

INTRODUCCIÓN A BASE DE DATOS

Mantenimiento y consultas a una base de datos

CAPACIDAD EN PROCESO:

Desarrolla métodos para el mantenimiento y consulta de información en una base de datos.

Mantenimiento y consultas a una base de datos

Luego de haber logrado una conexión exitosa, ahora desarrollamos la clase

Mantenimiento con el siguiente atributo y constructor:

```
package controlador;

import java.sql.ResultSet;
import java.sql.Connection;
import java.sql.Statement;

public class Mantenimiento {
    protected Conexion conecta;

    public Mantenimiento() {
        conecta = null;
    }
}
```

Luego desarrollamos el método para registrar un nuevo empleado en la base de datos:

```
public void nuevo(Empleado e){
    String sql;

    sql = "insert into empleados (nombre,apellidos,dni,sueldo,area) "
        + "values ('"+e.getNombre()+"', '"+
            e.getApellidos()+"', '"+
            e.getDni()+"', '"+
            e.getSueldo()+"', '"+
            e.getArea()+"') ";

    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        st=con.createStatement();
        st.executeUpdate(sql);
        con.close();
        st.close();
    }catch(Exception ex){
        System.out.println("Error en el ingreso nuevo: "+ex.getMessage());
    }
}
```

Ahora desarrollamos el método para hacer una modificación en un empleado:

```
public void modifica(Empleado e){
    String sql;

    sql = "update empleados set nombre = '"+e.getNombre()+
        "',apellidos = '"+e.getApellidos()+
        "',dni = '"+e.getDni()+
        "',sueldo = '"+e.getSueldo()+
        "',area = '"+e.getArea()+
        "'where dni = '"+e.getDni()+"' ";

    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        st=con.createStatement();
        st.executeUpdate(sql);
        con.close();
        st.close();
    }catch(Exception ex){
        System.out.println("Error en la modificacion: "+ex.getMessage());
    }
}
```

Ahora desarrollamos el método para hacer una eliminación de un empleado:

```
public void elimina(String dni){
    String sql;

    sql = "delete from empleados where dni = '"+dni+"'";

    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        st=con.createStatement();
        st.executeUpdate(sql);
        con.close();
        st.close();
    }catch(Exception ex){
        System.out.println("Error en la eliminacion: "+ex.getMessage());
    }
}
```

Ahora desarrollamos el método para hacer una consulta de un empleado:

```
public Empleado consulta(String dni){
    String sql;
    sql = "select * from empleados where dni = '"+dni+"'";
    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        st=con.createStatement();
        ResultSet rs= st.executeQuery(sql);
        if (rs.next()){ // lo encontré
            return new Empleado(rs.getString(1),
                                rs.getString(2),
                                rs.getString(3),
                                rs.getDouble(4),
                                rs.getInt(5));
        }
        con.close(); st.close();
    }catch(Exception ex){
        System.out.println("Error en la eliminacion: "+ex.getMessage());
    }
    return null;
}
```

Finalmente, desarrollamos el método que retorna la información de toda la tabla de empleados:

```

public String lista(){
    String sql, resultado="";
    sql = "select * from empleados";
    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        ResultSet rs;
        st=con.createStatement();
        rs = st.executeQuery(sql);
        while (rs.next()){
            resultado += (rs.getString(1) +"\t"+
                           rs.getString(2)+"\t"+
                           rs.getString(3)+"\t"+
                           rs.getDouble(4)+"\t"+
                           rs.getInt(5)+"\n");
        }
        con.close(); st.close();
    }catch(Exception ex){
        System.out.println("Error en la eliminacion: "+ex.getMessage());
    }
    return resultado;
}

```

Ahora vamos a programar los botones del PanelEmpleados:

```

private void btnNuevoActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Mantenimiento man = new Mantenimiento();
    Empleado e = man.consulta(leeDNI());
    if (e!=null)
        mensaje("DNI repetido!");
    else{
        Empleado nuevo = new Empleado(leeNombre(),
                                         leeApellidos(),
                                         leeDNI(),
                                         leeSueldo(),
                                         leeArea());
        man.nuevo(nuevo);
        btnLista.doClick();
    }
}

```

```
private void btnConsultaActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    Mantenimiento man = new Mantenimiento();  
  
    Empleado e = man.consulta(leeDNI());  
    if (e==null)  
        mensaje("DNI no registrado!");  
    else{  
        txtNombres.setText(e.getNombre());  
        txtApellidos.setText(e.getApellidos());  
        txtDNI.setText(e.getDni());  
        txtSueldo.setText(e.getSueldo()+"");  
        cboArea.setSelectedIndex(e.getArea());  
    }  
}
```

```
private void btnModificaActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    Mantenimiento man = new Mantenimiento();  
    Empleado actual = man.consulta(leeDNI());  
    if(actual == null)  
        mensaje("DNI no registrado!");  
    else{  
        actual = new Empleado(leeNombre(),  
                                leeApellidos(),  
                                leeDNI(),  
                                leeSueldo(),  
                                leeArea());  
        man.modifica(actual);  
        btnLista.doClick();  
    }  
}
```

