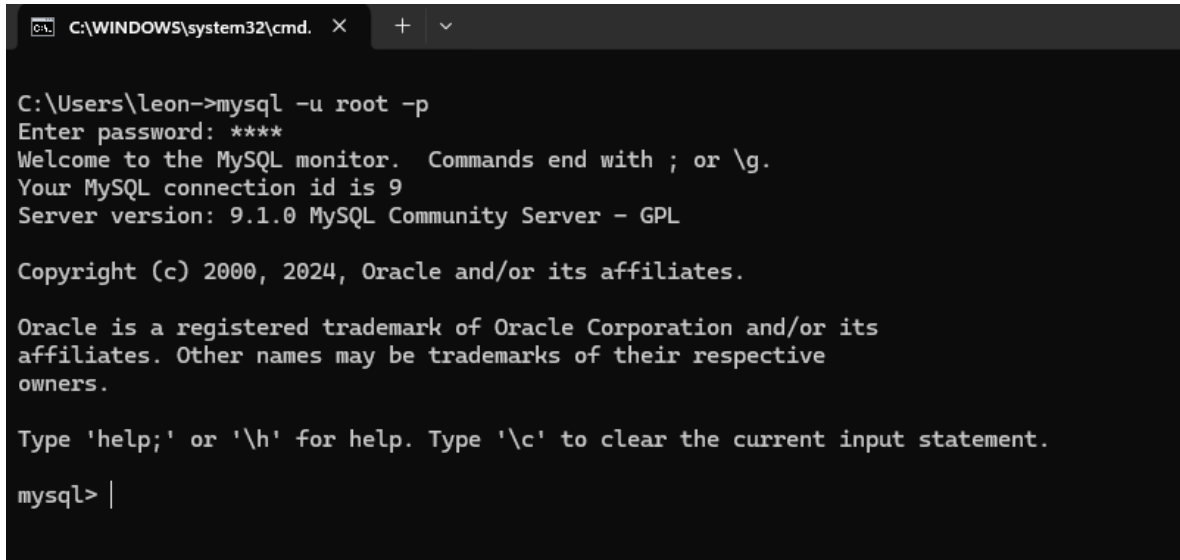


## Creacion de la base de datos utilizando mysql

Accedemos a mysql en cmd utilizando

`mysql -u root -p`



```
C:\WINDOWS\system32\cmd. X + v

C:\Users\leon->mysql -u root -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 9.1.0 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

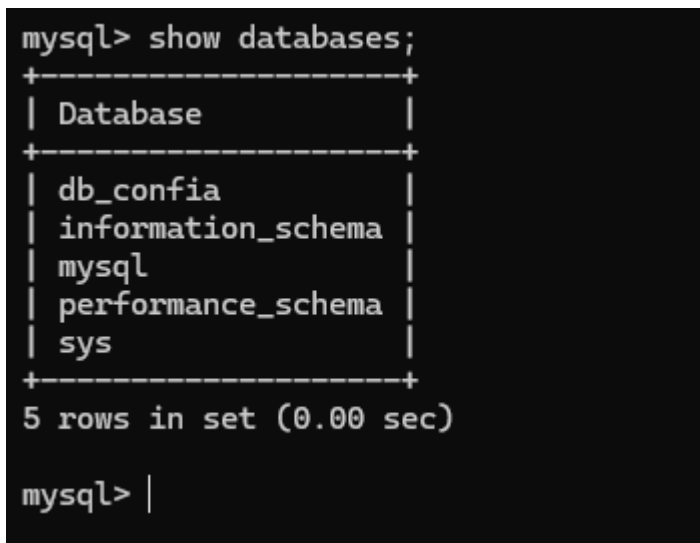
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> |
```

Enlistamos las bases de datos

`show databases;`



```
mysql> show databases;
+-----+
| Database |
+-----+
| db_confia |
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.00 sec)

mysql> |
```

Creamos la base de datos con nombre 7cm1

```
mysql> create database if not exists 7cm1;  
Query OK, 1 row affected (0.03 sec)
```

```
mysql> show databases;  
+-----+  
| Database |  
+-----+  
| 7cm1     |  
| db_confia |  
| information_schema |  
| mysql    |  
| performance_schema |  
| sys      |  
+-----+  
6 rows in set (0.00 sec)  
  
mysql> |
```

Utilizamos la base 7cm1

```
mysql> use 7cm1  
Database changed  
mysql> |
```

Procedemos a la creación de la tabla

```
CREATE TABLE PeriodoEscolar (  
    idPeriodoEscolar INT AUTO_INCREMENT PRIMARY KEY,  
    nombre VARCHAR(50) NOT NULL,  
    fechaInicio DATE,  
    fechaTermino DATE,  
    status BOOLEAN  
);
```

```
mysql> show tables
-> ;
+-----+
| Tables_in_7cm1 |
+-----+
| periodoescolar |
+-----+
1 row in set (0.00 sec)

mysql> describe PeriodoEscolar;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| idPeriodoEscolar | int           | NO   | PRI | NULL    | auto_increment |
| nombre          | varchar(50)   | NO   |     | NULL    |                 |
| fechaInicio     | date          | YES  |     | NULL    |                 |
| fechaTermino    | date          | YES  |     | NULL    |                 |
| status          | tinyint(1)    | YES  |     | NULL    |                 |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)

mysql> |
```

Crear procedimientos almacenados:

- **Insertar un Periodo Escolar:**

DELIMITER //

CREATE PROCEDURE insertar\_periodo(

IN p\_nombre VARCHAR(50),

IN p\_fechaInicio DATE,

IN p\_fechaTermino DATE,

IN p\_status BOOLEAN

)

BEGIN

INSERT INTO PeriodoEscolar (nombre, fechaInicio, fechaTermino, status)

VALUES (p\_nombre, p\_fechaInicio, p\_fechaTermino, p\_status);

END //

DELIMITER ;

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE insertar_periodo(
    -> IN p_nombre VARCHAR(50),
    -> IN p_fechaInicio DATE,
    -> IN p_fechaTermino DATE,
    -> IN p_status BOOLEAN
    -> )
    -> BEGIN
    -> INSERT INTO PeriodoEscolar (nombre, fechaInicio, fechaTermino, status)
    -> VALUES (p_nombre, p_fechaInicio, p_fechaTermino, p_status);
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER ;
mysql> |
```

- Consultar todos los Periodos Escolares:

DELIMITER //

CREATE PROCEDURE consultar\_periodos()

BEGIN

SELECT \* FROM PeriodoEscolar;

END //

DELIMITER ;

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE consultar_periodos()
    -> BEGIN
    ->     SELECT * FROM PeriodoEscolar;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> |
```

- Eliminar un Periodo Escolar:

DELIMITER //

CREATE PROCEDURE eliminar\_periodo(

IN p\_idPeriodoEscolar INT

)

BEGIN

UPDATE PeriodoEscolar SET status = FALSE WHERE idPeriodoEscolar =  
p\_idPeriodoEscolar;

END //

DELIMITER ;

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE eliminar_periodo(
  -> IN p_idPeriodoEscolar INT
  -> )
  -> BEGIN
  -> UPDATE PeriodoEscolar SET status = FALSE WHERE idPeriodoEscolar = p_idPeriodoEscolar;
  -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> |
```

- Buscar un Periodo Escolar por ID:

DELIMITER //

CREATE PROCEDURE buscar\_periodo(

IN p\_idPeriodoEscolar INT

)

BEGIN

SELECT \* FROM PeriodoEscolar WHERE idPeriodoEscolar = p\_idPeriodoEscolar;

END //

DELIMITER ;

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE buscar_periodo(
  -> IN p_idPeriodoEscolar INT
  -> )
  -> BEGIN
  -> SELECT * FROM PeriodoEscolar WHERE idPeriodoEscolar = p_idPeriodoEscolar;
  -> END //
Query OK, 0 rows affected (0.00 sec)

mysql> DELIMITER ;
mysql> |
```

- Actualizar un Periodo Escolar:

DELIMITER //

```
CREATE PROCEDURE actualizar_periodo(  
    IN p_idPeriodoEscolar INT,  
    IN p_nombre VARCHAR(50),  
    IN p_fechaInicio DATE,  
    IN p_fechaTermino DATE,  
    IN p_status BOOLEAN  
)  
BEGIN  
    UPDATE PeriodoEscolar  
    SET nombre = p_nombre,  
        fechaInicio = p_fechaInicio,  
        fechaTermino = p_fechaTermino,  
        status = p_status  
    WHERE idPeriodoEscolar = p_idPeriodoEscolar;  
END //  
DELIMITER ;
```

```
mysql> DELIMITER //
```

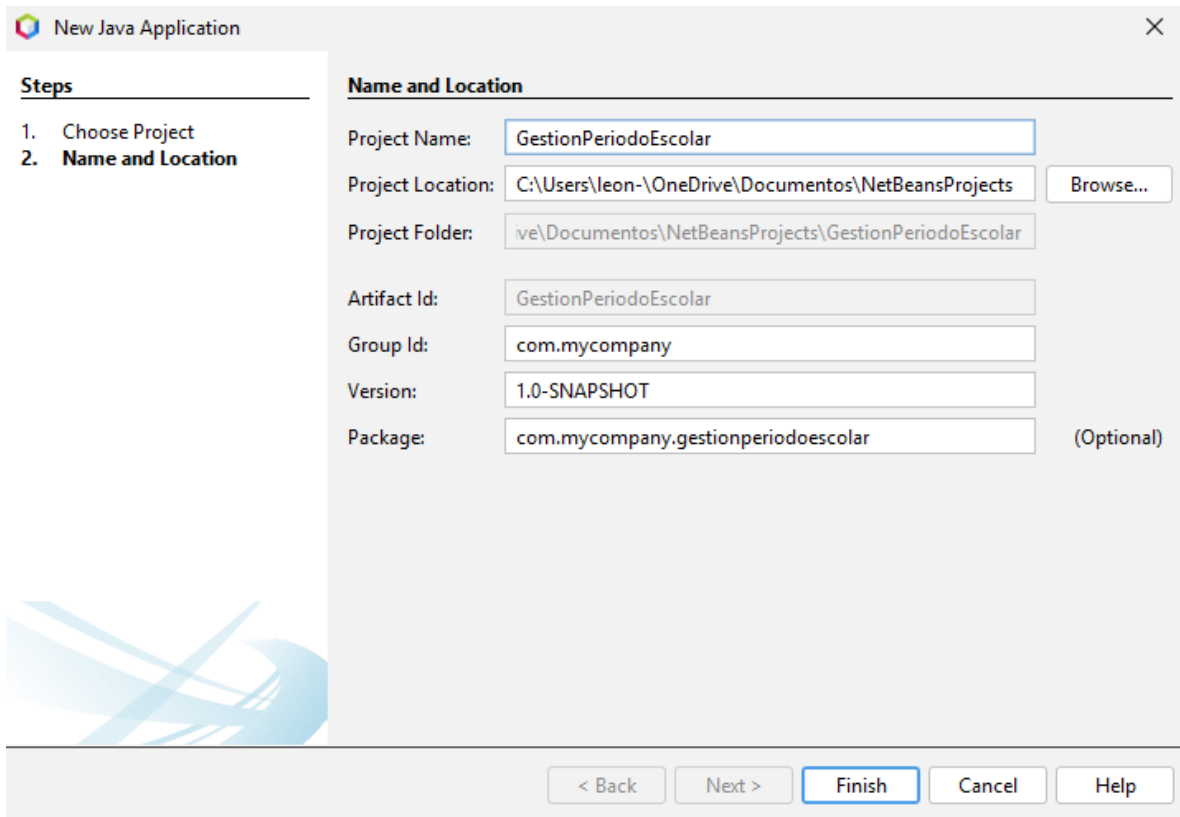
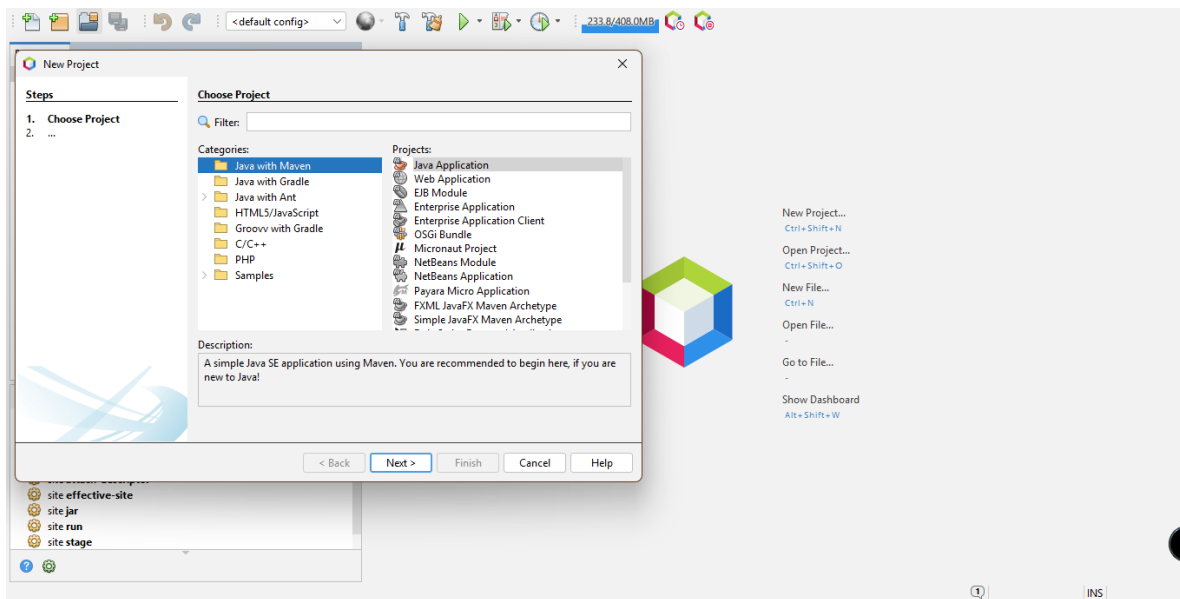
```
mysql> CREATE PROCEDURE actualizar_periodo(  
->     IN p_idPeriodoEscolar INT,  
->     IN p_nombre VARCHAR(50),  
->     IN p_fechaInicio DATE,  
->     IN p_fechaTermino DATE,  
->     IN p_status BOOLEAN  
-> )  
-> BEGIN  
->     UPDATE PeriodoEscolar  
->     SET nombre = p_nombre,  
->         fechaInicio = p_fechaInicio,  
->         fechaTermino = p_fechaTermino,  
->         status = p_status  
->     WHERE idPeriodoEscolar = p_idPeriodoEscolar;  
-> END //
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> DELIMITER ;  
mysql> |
```

# Creación del Proyecto en Java (NetBeans)

Se abre netbeans y se da crear nuevo proyecto





**Agregar las librerías necesarias:** Para que tu aplicación Java se comunice con MySQL, necesitas agregar el **JDBC driver** de MySQL (conector MySQL).

- Se descarga el conector JDBC de MySQL

Se indica en el archivo pom.xml



## Estructura del Proyecto

src

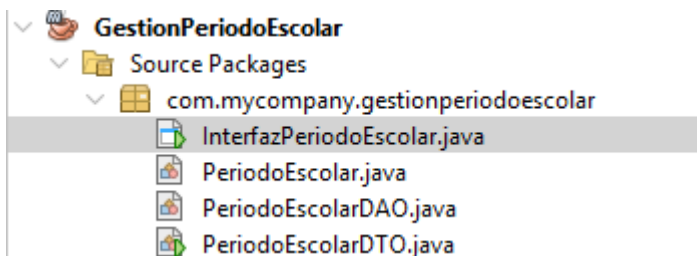
└─ com.mycompany.gestionperiodoescolar

    ├─ PeriodoEscolar.java

    ├─ PeriodoEscolarDAO.java

    ├─ PeriodoEscolarDTO.java

└─ InterfazPeriodoEscolar.java



## DESCRIPCIÓN DE LAS CLASES IMPLEMENTADAS

### PeriodoEscolar.java

Entidad que representa el modelo de Periodo Escolar con atributos:

- idPeriodoEscolar
- nombre
- fechaInicio
- fechaTermino
- status

Representa la estructura lógica de la entidad PeriodoEscolar, la cual es una abstracción directa de la tabla PeriodoEscolar en la base de datos MySQL.

