

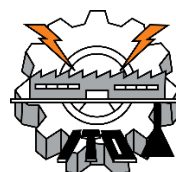


TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE OAXACA

ASIGNATURA: TOPICOS AVANZADOS DE PROGRAMACIÓN
CATEDRÁTICO: HERNANDEZ ABREGO ANAYANSI CRISTINA
ALUMNO: GARCÍA GARCÍA JOSÉ ÁNGEL
UNIDAD: 4

PRACTICA 4.3- Conexión a BD

OAXACA DE JÚAREZ, OAX, 30/MARZO/2020



Índice

BITACORA	3
CÓDIGO.....	4
CLASE MANIPULA DATOS	4
CLASE MODELO TABLA MEDIDAS	27
CLASE MODELO TABLA PERSONAS	28
CLASE MANEJO DATOS	30
CLASE CONEXIÓN.....	33
PRUEBAS	36




BITACORA

Se utilizó como bitácora a GitHub.

History for [Temas](#) / Unidad 4

Commits on Mar 29, 2020

Validacion ...

 ChepeAicrag12 committed 5 minutes ago

Se han validado los campos de texto.

Ahora podemos eliminar a una persona con registros de medidas



c09312a



Commits on Mar 28, 2020

AjusteGUI ...

 ChepeAicrag12 committed 16 hours ago

Agregue imagenes a la GUI, a las tablas les asigne su encabezado.



b032bb7



InteraccionConDB ...

 ChepeAicrag12 committed 20 hours ago

Se ha completado la la accion de los botones de cada panel. Tambien se reordenó la GUI para verse lo más similar al solicitado.




0d500e8



Commits on Mar 27, 2020

ValidacionFecha ...

 ChepeAicrag12 committed 2 days ago

Se valido el campo de Edad



93e2c67



PanelPersonas ...

 ChepeAicrag12 committed 2 days ago


Ya se puede insertar y eliminar correctamente.



8c11de0



InterfacesImplementadas ...

 ChepeAicrag12 committed 2 days ago

Se ha termiando el diseño de las interfaces. Falta agregar los eventos a los botones.




5a56557



Commits on Mar 26, 2020

InicioProyecto_tap09

 ChepeAicrag12 committed 3 days ago



dc57095



CÓDIGO

CLASE MANIPULA DATOS

```
package practica_09;

import java.awt.BorderLayout;
import java.awt.Container;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.FocusEvent;
import java.awt.event.FocusListener;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.sql.Date;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.List;
import javax.swing.*.*;
import javax.swing.border.TitledBorder;
import javax.swing.event.ChangeEvent;
import javax.swing.event.ChangeListener;
import javax.swing.table.JTableHeader

/**
 * @author Garcia Garcia Jose Angel
 */

public class ManipulaDatos extends JFrame {
```



```

// Elementos para leer los datos
private JTextField nombre;
private JSpinner fechaNac;
private JSpinner estatura;
private JRadioButton hombre;
private JRadioButton mujer;
private JTextField peso;
private JTextField cintura;
private JTextField cadera;
private JComboBox actividad;

// Elementos para el control de acciones
private JButton insertaPer; //Activa la accion para registrar una persona
private JButton limpiar; // Activa borrar valores de los campos
private JButton eliminar; // Activa Elimina registro persona
private JButton agregaMed; // Toma datos de person para medidas
private JButton insertaMed; // Inserta registro a la tabla mediciones
private JButton limpiaMed; // Limpia los campos de mediciones
private JButton eliminaMed; // Elimina registro de mediciones de la persona seleccionada

// Declara los atributos que permiten acceder y visualizar los datos de la BD
private ManejoDatos manejoDatos;
private ModeloTablaPersona modeloTablaPersona;
private JTable tablaPersona;
private ModeloTablaMedidas modeloTablaMedidas;
private JTable tablaMedidas;
private int idPerSel; // id de persona seleccionada
private JLabel nombrePerSel; // Despliega el nombre de persona seleccioanda

// Declaracion de los datos que permiten contorlar la ediccion y validacion de algunos datos
private final int MIN_EDAD = 20; // Edad minima

```



```

private final int MAX_EDAD = 65; // Edad maxima

private final double MIN_ESTATURA = 1.40; // Estatura minima

private final double MAX_ESTATURA = 1.95; // Estatura maxima

private final String DA_NOMBRE = "Da tu nombre"; // Titulo predefinido agregar medida en el campo
nombre

private final String DA_PES = "tu peso ?"; // Titulo predeagregaMEdido en el campo peso

private boolean validaNombre; // Valida el nombre

private boolean validaPeso; // Valida el valor de peso

private boolean validaCadera; // Valida el valor de la cadera

private boolean validaCintura; // Valida el valor de cintura

private boolean validaHombre; // Valida seleccion hombre

private boolean validaMujer; // Valida seleccion mujer

private String nombrePer;


public ManipulaDatos() {
    setSize(560, 500);

    Container base = getContentPane();
    base.setLayout(new BorderLayout());
    base.add(principal());
    setVisible(true);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setResizable(false);
    setLocationRelativeTo(null);
}


private JTabbedPane principal() {
    JTabbedPane panelPrincipal = new JTabbedPane();

    SpringLayout sM = new SpringLayout();

    SpringLayout s = new SpringLayout();

    JPanel panelPersona = new JPanel(s);

```



```

JPanel panelMedidas = new JPanel(sM);

JLabel titulo = new JLabel("Manipulacion de datos de personas");
titulo.setHorizontalAlignment(SwingConstants.CENTER);
titulo.setFont(new Font("Serif", Font.BOLD, 18));
titulo.setBorder(BorderFactory.createEmptyBorder(10, 0, 10, 0));
panelPersona.add(titulo);

s.putConstraint(SpringLayout.NORTH, titulo, 0, SpringLayout.NORTH, panelPersona);
s.putConstraint(SpringLayout.WEST, titulo, 70, SpringLayout.WEST, panelPersona);
s.putConstraint(SpringLayout.EAST, titulo, -70, SpringLayout.EAST, panelPersona);

JPanel fcp = formarCamposPersona();
panelPersona.add(fcp);

s.putConstraint(SpringLayout.NORTH, fcp, 10, SpringLayout.SOUTH, titulo);
s.putConstraint(SpringLayout.WEST, fcp, 80, SpringLayout.WEST, panelPersona);
s.putConstraint(SpringLayout.SOUTH, fcp, -270, SpringLayout.SOUTH, panelPersona);
s.putConstraint(SpringLayout.EAST, fcp, -70, SpringLayout.EAST, panelPersona);

JPanel frP = formarPanelResultados();
panelPersona.add(frP);

s.putConstraint(SpringLayout.NORTH, frP, 8, SpringLayout.SOUTH, fcp);
s.putConstraint(SpringLayout.WEST, frP, 5, SpringLayout.WEST, panelPersona);
s.putConstraint(SpringLayout.SOUTH, frP, 0, SpringLayout.SOUTH, panelPersona);
s.putConstraint(SpringLayout.EAST, frP, -5, SpringLayout.EAST, panelPersona);

JPanel fCM = formarCamposMedidas();
JPanel fPM = formarPanelMedidas();
panelMedidas.add(fCM);

sM.putConstraint(SpringLayout.NORTH, fCM, 0, SpringLayout.NORTH, panelMedidas);
sM.putConstraint(SpringLayout.WEST, fCM, 0, SpringLayout.WEST, panelMedidas);
sM.putConstraint(SpringLayout.EAST, fCM, 0, SpringLayout.EAST, panelMedidas);
sM.putConstraint(SpringLayout.SOUTH, fCM, -250, SpringLayout.SOUTH, panelMedidas);
panelMedidas.add(fPM);

