ACTIVIDADES TEMA 6 PARTE IV PL/SQL PAQUETES E INTRODUCCIÓN A LAS EXPECCIONES

1. ***Crear un paquete llamado MIPACK que tenga los siguientes procedimientos y funciones:***
   a. ***Una función que a partir del dept_no devuelva el nombre del departamento (BUSQ_NOM_DEP). Utilizar las excepciones para detectar cuando no existe un departamento***
   b. ***Procedimiento para que a partir de un emp_no, busque y visualice todos los datos del empleado (BUSQ_EMP) . Utilizar las excepciones para detectar cuando no existe un empleado***
   c. ***Una función que a partir de un emp_no calcule su sueldo (CALC_SUELDO). Salario + nomina, recordar que si la nómina es null será necesario convertirla a número NVL***
   d. ***Un procedimiento que visualice todos los empleados de un departamento que se pasa por parámetro (VER_EMPLEADOS_DEP)***

   Recordar que primero hay que definir la cabecera y luego el cuerpo del paquete.

   NOTA: Realizar el proceso en un principio solo para el punto A y cuando funcione ir ampliándolo sucesivamente con el punto B, punto C,...

   Recordar:
   - ***el objetivo de crear un paquete es tener reunido en un solo sitio todos los procedimientos y funciones***
   - ***para usar un procedimiento o una función de un paquete NOMPAQUETE.nom_procd o NOMPAQUETE.nom_func***

```
CREATE OR REPLACE PACKAGE mipack AS

      /*Una función que a partir del dept_no devuelva el nombre del departamento
(BUSQ_NOM_DEP)*/
      function busq_nom_dep(numdept depart.dept_no%type) return varchar2;

      /*Procedimiento para que a partir de un emp_no, busque y visualice todos los datos
del empleado (BUSQ_EMP)*/
      procedure busq_emp(numemp emple.emp_no%type);

            /*Una función que a partir de un emp_no calcule su sueldo (CALC_SUELDO)*/
      function calc_sueldo(numemp emple.emp_no%type) RETURN number;

      /*Un procedimiento que visualice todos los empleados de un departamento
(BUSQ_EMP_DEP)*/
      procedure busq_empleados_dep(numdep emple.dept_no%type);


END mipack;
/
```

```
CREATE OR REPLACE PACKAGE BODY mipack AS

      /*Procedimiento para que a partir de un emp_no, busque y visualice todos los datos
del empleado (BUSQ_EMP)*/
      procedure busq_emp(numemp emple.emp_no%type )
      IS
       v_reg emple%rowtype;
      BEGIN
            select * into v_reg from emple where emp_no=numemp;
            -- utilizo el procedimiento de buscar nombre de departamento para mostrar el
nombre del departamento entre los datos del empleado
            dbms_output.put_line(v_reg.emp_no||' '||v_reg.apellido||' '||v_reg.oficio||' '
                  ||v_reg.dir||' '||v_reg.salario||' '||v_reg.comision||' '
                  ||v_reg.dept_no||'-' ||busq_nom_dep(v_reg.dept_no));
      EXCEPTION
            when NO_DATA_FOUND then
                  dbms_output.put_line('El empleado  NO existe');
      END busq_emp;

      /*Una función que a partir del dept_no devuelva el nombre del departamento
(BUSQ_NOM_DEP)*/
      function busq_nom_dep(numdept depart.dept_no%type) return varchar2
      IS
       nom_dept depart.dnombre%type;
      BEGIN
             select dnombre into nom_dept from depart where dept_no=numdept;
            return (nom_dept);
      EXCEPTION
            when NO_DATA_FOUND then
                  return 'El departamento  NO existe';
      END busq_nom_dep;

      /*Una función que a partir de un emp_no calcule su sueldo (CALC_SUELDO)*/
      function calc_sueldo(numemp emple.emp_no%type) RETURN number
      IS
        sueldo number;
        sal  number;
        com  number;
      BEGIN
            select salario,nvl(comision,0) into sal,com from emple where emp_no=numemp;
            sueldo:=sal+com;
            return sueldo;
      END;

      /*Un procedimiento que visualice todos los empleados de un departamento
(BUSQ_EMP_DEP)*/
      procedure busq_empleados_dep(numdep emple.dept_no%type)
      IS
        cursor cur_emp_dep is select * from emple where dept_no=numdep;
      begin
            for v_reg in cur_emp_dep loop
                  dbms_output.put_line(v_reg.emp_no||' '||v_reg.apellido||'
'||v_reg.oficio||' '||v_reg.dir||' '||v_reg.salario||' '||v_reg.comision||'
'||v_reg.dept_no||' ');
            end loop;
      end;

      END mipack;
/
```

2. Crear un bloque que a partir de un número de empleado introducido por teclado muestre

- ***El departamento al que pertenece***
- ***Los datos del empleado***
- ***Y el sueldo que cobra***

Utilizar las funciones y procedimientos del paquete.

```
accept emp prompt "introduce el numero del empleado "
begin
                    --ver los datos de un empleado
       dbms_output.put_line('los datos del empleado son:');
       mipack.busq_emp(&emp);
       dbms_output.put_line('El sueldo del empleado es '||mipack.calc_sueldo(&emp));
end;
```

