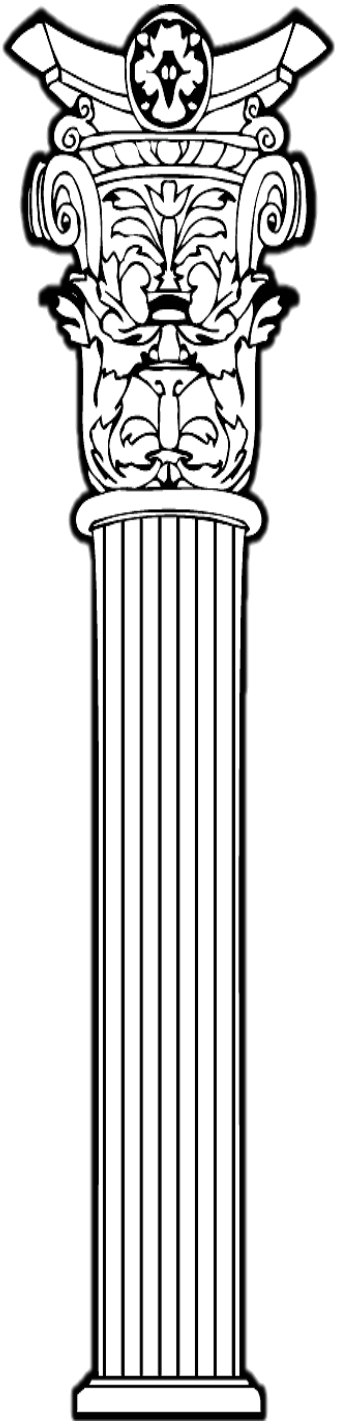
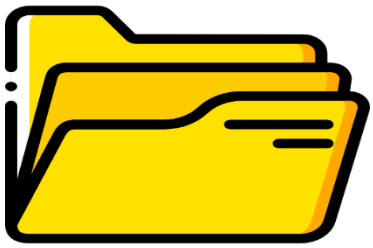
Universidad Autónoma del Estado De México

Secretaria de Docencia

Dirección de Infraestructura Académica

Centro Universitario UAEM Atlacomulco





Unidad de Aprendizaje: Base de datos I

Actividad: Proyecto JDBC.

Nombre del Alumno:

Luz Magali Cruz Isidro

Fredy Flores Cruz

Saul Israel Valencia Vargas

Jesús González Becerril

Nombre del docente: Laura Rivas Colín

Ingeniería en computación

Tercer Semestre (ICO-25)

Contenido

Proyecto JDBC

1. Resumen

JDBC significa Java™ EE Database Connectivity (conectividad de bases de datos Java). En desarrollo de Java EE se trata de una tecnología muy conocida que se utiliza de forma habitual para implementar la interacción de bases de datos. JDBC es una API de nivel de llamada, lo que significa que las sentencias SQL se pasan como series a la API que, posteriormente, se encarga de ejecutarlas en RDMS. Por ello, el valor de estas series se puede modificar durante el tiempo de ejecución, haciendo que JBDC sea dinámica.

Mientras que los programas JDBC se ejecutan de forma más lenta que sus equivalentes SQLJ, una ventaja de este método es un concepto denominado "Write once, call anywhere" (escribir el código una sola vez y ejecutarlo en cualquier plataforma). Esto significa que, puesto que no se necesita ninguna interacción hasta el tiempo de ejecución, un programa JDBC es muy portable y se puede emplear entre dos sistemas distintos sin ningún tipo de preocupación.

2. Introducción

La JDBC nos permitirá acceder a bases de datos (BD) desde Java. Con JDBC no es necesario escribir distintos programas para distintas BD, sino que un único programa sirve para acceder a BD de distinta naturaleza. Incluso, podemos acceder a más de una BD de distinta fuente (Oracle, Access, MySql, etc.) en la misma aplicación. Podemos pensar en JDBC como el puente entre una base de datos y nuestro programa Java. Un ejemplo sencillo puede ser un applet que muestra dinámicamente información contenida en una base de datos. El applet utilizará JDBC para obtener dichos datos.

El esquema a seguir en un programa que use JDBC es el siguiente:

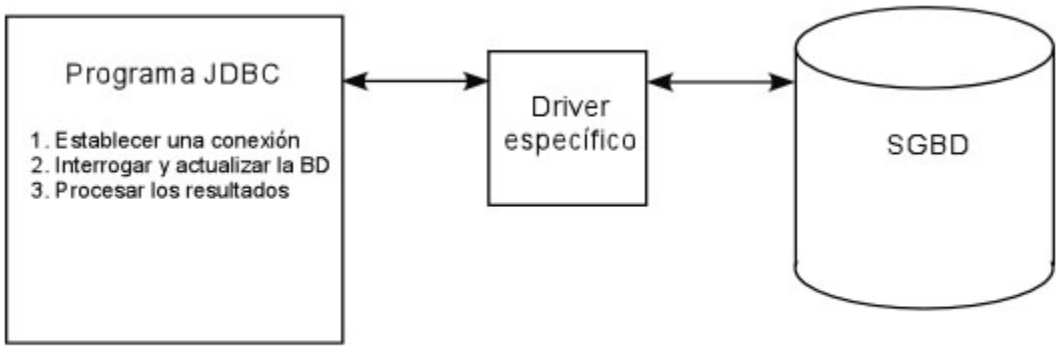


Figura 1. Esquema general de conexión con una base de datos.

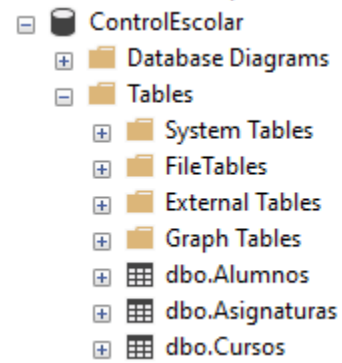
3.Desarrollo de la practica

3.1 Problemática

Realizar un CRUD mediante JAVA y SQL Server usando conexión JDBC.