```



```

sM.putConstraint(SpringLayout.NORTH, fPM, 5, SpringLayout.SOUTH, fCM);
sM.putConstraint(SpringLayout.WEST, fPM, 0, SpringLayout.WEST, panelMedidas);
sM.putConstraint(SpringLayout.EAST, fPM, 0, SpringLayout.EAST, panelMedidas);
sM.putConstraint(SpringLayout.SOUTH, fPM, 0, SpringLayout.SOUTH, panelMedidas);
panelPrincipal.add(panelPersona);
panelPrincipal.add(panelMedidas);
panelPrincipal.setTitleAt(0, "Personas");
panelPrincipal.setTitleAt(1, "Medidas");
return panelPrincipal;
}

```

```

private JPanel formarCamposPersona() {
    SpringLayout s = new SpringLayout();
    JPanel p = new JPanel(s);
    AdmoEventTeclado adTec = new AdmoEventTeclado();
    AdmoAccion ad = new AdmoAccion();
    p.setBorder(new TitledBorder("Proporciona los datos"));
    JLabel etqName = new JLabel("Nombre:");
    nombre = new JTextField(DA_NOMBRE, 15);
    nombre.setActionCommand("nombre");
    nombre.addKeyListener(adTec);
    nombre.addActionListener(ad);
    nombre.addMouseListener(new AdmoMouse());
    nombre.addFocusListener(new AdmoFocus());
    JLabel etqFecha = new JLabel("Fecha Nacimiento : ");
    Date today = new Date(1999, 00, 01);
    SpinnerModel sp = new SpinnerDateModel(today, null, null, Calendar.MONTH);
    fechaNac = new JSpinner(sp);
    JSpinner.DateEditor editor = new JSpinner.DateEditor(fechaNac, "dd/MM/yy ");
}

```




```

fechaNac.setEditor(editor);

JLabel etqAños = new JLabel("Años: " + 20);

fechaNac.addChangeListener(new ChangeListener() {

    @Override

    public void stateChanged(ChangeEvent ce) {

        int edad = calcular((java.util.Date) fechaNac.getValue());

        if (edad >= MIN_EDAD && edad <= MAX_EDAD) {

            etqAños.setText(" Años: " + edad);

        } else {

            fechaNac.setValue(new Date(1999, 00, 01));

        }

    }

});

public int calcular(java.util.Date fechaNac) {

    Date today = new Date(Calendar.getInstance().getTimeInMillis());

    int years = today.getYear() - fechaNac.getYear();

    int months = today.getMonth() - fechaNac.getMonth();

    int days = today.getDay() - fechaNac.getDay();

    if (months < 0 || (months == 0 && days < 0)) {

        years--;

    }

    return years;

}

});

JLabel etqSexo = new JLabel("Sexo : ");

hombre = new JRadioButton("Hombre");

hombre.setActionCommand("hombre");

hombre.addActionListener(ad);

mujer = new JRadioButton("Mujer");

```



```

mujer.setActionCommand("mujer");
mujer.addActionListener(ad);
p.add(etqName);
s.putConstraint(SpringLayout.NORTH, etqName, 12, SpringLayout.NORTH, p);
s.putConstraint(SpringLayout.WEST, etqName, 12, SpringLayout.WEST, p);
p.add(nombre);
s.putConstraint(SpringLayout.NORTH, nombre, 12, SpringLayout.NORTH, p);
s.putConstraint(SpringLayout.WEST, nombre, 75, SpringLayout.EAST, etqName);
p.add(etqFecha);
s.putConstraint(SpringLayout.NORTH, etqFecha, 12, SpringLayout.SOUTH, etqName);
s.putConstraint(SpringLayout.WEST, etqFecha, 12, SpringLayout.WEST, p);
p.add(fechaNac);
s.putConstraint(SpringLayout.NORTH, fechaNac, 12, SpringLayout.SOUTH, etqName);
s.putConstraint(SpringLayout.WEST, fechaNac, 12, SpringLayout.EAST, etqFecha);
p.add(etqAños);
s.putConstraint(SpringLayout.NORTH, etqAños, 12, SpringLayout.SOUTH, etqName);
s.putConstraint(SpringLayout.WEST, etqAños, 12, SpringLayout.EAST, fechaNac);
p.add(etqSexo);
s.putConstraint(SpringLayout.NORTH, etqSexo, 12, SpringLayout.SOUTH, etqFecha);
s.putConstraint(SpringLayout.WEST, etqSexo, 12, SpringLayout.WEST, p);
p.add(hombre);
s.putConstraint(SpringLayout.NORTH, hombre, 7, SpringLayout.SOUTH, fechaNac);
s.putConstraint(SpringLayout.WEST, hombre, 12, SpringLayout.EAST, etqSexo);
p.add(mujer);
s.putConstraint(SpringLayout.NORTH, mujer, 7, SpringLayout.SOUTH, fechaNac);
s.putConstraint(SpringLayout.WEST, mujer, 12, SpringLayout.EAST, hombre);
return p;
}

