Manual para la base de datos relacional

Autor: Lidier Máximo lopez raccioppe

Contexto

Decidí usar el MySQL, y estare usando una maquina virtual, asi como acompañarlo con un codigo para mostrar.

Índice

Descargar e Instalar

Antes de todo se tendrá que tener los archivos necesarios:

el conector de MySQL (driver)
Netbeans 17 (cualquier version funcional)
Algo para los servidores, en este caso use XAMPP

Se recomienda usar las páginas oficiales para las descargas

Pasos a Seguir para Programar

Estan divididos en grupos mayores para una mas facil navegacion

Preparar la base de datos

Para poder hacer los pasos siguientes se necesita tener ya antes una base de datos de la cual tengamos un usuario, con su respectiva contraseña, dirección URL, y el nombre de la base, y el numero de puerto si es que no se esta usando el predefinido.

Esta al ser ya la primera vez necesitara diversas cosas.

1- Necesitamos el XAMPP instalado y con el servidor de MySQL abierto

2- Debemos abrir el MySQL con la terminal o la consola de comando, abriendo la terminal en caso de linux yendo a la opción de los puntitos o en caso de Windows escribiendo cmd en la barra buscadora, para eso se debe seguir esta ruta C:\xampp\mysql\bin que en cualquier caso sería la ruta de instalacion del XAMPP, ahora se debe de usar la linea anterior pero antes con un cd escrito en la terminal o consola, y darle enter, despues escribiendo mysql -u root -h localhost -p .

```
C:\xampp\mysql\bin>cd C:\xampp\mysql\bin

C:\xampp\mysql\bin>mysql -u root -h localhost -p
```

Explicando, la línea de mysql que abre el programa mysql que estaba en esta carpeta, mientras que el “-u” es para indicarle que lo siguiente es con el usuario con el que vamos a entrar en la base, en este caso siendo root debido a que recién se lanzó el servidor y XAMMP usa esto a la hora de hacerlo, pero de otra forma basta con poner tu nombre de usuario, la “-h” indica el nombre del host o la dirección IP en donde esta el servidor que te quieres conectar, en mi caso al haberlo levantado yo en la máquina actual basta con escribir “localhost” o “127.0.0.1” que es en IP donde estaria de ser local, la “-p” es para referirse a la contraseña del servidor al que te vas a conectar. Con eso se deberia de abrir el servidor para el uso en la terminal o consola

3- Prepara la base de datos, ahora deberíamos crear una base de datos si es que no tenemos una, esto se hace con el comando “create database ‘nombreDeLaBase’”.

```
MariaDB [(none)]> create database registro;
```

Se recomienda que despues de cada paso se compruebe que todo fue hecho correctamente, para comprobarlo se puede usar el comando “show databases”.

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| phpmyadmin         |
| registro           |
| test               |
+--------------------+
```

Ahora hay que entrar en la base de datos usando el comando “use ‘nombreDeLaBase’”

```
MariaDB [(none)]> use registro;
Database changed
MariaDB [registro]>
```

Ahora se tiene que crear la tabla con la que estaremos usando con este comando

create table Civiles (CI int not null primary key, nombres nvarchar(20), apellidos nvarchar(20));

Esto siendo con objetivo demostrativo, siendo de esta una tabla con nombre Civiles, que tiene 3 campos, uno siendo “CI”, de tipo int (un tipo de data numérico entero) y que representa la identidad de la persona, por lo que es única y por tanto será usada como la clave primaria por lo que no se podrá repetir y servirá para identificar a la persona dentro de la base asi como no podra ser nula, luego el “nombres” que es otra forma de conocer a la persona, pero como este se puede repetir entre varias personas no sera usado como la clave primaria y el “(20)” indica que esta limitada a 20 nvarchar, que son los tipo char o string, y lo mismo ocurre con “apellidos”.

Ahora se puede seguir desde java.

Librería o Jar

Dependiendo de si quieres usarlo como una librería o como un Jar. Los pasos cambian ligeramente pero se entiende.

1- Extraer el ConnectorJ

2- En Netbeans ir a Projecto, Librerías, Nueva Librería.

3- Eliges el nombre que le quieras dar a la librería y el tipo, en este caso usaremos La librería de clase.

4- Seleccione añadir Jar/Carpeta ve a la carpeta en la que extrajiste el ConnectorJ, y seleccionalo para luego darle a añadir JAR/Folder.

Paralelamente o independientemente si estas usando una version de Maven puede agregar una dependencia nueva y buscar por las que sean de mysql y java, se recomienda que descarges la ultima version

Netbeans

Esta seccion es para lo que es necesario en la configuración de Netbeans

1- Creamos un nuevo proyecto en Netbeans, con un nombre a cualquier elección.

2- En la seccion de proyectos se hará click derecho y seleccionar añadir Librería, y seleccionamos la que creamos con el JAR del ConnectorJ

Codificación

Buscamos ahora el codificar todo y usarlo en el, debemos hacer una clase para la conexion

1- Importar el java.sqlConnection y el java.sql.DriveManager

2- Establecer la url string como estatica public static final String URL = "url", aca se introduciria la cadenas de url que preparamos en la base, puerto de la base y el nombre de la misma

3- Ingresar un usuario y contraseña para el mismo public static final String USER = “root”; y public static final String CLAVE “root”

4- Debemos ahora cargar el driver, para eso se recomienda que sea en una clase aparte y en un método que sea para esto como getConexion() , tendria un resultado similar a este

