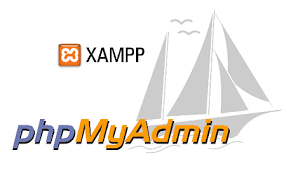


**Título: Script SQL para crear la base de datos "Escuela" y la tabla "estudiantes**



**CREATE DATABASE Escuela;**

**USE Escuela;**

**CREATE TABLE estudiantes (**

**id INT AUTO\_INCREMENT PRIMARY KEY,**

**nombre VARCHAR(50) NOT NULL,**

**apellido VARCHAR(50) NOT NULL,**

**edad INT NOT NULL,**

**dni VARCHAR(20) NOT NULL UNIQUE,**

**telefono VARCHAR(20),**

**email VARCHAR(100),**

**nivel\_estudios VARCHAR(100),**

**carrera VARCHAR(100),**

**ano INT NOT NULL,**

**nacionalidad VARCHAR(50)**

**);**

**Título: Script SQL para insertar registros en la tabla "estudiantes" de la base de datos "Escuela"**

**INSERT INTO estudiantes (nombre, apellido, edad, dni, telefono, email, nivel\_estudios, carrera, ano, nacionalidad)**

**VALUES ('Juan', 'Perez', 20, '12345678A', '555-1234', 'juan.perez@example.com', 'Licenciatura', 'Ingenieria Informatica', 1, 'Panameno');**

**INSERT INTO estudiantes (nombre, apellido, edad, dni, telefono, email, nivel\_estudios, carrera, ano, nacionalidad)**

**VALUES ('Maria', 'Gomez', 22, '87654321B', '555-5678', 'maria.gomez@example.com', 'Maestria', 'Administracion de Empresas', 2, 'Colombiana');**

**INSERT INTO estudiantes (nombre, apellido, edad, dni, telefono, email, nivel\_estudios, carrera, ano, nacionalidad)**

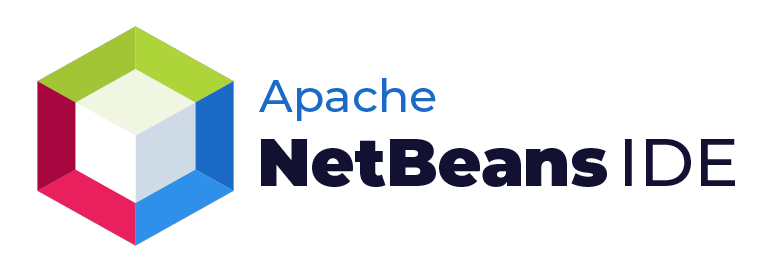
**VALUES ('Carlos', 'Lopez', 19, '45678912C', '555-9101', 'carlos.lopez@example.com', 'Bachillerato', 'Biologia', 4, 'Mexicano');**

**INSERT INTO estudiantes (nombre, apellido, edad, dni, telefono, email, nivel\_estudios, carrera, ano, nacionalidad)**

**VALUES ('Ana', 'Rodriguez', 21, '23456789D', '555-1122', 'ana.rodriguez@example.com', 'Licenciatura', 'Derecho', 5, 'Argentina');**

**INSERT INTO estudiantes (nombre, apellido, edad, dni, telefono, email, nivel\_estudios, carrera, ano, nacionalidad)**

**VALUES ('Luis', 'Fernandez', 23, '98765432E', '555-3344', 'luis.fernandez@example.com', 'Doctorado', 'Fisica', 3, 'Chileno');**



**El nombre del proyecto en NetBeans es Panschool**.

 **Haz clic en Source Package para crear un nuevo source package llamado Main.**

 **Haz clic en el source package Main para crear las clases Estudiante, EstudianteDAO y Main.**

**Título: Clase `Estudiante` para representar estudiantes con sus atributos y métodos en Java**

**Estudiante.java**

**package Main;**

**// Clase que representa un estudiante con sus atributos y métodos asociados**

**public class Estudiante {**

**// Atributos del estudiante**

**private int id; // Identificador único del estudiante**

**private String nombre; // Nombre del estudiante**

**private String apellido; // Apellido del estudiante**

**private int edad; // Edad del estudiante**

**private String dni; // Documento Nacional de Identidad del estudiante**

**private String telefono; // Teléfono de contacto del estudiante**

**private String email; // Correo electrónico del estudiante**

**private String nivelEstudios; // Nivel de estudios del estudiante**

**private String carrera; // Carrera que estudia el estudiante**

**private int ano; // Año en que se encuentra el estudiante**

**private String nacionalidad; // Nacionalidad del estudiante**

**// Constructor para inicializar un estudiante con todos sus atributos**

**public Estudiante(String nombre, String apellido, int edad, String dni, String telefono, String email, String nivelEstudios, String carrera, int ano, String nacionalidad) {**

