EJERCICIOS DE ACCESO A BBDD

1.- Crea la base de datos Instituto, cuyas tablas y valores se encuentran en el fichero instituto.sql. Una vez creada la base de datos haz un programa en Java que cree una nueva tabla llamada **NotasFinales** que tendrá la siguiente estructura:

NotasFinales(Mat, Cod, NotaMedia);

Y cuyos valores se sacarán de la tabla Notas. En esta tabla se muestra para cada asignatura y alumno, la nota media de las 3 evaluaciones.

Una vez creada la tabla, el programa muestra un listado de la tabla NotasFinales.

Por último, se imprimirá un listado de todos los alumnos con el siguiente formato (a partir de las tablas que tenemos):

Nombre Alumno Nombre Asignatura Nota 1 Nota 2 Nota 3 NotaMedia

2.- Se tiene la base de datos América, compuesta por las tablas Personas y Países. Haz un programa en Java que cree la tabla PersonasPaises que tendrá los siguientes atributos:

IdPersona, Nombre, Apellido, Edad, NombrePais y Tamaño.

La información que va a almacenar es la sacada de las otras dos tablas. Tras crear dicha tabla, actualízala sumando 1 a la edad de las personas de Costa Rica. Finalmente saca un listado con toda la información de la nueva tabla.

3.- Realiza un programa en Java para trabajar con la base de datos Empresa (empleados y departamentos), que haga lo siguiente:

a. Conexión a la base de datos (carga del driver y establecimiento de conexión).

b. Inserción de un departamento. Escribe un método llamado **insertarDepto** que recibirá tres argumentos (número, nombre y localidad del departamento).

c. Inserción de un departamento. El mismo nombre de método que b, pero recibiendo un solo argumento: un objeto de la clase Departamento. Será necesario, por tanto, crear una clase Departamento, con sus atributos y métodos getter y setter.

d. Método (**listarDepartamentos**) que **devuelva un ArrayList de objetos Departamento** a partir de una consulta que devuelva todos los departamentos de la tabla departamentos.

e. Método (**borrarDepartamento**) que reciba un número de departamento y lo dé de baja.

f. Método (**actualizarDepartamento**) que reciba un número de departamento y una localidad y actualice su localidad, con ese nuevo valor.

g. Método (**actualizarDepartamento**) que reciba un objeto departamento y actualice el departamento con el número de departamento indicado a los valores dados en el objeto.

h. Método (**devolverDepartamento**) que reciba un número de departamento y devuelva un objeto con sus datos

i. Método (**subirSalario**) que reciba una cantidad y un número de departamento e incremente el sueldo de todos los empleados de ese departamento en esa cantidad.

(***Ampliación***:)

j. Método que imprima el gestor de base de datos empleado, el driver utilizado y el usuario conectado.

k. Método que imprima del esquema actual todas las tablas y vistas que contiene, indicando además del nombre, si se trata de una tabla o una vista.

l. Método que reciba una consulta (p.ej. SELECT * FROM departamentos) e imprima el número de columnas recuperadas, y por cada columna el nombre, tipo, tamaño y si admite o no nulos.

4.- Hacer una aplicación que realice las siguientes operaciones:

- Serializar todos los objetos de tipo Producto que tenemos en la base de datos Tienda para escribirlos en un fichero llamado "**productos.dat**".
- Se recuperarán todos los productos del fichero "productos.dat" y se asignarán a un ArrayList de productos.
- Se le presentarán al usuario todos los productos (desde el ArrayList) que sean susceptibles de ser vendidos, es decir, aquellos cuya cantidad sea > 0. El usuario tendrá que elegir uno y la cantidad que desea comprar, actualizándose en el ArrayList la cantidad que queda. Se pregunta al usuario si quiere seguir comprando, y no se para el proceso hasta que decida dejar de comprar.
- Cuando se venda un producto, se irá guardando en un fichero de texto ("**vendidos.txt"**) con los datos del producto vendido, así como la cantidad y el importe total (código – nombre – cantidad vendida – importe total).
- El fichero "**vendidos.txt"** con los datos de productos vendidos se imprimirá cuando acabe el proceso de venta.
- Actualiza a continuación la **base de datos de acuerdo con lo vendido** (la información se saca del ArrayList, que es donde se ha ido descontando lo que se ha vendido).
- Actualiza también el fichero de objetos con el contenido de la base de datos
- Y, por último, visualiza el contenido de la base de datos y el fichero para comprobar que son iguales.

Organiza el código de la manera que quieras para que la estructura tenga sentido según las distintas funcionalidades.

Clase Producto

• ***Atributos***:

int codigo;
String nombre;
int precio;
int cantidad;

Todos ellos de tipo privados

• ***Métodos obligatorios***

- Método get y set para cada uno de los atributos.
- Método para añadir más cantidad de un producto determinado.
- Método para disminuir la cantidad de un producto determinado cuando se haga una venta. Si no hubiese suficientes unidades se lanzará una excepción propia que indicará el problema
- Método que escriba toda la información de un producto en concreto.