**Creacion de la base de datos utilizando mysql**

Accedemos a mysql en cmd utilizando

mysql -u root -p

Texto

El contenido generado por IA puede ser incorrecto.

Enlistamos las bases de datos

show databases;

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.

Creamos la base de datos con nombre 7cm1

Texto

El contenido generado por IA puede ser incorrecto.

Utilizamos la base 7cm1

Texto

El contenido generado por IA puede ser incorrecto.

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

);

Texto

El contenido generado por IA puede ser incorrecto.

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 ;

Texto

El contenido generado por IA puede ser incorrecto.

* Consultar todos los Periodos Escolares:

DELIMITER //

CREATE PROCEDURE consultar\_periodos()

BEGIN

SELECT \* FROM PeriodoEscolar;

END //

DELIMITER ;

Texto

El contenido generado por IA puede ser incorrecto.

* 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 ;

Texto

El contenido generado por IA puede ser incorrecto.

* 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 ;

Texto

El contenido generado por IA puede ser incorrecto.

* 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 ;

Texto

El contenido generado por IA puede ser incorrecto.

**Creación del Proyecto en Java (NetBeans)**

Se abre netbeans y se da crear nuevo proyecto  
Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

**Agregar las librerías necesarias:** Para que tu aplicación Java se comunique 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

Interfaz de usuario gráfica, Texto, Aplicación, Word

El contenido generado por IA puede ser incorrecto.



**Estructura del Proyecto**

src

└── com.mycompany.gestionperiodoescolar

├── PeriodoEscolar.java

├── PeriodoEscolarDAO.java

├── PeriodoEscolarDTO.java

└── InterfazPeriodoEscolar.java

Texto

El contenido generado por IA puede ser incorrecto.

**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.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

**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.

**Texto

El contenido generado por IA puede ser incorrecto.**

**Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.**

**Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.**

**Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.**

**Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.**

**Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.**

**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:**

* getConexion() 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**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación, Word

El contenido generado por IA puede ser incorrecto.

Tabla

El contenido generado por IA puede ser incorrecto.

Etiquetas (JLabel):

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Campos de texto (JTextField):

Interfaz de usuario gráfica, Aplicación, Word

El contenido generado por IA puede ser incorrecto.

Cambia el variable name:

txtId

txtNombre

txtFechaInicio

txtFechaTermino

Texto

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Botones (JButton)

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Tabla (JTable)

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Tabla

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación, Word

El contenido generado por IA puede ser incorrecto.



Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

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 deselecciona filas en la JTable.

**Pruebas de Operaciones CRUD**

Se ejecuta como main el programa interfaz

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

**Consulta de registros**

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

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

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

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

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Aquí editamos los datos del periodo a nuestro gusto

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

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

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

**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

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Procedemos a dar en guardar

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

**Select**

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

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

**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

**Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.**

**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 getConexion() 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()));