**this.nombre = nombre;**

**this.apellido = apellido;**

**this.edad = edad;**

**this.dni = dni;**

**this.telefono = telefono;**

**this.email = email;**

**this.nivelEstudios = nivelEstudios;**

**this.carrera = carrera;**

**this.ano = ano;**

**this.nacionalidad = nacionalidad;**

**}**

**// Métodos getter y setter para acceder y modificar los atributos**

**public int getId() {**

**return id;**

**}**

**public void setId(int id) {**

**this.id = id;**

**}**

**public String getNombre() {**

**return nombre;**

**}**

**public void setNombre(String nombre) {**

**this.nombre = nombre;**

**}**

**public String getApellido() {**

**return apellido;**

**}**

**public void setApellido(String apellido) {**

**this.apellido = apellido;**

**}**

**public int getEdad() {**

**return edad;**

**}**

**public void setEdad(int edad) {**

**this.edad = edad;**

**}**

**public String getDni() {**

**return dni;**

**}**

**public void setDni(String dni) {**

**this.dni = dni;**

**}**

**public String getTelefono() {**

**return telefono;**

**}**

**public void setTelefono(String telefono) {**

**this.telefono = telefono;**

**}**

**public String getEmail() {**

**return email;**

**}**

**public void setEmail(String email) {**

**this.email = email;**

**}**

**public String getNivelEstudios() {**

**return nivelEstudios;**

**}**

**public void setNivelEstudios(String nivelEstudios) {**

**this.nivelEstudios = nivelEstudios;**

**}**

**public String getCarrera() {**

**return carrera;**

**}**

**public void setCarrera(String carrera) {**

**this.carrera = carrera;**

**}**

**public int getAno() {**

**return ano;**

**}**

**public void setAno(int ano) {**

**this.ano = ano;**

**}**

**public String getNacionalidad() {**

**return nacionalidad;**

**}**

**public void setNacionalidad(String nacionalidad) {**

**this.nacionalidad = nacionalidad;**

**}**

**// Método para representar al estudiante como una cadena**

**@Override**

**public String toString() {**

**return "Estudiante{" +**

**"id=" + id +**

**", nombre='" + nombre + '\'' +**

**", apellido='" + apellido + '\'' +**

**", edad=" + edad +**

**", dni='" + dni + '\'' +**

**", telefono='" + telefono + '\'' +**

**", email='" + email + '\'' +**

**", nivelEstudios='" + nivelEstudios + '\'' +**

**", carrera='" + carrera + '\'' +**

**", ano=" + ano +**

**", nacionalidad='" + nacionalidad + '\'' +**

**'}';**

**}**

**}**

**Título: Implementación de la clase `EstudianteDAO` para la gestión de estudiantes en una base de datos MySQL en Java.**

**EstudianteDAO.java**

**package Main;**

**import java.sql.\*;**

**import java.util.ArrayList;**

**import java.util.List;**

**public class EstudianteDAO {**

**// URL de conexión a la base de datos**

**private static final String URL = "jdbc:mysql://localhost:3306/Escuela";**

**private static final String USER = "root"; // Cambia si es necesario**

**private static final String PASSWORD = ""; // Cambia si es necesario**

**// Método para establecer la conexión a la base de datos**

**private Connection connect() throws SQLException {**

**return DriverManager.getConnection(URL, USER, PASSWORD);**

**}**

**/\*\***

**\* Agrega un nuevo estudiante a la base de datos.**

**\* @param estudiante El objeto Estudiante a agregar.**

**\*/**

**public void agregarEstudiante(Estudiante estudiante) {**

**String sql = "INSERT INTO estudiantes (nombre, apellido, edad, dni, telefono, email, nivel\_estudios, carrera, ano, nacionalidad) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";**

