Servidor: 127.0.0.1 » Base de datos: umg » Tabla: persona

Examinar | Estructura | SQL | Buscar | Insertar | Exportar | Importar

La selección actual no contiene una columna única. La edición de la grilla y los enlaces de copiado, eliminación y edici

Mostrando filas 0 - 1 (total de 2, La consulta tardó 0,0002 segundos.)

```
SELECT * FROM `persona`
```

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

Mostrar todo | Número de filas: 25 Filtrar filas: Buscar en esta tabla

Opciones extra

ID	Nombre	Apellido	Edad
1	Ana	LOPEZ	4
1	Leonel	Taracena	25

Mostrar todo | Número de filas: 25 Filtrar filas: Buscar en esta tabla

Operaciones sobre los resultados de la consulta

Imprimir | Copiar al portapapeles | Exportar | Mostrar gráfico | Crear vista

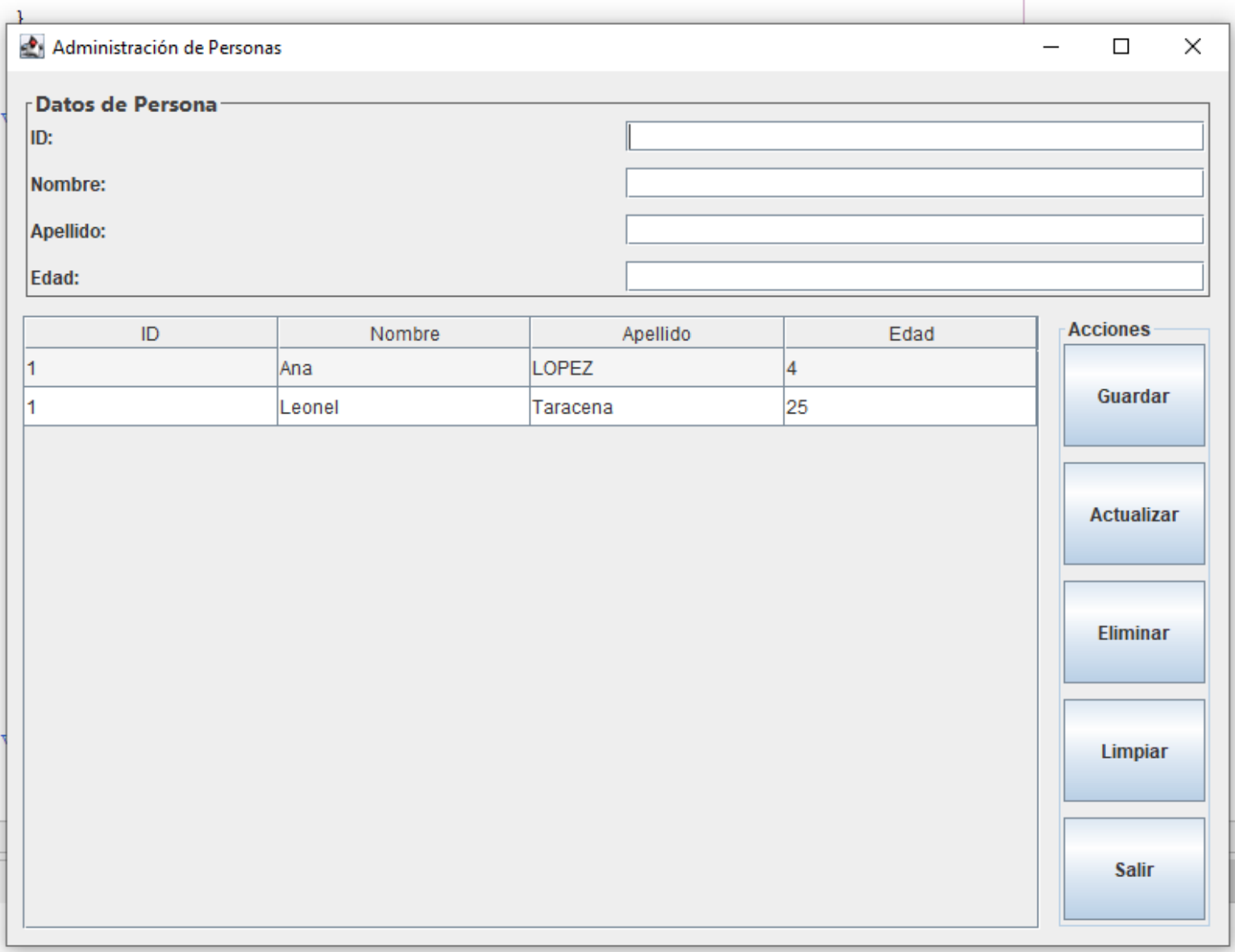
Guardar esta consulta en favoritos

Etiqueta: Permitir que todo usuario pueda acceder a este favorito

Guardar esta consulta en favoritos

```
        }
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage());
    }
}

priv
```




```
}

priv
```

(run)

CLASE MAIN

```
package prueba;


import javax.swing.SwingUtilities;


public class Prueba {
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new PersonasForm().setVisible(true));
    }
}
```