Se encarga de almacenar y manipular la información referente a un periodo escolar dentro del sistema, actuando como una unidad de datos que viaja entre las capas del proyecto (DAO, Interfaz gráfica, etc.).

### Atributos (Variables de Instancia)

private int idPeriodoEscolar;

private String nombre;

private Date fechaInicio;

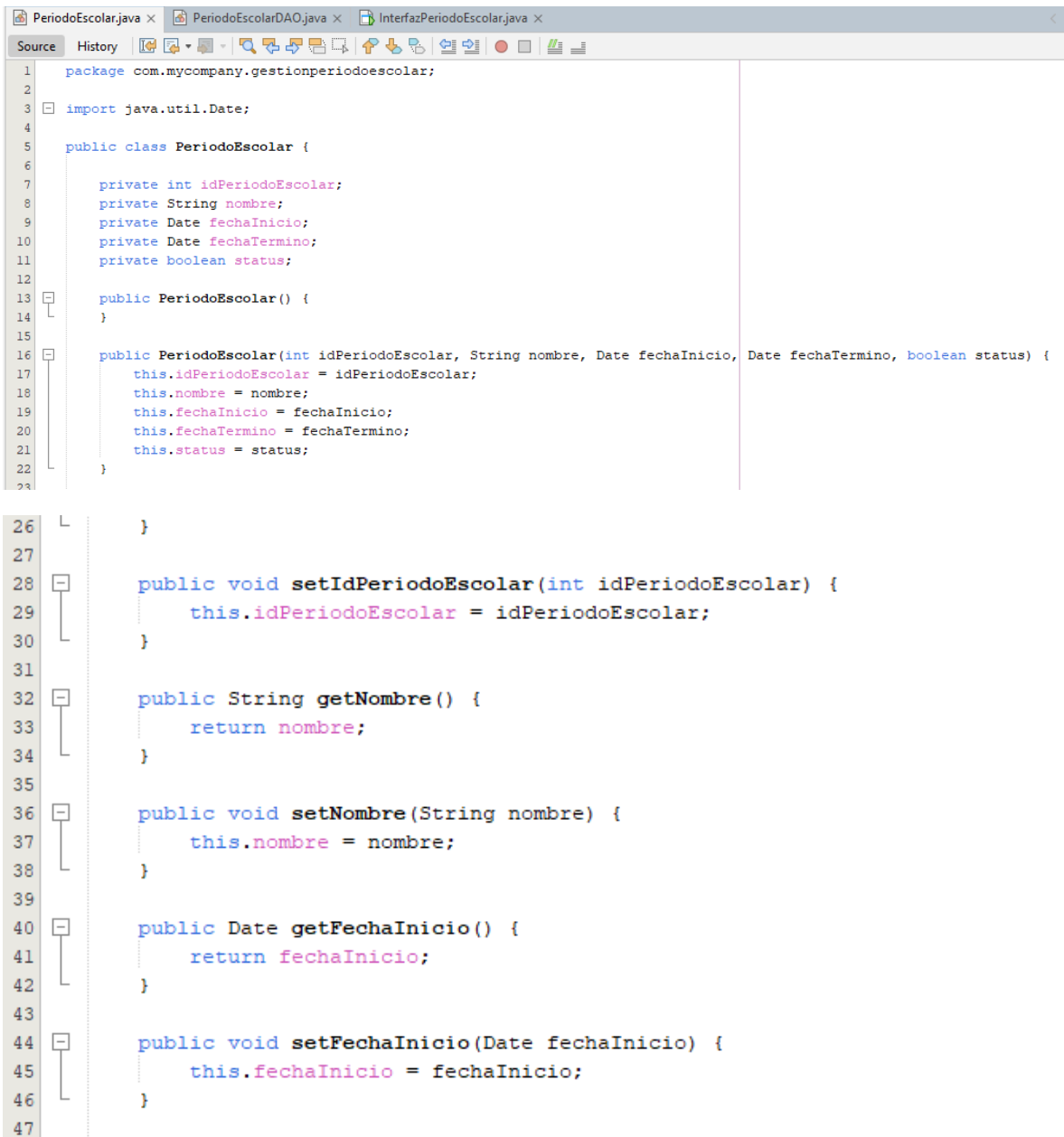
private Date fechaTermino;

private boolean status;

- idPeriodoEscolar: Identificador único del periodo escolar. Es la **clave primaria** en la tabla de la base de datos.
- nombre: Nombre del periodo escolar (ejemplo: "Periodo Agosto-Diciembre 2025").
- fechaInicio: Fecha de inicio del periodo escolar.
- fechaTermino: Fecha de término del periodo escolar.
- status: Booleano que indica si el periodo está activo (true) o inactivo (false).

La clase PeriodoEscolar:

- Sirve como **estructura de datos** para enviar y recibir información sobre periodos escolares en el sistema.
- Es utilizada por el PeriodoEscolarDAO para **intercambiar datos** entre la base de datos y el programa Java.
- En la interfaz gráfica, representa los objetos que se muestran y manipulan en la tabla JTable.



```

1  package com.myccompany.gestionperiodoescolar;
2
3  import java.util.Date;
4
5  public class PeriodoEscolar {
6
7      private int idPeriodoEscolar;
8      private String nombre;
9      private Date fechaInicio;
10     private Date fechaTermino;
11     private boolean status;
12
13     public PeriodoEscolar() {
14     }
15
16     public PeriodoEscolar(int idPeriodoEscolar, String nombre, Date fechaInicio, Date fechaTermino, boolean status) {
17         this.idPeriodoEscolar = idPeriodoEscolar;
18         this.nombre = nombre;
19         this.fechaInicio = fechaInicio;
20         this.fechaTermino = fechaTermino;
21         this.status = status;
22     }
23
24
25
26     }
27
28     public void setIdPeriodoEscolar(int idPeriodoEscolar) {
29         this.idPeriodoEscolar = idPeriodoEscolar;
30     }
31
32     public String getNombre() {
33         return nombre;
34     }
35
36     public void setNombre(String nombre) {
37         this.nombre = nombre;
38     }
39
40     public Date getFechaInicio() {
41         return fechaInicio;
42     }
43
44     public void setFechaInicio(Date fechaInicio) {
45         this.fechaInicio = fechaInicio;
46     }
47

```

PeriodoEscolar.java × PeriodoEscolarDAO.java × InterfazPeriodoEscolar.java ×

Source History

48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
66  
67  
68  
69  
70  
71  
72  
73  
74

```
public Date getFechaTermino() {  
    return fechaTermino;  
}  
  
public void setFechaTermino(Date fechaTermino) {  
    this.fechaTermino = fechaTermino;  
}  
  
public boolean isStatus() {  
    return status;  
}  
  
public void setStatus(boolean status) {  
    this.status = status;  
}  
  
@Override  
public String toString() {  
    return "PeriodoEscolar{" +  
        "idPeriodoEscolar=" + idPeriodoEscolar +  
        ", nombre='" + nombre + '\'' +  
        ", fechaInicio=" + fechaInicio +  
        ", fechaTermino=" + fechaTermino +  
        ", status=" + status +  
        '}';  
}  
}
```

## PeriodoEscolarDAO.java

La clase PeriodoEscolarDAO es el **Data Access Object (DAO)** responsable de realizar todas las **operaciones CRUD** (Crear, Leer, Actualizar y Eliminar) sobre la tabla PeriodoEscolar en la base de datos **MySQL**.

Se apoya en procedimientos almacenados para realizar las transacciones y mantiene la lógica de acceso a datos **separada** de la lógica de presentación (la interfaz gráfica) y del modelo (PeriodoEscolar).

