



TECNOLOGICO NACIONAL DE MEXICO

Instituto Tecnológico de la Laguna

Ingenieria en Sistemas Computacionales

TOPICOS AVANZADOS DE PROGRAMACION

PERIODO: Ene - Jun / 2020

GRUPO: "B" 17 – 18 Hrs

PRACTICA No. U3P02

Aplicación de base de datos con JavaDB

ALUMNOS:

18131209	José Misael Adame Sandoval
18131227	Ricardo Raúl Castro Luna
18131238	Jorge Arturo Galindo Uribe

PROFESOR:

Ing. Luis Fernando Gil Vázquez

Torreón, Coah. a 26 de mayo de 2020

Situación didáctica

En una escuela secundaria local se está buscando a desarrolladores de software que puedan programar una aplicación de base de datos en Java para el control de calificaciones de los alumnos. La primera versión se espera que cuente con tres funciones principales:

- Gestión de un catálogo de alumnos.
- Gestión de un catálogo de materias.
- Gestión del kardex de calificación de alumnos.

Derivado de un levantamiento de requerimientos preliminar se ha determinado lo siguiente:

1. La gestión de cada catálogo debe contar con las operaciones **CRUD** (Create, Update, Delete), también conocidas como ABC (Altas-Bajas-Cambios).
2. Para el catálogo de alumnos requiere: matrícula, nombre del alumno, apellidos, promedio
3. Para catálogo de materias se requiere: clave de la materia, materia y créditos.
4. Para el kardex de calificaciones se requiere: matrícula del alumno, materia, periodo, calificación.
5. Se requiere que el sistema proporcione opciones de consulta para filtrar datos que cumplan un cierto criterio.
6. Se requiere que el sistema cuente con una función de recálculo de promedios.

Como una restricción la escuela solicita que la base de datos del sistema use **JavaDB** como DBMS.

En la sección de **ANÁLISIS** presentar toda la siguiente información:

- Información de conexión a la base de datos:
 - Nombre de la base de datos
 - Usuario y contraseña
- Presentar un cuadro con la siguiente información de las tablas a crear en la base de datos, indicando cuáles columnas serán llave primaria y llave foránea:
 - Nombre de la tabla
 - Nombre de columnas
 - Tipo de dato de columnas

En la sección de **DISEÑO** va el diagrama de clases UML de la aplicación.

Sección adicional : **ANEXOS**. Colocar aquí el script para poblar las tablas de la base de datos de manera inicial.

En la sección de **PRUEBA DE EJECUCIÓN** presentar un minimanual de usuario incluyendo capturas de pantalla de la aplicación.

En la entrega de esta práctica el archivo RAR debe incluir lo siguiente:

1. Este documento en PDF
2. Código fuente de la aplicación
3. Scripts de creación de la base de datos y de población inicial de tablas.
4. Carpeta que contiene la base de datos

Análisis

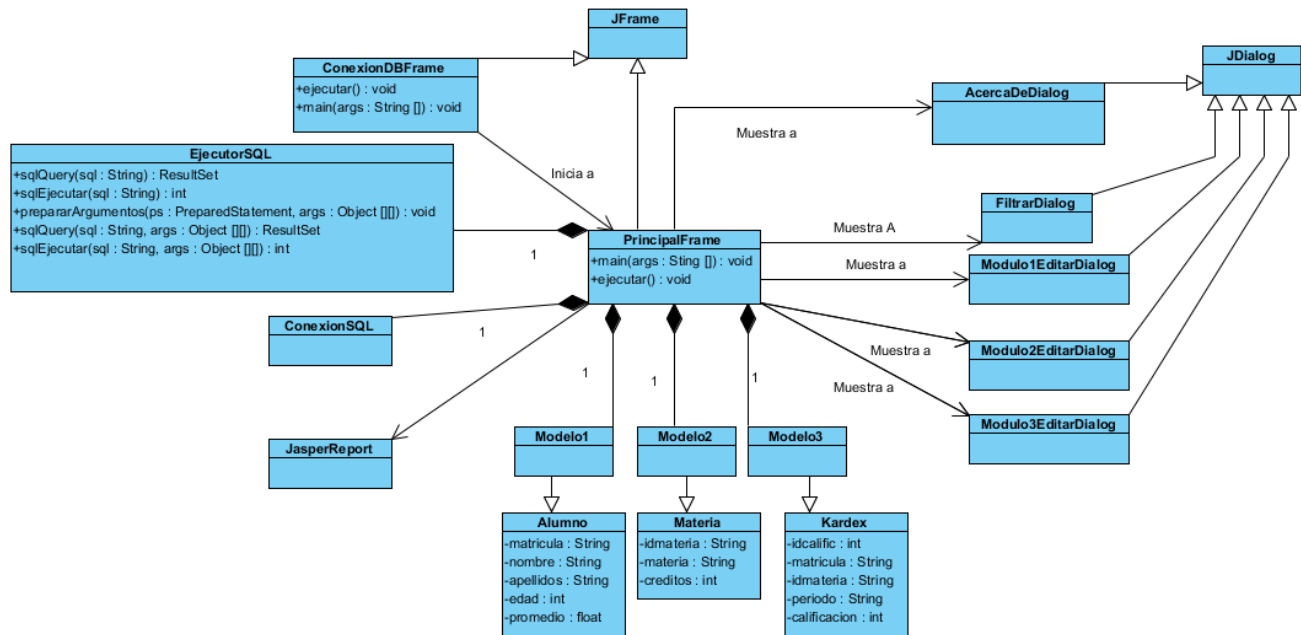
- Información de conexión a la base de datos:
 - Nombre de la base de datos: TAPdb
 - Usuario: topicos
 - contraseña: 1234

Alumnos			
Nombre de Columna	Tipo de dato de columna	Llave primaria	Llave foranea
MATRICULA	VARCHAR(10)	•	
NOMBRE	VARCHAR(30)		
APELLIDOS	VARCHAR(50)		
EDAD	INTEGER		
PROMEDIO	REAL		

Materias			
Nombre de Columna	Nombre de Columna	Nombre de Columna	Nombre de Columna
IDMATERIA	VARCHAR(10)	•	
MATERIA	VARCHAR(50)		
CREDITOS	INTEGER		

Kardex			
Nombre de Columna	Nombre de Columna	Nombre de Columna	Nombre de Columna
IDCALIFIC	INTEGER	•	
MATRICULA	VARCHAR(10)		Alumnos(MATRICULA)
IDMATERIA	VARCHAR(10)		Materias(IDMATERIA)
PERIODO	VARCHAR(5)		
CALIFICACION	INTEGER		

Diseño



Código

script-Creacion de tablas en bd TAPdb.sql

```

/*-----
*:          INSTITUTO TECNOLÓGICO DE LA LAGUNA
*:          INGENIERÍA EN SISTEMAS COMPUTACIONALES
*:          TÓPICOS AVANZADOS DE PROGRAMACIÓN "B"
*:
*:          SEMESTRE: ENE-JUN/2020    HORA: 17-18 HRS
*:
*:          Script de Creación de Tablas de la BD TAPdb
*:
*: Archivo      : script-creacion de tablas en bd TAPdb.sql
*: Autor        : José Misael Adame Sandoval      18131209
*:              Ricardo Raúl Castro Luna         18131227
*:              Jorge Arturo Galindo Uribe       18131238
*: Fecha        : 04/May/2020
*: Descripción  : Script SQL para crear las tablas Alumnos, Materias y Kardex
*:              y poblarlas con registros de prueba
*:
*:
*: Última modif:
*: Fecha      Modificó      Motivo
*:=====
*: 05/May/2020 Misael Adame      Agregar prólogo.
*:-----

/* CREACION DE TABLAS */

CREATE TABLE Alumnos
(
    MATRICULA VARCHAR ( 10 ) PRIMARY KEY,
    NOMBRE VARCHAR ( 30 ) NOT NULL,
    APELLIDOS VARCHAR ( 50 ) NOT NULL,
    EDAD INTEGER,
    PROMEDIO REAL
);

CREATE TABLE Materias
(
    IDMATERIA VARCHAR ( 10 ) PRIMARY KEY,
    MATERIA VARCHAR ( 50 ) NOT NULL,
    CREDITOS INTEGER
);

CREATE TABLE Kardex
(
    IDCALIFIC INTEGER PRIMARY KEY,
    MATRICULA VARCHAR ( 10 ),
    IDMATERIA VARCHAR ( 10 ),
    PERIODO VARCHAR ( 5 ),
    CALIFICACION INTEGER
);

/* CREACION DE LLAVES FORANEAS PARA LA INTEGRIDAD REFERENCIAL */
ALTER TABLE Kardex
ADD CONSTRAINT fk_matricula
FOREIGN KEY ( MATRICULA )
REFERENCES Alumnos ( MATRICULA );

ALTER TABLE Kardex
ADD CONSTRAINT fk_idmateria
FOREIGN KEY ( IDMATERIA )
REFERENCES Materias ( IDMATERIA );

```

PrincipalFrame.java

```

/*-----
*:          INSTITUTO TECNOLÓGICO DE LA LAGUNA
*:          INGENIERÍA EN SISTEMAS COMPUTACIONALES
*:          TÓPICOS AVANZADOS DE PROGRAMACIÓN "B"
*:
*:          SEMESTRE: ENE-JUN/2020    HORA: 17-18 HRS
*:
*:          Aplicación GUI para el entorno de un Sistema de Calificaciones
*:
*: Archivo      : PrincipalFrame.java

```

```

:* Autor      : José Misael Adame Sandoval      18131209
:*            : Ricardo Raúl Castro Luna        18131227
:*            : Jorge Arturo Galindo Uribe      18131238
:* Fecha      : 24/May/2020
:* Compilador : JAVA J2SE vl.8.0
:* Descripción : Aplicación que realiza funciones para modificar la base de datos de un
:*             : Sistema de calificaciones como Agregar, Editar, Eliminar un registro de
:*             : las tablas, como también consultar información específica de las tablas
:*             : mediante un filtrado, además recalcula los promedios de los alumnos y
:*             : despliega reportes de una tabla ya sea Alumnos, Materias o Kardex y estos
:*             : pueden ser exportados a Word o PDF.
:*
:*
:*
:* Ultima modif:
:* Fecha      Modificó      Motivo
:*-----
:* 24/May/2020 Misael Adame      Agregar prólogo.
:*-----*/

```

```
package app;
```

```

import java.awt.Color;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.SQLIntegrityConstraintViolationException;
import java.text.DecimalFormat;
import java.util.Properties;
import java.util.Vector;
import javax.swing.Icon;
import javax.swing.JButton;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import modelo.Modelo1;
import modelo.Modelo2;
import modelo.Modelo3;
import mx.edu.itl.jdbc.*;
import mx.edu.itl.util.Imagenes;
import net.sf.jasperreports.engine.JRException;
import net.sf.jasperreports.engine.JasperCompileManager;
import net.sf.jasperreports.engine.JasperFillManager;
import net.sf.jasperreports.engine.JasperPrint;
import net.sf.jasperreports.engine.JasperReport;
import net.sf.jasperreports.view.JasperViewer;

```

```
public class PrincipalFrame extends javax.swing.JFrame {
```

```

//-----

public static final String NUEVO = "Nuevo";
public static final String EDITAR = "Editar";

public static final String TIT_FRAME = "Sistema de Calificaciones";
public static final String TIT_INICIO = "TecLag Software";
public static final String TIT_MODULO1 = "Alumnos";
public static final String TIT_MODULO2 = "Materias";
public static final String TIT_MODULO3 = "Kardex";

public static final String ALUMNOS_TODOS_POR_NOMBRE = "alumnos_todos_por_nombre";
public static final String ALUMNOS_TODOS_SIN_ORDEN = "alumnos_todos_sin_orden";
public static final String ALUMNOS_ELIMINAR_X_MATRICULA = "alumnos_eliminar_x_matricula";
public static final String ALUMNOS_ACTUALIZA_DATOS = "alumnos_actualiza_datos";
public static final String ALUMNOS_INSERTA_NUEVO = "alumnos_inserta_nuevo";
public static final String ALUMNOS_ACTUALIZA_PROMEDIOS = "alumnos_actualiza_promedios";
public static final String MATERIAS_TODOS_POR_MATERIA = "materias_todos_por_materia";
public static final String MATERIAS_TODOS_SIN_ORDEN = "materias_todos_sin_orden";
public static final String MATERIAS_ELIMINAR_X_IDMATERIA = "materias_eliminar_x_idmateria";
public static final String MATERIAS_ACTUALIZA_DATOS = "materias_actualiza_datos";
public static final String MATERIAS_INSERTA_NUEVO = "materias_inserta_nuevo";
public static final String KARDEX_TODOS_POR_MATRICULA = "kardex_todos_por_matricula";
public static final String KARDEX_TODOS_SIN_ORDEN = "kardex_todos_sin_orden";
public static final String KARDEX_ELIMINAR_X_IDCALIFIC = "kardex_eliminar_x_idcalific";
public static final String KARDEX_ACTUALIZA_DATOS = "kardex_actualiza_datos";
public static final String KARDEX_INSERTA_NUEVO = "kardex_inserta_nuevo";
public static final String KARDEX_OBT_MAX_ID = "kardex_obt_max_id";

//-----

private AcercaDeDialog acercaDeDialog;