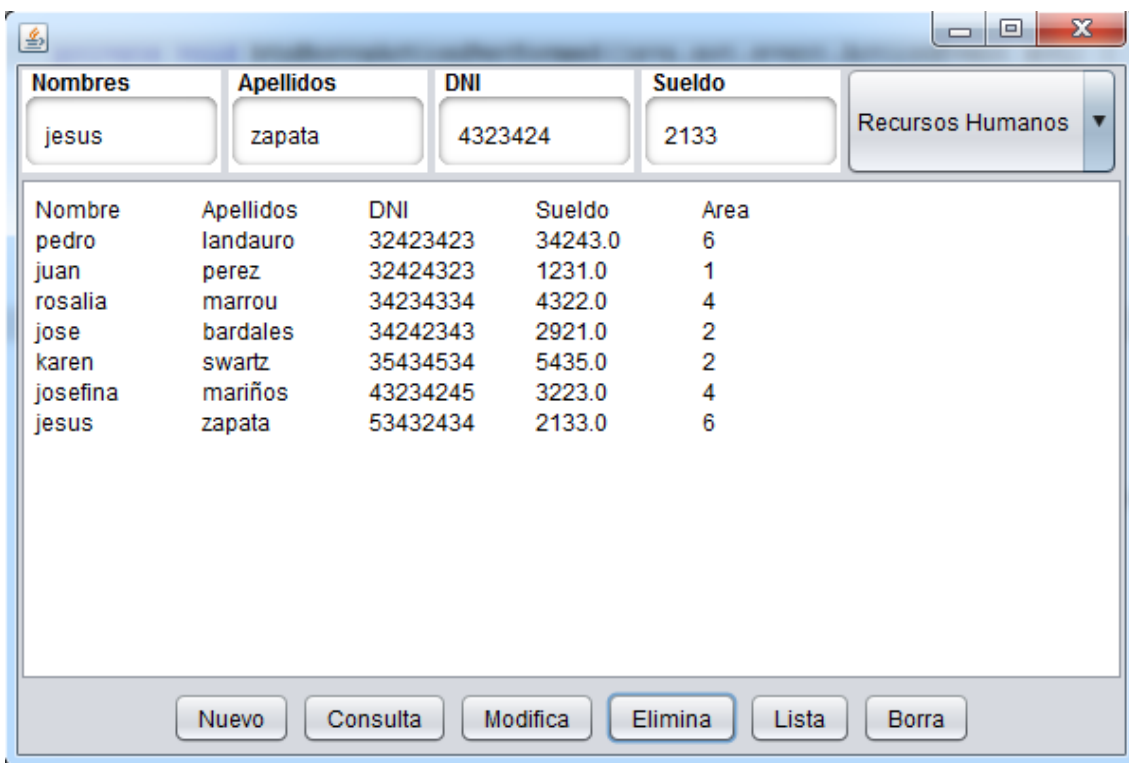
```
private void btnEliminaActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    Mantenimiento man = new Mantenimiento();  
    Empleado actual = man.consulta(leeDNI());  
    if(actual == null)  
        mensaje("DNI no registrado!");  
    else{  
        man.elimina(leeDNI());  
        btnLista.doClick();  
    }  
}
```

```
private void btnListaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Mantenimiento man = new Mantenimiento();
    txtSalida.setText("Nombre\tApellidos\tDNI\tSueldo\tArea\n");
    imprime(man.lista());
}
}
```

Métodos complementarios:

```
private String leeNombre(){ return txtNombres.getText();}
private String leeApellidos() { return txtApellidos.getText();}
private String leeDNI(){ return txtDNI.getText(); }
private double leeSueldo(){ return Double.parseDouble(txtSueldo.getText());}
private int leeArea(){ return cboArea.getSelectedIndex(); }
private void imprime(String s){
    txtSalida.append(s+"\n");
}
private void mensaje(String s){
    JOptionPane.showMessageDialog(this, s);
}
}
```

Ejecute su aplicación:



Nombre	Apellidos	DNI	Sueldo	Area
pedro	landauro	32423423	34243.0	6
juan	perez	32424323	1231.0	1
rosalia	marrou	34234334	4322.0	4
jose	bardales	34242343	2921.0	2
karen	swartz	35434534	5435.0	2
josefina	mariños	43234245	3223.0	4
jesus	zapata	53432434	2133.0	6

La información ya está registrada en la base de datos, puede cerrar su aplicación y volver a ejecutarlo para comprobarlo.