### Principales Características

- Usa el **patrón DAO** para facilitar el acceso y manipulación de datos.
- Invoca **procedimientos almacenados** definidos en la base de datos.
- Administra la conexión mediante el método PeriodoEscolarDTO.getConexion() (que sirve como un **gestor de conexión**).
- Maneja los resultados y excepciones de forma controlada.

**Función:** Proporciona el acceso a datos para realizar operaciones CRUD sobre la tabla PeriodoEscolar en MySQL. Esta clase implementa el patrón DAO para separar el acceso a datos del resto de la aplicación. Utiliza procedimientos almacenados para garantizar la integridad de los datos y facilitar la administración de la lógica de negocio en el lado del servidor.

### Métodos implementados:

- create(PeriodoEscolar periodo): Inserta un nuevo periodo escolar.
- select(): Consulta la lista de periodos activos.
- selectById(int id): Consulta un periodo por su ID.
- update(PeriodoEscolar periodo): Actualiza un periodo existente.
- delete(int id): Desactiva un periodo escolar, realizando una eliminación lógica.

### Relación con la interfaz:

Esta clase es llamada por InterfazPeriodoEscolar cuando el usuario ejecuta alguna acción desde la GUI.

### Gestión de la conexión:

Utiliza PeriodoEscolarDTO.getConexion() para obtener el objeto Connection.

// Nombres de los procedimientos almacenados

```
private static final String SQL_INSERT = "call insertar_periodo(?, ?, ?, ?)";  
private static final String SQL_SELECT = "call consultar_periodos()";  
private static final String SQL_DELETE = "call eliminar_periodo(?)";  
private static final String SQL_SELECT_BY_ID = "call buscar_periodo(?)";  
private static final String SQL_UPDATE = "call actualizar_periodo(?, ?, ?, ?, ?)";
```

Estas constantes definen **las instrucciones SQL** que se ejecutarán a través de PreparedStatement o CallableStatement en los métodos de la clase PeriodoEscolarDAO. Cada constante representa una **llamada a un procedimiento almacenado** en MySQL que realiza una operación específica sobre la tabla PeriodoEscolar.

```
PeriodoEscolar.java x PeriodoEscolarDAO.java x InterfazPeriodoEscolar.java x PeriodoEscolarDTO.java x
Source History
1 package com.mycompany.gestionperiodoescolar;
2
3 import java.sql.Connection;
4 import java.sql.Date;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import java.sql.SQLException;
8 import java.util.ArrayList;
9 import java.util.List;
10
11 /**
12  * Esta clase se encarga de realizar operaciones CRUD sobre la tabla PeriodoEscolar.
13  * Utiliza los procedimientos almacenados en MySQL.
14  */
15 public class PeriodoEscolarDAO {
16
17     // Nombres de los procedimientos almacenados
18     private static final String SQL_INSERT = "call insertar_periodo(?, ?, ?, ?)";
19     private static final String SQL_SELECT = "call consultar_periodos()";
20     private static final String SQL_DELETE = "call eliminar_periodo(?)";
21     private static final String SQL_SELECT_BY_ID = "call buscar_periodo(?)";
22     private static final String SQL_UPDATE = "call actualizar_periodo(?, ?, ?, ?, ?)";
23
24     /**
25      * Inserta un nuevo Periodo Escolar.
26      */
```

```
27
28     public void create(PeriodoEscolar periodo) {
29         try (Connection conn = PeriodoEscolarDTO.getConexion();
30              PreparedStatement stmt = conn.prepareStatement(SQL_INSERT)) {
31
32             stmt.setString(1, periodo.getNombre());
33             stmt.setDate(2, new Date(periodo.getFechaInicio().getTime())); // java.util.Date -> java.sql.Date
34             stmt.setDate(3, new Date(periodo.getFechaTermino().getTime()));
35             stmt.setBoolean(4, periodo.isStatus());
36
37             int rowsInserted = stmt.executeUpdate();
38             if (rowsInserted > 0) {
39                 System.out.println("Periodo Escolar insertado exitosamente.");
40             }
41         } catch (SQLException e) {
42             System.out.println("Error al insertar el periodo escolar.");
43             e.printStackTrace();
44         }
45     }
46
47     /**
48      * Consulta todos los Periodos Escolares.
49      */
50     public List<PeriodoEscolar> select() {
51         List<PeriodoEscolar> listaPeriodos = new ArrayList<>();
```

```

50 List<PeriodoEscolar> listaPeriodos = new ArrayList<>();
51
52 try (Connection conn = PeriodoEscolarDTO.getConexion();
53      PreparedStatement stmt = conn.prepareStatement(SQL_SELECT);
54      ResultSet rs = stmt.executeQuery()) {
55
56     while (rs.next()) {
57         PeriodoEscolar periodo = new PeriodoEscolar();
58
59         periodo.setIdPeriodoEscolar(rs.getInt("idPeriodoEscolar"));
60         periodo.setNombre(rs.getString("nombre"));
61         periodo.setFechaInicio(rs.getDate("fechaInicio"));
62         periodo.setFechaTermino(rs.getDate("fechaTermino"));
63         periodo.setStatus(rs.getBoolean("status"));
64
65         listaPeriodos.add(periodo);
66     }
67
68 } catch (SQLException e) {
69     System.out.println("Error al consultar los periodos escolares.");
70     e.printStackTrace();
71 }
72
73 return listaPeriodos;
74 }

```

```

76 /**
77  * Consulta un Periodo Escolar por su ID.
78  */
79 public PeriodoEscolar selectById(int id) {
80     PeriodoEscolar periodo = null;
81
82     try (Connection conn = PeriodoEscolarDTO.getConexion();
83          PreparedStatement stmt = conn.prepareStatement(SQL_SELECT_BY_ID)) {
84
85         stmt.setInt(1, id);
86         try (ResultSet rs = stmt.executeQuery()) {
87             if (rs.next()) {
88                 periodo = new PeriodoEscolar();
89                 periodo.setIdPeriodoEscolar(rs.getInt("idPeriodoEscolar"));
90                 periodo.setNombre(rs.getString("nombre"));
91                 periodo.setFechaInicio(rs.getDate("fechaInicio"));
92                 periodo.setFechaTermino(rs.getDate("fechaTermino"));
93                 periodo.setStatus(rs.getBoolean("status"));
94             } else {
95                 System.out.println("No se encontró el Periodo Escolar con ID: " + id);
96             }
97         }
98     }
99 } catch (SQLException e) {
100     System.out.println("X Error al buscar el periodo escolar por ID.");
101     e.printStackTrace();
102 }

```



```

103
104     return periodo;
105 }
106
107 /**
108  * Actualiza los datos de un Periodo Escolar existente.
109  */
110 public void update(PeriodoEscolar periodo) {
111     try (Connection conn = PeriodoEscolarDTO.getConexion();
112         PreparedStatement stmt = conn.prepareStatement(SQL_UPDATE)) {
113
114         stmt.setInt(1, periodo.getIdPeriodoEscolar());
115         stmt.setString(2, periodo.getNombre());
116         stmt.setDate(3, new Date(periodo.getFechaInicio().getTime()));
117         stmt.setDate(4, new Date(periodo.getFechaTermino().getTime()));
118         stmt.setBoolean(5, periodo.isStatus());
119
120         int rowsUpdated = stmt.executeUpdate();
121         if (rowsUpdated > 0) {
122             System.out.println("Periodo Escolar actualizado exitosamente.");
123         } else {
124             System.out.println("No se encontró el Periodo Escolar con ID: " + periodo.getIdPeriodoEscolar());
125         }
126     } catch (SQLException e) {
127         System.out.println("X Error al actualizar el periodo escolar.");
128         e.printStackTrace();
129     }
130 }

```

```

133 /**
134  * Elimina (desactiva) un Periodo Escolar por su ID.
135  */
136 public void delete(int id) {
137     try (Connection conn = PeriodoEscolarDTO.getConexion();
138         PreparedStatement stmt = conn.prepareStatement(SQL_DELETE)) {
139
140         stmt.setInt(1, id);
141
142         int rowsDeleted = stmt.executeUpdate();
143         if (rowsDeleted > 0) {
144             System.out.println("Periodo Escolar eliminado (desactivado) exitosamente.");
145         } else {
146             System.out.println("No se encontró el Periodo Escolar con ID: " + id);
147         }
148     } catch (SQLException e) {
149         System.out.println("Error al eliminar el periodo escolar.");
150         e.printStackTrace();
151     }
152 }
153
154 }
155
156

```

## PeriodoEscolarDTO.java

**Función:** Proporciona una única fuente de acceso a la base de datos mediante una conexión centralizada. Permite a las clases DAO (como PeriodoEscolarDAO) obtener una conexión segura y consistente sin necesidad de replicar el código de conexión en cada método. Este DTO actúa como un **proveedor de conexión** (Connection Provider) para el PeriodoEscolarDAO, asegurando que todas las operaciones CRUD se realicen con la misma configuración de conexión.

