

COMO CONECTARSE A UNA BASE DE DATOS MEDIANTE NETBEANS Y SQLITE

IMPORTANTE:

Este manual está realizado en base Ubuntu.

Para poder conectarnos a una base de datos y realizar cambios en ella debemos saber que necesitamos una base de datos con extensión (.db) , un archivo (.sql) y un programa de programación para poder conectarnos a ella y poder usarla.

Lo principal que debemos tener instalado son:

-NetBeans JDK 8.2 actualizado

Link de descarga:

<http://www.oracle.com/technetwork/es/java/javase/downloads/jdk-netbeans-jsp-3413139-esa.html>

Con este programa podremos realizar el código para poder conectarnos a nuestra base de datos, modificarla, etc.

-Sqlite3 actualizado

Comandos de descarga:

```
$ sudo apt-get update
```

```
$ sudo apt-get install sqlite3
```

Con este otro realizaremos la base de datos, la podemos realizar mediante entorno gráfico o mediante terminal.

Este manual está realizado por terminal.

CONFIGURACIÓN DE LOS PROGRAMAS

PRIMER PASO DESCARGA DE SQLITE.JAR:

Una vez tengamos los 2 programas anteriores mencionados, instalados y actualizados, vamos a descargar el archivo (sqlite-jdbc 3.16) en formato (.jar) para añadirlo a la librería de nuestro proyecto y que no nos de problemas al compilar el código.

Enlace de decarga archivo sqlite:

<https://mvnrepository.com/artifact/org.xerial/sqlite-jdbc/3.16.1>



SEGUNDO PASO CREACIÓN DEL PROYECTO:

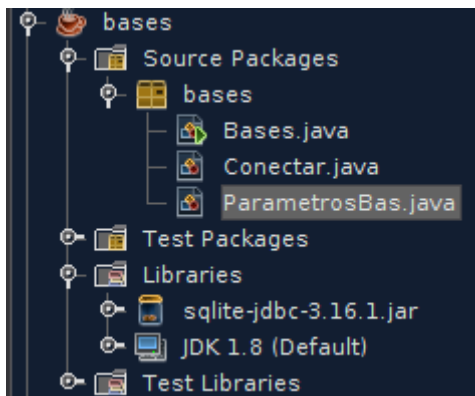
Crear un nuevo proyecto en NetBeans 8.2. Este proyecto será sencillo sin necesidad de hacer proyectos especiales para realizar bases de datos:

File – New Project – Java Application - “nombre_del_proyecto”-Finish

Crearemos una clase nueva que se va a llamar ParametrosBas que será la que usaremos para introducir los constructores, get/set y toString. También crearemos la clase Conectar que será la encargada de llevar los métodos necesarios para poder conectarnos y utilizar la base de datos. Cuando tengamos el proyecto creado con sus main y clases creadas, en la parte izquierda de la pantalla donde tenemos todos los proyectos, abrimos la pestaña Libraries click derecho en el y ADD Jar/folder...

Buscamos el archivo sqlite.jar que descargamos anteriormente y lo seleccionamos.

Ya tenemos importada en la librería el archivo sqlite.jar, ya podemos trabajar sin que nos de problemas.



TERCER PASO CONFIGURACIÓN DE CLASE PARÁMETROS:

Ya tenemos el proyecto a punto para introducir métodos en las clases.

En primer lugar iremos a la clase ParametrosBas.java y declararemos los nombres que necesitamos en la base de datos:

```
package bases;

/**
 *
 * @author ped90
 */
public class ParametrosBas {
    String nombre, apellidos,dni,edad;
}
```

Ahora crearemos (constructor sin parámetros y con parámetros, getters y setters, to String)

```
public ParametrosBas() {
}

public ParametrosBas(String nombre, String apellidos, String dni, String edad) {
    this.nombre = nombre;
    this.apellidos = apellidos;
    this.dni = dni;
    this.edad = edad;
}

public String getNombre() {
    return nombre;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

public String getApellidos() {
    return apellidos;
}

public void setApellidos(String apellidos) {
    this.apellidos = apellidos;
}

public String getDni() {
    return dni;
}

public void setDni(String dni) {
    this.dni = dni;
}

public String getEdad() {
    return edad;
}

public void setEdad(String edad) {
    this.edad = edad;
}

@Override
public String toString() {
    return "ParametrosBas{" + "nombre=" + nombre + ", apellidos=" + apellidos + ", dni=" + dni + ", edad=" + edad + "}";
}
```