```



```

private JPanel formarPanelResultados() {
    manejoDatos = new ManejoDatos();
    modeloTablaPersona = new ModeloTablaPersona();
    cargarDatosPersona();
    tablaPersona = new JTable(modeloTablaPersona);
    JTableHeader hTab = tablaPersona.getTableHeader();
    SpringLayout s = new SpringLayout();
    JPanel p = new JPanel(s);
    JPanel p1 = new JPanel(new FlowLayout(FlowLayout.LEFT, 5, 0));
    AdmoAccion action = new AdmoAccion();
    insertaPer = new JButton("Agregar registro");
    insertaPer.setActionCommand("bInsertar");
    insertaPer.addActionListener(action);
    limpiar = new JButton("Inicia valores");
    limpiar.setActionCommand("bLimpiar");
    limpiar.addActionListener(action);
    eliminar = new JButton("Eliminar Reg. Sel.");
    eliminar.setActionCommand("bEliminar");
    eliminar.addActionListener(action);
    agregaMed = new JButton("Agregar Medidas");
    agregaMed.setActionCommand("bAgregaMed");
    agregaMed.addActionListener(action);
    p1.add(insertaPer);
    p1.add(limpiar);
    p1.add(eliminar);
    p1.add(agregaMed);
    p.add(p1);
    s.putConstraint(SpringLayout.NORTH, p1, 0, SpringLayout.NORTH, p);
    s.putConstraint(SpringLayout.WEST, p1, 9, SpringLayout.WEST, p);
}

```



```

s.putConstraint(SpringLayout.EAST, p1, -9, SpringLayout.EAST, p);
p.add(hTab);
s.putConstraint(SpringLayout.NORTH, hTab, 5, SpringLayout.SOUTH, p1);
s.putConstraint(SpringLayout.WEST, hTab, 10, SpringLayout.WEST, p);
s.putConstraint(SpringLayout.EAST, hTab, -10, SpringLayout.EAST, p);
p.add(tablaPersona);
s.putConstraint(SpringLayout.NORTH, tablaPersona, 0, SpringLayout.SOUTH, hTab);
s.putConstraint(SpringLayout.WEST, tablaPersona, 10, SpringLayout.WEST, p);
s.putConstraint(SpringLayout.EAST, tablaPersona, -10, SpringLayout.EAST, p);
s.putConstraint(SpringLayout.SOUTH, tablaPersona, 0, SpringLayout.SOUTH, p);
return p;
}

```

```

public JPanel formarCamposMedidas() {
    SpringLayout s = new SpringLayout();
    SpringLayout s2 = new SpringLayout();
    SpringLayout s3 = new SpringLayout();
    JPanel p = new JPanel(s); // Principal
    JPanel p1 = new JPanel(new BorderLayout());
    JLabel etq = new JLabel(String.format("%95s", "Manipulacion de datos de mediciones para :"));
    nombrePerSel = new JLabel(String.format("%90s", ""));
    p1.add(etq, BorderLayout.NORTH);
    p1.add(nombrePerSel, BorderLayout.SOUTH);
    JPanel p2 = new JPanel(s2);
    AdmoEventTeclado k = new AdmoEventTeclado();
    AdmoAccion ad = new AdmoAccion();
    AdmoFocus f = new AdmoFocus();
    AdmoMouse m = new AdmoMouse();
    p2.setBorder(new TitledBorder("Proporciona los datos"));
}

```



```

JLabel etqEst = new JLabel("Estatura (mts):");

estatura = new JSpinner(new SpinnerNumberModel(MIN_ESTATURA, MIN_ESTATURA, MAX_ESTATURA,
0.01));

JLabel etqPeso = new JLabel("Peso (kgs):");

peso = new JTextField(3);

peso.setActionCommand("peso");

peso.addKeyListener(k);

peso.addActionListener(ad);

peso.addFocusListener(f);

peso.addMouseListener(m);

JLabel etqCin = new JLabel("Cintura (cms):");

cintura = new JTextField(3);

cintura.setActionCommand("cintura");

cintura.addKeyListener(k);

cintura.addActionListener(ad);

cintura.addFocusListener(f);

cintura.addMouseListener(m);

JLabel etqCad = new JLabel("Cadera (cms):");

cadera = new JTextField(3);

cadera.setActionCommand("cadera");

cadera.addKeyListener(k);

cadera.addActionListener(ad);

cadera.addFocusListener(f);

cadera.addMouseListener(m);

JPanel p3 = new JPanel(s3);

p3.setBorder(new TitledBorder("Selecciona la actividad"));

List<Object[]> ob = manejoDatos.conexionConsultaActividad("SELECT * FROM CHEPE.TIPOACTIVIDAD");

String[] objcom = new String[ob.size()];

String[] desActi = new String[ob.size()];

for (int i = 0; i < ob.size(); i++) {

```



```

    objCom[i] = (String) ob.get(i)[1];
    desActi[i] = (String) ob.get(i)[2];
}

actividad = new JComboBox(objCom);

p2.add(etqEst);

s2.putConstraint(SpringLayout.NORTH, etqEst, 12, SpringLayout.NORTH, p2);
s2.putConstraint(SpringLayout.WEST, etqEst, 12, SpringLayout.WEST, p2);

p2.add(estatura);

s2.putConstraint(SpringLayout.NORTH, estatura, 10, SpringLayout.NORTH, p2);
s2.putConstraint(SpringLayout.WEST, estatura, 8, SpringLayout.EAST, etqEst);

p2.add(etqPeso);

s2.putConstraint(SpringLayout.NORTH, etqPeso, 12, SpringLayout.SOUTH, etqEst);
s2.putConstraint(SpringLayout.WEST, etqPeso, 12, SpringLayout.WEST, p2);

p2.add(peso);

s2.putConstraint(SpringLayout.NORTH, peso, 7, SpringLayout.SOUTH, estatura);
s2.putConstraint(SpringLayout.WEST, peso, 28, SpringLayout.EAST, etqPeso);

p2.add(etqCin);

s2.putConstraint(SpringLayout.NORTH, etqCin, 12, SpringLayout.SOUTH, etqPeso);
s2.putConstraint(SpringLayout.WEST, etqCin, 12, SpringLayout.WEST, p2);

p2.add(cintura);

s2.putConstraint(SpringLayout.NORTH, cintura, 9, SpringLayout.SOUTH, peso);
s2.putConstraint(SpringLayout.WEST, cintura, 12, SpringLayout.EAST, etqCin);

p2.add(etqCad);

s2.putConstraint(SpringLayout.NORTH, etqCad, 12, SpringLayout.SOUTH, etqCin);
s2.putConstraint(SpringLayout.WEST, etqCad, 12, SpringLayout.WEST, p2);

p2.add(cadera);

s2.putConstraint(SpringLayout.NORTH, cadera, 9, SpringLayout.SOUTH, cintura);
s2.putConstraint(SpringLayout.WEST, cadera, 12, SpringLayout.EAST, etqCad);

p3.add(actividad);

