Conexión:

```
package conexion;



import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
/**
 *
 * @author Daniel Xocol
 */
public class conexion {

    public Connection _conexion;
    Statement _st;
    ResultSet _rs;

    public Connection CrearConexion() {
        try {

            if (_conexion == null || _conexion.isClosed()) {
                _conexion = DriverManager.getConnection("jdbc:mysql://localhost" ,"ejercicio","123456");
                _conexion.setTransactionIsolation(Connection.TRANSACTION_READ_COMMITTED);
                _conexion.setAutoCommit(false);
```

```
            }
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(null, ex.getMessage(), "Error al intentar conectar a la base de datos", JOptionPane.ERROR_MESSAGE);
        }
        return _conexion;
    }

    public ResultSet EjecutarConsulta(String query) {
        _st = null;
        _rs = null;
        CrearConexion();
        try {
            _st = _conexion.createStatement();
            _rs = _st.executeQuery(query);
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(null, ex.getMessage(), "Error al ejecutar consulta", JOptionPane.ERROR_MESSAGE);

        }
        return _rs;
    }

    public String EjecutarEscalar(String query) {
        _st = null;
        _rs = null;
        CrearConexion();
        try {
            _st = _conexion.createStatement();
```

```
            _rs = _st.executeQuery(query);
            if (_rs.next()) {
                return _rs.getString(1);
            }
        } catch (SQLException ex) {

            JOptionPane.showMessageDialog(null, ex.getMessage(), "Error al ejecutar consulta",
JOptionPane.ERROR_MESSAGE);

        } finally {
            CerrarConexion();
        }
        return "";
    }

    public String Ejecutar(String query) {
        String resultado = "";
        try {
            CrearConexion();
            _conexion.setAutoCommit(false);
            Statement st = _conexion.createStatement();
            st.execute(query);
            _conexion.commit();
        } catch (SQLException ex) {
            resultado = ex.toString();
            JOptionPane.showMessageDialog(null, ex, "Error al intentar conectar a la base de datos",
JOptionPane.ERROR_MESSAGE);
            try {
                _conexion.rollback();
```

```
            } catch (SQLException ex1) {
                Logger.getLogger(conexion.class.getName()).log(Level.SEVERE, null, ex1);
            }
        } finally {

            CerrarConexion();
        }
        return resultado;
    }


    public void CerrarConexion() {
        try {
            if (_conexion != null || !_conexion.isClosed()) {
                _conexion.close();
                System.gc();
            }
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(null, ex, "Error al cerrar la Base de Datos",
JOptionPane.ERROR_MESSAGE);
        }

    }

}
```

Métodos (Clase proveedores):

```
package proveedores;
```

```
import conexion.conexion;
import java.sql.ResultSet;

/**
 *
 * @author Daniel Xocol
 */
public class NewClass implements interfaz {
private int nit;
    private String nombre;
    private String razón_social;
    private String dirección;
    private String teléfono;
    private int estado;
    private String consulta;
    private String Mensaje;
    private final conexion con;

    public NewClass() {
        con = new conexion();
    }

    public int getNit() {
        return nit;
    }

    public void setNit(int nit) {
        this.nit = nit;
    }
```

```
public String getNombre() {
   return nombre;
}

public void setNombre(String nombre) {
   this.nombre = nombre;
}

public String getRazón_social() {
   return razón_social;
}

public void setRazón_social(String razón_social) {
   this.razón_social = razón_social;
}

public String getDirección() {
   return dirección;
}

public void setDirección(String dirección) {
   this.dirección = dirección;
}

public String getTeléfono() {
   return teléfono;
}
```

```
public void setTeléfono(String teléfono) {
    this.teléfono = teléfono;
}

public int getEstado() {
    return estado;
}

public void setEstado(int estado) {
    this.estado = estado;
}

public String getConsulta() {
    return consulta;
}

public void setConsulta(String consulta) {
    this.consulta = consulta;
}

public String getMensaje() {
    return Mensaje;
}

public void setMensaje(String Mensaje) {
    this.Mensaje = Mensaje;
}

@Override
```

```
    public boolean Guardar() {
        try {
            consulta = "insert into proveedores (nit, nombre, razón_social, dirección, teléfono, estado) values
"
                    + "(" + nit + ", '" + nombre + "', '" + razón_social + "', '" + dirección + "', '" + teléfono + "', " +
estado + ")";
            String respuesta = con.Ejecutar(consulta);
            if (respuesta.equals("")) {
                return true;
            } else {
                Mensaje = "Error la informacion no se guardo, " + respuesta;
                return false;
            }
        } catch (Exception x) {
            Mensaje = "Error la informacion no se guardo, " + x.toString();
            return false;
        }
    }

    @Override
    public boolean Borrar() {
        try {
            consulta = "delete from proveedores where nit = " + nit;
            String respuesta = con.Ejecutar(consulta);
            if (respuesta.equals("")) {
                return true;
            } else {
                Mensaje = "Error la informacion no se elimino, " + respuesta;
                return false;
```

```
        }
    } catch (Exception x) {
        Mensaje = "Error la informacion no se elimino, " + x.toString();
        return false;
    }
}

@Override
public boolean Actualizar() {
  try {
        consulta = "update proveedores set "
                + "nombre = '" + nombre + "', "
                + "razón_social = '" + razón_social + "', "
                + "dirección = '" + dirección + "', "
                + "teléfono = '" + teléfono + "', "
                + "estado = " + estado + " "
                + "where nit = " + nit;
        String respuesta = con.Ejecutar(consulta);
        if (respuesta.equals("")) {
            return true;
        } else {
            Mensaje = "Error la informacion no se actualizo, " + respuesta;
            return false;
        }
    } catch (Exception x) {
        Mensaje = "Error la informacion no se actualizo, " + x.toString();
        return false;
    }
}
```

```
    @Override
    public ResultSet verInformacion() {
      try {
        consulta = "select * from ventas";
        ResultSet rs = con.EjecutarConsulta(consulta);
        return rs;
    } catch (Exception x) {
        Mensaje = "Error al obtener la información, " + x.toString();
        return null;
    }
    }

}
```