### Método Principal:

- getConnection() devuelve un objeto Connection que se utiliza en las operaciones CRUD.

### Ventajas:

- Centralización de la configuración de conexión.
- Facilita el mantenimiento y la reutilización del código.
- Desacopla la lógica de acceso a la base de datos del resto de la aplicación.

### Código

```
package com.mycompany.gestionperiodoescolar;
```

```
import java.sql.Connection;
```

```
import java.sql.DriverManager;
```

```
import java.sql.SQLException;
```

```
/**
```

```
 * PeriodoEscolarDTO
```

```
 *
```

```
 * Esta clase se encarga de gestionar la conexión a la base de datos MySQL.
```

```
 * Es un equivalente a lo que en el proyecto anterior era ProgramaAcademicoDTO.
```

```
 * Aquí centralizamos la conexión para que pueda ser utilizada por el DAO.
```

```
 */
```

```

public class PeriodoEscolarDTO {

    // Atributo opcional: representa un objeto del modelo PeriodoEscolar
    PeriodoEscolar periodoEscolar;

    // Configuración de la conexión a la base de datos
    private static final String URL = "jdbc:mysql://localhost:3306/7cm1";
    private static final String USUARIO = "root";    // Usuario de tu base de datos MySQL
    private static final String PASSWORD = "root";  // Contraseña de tu base de datos
    MySQL

    /**
     * Constructor vacío.
     * Puedes usarlo si necesitas instanciar el DTO con un objeto PeriodoEscolar.
     */
    public PeriodoEscolarDTO() {
        // Constructor sin lógica específica por ahora
    }

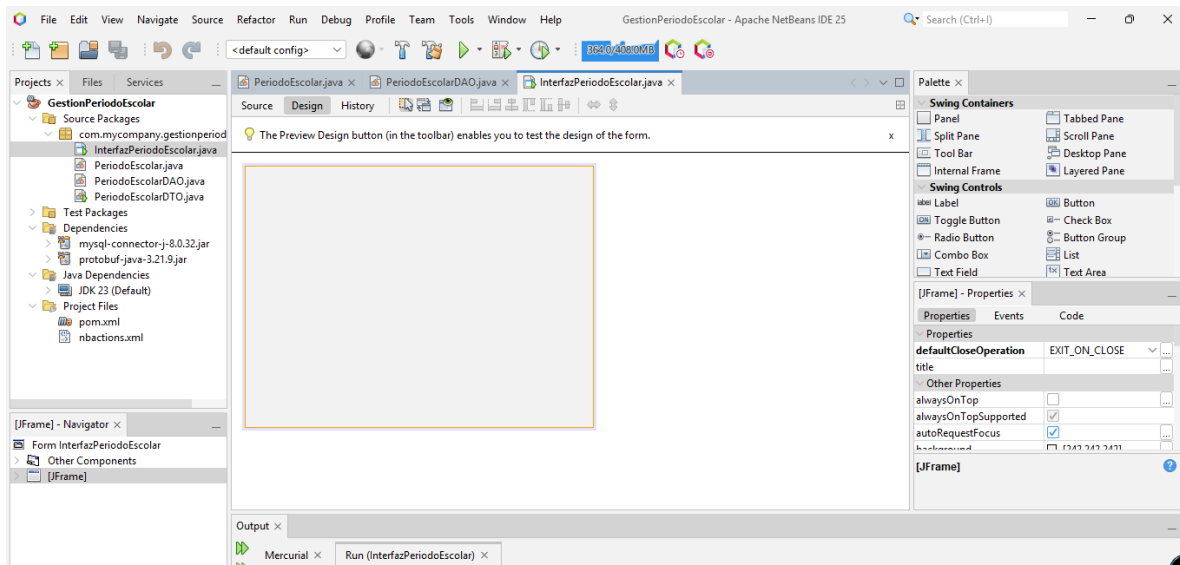
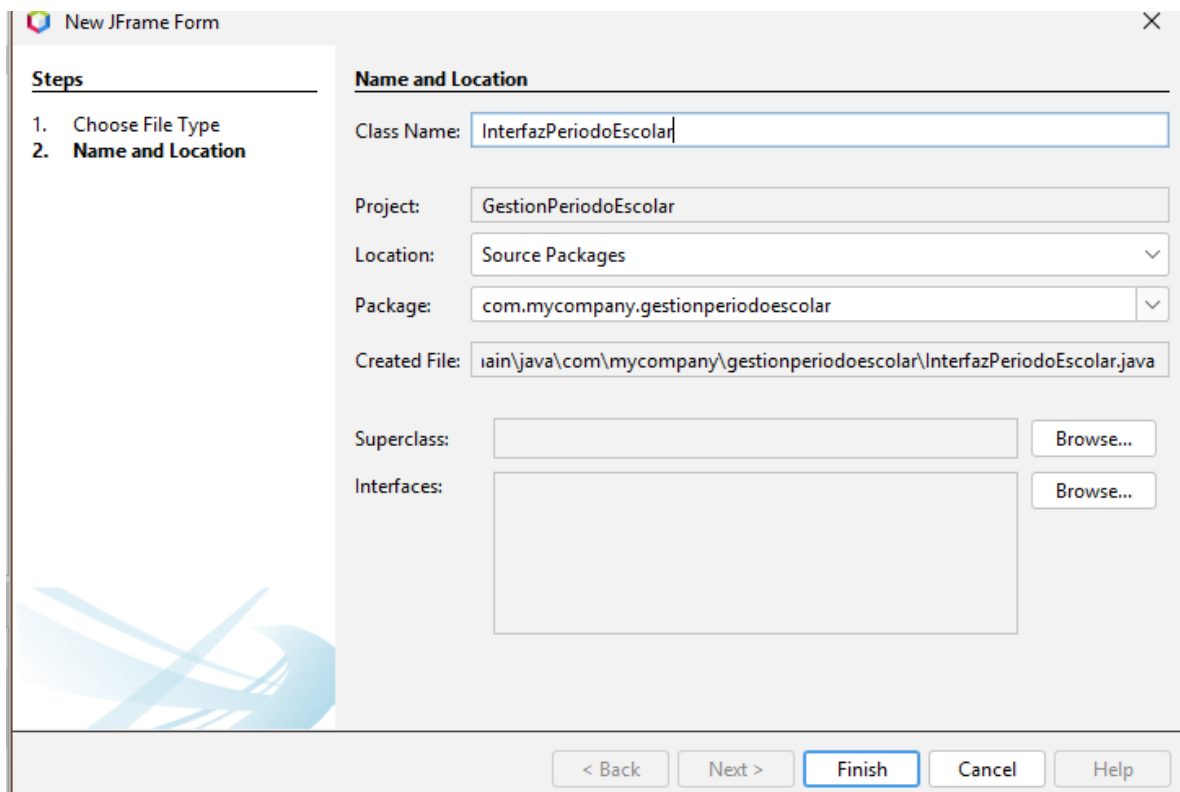
    /**
     * Método estático que devuelve la conexión a la base de datos.
     * Es llamado por el DAO para obtener un objeto Connection.
     *
     * @return Una conexión activa a la base de datos MySQL.
     * @throws SQLException Si falla la conexión.
     */
}