```



```

s3.putConstraint(SpringLayout.NORTH, actividad, 5, SpringLayout.NORTH, p3);
s3.putConstraint(SpringLayout.SOUTH, actividad, -15, SpringLayout.SOUTH, p3);
s3.putConstraint(SpringLayout.WEST, actividad, 50, SpringLayout.WEST, p3);
s3.putConstraint(SpringLayout.EAST, actividad, -50, SpringLayout.EAST, p3);
JPanel p4 = new JPanel(new BorderLayout());
p4.setBorder(BorderFactory.createTitledBorder(""));
JLabel etqActividad = new JLabel(desActi[0]);
JLabel img = new JLabel(imgActi(0));
p4.add(img, BorderLayout.CENTER);
p4.updateUI();
actividad.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent ae) {
        int pos = actividad.getSelectedIndex();
        etqActividad.setText(desActi[pos]);
        p4.removeAll();
        p4.add(etqActividad, BorderLayout.NORTH);
        p4.add(new JLabel(imgActi(pos)), BorderLayout.CENTER);
        p4.updateUI();
    }
});
p4.add(etqActividad, BorderLayout.NORTH);
p.add(p1);
s.putConstraint(SpringLayout.NORTH, p1, 0, SpringLayout.NORTH, p);
s.putConstraint(SpringLayout.WEST, p1, 0, SpringLayout.WEST, p);
s.putConstraint(SpringLayout.EAST, p1, 0, SpringLayout.EAST, p);
p.add(p2);
s.putConstraint(SpringLayout.NORTH, p2, 10, SpringLayout.SOUTH, p1);

```



```

s.putConstraint(SpringLayout.WEST, p2, 30, SpringLayout.WEST, p);
s.putConstraint(SpringLayout.EAST, p2, -340, SpringLayout.EAST, p);
s.putConstraint(SpringLayout.SOUTH, p2, 0, SpringLayout.SOUTH, p);
p.add(p3);
s.putConstraint(SpringLayout.NORTH, p3, 10, SpringLayout.SOUTH, p1);
s.putConstraint(SpringLayout.WEST, p3, 15, SpringLayout.EAST, p2);
s.putConstraint(SpringLayout.EAST, p3, -40, SpringLayout.EAST, p);
s.putConstraint(SpringLayout.SOUTH, p3, -90, SpringLayout.SOUTH, p);
p.add(p4);
s.putConstraint(SpringLayout.NORTH, p4, 1, SpringLayout.SOUTH, p3);
s.putConstraint(SpringLayout.WEST, p4, 15, SpringLayout.EAST, p2);
s.putConstraint(SpringLayout.EAST, p4, -40, SpringLayout.EAST, p);
s.putConstraint(SpringLayout.SOUTH, p4, 0, SpringLayout.SOUTH, p);
return p;
}

```

```

private ImagenIcon imgActi(int pos) {
    String ruta = "/imagenes/";
    String[] name = {"sedentarismo.jpg", "lige_activa.jpeg", "mode_activa.jpg", "muy_activa.jpg"};
    ImagenIcon icon_2 = new ImagenIcon(getClass().getResource(ruta + name[pos]));
    ImagenIcon icon2 = new ImagenIcon(icon_2.getImage().getScaledInstance(100, 70, Image.SCALE_DEFAULT));
    return icon2;
}

```

```

private JPanel formarPanelMedidas() {
    SpringLayout s = new SpringLayout();
    JPanel p = new JPanel(s);
    JPanel p1 = new JPanel(new FlowLayout(FlowLayout.CENTER, 10, 0));
    insertaMed = new JButton("Agregar Registro");
}

```




```

insertaMed.setActionCommand("bInsertarMed");
insertaMed.addActionListener(new AdmoAccion());
limpiaMed = new JButton("Inicia valores");
limpiaMed.setActionCommand("bLimpMed");
limpiaMed.addActionListener(new AdmoAccion());
eliminaMed = new JButton("Eliminar Reg. Seleccionado");
eliminaMed.setActionCommand("bEliminarMed");
eliminaMed.addActionListener(new AdmoAccion());
modeloTablaMedidas = new ModeloTablaMedidas();
tablaMedidas = new JTable(modeloTablaMedidas);
JTableHeader hTab = tablaMedidas.getTableHeader();
p1.add(insertaMed);
p1.add(limpiaMed);
p1.add(eliminaMed);
p.add(p1);
s.putConstraint(SpringLayout.NORTH, p1, 0, SpringLayout.NORTH, p);
s.putConstraint(SpringLayout.WEST, p1, 10, SpringLayout.WEST, p);
s.putConstraint(SpringLayout.EAST, p1, -10, SpringLayout.EAST, p);
p.add(hTab);
s.putConstraint(SpringLayout.NORTH, hTab, 5, SpringLayout.SOUTH, p1);
s.putConstraint(SpringLayout.WEST, hTab, 10, SpringLayout.WEST, p);
s.putConstraint(SpringLayout.EAST, hTab, -10, SpringLayout.EAST, p);
p.add(tablaMedidas);
s.putConstraint(SpringLayout.NORTH, tablaMedidas, 0, SpringLayout.SOUTH, hTab);
s.putConstraint(SpringLayout.WEST, tablaMedidas, 10, SpringLayout.WEST, p);
s.putConstraint(SpringLayout.EAST, tablaMedidas, -10, SpringLayout.EAST, p);
s.putConstraint(SpringLayout.SOUTH, tablaMedidas, 0, SpringLayout.SOUTH, p);
return p;
}

```



```

private class AdmoMouse implements MouseListener {

    @Override
    public void mouseClicked(MouseEvent me) {
        JTextField txt = (JTextField) me.getSource();
        if (txt.getCaretPosition() == 0) {
            txt.setText("");
        }
    }

    @Override
    public void mousePressed(MouseEvent me) {

    }

    @Override
    public void mouseReleased(MouseEvent me) {
        JTextField txt = (JTextField) me.getSource();
        if (txt.getSelectionStart() == 0 && txt.getSelectionEnd() == txt.getText().length()) {
            if (txt.getCaretPosition() == 0) {
                mouseClicked(me);
            }
        }
    }

    @Override
    public void mouseEntered(MouseEvent me) {

    }

    @Override

```



```

public void mouseExited(MouseEvent me) {
    JTextField txt = (JTextField) me.getSource();
    if (txt.getText().isEmpty()) {
        txt.requestFocus();
    }
}

}

private class AdmoEventTeclado extends KeyAdapter {

    @Override
    public void keyTyped(KeyEvent ke) {
        char k = ke.getKeyChar();
        JTextField productor = (JTextField) ke.getSource();
        if (productor == nombre) {
            if (nombre.getSelectedText() != null) {
                nombre.setText("");
            }
            if (!Character.isAlphabetic(k) && k != KeyEvent.VK_SPACE) {
                ke.consume();
            }
        }
        if (productor == peso || productor == cadera || productor == cintura) {
            if (!Character.isDigit(k)) {
                ke.consume();
            }
        }
    }
}