3. ***Añadir MIPACK los siguientes procedimiento:***

- ***Función llamada ULT_EMPNO que permita calcular el último emp_no que existe en la tabla EMPLE***

A insertar en la cabecera

```
      /*función que calcula el valor del empno mas alto*/
      function ult_empno return number;
```

A insertar en el body

```
/*función que calcula el valor del empno mas alto*/
      function ult_empno return number is
        ultimo number;
      begin
        select max(emp_no) into ultimo from emple;
        return ultimo;
      end;
```

- ***Procedimiento para insertar un departamento INSERT_DEP. Se pasarán como parámetros el número de departamento, el nombre del departamento y la localidad. Habrá que validar que no exista ya dicho departamento utilizando para ello la función ya existente en MIPACK***

A insertar en la cabecera

```
/*Un procedimiento para insertar un departamento pasando como parámetros NUMERO, NOMBRE y
LOCALIDAD con depuración de errores*/
      procedure INSERT_DEP(numdep depart.dept_no%type, nom depart.dnombre%type,localidad
depart.loc%type);
```

A insertar en el body

```
/*Un procedimiento para insertar un departamento pasando como parámetros NUMERO, NOMBRE y
LOCALIDAD con depuración de errores*/
      procedure INSERT_DEP(numdep depart.dept_no%type, nom depart.dnombre%type,localidad
depart.loc%type)
      IS
        d           integer;
      begin
            --consultamos si existe el departamento y si se produce una execption es que
no existe y podemos insertar
```

```
            select dept_no into d from depart where dept_no = numdep;
            dbms_output.put_line ('El departamento con este código ya existe y es '
           ||busq_nom_dep(numdep));
      EXCEPTION
            when NO_DATA_FOUND then
                  insert into depart values(numdep, nom, localidad);
                  commit;
                  dbms_output.put_line('El departamento se ha insertado con los valores '
                 ||numdep||' '||nom||' ' ||localidad);
      end;
```

- ***Procedimiento para borrar un departamento. Se pasarán los mismos parámetros que en el apartado anterior. Deberá de verificarse que dicho departamento existe en DEPART***

A insertar en la cabecera

```
/*Un procedimiento para iborrar un departamento con depuración de errores*/
      procedure BORRAR_DEP(numdep depart.dept_no%type);
```

A insertar en el body

```
/*Un procedimiento para iborrar un departamento con depuración de errores*/
      procedure BORRAR_DEP(numdep depart.dept_no%type)
      IS
            d           integer;
      begin
        --consultamos si existe el departamento y si se produce una execption es que no
existe
            select dept_no into d from depart where dept_no = numdep;
            delete from depart where dept_no= numdep;
            commit;
            dbms_output.put_line ('El departamento se ha borrado');
      EXCEPTION
            when NO_DATA_FOUND then
                  dbms_output.put_line('El departamento '||numdep||' no existe ');
      end;
```

4. Con el objetivo de tener organizados todos los procedimientos y funciones diseñar dos paquetes PACK_EMPLE y PACK_DEPART y realizar los siguientes ejercicios

- ***Copiar los subprogramas de MIPACK y pegarlos en el correspondiente paquete PACK_EMPLE o PACK_DEPART según corresponda***

- ***Diseñar los siguientes subprogramas para cada uno de los paquetes:***

PACK_EMPLE:

- ***Busq_jefe****: que pasado como parámetros un dept_no y un oficio devuelva el número de empleado que es el jefe.*

- ***Insertar_nuevo_empleado:*** *permite insertar un empleado nuevo. Los datos nombre, oficio, departamento se introducirán por teclado; el numero de empleado será el siguiente al mayor emp_no que existe en la tabla, como fecha de alta la fecha del sistema, la comisión será nula, como jefe (DIR) el dir que tiene asignado los usuarios dicho oficio para ese departamento y como salario el menor de los salarios para dicho oficio.*

- ***Empno_Emple:*** *función que devuelve el emp_no de un empleado a partir de su apellido.*

- ***Modif_comision_empno:*** *procedimiento que permita modificar la comisión de un empleado que se pasa como parámetro.*

- ***Incrementar_salario****: procedimiento que pasado dos parámetros dept_no y valor, permita modificar el salario incrementándolo con dicho valor para todos los empleados del departamento*

GEST_DEPART:

- ***Insertar_nuevo_depart****: permite insertar un departamento nuevo. El procedimiento recibe el nombre y la localidad del nuevo departamento. Creará el nuevo departamento comprobando que el nombre no se duplica y le asignará como número de departamento la decena siguiente al último número de departamento utilizado.*

- ***Borrar_depart****: permite borrar un departamento. El procedimiento recibirá dos números de departamento, de los cuales el primero corresponde al departamento que queremos borrar y el segundo al departamento al que pasarán los empleados del departamento que se va a eliminar. El procedimiento se encargará de realizar los cambios oportunos en los números de departamento de los empleados correspondientes.*

- ***Modif_loc_depart****: modifica la localidad del departamento. El procedimiento recibirá el número del departamento que se modifica y la nueva localidad y realizará el cambio solicitado.*

- ***Visualizar_datos_depart****: visualizará los datos de un departamento cuyo número se pasará en la llamada. Deberá de comprobar que exista el departamento utilizando una función específica para ello. En caso de existir además de los datos relativos al departamento, se visualizará el número de empleados que pertenecen actualmente al departamento.*