```

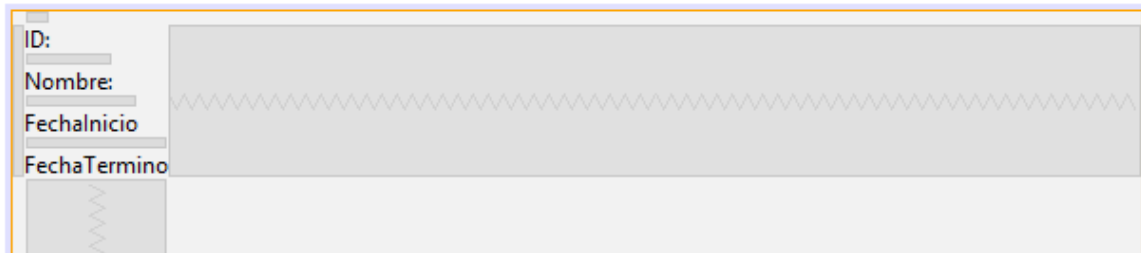
```
public static Connection getConexion() throws SQLException {  
    // Devuelve un objeto Connection usando los parámetros de conexión  
    return DriverManager.getConnection(URL, USUARIO, PASSWORD);  
}  
  
}
```

## InterfazPeriodoEscolar.java

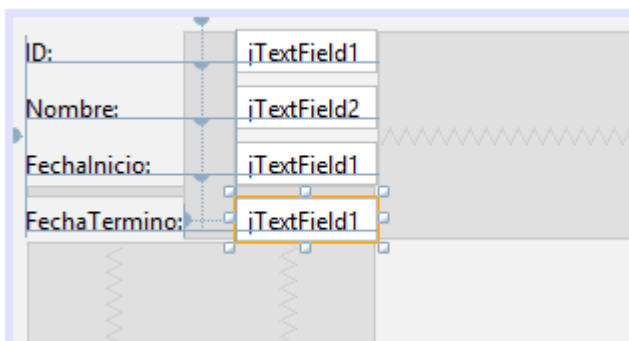
### Crear el JFrame Form en NetBeans



Etiquetas (JLabel):



Campos de texto (JTextField):



Cambia el variable name:

txtId

txtNombre

txtFechaInicio

txtFechaTermino

- ☐ txtId [JTextField]
- ☐ txtNombre [JTextField]
- ☐ txtFechaInicio [JTextField]
- ☒ txtFechaTermino [JTextField]

ID:	<input type="text"/>
Nombre:	<input type="text"/>
FechaInicio:	<input type="text"/>
FechaTermino:	<input type="text"/>
Activo:	<input type="checkbox"/>

[JFrame]

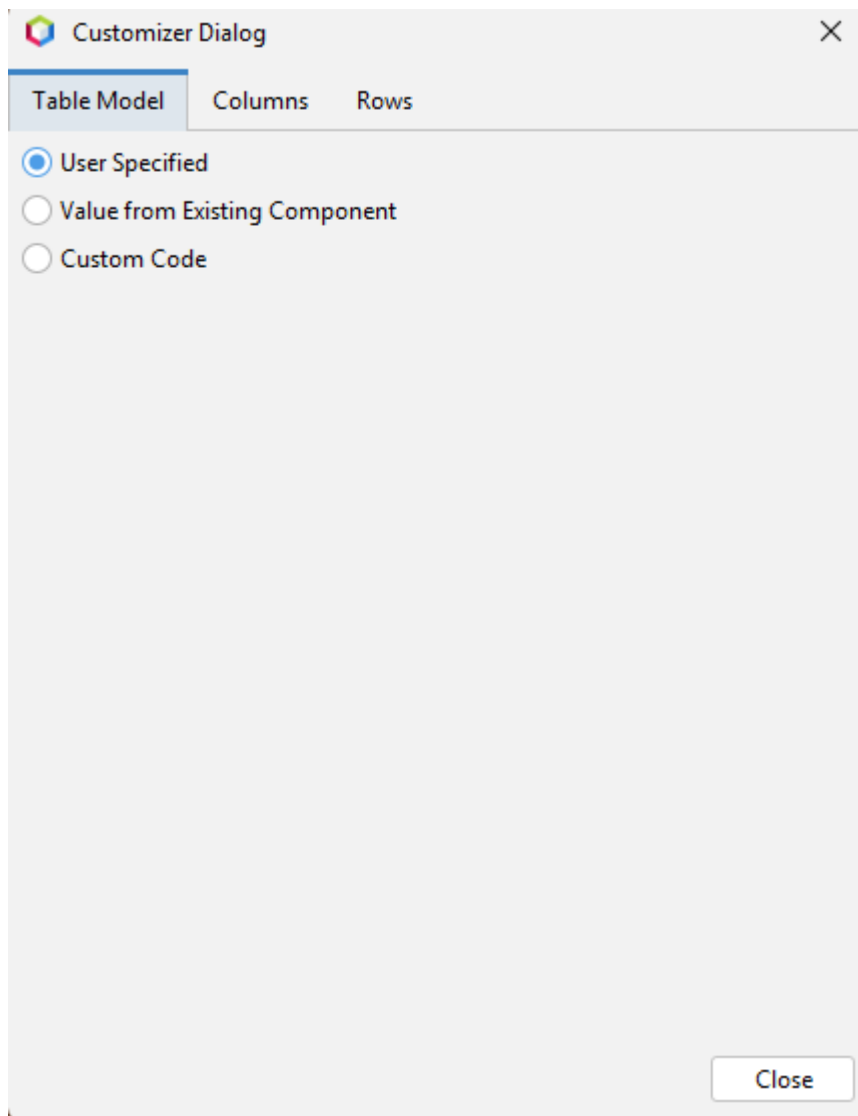
- label jLabel1 [JLabel]
- label jLabel2 [JLabel]
- label jLabel3 [JLabel]
- label jLabel4 [JLabel]
- txtId [JTextField]
- txtNombre [JTextField]
- txtFechaInicio [JTextField]
- txtFechaTermino [JTextField]
- label jLabel5 [JLabel]
- ☒ chkStatus [JCheckBox]

## Botones (JButton)

ID:	<input type="text"/>	Nuevo
Nombre:	<input type="text"/>	Guardar
FechaInicio:	<input type="text"/>	Editar
FechaTermino:	<input type="text"/>	Eliminar
Activo:	<input type="checkbox"/>	Buscar
		Listar

- ☒ btnNuevo [JButton]
- ☒ btnGuardar [JButton]
- ☒ btnEditar [JButton]
- ☒ btnEliminar [JButton]
- ☒ btnBuscar [JButton]
- ☒ btnListar [JButton]

## Tabla (JTable)





Customizer Dialog

×

Table Model

Columns

Rows

Title	Type	Resizable	Editable
ID	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Nombre	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Fecha Inicio	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Fecha Térm...	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Status	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Count: 5

◇

Insert

Delete

Move Up

Move Down

Title:

☒ Resizable ☒ Editable

Type:

Pref. Width:

Editor:

Min. Width:

Renderer:

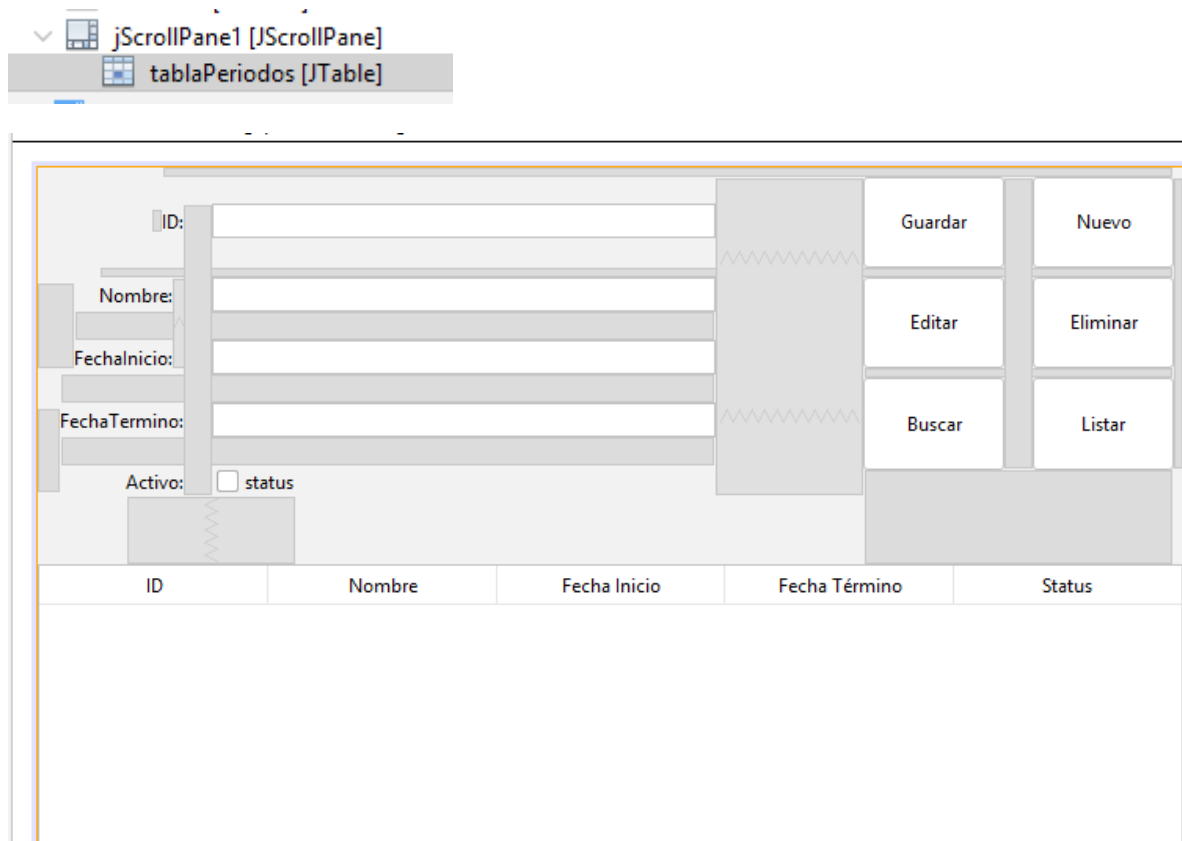
Max. Width:

Selection Model:

☒ Allow to reorder columns by drag and drop

Close

ID	Nombre	Fecha Inicio	Fecha Término	Status



La clase InterfazPeriodoEscolar es la **interfaz gráfica de usuario (GUI)** del proyecto, diseñada con **Java Swing** y desplegada en una ventana JFrame.

Esta interfaz permite realizar las operaciones CRUD (Crear, Leer, Actualizar y Eliminar) sobre los registros de la tabla PeriodoEscolar en la base de datos, a través de botones y componentes gráficos.

## Estructura del Código y Funcionalidad

### 1. Paquete e Imports

```
package com.mycompany.gestionperiodoescolar;
```

```
import javax.swing.*;
```

```
import javax.swing.table.DefaultTableModel;
```

```
import java.awt.event.*;
```

```
import java.util.List;
```

```
import java.text.SimpleDateFormat;
```

```
import java.sql.Date;
```

- Se importan los paquetes necesarios para Swing (JFrame, JButton, JTable, etc.) y para el manejo de fechas (SimpleDateFormat).
- Se incluyen utilidades como List para manejar las listas de datos.

## 2. Definición de la Clase

```
public class InterfazPeriodoEscolar extends JFrame
```

- Hereda de JFrame para crear una **ventana** principal.

## 3. Atributos de la Interfaz

```
// Componentes visuales
```

```
private JTextField txtId, txtNombre, txtFechaInicio, txtFechaTermino;
```

```
private JCheckBox chkStatus;
```

```
private JButton btnNuevo, btnGuardar, btnEditar, btnEliminar, btnBuscar, btnListar;
```

```
private JTable tablaPeriodos;
```

```
private DefaultTableModel modeloTabla;
```

### Función:

- Los JTextField almacenan los datos del formulario.
- JCheckBox permite indicar el estado del periodo.
- Los JButton son **acciones** que el usuario puede ejecutar.
- JTable despliega los datos recuperados de la base de datos.
- DefaultTableModel es el modelo de la tabla para insertar o eliminar filas dinámicamente.

## 4. Constructor y Inicialización

```
public InterfazPeriodoEscolar() {
```

```
    initComponents();
```

```
}
```

### Función:

- Inicializa la interfaz gráfica y los componentes visuales.

- Se llama a initComponents() que es el método generado automáticamente si usaste el **editor visual de NetBeans**.

## 5. Métodos Personalizados (Acciones CRUD)

### **private void guardarPeriodo()**

- Captura los datos de los campos del formulario y crea un objeto PeriodoEscolar.
- Llama al método create() del PeriodoEscolarDAO para insertar el registro.
- Limpia los campos y actualiza la tabla.

### **private void listarPeriodos()**

- Llama al método select() de PeriodoEscolarDAO.
- Llena la JTable con los datos retornados.

### **private void buscarPeriodo()**

- Recupera el ID ingresado.
- Llama a selectById() de PeriodoEscolarDAO.
- Si el periodo es encontrado, llena los campos de la interfaz con sus datos.

### **private void editarPeriodo()**

- Toma los datos actuales de los campos de texto.
- Llama a update() de PeriodoEscolarDAO para actualizar el registro correspondiente.

### **private void eliminarPeriodo()**

- Toma el ID del periodo seleccionado.
- Llama a delete() de PeriodoEscolarDAO.

## 6. Eventos de Botones

```
btnGuardar.addActionListener(e -> guardarPeriodo());
```

```
btnListar.addActionListener(e -> listarPeriodos());
```

```
btnBuscar.addActionListener(e -> buscarPeriodo());
```

```
btnEditar.addActionListener(e -> editarPeriodo());
```

```
btnEliminar.addActionListener(e -> eliminarPeriodo());
```

```
btnNuevo.addActionListener(e -> limpiarCampos());
```

**Función:**

- Cada botón tiene asignada una acción que ejecuta el método correspondiente.
- Ejemplo: Al presionar Guardar, se invoca el método guardarPeriodo().

**7. Método limpiarCampos()**

- Limpia todos los campos de texto (txtId, txtNombre, etc.).
- Resetea el estado del chkStatus y deselectiona filas en la JTable.

## Pruebas de Operaciones CRUD

Se ejecuta como main el programa interfaz

[illegible]

## Consulta de registros

Para realizar la consulta se oprime listar y se muestran los registros

The screenshot shows a web application titled "Gestion de Periodos Escolares". It features a form with the following fields: ID, Nombre, FechaInicio, FechaTermino, and an "Activo" checkbox with a "status" label. To the right of the form are five buttons: "Guardar", "Nuevo", "Editar", "Eliminar", and "Listar" (which is highlighted with a blue border). Below the form is a table with the following data:

ID	Nombre	Fecha Inicio	Fecha Término	Status
1	Periodo Marzo-Junio 20...	2025-03-19	2025-07-15	Inactivo
2	agosto	2025-03-14	2025-06-12	Activo

Below the application window, a terminal window shows a MySQL command prompt with the following output:

```
C:\WINDOWS\system32\cmd. X + v
+-----+-----+-----+-----+-----+
| idPeriodoEscolar | nombre | fechaInicio | fechaTermino | status |
+-----+-----+-----+-----+-----+
| 1 | Periodo Marzo-Junio 2025 | 2025-03-19 | 2025-07-15 | 0 |
| 2 | agosto | 2025-03-14 | 2025-06-12 | 1 |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

### Modificación de un periodo existente (Update).

Se da clic en el periodo a modificar y serán colocados en los labels

The screenshot shows the same application, but now the form is populated with data for the first row of the table. The "ID" field contains "1", "Nombre" contains "Periodo Marzo-Junio 2025", "FechaInicio" contains "2025-03-19", and "FechaTermino" contains "2025-07-15". The "Activo" checkbox is still unchecked. The table below the form remains the same as in the previous screenshot.

Aquí editamos los datos del periodo a nuestro gusto

Gestion de Periodos Escolares

ID: 1

Nombre: PPeriodo 0 a 1

FechaInicio: 2025-03-25

FechaTermino: 2025-07-31

Activo: ☒ status

Guardar Nuevo

Editar Eliminar

Buscar Listar

ID	Nombre	Fecha Inicio	Fecha Término	Status
1	Periodo Marzo-Junio 20...	2025-03-19	2025-07-15	Inactivo
2	agosto	2025-03-14	2025-06-12	Activo

Se da en editar, si damos en guardar se crearía uno nuevo



Gestion de Periodos Escolares

ID:

Nombre:

FechaInicio:

FechaTermino:

Activo: ☐ status

Guardar

Nuevo


Editar

Eliminar

Buscar

Listar

Message



✓ Periodo actualizado correctamente.

OK

ID	Nombre	Fecha Inicio	Fecha Término	Status
1	PERiodo 0 a 1	2025-03-25	2025-07-31	Activo
2	agosto	2025-03-14	2025-06-12	Activo

C:\WINDOWS\system32\cmd.