```
String cadena = "jdbc:mysql://" + URL + ":" + PUERTO + "/" + BASE;

    public Connection conectandose() {

        try {
            // Con es lo siguiente inicializamos dinámicamente un objeto de la clase de dentro
            Class.forName("com.mysql.cj.jdbc.Driver");

            //con esto cargamos el driver
            con = DriverManager.getConnection(cadena, USER, CLAVE);
            //  Statement stmt = con.createStatement();
            /*
            Con la declaración siguiente le decimos que cree una
            base de datos con el nombre Civil
            */
            // stmt.execute("create database Civil");
            // Para tener una retroalimentación de que salio bien
            JOptionPane.showMessageDialog(null,"Base de datos creada correctamente");
        } catch (Exception e){
            JOptionPane.showMessageDialog(null,"Error en la conexión a la base de datos" +
e.toString());
        } finally {
            try {
                // Cerramos posibles conexiones abiertas
                if (con != null) {
                    con.close();
                }
            } catch (Exception e) {
                System.out.println("Error cerrando conexiones: "
                        + e.toString());
            }
        }

        return con;
    }
```

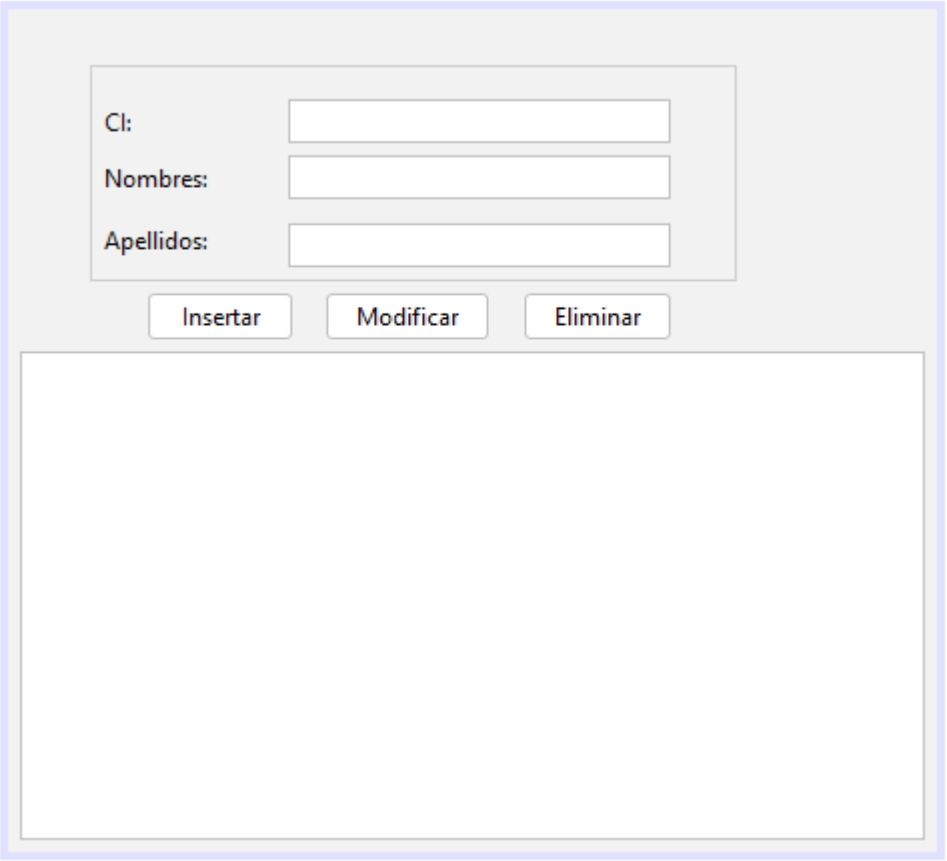
5- Ahora tenemos que abrira la conexión peculiar de MySql usando esto, siendo el metodo que hicimos en la clase Conection.

Conection conect = new Conection();
conect.conectandose()

6- crear la clase Civil y valores iniciales.
En esto decidí usar 3 variables siendo las misma que en la creación de la base, la CI que será un número (int) único que identificará a cada persona, el campo de String de nombre, con los nombres de la persona, y el campo de tipo String de apellido, con los apellidos de la persona, junto con sus getter y setter.