3.4 Código y estructura

El modelo de Base de datos a usar será el siguiente:



ARTHUX-PC.Control...lar - dbo.Alumnos

ID	Nombre	No_lista	Grupo	Tutores	Clave_C	Clave_A	No_Cuenta
39	Maria	29	ICO25	Martin	5	22	2023786
60	Maria	16	ICO25	Javier	5	23	3456324
61	Jesus	18	ICO25	Jose Luis	5	22	2024493
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

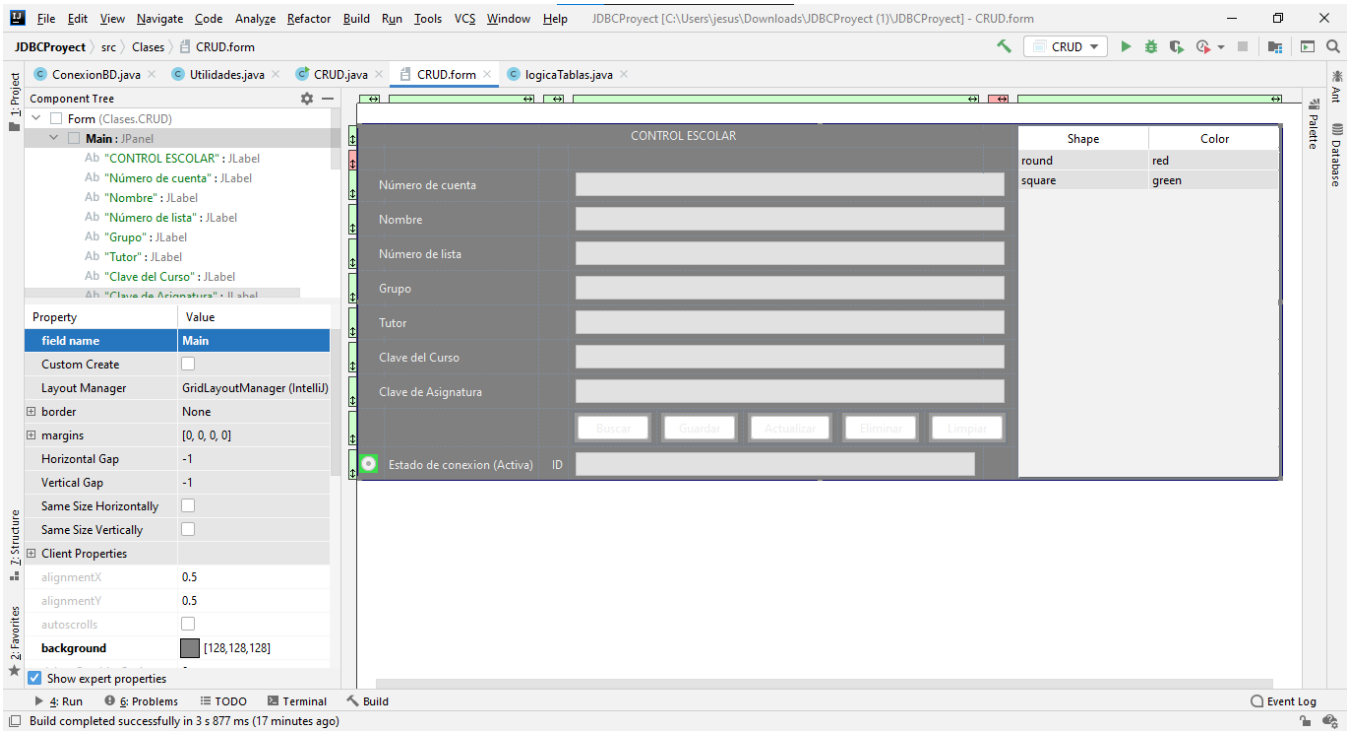
ARTHUX-PC.Control...- dbo.Asignaturas

	Clave	Nombre_A	Horario	Creditos	Docente	GrupoID
▶	16	Calculo	7:00 am	15	David	ICO25
	22	Base de datos	9:00 am	20	Laura	ICO25
	23	Probabilidad	13:00 pm	15	Monse	ICO25
	24	Inglés	9:00 pm	15	Victor	*NULL*
*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

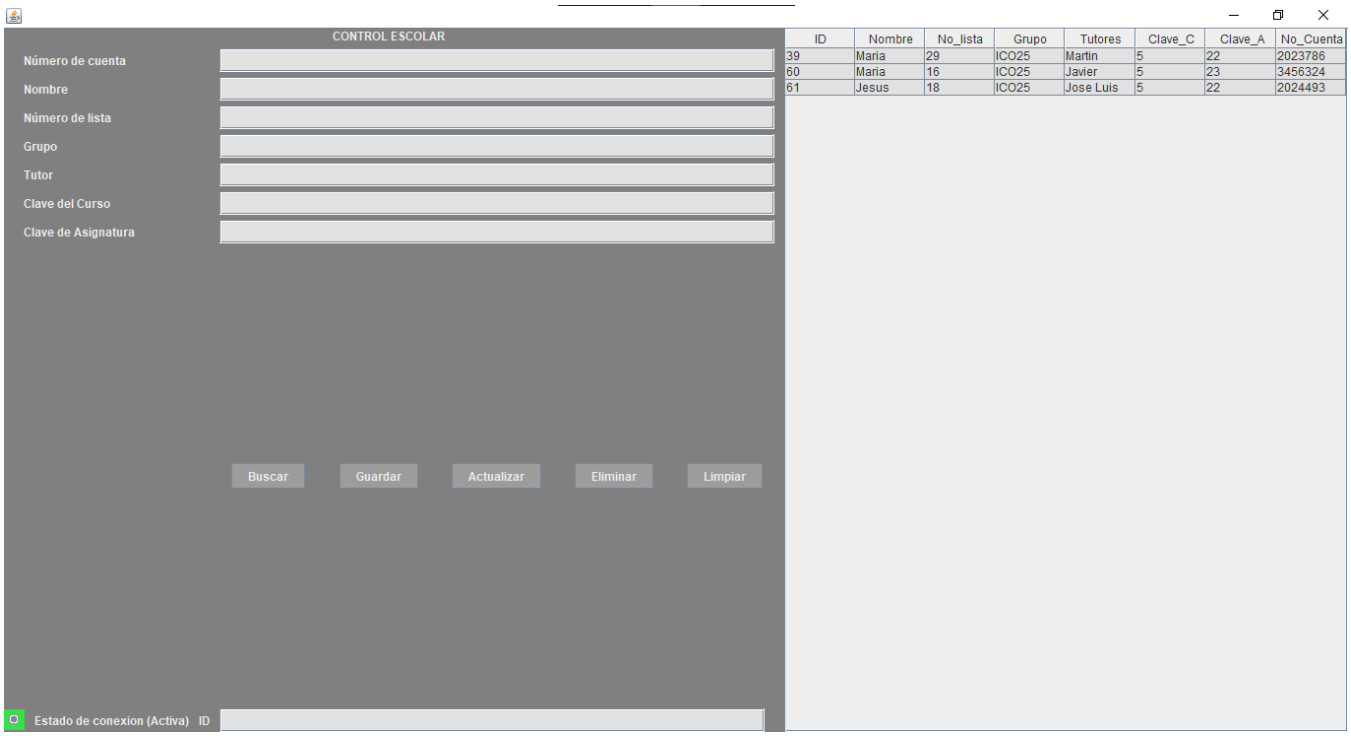
ARTHUX-PC.ControlEscolar - dbo.Cursos

	Clave_Curso	Nombre_C	Modulo	Duracion	Nivel
▶	5	Aplicaciones	II	30 dias	avanzado
	6	Desarrollo Web	II	60 dias	Intermedio
*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