```



```
}
```

```
private class AdmoFocus implements FocusListener {
```

```
    @Override
```

```
    public void focusGained(FocusEvent fe) {
```

```
        Object o = fe.getSource();
```

```
        if (o instanceof JTextField) {
```

```
            JTextField txt = (JTextField) o;
```

```
            txt.setSelectionEnd(txt.getText().length());
```

```
            txt.setSelectionStart(0);
```

```
        }
```

```
    }
```

```
    @Override
```

```
    public void focusLost(FocusEvent fe) {
```

```
    }
```

```
}
```

```
private class AdmoAccion implements ActionListener {
```

```
    // Clase usada para administracion de acciones
```

```
    @Override
```

```
    public void actionPerformed(ActionEvent ae) {
```

```
        String op = ae.getActionCommand();
```

```
        switch (op) {
```

```
            case "nombre":
```

```
                validaNombre = false;
```



```

if (!nombre.getText().isEmpty() && !nombre.getText().equals(DA_NOMBRE)) {
    validaNombre = true;
    fechaNac.requestFocus();
} else {
    nombre.requestFocus();
}

break;

case "peso":
    double vPeso = 0;
    validaPeso = false;
    if (!peso.getText().isEmpty() && !peso.getText().equals(DA_PES)) {
        vPeso = Double.parseDouble(peso.getText());
        if (vPeso <= 120 && vPeso >= 40) {
            validaPeso = true;
            cintura.requestFocus();
        } else {
            peso.requestFocus();
        }
    }

    break;

case "cintura":
    validaCintura = false;
    int valCintura = 0;
    if (!cintura.getText().isEmpty()) {
        valCintura = Integer.parseInt(cintura.getText());
        if (valCintura <= 150 && valCintura >= 40) {
            validaCintura = true;
            cadera.requestFocus();
        } else {

```



```

        cintura.requestFocus();
    }

}

break;
case "cadera":
    validaCadera = false;
    int valCadera = 0;
    if (!cadera.getText().isEmpty()) {
        valCadera = Integer.parseInt(cadera.getText());
        if (valCadera <= 150 && valCadera >= 40) {
            validaCadera = true;
            agregaMed.requestFocus();
        } else {
            cadera.requestFocus();
        }
    }
    break;
case "hombre":
    validaHombre = false;
    if (hombre.isSelected()) {
        validaHombre = true;
        mujer.setSelected(false);
    }
    break;
case "mujer":
    validaMujer = false;
    if (mujer.isSelected()) {
        validaMujer = true;

```



```

        hombre.setSelected(false);
    }
    break;
case "bInsertar":
    if (validaNombre) {
        SimpleDateFormat ff = new SimpleDateFormat("YYYY-MM-dd");
        String vSexo = "";
        if (hombre.isSelected()) {
            vSexo = "" + hombre.getText().charAt(0);
        } else {
            vSexo = "" + mujer.getText().charAt(0);
        }
        String datos = "INSERT INTO CHEPE.PERSONA" + "(nombre,fechanac,sexo) values "
            + "(" + nombre.getText() + "," + ff.format(fechaNac.getValue()) + "," + vSexo + ")";
        manejoDatos.actualizaDatos(datos);
        cargarDatosPersona();
        tablaPersona.updateUI();
    }
    break;
case "bEliminar":
    String de = "";
    int rs = tablaPersona.getSelectedRow();
    if (rs >= 0) {
        idPerSel = Integer.parseInt((String) tablaPersona.getValueAt(rs, 0));
        nombrePer = (String) tablaPersona.getValueAt(rs, 1);
        int confirmado = JOptionPane.showConfirmDialog(null, "Eliminas el registro de " + nombrePer +
"?");
        if (confirmado == JOptionPane.OK_OPTION) {
            de = "DELETE FROM CHEPE.PERSONA WHERE IDPERSONA=" + idPerSel;

```



```

String seleccionMedPer = "SELECT * FROM CHEPE.MEDICIONES WHERE IDPERSONA=" +
idPerSel;

modeloTablaMedidas.setDatos(manejoDatos.conexionConsultaMediciones(seleccionMedPer));

for (int i = 0; i < tablaMedidas.getRowCount(); i++) {

    int idMed = Integer.parseInt((String) tablaMedidas.getValueAt(i, 0));

    System.out.println("Eliminando medida " + idMed);

    String de2 = "DELETE FROM CHEPE.MEDICIONES WHERE IDMEDIDAS=" + idMed;

    manejoDatos.actualizaDatos(de2);

}

tablaMedidas.updateUI();

manejoDatos.actualizaDatos(de);

cargarDatosPersona();

tablaPersona.updateUI();

} else {

    tablaPersona.clearSelection();

}

}

break;

case "bEliminarMed":

    de = "";

    rs = tablaPersona.getSelectedRow();

    if (rs >= 0) {

        idPerSel = Integer.parseInt((String) tablaPersona.getValueAt(rs, 0));

        nombrePer = (String) tablaPersona.getValueAt(rs, 1);

        int idMed = Integer.parseInt((String) tablaMedidas.getValueAt(tablaMedidas.getSelectedRow(),

0));

        int confirmado = JOptionPane.showConfirmDialog(null, "Eliminas el registro de" + nombrePer +

"?");

        if (confirmado == JOptionPane.OK_OPTION) {

```




```

        de = "DELETE FROM CHEPE.MEDICIONES WHERE IDMEDIDAS=" + idMed;

        manejoDatos.actualizaDatos(de);

        cargarDatosMedidas(idPerSel);

        tablaMedidas.updateUI();

    } else {

        tablaMedidas.clearSelection();

    }

}

break;

case "bAgregaMed":

    rs = tablaPersona.getSelectedRow();

    if (rs >= 0) {

        idPerSel = Integer.parseInt((String) tablaPersona.getValueAt(rs, 0));

        String seleccionMedPer = "SELECT * FROM CHEPE.MEDICIONES WHERE IDPERSONA=" + idPerSel;

        modeloTablaMedidas.setDatos(manejoDatos.conexionConsultaMediciones(seleccionMedPer));

        tablaMedidas.updateUI();

        String nps = (String) modeloTablaPersona.getValueAt(rs, 1);

        nombrePerSel.setText(String.format("%90s", "----- " + nps + "-----"));

    }

    break;

case "bInsertarMed":

    if (tablaPersona.getSelectedRow() >= 0) {

        Calendar fecha = Calendar.getInstance();

        SimpleDateFormat ff = new SimpleDateFormat("YYYY-MM-dd");

        Long estaturaL = ((Double) estatura.getValue()).longValue() * 100;

        int idAct = actividad.getSelectedIndex() + 1;

        if (validaPeso && validaCintura && validaCadera) {

            double pesoE = Double.parseDouble(peso.getText());