```
public class Civil {
    private int ci;
    private String nombresCiviles;
    private String apellidosCiviles;
```

7- Hacer la interface gráfica.
Sabiendo que necesito poder: insertar, cambiar, y eliminar, tendré que hacer una Interface que de eso, por lo que decidí hacerlo en base a un registro simple de civil basando en Uruguay, motivo por lo que uso CI y no DNI, o Nombres envés de Nombre.



Siendo el primer campo donde se ingresa la CI del civil, luego los nombres y luego los apellidos, los botones Insertar, Modificar y Eliminar harán sus eventos pero antes se debe de programar, y luego está la tabla en la que se mostrarán los datos, como de momento está vacía no tiene ninguno dentro.

Los botones son bastante autoexplicativos, ahora se explicara sus métodos

8-Agregar los métodos a Civil

Seguiremos haciendo los métodos de insertar los datos de un civil, mostrar los datos del mismo, seleccionar una fila en la tabla para facilitar su uso, el modificar un valor y por último el de eliminar.

Según se va haciendo el codigo este se ira explicando, antes dire que todos estos métodos se encuentran en la clase de Civil.

Insertar -

```
//este metodo inserta valores de los campos de texto en la base de datos


public void insertarCivil(JTextField CIObt,JTextField nombresObt,JTextField apellidosObt){
        setCi(parseInt(CIObt.getText()));
        setNombresCiviles(nombresObt.getText());
        setApellidosCiviles(apellidosObt.getText());
        //esto sera usado para despues
        String consulta = "insert into Civiles (CI, nombres, apellidos) values (?,?,?);";
        Conection conect = new Conection();

        try {
            /*
            prepara un comando usando el string que les demos, pero antes debe de conectarse
al base,
            tambien debemos poner los argumentos
            */
            CallableStatement cs = conect.conectandose().prepareCall(consulta);
            /*esto es para que en los interogantes que dejamos antes estos
            sean llenados en el orden de los numero que pondremos ahora*/
            cs.setString (1, Integer.toString(getCi()));
            cs.setString (2, getNombresCiviles());
            cs.setString (3, getApellidosCiviles());
            // Ejecutara todas la linea preparadas disponibles en orden
            cs.execute();
            // para la retroalimentacion
            JOptionPane.showMessageDialog(null, "La Insercion fue exitosa");
        }catch(Exception e){
```

```
            JOptionPane.showMessageDialog(null, "Hubo un error en la
insercion"+e.getMessage());
        }
    }
```

Mostrar -

```
// esto mostrara los cambios que sufra la tabla dentro de la base de datos y los mostrara
// en la tabla de la aplicacion


public void mostrarCiviles (JTable datosTablaCivil){
        Conection objConexion = new Conection();
        /*
        Este nos permite usar metodos que manejen el uso de tablas, del
        estilo de JTable sin tener que implementar TableModel, que
        es una interface.
        */
        DefaultTableModel modeloTabla = new DefaultTableModel();
        /*
        TableRowSorter, permite el ordenar y filtrar tablas del estilo
        de JTable (aca usaremos las de DefaultTableModel).
        */
        TableRowSorter<TableModel> ordenarTabla = new TableRowSorter<TableModel>
(modeloTabla);
        datosTablaCivil.setRowSorter(ordenarTabla);
        //la consulta
        String sql="select * from Civil;";
        // cada una agrega una columna extra a la visualizacion con el nombre que elijamos
        modeloTabla.addColumn("id");
        modeloTabla.addColumn("nombres");
        modeloTabla.addColumn("apellidos");
        // Esto actualiza la tabla con los nuevos cambios que sufrio
        // desde su ultima setModel o iniciacion
        datosTablaCivil.setModel(modeloTabla);

        /*
        Esto sera usado para obtener los datos de la base de datos por
        linea
        */
        String [] datos = new String [3];
        //Esto se usara para executar la consulta (aca lo llamamos sql);
        Statement st;
        try{
            st = objConexion.conectandose().createStatement();
            // el ResultSet sera para guardar los resultados de la consulta
```