**try (Connection conn = connect(); PreparedStatement stmt = conn.prepareStatement(sql)) {**

**stmt.setString(1, estudiante.getNombre());**

**stmt.setString(2, estudiante.getApellido());**

**stmt.setInt(3, estudiante.getEdad());**

**stmt.setString(4, estudiante.getDni());**

**stmt.setString(5, estudiante.getTelefono());**

**stmt.setString(6, estudiante.getEmail());**

**stmt.setString(7, estudiante.getNivelEstudios());**

**stmt.setString(8, estudiante.getCarrera());**

**stmt.setInt(9, estudiante.getAno());**

**stmt.setString(10, estudiante.getNacionalidad());**

**stmt.executeUpdate(); // Ejecuta la actualización de la base de datos**

**} catch (SQLException e) {**

**e.printStackTrace(); // Manejo de excepciones**

**}**

**}**

**/\*\***

**\* Actualiza la información de un estudiante en la base de datos.**

**\* @param estudiante El objeto Estudiante con la información actualizada.**

**\*/**

**public void actualizarEstudiante(Estudiante estudiante) {**

**String sql = "UPDATE estudiantes SET nombre = ?, apellido = ?, edad = ?, telefono = ?, email = ?, nivel\_estudios = ?, carrera = ?, ano = ?, nacionalidad = ? WHERE dni = ?";**