```

```

private String moduloActual;
private Vector<String> vecNombresColumnas;
private Vector<String> vecNombresColumnasBD;
private Vector<String> vecTiposColumnas;
private DefaultTableModel dtmPrincipal;
private Properties propConsultasSQL;
private int totRegistros;

//-----
// Constructor

public PrincipalFrame () {
    initComponents();

    this.setTitle ( TIT_FRAME );
    lblMensajeDelSistema.setText ( "" );
    jtoolbPrincipal.setVisible ( false );
    jpn1Tabla.setVisible ( false );
    jpn1LogoPrincipal.setVisible ( true );
    lblLeyendaPrincipal.setText ( TIT_INICIO );
    lblModulo1.setText ( TIT_MODULO1 );
    lblModulo2.setText ( TIT_MODULO2 );
    lblModulo3.setText ( TIT_MODULO3 );
    jmn1ReportesModulo1.setText ( TIT_MODULO1 );
    jmn1ReportesModulo2.setText ( TIT_MODULO2 );
    jmn1ReportesModulo3.setText ( TIT_MODULO3 );

    // Ajustar el tamaño de las imágenes
    Icon alumno = Imagenes.escalarImagen ( jbtnModulo1.getIcon(),
                                            jbtnModulo1.getWidth(),
                                            jbtnModulo1.getHeight() );

    jbtnModulo1.setIcon ( alumno );

    Icon materia = Imagenes.escalarImagen ( jbtnModulo2.getIcon(),
                                            jbtnModulo2.getWidth(),
                                            jbtnModulo2.getHeight() );

    jbtnModulo2.setIcon ( materia );

    Icon kardex = Imagenes.escalarImagen ( jbtnModulo3.getIcon(),
                                            jbtnModulo3.getWidth(),
                                            jbtnModulo3.getHeight() );

    jbtnModulo3.setIcon ( kardex );

    prepararSentenciasSQL ();

    new ConexionDBFrame ( this ).setVisible ( true );
}

//-----

private void prepararSentenciasSQL () {
    propConsultasSQL = new Properties ();

    propConsultasSQL.put ( ALUMNOS_TODOS_POR_NOMBRE,
        "SELECT * FROM alumnos ORDER BY nombre"
    );

    propConsultasSQL.put ( ALUMNOS_TODOS_SIN_ORDEN,
        "SELECT * FROM alumnos"
    );

    propConsultasSQL.put ( ALUMNOS_ELIMINAR_X_MATRICULA,
        "DELETE FROM alumnos WHERE matrícula = ?"
    );

    propConsultasSQL.put ( ALUMNOS_ACTUALIZA_DATOS,
        "UPDATE alumnos SET nombre = ?, apellidos = ?, edad = ?, " +
        "promedio = ? WHERE matrícula = ?"
    );

    propConsultasSQL.put ( ALUMNOS_INSERTA_NUEVO,
        "INSERT INTO alumnos VALUES ( ?, ?, ?, ? )"
    );

    propConsultasSQL.put ( ALUMNOS_ACTUALIZA_PROMEDIOS,
        "UPDATE alumnos A SET promedio =( " +
        "SELECT AVG ( CAST ( calificacion AS FLOAT ) ) FROM kardex K " +
        "WHERE K.matricula = A.matricula )"
    );

    propConsultasSQL.put ( MATERIAS_TODOS_POR_MATERIA,

```

```

        "SELECT * FROM materias ORDER BY materia"
    );
    propConsultasSQL.put ( MATERIAS_TODOS_SIN_ORDEN,
        "SELECT * FROM materias"
    );
    propConsultasSQL.put ( MATERIAS_ELIMINAR_X_IDMATERIA,
        "DELETE FROM materias WHERE idmateria = ?"
    );
    propConsultasSQL.put ( MATERIAS_ACTUALIZA_DATOS,
        "UPDATE materias SET materia = ?, creditos = ? " +
        "WHERE idmateria = ?"
    );
    propConsultasSQL.put ( MATERIAS_INSERTA_NUEVO,
        "INSERT INTO materias VALUES ( ?, ?, ? )"
    );
    propConsultasSQL.put ( KARDEX_TODOS_POR_MATRICULA,
        "SELECT K.IDCALIFIC, K.MATRÍCULA, A.NOMBRE, A.APELLIDOS, "
        + "K.PERIODO, M.IDMATERIA, M.MATERIA, K.CALIFICACION " +
        "FROM Kardex K, Alumnos A, Materias M " +
        "WHERE K.MATRÍCULA = A.MATRÍCULA AND K.IDMATERIA = M.IDMATERIA " +
        "ORDER BY K.MATRÍCULA"
    );
    propConsultasSQL.put ( KARDEX_TODOS_SIN_ORDEN,
        "SELECT K.IDCALIFIC, K.MATRÍCULA, A.NOMBRE, A.APELLIDOS, "
        + "K.PERIODO, M.IDMATERIA, M.MATERIA, K.CALIFICACION " +
        "FROM Kardex K JOIN Alumnos A ON K.MATRÍCULA = A.MATRÍCULA " +
        "JOIN Materias M ON K.IDMATERIA = M.IDMATERIA"
    );
    propConsultasSQL.put ( KARDEX_ELIMINAR_X_IDCALIFIC,
        "DELETE FROM kardex WHERE idcalific = ?"
    );
    propConsultasSQL.put ( KARDEX_ACTUALIZA_DATOS,
        "UPDATE kardex SET matricula = ?, idmateria = ?, periodo = ?, " +
        "calificacion = ? WHERE idcalific = ?";
    propConsultasSQL.put ( KARDEX_INSERTA_NUEVO,
        "INSERT INTO kardex VALUES ( ?, ?, ?, ? )" );
    propConsultasSQL.put ( KARDEX_OBT_MAX_ID,
        "SELECT MAX ( idcalific ) AS maxid FROM kardex");
}

```

```
//-----
```

```

private void jbtnNuevoActionPerformed(java.awt.event.ActionEvent evt) {
    if ( moduloActual.equals ( TIT_MODULO1 ) ) {
        Modulo1EditarDialog dialog = new Modulo1EditarDialog ( this, null );
        dialog.setVisible ( true );

    } else if ( moduloActual.equals ( TIT_MODULO2 ) ) {
        Modulo2EditarDialog dialog = new Modulo2EditarDialog ( this, null );
        dialog.setVisible ( true );

    } else if ( moduloActual.equals ( TIT_MODULO3 ) ) {
        Modulo3EditarDialog dialog = new Modulo3EditarDialog ( this, null );
        dialog.setVisible ( true );

    }
}

```

```
//-----
```

```

private void formWindowActivated(java.awt.event.WindowEvent evt) {
    if ( ConexionDB.getInstancia ().conectado () ) {
        lblEstatusConexion.setText ( "Conectado" );
        lblEstatusConexion.setForeground ( Color.blue );
    } else {
        lblEstatusConexion.setText ( "Desconectado" );
        lblEstatusConexion.setForeground ( Color.red );
    }
}

```

```
//-----
```

```

private void jmnSalirActionPerformed(java.awt.event.ActionEvent evt) {
    ConexionDB.getInstancia ().desconectar ();
    dispose ();
}

```

```
//-----
```

```

private void jbtnModulo1ActionPerformed(java.awt.event.ActionEvent evt) {
    prepararVistaModulo ( TIT_MODULO1 );

    String sql = propConsultasSQL.getProperty ( ALUMNOS_TODOS_POR_NOMBRE );
}

```



```
        desplegarRegistros ( sql, null );
    }

    //-----

    private void jButtonModulo2ActionPerformed(java.awt.event.ActionEvent evt) {
        prepararVistaModulo ( TIT_MODULO2 );

        String sql = propConsultasSQL.getProperty ( MATERIAS_TODOS_POR_MATERIA );
        desplegarRegistros ( sql, null );
    }

    //-----

    private void jButtonModulo3ActionPerformed(java.awt.event.ActionEvent evt) {
        prepararVistaModulo ( TIT_MODULO3 );

        String sql = propConsultasSQL.getProperty ( KARDEX_TODOS_POR_MATRICULA );
        desplegarRegistros ( sql, null );
    }

    //-----

    private void jButtonInicioActionPerformed(java.awt.event.ActionEvent evt) {
        jButtonPrincipal.setVisible ( false );
        jPanelLogoPrincipal.setVisible ( true );
        jPanelTabla.setVisible ( false );

        jLabelLeyendaPrincipal.setText ( TIT_INICIO );
        moduloActual = "";
    }

    //-----

    private void jButtonFiltrarActionPerformed(java.awt.event.ActionEvent evt) {
        FilterDialog filtrarDialog = new FilterDialog ( this, true );
        filtrarDialog.setVisible ( true );
    }

    //-----

    private void jButtonEliminarActionPerformed(java.awt.event.ActionEvent evt) {
        int pos = jTablePrincipal.getSelectedRow ();
        if ( pos == -1 ) {
            JOptionPane.showMessageDialog (
                this, "No hay un registro seleccionado", "Eliminar", JOptionPane.ERROR_MESSAGE );
            return;
        }
        int confirma = JOptionPane.showConfirmDialog (
            this,
            "¿Eliminar el registro seleccionado?",
            "Eliminar",
            JOptionPane.YES_NO_OPTION,
            JOptionPane.QUESTION_MESSAGE );

        if ( confirma == JOptionPane.NO_OPTION )
            return;

        String valorLlavePrim = jTablePrincipal.getValueAt ( pos, 0 ).toString ();
        String tipoLlavePrim = vecTiposColumnas.elementAt ( 0 );
        String sql = "";
        JButton jButtonModulo = null;

        switch ( moduloActual ) {
            case TIT_MODULO1:
                sql = propConsultasSQL.getProperty ( ALUMNOS_ELIMINAR_X_MATRICULA );
                jButtonModulo = jButtonModulo1;
                break;
            case TIT_MODULO2:
                sql = propConsultasSQL.getProperty ( MATERIAS_ELIMINAR_X_IDMATERIA );
                jButtonModulo = jButtonModulo2;
                break;
            case TIT_MODULO3:
                sql = propConsultasSQL.getProperty ( KARDEX_ELIMINAR_X_IDCALIFIC );
                jButtonModulo = jButtonModulo3;
                break;
        }

        Object [][] args = { { tipoLlavePrim, valorLlavePrim } };
        try {
```

```

        int regs = EjecutorSQL.sqlEjecutar ( sql, args );
        if ( regs == 1 ) {
            jbtnModulo.doClick ();
            JOptionPane.showMessageDialog (
                this, "El registro ha sido eliminado", "Eliminar",
                JOptionPane.INFORMATION_MESSAGE );
        }
    } catch ( SQLIntegrityConstraintViolationException ex ) {
        JOptionPane.showMessageDialog (
            this,
            "No se puede eliminar el registro actual porque tiene " +
            "registros asociados en otras tablas",
            "Error",
            JOptionPane.ERROR_MESSAGE );
    } catch ( SQLException ex ) {
        JOptionPane.showMessageDialog (
            this, ex, "Error", JOptionPane.ERROR_MESSAGE );
    }
}

//-----

private void formWindowClosing(java.awt.event.WindowEvent evt) {
    jmniSalir.doClick ();
}

//-----

private void jbtnEditarActionPerformed(java.awt.event.ActionEvent evt) {
    int pos = jTablePrincipal.getSelectedRow ();
    if ( pos == -1 ) {
        JOptionPane.showMessageDialog (
            this, "No hay un registro seleccionado", "Eliminar", JOptionPane.ERROR_MESSAGE );
        return;
    }
    if ( moduloActual.equals ( TIT_MODULO1 ) ) {
        String matricula = jTablePrincipal.getValueAt ( pos, 0 ).toString ();
        String nombre = jTablePrincipal.getValueAt ( pos, 1 ).toString ();
        String apellidos = jTablePrincipal.getValueAt ( pos, 2 ).toString ();
        int edad = Integer.parseInt ( jTablePrincipal.getValueAt ( pos, 3 ).toString () );
        float promedio = Float.parseFloat ( jTablePrincipal.getValueAt ( pos, 4 ).toString () );

        Modelo1 modelo1 = new Modelo1 ( matricula, nombre, apellidos, edad, promedio );

        Modulo1EditarDialog dialog = new Modulo1EditarDialog ( this, modelo1 );
        dialog.setVisible ( true );
    } else if ( moduloActual.equals ( TIT_MODULO2 ) ) {
        String idmateria = jTablePrincipal.getValueAt ( pos, 0 ).toString ();
        String materia = jTablePrincipal.getValueAt ( pos, 1 ).toString ();
        int creditos = Integer.parseInt ( jTablePrincipal.getValueAt ( pos, 2 ).toString () );

        Modelo2 modelo = new Modelo2 ( idmateria, materia, creditos );

        Modulo2EditarDialog dialog = new Modulo2EditarDialog ( this, modelo );
        dialog.setVisible ( true );
    } else if ( moduloActual.equals ( TIT_MODULO3 ) ) {
        int idcalific = Integer.parseInt ( jTablePrincipal.getValueAt ( pos, 0 ).toString () );
        String matricula = jTablePrincipal.getValueAt ( pos, 1 ).toString ();
        String idmateria = jTablePrincipal.getValueAt ( pos, 4 ).toString ();
        String periodo = jTablePrincipal.getValueAt ( pos, 6 ).toString ();
        String califica = jTablePrincipal.getValueAt ( pos, 7 ).toString ();
        int calificacion = Integer.parseInt ( califica );

        Modelo3 modelo = new Modelo3 ( idcalific, matricula, idmateria,
            periodo, calificacion );

        Modulo3EditarDialog dialog = new Modulo3EditarDialog ( this, modelo );
        dialog.setVisible ( true );
    }
}

//-----

private void jmniRecalcPromediosActionPerformed(java.awt.event.ActionEvent evt) {
    String sql = propConsultasSQL.getProperty ( ALUMNOS_ACTUALIZA_PROMEDIOS );
    try {
        int regs = EjecutorSQL.sqlEjecutar ( sql );
        jbtnModulo1.doClick ();
    }
}