```



```

        int cinturaE = Integer.parseInt(cintura.getText());

        int caderaE = Integer.parseInt(cadera.getText());

        System.out.println("Insercion de " + idPerSel + " Estatura " + estaurL);

        String medidas = "INSERT INTO CHEPE.MEDICIONES"

            + "(fecha,estatura,peso,cintura,cadera,idtipoact,idpersona) values('"

            + ff.format(fecha.getTime()) + "','" + estaurL + "','" + pesoE + "','" + cinturaE + "','" + caderaE

            + "','" + idAct + "','" + idPerSel + "')";

        manejoDatos.actualizaDatos(medidas);

        cargarDatosMedidas(idPerSel);

        tablaMedidas.updateUI();

    }

}

break;

case "bLimpiar":

    nombre.setText(DA_NOMBRE);

    fechaNac.setValue(new Date(99, 00, 01));

    hombre.setSelected(false);

    mujer.setSelected(false);

    break;

case "bLimpMed":

    peso.setText("");

    estatura.setValue(1.40);

    cadera.setText("");

    cintura.setText("");

    actividad.setSelectedIndex(0);

    break;    } } }

public void cargarDatosPersona() {

    String ConsultaPersona = "select * from CHEPE.Persona";

    modeloTablaPersona.setDatos(manejoDatos.conexionConsultaPersona(ConsultaPersona));

```



```

    }

    public void cargarDatosMedidas(int idPers) {

        String consultaMedidas = "SELECT * FROM CHEPE.MEDICIONES WHERE IDPERSONA=" + idPers;

        modeloTablaMedidas.setDatos(manejoDatos.conexionConsultaMediciones(consultaMedidas));

    }

}

```

CLASE MODELO TABLA MEDIDAS

```

package practica_09;

import java.util.List;
import javax.swing.table.AbstractTableModel;
import javax.swing.table.DefaultTableModel;

/**
 * @author Garcia Garcia Jose Angel
 */
public class ModeloTablaMedidas extends AbstractTableModel{

    private List<Object[]> dato;

    private String encabezado[] = new String[]{

        "IdMed","Fecha","Estaura(cms)","Peso(kg)","Cintura(cm)","Cadera(cm)","Tipo act"};

    private Class tipos[] = new Class[]{

        String.class,String.class,String.class,String.class,String.class,String.class,String.class};

    public void setDatos(List<Object[]> d) {

        dato = d;

    }

    @Override

```



```

public Class getColumnClass(int c){
    return tipos[c];
}

@Override
public int getRowCount() {
    try{
        return dato.size();
    }catch(Exception e){
        return 0;
    }
}

@Override
public int getColumnCount() {
    return encabezado.length;
}

@Override
public Object getValueAt(int r, int c) {
    return dato.get(r)[c];
}

@Override
public String getColumnName(int col){
    return encabezado[col];
}
}

```

CLASE MODELO TABLA PERSONAS

```

package practica_09;

import java.util.List;

import javax.swing.table.AbstractTableModel;

```



```

/**
 * @author Garcia Garcia Jose Angel
 */
public class ModeloTablaPersona extends AbstractTableModel{

    private List<Object[]> dato;

    private String encabezado[] = new String[]{

        "No.Iden","Nombre","Fecha Nac.," "Sexo"};

    private Class tipos[] = new Class[]{

        String.class,String.class,String.class,String.class};

    @Override

    public Class getColumnClass(int c){

        return tipos[c];

    }

    @Override

    public int getRowCount() {

        return dato.size();

    }

    @Override

    public int getColumnCount() {

        return tipos.length;

    }

    @Override

    public Object getValueAt(int r, int c) {

        return dato.get(r)[c];

    }

    @Override

    public String getColumnName(int col){

        return encabezado[col];

    }

```



```

public void setDatos(List<Object[]> d){
    dato = d;
}
}

```

CLASE MANEJO DATOS

```

package practica_09;

import java.sql.Connection;
import java.sql.Date;
import java.sql.ResultSet;
import java.sql.Statement;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;

/**
 * @author Garcia Garcia Jose Angel
 */
public class ManejoDatos {
    private Connection conexion ; // Acceso a conexion
    private Conexion crearConexion;// Crea conexion
    private final int CAMPOS_PERSONA = 4;
    private final int CAMPOS_ACTIVIDAD = 3;
    private final int CAMPOS_MEDICIONES = 8;
    public ManejoDatos(){
        try{
            String usuario = "chepe",contraseña = "chepe123";

            crearConexion =
            crearConexion.getConexion("jdbc:derby://localhost:1527/mediciones_personas",usuario,contraseña);

```



```

    conexion = crearConexion.getConnection();
} catch (Exception e) {
}
}

public List <Object[]> conexionConsultaPersona(String sql){
    // Regresa los registros de las personas en una lista
    List <Object []> datos = new ArrayList<Object[]>();
    DateFormat fecha = new SimpleDateFormat("yyyy-MM-dd");
    try {
        Statement ps = conexion.createStatement();
        ResultSet rs = ps.executeQuery(sql);
        while (rs.next()) {
            // Estructura del registro de persona pasado como cadena
            String[] dat = new String[CAMPOS_PERSONA];
            dat[0] = String.valueOf((Integer) rs.getInt(1));
            dat[1] = rs.getString(2);
            dat[2] = fecha.format((Date) rs.getDate(3));
            dat[3] = rs.getString(4);
            datos.add(dat);
        }
    } catch (Exception e) {
        System.err.println("Error al consultar personas " + e);
    }
    return datos;
}

```

```

public List <Object[]> conexionConsultaActividad(String sql){
    // Regresa los registros de actividad
    List <Object[]> datos = new ArrayList<Object[]>();

```



```

try {
    Statement ps = conexion.createStatement();
    ResultSet rs = ps.executeQuery(sql);
    while (rs.next()) {
        // Estructura del registro activiad
        String [] dat = new String[CAMPOS_ACTIVIDAD];
        dat[0] = String.valueOf(rs.getInt(1));
        dat[1] = rs.getString(2);
        dat[2] = rs.getString(3);
        datos.add(dat);
    }
} catch (Exception e) {
    System.err.println("Error al consultar actividades " + e);
}
return datos;
}

public List <Object[]> conexionConsultaMediciones(String sql){
    // Regresa los registros de actividad
    List <Object[]> datos = new ArrayList<Object[]>();
    DateFormat fecha = new SimpleDateFormat("yyyy-MM-dd");
    try {
        Statement ps = conexion.createStatement();
        ResultSet rs = ps.executeQuery(sql);
        while (rs.next()) {
            // Estructura del registro activiad
            String [] dat = new String[CAMPOS_MEDICIONES];
            dat[0] = String.valueOf((Integer)rs.getInt(1));
            dat[1] = fecha.format((Date) rs.getDate(2));
            dat[2] = String.valueOf((Integer) rs.getInt(3));