```

|          1 | Periodo Marzo-Junio 2025 | 2025-03-19 | 2025-07-15 |
|          2 | agosto                     | 2025-03-14 | 2025-06-12 |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from PeriodoEscolar;
+-----+-----+-----+-----+-----+
| idPeriodoEscolar | nombre          | fechaInicio | fechaTermino | status |
+-----+-----+-----+-----+-----+
|          1       | PERiodo 0 a 1  | 2025-03-25  | 2025-07-31  |      1 |
|          2       | agosto         | 2025-03-14  | 2025-06-12  |      1 |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> |

```

### Alta de un nuevo Periodo Escolar (Insert).

Al dar clic en nuevo se limpian nuestros labels para la inserción de los datos del nuevo periodo y se procede a agregar la información

Gestion de Períodos Escolares

ID: 3

Nombre: periodo semana santa

FechaInicio: 2025-4-15

FechaTermino: 2025-4-20

Activo: ☒ status

Guardar Nuevo

Editar Eliminar

Buscar Listar

ID	Nombre	Fecha Inicio	Fecha Término	Status
1	PERiodo 0 a 1	2025-03-25	2025-07-31	Activo
2	agosto	2025-03-14	2025-06-12	Activo

Procedemos a dar en guardar

Gestion de Periodos Escolares

ID:

Nombre:

FechaInicio:

FechaTermino:


Activo: ☐ status

Guardar Nuevo

Editar Eliminar

Buscar Listar

Message

 ✓ Periodo guardado correctamente.

OK

ID	Nombre	Fecha Inicio	Fecha Término	Status
1	PERiodo 0 a 1	2025-03-25	2025-07-31	Activo
2	agosto	2025-03-14	2025-06-12	Activo
3	periodo semana santa	2025-04-15	2025-04-20	Activo

```

C:\WINDOWS\system32\cmd. X + v
|          2 | agosto          | 2025-03-14 | 2025-06-12 | 1 |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from PeriodoEscolar;
+-----+-----+-----+-----+-----+
| idPeriodoEscolar | nombre          | fechaInicio | fechaTermino | status |
+-----+-----+-----+-----+-----+
| 1 | PERiodo 0 a 1 | 2025-03-25 | 2025-07-31 | 1 |
| 2 | agosto       | 2025-03-14 | 2025-06-12 | 1 |
| 3 | periodo semana santa | 2025-04-15 | 2025-04-20 | 1 |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> |

```

## Select

Se indica en los apartados el dato del registro a buscar y se da buscar y se anexa la info del mismo

Gestion de Periodos Escolares

ID: 2

Nombre: agosto

FechaInicio: 2025-03-14

FechaTermino: 2025-06-12

Activo: ☒ status

Guardar

Nuevo

Editar

Eliminar

Buscar

Listar

ID	Nombre	Fecha Inicio	Fecha Término	Status
1	PERiodo 0 a 1	2025-03-25	2025-07-31	Activo
2	agosto	2025-03-14	2025-06-12	Activo
3	periodo semana santa	2025-04-15	2025-04-20	Activo

### Eliminación lógica (Delete).

Para la eliminación, por motivos de buenas practicas no se hace una eliminación al 100%, solamente se desactiva el registro que deseamos eliminar por si en algún momento lo deseamos recuperar

Gestion de Periodos Escolares

ID:

Nombre:

FechaInicio:

FechaTermino:

Activo: ☐ status

Guardar

Nuevo

Editar

Eliminar

Buscar

Listar

Message

✓ Periodo eliminado correctamente.

OK

ID	Nombre	Fecha Inicio	Fecha Término	Status
1	PERiodo 0 a 1	2025-03-25	2025-07-31	Activo
2	agosto	2025-03-14	2025-06-12	Inactivo
3	periodo semana santa	2025-04-15	2025-04-20	Activo

C:\WINDOWS\system32\cmd.

3 | periodo semana santa | 2025-04-15 | 2025-04-20 | 1 |

3 rows in set (0.00 sec)

mysql> select \* from PeriodoEscolar;

idPeriodoEscolar	nombre	fechaInicio	fechaTermino	status
1	PERiodo 0 a 1	2025-03-25	2025-07-31	1
2	agosto	2025-03-14	2025-06-12	0
3	periodo semana santa	2025-04-15	2025-04-20	1

3 rows in set (0.00 sec)

mysql> |

## Conclusiones

El desarrollo del proyecto "**Gestión de Periodos Escolares**" permitió integrar diferentes tecnologías, tales como **Java (Swing)**, **MySQL** y el uso de **procedimientos almacenados**, aplicando principios de programación estructurada y buenas prácticas en la organización del código.

### Principales aprendizajes:

- Se logró establecer una conexión robusta y segura entre Java y MySQL utilizando el **Driver JDBC**.
- Se implementaron **procedimientos almacenados**, lo que permitió tener una lógica de negocio centralizada en el servidor de base de datos, mejorando el rendimiento y la seguridad.
- Se aplicó el **patrón DAO (Data Access Object)** para separar la lógica de acceso a datos de la lógica de presentación (Interfaz Gráfica), facilitando el mantenimiento y escalabilidad del sistema.
- Se diseñó una **interfaz gráfica amigable** mediante el uso de **Java Swing**, que permite al usuario gestionar periodos escolares de manera intuitiva.

### Tipos de errores más comunes encontrados durante el desarrollo

#### 1. Errores de Conexión a la Base de Datos

- **Descripción:** La conexión fallaba por datos incorrectos en la URL, usuario o contraseña.
- **Solución:** Revisar el método `getConnection()` de `PeriodoEscolarDTO.java` y verificar los parámetros:

```
private static final String URL = "jdbc:mysql://localhost:3306/7cm1";
```

```
private static final String USUARIO = "root";
```

```
private static final String PASSWORD = "root";
```

#### 2. Errores de Driver JDBC no encontrado

- **Descripción:** Al no agregar correctamente el `mysql-connector-j` al proyecto, la aplicación no podía establecer la conexión.
- **Solución:** Descargar el archivo `.jar` del **MySQL Connector/J** e incluirlo en el proyecto Maven o NetBeans como librería.

### 3. Errores de Sintaxis SQL en los Procedimientos Almacenados

- **Descripción:** Los procedimientos almacenados tenían errores en su creación o en los nombres de los parámetros, lo que causaba fallos al invocarlos desde Java.
- **Solución:** Revisar la sintaxis SQL, usar DELIMITER // correctamente y comprobar los nombres en las constantes del DAO:

```
private static final String SQL_INSERT = "call insertar_periodo(?, ?, ?, ?)";
```

### 4. Errores de Conversión de Fechas

- **Descripción:** Diferencias entre java.util.Date y java.sql.Date producían errores al intentar insertar o actualizar datos.
- **Solución:** Realizar la conversión correcta al asignar parámetros en los PreparedStatement:

```
stmt.setDate(2, new java.sql.Date(periodo.getFechaInicio().getTime()));
```