```

```

        JOptionPane.showMessageDialog (
            this,
            "Proceso terminado. " + regs + " registros fueron actualizados",
            "Promedio",
            JOptionPane.INFORMATION_MESSAGE
        );
    } catch ( SQLException ex ) {
        JOptionPane.showMessageDialog (
            this,
            ex,
            "Error",
            JOptionPane.ERROR_MESSAGE
        );
    }
}

//-----

private void jmnAcercaDeActionPerformed(java.awt.event.ActionEvent evt) {
    acercaDeDialog = new AcercaDeDialog ( this, true );
    acercaDeDialog.setVisible( true );
}

//-----

private void jmnReportesModulo1ActionPerformed(java.awt.event.ActionEvent evt) {
    String reporte = "src\\reportes\\AlumnosReport.jrxml";
    JasperReport jr;
    try {
        jr = JasperCompileManager.compileReport ( reporte );
        JasperPrint jp = JasperFillManager.fillReport ( jr, null,
ConexionDB.getInstancia().getConexion() );
        JasperViewer jv = new JasperViewer ( jp, false );
        jv.setTitle ( "Reporte Alumnos" );
        jv.setVisible ( true );
    } catch ( JRException ex ) {
        JOptionPane.showMessageDialog ( this,
            ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}

//-----

private void jmnReportesModulo2ActionPerformed(java.awt.event.ActionEvent evt) {
    String reporte = "src\\reportes\\MateriasReport.jrxml";
    JasperReport jr;
    try {
        jr = JasperCompileManager.compileReport ( reporte );
        JasperPrint jp = JasperFillManager.fillReport ( jr, null,
ConexionDB.getInstancia().getConexion() );
        JasperViewer jv = new JasperViewer ( jp, false );
        jv.setTitle ( "Reporte Materias" );
        jv.setVisible ( true );
    } catch ( JRException ex ) {
        JOptionPane.showMessageDialog ( this,
            ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}

//-----

private void jmnReportesModulo3ActionPerformed(java.awt.event.ActionEvent evt) {
    String reporte = "src\\reportes\\KardexReport.jrxml";
    JasperReport jr;
    try {
        jr = JasperCompileManager.compileReport ( reporte );
        JasperPrint jp = JasperFillManager.fillReport ( jr, null,
ConexionDB.getInstancia().getConexion() );
        JasperViewer jv = new JasperViewer ( jp, false );
        jv.setTitle ( "Reporte Kardex" );
        jv.setVisible ( true );
    } catch ( JRException ex ) {
        JOptionPane.showMessageDialog ( this,
            ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}

//-----

private void prepararVistaModulo ( String modulo ) {
    moduloActual = modulo;
}

```

```

jtoolbPrincipal.setVisible ( true );
jpnLogoPrincipal.setVisible ( false );
jpnTabla.setVisible ( true );

jlblLeyendaPrincipal.setText ( modulo );
determinarNombresColumnas ( modulo );

dtmPrincipal = new DefaultTableModel ( vecNombresColumnas, 0 );
jtblPrincipal.setModel ( dtmPrincipal );
}

//-----

private void determinarNombresColumnas ( String modulo ) {
    vecNombresColumnas = new Vector<String> ();
    vecNombresColumnasBD = new Vector<String> ();
    vecTiposColumnas = new Vector<String> ();

    switch ( modulo ) {
        case TIT_MODULO1:
            vecNombresColumnas.add ( "Matricula" );
            vecNombresColumnas.add ( "Nombre" );
            vecNombresColumnas.add ( "Apellidos" );
            vecNombresColumnas.add ( "Edad" );
            vecNombresColumnas.add ( "Promedio" );

            vecNombresColumnasBD.add ( "matricula" );
            vecNombresColumnasBD.add ( "nombre" );
            vecNombresColumnasBD.add ( "apellidos" );
            vecNombresColumnasBD.add ( "edad" );
            vecNombresColumnasBD.add ( "promedio" );

            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.INT );
            vecTiposColumnas.add ( EjecutorSQL.FLOAT );

            break;
        case TIT_MODULO2:
            vecNombresColumnas.add ( "ID Materia" );
            vecNombresColumnas.add ( "Materia" );
            vecNombresColumnas.add ( "Creditos" );

            vecNombresColumnasBD.add ( "idmateria" );
            vecNombresColumnasBD.add ( "materia" );
            vecNombresColumnasBD.add ( "creditos" );

            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.INT );

            break;
        case TIT_MODULO3:
            vecNombresColumnas.add ( "ID Calific" );
            vecNombresColumnas.add ( "Matricula" );
            vecNombresColumnas.add ( "Nombre" );
            vecNombresColumnas.add ( "Apellidos" );
            vecNombresColumnas.add ( "ID Materia" );
            vecNombresColumnas.add ( "Materia" );
            vecNombresColumnas.add ( "Periodo" );
            vecNombresColumnas.add ( "Calificación" );

            vecNombresColumnasBD.add ( "K.idcalific" );
            vecNombresColumnasBD.add ( "K.matricula" );
            vecNombresColumnasBD.add ( "A.nombre" );
            vecNombresColumnasBD.add ( "A.apellidos" );
            vecNombresColumnasBD.add ( "M.idmateria" );
            vecNombresColumnasBD.add ( "M.materia" );
            vecNombresColumnasBD.add ( "K.periodo" );
            vecNombresColumnasBD.add ( "K.calificacion" );

            vecTiposColumnas.add ( EjecutorSQL.INT );
            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.STRING );
            vecTiposColumnas.add ( EjecutorSQL.INT );
    }
}

```

```

        break;
    }
}

//-----

public void desplegarRegistros ( String sql, Object [][] args ) {
    ResultSet rs;
    try {

        rs = EjecutorSQL.sqlQuery( sql , args );

        dtmPrincipal = new DefaultTableModel ( vecNombresColumnas, 0 ) {
            @Override
            public boolean isCellEditable ( int fila, int columna ) {
                return false;
            }
        };

        while ( rs.next() ) {
            Object [] fila = crearFila ( rs );
            dtmPrincipal.addRow ( fila );
        }
        rs.close ();
        jTablePrincipal.setModel( dtmPrincipal );
        totRegistros = dtmPrincipal.getRowCount ();
        jLabelMensajeDelSistema.setText ( totRegistros + " registros" );

    } catch ( SQLException ex ) {
        JOptionPane.showMessageDialog ( this,
            ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}

//-----

private Object [] crearFila ( ResultSet rs) throws SQLException {
    switch ( moduloActual ) {
        case TIT_MODULO1:
            {
                DecimalFormat df = new DecimalFormat("0.00");

                String matricula = rs.getString ( "Matricula" );
                String nombre = rs.getString ( "Nombre" );
                String apellidos = rs.getString ( "Apellidos" );
                int edad = rs.getInt ( "Edad" );
                String promedio = df.format ( rs.getFloat ( "Promedio" ) );

                Object [] fila = { matricula, nombre, apellidos, edad, promedio };
                return fila;
            }
        case TIT_MODULO2:
            {
                String idmateria = rs.getString ( "IDmateria" );
                String materia = rs.getString ( "Materia" );
                int creditos = rs.getInt( "Creditos" );

                Object [] fila = { idmateria, materia, creditos };
                return fila;
            }
        case TIT_MODULO3:
            {
                int idcalific = rs.getInt ( "idcalific" );
                String matricula = rs.getString ( "Matricula" );
                String nombre = rs.getString ( "Nombre" );
                String apellidos = rs.getString ( "Apellidos" );
                String idmateria = rs.getString ( "idmateria" );
                String materia = rs.getString ( "materia" );
                String periodo = rs.getString ( "periodo" );
                int calificacion = rs.getInt ( "calificacion" );

                Object [] fila = { idcalific, matricula, nombre, apellidos, idmateria,
                    materia, periodo, calificacion };
                return fila;
            }
        default:
            return null;
    }
}

```

```
//-----
public static void ejecutar() {
    /* Set the Nimbus look and feel */
    //
```

Modulo1EditorDialog.java

```

/*
*          INSTITUTO TECNOLOGICO DE LA LAGUNA
*          INGENIERIA EN SISTEMAS COMPUTACIONALES
*          TOPICOS AVANZADOS DE PROGRAMACION "B"
*
*          SEMESTRE: ENE-JUN/2020      HORA: 17-18 HRS
*
*          Aplicación GUI para editar o crear registros en el Módulo 1
*
* Archivo      : ModulolEditarDialog.java
* Autor        : José Misael Adame Sandoval      18131209
*              : Ricardo Raúl Castro Luna        18131227
*              : Jorge Arturo Galindo Uribe      18131238
* Fecha        : 24/May/2020
* Compilador   : JAVA J2SE vl.8.0
* Descripción  : Aplicación que crea o edita un registro de la tabla Alumnos con los
*               siguientes datos:
*               1. Matricula
*               2. Nombre
*               3. Apellidos
*               4. Edad
*               5. Promedio
*
*
*
*
* Última modif:
* Fecha      Modificó      Motivo

```

```

: *=====
: * 24/May/2020 Misael Adame      Agregar prólogo.
: *-----*/

package app;

import java.sql.SQLException;
import java.util.Vector;
import javax.swing.Icon;
import javax.swing.JOptionPane;
import modelo.Modelo1;
import mx.edu.itl.jdbc.EjecutorSQL;
import mx.edu.itl.util.Imagenes;

public class Modulo1EditarDialog extends javax.swing.JDialog {

    //-----

    private PrincipalFrame frmPrincipal;
    private Modelo1 modelo;
    private String accion;
    private Vector<String> vecTiposColumnas;
    private String sql;

    //-----

    public Modulo1EditarDialog ( java.awt.Frame parent, Modelo1 modelo ) {
        super ( parent, true );
        initComponents ();

        frmPrincipal = (PrincipalFrame) parent;
        this.modelo = modelo;
        vecTiposColumnas = frmPrincipal.getVecTiposColumnas ();

        accion = ( modelo == null )? PrincipalFrame.NUEVO : PrincipalFrame.EDITAR;
        setTitle ( accion );

        Icon alumno = Imagenes.escalarImagen ( lblLogo.getIcon(),
                                                lblLogo.getWidth(),
                                                lblLogo.getHeight() );

        lblLogo.setIcon ( alumno );

        inicializarFormulario ();
    }

    //-----

    public void inicializarFormulario () {
        if ( accion.equals ( PrincipalFrame.NUEVO ) ) {
            txtMatricula.requestFocus ();
        } else if ( accion.equals ( PrincipalFrame.EDITAR ) ) {
            txtMatricula.setText ( modelo.getMatricula () );
            txtNombre.setText ( modelo.getNombre () );
            txtApellidos.setText ( modelo.getApellidos () );
            jsnEdad.setValue ( modelo.getEdad () );
            jftxFPromedio.setText ( modelo.getPromedio () + "" );

            txtNombre.requestFocus ();
            txtMatricula.setEditable ( false );
        }
    }

    //-----

    private void jButtonCancelarActionPerformed(java.awt.event.ActionEvent evt) {
        dispose ();
    }

    //-----

    private void jButtonGuardarActionPerformed(java.awt.event.ActionEvent evt) {

        Object [][] args = null;
        String mensaje = "";

        if ( validarDatos () == false )
            return;

        // Se determina la sentencia SQL a ejecutar y formar la matriz de argumentos
    }

