUTPUT.PUT_LINE(RPAD(vr_emp.apellido,10)|| ' * '
                  ||LPAD(TO_CHAR(vr_emp.salario,'9,999,999'),12));

            /* Incrementar y acumular */
            cont_emple := cont_emple + 1;
            sum_sal:=sum_sal + vr_emp.salario;

            FETCH c1 INTO vr_emp;
      END LOOP;
      CLOSE c1;

      IF cont_emple > 0 THEN
            /* Escribir datos del último departamento */
            DBMS_OUTPUT.PUT_LINE('*** DEPTO: '|| dep_ant ||' NUM EMPLEADOS:'
                  || cont_emple ||' SUM. SALARIOS:  '||sum_sal);
            dep_ant := vr_emp.dept_no;
            tot_emple := tot_emple + cont_emple;
            tot_sal:= tot_sal + sum_sal;
            cont_emple:=0;
            sum_sal:=0;

            /* Escribir totales informe */
          DBMS_OUTPUT.PUT_LINE(' ****** NUMERO TOTAL EMPLEADOS: '||tot_emple
                        ||' TOTAL SALARIOS: '|| tot_sal);
      END IF;
END;
```

7. Escribir un procedimiento que suba el sueldo a todos los empleados que ganen menos que el salario medio de su oficio. La subida será del 50% de la diferencia entre el salario del empleado y la media de su oficio.

```
CREATE OR REPLACE PROCEDURE subida_50porciento
AS
      --creo una lista con todos los oficios y su salario medio ordenado por oficio
      CURSOR c_ofi_sal IS SELECT oficio, AVG(salario) salario FROM emple
                                                GROUP BY oficio ORDER BY OFICIO;

      --creo una lista con todos los empleados cuyo salario es menor que la media en su
      --oficio. El listado estará ordenado por oficio
      CURSOR c_emp_sal IS SELECT emp_no,oficio, salario FROM emple E1
                              WHERE salario <
                                          (SELECT AVG(salario) FROM emple E2
                                            WHERE E2.oficio = E1.oficio)
                              ORDER BY oficio, salario FOR UPDATE OF salario;

      vr_ofi_sal c_ofi_sal%ROWTYPE;
      vr_emp_sal c_emp_sal%ROWTYPE;
      v_incremento emple.salario%TYPE;

BEGIN
      -- El proceso consistirá en ir leyendo de una lista y de la otra,
      -- comparando y actualizando, avanzando según la comparación en una lista u -- en otra.

      COMMIT;
      OPEN c_emp_sal;
      FETCH c_emp_sal INTO vr_emp_sal;
      OPEN c_ofi_sal;
      FETCH c_ofi_sal INTO vr_ofi_sal;
      WHILE c_ofi_sal%FOUND AND c_emp_sal%FOUND LOOP
            /* calcular incremento */
            v_incremento :=    (vr_ofi_sal.salario - vr_emp_sal.salario) / 2;

            /* actualizar */
            UPDATE emple SET salario = salario + v_incremento
                                       WHERE CURRENT OF c_emp_sal;

            /* siguiente empleado */
            FETCH c_emp_sal INTO vr_emp_sal;

            /* comprobar si es otro oficio */
            IF c_ofi_sal%FOUND and  vr_ofi_sal.oficio <> vr_emp_sal.oficio THEN
                                    FETCH c_ofi_sal INTO vr_ofi_sal;
            END IF;
      END LOOP;
      CLOSE c_emp_sal;
      CLOSE c_ofi_sal;
--    COMMIT;
END subida_50porciento;
```

8. Diseñar un procedimiento que simule un listado de liquidación de todos los empleado según la siguientes especificaciones:

****

Liquidación del empleado:..... Departamento:.....
Oficio :.....
Salario :.....
Trienios :.....
Comp. Responsabilidad :.....
Comision :.....

Total :.....

****

Un trienio son 3 años completos desde la fecha de alta hasta la actual, y supone 30€ por trienio.

El complemento de responsabilidad será de 60€ por cada empleado que se encuentre a cargo del empleado.