CUARTO PASO CONFIGURACIÓN CLASE CONECTAR:

Ahora abrimos la clase Conectar.java, como queremos crear una tabla con parámetros y la crearemos y rellenaremos a partir de métodos en NetBeans (sin necesidad de programas externos), lo más fácil y rápido es crear un Array. Para ello pondremos:

```
public class Conectar {

    ParametrosBas jug = new ParametrosBas();
    ArrayList<ParametrosBas> lista = new ArrayList<>();
```

Pondremos ya la ruta donde vamos a guardar el fichero.db:

```
private Connection conectar;
    ResultSet result;
    private final String ruta = "/home/ped90/Descargas/lista.db";
```

Otro método que coja los drivers de sqlite JDBC:

```
public void driversBase(){
    try {
        Class.forName("org.sqlite.JDBC");
    } catch (ClassNotFoundException ex) {
        System.out.println("DRIVERS NOT FOUND");
    }
}
```

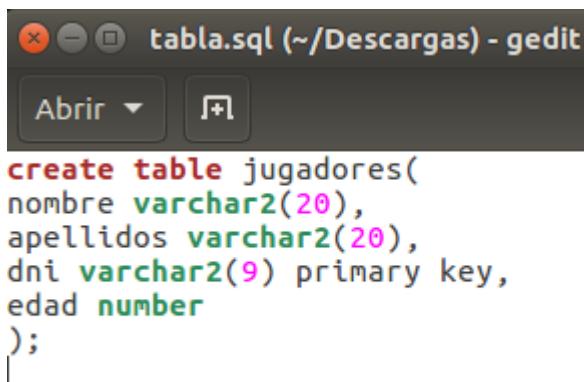
Crearemos el método para introducir los parámetros en el Array:

```
public void cargaArray(){
    lista.add(new ParametrosBas(
        JOptionPane.showInputDialog("Inserta el Nombre"),
        JOptionPane.showInputDialog("Inserta el Apellido "),
        JOptionPane.showInputDialog("Inserta el DNI"),
        JOptionPane.showInputDialog("Inserta la Edad")));
}
```

Con estos métodos podemos ya conectarnos a la base de datos y insertar los parámetros, pero, no se van a visualizar en el fichero. Para ello tenemos que crear un fichero (.sql) donde esta el fichero (.db) que será el encargado de darle la tabla con los parámetros necesarios creada al fichero (.db)

QUINTO PASO CREACIÓN DE LA TABLA:

Como se dijo en el paso anterior, creamos el fichero (.sql) en donde tenemos el fichero(.db), abrimos el archivo (.sql) y introducimos lo siguiente:



```
create table jugadores(
nombre varchar2(20),
apellidos varchar2(20),
dni varchar2(9) primary key,
edad number
);
```

creamos la tabla jugadores:

nombre (aceptará letras y números con un valor máximo de 20 caracteres)

apellidos (aceptará letras y números con un valor máximo de 20 caracteres)

dni (aceptará letras y números con un valor máximo de 9 caracteres y es obligatorio rellenarlo)

edad (valor infinito y solo admite números).

Una vez creada la tabla en el archivo (.sql) abrimos el terminal (comenzaremos a utilizar el sqlite3 por terminal), estos pasos son complicados así que mucha atención a los comandos mostrados y poner bien las rutas ya que no podrán corresponder con las mías mostradas.