```

```

if ( accion.equals ( PrincipalFrame.NUEVO ) ) {
    mensaje = "El registro ha sido agregado";
    sql = frmPrincipal.getPropConsultasSQL().getProperty (
        PrincipalFrame.ALUMNOS_INSERTA_NUEVO );
    args = new Object [][] {
        { vecTiposColumnas.elementAt ( 0 ), modelo.getMatricula () },
        { vecTiposColumnas.elementAt ( 1 ), modelo.getNombre () },
        { vecTiposColumnas.elementAt ( 2 ), modelo.getApellidos () },
        { vecTiposColumnas.elementAt ( 3 ), modelo.getEdad () },
        { vecTiposColumnas.elementAt ( 4 ), modelo.getPromedio () }
    };
} else if ( accion.equals ( PrincipalFrame.EDITAR ) ) {
    mensaje = "El registro ha sido actualizado";
    sql = frmPrincipal.getPropConsultasSQL().getProperty (
        PrincipalFrame.ALUMNOS_ACTUALIZA_DATOS );
    args = new Object [][] {
        { vecTiposColumnas.elementAt ( 1 ), modelo.getNombre () },
        { vecTiposColumnas.elementAt ( 2 ), modelo.getApellidos () },
        { vecTiposColumnas.elementAt ( 3 ), modelo.getEdad () },
        { vecTiposColumnas.elementAt ( 4 ), modelo.getPromedio () },
        { vecTiposColumnas.elementAt ( 0 ), modelo.getMatricula () }
    };
}

try {
    int regs = EjecutorSQL.sqlEjecutar ( sql, args );
    if ( regs == 1 ) {
        frmPrincipal.getJbtnModulo1 ().doClick();
        JOptionPane.showMessageDialog(
            this,
            "El registro ha sido actualizado",
            mensaje,
            JOptionPane.INFORMATION_MESSAGE );
    }
} catch ( SQLException ex ) {
    dialogoMensaje ( ex.toString () );
}
dispose ();
}

//-----

private boolean validarDatos () {
    String matricula = jtxfMatricula.getText ();
    if ( matricula.trim ().equals ( "" ) ) {
        dialogoMensaje ( "No se permite un valor en blanco" );
        jtxfMatricula.requestFocus ();
        return false;
    }

    String nombre = jtxfNombre.getText ();
    if ( nombre.trim ().equals ( "" ) ) {
        dialogoMensaje ( "No se permite un valor en blanco" );
        jtxfNombre.requestFocus ();
        return false;
    }

    String apellidos = jtxfApellidos.getText ();
    if ( apellidos.trim ().equals ( "" ) ) {
        dialogoMensaje ( "No se permite un valor en blanco" );
        jtxfApellidos.requestFocus ();
        return false;
    }

    int edad = 0;
    try {
        edad = Integer.parseInt( jspnEdad.getValue ().toString() );
    } catch ( NumberFormatException ex ) {
        dialogoMensaje ( "Debe capturar un valor numerico valido" );
        jspnEdad.requestFocus ();
        return false;
    }

    float promedio;
    try {
        promedio = Float.parseFloat ( jftxfPromedio.getText () );
        if ( promedio < 0 || promedio > 100 ) {
            dialogoMensaje ( "Debe capturar un valor numerico valido" );
            jftxfPromedio.requestFocus ();
            return false;
        }
    }
}

```



```

    }
    } catch ( NumberFormatException ex ) {
        dialogoMensaje ( "Debe capturar un valor numerico valido");
        jftxfPromedio.requestFocus ();
        return false;
    }

    modelo = new Modelo1 ( matricula, nombre, apellidos, edad, promedio );
    return true;
}

//-----

private void dialogoMensaje ( String mensaje ) {
    JOptionPane.showMessageDialog ( this, mensaje, "Error", JOptionPane.ERROR_MESSAGE );
}

//-----

public static void ejecutar () {
    /* Set the Windows look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
    javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Windows".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {

    java.util.logging.Logger.getLogger(Modulo1EditorDialog.class.getName()).log(java.util.logging.Level.SEVERE,
    null, ex);
    } catch (InstantiationException ex) {

    java.util.logging.Logger.getLogger(Modulo1EditorDialog.class.getName()).log(java.util.logging.Level.SEVERE,
    null, ex);
    } catch (IllegalAccessException ex) {

    java.util.logging.Logger.getLogger(Modulo1EditorDialog.class.getName()).log(java.util.logging.Level.SEVERE,
    null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

    java.util.logging.Logger.getLogger(Modulo1EditorDialog.class.getName()).log(java.util.logging.Level.SEVERE,
    null, ex);
    }
    //</editor-fold>

    /* Create and display the dialog */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            Modulo1EditorDialog dialog = new Modulo1EditorDialog(new javax.swing.JFrame(), null );
            dialog.addWindowListener(new java.awt.event.WindowAdapter() {
                @Override
                public void windowClosing(java.awt.event.WindowEvent e) {
                    System.exit(0);
                }
            });
            dialog.setVisible(true);
        }
    });
}

//-----

public static void main(String args[]) {
    ejecutar ();
}

//-----

```

Modulo2EditorDialog.java

```

/*-----
*:          INSTITUTO TECNOLÓGICO DE LA LAGUNA
*:          INGENIERÍA EN SISTEMAS COMPUTACIONALES
*:          TEMAS AVANZADOS DE PROGRAMACIÓN "B"

```

```

:*
:*
:*          SEMESTRE: ENE-JUN/2020      HORA: 17-18 HRS
:*
:*          Aplicación GUI para editar o crear registros en el Módulo 2
:*
:* Archivo      : Modulo2EditarDialog.java
:* Autor        : José Misael Adame Sandoval      18131209
:*              Ricardo Raúl Castro Luna          18131227
:*              Jorge Arturo Galindo Uribe        18131238
:* Fecha        : 24/May/2020
:* Compilador   : JAVA J2SE v1.8.0
:* Descripción  : Aplicación que crea o edita un registro de la tabla Materias con los
:*                siguientes datos:
:*                1. ID Materia
:*                2. Materia
:*                3. Creditos
:*
:*
:* Última modif:
:* Fecha      Modificó          Motivo
:*=====
:* 24/May/2020 Misael Adame      Agregar prólogo.
:*-----*/

package app;

import java.sql.SQLException;
import java.util.Vector;
import javax.swing.Icon;
import javax.swing.JOptionPane;
import modelo.Modelo2;
import mx.edu.itl.jdbc.EjecutorSQL;
import mx.edu.itl.util.Imagenes;

public class Modulo2EditarDialog extends javax.swing.JDialog {

    //-----

    private PrincipalFrame frmPrincipal;
    private Modelo2 modelo;
    private String accion;
    private Vector<String> vecTiposColumnas;
    private String sql;

    //-----

    public Modulo2EditarDialog ( java.awt.Frame parent, Modelo2 modelo ) {
        super ( parent, true );
        initComponents ();

        frmPrincipal = (PrincipalFrame) parent;
        this.modelo = modelo;
        vecTiposColumnas = frmPrincipal.getVecTiposColumnas ();

        accion = ( modelo == null ) ? PrincipalFrame.NUEVO : PrincipalFrame.EDITAR;
        setTitle ( accion );

        Icon materia = Imagenes.escalarImagen ( lblLogo.getIcon(),
                                                lblLogo.getWidth(),
                                                lblLogo.getHeight() );

        lblLogo.setIcon ( materia );

        inicializarFormulario ();
    }

    //-----

    public void inicializarFormulario () {
        if ( accion.equals ( PrincipalFrame.NUEVO ) ) {
            jtxfIDMateria.requestFocus ();
        } else if ( accion.equals ( PrincipalFrame.EDITAR ) ) {
            jtxfIDMateria.setText ( modelo.getIdmateria () );
            jtxfMateria.setText ( modelo.getMateria () );
            jsnpCreditos.setValue ( modelo.getCreditos () );

            jtxfMateria.requestFocus ();
            jtxfIDMateria.setEditable ( false );
        }
    }
}

```

```
//-----

private void jButtonCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    dispose ();
}

//-----

private void jButtonGuardarActionPerformed(java.awt.event.ActionEvent evt) {
    Object [][] args = null;
    String mensaje = "";

    if ( validarDatos () == false )
        return;

    // Se determina la sentencia SQL a ejecutar y formar la matriz de argumentos
    if ( accion.equals ( PrincipalFrame.NUEVO ) ) {
        mensaje = "El registro ha sido agregado";
        sql = frmPrincipal.getPropConsultasSQL().getProperty (
            PrincipalFrame.MATERIAS_INSERTA_NUEVO );
        args = new Object [][] {
            { vecTiposColumnas.elementAt ( 0 ), modelo.getIdmateria () },
            { vecTiposColumnas.elementAt ( 1 ), modelo.getMateria () },
            { vecTiposColumnas.elementAt ( 2 ), modelo.getCreditos () },
        };
    } else if ( accion.equals ( PrincipalFrame.EDITAR ) ) {
        mensaje = "El registro ha sido actualizado";
        sql = frmPrincipal.getPropConsultasSQL().getProperty (
            PrincipalFrame.MATERIAS_ACTUALIZA_DATOS );
        args = new Object [][] {
            { vecTiposColumnas.elementAt ( 1 ), modelo.getMateria () },
            { vecTiposColumnas.elementAt ( 2 ), modelo.getCreditos () },
            { vecTiposColumnas.elementAt ( 0 ), modelo.getIdmateria () }
        };
    }

    try {
        int regs = EjecutorSQL.sqlEjecutar ( sql, args );
        if ( regs == 1 ) {
            frmPrincipal.getJbtnModulo2 ().doClick();
            JOptionPane.showMessageDialog(
                this,
                "El registro ha sido actualizado",
                mensaje,
                JOptionPane.INFORMATION_MESSAGE );
        }
    } catch ( SQLException ex ) {
        dialogoMensaje ( ex.toString () );
    }
    dispose ();
}

//-----

private boolean validarDatos () {
    String idmateria = jtxfIDMateria.getText ();
    if ( idmateria.trim ().equals ( "" ) ) {
        dialogoMensaje ( "No se permite un valor en blanco" );
        jtxfIDMateria.requestFocus ();
        return false;
    }

    String materia = jtxfMateria.getText ();
    if ( materia.trim ().equals ( "" ) ) {
        dialogoMensaje ( "No se permite un valor en blanco" );
        jtxfMateria.requestFocus ();
        return false;
    }

    int creditos = 0;
    try {
        creditos = Integer.parseInt ( jsnpCreditos.getValue ().toString() );
    } catch ( NumberFormatException ex ) {
        dialogoMensaje ( "Debe capturar un valor numerico valido" );
        jsnpCreditos.requestFocus ();
        return false;
    }

    modelo = new Modelo2 ( idmateria, materia, creditos );
}
```

```

        return true;
    }

    //-----

    private void dialogoMensaje ( String mensaje ) {
        JOptionPane.showMessageDialog ( this, mensaje, "Error", JOptionPane.ERROR_MESSAGE );
    }

    //-----

    public static void ejecutar () {
        /* Set the Windows look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
                javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Windows".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

        }

        java.util.logging.Logger.getLogger(Modulo2EditarDialog.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
        } catch (InstantiationException ex) {

        }

        java.util.logging.Logger.getLogger(Modulo2EditarDialog.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
        } catch (IllegalAccessException ex) {

        }

        java.util.logging.Logger.getLogger(Modulo2EditarDialog.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

        }

        java.util.logging.Logger.getLogger(Modulo2EditarDialog.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
        }
    }
    //</editor-fold>

    /* Create and display the dialog */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            Modulo2EditarDialog dialog = new Modulo2EditarDialog(new javax.swing.JFrame(), null);
            dialog.addWindowListener(new java.awt.event.WindowAdapter() {
                @Override
                public void windowClosing(java.awt.event.WindowEvent e) {
                    System.exit(0);
                }
            });
            dialog.setVisible(true);
        }
    });
}

//-----

public static void main(String args[]) {
    ejecutar ();
}

//-----

```

Modulo3EditarDialog.java

```

/*-----
*
*          INSTITUTO TECNOLÓGICO DE LA LAGUNA
*          INGENIERÍA EN SISTEMAS COMPUTACIONALES
*          TEMAS AVANZADOS DE PROGRAMACIÓN "B"
*
*          SEMESTRE: ENE-JUN/2020      HORA: 17-18 HRS
*
*          Aplicación GUI para editar o crear registros en el Módulo 3
*
*          Archivo      : Modulo3EditarDialog.java