Iniciamos diseñando la siguiente interfaz grafica sobre la que se mostraran los datos de este CRUD:



La cual genera la siguiente ventana:



El cual posee Etiquetas para definir cada campo de texto, una tabla, un Radio Botón que se encarga de mostrar la conexión y un selector de ID para las operaciones de búsqueda y eliminación.
El botón buscar toma el valor contenido en el campo ID, los busca en la BD y lo muestra, el botón guardar funciona cuando se ingresan datos validos en los campos de texto pertenecientes a un nuevo registro valido y lo almacena en la BD, el botón actualizar toma los valores de los campos de texto y el valor del campo ID para modificar los datos en la fila correspondientes al ID, el botón eliminar toma el valor de ID para eliminar registros por su respectivo ID, y finalmente el botón limpiar se encarga de eliminar cualquier dato en los campos de texto parea empezar nuevamente.

Y el código:

Conexión.java

```
package Clases;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class ConexionBD {
    Connection con = null;

    public Connection getConexion() {
        try {
            String conexionUrl =
"jdbc:sqlserver://localhost:1433;database=ControlEscolar;user=sa;password=Shared198
0;";
            con = DriverManager.getConnection(conexionUrl);
            //System.out.println("Conexion exitosa");
```

```
        }catch (SQLException e){
            System.out.println(e.getMessage());
            System.out.println(e.getErrorCode());
        }
        return con;
    }
}
```

CRUD.java

```
package Clases;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.*;
import java.util.logging.Level;
import java.util.logging.Logger;

public class CRUD extends JFrame {
    public static String Instancia;
    public static String Puerto;
    public static String User;
    public static String Pass;
    public static String BaseName;

    private JTextField id;
    private JTextField ncuenta;
    private JTextField nombre;
    private JTextField nlista;
    private JTextField grupo;
    private JTextField tutor;
    private JTextField curso;
    private JTextField asignatura;
    private JTable tbce;
    private JPanel Main;
    private JButton Buscar;
    private JButton Guardar;
    private JButton Actualizar;
    private JButton Eliminar;
    private JButton Limpiar;
    private JRadioButton estadoDeConexionRadioButton;


    void inializarComponente() {

        JPanel panel = new JPanel();
        panel.setLayout(null);
        panel.setBackground(new Color(128, 128, 128));
        getContentPane().add(panel);

        //Encabezado (ACCESO)
        JLabel etiqueta1 = new JLabel("Acceso");
        etiqueta1.setBounds(143, 5, 100, 20);
```

```
        etiqueta1.setForeground(Color.WHITE);
        Font letra = new Font("Times New Roman", Font.PLAIN, 25);
        etiqueta1.setFont(letra);
        panel.add(etiqueta1);

        //etiqueta Img
        ImageIcon TIENDA = new ImageIcon("src/Recursos/login.png");
        JLabel etiquetaIMG = new JLabel(TIENDA);
        etiquetaIMG.setBounds(88, 10, 190, 190);
        etiquetaIMG.setIcon(new ImageIcon(TIENDA.getImage().getScaledInstance(140,
140, 5)));
        panel.add(etiquetaIMG);

        //Etiqueta Datos
        //Campo Instancia
        JLabel etiquetadatos1 = new JLabel("Instancia:");
        etiquetadatos1.setBounds(55, 177, 500, 30);
        etiquetadatos1.setForeground(Color.BLACK);
        Font letradatos1 = new Font("Times New Roman", Font.PLAIN, 15);
        etiquetadatos1.setFont(letradatos1);
        panel.add(etiquetadatos1);

        JTextField textodatos1 = new JTextField();
        textodatos1.setBounds(115, 185, 140, 20);
        panel.add(textodatos1);
        //Campo Puerto
        JLabel etiquetadatos2 = new JLabel("Puerto:");
        etiquetadatos2.setBounds(55, 208, 500, 30);
        etiquetadatos2.setForeground(Color.BLACK);
        Font letradatos2 = new Font("Times New Roman", Font.PLAIN, 15);
        etiquetadatos2.setFont(letradatos2);
        panel.add(etiquetadatos2);
        JTextField textodatos2 = new JTextField();
        textodatos2.setBounds(110, 215, 60, 20);
        panel.add(textodatos2);
        //Campo Usuario
        JLabel etiquetadatos3 = new JLabel("Usuario:");
        etiquetadatos3.setBounds(55, 238, 500, 30);
        etiquetadatos3.setForeground(Color.BLACK);
        Font letradatos3 = new Font("Times New Roman", Font.PLAIN, 15);
        etiquetadatos3.setFont(letradatos3);
        panel.add(etiquetadatos3);
        JTextField textodatos3 = new JTextField();
        textodatos3.setBounds(115, 245, 100, 20);
        panel.add(textodatos3);
        //Campo Contrasena
        JLabel etiquetadatos4 = new JLabel("Contraseña:");
        etiquetadatos4.setBounds(55, 268, 500, 30);
        etiquetadatos4.setForeground(Color.BLACK);
        Font letradatos4 = new Font("Times New Roman", Font.PLAIN, 15);
        etiquetadatos4.setFont(letradatos4);
        panel.add(etiquetadatos4);
        //Campo Database
        JTextField textodatos4 = new JTextField();
        textodatos4.setBounds(133, 275, 100, 20);
        panel.add(textodatos4);
        //Campo Base de datos
```

```
        JLabel etiquetadatos5 = new JLabel("Base de Datos:");
        etiquetadatos5.setBounds(55, 298, 500, 30);
        etiquetadatos5.setForeground(Color.BLACK);
        Font letradatos5 = new Font("Times New Roman", Font.PLAIN, 15);
        etiquetadatos5.setFont(letradatos5);
        panel.add(etiquetadatos5);

        JTextField textodatos5 = new JTextField();
        textodatos5.setBounds(150, 305, 100, 20);
        panel.add(textodatos5);


        //Etiqueta Boton
        ActionListener accion = new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                //Llamas a las variables donde almacenaras los datos solo llamas a
los valores
                //hasta que tocas el boton
                //Les almacenamos el valor contenido en los Campos de texto
                Instancia = textodatos1.getText();
                Puerto = textodatos2.getText();
                User = textodatos3.getText();
                Pass = textodatos4.getText();
                BaseName = textodatos5.getText();

                //Los mostramos en consola como medida de control (puedes borrarlos
despues)
                System.out.println("Instancia: " + Instancia);
                System.out.println("Puerto: " + Puerto);
                System.out.println("Usuario: " + User);
                System.out.println("Password: " + Pass);
                System.out.println("Database: " + BaseName);

                //Creamos la conexion pero solo despues de guardar los datos
                ConexionBD cc = new ConexionBD();
                Connection cn = cc.getConexion();
            }

        };

        JButton boton = new JButton("Entrar");
        boton.setBounds(125, 340, 100, 25);
        panel.add(boton);
        boton.addActionListener(accion);

    }

    void mostrardatos(String valor) {
        DefaultTableModel modelo = new DefaultTableModel();
        modelo.addColumn("ID");
        modelo.addColumn("Nombre");
        modelo.addColumn("No_lista");
        modelo.addColumn("Grupo");
        modelo.addColumn("Tutores");
        modelo.addColumn("Clave_C");
        modelo.addColumn("Clave_A");
```

```
        modelo.addColumn("No_Cuenta");
        tbce.setModel(modelo);
        String sql = "";
        if (valor.equals("")) {
            sql = "SELECT * FROM Alumnos";
        } else {
            sql = "SELECT * FROM Alumnos WHERE ID='" + valor + "'";
        }
        String[] registros = new String[8];
        try {
            Statement st = cn.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                registros[0] = rs.getString(1);
                registros[1] = rs.getString(2);
                registros[2] = rs.getString(3);
                registros[3] = rs.getString(4);
                registros[4] = rs.getString(5);
                registros[5] = rs.getString(6);
                registros[6] = rs.getString(7);
                registros[7] = rs.getString(8);
                modelo.addRow(registros);
            }
            tbce.setModel(modelo);

        } catch (SQLException ex) {
            Logger.getLogger(CRUD.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    void limpiarcampos() {
        id.setText(null);
        ncuenta.setText(null);
        nombre.setText(null);
        nlista.setText(null);
        grupo.setText(null);
        tutor.setText(null);
        curso.setText(null);
        asignatura.setText(null);
    }

    public CRUD() {
        mostrardatos("");
        Limpiar.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                limpiarcampos();
            }
        });
        Guardar.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {

                    PreparedStatement pst = cn.prepareStatement("INSERT INTO
Alumnos"
                            +
```

```
"(No_cuenta,Nombre,No_lista,Grupo,Tutores,Clave_C,Clave_A)VALUES(?,?,?,?,?,?,?)");
                    pst.setString(1, ncuenta.getText());
                    pst.setString(2, nombre.getText());
                    pst.setString(3, nlista.getText());
                    pst.setString(4, grupo.getText());
                    pst.setString(5, tutor.getText());
                    pst.setString(6, curso.getText());
                    pst.setString(7, asignatura.getText());
                    pst.executeUpdate();
                    mostrardatos("");
                    limpiarcampos();
                    JOptionPane.showMessageDialog(null, "DATOS GUARDADOS
CORRECTAMENTE");
                } catch (HeadlessException | SQLException ex) {
                    JOptionPane.showMessageDialog(null, ex);
                }
            }
        });
        Buscar.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                mostrardatos(id.getText());
            }
        });
        Actualizar.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    PreparedStatement pst = cn.prepareStatement
                            ("UPDATE Alumnos SET No_cuenta='" + ncuenta.getText() +
"'," + "Nombre='" + nombre.getText() + "',"
                                    + "No_lista='" + nlista.getText() + "'," +
"Grupo='" + grupo.getText() + "',"
                                    + "Tutores='" + tutor.getText() + "'," +
"Clave_C='" + curso.getText() + "',"
                                    + "Clave_A='" + asignatura.getText() + "'"
                                    + "WHERE ID='" + id.getText() + "'");
                    pst.executeUpdate();
                    mostrardatos("");
                    limpiarcampos();
                    JOptionPane.showMessageDialog(null, "DATOS MODIFICADOS
CORRECTAMENTE");
                } catch (SQLException ex) {
                    System.out.println(ex.getMessage());
                }
            }
        });
        Eliminar.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    PreparedStatement pst = cn.prepareStatement
                            ("DELETE FROM Alumnos WHERE ID='" + id.getText() +
"'");
                    pst.executeUpdate();
                    mostrardatos("");
                    limpiarcampos();
```