```
        ResultSet resultado = st.executeQuery(sql);
         /*
        Esto sera lo que tome lo obtenido de la consulta y
        lo ingrese en el visual
        */
        while (resultado.next()){
            datos[0]=resultado.getString(1);
            datos[1]=resultado.getString(2);
            datos[2]=resultado.getString(3);
            // esto añade los datos en las lineas
            modeloTabla.addRow(datos);
        }
        //esto actuliza la apariencia de la tabla
        datosTablaCivil.setModel(modeloTabla);
        }catch (Exception ex){
            JOptionPane.showMessageDialog(null, "Error en la lectura de
datos"+ex.toString());
        }
    }
```

Seleccionar -

//este metodo es para que cuando alguien seleccione una fila en la tabla, este le de los valores

```
public void seleccionarCivil(JTable datosTablaCivil, JTextField datosCI,JTextField
datosNombres,JTextField datosApellidos){

    try{
        //getSelectRow es para ver el indice de la linea
        int fila = datosTablaCivil.getSelectedRow();
        // es para ver si se selecciono algo
        if (fila >= 0 ){
            /*
            Esto tomara los datos de la tabla y sus campos y se los dara a los
            textFields que corresponda, esto facilitara por si se tiene que
            hacer modificaciones pequeñas o solo cambiar un campo
            */
            datosCI.setText(datosTablaCivil.getValueAt(fila, 0).toString());
            datosNombres.setText(datosTablaCivil.getValueAt(fila, 1).toString());
            datosApellidos.setText(datosTablaCivil.getValueAt(fila, 2).toString());
            // aca no pongo ningun feedback por que se ve si se llenan los textFields
        }
        else{
            //por si no han seleccionado la linea
            JOptionPane.showMessageDialog(null, "Fila no se selecciono");
        }
```

```
        }catch (Exception ex){
            JOptionPane.showMessageDialog(null, "Error en la seleccion"+ex.toString());
        }
    }
```

Modificar -

// esto permite alterar los valores de una fila dentro de la tabla sin tener que eliminarla y volverla a insertar

```
public void modificarCivil(JTextField datosCI, JTextField datosNombres, JTextField
datosApellidos){

        //obtengo y guardo los datos
        setCi(Integer.parseInt(datosCI.getText()));
        setNombresCiviles(datosCI.getText());
        setApellidosCiviles(datosCI.getText());
        // establesco conexion
        Conection objCon = new Conection();
        // la consulta que actualizara los campos nombres y apellidos donde la id
        // sea igual a la id ingresada
        String consulta = "update Civil set civil.nombres = ?, civil.apellidos = ? where civil.id =
?";

        try {
            /*
            prepara un comando usando el string que les demos, pero antes debe de conectarse
al base,
            tambien debemos poner los argumentos
            */
            CallableStatement cs = objCon.conectandose().prepareCall(consulta);
            /*esto es para que en los interogantes que dejamos antes estos
            sean llenados en el orden de los numero que pondremos ahora*/
            cs.setInt(1, getCi());
            cs.setString(2, getNombresCiviles());
            cs.setString(3, getApellidosCiviles());
            // Ejecutara todas la linea preparadas disponibles en orden
            cs.execute();
            // para la retroalimentacion
            JOptionPane.showMessageDialog(null, "Modificado correctamente");
        }catch (Exception ex){
            JOptionPane.showMessageDialog(null, "Error en la modificacion"+ex.toString());
        }
    }
```

Eliminar -

```
// esto permitirá eliminar a un civil del registro

public void eliminarCiviles(JTextField dataCI){
        //obtengo y guardo los datos
        setCi(Integer.parseInt(dataCI.getText()));
        // cargo la conexion
        Conection objCon = new Conection();
        // consulta en sql que borrara la linea que contenga
        // la ci indicada
        String consulta = "delete from Civil where alumnos.ci = ? ;";

        try{
            /*
            prepara un comando usando el string que les demos, pero antes debe de conectarse
al base,
            tambien debemos poner los argumentos
            */
            CallableStatement cs = objCon.conectandose().prepareCall(consulta);
            /*esto es para que en los interogantes que dejamos antes estos
            sean llenados en el orden de los numero que pondremos ahora*/
            cs.setInt(1, getCi());
            // Ejecutara todas la linea preparadas disponibles en orden
            cs.execute();
            // para la retroalimentacion
            JOptionPane.showMessageDialog(null, "Eliminado correctamente");
        }catch (Exception ex){
            JOptionPane.showMessageDialog(null, "Error al eliminar"+ex.toString());
        }
    }
```

9- Establecer los eventos
Uso 5 eventos, 3 son para los botones y 2 para la tabla,

Evento Insertar-