```

```

:* Autor      : José Misael Adame Sandoval      18131209
:*            : Ricardo Raúl Castro Luna       18131227
:*            : Jorge Arturo Galindo Uribe     18131238
:* Fecha      : 24/May/2020
:* Compilador : JAVA J2SE v1.8.0
:* Descripción : Aplicación que crea o edita un registro de la tabla Kardex con los
:*             siguientes datos:
:*             1. Alumno
:*             2. Materia
:*             3. Periodo
:*             4. Calificación
:*
:*
:* Última modif:
:* Fecha      Modificó      Motivo
:*=====
:* 24/May/2020 Misael Adame      Agregar prólogo.
:*-----*/

```

```
package app;
```

```

import mx.edu.itl.jdbc.EjecutorSQL;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Vector;
import javax.swing.Icon;
import javax.swing.JOptionPane;
import modelo.Modelo1;
import modelo.Modelo2;
import modelo.Modelo3;
import mx.edu.itl.util.Imagenes;

```

```
public class Modulo3EditarDialog extends javax.swing.JDialog {
```

```
//-----
```

```

private PrincipalFrame frmPrincipal;
private Modelo3 modelo;
private String accion;
private Vector<String> vecTiposColumnas;
private String sql;

```

```
//-----
```

```

public Modulo3EditarDialog(java.awt.Frame parent, Modelo3 modelo ) {
    super ( parent, true );
    initComponents ();

    frmPrincipal = ( PrincipalFrame ) parent;
    this.modelo = modelo;
    vecTiposColumnas = frmPrincipal.getVecTiposColumnas ();

    accion = ( modelo == null )? PrincipalFrame.NUEVO : PrincipalFrame.EDITAR;
    setTitle ( accion );

    Icon kardex = Imagenes.escalarImagen ( lblLogo.getIcon(),
                                           lblLogo.getWidth(),
                                           lblLogo.getHeight() );

    lblLogo.setIcon ( kardex );

    llenarCombo1DesdeTabla ();
    llenarCombo2DesdeTabla ();
    inicializarFormulario ();
}

```

```
//-----
```

```

private void llenarCombo1DesdeTabla () {
    ResultSet rs;
    Modelo1 modelo1;

    // Llenado del comboBox de alumnos
    sql = frmPrincipal.getPropConsultasSQL ()
        .getProperty ( PrincipalFrame.ALUMNOS_TODOS_POR_NOMBRE );
    try {
        rs = EjecutorSQL.sqlQuery ( sql );

        jcbAlumno.addItem ( new Modelo1 ( "", "--Seleccione--", "", 0, 0.0f ) );
        while ( rs.next () ) {
            String matricula = rs.getString ( "matricula" );

```

```
String nombre = rs.getString ( "nombre" );
String apellidos = rs.getString ( "apellidos" );

modelo1 = new Modelo1 ( matricula, nombre, apellidos, 0, 0.0f );
jcboAlumno.addItem ( modelo1 );

if ( modelo != null &&
    matricula.equals ( modelo.getMatricula () ) ) {
    int index = jcboAlumno.getModel ().getSize ();
    jcboAlumno.setSelectedIndex ( index - 1 );
}
}
rs.close ();

} catch (SQLException ex) {
    dialogoMensaje ( ex.toString() );
}
}

//-----

private void llenarCombo2DesdeTabla () {
    ResultSet rs;
    Modelo2 modelo2;

    // Llenado del comboBox de materias
    sql = frmPrincipal.getPropConsultasSQL ()
        .getProperty ( PrincipalFrame.MATERIAS_TODOS_POR_MATERIA );
    try {
        rs = EjecutorSQL.sqlQuery ( sql );

        jcboMateria.addItem ( new Modelo2 ( "", "--Seleccione--", 0 ) );
        while ( rs.next () ) {
            String idmateria = rs.getString ( "idmateria" );
            String materia = rs.getString ( "materia" );

            modelo2 = new Modelo2 ( idmateria, materia, 0 );
            jcboMateria.addItem ( modelo2 );

            if ( modelo != null &&
                idmateria.equals ( modelo.getIdmateria () ) ) {
                int index = jcboMateria.getModel ().getSize ();
                jcboMateria.setSelectedIndex ( index - 1 );
            }
        }
        rs.close ();

    } catch (SQLException ex) {
        dialogoMensaje ( ex.toString() );
    }
}

//-----

private void inicializarFormulario () {
    if ( accion.equals ( PrincipalFrame.NUEVO ) ) {

    } else if ( accion.equals ( PrincipalFrame.EDITAR ) ) {
        jcboPeriodo.setSelectedItem ( modelo.getPeriodo() );
        jftxfCalificacion.setText ( modelo.getCalificacion () + "" );
    }
    jcboAlumno.requestFocus ();
}

//-----

private void dialogoMensaje ( String mensaje ) {
    JOptionPane.showMessageDialog ( this, mensaje, "Error", JOptionPane.ERROR_MESSAGE );
}

//-----

private void jbbtnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    dispose ();
}

//-----

private void jbbtnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
    Object [][] args = null;
    String mensaje = "";
}
```

```

if ( validarDatos () == false )
    return;

// Se determina la sentencia SQL a ejecutar y formar la matriz de argumentos
if ( accion.equals ( PrincipalFrame.NUEVO ) ) {
    mensaje = "El registro ha sido agregado";
    sql = frmPrincipal.getPropConsultasSQL().getProperty (
        PrincipalFrame.KARDEX_INSERTA_NUEVO );
    args = new Object [][] {
        { vecTiposColumnas.elementAt ( 0 ), modelo.getIdcalific ( ) },
        { vecTiposColumnas.elementAt ( 1 ), modelo.getMatricula ( ) },
        { vecTiposColumnas.elementAt ( 4 ), modelo.getIdmateria ( ) },
        { vecTiposColumnas.elementAt ( 6 ), modelo.getPeriodo ( ) },
        { vecTiposColumnas.elementAt ( 7 ), modelo.getCalificacion ( ) }
    };

} else if ( accion.equals ( PrincipalFrame.EDITAR ) ) {
    mensaje = "El registro ha sido actualizado";
    sql = frmPrincipal.getPropConsultasSQL().getProperty (
        PrincipalFrame.KARDEX_ACTUALIZA_DATOS );
    args = new Object [][] {
        { vecTiposColumnas.elementAt ( 1 ), modelo.getMatricula ( ) },
        { vecTiposColumnas.elementAt ( 4 ), modelo.getIdmateria ( ) },
        { vecTiposColumnas.elementAt ( 6 ), modelo.getPeriodo ( ) },
        { vecTiposColumnas.elementAt ( 7 ), modelo.getCalificacion ( ) },
        { vecTiposColumnas.elementAt ( 0 ), modelo.getIdcalific ( ) }
    };
}

try {
    int regs = EjecutorSQL.sqlEjecutar ( sql, args );
    if ( regs == 1 ) {
        frmPrincipal.getJbtnModulo3 ().doClick();
        JOptionPane.showMessageDialog(
            this,
            "El registro ha sido actualizado",
            mensaje,
            JOptionPane.INFORMATION_MESSAGE );
    }
} catch ( SQLException ex ) {
    dialogoMensaje ( ex.toString () );
}
dispose ();
}

//-----

private boolean validarDatos () {
    Modelo1 modelo1 = ( Modelo1 ) jcbAlumno.getSelectedItem ();
    String matricula = modelo1.getMatricula ();
    String nombre = modelo1.getNombre ();
    if ( nombre.equals ( "--Seleccione--" ) ) {
        dialogoMensaje ( "Debe seleccionar un elemento de la lista" );
        jcbAlumno.requestFocus ();
        return false;
    }

    Modelo2 modelo2 = ( Modelo2 ) jcbMateria.getSelectedItem ();
    String idmateria = modelo2.getIdmateria ();
    String materia = modelo2.getMateria ();
    if ( materia.equals ( "--Seleccione--" ) ) {
        dialogoMensaje ( "Debe deseleccionar un elemento de la lista" );
        jcbMateria.requestFocus ();
        return false;
    }

    int calificacion = 0;
    try {
        calificacion = Integer.parseInt ( jftxfCalificacion.getText () );
        if ( calificacion < 0 || calificacion > 100 ) {
            dialogoMensaje ( "Debe capturar un valor numerico valido" );
            jftxfCalificacion.requestFocus ();
            return false;
        }
    } catch ( NumberFormatException ex ) {
        dialogoMensaje ( "Debe capturar un valor numerico valido" );
        jftxfCalificacion.requestFocus ();
        return false;
    }
}

```

```

String periodo = jcboPeriodo.getSelectedItem ().toString ();

int idcalific = 0;
if ( accion.equals ( PrincipalFrame.NUEVO ) ) {
    ResultSet rs;

    // Determinar el consecutivo del ID CALIFICACION e incrementarlo en 1
    sql = frmPrincipal.getPropConsultasSQL().getProperty( PrincipalFrame.KARDEX_OBT_MAX_ID );
    try {
        rs = EjecutorSQL.sqlQuery ( sql );
        if ( rs.next() ) {
            idcalific = rs.getInt ( "maxid" );
            idcalific++;
        }
    } catch ( SQLException ex ) {
        dialogoMensaje ( ex.toString () );
    }
} else if ( accion.equals ( PrincipalFrame.EDITAR ) ) {
    idcalific = modelo.getIdcalific ();
}

modelo = new Modelo3 ( idcalific, matricula, idmateria, periodo, calificacion );
return true;
}

//-----

public static void ejecutar () {
    /* Set the Windows look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
    * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
        javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Windows".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {

        java.util.logging.Logger.getLogger(Modulo3EditarDialog.class.getName()).log(java.util.logging.Level.SEVERE,
        null, ex);
    } catch (InstantiationException ex) {

        java.util.logging.Logger.getLogger(Modulo3EditarDialog.class.getName()).log(java.util.logging.Level.SEVERE,
        null, ex);
    } catch (IllegalAccessException ex) {

        java.util.logging.Logger.getLogger(Modulo3EditarDialog.class.getName()).log(java.util.logging.Level.SEVERE,
        null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

        java.util.logging.Logger.getLogger(Modulo3EditarDialog.class.getName()).log(java.util.logging.Level.SEVERE,
        null, ex);
    }
    //</editor-fold>

    /* Create and display the dialog */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            Modulo3EditarDialog dialog = new Modulo3EditarDialog(new javax.swing.JFrame(), null);
            dialog.addWindowListener(new java.awt.event.WindowAdapter() {
                @Override
                public void windowClosing(java.awt.event.WindowEvent e) {
                    System.exit(0);
                }
            });
            dialog.setVisible(true);
        }
    });
}

//-----

public static void main(String args[]) {
    ejecutar ();
}