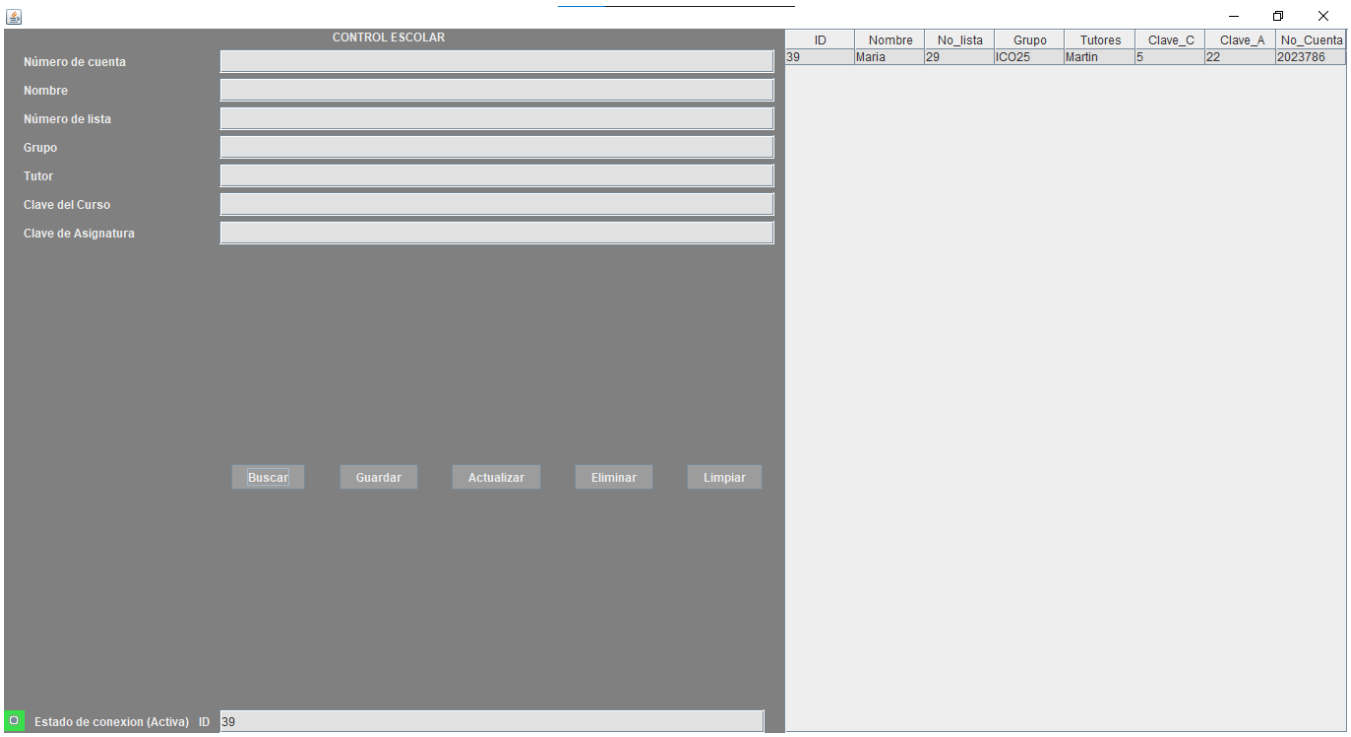
```
                    JOptionPane.showMessageDialog(null, "REGISTRO ELIMINADO
CORRECTAMENTE");
                } catch (SQLException ex) {
                }
            }
        });
    }

    public static void main(String[] args) {

        CRUD crud = new CRUD();
        crud.setContentPane(new CRUD().Main);
        crud.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        crud.setVisible(true);
        crud.pack();

    }

    ConexionBD cc = new ConexionBD();
    Connection cn = cc.getConexion();
}
```