**try (Connection conn = connect(); PreparedStatement stmt = conn.prepareStatement(sql)) {**

**stmt.setString(1, estudiante.getNombre());**

**stmt.setString(2, estudiante.getApellido());**

**stmt.setInt(3, estudiante.getEdad());**

**stmt.setString(4, estudiante.getTelefono());**

**stmt.setString(5, estudiante.getEmail());**

**stmt.setString(6, estudiante.getNivelEstudios());**

**stmt.setString(7, estudiante.getCarrera());**

**stmt.setInt(8, estudiante.getAno());**

**stmt.setString(9, estudiante.getNacionalidad());**

**stmt.setString(10, estudiante.getDni());**

**stmt.executeUpdate(); // Ejecuta la actualización de la base de datos**

**} catch (SQLException e) {**

**e.printStackTrace(); // Manejo de excepciones**

**}**

**}**

**/\*\***

**\* Elimina un estudiante de la base de datos basado en su DNI.**

**\* @param dni El DNI del estudiante a eliminar.**

**\* @return true si se eliminó exitosamente, false en caso contrario.**

**\*/**

**public boolean eliminarEstudiante(String dni) {**

**String sql = "DELETE FROM estudiantes WHERE dni = ?";**

**boolean isDeleted = false; // Variable para almacenar el resultado de la eliminación**

**try (Connection conn = connect(); PreparedStatement stmt = conn.prepareStatement(sql)) {**

**stmt.setString(1, dni);**

**int rowsAffected = stmt.executeUpdate(); // Número de filas afectadas por la eliminación**

**isDeleted = (rowsAffected > 0); // Si se afectó al menos una fila, la eliminación fue exitosa**

**} catch (SQLException e) {**

**e.printStackTrace(); // Manejo de excepciones**

**}**

**return isDeleted; // Retorna true si se eliminó, false si no**

**}**

**/\*\***

**\* Busca un estudiante en la base de datos por su DNI.**

**\* @param dni El DNI del estudiante a buscar.**

**\* @return El objeto Estudiante encontrado, o null si no se encuentra.**

**\*/**

**public Estudiante buscarEstudiantePorDni(String dni) {**

**String sql = "SELECT \* FROM estudiantes WHERE dni = ?";**

**Estudiante estudiante = null;**

**try (Connection conn = connect(); PreparedStatement stmt = conn.prepareStatement(sql)) {**

**stmt.setString(1, dni);**

**ResultSet rs = stmt.executeQuery(); // Ejecuta la consulta**

**if (rs.next()) {**

**// Crea un nuevo objeto Estudiante a partir de los datos obtenidos**

**estudiante = new Estudiante(**

**rs.getString("nombre"),**

**rs.getString("apellido"),**

**rs.getInt("edad"),**

**rs.getString("dni"),**

**rs.getString("telefono"),**

**rs.getString("email"),**

**rs.getString("nivel\_estudios"),**

**rs.getString("carrera"),**

**rs.getInt("ano"),**

**rs.getString("nacionalidad")**

**);**

**estudiante.setId(rs.getInt("id")); // Establece el ID del estudiante**

**}**

**} catch (SQLException e) {**

**e.printStackTrace(); // Manejo de excepciones**

**}**

**return estudiante; // Retorna el estudiante encontrado o null**

**}**

**/\*\***

**\* Consulta todos los estudiantes en la base de datos.**

**\* @return Una lista de objetos Estudiante.**

**\*/**

**public List<Estudiante> consultarTodos() {**

**List<Estudiante> estudiantes = new ArrayList<>();**

**String sql = "SELECT \* FROM estudiantes";**

**try (Connection conn = connect(); PreparedStatement stmt = conn.prepareStatement(sql); ResultSet rs = stmt.executeQuery()) {**

**while (rs.next()) {**

**// Crea un nuevo objeto Estudiante para cada registro en el ResultSet**

**Estudiante estudiante = new Estudiante(**

**rs.getString("nombre"),**

**rs.getString("apellido"),**

**rs.getInt("edad"),**

**rs.getString("dni"),**

**rs.getString("telefono"),**

**rs.getString("email"),**

**rs.getString("nivel\_estudios"),**

**rs.getString("carrera"),**

**rs.getInt("ano"),**

**rs.getString("nacionalidad")**

**);**

**estudiante.setId(rs.getInt("id")); // Establece el ID del estudiante**

**estudiantes.add(estudiante); // Agrega el estudiante a la lista**

**}**

**} catch (SQLException e) {**

**e.printStackTrace(); // Manejo de excepciones**

**}**

**return estudiantes; // Retorna la lista de estudiantes**

**}**

**/\*\***

**\* Imprime todos los estudiantes en la consola.**

**\*/**

**public void consultarTabla() {**

**List<Estudiante> estudiantes = consultarTodos(); // Obtiene todos los estudiantes**

**for (Estudiante estudiante : estudiantes) {**

**System.out.println(estudiante); // Imprime cada estudiante**

**}**

**}**

**}**

**Título: Aplicación Java para Gestión de Estudiantes con Funcionalidades de CRUD**

**Main.java**

package Main;

// Importar las clases necesarias

import java.util.\*;

// Clase principal que contiene la lógica del programa

public class Main {

// Inicialización del objeto Scanner para leer entradas del usuario

private static final Scanner scanner = new Scanner(System.in);

// Inicialización de la instancia de EstudianteDAO para interactuar con la base de datos de estudiantes

private static EstudianteDAO estudianteDAO = new EstudianteDAO();

// Método principal del programa, punto de entrada

public static void main(String[] args) {

int opcion; // Variable para almacenar la opción del menú seleccionada por el usuario

do {

mostrarMenu(); // Mostrar el menú de opciones

opcion = scanner.nextInt(); // Leer la opción seleccionada

scanner.nextLine(); // Consumir el salto de línea

// Estructura de control para manejar la opción seleccionada

switch (opcion) {

case 1:

agregarEstudiante(); // Llamar al método para agregar un estudiante

break;

case 2:

actualizarEstudiante(); // Llamar al método para actualizar un estudiante

break;

case 3:

buscarEstudiantePorDni(); // Llamar al método para buscar un estudiante por su DNI

break;

case 4:

consultarTodosLosEstudiantes(); // Llamar al método para consultar todos los estudiantes

break;

case 5:

eliminarEstudiante(); // Llamar al método para eliminar un estudiante

break;

case 6:

System.out.println("Saliendo del programa..."); // Mensaje de salida

break;

default:

System.out.println("Opcion no válida. Intente nuevamente."); // Mensaje de error si la opción es inválida

}

} while (opcion != 6); // Repetir hasta que el usuario elija salir

}

// Método que muestra el menú de opciones disponibles

private static void mostrarMenu() {

System.out.println("Menu de opciones:");

System.out.println("1. Agregar Estudiante");

System.out.println("2. Actualizar Estudiante");

System.out.println("3. Buscar Estudiante por DNI");

System.out.println("4. Consultar Todos los Estudiantes");

System.out.println("5. Eliminar Estudiante");

System.out.println("6. Salir");

}

// Método para agregar un nuevo estudiante

private static void agregarEstudiante() {

System.out.println("Ingrese los datos del estudiante:");