```



```
//-----
```

FiltrarDialog.java

```
/*-----
*:
*: INSTITUTO TECNOLÓGICO DE LA LAGUNA
*: INGENIERÍA EN SISTEMAS COMPUTACIONALES
*: TEMAS AVANZADOS DE PROGRAMACIÓN "B"
*:
*: SEMESTRE: ENE-JUN/2020 HORA: 17-18 HRS
*:
*: Aplicación GUI para el filtrado de consultas
*:
*: Archivo : FiltrarDialog.java
*: Autor : José Misael Adame Sandoval 18131209
*: Ricardo Raúl Castro Luna 18131227
*: Jorge Arturo Galindo Uribe 18131238
*: Fecha : 24/May/2020
*: Compilador : JAVA J2SE v1.8.0
*: Descripción : Aplicación que realiza un filtrado de consultas SQL de un módulo
*: seleccionado con los siguientes campos:
*: 1. Columna
*: 2. Comparador
*: 3. Valor
*:
*:
*: Última modif:
*: Fecha Modificó Motivo
*:-----
*: 24/May/2020 Misael Adame Agregar prólogo.
*:-----*/

package app;

import mx.edu.itl.jdbc.EjecutorSQL;
import java.util.Vector;
import javax.swing.JOptionPane;

public class FiltrarDialog extends javax.swing.JDialog {

    //-----

    private PrincipalFrame frmPrincipal;
    private Vector<String> vecColumnas;
    private Vector<String> vecColumnasBD;
    private Vector<String> vecTipos;
    private boolean inicializando = true;

    //-----

    public FiltrarDialog ( java.awt.Frame parent, boolean modal ) {
        super ( parent, modal );
        initComponents ();

        frmPrincipal = (PrincipalFrame) parent;
        vecColumnas = frmPrincipal.getVecNombresColumnas ();
        vecColumnasBD = frmPrincipal.getVecNombresColumnasBD ();
        vecTipos = frmPrincipal.getVecTiposColumnas ();

        jcboColumna.removeAllItems ();
        for ( int i = 0; i < vecColumnas.size (); i++ ) {
            jcboColumna.addItem ( vecColumnas.elementAt ( i ) );
        }

        cargarComparadores ( 0 );
        jcboColumna.setSelectedIndex ( 0 );
        jcboColumna.requestFocus ();
        inicializando = false;
    }

    //-----

    private void cargarComparadores ( int pos ) {
        jcboComparador.removeAllItems ();
        switch ( vecTipos.elementAt ( pos ) ) {
            case EjecutorSQL.STRING:
                jcboComparador.addItem ( "Sea igual a" );
        }
    }
}
```

```

        jcboComparador.addItem ( "Empiece con" );
        jcboComparador.addItem ( "Termine con" );
        jcboComparador.addItem ( "Contenga" );
        jcboComparador.addItem ( "Sea diferente" );
        break;
    case EjecutorSQL.INT:
    case EjecutorSQL.FLOAT:
        jcboComparador.addItem ( "=" );
        jcboComparador.addItem ( "<>" );
        jcboComparador.addItem ( ">" );
        jcboComparador.addItem ( "<" );
        jcboComparador.addItem ( ">=" );
        jcboComparador.addItem ( "<=" );
        break;
    }
    jcboComparador.setSelectedIndex ( 0 );
}

//-----

private void jbbtnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
    dispose ();
}

//-----

private void jbbtnAceptarActionPerformed(java.awt.event.ActionEvent evt) {
    if ( jtxfValor.getText ().trim ().equals ( "" ) ) {
        JOptionPane.showMessageDialog (
            this, "Proporcione un valor", "Filtrar", JOptionPane.ERROR_MESSAGE );
        jtxfValor.requestFocus ();
        return;
    }

    String sql = "";
    String orderBy = "";
    switch ( frmPrincipal.getModuloActual () ) {
        case PrincipalFrame.TIT_MODULO1:
            sql = frmPrincipal.getPropConsultasSQL().getProperty (
                PrincipalFrame.ALUMNOS_TODOS_SIN_ORDEN
            );
            orderBy = " ORDER BY nombre ";
            break;
        case PrincipalFrame.TIT_MODULO2:
            sql = frmPrincipal.getPropConsultasSQL().getProperty (
                PrincipalFrame.MATERIAS_TODOS_SIN_ORDEN
            );
            orderBy = " ORDER BY materia ";
            break;
        case PrincipalFrame.TIT_MODULO3:
            sql = frmPrincipal.getPropConsultasSQL().getProperty (
                PrincipalFrame.KARDEX_TODOS_SIN_ORDEN
            );
            orderBy = " ORDER BY matricula ";
            break;
    }

    sql += " WHERE " + vecColumnasBD.elementAt ( jcboColumna.getSelectedIndex () );

    String tipo = vecTipos.elementAt ( jcboColumna.getSelectedIndex () );
    String valor = jtxfValor.getText ();

    switch ( jcboComparador.getSelectedItem ().toString () ) {
        case "Sea igual a":
            sql += " = ? ";
            break;
        case "Empiece con":
            sql += " LIKE ? ";
            valor += "%";
            break;
        case "Termine con":
            sql += " LIKE ? ";
            valor = "%" + valor;
            break;
        case "Contenga":
            sql += " LIKE ? ";
            valor = "%" + valor + "%";
            break;
        case "Sea diferente":
            sql += " <> ? ";
    }
}

```

```

        break;
    default:
        sql += " " + jcboComparador.getSelectedItem().toString () + " ? ";
    }

    sql += orderBy;

    Object [][] args = { { tipo, valor } };
    frmPrincipal.desplegarRegistros ( sql, args );
    dispose ();
}

//-----

private void jcboColumnaItemStateChanged(java.awt.event.ItemEvent evt) {
    if ( ! inicializando ) {
        cargarComparadores ( jcboColumna.getSelectedIndex () );
    }
}

//-----

public static void ejecutar () {
    /* Set the Windows look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Windows".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(FiltrarDialog.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(FiltrarDialog.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(FiltrarDialog.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(FiltrarDialog.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
    }
    //</editor-fold>

    /* Create and display the dialog */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            FiltrarDialog dialog = new FiltrarDialog(new javax.swing.JFrame(), true);
            dialog.addWindowListener(new java.awt.event.WindowAdapter() {
                @Override
                public void windowClosing(java.awt.event.WindowEvent e) {
                    System.exit(0);
                }
            });
            dialog.setVisible(true);
        }
    });
}

//-----

public static void main ( String args[] ) {
    ejecutar ();
}

//-----

```

AcercaDialog.java

```

/*-----

```

```

:*          INSTITUTO TECNOLÓGICO DE LA LAGUNA
:*          INGENIERÍA EN SISTEMAS COMPUTACIONALES
:*          TEMAS AVANZADOS DE PROGRAMACIÓN "B"
:*
:*          SEMESTRE: ENE-JUN/2020      HORA: 17-18 HRS
:*
:*          Aplicación GUI de un Acerca de..
:*
:* Archivo      : AcercaDeDialog.java
:* Autor        : José Misael Adame Sandoval      18131209
:*              : Ricardo Raúl Castro Luna        18131227
:*              : Jorge Arturo Galindo Uribe      18131238
:* Fecha        : 24/May/2020
:* Compilador   : JAVA J2SE vl.8.0
:* Descripción  : Aplicación con el AcercaDeBean que muestra información sobre los autores
:*              : de la aplicación U3BDCalificacionesApp
:*
:*
:* Última modif:
:* Fecha        Modificó          Motivo
:*=====
:* 24/May/2020 Misael Adame      Agregar prólogo.
:*-----*/

package app;

public class AcercaDeDialog extends javax.swing.JDialog {

    //-----

    public AcercaDeDialog ( java.awt.Frame parent, boolean modal ) {
        super ( parent, modal );
        initComponents ();
    }

    //-----
    //-----

    public static void ejecutar () {
        /* Set the Windows look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
                javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Windows".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

        }
        java.util.logging.Logger.getLogger(AcercaDeDialog.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
        } catch (InstantiationException ex) {

        }
        java.util.logging.Logger.getLogger(AcercaDeDialog.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
        } catch (IllegalAccessException ex) {

        }
        java.util.logging.Logger.getLogger(AcercaDeDialog.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

        }
        java.util.logging.Logger.getLogger(AcercaDeDialog.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
        }
        //</editor-fold>

        /* Create and display the dialog */
        java.awt.EventQueue.invokeLater(new Runnable () {
            public void run () {
                AcercaDeDialog dialog = new AcercaDeDialog(new javax.swing.JFrame(), true);
                dialog.addWindowListener(new java.awt.event.WindowAdapter() {
                    @Override
                    public void windowClosing(java.awt.event.WindowEvent e) {
                        System.exit(0);
                    }
                });
            }
        });
    }

```

```

        dialog.setVisible ( true );
    }
}

//-----

public static void main ( String args[] ) {
    ejecutar ();
}

//-----

```

ConexionDB.java

```

/*-----
*:          INSTITUTO TECNOLÓGICO DE LA LAGUNA
*:          INGENIERÍA EN SISTEMAS COMPUTACIONALES
*:          TEMAS AVANZADOS DE PROGRAMACIÓN "B"
*:
*:          SEMESTRE: ENE-JUN/2020      HORA: 17-18 HRS
*:
*:          Clase que conecta a una Base de Datos
*:
*: Archivo      : ConexionDB.java
*: Autor        : José Misael Adame Sandoval      18131209
*:              Ricardo Raúl Castro Luna         18131227
*:              Jorge Arturo Galindo Uribe       18131238
*: Fecha        : 24/May/2020
*: Compilador   : JAVA J2SE v1.8.0
*: Descripción  : Clase que permite conectarse a diversas bases de datos como:
*:                - Java DB
*:                - SQL Server
*:                - MySQL
*:                - Oracle XE
*:                - Access
*:
*: Última modif:
*: Fecha      Modificó      Motivo
*:-----
*: 24/May/2020 Misael Adame      Agregar prólogo.
*:-----*/

package mx.edu.itl.jdbc;

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

public class ConexionDB {

    //-----

    public static final String JAVADB = "Java DB";
    public static final String SQLSERVER = "SQL Server";
    public static final String MYSQL = "MySQL";
    public static final String ORACLEXE = "Oracle XE";
    public static final String ACCESS = "MS Access";

    //-----

    private static ConexionDB instancia = null;
    private Connection conexion;

    //-----

    private static String driverJDBC = "";
    private static String urlConexion = "";
    private static String servidor = "";
    private static String puerto = "";
    private static String baseDeDatos = "";
    private static String usuario = "";
    private static String contrasena = "";

    //-----

    public static ConexionDB getInstancia () {
        if ( instancia == null )
            instancia = new ConexionDB ();
        return instancia;
    }
}

```

```

}

//-----

public void conectar ( String dbms, String servidor, String puerto, String bd,
                      String usuario, String contraseña ) {

    ConexionDB.servidor = servidor;
    ConexionDB.puerto = puerto;
    ConexionDB.baseDeDatos = bd;
    ConexionDB.usuario = usuario;
    ConexionDB.contrasena = contraseña;

    if ( dbms.equals ( JAVADB ) ) {
        driverJDBC = "org.apache.derby.jdbc.ClientDriver";
        urlConexion = "jdbc:derby://" + servidor + ":" + puerto + "/";
    } else if ( dbms.equals ( SQLSERVER ) ) {
        driverJDBC = "com.microsoft.sqlserver.jdbc.SQLServerDriver";
        urlConexion = "jdbc:sqlserver://" + servidor + ":" + puerto + ";databaseName=";
    } else if ( dbms.equals ( MYSQL ) ) {
        driverJDBC = "com.mysql.cj.jdbc.Driver";
        urlConexion = "jdbc:mysql://" + servidor + ":" + puerto + "?useJDBCCompliantTimezone"
            + "Shift=true&useLegacyDatetimeCode=false&serverTimezone=UTC&";
    } else if ( dbms.equals ( ORACLE ) ) {
        driverJDBC = "oracle.jdbc.OracleDriver";
        urlConexion = "jdbc:oracle:thin:@" + servidor + ":" + puerto + ":XE";
    } else if ( dbms.equals ( ACCESS ) ) {
        driverJDBC = "net.ucanaccess.jdbc.UcanaccessDriver";
        urlConexion = "jdbc:ucanaccess://";
    }

    // Intentar hacer la conexión con la BD
    conexion = null;
    try {
        // Realizar la conexión a la DB
        Class.forName ( driverJDBC );

        conexion = DriverManager.getConnection ( urlConexion + baseDeDatos,
                                                usuario, contraseña );
    } catch ( Exception ex ) {
        System.out.println ( ex );
    }
}

//-----

public Connection getConexion () {
    return conexion;
}

public void desconectar () {
    if ( conexion != null ) {
        try {
            conexion.close();
        } catch ( SQLException ex ) {
            System.out.println ( ex );
        }
    }
}

//-----

public boolean conectado () {
    if ( conexion != null ) {
        return true;
    }
    return false;
}

//-----
}

```

EjecutorSQL.java

```

/*-----
*:          INSTITUTO TECNOLÓGICO DE LA LAGUNA
*:          INGENIERÍA EN SISTEMAS COMPUTACIONALES
*:          TEMAS AVANZADOS DE PROGRAMACIÓN "B"
*:
*:          SEMESTRE: ENE-JUN/2020      HORA: 17-18 HRS
*:
*/

```

```

:*                               Clase que ejecuta sentencias SQL
:*
:* Archivo      : EjecutorSQL.java
:* Autor       : José Misael Adame Sandoval      18131209
:*             Ricardo Raúl Castro Luna        18131227
:*             Jorge Arturo Galindo Uribe      18131238
:* Fecha        : 24/May/2020
:* Compilador   : JAVA J2SE v1.8.0
:* Descripción  : Clase que prepara y ejecuta sentencias SQL ya sea de manera directa
:*               o parametrizada, es decir que sustituye argumentos en este caso de
:*               tipo INT, FLOAT o STRING.
:*
:*
:*
:* Última modif:
:* Fecha      Modificó      Motivo
:*=====
:* 24/May/2020 Misael Adame      Agregar prólogo.
:*-----*/

package mx.edu.itl.jdbc;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class EjecutorSQL {

    //-----

    public static final String INT = "int";
    public static final String FLOAT = "float";
    public static final String STRING = "string";

    //-----
    // Ejecución de una sentencia sql directa, no parametrizada. es decir que
    // no requiere sustituir argumentos en ella

    public static ResultSet sqlQuery ( String sql ) throws SQLException {
        return sqlQuery ( sql, null );
    }

    //-----

    public static ResultSet sqlQuery ( String sql, Object [][] args ) throws SQLException {

        PreparedStatement ps = ConexionDB.getInstancia ().getConnection ()
        .prepareStatement ( sql );

        prepararArgumentos ( ps, args );
        return ps.executeQuery ();
    }

    //-----

    public static int sqlEjecutar ( String sql ) throws SQLException {
        return sqlEjecutar ( sql, null );
    }

    //-----

    public static int sqlEjecutar ( String sql, Object [][] args ) throws SQLException {
        PreparedStatement ps = ConexionDB.getInstancia ().getConnection ()
        .prepareStatement ( sql );

        prepararArgumentos ( ps, args );
        return ps.executeUpdate ();
    }

    //-----

    public static void prepararArgumentos ( PreparedStatement ps, Object [][] args )
        throws SQLException {

        final int TIPO = 0;
        final int VALOR = 1;

        // Verificar si se debe realizar la sustitución de parametros
        if ( args != null ) {
            int numArg = 1;
            for ( Object [] arg : args ) {
                switch ( arg [ TIPO ].toString() ) {