```
CREATE OR REPLACE PROCEDURE liquidar
AS
      CURSOR c_emp IS   SELECT apellido, emp_no, oficio, salario,NVL(comision,0) comision,
dept_no, fecha_alt
            FROM emple  ORDER BY apellido;

      vr_emp c_emp%ROWTYPE;
      v_trien NUMBER(9) DEFAULT 0;
      v_comp_r NUMBER(9);
      v_total NUMBER(10);

BEGIN
      FOR vr_emp in c_emp LOOP

        /* Calcular trienios */
        v_trien:=TRUNC(TRUNC(MONTHS_BETWEEN(SYSDATE,vr_emp.fecha_alt)/12)/ 3)*30;

            /* Calcular complemento de responsabilidad. Se encierra en un bloque
            pues levantará NO_DATA_FOUND*/
            BEGIN
                  SELECT COUNT(*) INTO v_comp_r  FROM EMPLE
                                    WHERE DIR = vr_emp.emp_no;
                  v_comp_r := v_comp_r *60;
            EXCEPTION   /* se ejecuta esta parte si no encuentra ninguna fila */
                  WHEN NO_DATA_FOUND THEN
                        v_comp_r:=0;
            END;

            /* Calcular el total del empleado */
            v_total := vr_emp.salario + vr_emp. comision +
            v_trien + v_comp_r;

            /* Visualizar datos del empleado */
            DBMS_OUTPUT.PUT_LINE('************************************');
            DBMS_OUTPUT.PUT_LINE(' Liquidacion de : '||vr_emp.apellido||' Dpto: '
                              || vr_emp.dept_no|| '  Oficio:  ' || vr_emp.oficio);
```

```
            DBMS_OUTPUT.PUT_LINE(RPAD('Salario:',16)||
                              LPAD(TO_CHAR(vr_emp.salario, '9,999,999'),12));
            DBMS_OUTPUT.PUT_LINE(RPAD('Trienios:  ',16)||
                              LPAD(TO_CHAR(v_trien,'9,999,999'),12));
            DBMS_OUTPUT.PUT_LINE('Comp. Respons:  ' ||
                              LPAD(TO_CHAR(v_comp_r,'9,999,999'),12));
            DBMS_OUTPUT.PUT_LINE(RPAD('Comision:  ' ,16)||
                              LPAD(TO_CHAR(vr_emp.comision,'9,999,999'),12));
            DBMS_OUTPUT.PUT_LINE('------------------');
            DBMS_OUTPUT.PUT_LINE(RPAD(' Total  : ',16)||
                              LPAD(TO_CHAR(v_total,'9,999,999') ,12));
            DBMS_OUTPUT.PUT_LINE('************************************');
      END LOOP;
EXCEPTION
      WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No se ha encontrado ninguna fila');
END liquidar;
```

9. *Modificar el procedimiento anterior para que la liquidación la realice de un empleado que se introduce por teclado*

```
CREATE OR REPLACE PROCEDURE liquidar(emp_a_buscar number)
AS
      CURSOR c_emp IS   SELECT apellido, emp_no, oficio, salario,NVL(comision,0) comision,
dept_no, fecha_alt      FROM emple where emp_no=emp_a_buscar;

      vr_emp c_emp%ROWTYPE;
      v_trien NUMBER(9) DEFAULT 0;
      v_comp_r NUMBER(9);
      v_total NUMBER(10);

BEGIN
       OPEN C_EMP;

       FETCH c_emp into vr_emp;
       If  c_emp%FOUND then         /*localizado empleado */

       /* Calcular trienios */
         V_trien:=TRUNC(TRUNC(MONTHS_BETWEEN(SYSDATE,vr_emp.fecha_alt)/12)/3)*30;

         /* Calcular complemento de responsabilidad. Se encierra en un bloque
         pues levantará NO_DATA_FOUND*/
            BEGIN
                  SELECT COUNT(*) INTO v_comp_r  FROM EMPLE
                                       WHERE DIR = vr_emp.emp_no;
                  v_comp_r := v_comp_r *60;
            EXCEPTION  /* se ejecuta esta parte si no encuentra ninguna fila*/
                  WHEN NO_DATA_FOUND THEN
                        v_comp_r:=0;
            END;