Paso 1-

Poner la ruta donde tenemos el archivo .db

Y a continuación abrirlo con el sqlite

```
ped90@ped90-GE62VR-7RF:~$ cd Descargas/  
ped90@ped90-GE62VR-7RF:~/Descargas$ sqlite3 lista.db
```

Paso 2-

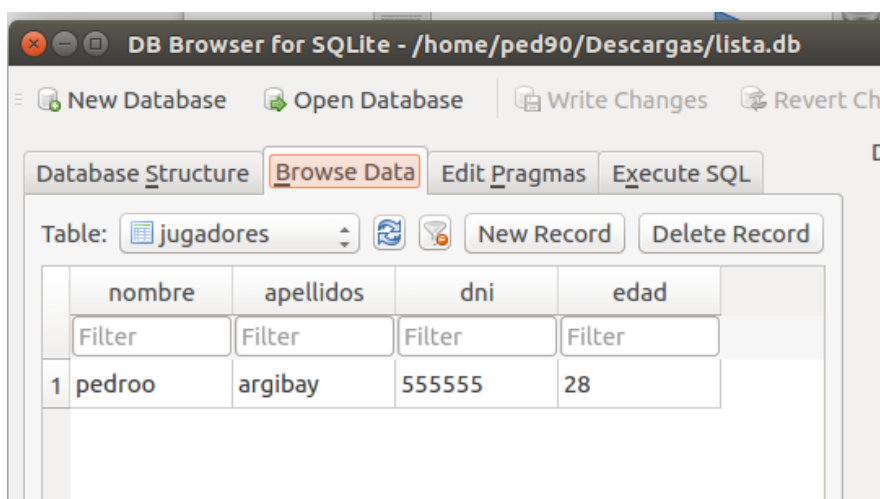
Abrir el archivo .sql para indicarle al sqlite3 que este será el que ponga la tabla

```
sqlite> .read tabla.sql
```

Ahora el archivo lista.db pasará de este aspecto a este otro:



Esto indica que el archivo lista.db esta preparada y lista para recibir datos como base de datos. Si quisiéramos abrir el documento nos saltaría un error de que nos haría falta un programa para abrirlo, y si lo abrimos con uno predeterminado:





El archivo como se ve está encriptado.

SEXTO PASO INTRODUCIR DATOS EN LA TABLA:

Para insertar los datos en nuestra tabla debemos ir al NetBeans y a la clase Conectar.java, y hacer este método:

```
public void insertarJugadores(){
    try {
        PreparedStatement ps = conectar.prepareStatement("Insert into Jugadores(Nombre, Apellidos, DNI, Edad) values(?,?,?,?)");
        for(int i=0;i<lista.size();i++){
            ps.setString(1,lista.get(i).getNombre());
            ps.setString(2,lista.get(i).getApellidos());
            ps.setString(3,lista.get(i).getDni());
            ps.setString(4,lista.get(i).getEdad());
            ps.execute();
        }
    } catch (SQLException ex) {
        System.out.println("Fallo del sistema al insertar :"+ex.getMessage());
    }
}
```

Este método lo que va a realizar es que cuando cargues datos en el Array transforma esos datos en una consulta de base de datos y la introduce en la tabla. Si no se le añade este método en la main el programa no fallará, pero no te va a introducir los datos en la tabla.

Para comprobar que los datos fueron añadidos con éxito vamos a la clase Bases.java (main):

```
Conectar obx = new Conectar();

obx.conectarBase();
obx.driversBase();
obx.cargaArray();
obx.insertarJugadores();
```

Creemos un objeto de tipo Conectar (será la clase Conectar.java para llamar a los métodos)

- 1-Llamamos primero a conectarBase → buscará el archivo .db
- 2-Llamamos a drivers → buscará los drivers y arrancará
- 3-Llamamos a cargaArray → para introducir nosotros los datos
- 4-Llamamos a insertarJugadores → este coge los datos del array y los introduce en la tabla

Ejecutamos el programa, por lógica el primer mensaje en consola es este:

```
run:
La conexión se ha realizado correctamente a la ruta:/home/ped90/Descargas/lista.db
```

El programa ha encontrado satisfactoriamente el archivo .db

De seguido nos pedirá el Array los datos de “jugadores” en mi caso.

