AT06-03- Procedimientos, funciones y disparadores

Para llevar a cabo esta actividad emplearemos la base de datos “Ventas” creada en la unidad anterior.
Crea en dicha base de datos los siguientes procedimientos, funciones y disparadores:

FUNCIONES

1. Crea una función que devuelva el siguiente número de departamento que se deberá asignar cuando se registre un nuevo departamento (que funcione como un auto_increment para el dep_no), es decir, cuando se llama a la función, devolverá el máximo número de departamento existente en la tabla más 1.
2. Crea una función que devuelva el nombre del jefe de un empleado. La función aceptará como parámetro un número de empleado y buscará el nombre del jefe de ese empleado y devolverá ese valor.
3. Modifica la función anterior para que en caso de que no encuentre un jefe válido devuelva el valor “Jefe no encontrado”.

PROCEDIMIENTOS

4. Desarrollar un procedimiento que visualice el apellido, el nombre del jefe y la fecha de alta de todos los empleados ordenados por apellido. Para mostrar el nombre del jefe deberá utilizar la función creada en el ejercicio 3.
5. 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 mostrará el número de empleados incluidos en el listado.
6. Escribir un programa que visualice el apellido y el salario de los cinco empleados que tienen el salario más alto.
7. Codifica un procedimiento que muestre el nombre de cada departamento y el número de empleados que tiene.
8. Modifica el procedimiento anterior para que tras mostrar el nombre y número de empleados de cada departamento muestre un listado de los nombres de todos los empleados de ese departamento.
9. Crea un procedimiento que acepte un número de departamento como argumento y visualice todos los datos de los empleados que trabajan en ese departamento.
10. Crea un procedimiento que acepte un número de departamento y devuelva mediante dos parámetros de tipo OUT su nombre y localidad.
11. Crea un procedimiento que reciba como parámetro una localidad y cree automáticamente 1 departamento en esa localidad con los siguientes valores:
    a. número de departamento: será el valor devuelto por la función creada en el ejercicio 1
    b. nombre: el nombre del departamento tendrá el formato “departamento X“, siendo X el valor asignado como número de departamento.
    c. Localidad: se asignará el valor pasado como parámetro.
12. Crea un procedimiento al que se pase como parámetro un número de departamento y cree 3 empleados en ese departamento con la siguiente información:
    a. EMP_NO: se debe cubrir por medio de una función similar a la empleada para el código del departamento.
    b. APELLIDO: se asignará la cadena “desconocido”.
    c. OFICIO: se asignará a los tres empleados el oficio que menos empleados de la empresa tengan.
    d. DIRECTOR: se les asignará como director al jefe de la empresa (al empleado que no tenga ningún jefe).
    e. FECHA_ALTA: fecha actual.
    f. SALARIO: se les asignará el salario mínimo de la empresa (el menor salario que haya en la empresa).
    g. COMISION: nula
    h. DEP_NO: se asignará el número de departamento que se pase como parámetro.
    i. TELEFONO: nulo
13. Modifica el procedimiento anterior para que si le pasamos un número de departamento, en lugar de abortar al tratar de insertar un empleado en un departamento no existente, devuelva el error ‘Departamento no existente’ y finalice correctamente (sin abortar).
14. Modifica el procedimiento anterior para que no falle en caso de que no exista ningún oficio en la BD (al buscar el menos repetido no devolvería nada y trataría de insertar un oficio nulo). En este caso debería devolver el error ‘No es posible insertar un empleado sin oficio’ y finalizar.

15. Crea un procedimiento que busque los departamentos que no tienen ningún empleado, y en cada uno de ellos inserte 3 empleados empleando el procedimiento anterior.

TRIGGERS

16. Crea un trigger en la tabla empleados que asigne automáticamente la fecha actual al empleado en la fecha de contratación si la fecha de contratación que se ha indicado es nula.
17. Modifica el trigger anterior para que impida insertar empleados con número de empleado mayor que 1000.
18. ¿Qué ocurriría si intentas en lugar de modificar el trigger anterior creas un nuevo trigger before insert en empleados?
19. Añade un campo "importe" en la tabla PEDIDO de tipo numeric(9,2). Crea un disparador que se ejecute cada vez que se inserte o modifique una fila en la tabla PEDIDO que calcule el valor del campo importe de la siguiente manera: importe=precio del producto por cantidad. El precio del producto tendrás que buscarlo en la tabla PRODUTO.
20. Crea una tabla empleados_historico que guarde los registros de los empleados borrados; junto con los atributos del empleado cada registro guardará el nombre del usuario que hace el borrado y la fecha de borrado. Crea un trigger que se ejecute antes de hacer el borrado de un empleado y guarde en la tabla empleados_historico cada empleado que es borrado.
21. Crea un TRIGGER BEFORE INSERT que cuando se inserta un empleado sin salario le asigne el salario más bajo de cualquier empleado de la empresa (calculado sin tener en cuenta los salarios NULL o 0).
22. Elabora un mecanismo que impida que se inserte un registro en la tabla de pedidos cuando el número de unidades pedidas de un producto supera el stock disponible de ese producto. Además mostrará un mensaje indicando que no se puede realizar el pedido por falta de stock. En caso de que se pueda realizar el pedido restará al stock del producto la cantidad comprada en el pedido.
23. Indica cómo se puede conseguir lo siguiente: Deseamos evitar que se borre un empleado mientras sea jefe de otros empleados
24. Indica cómo se puede conseguir lo siguiente: Deseamos evitar que un empleado pueda ser jefe de más de 3 empleados (no se permitirá introducir o modificar a un empleado si su nuevo jefe ya tiene 3 subordinados, además mostrará un mensaje de error apropiado)
25. Indica cómo se puede conseguir lo siguiente: Deseamos evitar que por error se introduzca el mismo producto varias veces en la tabla de productos (aunque sea de distintos fabricantes).