CLASE PERSONASFORM

```
package prueba;
import javax.swing.*;
import javax.swing.border.TitledBorder;
import javax.swing.table.DefaultTableCellRenderer;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.*;


public class PersonasForm extends JFrame {
    private JTextField txtId, txtNombre, txtApellido, txtEdad;
    private JButton btnGuardar, btnActualizar, btnEliminar, btnLimpiar, btnSalir;
    private JTable tabla;
    private DefaultTableModel modelo;
```

```
public PersonasForm() {
    setTitle("Administración de Personas");
    setSize(800, 600);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLocationRelativeTo(null);

    //Panel principal
    JPanel panelPrincipal = new JPanel(new BorderLayout(10, 10));
    panelPrincipal.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));

    //Formulario
    JPanel panelFormulario = new JPanel(new GridLayout(4, 2, 10, 10));
    panelFormulario.setBorder(BorderFactory.createTitledBorder(
            BorderFactory.createLineBorder(new Color(100, 100, 100), 1, true),
            "Datos de Persona",
            TitledBorder.LEFT,
            TitledBorder.TOP,
            new Font("Segoe UI", Font.BOLD, 14),
            Color.DARK_GRAY
    ));

    JLabel lblId = new JLabel("ID:");
    JLabel lblNombre = new JLabel("Nombre:");
    JLabel lblApellido = new JLabel("Apellido:");
    JLabel lblEdad = new JLabel("Edad:");

    txtId = new JTextField();
    txtNombre = new JTextField();
    txtApellido = new JTextField();
```

```
txtEdad = new JTextField();

panelFormulario.add(lblId); panelFormulario.add(txtId);
panelFormulario.add(lblNombre); panelFormulario.add(txtNombre);
panelFormulario.add(lblApellido); panelFormulario.add(txtApellido);
panelFormulario.add(lblEdad); panelFormulario.add(txtEdad);

// Botones Laterales
JPanel panelBotones = new JPanel(new GridLayout(5, 1, 10, 10));
panelBotones.setBorder(BorderFactory.createTitledBorder("Acciones"));

btnGuardar = new JButton("Guardar");
btnActualizar = new JButton("Actualizar");
btnEliminar = new JButton("Eliminar");
btnLimpiar = new JButton("Limpiar");
btnSalir = new JButton("Salir");

panelBotones.add(btnGuardar);
panelBotones.add(btnActualizar);
panelBotones.add(btnEliminar);
panelBotones.add(btnLimpiar);
panelBotones.add(btnSalir);

//Tabla
modelo = new DefaultTableModel(new String[]{"ID", "Nombre", "Apellido", "Edad"}, 0);
tabla = new JTable(modelo);
tabla.setRowHeight(25);

tabla.setDefaultRenderer(Object.class, new DefaultTableCellRenderer() {
```

```
        @Override
        public Component getTableCellRendererComponent(JTable table, Object value,
                                                       boolean isSelected, boolean hasFocus, int row, int column) {
            Component c = super.getTableCellRendererComponent(table, value, isSelected, hasFocus,
row, column);
            if (!isSelected) {
                c.setBackground(row % 2 == 0 ? new Color(245, 245, 245) : Color.WHITE);
            }
            return c;
        }
    });

    JScrollPane scroll = new JScrollPane(tabla);

    //Distribución
    panelPrincipal.add(panelFormulario, BorderLayout.NORTH);
    panelPrincipal.add(panelBotones, BorderLayout.EAST);
    panelPrincipal.add(scroll, BorderLayout.CENTER);

    add(panelPrincipal);

    //Eventos
    btnGuardar.addActionListener(e -> guardarPersona());
    btnActualizar.addActionListener(e -> actualizarPersona());
    btnEliminar.addActionListener(e -> eliminarPersona());
    btnLimpiar.addActionListener(e -> limpiarCampos());
    btnSalir.addActionListener(e -> dispose());

    tabla.addMouseListener(new MouseAdapter() {
```

```
        public void mouseClicked(MouseEvent e) {
            int fila = tabla.getSelectedRow();