Al acabar de introducirlos este acabará con el mensaje:

```
run:
La conexión se ha realizado correctamente a la ruta:/home/ped90/Descargas/lista.db
BUILD SUCCESSFUL (total time: 9 seconds)
```

Ya tenemos los datos en nuestra tabla insertados.

SÉPTIMO PASO COMPROBACIÓN DE DATOS EN LA TABLA:

En el paso anterior hemos introducido los datos en la tabla, para comprobar que los datos están ahí debemos ir al terminal de nuevo y poner:

```
sqlite> select * from jugadores;
nombre      apellidos   dni          edad
-----
afdgdsfg    wtwertyrt   1111         23
pedro       argibay     555555       26
sqlite>
```

Como se puede ver mis datos se aprecian, pero quiero que se reflejen en NetBeans en la consola del mismo, para ello nos vamos a la clase Conectar.java y haremos un nuevo método que será este:

```
public void visualizarJugadores(){
    try {
        PreparedStatement ver = conectar.prepareStatement("Select * from Jugadores");
        result = ver.executeQuery();
        while(result.next()){
            System.out.println("Nombre "+" :"+result.getString("Nombre"));
            System.out.println("Apellidos "+" :"+result.getString("Apellidos"));
            System.out.println("dni"+" :"+result.getString("DNI"));
            System.out.println("edad"+" :"+result.getInt("Edad"));
        }
    } catch (SQLException ex) {
        System.out.println("Error al leer la BD: "+ex.getMessage());
    }
}
```

Este método lo que realiza es mediante el comando “select * from Jugadores” te devuelve los datos que está guardados en la base de datos.

Para ejecutar este método en la main Bases.java tendremos que ponerla así:

```
Conectar obx = new Conectar();

obx.conectarBase();
obx.driversBase();
obx.cargaArray();
obx.insertarJugadores();
obx.visualizarJugadores();
```

Creamos un objeto de tipo Conectar (será la clase Conectar.java para llamar a los métodos)

1-Llamamos primero a conectarBase → buscará el archivo .db

- 2-Llamamos a drivers → buscará los drivers y arrancará
- 3-Llamamos a cargaArray → para introducir nosotros los datos
- 4-Llamamos a insertarjugadores → este coge los datos del array y los introduce en la tabla
- 5-Llamamos a visualizarJugadores → este coge los datos de la tabla y los muestra

Visualización de los datos por NetBeans ventana comandos:

```
run:
La conexión se ha realizado correctamente a la ruta:/home/ped90/Descargas/lista.db
Nombre :afdgsfg
Apellidos :wtwertyrt
dni:1111
edad:23
Nombre :null
Apellidos :null
dni :null
edad :0
Nombre :pedro
Apellidos :argibay
dni :555555
edad :26
Nombre :pepe
Apellidos :perez
dni :343434
edad :23
BUILD SUCCESSFUL (total time: 11 seconds)
```

Se puede ver que añadí un dato mas (pepe, perez, 343434, 23) ese dato será el que borremos con el siguiente apartado.

OCTAVO PASO BORRAR DATOS DE LA TABLA:

Como vimos en el anterior apartado, visualizamos los datos que introducimos en la tabla, ahora borraremos el dato ese extra que introducí. Para ello volvemos a la clase Conectar.java y creamos el método siguiente:

```
public void borrarjugador(){
    Scanner teclado = new Scanner(System.in);
    System.out.println("Inserte el número del dni para borrar la fila");
    Integer reg=teclado.nextInt();
    try{
        Statement st = conectar.createStatement();
        st.execute("delete from Jugadores where dni="+reg.toString());
        System.out.println("Fila borrada con éxito");
    }catch(SQLException ex){
        System.out.println("Error al borrar la fila, compruebe que ha introducido bien el DNI: "+ex.getMessage());
    }
}
```

Este método lo que hace es que cuando le introduces por teclado el dni borra inmediatamente la fila que contenga ese dato.

