Práctica de subprogramas, cursores y excepciones

Realizar los siguientes ejercicios utilizando las tablas vistas en clase EMPLE y DEPART.

Ejercicio 1

Diseñar una función llamada **DevolverCodDept** que reciba el nombre de un departamento y devuelva su código.

Una posible llamada **DevolverCodDept('CONTABILIDAD')** y la función devolverá el código del departamento que se llame de esa manera, en el ejemplo, devolverá 10.

Se tratarán dos posibles excepciones: **NO_DATA_FOUND**, por si el nombre no existe para ningún departamento y **TOO_MANY_ROWS**, si hubiera más de un departamento con el mismo nombre.

Ejercicio 2

Realiza un procedimiento llamado **HallarNumEmp** que recibiendo un nombre de departamento, muestre en pantalla el número de empleados de dicho departamento. Hay que utilizar la función del ejercicio 1.

Una posible llamada: **HallarNumEmp('CONTABILIDAD'),**y se mostrará en pantalla el número de empleados que tenga ese departamento.

Si el departamento no tiene empleados deberá mostrar un mensaje informando de ello.

Si el departamento no existe se tratará la excepción correspondiente al llamar a la función del ejercicio 1 donde ya estaba tratada, es decir, se propagará hasta esta función.

Ejercicio 3

Realiza una función llamada **CalcularCosteSalarial** que reciba un nombre de departamento y devuelva la suma de los salarios y comisiones de los empleados de dicho departamento. Tened en cuenta que las comisiones pueden ser nulas.

Se llamará a la función del ejercicio 1.

Una posible llamada: **CalcularCosteSalarial('CONTABILIDAD')** y devolverá la suma de salarios y comisiones de ese departamento.

Además de la excepción que se va propagando a la función del ejercicio 1, crea una de usuario que se lanzará con RAISE.

Ejercicio 4

Realiza un procedimiento **MostrarCostesSalariales** que muestre los nombres de todos los departamentos y el coste salarial de cada uno de ellos. Será necesario utilizar un cursor para recorrer todos los departamentos y para cada uno de ellos utilizad la función creada en el ejercicio 3.

Este procedimiento no tendrá ningún argumento ya que debe ejecutarse para todos los registros de la tabla.

Ejercicio 5

Realiza un procedimiento **MostrarAbreviaturas** que muestre las tres primeras letras del nombre de cada empleado. Al tener que recorrer todos los nombres de la tabla, será necesario un cursor.

El procedimiento no tendrá ningún argumento.

Ejercicio 6

Diseñar una función **BuscarMasAntiguo** que reciba un código de departamento y devuelva el nombre del empleado más antiguo de dicho departamento. Será necesario tratar dos excepciones que se pueden producir, NO_DATA_FOUND, que el departamento esté vacio, o TOO_MANY_ROWS, que existan varios empleados con la misma fecha de antigüedad.

Una posible llamada: **BuscarMasAntiguo(10)**

Ejercicio 7

Realiza un procedimiento **MostrarMasAntiguos** que recorriendo la tabla de departamentos muestre el nombre de cada departamento y el nombre del empleado más antiguo de ese departamento. Trata las excepciones que consideres necesarias. Será necesario un cursor que recorra todas las filas de la tabla DEPART.

Ejercicio 8

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. Si un departamento cuyo nombre se pasa como argumento no tuviera jefes, se mostraría un mensaje.

Una posible llamada **MostrarJefes('CONTABILIDAD')** y mostrará los nombres de los empleados del departamento 10 que son jefes de otros empleados.

Si en dicho departamento no existiera ningún empleado jefe, se mostraría **'en el departamento CONTABILIDAD no hay ningún empleado jefe'.**

Ejercicio 9

Realiza un procedimiento **MostrarMejoresVendedores** que muestre los nombres de los tres vendedores con más comisiones de toda la empresa. Si en la empresa hubiera menos de tres vendedores con comisión, lanzar una excepción en primer lugar con RAISE_APPLICATION_ERROR y probar también una excepción de usuario declarada y lanzada con RAISE.

Ejercicio 10

Diseñar una función **DevolverCadenaAlReves** que reciba una cadena y la devuelva al revés. Una posible llamada: **DevolverCadenaAlReves('DADILIBATNOC')** devolverá CONTABILIDAD.

Ejercicio 11

Realiza un procedimiento **MostrarsodaelpmE** que reciba el nombre de un departamento al revés y muestre los nombres de los empleados de ese departamento. Utilizad la función del ejercicio anterior para darle la vuelta al nombre y crear una excepción que contemple el caso de que el departamento no exista.
Una posible llamada **MostrarsodaelpmE('DADILIBATNOC'),** si el departamento existe, visualizará los nombres de sus empleados y si no existe se lanzará una excepción que lo indique.

Ejercicio 12

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

Una posible llamada **RecortarSueldos('A')**, es decir a los empleados cuyo nombre empiece por 'A' les recortará su sueldo un 20%.

Se visualizará un mensaje indicando si no se ha actualizado el sueldo a ningún empleado o a cuántos, utilizad para ello el atributo SQL%NOTFOUND del cursor implícito que se ha utilizado.

Ejercicio 13

Realizar un procedimiento llamado **BorrarDosMasNuevos** que reciba un código de departamento y borre los dos empleados más nuevos de ese departamento.

Una posible llamada **BorrarDosMasNuevos(10)** y borrará teniendo en cuenta su fecha de contratación los dos más nuevos, es decir, los dos últimos en llegar.

Ejercicio 14

Realiza un procedimiento **BorrarBecarios** que borre a los dos empleados más nuevos de cada departamento. Para ello se necesitará un cursor que vaya recorriendo todos los departamentos y en cada uno de ellos, llamará al procedimiento anterior. Trata las excepciones que consideres necesarias.