Añada un comboBox de reportes para mostrar la siguiente información:

- Empleados que pertenecen a un área seleccionada
- Empleados con un sueldo inferior al sueldo promedio
- Empleados en orden alfabetico por apellidos

SEMANA 14

GUIA DE LABORATORIO 14

Mantenimiento y consultas en una base de datos

Abra el proyecto de la semana anterior y desarrolle las clases que faltan: Mantenimiento y programación de los botones del PanelEmpleados.

En el paquete **controlador** diseñe la clase **Mantenimiento** con el siguiente atributo y constructor:

```
package controlador;

import java.sql.ResultSet;
import java.sql.Connection;
import java.sql.Statement;

public class Mantenimiento {
    protected Conexion conecta;

    public Mantenimiento() {
        conecta = null;
    }
}
```

Luego desarrollamos el método para registrar un nuevo empleado en la base de datos:


```

public void nuevo(Empleado e){
    String sql;

    sql = "insert into empleados (nombre,apellidos,dni,sueldo,area) "
        + "values ('"+e.getNombre()+"', '"+
            e.getApellidos()+"', '"+
            e.getDni()+"', '"+
            e.getSueldo()+"', '"+
            e.getArea()+"') ";

    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        st=con.createStatement();
        st.executeUpdate(sql);
        con.close();
        st.close();
    }catch(Exception ex){
        System.out.println("Error en el ingreso nuevo: "+ex.getMessage());
    }
}

```

Ahora desarrollamos el método para hacer una modificación en un empleado:

```

public void modifica(Empleado e){
    String sql;

    sql = "update empleados set nombre = '"+e.getNombre()+"'
        + "',apellidos = '"+e.getApellidos()+"'
        + "',dni = '"+e.getDni()+"'
        + "',sueldo = '"+e.getSueldo()+"'
        + "',area = '"+e.getArea()+"'
        + "'where dni = '"+e.getDni()+"'";

    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        st=con.createStatement();
        st.executeUpdate(sql);
        con.close();
        st.close();
    }catch(Exception ex){
        System.out.println("Error en la modificacion: "+ex.getMessage());
    }
}

```

Ahora desarrollamos el método para hacer una eliminación de un empleado:

```
public void elimina(String dni){
    String sql;

    sql = "delete from empleados where dni = '"+dni+"'";

    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        st=con.createStatement();
        st.executeUpdate(sql);
        con.close();
        st.close();
    }catch(Exception ex){
        System.out.println("Error en la eliminacion: "+ex.getMessage());
    }
}
```

Ahora desarrollamos el método para hacer una consulta de un empleado:

```
public Empleado consulta(String dni){
    String sql;
    sql = "select * from empleados where dni = '"+dni+"'";
    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        st=con.createStatement();
        ResultSet rs= st.executeQuery(sql);
        if (rs.next()){ // lo encontré
            return new Empleado(rs.getString(1),
                                rs.getString(2),
                                rs.getString(3),
                                rs.getDouble(4),
                                rs.getInt(5));
        }
        con.close(); st.close();
    }catch(Exception ex){
        System.out.println("Error en la eliminacion: "+ex.getMessage());
    }
    return null;
}
```

Finalmente, desarrollamos el método que retorna la información de toda la tabla de empleados:

```
public String lista(){
    String sql, resultado="";
    sql = "select * from empleados";
    try{
        conecta = new Conexion();
        Connection con = conecta.getConnection();
        Statement st;
        ResultSet rs;
        st=con.createStatement();
        rs = st.executeQuery(sql);
        while (rs.next()){
            resultado += (rs.getString(1) + "\t" +
                           rs.getString(2) + "\t" +
                           rs.getString(3) + "\t" +
                           rs.getDouble(4) + "\t" +
                           rs.getInt(5) + "\n");
        }
        con.close(); st.close();
    }catch(Exception ex){
        System.out.println("Error en la eliminacion: "+ex.getMessage());
    }
    return resultado;
}
```