            if (fila >= 0) {
                txtId.setText(tabla.getValueAt(fila, 0).toString());
                txtNombre.setText(tabla.getValueAt(fila, 1).toString());
                txtApellido.setText(tabla.getValueAt(fila, 2).toString());
                txtEdad.setText(tabla.getValueAt(fila, 3).toString());
            }
        }
    });

    cargarDatos();
}

//Métodos
private void guardarPersona() {
    try (Connection con = ConexionBD.getConnection()) {
        String sql = "INSERT INTO persona (ID, Nombre, Apellido, Edad) VALUES (?, ?, ?, ?)";
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setInt(1, Integer.parseInt(txtId.getText()));
        ps.setString(2, txtNombre.getText());
        ps.setString(3, txtApellido.getText());
        ps.setInt(4, Integer.parseInt(txtEdad.getText()));
        ps.executeUpdate();

        JOptionPane.showMessageDialog(this, "Persona agregada.");
        limpiarCampos();
        cargarDatos();
    } catch (Exception ex) {
```

```
        JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage());
    }
}

private void actualizarPersona() {
    try (Connection con = ConexionBD.getConnection()) {
        String sql = "UPDATE persona SET Nombre=?, Apellido=?, Edad=? WHERE ID=?";
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setString(1, txtNombre.getText());
        ps.setString(2, txtApellido.getText());
        ps.setInt(3, Integer.parseInt(txtEdad.getText()));
        ps.setInt(4, Integer.parseInt(txtId.getText()));
        int filas = ps.executeUpdate();

        if (filas > 0) {
            JOptionPane.showMessageDialog(this, "Persona actualizada.");
            limpiarCampos();
            cargarDatos();
        } else {
            JOptionPane.showMessageDialog(this, "No existe persona con ese ID.");
        }
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage());
    }
}

private void eliminarPersona() {
    try (Connection con = ConexionBD.getConnection()) {
        String sql = "DELETE FROM persona WHERE ID=?";
```

```
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setInt(1, Integer.parseInt(txtId.getText()));
        int filas = ps.executeUpdate();

        if (filas > 0) {
            JOptionPane.showMessageDialog(this, "Persona eliminada.");
            limpiarCampos();
            cargarDatos();
        } else {
            JOptionPane.showMessageDialog(this, "No existe persona con ese ID.");
        }
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage());
    }
}

private void cargarDatos() {
    modelo.setRowCount(0);
    try (Connection con = ConexionBD.getConnection()) {
        PreparedStatement ps = con.prepareStatement("SELECT * FROM persona");
        ResultSet rs = ps.executeQuery();
        while (rs.next()) {
            modelo.addRow(new Object[]{
                rs.getInt("ID"),
                rs.getString("Nombre"),
                rs.getString("Apellido"),
                rs.getInt("Edad")
            });
        }
```

```
        } catch (Exception ex) {
            JOptionPane.showMessageDialog(this, "Error al cargar: " + ex.getMessage());
        }
    }

    private void limpiarCampos() {
        txtId.setText("");
        txtNombre.setText("");
        txtApellido.setText("");
        txtEdad.setText("");
    }
}
```

CLASE CONEXIONDB

```
package prueba;

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

public class ConexionBD {
    private static final String URL =
"jdbc:mysql://localhost:3306/umg?useSSL=false&serverTimezone=UTC";
    private static final String USER = "root"; // cambia si tienes contraseña en XAMPP
    private static final String PASSWORD = "";

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL, USER, PASSWORD);
    }
}
```