4.Resultados

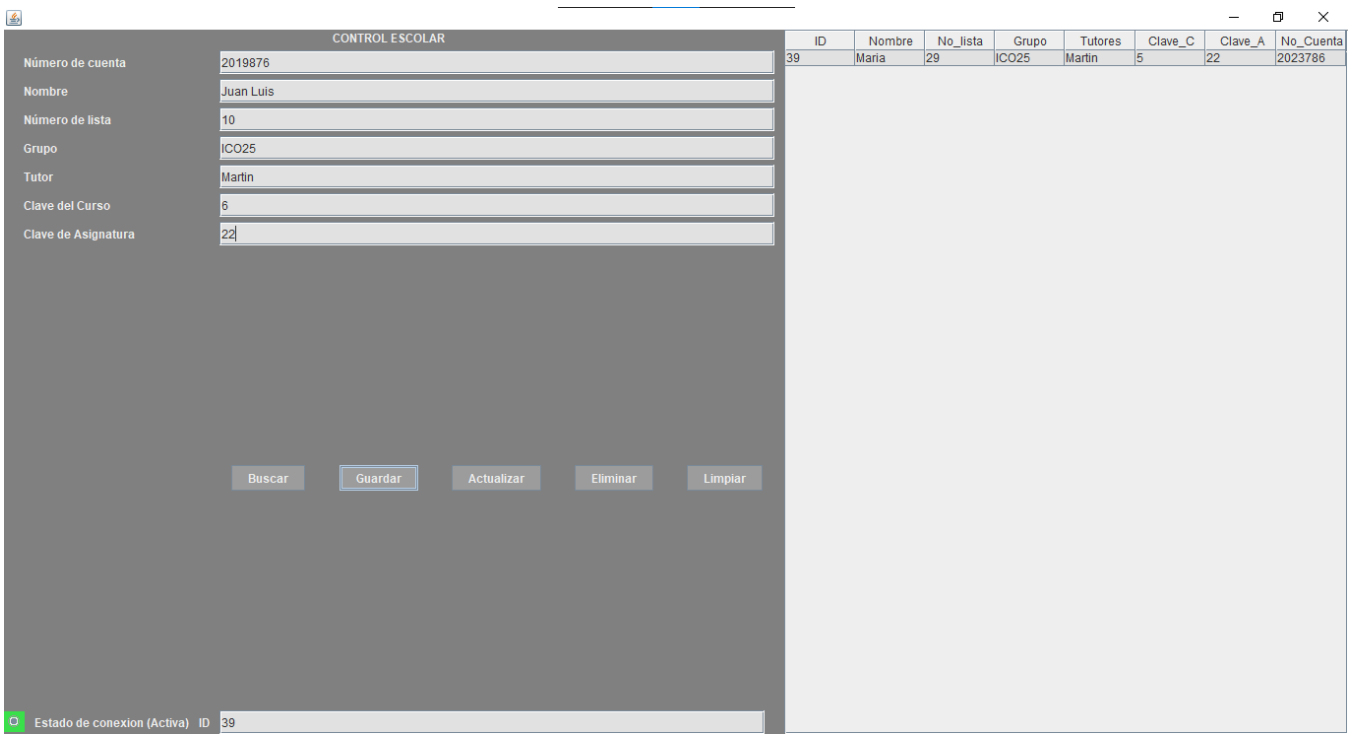
Para comprender mejor su funcionamiento hicimos sus respectivas pruebas.

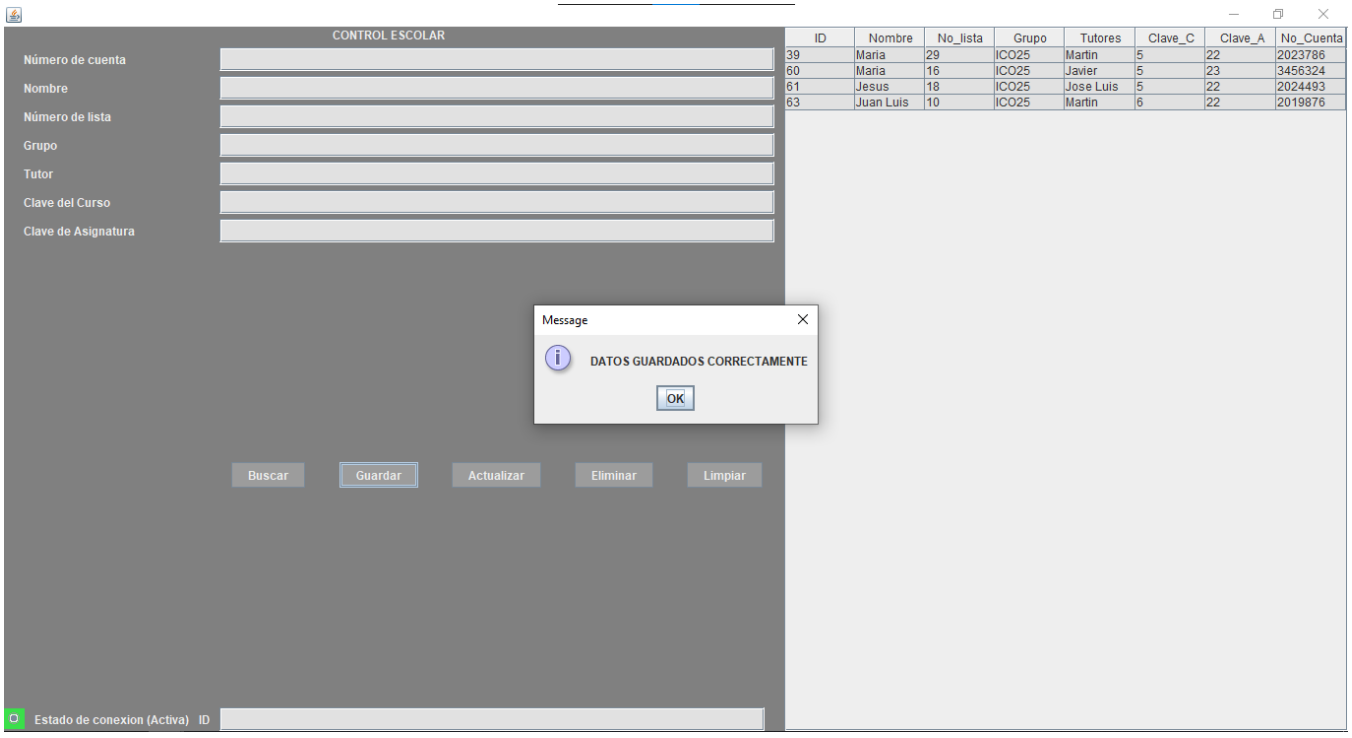
4.1 Salidas de pantalla

Buscar:

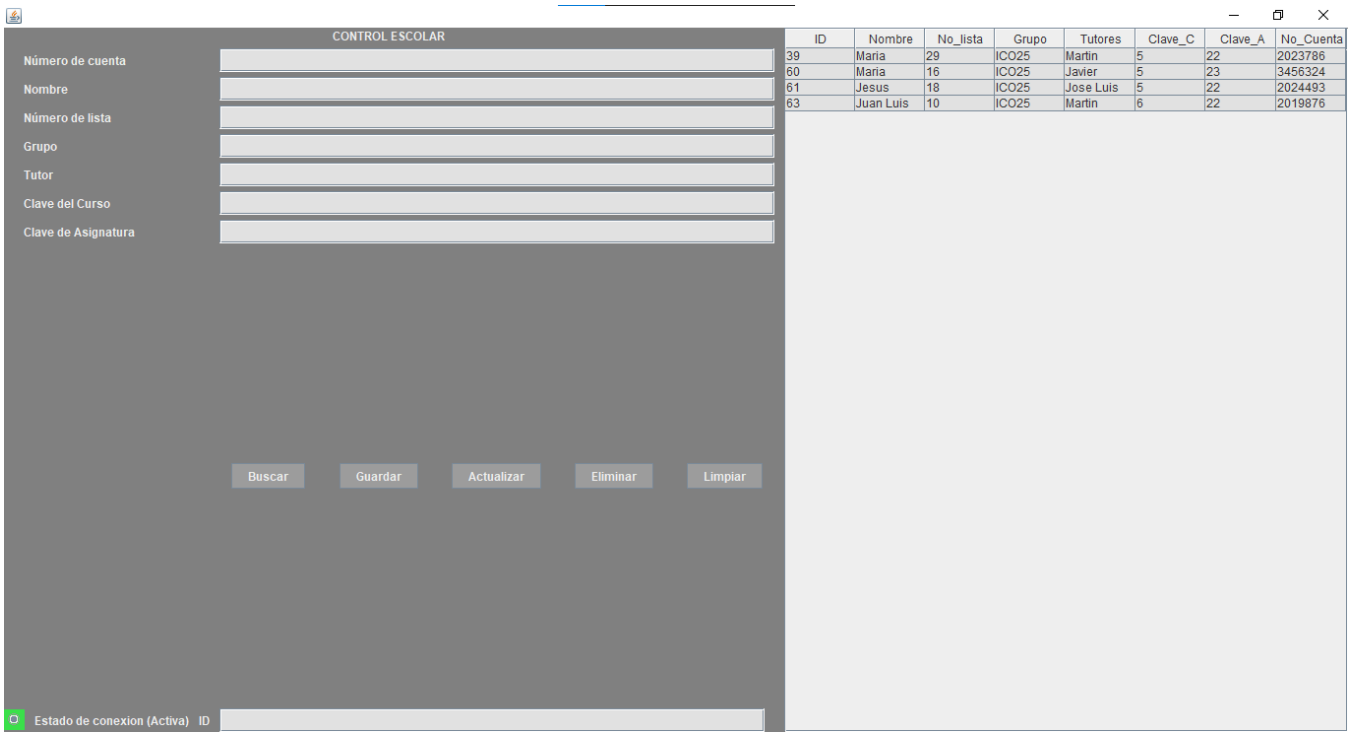


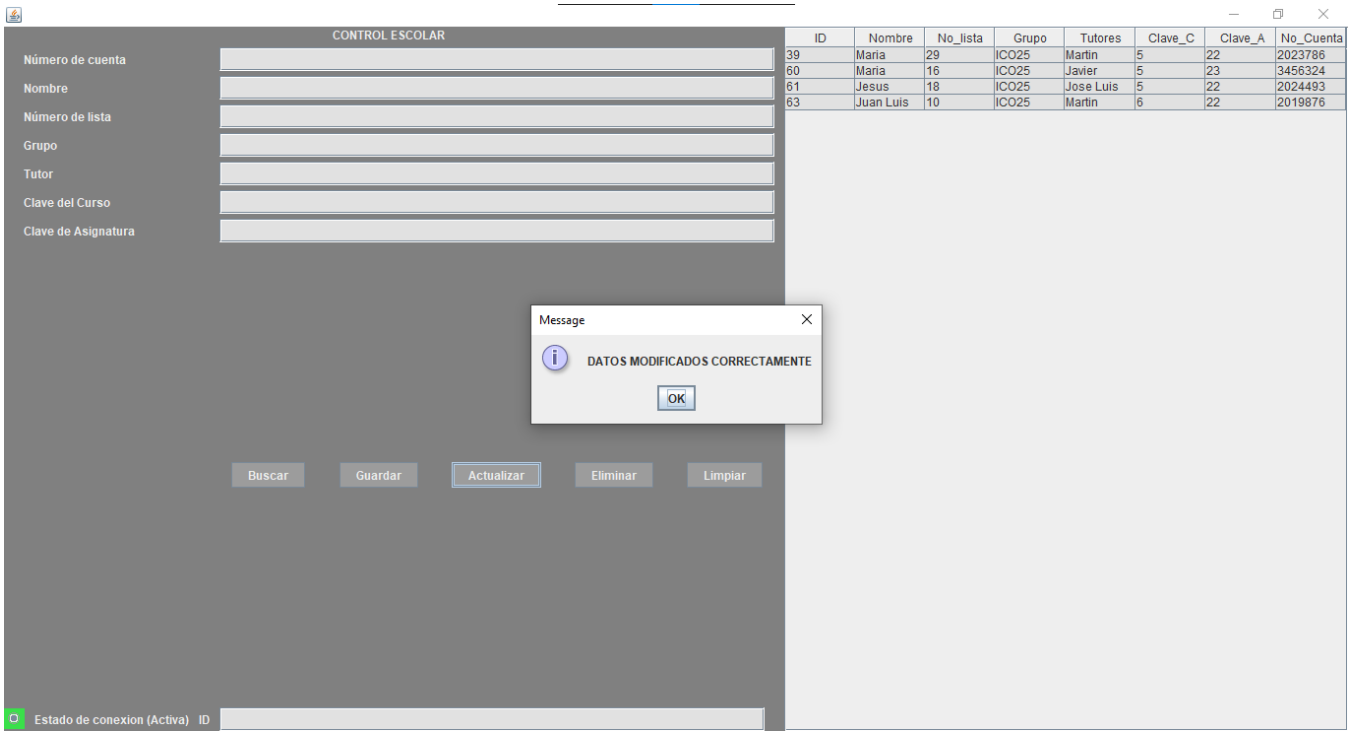
Guardar nuevo registro:



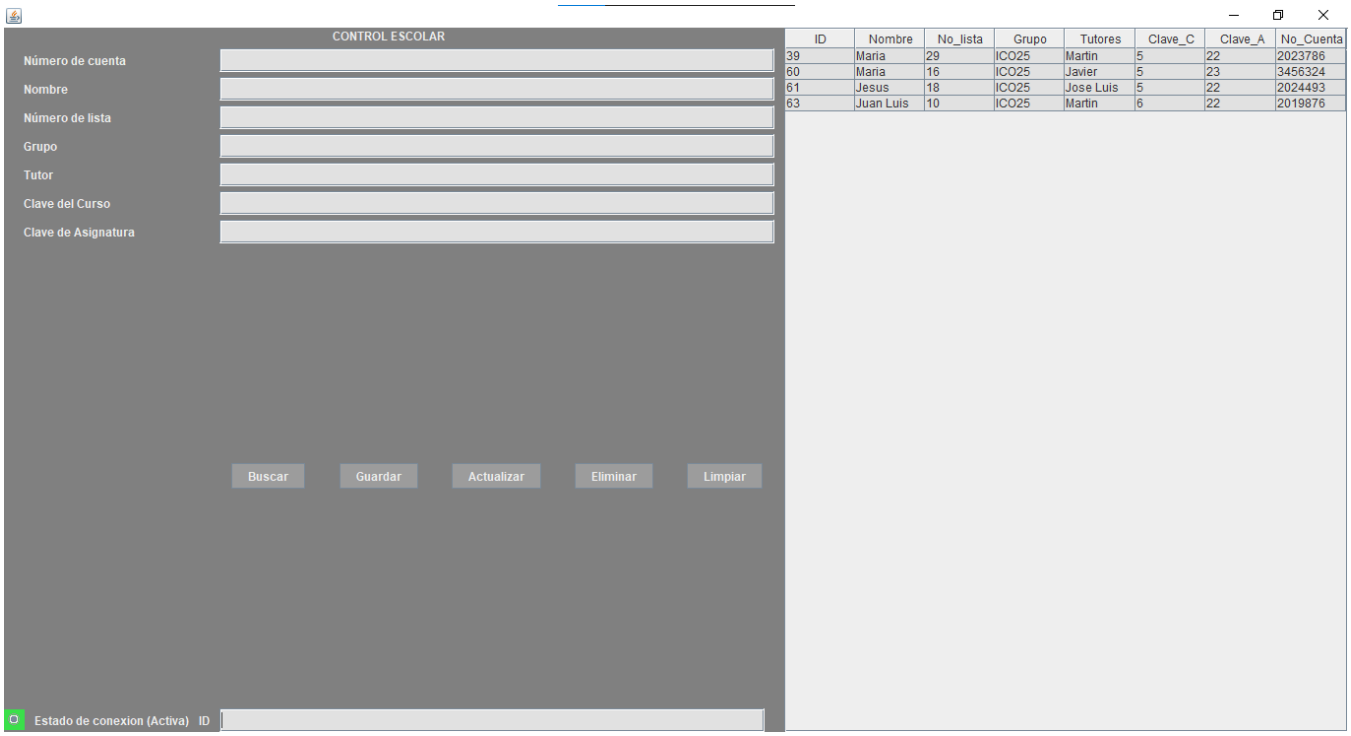


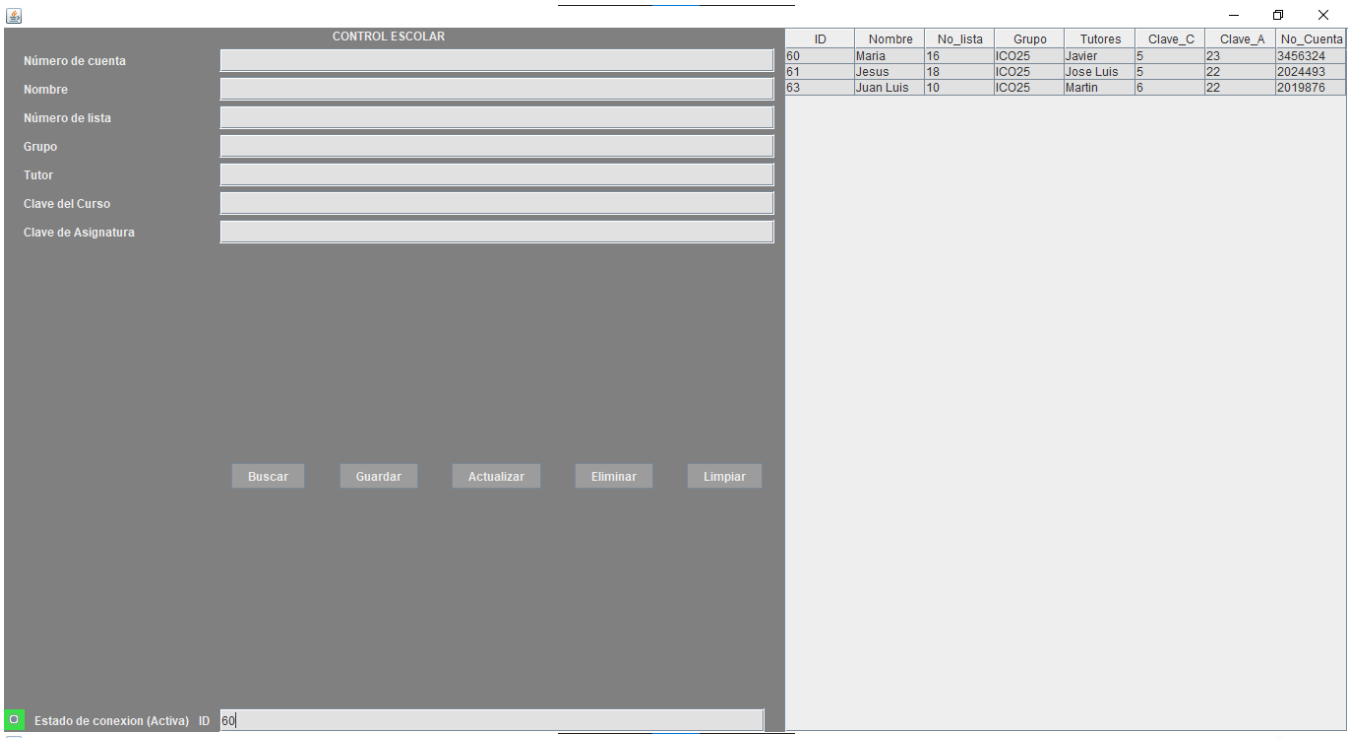
Actualizar registro:

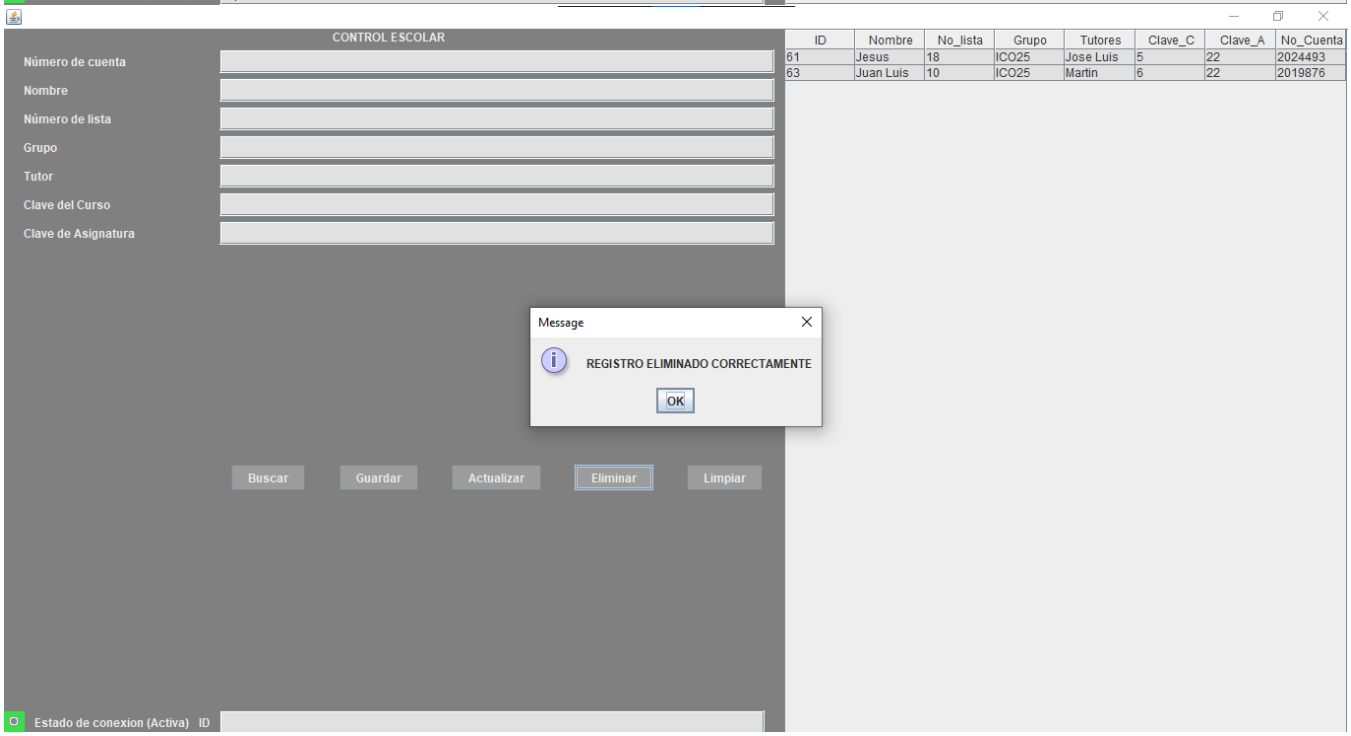




Eliminar registro:


CONTROL ESCOLAR
Número de cuenta
Nombre
Número de lista
Grupo
Tutor
Clave del Curso
Clave de Asignatura
Buscar
Guardar
Actualizar
Eliminar
Limpiar
Estado de conexion (Activa)
ID
ID Nombre No_lista Grupo Tutores Clave_C Clave_A No_Cuenta
39 Maria 29 ICO25 Martin 5 22 2023786
60 Maria 16 ICO25 Javier 5 23 3456324
61 Jesus 18 ICO25 Jose Luis 5 22 2024493
63 Juan Luis 10 ICO25 Martin 6 22 2019876


CONTROL ESCOLAR
Número de cuenta
Nombre
Número de lista
Grupo
Tutor
Clave del Curso
Clave de Asignatura
Buscar
Guardar
Actualizar
Eliminar
Limpiar
Estado de conexion (Activa) ID
60
ID Nombre No_lista Grupo Tutores Clave_C Clave_A No_Cuenta
60 Maria 16 ICO25 Javier 5 23 3456324
61 Jesus 18 ICO25 Jose Luis 5 22 2024493
63 Juan Luis 10 ICO25 Martin 6 22 2019876


CONTROL ESCOLAR
Número de cuenta
Nombre
Número de lista
Grupo
Tutor
Clave del Curso
Clave de Asignatura
Message
REGISTRO ELIMINADO CORRECTAMENTE
OK
Buscar
Guardar
Actualizar
Eliminar
Limpiar
Estado de conexion (Activa) ID
ID Nombre No_lista Grupo Tutores Clave_C Clave_A No_Cuenta
61 Jesus 18 ICO25 Jose Luis 5 22 2024493
63 Juan Luis 10 ICO25 Martin 6 22 2019876

5.Conclusiones

Sin duda los sistemas CRUD aunque pueden parecer simples estructuras generan un excelente desempeño y estabilidad para el manejo adecuado de información.