Ahora vamos a programar los botones del PanelEmpleados:

```
private void btnNuevoActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Mantenimiento man = new Mantenimiento();
    Empleado e = man.consulta(leeDNI());
    if (e!=null)
        mensaje("DNI repetido!");
    else{
        Empleado nuevo = new Empleado(leeNombre(),
                                         leeApellidos(),
                                         leeDNI(),
                                         leeSueldo(),
                                         leeArea());
        man.nuevo(nuevo);
        btnLista.doClick();
    }
}
```

```
private void btnConsultaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Mantenimiento man = new Mantenimiento();

    Empleado e = man.consulta(leeDNI());
    if (e==null)
        mensaje("DNI no registrado!");
    else{
        txtNombres.setText(e.getNombre());
        txtApellidos.setText(e.getApellidos());
        txtDNI.setText(e.getDni());
        txtSueldo.setText(e.getSueldo()+"");
        cboArea.setSelectedIndex(e.getArea());
    }
}
```

```
private void btnModificaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Mantenimiento man = new Mantenimiento();
    Empleado actual = man.consulta(leeDNI());
    if(actual == null)
        mensaje("DNI no registrado!");
    else{
        actual = new Empleado(leeNombre(),
                                leeApellidos(),
                                leeDNI(),
                                leeSueldo(),
                                leeArea());
        man.modifica(actual);
        btnLista.doClick();
    }
}
```

```
private void btnEliminaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Mantenimiento man = new Mantenimiento();
    Empleado actual = man.consulta(leeDNI());
    if(actual == null)
        mensaje("DNI no registrado!");
    else{
        man.elimina(leeDNI());
        btnLista.doClick();
    }
}
```

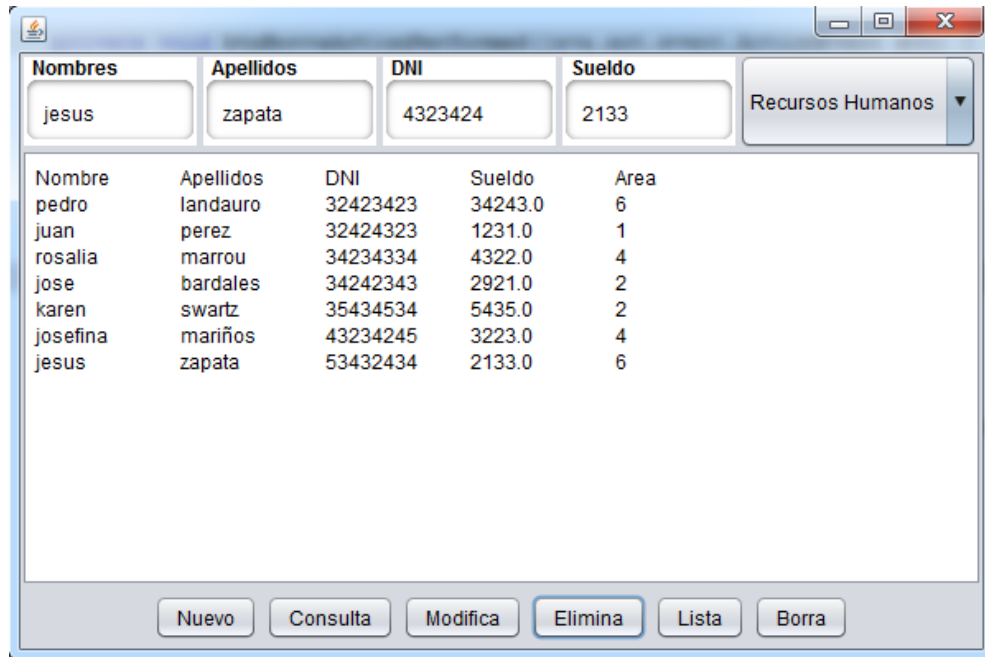
```
private void btnListaActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Mantenimiento man = new Mantenimiento();
    txtSalida.setText("Nombre\tApellidos\tDNI\tSueldo\tArea\n");
    imprime(man.lista());
}
```

Métodos complementarios:

```

private String leeNombre(){ return txtNombres.getText();}
private String leeApellidos() { return txtApellidos.getText();}
private String leeDNI(){ return txtDNI.getText(); }
private double leeSueldo(){ return Double.parseDouble(txtSueldo.getText());}
private int leeArea(){ return cboArea.getSelectedIndex(); }
private void imprime(String s){
    txtSalida.append(s+"\n");
}
private void mensaje(String s){
    JOptionPane.showMessageDialog(this, s);
}
    
```

Ejecute su aplicación:



Nombre	Apellidos	DNI	Sueldo	Area
pedro	landauro	32423423	34243.0	6
juan	perez	32424323	1231.0	1
rosalia	marrou	34234334	4322.0	4
jose	bardales	34242343	2921.0	2
karen	swartz	35434534	5435.0	2
josefina	mariños	43234245	3223.0	4
jesus	zapata	53432434	2133.0	6

La información ya está registrada en la base de datos, puede cerrar su aplicación y volver a ejecutarlo para comprobarlo.

Añada un comboBox de reportes para mostrar la siguiente información:

- Empleados que pertenecen a un área seleccionada
- Empleados con un sueldo inferior al sueldo promedio
- Empleados en orden alfabético por apellidos