```




```

        dat[3] = String.valueOf((Double) rs.getDouble(4));
        dat[4] = String.valueOf((Integer) rs.getInt(5));
        dat[5] = String.valueOf((Integer) rs.getInt(6));
        dat[6] = String.valueOf((Integer) rs.getInt(7));
        datos.add(dat);
    }
} catch (Exception e) {
    System.err.println("Error al consultar mediciones " + e);
}
return datos;
}

public boolean actualizaDatos(String SQL) {
    // Inserta actualiza o elimina
    System.out.println(SQL);
    boolean res = false;
    try {
        java.sql.Statement st = conexion.createStatement();
        st.executeUpdate(SQL);
        res = true;
        System.out.println((SQL.contains("INSERT")) ? "Se insertó" : "Se eliminó");
    } catch (Exception e) {
        System.err.println("Error al INSERTAR / ACTUALIZAR\n");
    }
    return res;
}
}

```

CLASE CONEXIÓN

```

package practica_09;

import com.sun.istack.internal.logging.Logger;

```



```

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.logging.Level;

/**
 * @author Garcia Garcia Jose Angel
 */
public class Conexion {

    private static Connection coneccion = null; // Contenedora en el paquete SQL
    private static Conexion conexion = null; // instancia a utilizar
    private static int numConexiones = 0; // controla el numero de veces que sucedio

    private Conexion(String url, String usuario, String password){
        try {
            // Clase usada para una conexion con Derby
            Class.forName("org.apache.derby.jdbc.ClientDriver");

            // Para MySql : "com.mysql.jdbc.Driver"

            // Para Postgres : "org.postgresql.Driver"

            coneccion = DriverManager.getConnection(url,usuario, password);
        } catch (SQLException e) {
            java.util.logging.Logger.getLogger(Conexion.class.getName()).log(Level.SEVERE,null,e);
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(Conexion.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    public static Conexion getConexion(String url, String Usuario, String password){
        numConexiones++;

        if(conexion == null)

            return conexion = new Conexion(url, Usuario, password);

        return conexion;
    }
}

```



```
}  
  
public static Connection getConeccion(){  
    return coneccion;  
}  
  
public boolean CerrarConexion(){  
    try {  
        if (coneccion != null)  
            if(numConexiones == 1){  
                coneccion.close();  
                return true;  
            }else  
                numConexiones--;  
        return false;  
    } catch (SQLException e) {  
        System.err.println(" Error al tratar de cerrar la conexion " + e);  
    }  
    return false;  
}}
```



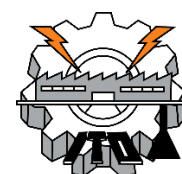
PRUEBAS

Así se muestra el panel de Personas al ejecutar.

No.Iden	Nombre	Fecha Nac.	Sexo
1	Miguel Angel Garcia	2020-01-01	H
2	Jose Angel Garcia	2020-01-01	H
3	Maria De Los Angele...	2020-01-01	M
4	Antonio Garcia	2019-06-13	H
5	Reyna Garcia	2019-08-13	M

Así se muestra el de Medidas.

IdMed	Fecha	Estaura(cm...	Peso(kg)	Cintura(cm)	Cadera(cm)	Tipo act
-------	-------	---------------	----------	-------------	------------	----------



Vamos a agregar una Persona.

Manipulacion de datos de personas

Proporciona los datos

Nombre:

Fecha Nacimiento: Años: 30

Sexo: ☒ Hombre ☐ Mujer

No.Iden	Nombre	Fecha Nac.	Sexo
1	Miguel Angel Garcia	2020-01-01	H
2	Jose Angel Garcia	2020-01-01	H
3	Maria De Los Angele...	2020-01-01	M
4	Antonio Garcia	2019-06-13	H
5	Reyna Garcia	2019-08-13	M
23	Juan Ruiz	1990-01-01	H

Ahora procederemos a agregar medidas.

**Manipulacion de datos de mediciones para :
----- Juan Ruiz-----**

Proporciona los datos

Estatura (mts):

Peso (kgs):

Cintura (cms):

Cadera (cms):

Selecciona la actividad

No hacer practicamente nada de ejercicio

IdMed	Fecha	Estaura(cm...	Peso(kg)	Cintura(cm)	Cadera(cm)	Tipo act
-------	-------	---------------	----------	-------------	------------	----------



Vamos a agregarle una medida.

Verificamos con la salida.

```
INSERT INTO CHEPE.PERSONA(nombre,fechanac,sexo) values ('Juan Ruiz','1990-01-01','H')
Se insertó
Insercion de 23 Estatura 100
INSERT INTO CHEPE.MEDICIONES (fecha,estatura,peso,cintura,cadera,idtipoact,idpersona) values ('2020-03-29',100,70.0,60,70,2,23)
Se insertó
```

También la verificamos en la base de datos.

#	IDMEDIDAS	FECHA	ESTATURA	PESO	CINTURA	CADERA	IDTIPOACT	IDPERSONA
1		2 2020-03-24		182	80.0	70	80	1
2		3 2020-03-24		180	67.0	75	76	2
3		4 2020-03-24		170	70.0	60	60	3
4		18 2020-03-28		100	12.0	12	12	1
5		19 2020-03-28		100	12.0	12	12	1
6		20 2020-03-28		100	12.5	12	12	1
7		21 2020-03-28		100	12.5	12	12	4
8		23 2020-03-28		100	78.88	70	70	3
9		25 2020-03-29		100	70.0	60	70	2

Ahora limpiaremos los datos de los campos en ambos paneles.



Procederemos a ver las medidas de otra persona, de nombre Miguel Angel Garcia.

Personas Medidas

Manipulacion de datos de mediciones para :
----- Miguel Angel Garcia -----

Proporciona los datos

Estatura (mts):

Peso (kgs):

Cintura (cms):

Cadera (cms):

Selecciona la actividad

SEDENTARIO

No hacer practicamente nada de ejercicio

Agregar Registro Inicia valores Eliminar Reg. Seleccionado

IdMed	Fecha	Estaura(cm...	Peso(kg)	Cintura(cm)	Cadera(cm)	Tipo act
18	2020-03-28	100	12.0	12	12	1
19	2020-03-28	100	12.0	12	12	1
20	2020-03-28	100	12.5	12	12	1
21	2020-03-28	100	12.5	12	12	4
23	2020-03-28	100	78.88	70	70	3

Lo que haremos será eliminar las ultimas 2 medidas de él, es decir, las de id 21 y 23.