Comprobamos que ha sido así, eliminaremos la fila que creamos de (pepe, perez, 343434, 23)

En la clase Bases.java (main) la pondremos de este modo:

```

Conectar obx = new Conectar();

obx.conectarBase();
obx.driversBase();
obx.visualizarJugadores();
obx.borrarJugador();

```

Creamos un objeto de tipo Conectar (será la clase Conectar.java para llamar a los métodos)

1-Llamamos primero a conectarBase → buscará el archivo .db

2-Llamamos a drivers → buscará los drivers y arrancará

3-Llamamos a visualizarJugadores → este coge los datos de la tabla y los muestra

4-Llamamos a borrarJugador → coge el dni que le introduzcamos y borra la fila entera

```

run:
La conexión se ha realizado correctamente a la ruta:/home/ped90/Descargas/lista.db
Nombre :afdgdsfg
Apellidos :wtwertyrt
dni :1111
edad :23
Nombre :null
Apellidos :null
dni :null
edad :0
Nombre :pedro
Apellidos :argibay
dni :555555
edad :26
Nombre :pepe
Apellidos :perez
dni :343434
edad :23
Inserte el número del dni para borrar la fila
343434
Fila borrada con éxito
BUILD SUCCESSFUL (total time: 23 seconds)

```

Al ejecutarlo, se ve que nos da la lista que tenemos en la base de datos, obviamente nos saltamos el proceso de añadir porque tendríamos que añadir cada vez que ejecutemos.

Se puede ver que nos pide “Inserte el número del dni para borrar la fila” hay que hacerlo por consola, introducimos el dni y ejecutamos. Nos sale el mensaje de “Fila borrada con éxito” comprobamos que ha sido así quitamos el método de la main obx.borrarJugador(); y ejecutamos:

```

run:
La conexión se ha realizado correctamente a la ruta:/home/ped90/Descargas/lista.db
Nombre :afdgsfg
Apellidos :wtwertyrt
dni :1111
edad :23
Nombre :null
Apellidos :null
dni :null
edad :0
Nombre :pedro
Apellidos :argibay
dni :555555
edad :26
BUILD SUCCESSFUL (total time: 0 seconds)

```

Lo ha borrado correctamente de la base de datos.

NOVENO PASO MODIFICAR UNA FILA DESEADA:

Vamos a modificar un dato de una fila que tengamos guardada en nuestra base de datos, para ello vamos a la clase Conectar.java y creamos el método:

```

public void actualizarJugador(){
    Scanner teclado = new Scanner(System.in);
    System.out.println("Inserte el número del dni para actualizar la fila correspondiente");
    Integer reg=teclado.nextInt();
    try{
        Statement actualiza = conectar.createStatement();
        actualiza.executeUpdate("update Jugadores set edad=27+" where dni="+reg.toString());
        System.out.println("Registro actualizado");
    }catch(SQLException ex){
        System.out.println("Error al actualizar el registro, verifique que ha introducido bien todos los datos "+ex.getMessage());
    }
}

```

Este método lo que va a hacer es introduciendole el dni que es la clave primaria y por ley no se puede repetir, modificará la edad por la de 27.

Si queremos modificar otros parámetros en vez de añadir edad le añades la columna que quieres modificar y el resultado.

Vamos a la clase Bases.java (main) y llamamos al método:

```

Conectar obx = new Conectar();

obx.conectarBase();
obx.driversBase();
obx.actualizarJugador();

```

Lo ejecutamos y nos saldrá en la ventana de comandos, que insertemos el dni de la persona que queremos realizar el cambio de edad.

```
run:
La conexión se ha realizado correctamente a la ruta:/home/ped90/Descargas/lista.db
Inserte el número del dni para actualizar la fila correspondiente
555555
Registro actualizado
```

Como se ve realizó el cambio sin problemas, ahora veremos si actualizó la edad, esta imagen es como estaba creado el usuario con dni 555555 con 26 años de edad:

```
run:
La conexión se ha realizado correctamente a la ruta:/home/ped90/Descargas/lista.db
Nombre :afdgsfg
Apellidos :wtwertyrt
dni :1111
edad :23
Nombre :null
Apellidos :null
dni :null
edad :0
Nombre :pedro
Apellidos :argibay
dni :555555
edad :26
BUILD SUCCESSFUL (total time: 0 seconds)
```

Y este es el resultado de la actualización de edad:

```
run:
La conexión se ha realizado correctamente a la ruta:/home/ped90/Descargas/lista.db
Nombre :null
Apellidos :null
dni :null
edad :0
Nombre :pedro
Apellidos :argibay
dni :555555
edad :27
```

Realizó el cambio sin problemas.