```
private void botonGuardarActionPerformed(java.awt.event.ActionEvent evt) {
        Civil objCivil = new Civil();
        objCivil.insertarCivil(textoCI, textoNombres, textoApellidos);
        objCivil.mostrarCiviles(tablaCivil);
    }
```

Evento Mostrar-

// este se encuentra en varios eventos, siendo esto para mostrar los cambios que sufra la tabla, en el unico que no esta es en el de hacer click en la tabla por que no tendría sentido

Evento Seleccionar-

```
private void tablaCivilMouseClicked(java.awt.event.MouseEvent evt) {
        Civil objCivil = new Civil();
        objCivil.seleccionarCivil(tablaCivil, textoCI, textoNombres, textoApellidos);
    }
```

Evento Modificar-

```
private void botonModificarActionPerformed(java.awt.event.ActionEvent evt) {
        Civil objCivil = new Civil();
        objCivil.modificarCivil(textoCI, textoNombres, textoApellidos);
        objCivil.mostrarCiviles(tablaCivil);
    }
```

Evento Eliminar-

```
private void botonEliminarActionPerformed(java.awt.event.ActionEvent evt) {
        Civil objCivil = new Civil();
        objCivil.eliminarCiviles(textoCI);
        objCivil.mostrarCiviles(tablaCivil);
    }
```

Pasos a Seguir para Usuario Administrativo

1- Abrir la aplicación

2- De querer añadir a una nueva persona al registro civil se tendrá que introducir la CI, recuerde que el CI nunca se puede repetir, que es un número entero, los nombres, que están limitados a 20 caracteres, apellidos, que también están limitados a 20 caracteres. Una vez ingresados estos se tendrá que apretar en el botón de Insertar y estará listo.

3- De querer ver los datos dentro solo tendrá que ver en la tabla y verá todos los registros en ella.

4- Si desea Modificar algún registro tendrá que hacer click izquierdo en el registro de desee cambiar, o ingresar la CI, debe saber que solo se puede cambiar los nombres y los apellidos, la CI e inmutable. Lo que tendrá que hacer es después de meter la CI debe de ingresar los nuevos nombres y/o nuevos apellidos del civil, y si desea solo cambiar uno tiene que dejar el de antes tal y como estaba previamente.

5- Para eliminar a alguien del registro debe de seleccionarlo con el click izquierdo del mouse o ingresar su CI, una vez debe de darle a Eliminar y ese registro será borrado de la tabla.

6- Para salir debe de apretar la equis roja y se cerrará el programa

Mi Github

Para seguir la guia se diseñara proyecto para acompañar
se encontrara en el siguiente github:
https://github.com/LidierRaccioppe/Base-de-datos-Java.git

Bibliografía

https://www.cablenaranja.com/como-conectar-java-con-mysql-en-netbeans/
https://lineadecodigo.com/java/conectar-mysql-java/
https://chuidiang.org/index.php?title=Conectar_java_con_mysql
https://es.stackoverflow.com/questions/138638/conectar-java-con-mysql
https://www.youtube.com/watch?v=65WgYJ5neMM
https://es.stackoverflow.com/questions/76160/por-qu%C3%A9-es-necesario-usar-class-fornamecom-mysql-jdbc-driver
https://www.youtube.com/watch?v=6_ncKPDvKEg&t=1064s
https://www.youtube.com/watch?v=kPCbb80_6GI
http://www.jtech.ua.es/historico/paj/restringido/apuntes/sesion10-apuntes.htm
https://www.youtube.com/watch?v=nxV1IJA8JCU
https://www.youtube.com/watch?v=vCJDwcFwHPE
https://cursos.virtual.uniandes.edu.co/csof5302/configuracion-de-la-conexion-a-una-base-de-datos-en-netbeans/
https://lineadecodigo.com/java/crear-una-base-de-datos-en-java/
https://www.youtube.com/watch?v=mIOZIWFzkAc
https://lineadecodigo.com/java/crear-una-base-de-datos-en-java/
https://www.clasesdeinformaticaweb.com/java-desde-cero/java-con-bases-de-datos/
https://chuidiang.org/index.php?title=Establecer_conexi%C3%B3n_con_base_de_datos_desde_java
https://snmb-desarrollo.readthedocs.io/en/develop/howtos/database-connection.html
https://www.oracle.com/database/technologies/appdev/jdbc.html
https://codigosdeprogramacion.com/cursos/?lesson=3-crud-en-java-y-mysql