Personas Medidas

Manipulacion de datos de mediciones para :
----- Miguel Angel Garcia -----

Proporciona los datos

Estatura (mts):

Peso (kgs):

Cintura (cms):

Cadera (cms):

Selecciona la actividad

SEDENTARIO

No hacer practicamente nada de ejercicio

Agregar Registro Inicia valores Eliminar Reg. Seleccionado

IdMed	Fecha	Estaura(cm...	Peso(kg)	Cintura(cm)	Cadera(cm)	Tipo act
18	2020-03-28	100	12.0	12	12	1
19	2020-03-28	100	12.0	12	12	1
20	2020-03-28	100	12.5	12	12	1

También podemos ver en la salida de consola que se eliminaron correctamente.

Output

Java DB Database Process x practica_09 (run) x

```
run:  
DELETE FROM CHEPE.MEDICIONES WHERE IDMEDIDAS=23  
Se eliminó  
DELETE FROM CHEPE.MEDICIONES WHERE IDMEDIDAS=21  
Se eliminó
```



Lo que haremos ahora es eliminar a una persona que no tenga medidas. Eliminaremos a la persona de nombre Reyna Garcia, en la siguiente imagen se muestra que no tiene registros de medida.

Personas Medidas

Manipulacion de datos de mediciones para :
----- Reyna Garcia-----

Proporciona los datos

Estatura (mts): 1.4

Peso (kgs):

Cintura (cms):

Cadera (cms):

Selecciona la actividad
SEDENTARIO

No hacer practicamente nada de ejercicio

Agregar Registro Inicia valores Eliminar Reg. Seleccionado

IdMed	Fecha	Estaura(cm...	Peso(kg)	Cintura(cm)	Cadera(cm)	Tipo act
-------	-------	---------------	----------	-------------	------------	----------

En panel personas, selecciona la opción de eliminar, se nos mostrará el siguiente cuadro de dialogo.

Seleccionar una Opción

? Eliminas el registro de Reyna Garcia?

Sí No Cancelar

Seleccionamos que si y veremos que se ha eliminado.

Personas Medidas

Manipulacion de datos de personas

Proporciona los datos

Nombre: Da tu nombre

Fecha Nacimiento: 01/01/99 Años: 20

Sexo: ☐ Hombre ☐ Mujer

Agregar registro Inicia valores Eliminar Reg. Sel. Agregar Medidas

No.Iden	Nombre	Fecha Nac.	Sexo
1	Miguel Angel Garcia	2020-01-01	H
2	Jose Angel Garcia	2020-01-01	H
3	Maria De Los Angele...	2020-01-01	M
4	Antonio Garcia	2019-06-13	H
23	Juan Ruiz	1990-01-01	H

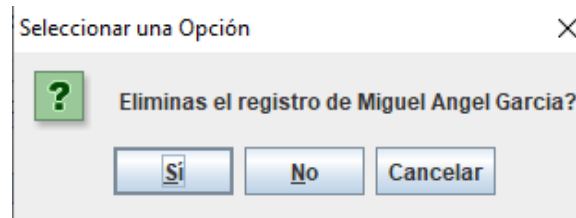
Y en la consola también lo podemos verificar.

```
DELETE FROM CHEPE.PERSONA WHERE IDPERSONA=5  
Se eliminó
```



Ahora procedemos a hacer el procedimiento de eliminar una persona que, si tenga registros, para ello vamos a eliminar a Miguel Angel Garcia, que ya vimos anteriormente que si tiene registros.

De igual forma se nos muestra el dialogo.



Revisamos que ha sido removido de la tabla.



No.Iden	Nombre	Fecha Nac.	Sexo
2	Jose Angel Garcia	2020-01-01	H
3	Maria De Los Angele...	2020-01-01	M
4	Antonio Garcia	2019-06-13	H
23	Juan Ruiz	1990-01-01	H

La salida en consola nos mostrará la eliminación de las medidas de esta persona, ya que, si eliminamos directamente, tendríamos un error. Lo que se hace, primero es eliminar todas sus medidas y posterior a ello proceder a borrarlo de la tabla persona.

```
Se eliminó
Eliminando medida 18
DELETE FROM CHEPE.MEDICIONES WHERE IDMEDIDAS=18
Se eliminó
Eliminando medida 19
DELETE FROM CHEPE.MEDICIONES WHERE IDMEDIDAS=19
Se eliminó
Eliminando medida 20
DELETE FROM CHEPE.MEDICIONES WHERE IDMEDIDAS=20
Se eliminó
DELETE FROM CHEPE.PERSONA WHERE IDPERSONA=1
Se eliminó
```

Ya hemos hecho pruebas de todas las acciones de los botones.




Ahora procederemos a ver los aspectos gráficos de la GUI.

Las imágenes que tiene cada actividad son la siguientes:

Selecciona la actividad

SEDENTARIO ▼

No hacer practicamente nada de ejercicio



Selecciona la actividad

LIGERAMENTE ACTIVAS ▼

Hacer ejercicio suave de 1 a 3 veces por semana



Selecciona la actividad

MODERADAMENTE ACTIVA ▼

Hacer deporte 3 a 5 veces por semana



Selecciona la actividad

MUY ACTIVAS ▼

Hacer deporte 6 a 7 días por semana



Ahora veremos los limites de los Spinner, el primero es correspondiente a la fecha.

Proporciona los datos

Nombre:

Fecha Nacimiento : Años: 20

Sexo : ☐ Hombre ☐ Mujer

Proporciona los datos

Nombre:

Fecha Nacimiento : Años: 65

Sexo : ☒ Hombre ☐ Mujer

Ahora los del Spinner de estatura.

Proporciona los datos

Estatura (mts):

Peso (kgs):

Cintura (cms):

Cadera (cms):

Proporciona los datos

Estatura (mts):

Peso (kgs):

Cintura (cms):

Cadera (cms):

También los campos de peso, cintura y cadera están validados.

El peso está en rangos de 40 – 120; La cintura en rengos de 40 – 150; La cadera en rangos de 40 – 150

La opción del sexo de igual forma está validad, para solo elegir un sexo.

Sexo : ☒ Hombre ☐ Mujer

Sexo : ☐ Hombre ☒ Mujer

