Nombre completo del alumno: Daniel Izquierdo Bonilla

EJERCICIOS DE FUNCIONES Y PROCEDIMIENTOS

4) Escribe un procedimiento que modifique la localidad de un dpto. (con dos parámetros: el número del departamento, id, y la nueva localidad, id)

```
-- Daniel Izquierdo Bonilla
-- Ej4
create or replace PROCEDURE PRC (did DEPARTMENTS.DEPARTMENT_ID%type, lid DEPARTMENTS.LOCATION_ID%type)
IS
BEGIN
    Update DEPARTMENTS set LOCATION_ID=lid where DEPARTMENT_ID=did;
END PRC;
```

5) Visualiza todos los procedimientos y funciones del usuario almacenados en la base de datos y su situación (valid o invalid)..

```
select * from USER_OBJECTS where OBJECT_TYPE='PROCEDURE' or OBJECT_TYPE='FUNCTION';
```

Output | SYS.USER_OBJECTS

6 rows | Tx: Manual | DDL | CSV

OBJECT_NAME	SUBOBJECT_NAME	OBJECT_ID	DATA_OBJECT_ID	OBJECT_TYPE	CREATED	LAST_DDL_TIME
SECURE_DML	<null>	73421	<null>	PROCEDURE	2022-03-15 13:18:31	2022-03-15 13:18
ADD_JOB_HISTORY	<null>	73423	<null>	PROCEDURE	2022-03-15 13:18:32	2022-03-15 13:18
LIST_SIZE	<null>	73561	<null>	FUNCTION	2022-04-22 12:50:38	2022-04-22 13:17
FACTORIAL	<null>	73526	<null>	FUNCTION	2022-04-19 13:26:25	2022-04-19 14:09
GET_FIRST	<null>	73562	<null>	FUNCTION	2022-04-22 13:25:19	2022-04-22 13:41
PRC	<null>	73586	<null>	PROCEDURE	2022-04-26 10:52:38	2022-04-26 10:52

6) Hacer una procedimiento que tenga dos parámetros tipo fecha uno de entrada y otro de salida. El de salida se hará la fecha del día siguiente a la entrada. Hacer lo mismo con un solo parámetro IN OUT. Hacer lo mismo con una función con un parámetro IN

```
-- Daniel Izquierdo Bonilla
-- Ej4
create or replace PROCEDURE FECHAS1 (fecha_entrada DATE, fecha_salida OUT DATE)
IS
BEGIN
    fecha_salida := fecha_entrada + 1;
END FECHAS1;
```

```
-- Daniel Izquierdo Bonilla
-- Ej4
create or replace PROCEDURE FECHAS2 (fecha IN OUT DATE)
IS
BEGIN
    fecha := fecha + 1;
END FECHAS2;
```

Nombre completo del alumno: Daniel Izquierdo Bonilla

```
-- Daniel Izquierdo Bonilla
-- Ej4
create or replace FUNCTION FECHAS3 (fecha_entrada IN DATE) RETURN DATE
IS
    fecha DATE;
BEGIN
    fecha := fecha_entrada + 1;
RETURN fecha;
END FECHAS3;
```

7) Crear un procedimiento llamado VEREMPLEADO que muestre el nombre, apellido y localidad del empleado cuyo ID se pasa como parámetro. (Tratar la excepción si no existe ese ID o si el empleado no tiene departamento)

```
-- Daniel Izquierdo Bonilla
-- Ej7
create or replace PROCEDURE VEREMPLEADO(eid EMPLOYEES.EMPLOYEE_ID%type)
    IS
    nombre    EMPLOYEES.FIRST_NAME%TYPE;
    apellido  EMPLOYEES.LAST_NAME%TYPE;
    localidad DEPARTMENTS.LOCATION_ID%TYPE;
    cuenta    NUMBER;
    cuenta2   NUMBER;
    NO_EMPLOYEE_ID EXCEPTION;
    NO_DEPARTMENT EXCEPTION;
BEGIN
    select count(*) into cuenta from EMPLOYEES where EMPLOYEE_ID = eid;
    select count(DEPARTMENT_ID) into cuenta2 from EMPLOYEES where EMPLOYEE_ID = eid;
    if cuenta = 0 then
        raise NO_EMPLOYEE_ID;
    end if;
    if cuenta2 = 0 then
        raise NO_DEPARTMENT;
    end if;
    select FIRST_NAME, LAST_NAME, LOCATION_ID
    into nombre,apellido,localidad
    from EMPLOYEES
             join DEPARTMENTS D on D.DEPARTMENT_ID = EMPLOYEES.DEPARTMENT_ID
    where EMPLOYEES.EMPLOYEE_ID = eid;
    DBMS_OUTPUT.PUT_LINE( A: 'Nombre: ' || nombre || ' Apellido: ' || apellido || ' Localidad: ' || localidad);
EXCEPTION
    WHEN NO_EMPLOYEE_ID THEN
        DBMS_OUTPUT.PUT_LINE( A: 'No existen empleados con la ID:' || eid);
    WHEN NO_DEPARTMENT THEN
        DBMS_OUTPUT.PUT_LINE( A: 'El empleado con ID '||eid||' no tiene departamento.');
END VEREMPLEADO;
```