Se puede añadir un método al final que será este:

```
public void cerrarBase(){
    try {
        conectar.close();
        System.out.println("Seguridad: Base de datos cerrada con éxito.");
    } catch (SQLException ex) {
        System.out.println("Error, la base de datos está cerrada, vuelva a conectarla: "+ex.getMessage());
    }
}
```

Esto al ejecutarlo, cerrará la base de datos y no dejará modificar nada.

DÉCIMO PASO CREACIÓN DEL MENÚ EN LA MAIN:

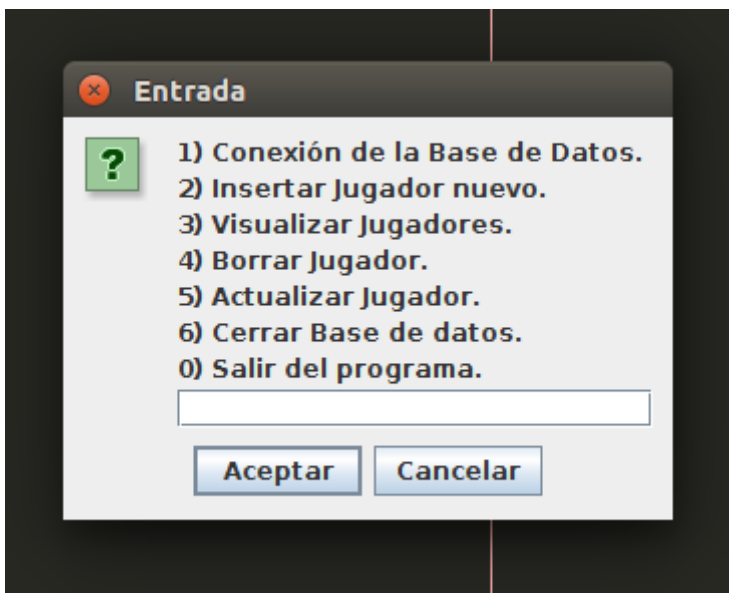
Por último paso vamos a modificar la clase Bases.java (main) y vamos a crear un menú para que esté mejor presentado y sea más claro.

```
public static void main(String[] args){
    // TODO code application logic here
    Conectar obx = new Conectar();

    int opcion;
    do{
        opcion= Integer.parseInt(OptionPane.showInputDialog("1) Conexión de la Base de Datos. \n2) Insertar Jugador nuevo. \n3) Visualizar Jugadores."
        + "\n4) Borrar Jugador. \n5) Actualizar Jugador. \n6) Cerrar Base de datos. \n0) Salir del programa."));

        switch(opcion){
            case 1:
                obx.conectarBase();
                obx.driversBase();
                break;
            case 2:
                obx.cargaArray();
                obx.insertarJugadores();
                break;
            case 3:
                obx.visualizarJugadores();
                break;
            case 4:
                obx.borrarJugador();
                break;
            case 5:
                obx.actualizarJugador();
                break;
            case 6:
                obx.cerrarBase();
                break;
            case 0:
                JOptionPane.showMessageDialog(null,"Pulse para salir del programa");
                System.exit(0);
                break;
            default:
                JOptionPane.showMessageDialog(null,"Error");
        }
    }while(opcion!=0);
}
```

Así a la hora de ejecutar el programa podremos elegir entre que opción queremos:



Recordar siempre cargar la base de datos y drivers, si no el programa no funcionará.

Un cordial saludo.

Pedro José Argibay Calvo DAM1 6150