            /* Calcular el total del empleado */
            v_total := vr_emp.salario + vr_emp. comision +
            v_trien + v_comp_r;

            /* Visualizar datos del empleado */
            DBMS_OUTPUT.PUT_LINE('************************************');
```

```
            DBMS_OUTPUT.PUT_LINE(' Liquidacion de : '||vr_emp.apellido||' Dpto: '
                                || vr_emp.dept_no|| '  Oficio:  ' || vr_emp.oficio);

            DBMS_OUTPUT.PUT_LINE(RPAD('Salario:',16)||
                                LPAD(TO_CHAR(vr_emp.salario, '9,999,999'),12));
            DBMS_OUTPUT.PUT_LINE(RPAD('Trienios:  ',16)||
                                LPAD(TO_CHAR(v_trien,'9,999,999'),12));
            DBMS_OUTPUT.PUT_LINE('Comp. Respons:  ' ||
                                LPAD(TO_CHAR(v_comp_r,'9,999,999'),12));
            DBMS_OUTPUT.PUT_LINE(RPAD('Comision:  ' ,16)||
                                LPAD(TO_CHAR(vr_emp.comision,'9,999,999'),12));
            DBMS_OUTPUT.PUT_LINE('------------------');
            DBMS_OUTPUT.PUT_LINE(RPAD(' Total  : ',16)||
                                LPAD(TO_CHAR(v_total,'9,999,999') ,12));
            DBMS_OUTPUT.PUT_LINE('************************************');
          Else
            Dbms_output.put_line('Ese empleado no existe');
        END if;
        Close c_emp;
END liquidar;
```

10. Crear una tabla T_LIQUIDACION con las columnas, apellido, departamento, oficio, salario, trienios, comp_responsabilidad, comision y total; y modificar el bloque anterior para que en vez de visualizar los datos en pantalla, guarde los datos en una tabla.

```
CREATE TABLE t_liquidacion (
 APELLIDO                VARCHAR2(10),
 DEPARTAMENTO       NUMBER(2),
 OFICIO                  VARCHAR2(10),
 SALARIO                 NUMBER(10),
 TRIENIOS                NUMBER(10),
 COMP_RESPONSABILIDAD    NUMBER(10),
 COMISION                NUMBER(10),
 TOTAL                   NUMBER(10)
);

CREATE OR REPLACE PROCEDURE liquidar2
AS
      CURSOR c_emp IS
      SELECT apellido, emp_no, oficio, salario,NVL(comision,0) comision, dept_no, fecha_alt
            FROM emple  ORDER BY apellido;
      vr_emp c_emp%ROWTYPE;
      v_trien NUMBER(9) DEFAULT 0;
      v_comp_r NUMBER(9);
      v_total NUMBER(10);
BEGIN
      FOR vr_emp in c_emp LOOP
            /* Calcular trienios. */
        v_trien :=TRUNC(TRUNC(MONTHS_BETWEEN(SYSDATE,vr_emp.fecha_alt)/12)/3)*30;
            /* Calcular complemento de responsabilidad. Se
             encierra en un bloque pues levantará NO_DATA_FOUND*/
            BEGIN
                  SELECT COUNT(*) INTO v_comp_r FROM EMPLE WHERE DIR = vr_emp.emp_no;
                  v_comp_r := v_comp_r *60;
            EXCEPTION   /*se ejecutará esta parte si no encuentra ninguna fila*/
                  WHEN NO_DATA_FOUND THEN
                        v_comp_r:=0;
            END;
            /* Calcular el total del empleado */
            v_total :=vr_emp.salario + vr_emp.comision+v_trien + v_comp_r;
            /* Insertar los datos en la tabla T_liquidacion */
            INSERT INTO t_liquidacion (APELLIDO,OFICIO, SALARIO, TRIENIOS,
                                       COMP_RESPONSABILIDAD, COMISION, TOTAL)
              VALUES
                  (vr_emp.apellido, vr_emp.oficio, vr_emp.salario,
                   v_trien, v_comp_r, vr_emp.comision, v_total);
      END LOOP;
END liquidar2;
```