// Solicitar y almacenar los datos del estudiante

System.out.print("Nombre: ");

String nombre = scanner.nextLine();

System.out.print("Apellido: ");

String apellido = scanner.nextLine();

System.out.print("Edad: ");

int edad = scanner.nextInt();

scanner.nextLine(); // Consumir el salto de línea

System.out.print("DNI: ");

String dni = scanner.nextLine();

System.out.print("Telefono: ");

String telefono = scanner.nextLine();

System.out.print("Email: ");

String email = scanner.nextLine();

System.out.print("Nivel de Estudios: ");

String nivelEstudios = scanner.nextLine();

System.out.print("Carrera: ");

String carrera = scanner.nextLine();

System.out.print("Ano: ");

int ano = scanner.nextInt();

scanner.nextLine(); // Consumir el salto de línea

System.out.print("Nacionalidad: ");

String nacionalidad = scanner.nextLine();

// Crear un objeto Estudiante y agregarlo a la base de datos

Estudiante estudiante = new Estudiante(nombre, apellido, edad, dni, telefono, email, nivelEstudios, carrera, ano, nacionalidad);

estudianteDAO.agregarEstudiante(estudiante);

System.out.println("Estudiante agregado exitosamente.");

// Mostrar encabezado, detalles y pie de tabla para el estudiante agregado

mostrarEncabezadoTabla();

mostrarEstudiante(estudiante);

mostrarPieTabla();

}

// Método para actualizar los datos de un estudiante existente

private static void actualizarEstudiante() {

System.out.print("Ingrese el DNI del estudiante a actualizar: ");

String dni = scanner.nextLine(); // Leer el DNI del estudiante a actualizar

// Buscar el estudiante por DNI

Estudiante estudiante = estudianteDAO.buscarEstudiantePorDni(dni);

if (estudiante != null) { // Verificar si el estudiante fue encontrado

System.out.println("Ingrese los nuevos datos del estudiante (deje en blanco para conservar el actual):");

// Solicitar y permitir que el usuario actualice cada atributo

System.out.print("Nombre (" + estudiante.getNombre() + "): ");

String nombre = scanner.nextLine();

if (!nombre.isEmpty()) {

estudiante.setNombre(nombre); // Actualizar nombre si se ingresó uno nuevo

}

System.out.print("Apellido (" + estudiante.getApellido() + "): ");

String apellido = scanner.nextLine();

if (!apellido.isEmpty()) {

estudiante.setApellido(apellido); // Actualizar apellido si se ingresó uno nuevo

}

System.out.print("Edad (" + estudiante.getEdad() + "): ");

String edadInput = scanner.nextLine();

if (!edadInput.isEmpty()) {

int edad = Integer.parseInt(edadInput);

estudiante.setEdad(edad); // Actualizar edad si se ingresó uno nuevo

}

System.out.print("Telefono (" + estudiante.getTelefono() + "): ");

String telefono = scanner.nextLine();

if (!telefono.isEmpty()) {

estudiante.setTelefono(telefono); // Actualizar teléfono si se ingresó uno nuevo

}

System.out.print("Email (" + estudiante.getEmail() + "): ");

String email = scanner.nextLine();

if (!email.isEmpty()) {

estudiante.setEmail(email); // Actualizar email si se ingresó uno nuevo

}

System.out.print("Nivel de Estudios (" + estudiante.getNivelEstudios() + "): ");

String nivelEstudios = scanner.nextLine();

if (!nivelEstudios.isEmpty()) {

estudiante.setNivelEstudios(nivelEstudios); // Actualizar nivel de estudios si se ingresó uno nuevo

}

System.out.print("Carrera (" + estudiante.getCarrera() + "): ");

String carrera = scanner.nextLine();

if (!carrera.isEmpty()) {

estudiante.setCarrera(carrera); // Actualizar carrera si se ingresó uno nuevo

}

System.out.print("Ano (" + estudiante.getAno() + "): ");

String anoInput = scanner.nextLine();

if (!anoInput.isEmpty()) {

int ano = Integer.parseInt(anoInput);

estudiante.setAno(ano); // Actualizar año si se ingresó uno nuevo

}

System.out.print("Nacionalidad (" + estudiante.getNacionalidad() + "): ");