Nombre completo del alumno: Daniel Izquierdo Bonilla

8) Crear un procedimiento que se llame VERDEPARTAMENTO que liste los empleados (nombre y apellido) de un determinado departamento cuyo ID se pasa como parámetro (Usar un cursor explícito y tratar la excepción se que no exista ese departamento o bien no tenga empleados)

```
-- Daniel Izquierdo Bonilla
-- Ej8
create or replace PROCEDURE VERDEPARTAMENTO(did EMPLOYEES.DEPARTMENT_ID%type)
    IS
    CURSOR c_emp is select FIRST_NAME, LAST_NAME
                    from EMPLOYEES
                    where DEPARTMENT_ID = did;
    nombre   EMPLOYEES.FIRST_NAME%TYPE;
    apellido EMPLOYEES.LAST_NAME%TYPE;
    cuenta   NUMBER;
    cuenta2  NUMBER;
    NO_EMPLOYEES EXCEPTION;
    NO_DEPARTMENT EXCEPTION;
BEGIN
    open c_emp;

    select count(*) into cuenta from EMPLOYEES where DEPARTMENT_ID = did;
    select count(*) into cuenta2 from DEPARTMENTS where DEPARTMENT_ID = did;
    if cuenta2 = 0 then
        raise NO_DEPARTMENT;
    end if;
    if cuenta = 0 then
        raise NO_EMPLOYEES;
    end if;

    LOOP
        FETCH c_emp into nombre,apellido;
        Exit when c_emp%notfound;
        DBMS_OUTPUT.PUT_LINE( A: 'Nombre: ' || nombre || ' Apellido: ' || apellido);
    end loop;

    close c_emp;
EXCEPTION
    WHEN NO_EMPLOYEES THEN
        DBMS_OUTPUT.PUT_LINE( A: 'No existen empleados en el dartamento con ID:' || did);
    WHEN NO_DEPARTMENT THEN
        DBMS_OUTPUT.PUT_LINE( A: 'El departamento con ID:' || did || ' no existe');
END VERDEPARTAMENTO;
```

9) Crear el procedimiento MODIFICAR_SALARIO_EMPLEADO que modifique el salario de un empleado pasándole su ID y el nuevo salario. El procedimiento comprobará que la variación de precio no supere el 20% (arriba o abajo) y lanzará una excepción en caso contrario y no efectuará la modificación. (También lanza excepción si no existe el ID)

```
-- Daniel Izquierdo Bonilla
-- Ej9
create or replace PROCEDURE MODIFICAR_SALARIO_EMPLEADO(eid EMPLOYEES.EMPLOYEE_ID%type,salario EMPLOYEES.SALARY%type)
    IS
    maximo   NUMBER;
    minimo  NUMBER;
    auxSalario NUMBER;
    SALARY_OUT_OF_RANGE EXCEPTION;
BEGIN
    select SALARY into auxSalario from EMPLOYEES where EMPLOYEE_ID=eid;
    maximo := auxSalario * 1.2;
    minimo := auxSalario / 1.2;
    if salario > maximo or salario < minimo then
        raise SALARY_OUT_OF_RANGE;
    end if;
    Update EMPLOYEES set SALARY=salario where EMPLOYEE_ID=eid;
    DBMS_OUTPUT.PUT_LINE( A: 'Salario modificado a: '||salario);
EXCEPTION
    WHEN SALARY_OUT_OF_RANGE THEN
        DBMS_OUTPUT.PUT_LINE( A: 'El salario introducido esta fuera del rango valido');
END MODIFICAR_SALARIO_EMPLEADO;
```

Nombre completo del alumno: Daniel Izquierdo Bonilla

10) Crea una función CALIFICACION que devuelva una nota final en número y letra (“Suspenso” “Aprobado”, “Notable” o “Sobresaliente”) en función de dos notas númericas de tal manera que la primera nota cuente el 60% y la segunda un 40% y que si cualquiera de las dos se suspende (<4,5) la nota final no puede ser mayor que 4.

```
-- Daniel Izquierdo Bonilla
-- Ej10
create or replace FUNCTION CALIFICACION(nota1 NUMBER, nota2 NUMBER) RETURN VARCHAR2
    IS
    nota_final NUMBER;
    nota       VARCHAR2(15);
BEGIN
    if nota1 < 4.5 or nota2 < 4.5 then
        nota_final := nota1 * 0.6 + nota2 * 0.4;
        if nota_final > 4 then
            nota_final := 4;
        else
            nota_final := nota1 * 0.6 + nota2 * 0.4;
        end if;
    end if;

    if nota_final >= 5 and nota_final <= 6 then
        nota := 'Aprobado';
    elsif nota_final >= 7 and nota_final <= 8 then
        nota := 'Notable';
    elsif nota_final >= 9 and nota_final <= 10 then
        nota := 'Sobresaliente';
    else
        nota := 'Suspenso';
    end if;
    RETURN nota_final || ' ' || nota;
END CALIFICACION;
```