```

```

        case INT : ps.setInt (
                        numArg,
                        Integer.parseInt ( arg [ VALOR ].toString() )
                    );
        break;

        case FLOAT : ps.setFloat (
                        numArg,
                        Float.parseFloat ( arg [ VALOR ].toString() )
                    );
        break;

        case STRING : ps.setString (
                        numArg,
                        arg [ VALOR ].toString ()
                    );
        break;
    }
    numArg++;
}
}
}

//-----
}

```

ConexionDBFrame.java

```

/*-----
*:
*:          INSTITUTO TECNOLÓGICO DE LA LAGUNA
*:          INGENIERÍA EN SISTEMAS COMPUTACIONALES
*:          TEMAS AVANZADOS DE PROGRAMACIÓN "B"
*:
*:          SEMESTRE: ENE-JUN/2020      HORA: 17-18 HRS
*:
*:          Aplicación GUI que conecta a una Base de Datos
*:
*: Archivo      : ConexionDBFrame.java
*: Autor        : José Misael Adame Sandoval      18131209
*:              : Ricardo Raúl Castro Luna        18131227
*:              : Jorge Arturo Galindo Uribe      18131238
*: Fecha        : 24/May/2020
*: Compilador   : JAVA J2SE v1.8.0
*: Descripción  : Aplicación que accede a la clase ConexionDB y permite conectarse a una Base
*:                de Datos de manera gráfica en que el usuario ingresa los datos de conexión.
*:
*:
*: Última modif:
*: Fecha        Modificó          Motivo
*:-----
*: 24/May/2020 Misael Adame      Agregar prólogo.
*:-----*/

```

```

package mx.edu.itl.jdbc;

import java.awt.Frame;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JOptionPane;

public class ConexionDBFrame extends javax.swing.JFrame {

    //-----

    private Frame frmPadre = null;

    //-----

    public ConexionDBFrame () {
        initComponents ();
        // Inicializar la lista de DBMS
        jcboDBMS.addItem ( ConexionDB.JAVADB );
        jcboDBMS.addItem ( ConexionDB.ACCESS );
        jcboDBMS.addItem ( ConexionDB.MYSQL );
        jcboDBMS.addItem ( ConexionDB.SQLSERVER );
        jcboDBMS.addItem ( ConexionDB.ORACLEXE );

        jcboDBMS.addActionListener ( new ActionListener () {

            @Override
            public void actionPerformed ( ActionEvent e ) {

```



```

        if ( jcboDBMS.getSelectedIndex () == 4 ) {
            jLabelTextoUsuario.setText("Base de Datos:");

            jLabelTextoBD.setVisible ( false );
            jtxfBaseDeDatos.setVisible( false );

            jtxfServidor.setEnabled ( true );
            jtxfPuerto.setEnabled ( true );
            jtxfUsuario.setEnabled ( true );
            jpswContrasena.setEnabled ( true );

        } else if ( jcboDBMS.getSelectedIndex () == 1 ) {
            jLabelTextoUsuario.setText ( "Usuario:" );
            jLabelTextoBD.setText ( "URL Base de Datos:" );

            jtxfBaseDeDatos.setVisible ( true );
            jLabelTextoBD.setVisible ( true );

            jtxfServidor.setEnabled ( false );
            jtxfPuerto.setEnabled ( false );
            jtxfUsuario.setEnabled ( false );
            jpswContrasena.setEnabled ( false );

        } else {
            jLabelTextoUsuario.setText ( "Usuario:" );
            jLabelTextoBD.setText ( "Base de Datos:" );

            jLabelTextoBD.setVisible ( true );
            jtxfBaseDeDatos.setVisible ( true );

            jtxfServidor.setEnabled ( true );
            jtxfPuerto.setEnabled ( true );
            jtxfUsuario.setEnabled ( true );
            jpswContrasena.setEnabled ( true );
        }
    }
});
}

//-----

public ConexionDBFrame ( Frame padre ) {
    this ();
    frmPadre = padre;
}

//-----

private void jbbtnSalirActionPerformed(java.awt.event.ActionEvent evt) {
    ConexionDB.getInstancia ().desconectar ();
    dispose ();
}

//-----

private void jbbtnAceptarActionPerformed(java.awt.event.ActionEvent evt) {
    ConexionDB.getInstancia ().conectar (
        jcboDBMS.getSelectedItem ().toString (),
        jtxfServidor.getText (),
        jtxfPuerto.getText (),
        jtxfBaseDeDatos.getText (),
        jtxfUsuario.getText (),
        new String ( jpswContrasena.getPassword () )
    );

    if ( ConexionDB.getInstancia().conectado() ) {
        if ( frmPadre != null ) {
            frmPadre.setVisible ( true );
            dispose ();
        } else
            JOptionPane.showMessageDialog ( this, "Conectado!" );
    } else {
        JOptionPane.showMessageDialog ( this, "No se logró la conexión. Verifique la información." );
    }
}

//-----

public static void main ( String args[] ) {
    ejecutar ();
}

```

```
}

//-----

public static void ejecutar () {
    /* Set the Windows look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
            javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Windows".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {

        java.util.logging.Logger.getLogger(ConexionDBFrame.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
    } catch (InstantiationException ex) {

        java.util.logging.Logger.getLogger(ConexionDBFrame.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
    } catch (IllegalAccessException ex) {

        java.util.logging.Logger.getLogger(ConexionDBFrame.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

        java.util.logging.Logger.getLogger(ConexionDBFrame.class.getName()).log(java.util.logging.Level.SEVERE,
            null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new ConexionDBFrame().setVisible(true);
        }
    });
}

//-----
```

Anexo

```
INSERT INTO Alumnos VALUES ( '17130856', 'Olivia', 'Olivares Oliveira', 19, 78.25 );
INSERT INTO Alumnos VALUES ( '17130433', 'Hernan', 'Hernandez', 21, 82.87 );
INSERT INTO Alumnos VALUES ( '18130945', 'Choncho', 'Renigris', 20, 92.14 );
```

```
INSERT INTO Materias VALUES ('ED1', 'Estructura de Datos', 8 );
INSERT INTO Materias VALUES ('TAP1', 'Temas Av. de Progr.', 6 );
INSERT INTO Materias VALUES ('TBD1', 'Taller de B. de D.', 10 );
INSERT INTO Materias VALUES ('IA1', 'Inteligencia Artificial', 8 );
```

```
INSERT INTO Kardex VALUES ( 1, '17130856', 'TBD1', '2020A', 91 );
INSERT INTO Kardex VALUES ( 2, '17130856', 'ED1', '2020A', 75 );
INSERT INTO Kardex VALUES ( 3, '17130856', 'TAP1', '2020A', 81 );
INSERT INTO Kardex VALUES ( 4, '17130856', 'IA1', '2020A', 78 );
```

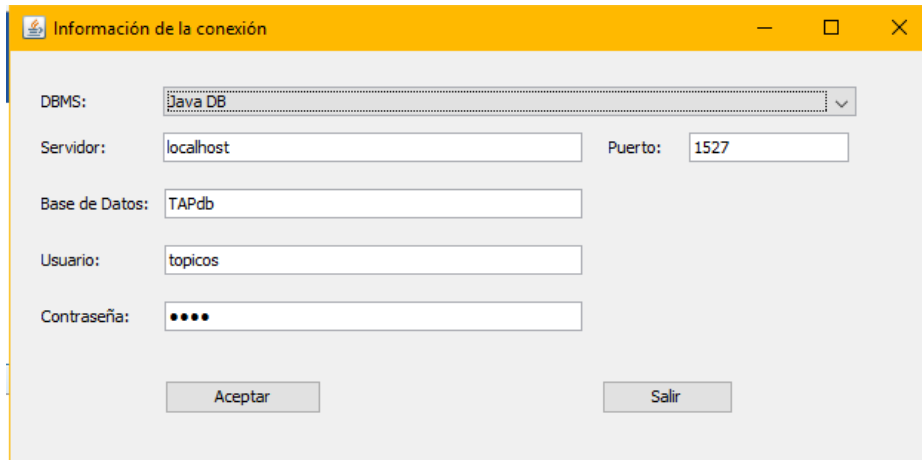
```
INSERT INTO Kardex VALUES ( 5, '17130433', 'TBD1', '2020A', 78 );
INSERT INTO Kardex VALUES ( 6, '17130433', 'ED1', '2020A', 81 );
INSERT INTO Kardex VALUES ( 7, '17130433', 'TAP1', '2020A', 75 );
INSERT INTO Kardex VALUES ( 8, '17130433', 'IA1', '2020A', 91 );
```

```
INSERT INTO Kardex VALUES ( 9, '18130945', 'TBD1', '2020A', 88 );
INSERT INTO Kardex VALUES ( 10, '18130945', 'ED1', '2020A', 86 );
INSERT INTO Kardex VALUES ( 11, '18130945', 'TAP1', '2020A', 94 );
INSERT INTO Kardex VALUES ( 12, '18130945', 'IA1', '2020A', 73 );
```

Prueba de Ejecución / Manual de usuario

Ingresar a la aplicación

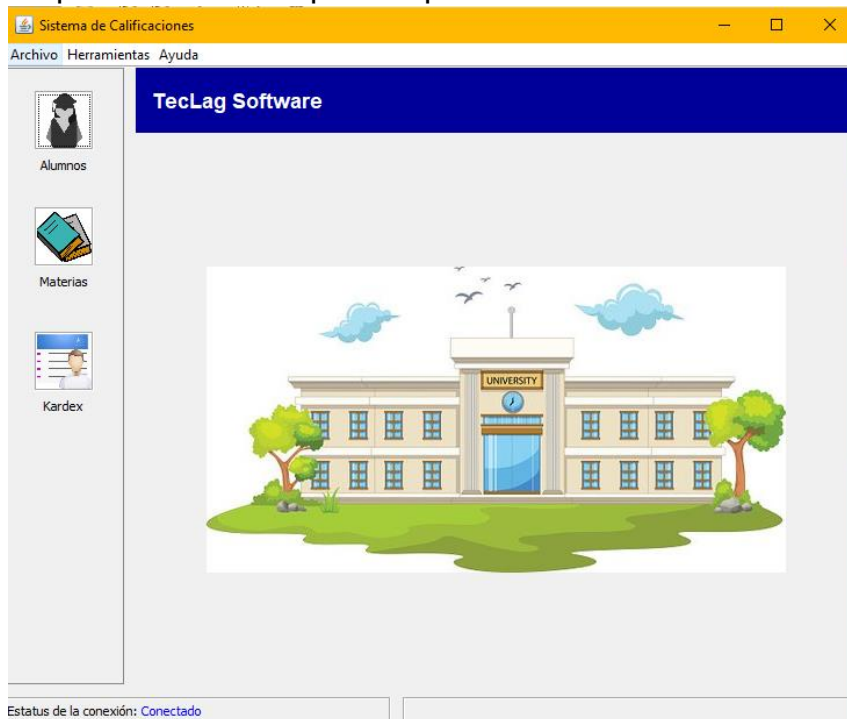
Al iniciar la aplicación, se mostrará un cuadro en el cual, el usuario deberá ingresar los datos de la conexión, como el servidor, la base de datos y las credenciales.



The screenshot shows a window titled "Información de la conexión" with a yellow header bar. It contains several input fields: "DBMS:" with a dropdown menu showing "Java DB"; "Servidor:" with a text box containing "localhost"; "Puerto:" with a text box containing "1527"; "Base de Datos:" with a text box containing "TAPdb"; "Usuario:" with a text box containing "topicos"; and "Contraseña:" with a masked text box showing four dots. At the bottom, there are two buttons: "Aceptar" and "Salir".

Menú principal

Una vez ingresados los datos, se llevará al menú inicial. En la parte superior, veremos 3 menús: Archivo, herramientas y ayuda. El menú lateral contiene los submenús para ir a los apartados principales: Alumno, Materia y Kardex. En la parte inferior de la pantalla, podemos ver el estado de la conexión.