String nacionalidad = scanner.nextLine();

if (!nacionalidad.isEmpty()) {

estudiante.setNacionalidad(nacionalidad); // Actualizar nacionalidad si se ingresó uno nuevo

}

// Actualizar el estudiante en la base de datos

estudianteDAO.actualizarEstudiante(estudiante);

System.out.println("Estudiante actualizado exitosamente.");

// Mostrar encabezado, detalles y pie de tabla para el estudiante actualizado

mostrarEncabezadoTabla();

mostrarEstudiante(estudiante);

mostrarPieTabla();

} else {

System.out.println("Estudiante no encontrado."); // Mensaje si el estudiante no existe

}

}

// Método para buscar un estudiante por su DNI

private static void buscarEstudiantePorDni() {

System.out.print("Ingrese el DNI del estudiante a buscar: ");

String dni = scanner.nextLine(); // Leer el DNI

// Buscar el estudiante

Estudiante estudiante = estudianteDAO.buscarEstudiantePorDni(dni);

if (estudiante != null) { // Verificar si el estudiante fue encontrado

System.out.println("Estudiante encontrado:");

mostrarEncabezadoTabla(); // Mostrar encabezado de la tabla

mostrarEstudiante(estudiante); // Mostrar los detalles del estudiante

mostrarPieTabla(); // Mostrar pie de la tabla

}

else

{

System.out.println("Estudiante no encontrado."); // Mensaje si el estudiante no existe

}

}

// Método para consultar todos los estudiantes registrados

private static void consultarTodosLosEstudiantes() {

List<Estudiante> estudiantes = estudianteDAO.consultarTodos(); // Obtener lista de estudiantes

mostrarEncabezadoTabla(); // Mostrar encabezado de la tabla

if (estudiantes.isEmpty()) { // Verificar si no hay estudiantes

System.out.println("| No hay estudiantes registrados. |");

} else {

// Mostrar cada estudiante en la tabla

for (Estudiante estudiante : estudiantes) {

mostrarEstudiante(estudiante);

}

}

mostrarPieTabla(); // Mostrar pie de la tabla

}

// Método para eliminar un estudiante por su DNI

private static void eliminarEstudiante() {

System.out.print("Ingrese el DNI del estudiante a eliminar: ");

String dni = scanner.nextLine(); // Leer el DNI

// Intentar eliminar el estudiante

boolean eliminado = estudianteDAO.eliminarEstudiante(dni);

if (eliminado) {

System.out.println("Estudiante eliminado exitosamente."); // Mensaje de éxito

} else {

System.out.println("Estudiante no encontrado."); // Mensaje si el estudiante no existe

}

}

// Método para mostrar el encabezado de la tabla

private static void mostrarEncabezadoTabla() {

System.out.println("+-------------------+-------------------+-----+-------------------+-------------------+----------------------------------+-------------------------------+------------------------------------+--------+-------------------+");

System.out.println("| Nombre | Apellido | Edad| DNI | Telefono | Email | Nivel de Estudios | Carrera | Ano | Nacionalidad |");

System.out.println("+-------------------+-------------------+-----+-------------------+-------------------+----------------------------------+-------------------------------+------------------------------------+--------+-------------------+");

}

// Método para mostrar el pie de la tabla

private static void mostrarPieTabla() {

System.out.println("+-------------------+-------------------+-----+-------------------+-------------------+----------------------------------+-------------------------------+------------------------------------+--------+-------------------+");

}

// Método para mostrar los detalles de un estudiante en formato tabular

private static void mostrarEstudiante(Estudiante estudiante) {

System.out.printf("| %-17s | %-17s | %-3d | %-17s | %-17s | %-32s | %-29s | %-34s | %-6d | %-17s |\n",

estudiante.getNombre(), estudiante.getApellido(), estudiante.getEdad(), estudiante.getDni(),

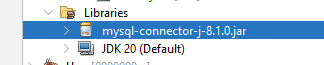
estudiante.getTelefono(), estudiante.getEmail(), estudiante.getNivelEstudios(),

estudiante.getCarrera(), estudiante.getAno(), estudiante.getNacionalidad());

}

}

**Título: Cadena de Conexión JDBC para MySQL**

****

**Finalmente, puedes ejecutarlo para ver la información y probar los mensajes de la tabla de agregar, eliminar y actualizar**