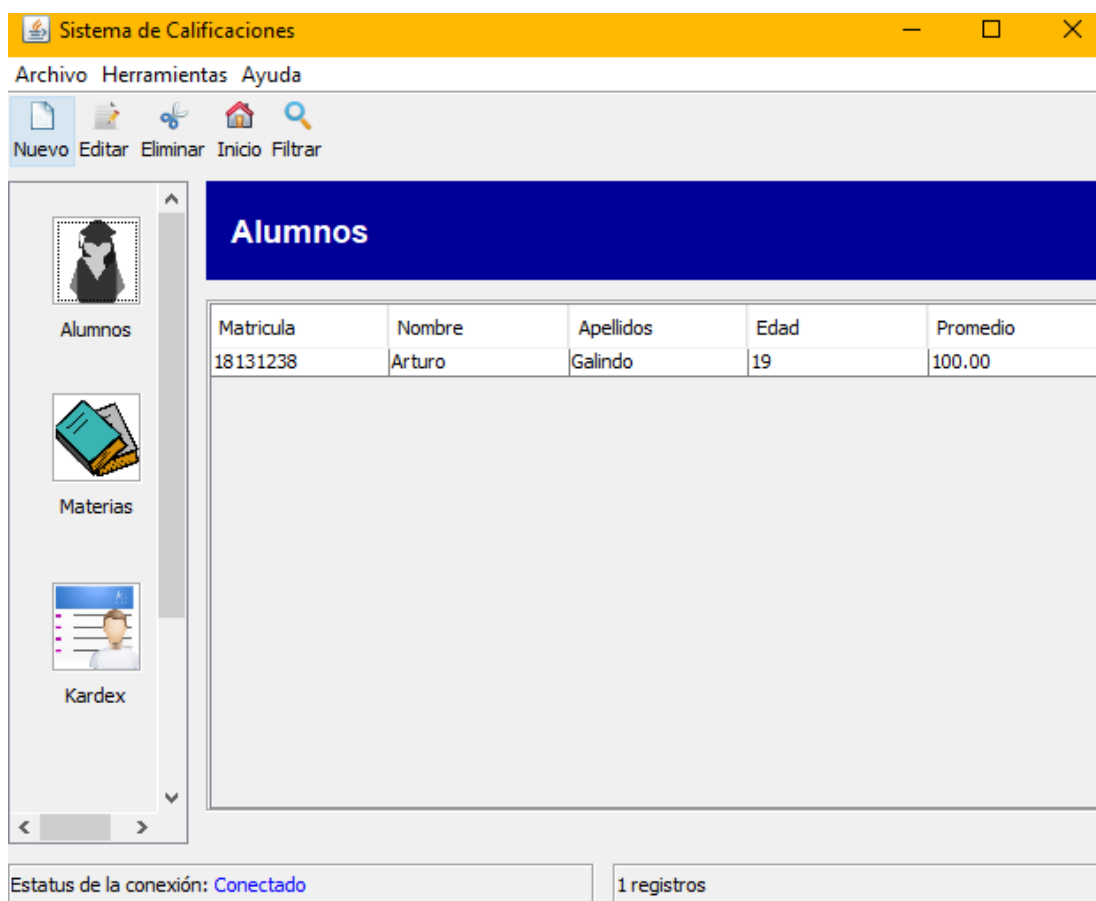


Submenú Alumnos

En esta pantalla, vemos una tabla donde podemos ver a los alumnos registrados con los siguientes datos:

- Matrícula
- Nombre
- Apellidos
- Edad
- Promedio

Así como un nuevo menú superior. En este menú, podremos ingresar un nuevo alumno, editar la información de uno ya existente, eliminar el registro de un alumno, volver al menú inicial y filtrar. En el menú filtrar, se podrán elegir según la columna (Edad, Promedio, Nombre), el comparador (Diferente a, igual a, que contenga) y el dato a buscar (nombre, número, letra).

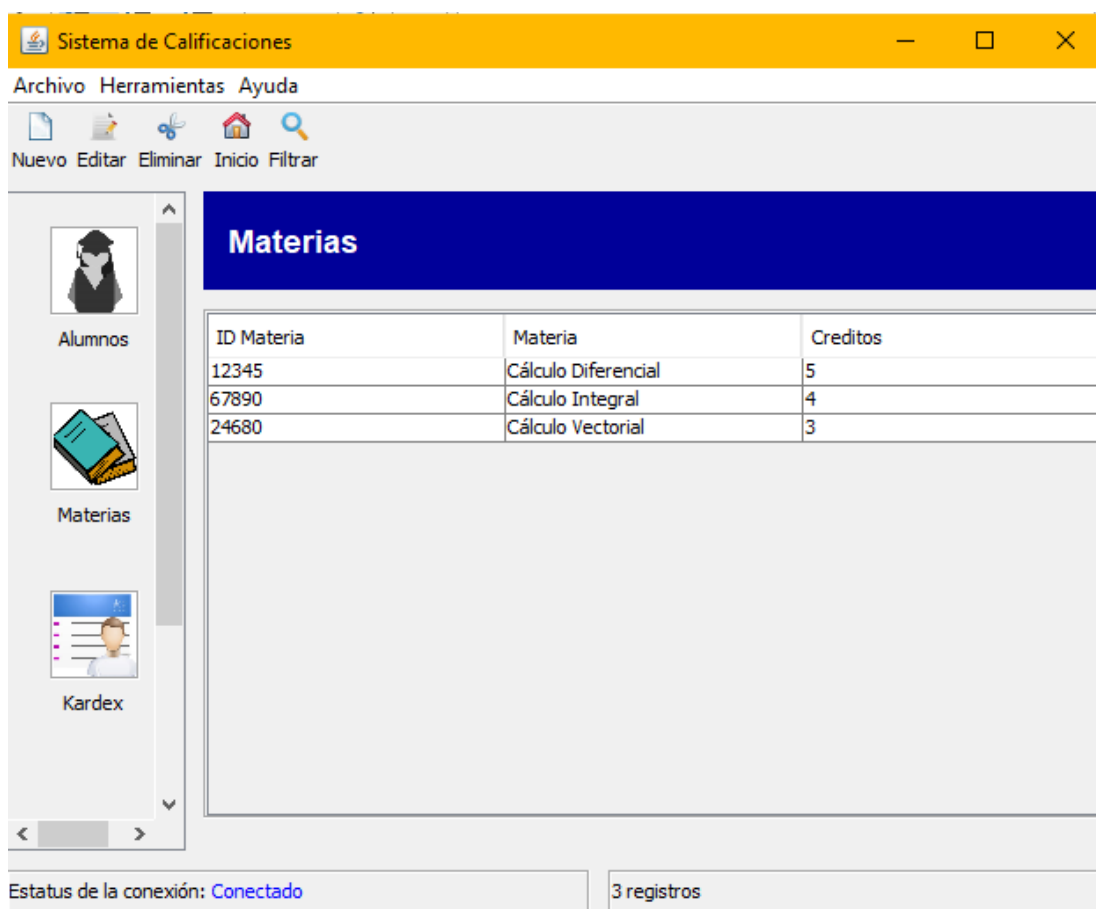


Submenú Materias

Al igual que con el menú anterior, nos encontramos con una tabla donde podemos ver a los datos de las materias registradas con la siguiente información:

- ID de la Materia
- Nombre de la Materia
- Número de Créditos
-

Así como un nuevo menú superior. En este menú, podremos ingresar una nueva materia, editar la información de una ya existente, eliminar el registro de una materia, volver al menú inicial y filtrar. En el menú filtrar, se podrán elegir según la columna (ID Materia, Nombre, No. Créditos), el comparador (Diferente a, igual a, que contenga) y el dato a buscar (nombre, número, letra).

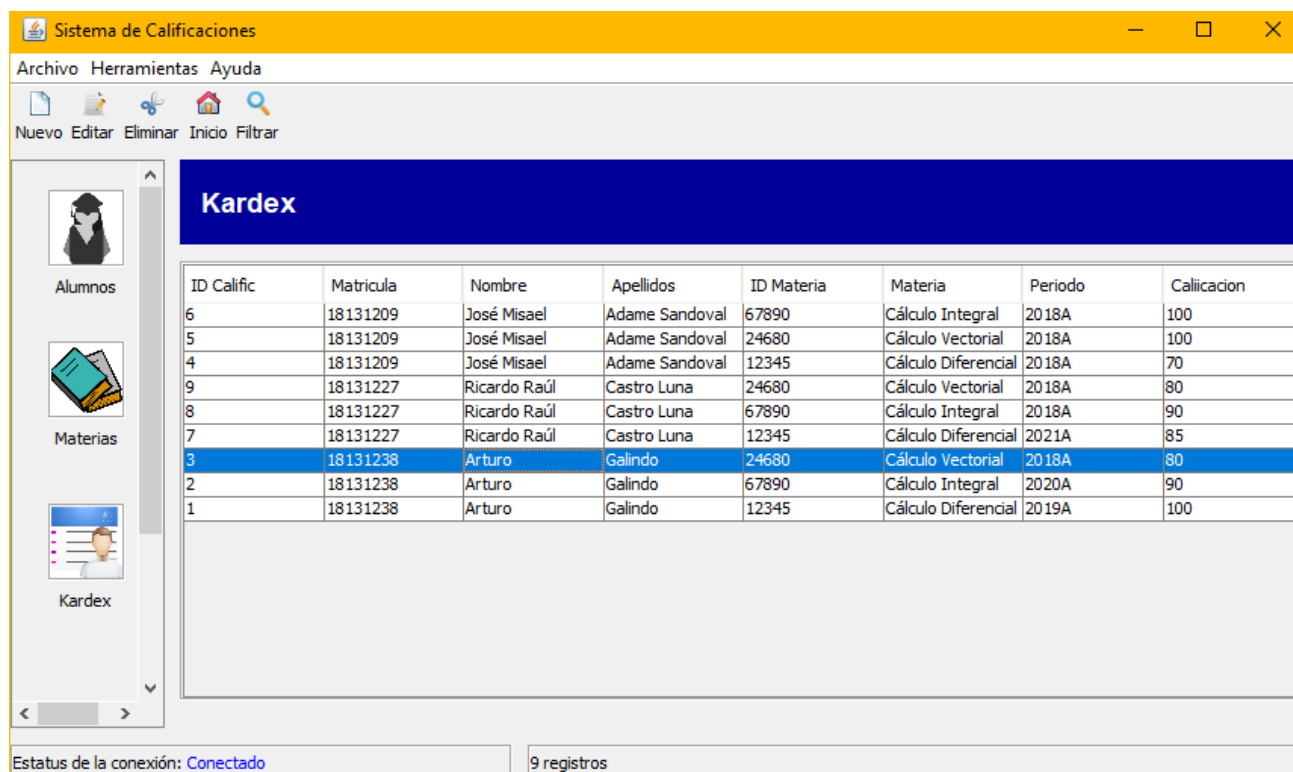


Submenú Kardex

En este menú, nuevamente encontraremos una tabla con la siguiente información:

- Matrícula (alumno)
- ID Materia
- Calificación
- Período

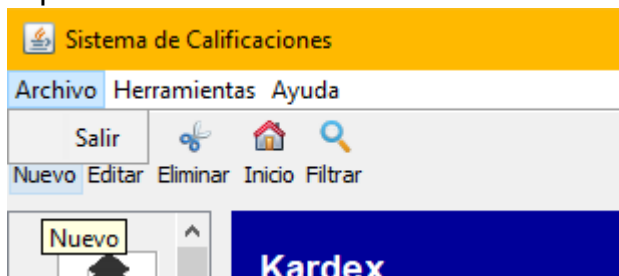
Esta página representa las calificaciones que han obtenido los alumnos en ciertas materias. Al igual que los menús anteriores, se cuenta con las opciones para crear un nuevo registro, modificar uno existente, eliminar uno existente, así como el de filtrar registros específicos.



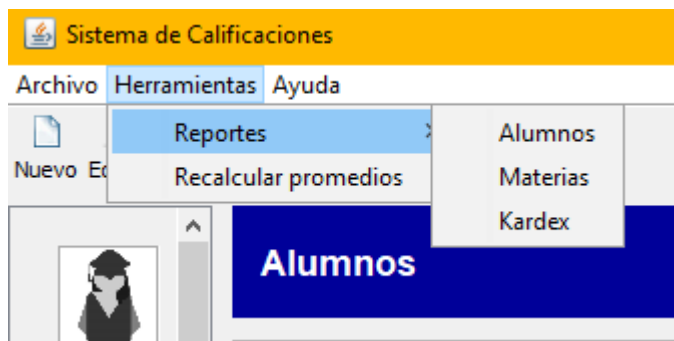
Submenú superior

En el menú superior, encontraremos 3 opciones disponibles:

- Archivo: Permite cerrar la aplicación con el botón “Salir”:



- **Herramientas:** En este menú tendremos 2 opciones: En la sección de reportes, podremos crear e imprimir un archivo con un reporte de la tabla especificada (Alumno, Materia, Kardex). La sección Recalcular promedios permite actualizar los promedios de cada alumno en base a las calificaciones registradas en el kardex.



- **Ayuda:** En este menú, se muestra la opción Acerca de... que nos permite ver la información del proyecto y de los integrantes del equipo:

