CUESTIÓN 3. Java CRUD

▪ Gestión de base de datos relacionales. Utilizando MySQL crea una base de datos en donde dispongamos de una tabla que permita almacenar lo siguientes datos de productos (idProducto, nombre, fechaEnvasado, unidades, precio, disponible). Disponible será un valor booleano.
▪ Realiza una aplicación que nos permita añadir, eliminar, actualizar y mostrar los productos.
Realizar esta tarea por consola, utiliza funciones.

RESPUESTA CUESTION 3.1.: PDF con la estructura de la BBDD, archivos correspondientes y en
Github

BDD

```
DROP DATABASE tienda;
CREATE DATABASE IF NOT EXISTS tienda;
USE tienda;

CREATE TABLE Productos (
    idProducto INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(255) NOT NULL,
    fechaEnvasado DATE NOT NULL,
    unidades INT NOT NULL,
    precio DECIMAL(10, 2) NOT NULL,
    disponible TINYINT(1) NOT NULL
);
```

Clase Tienda

```
package cuestion3;
import java.math.BigDecimal;
import java.sql.Date;
public class Tienda {
   private int idProducto;
   private String nombre;
   private Date fechaEnvasado;
   private int unidades;
   private BigDecimal precio;
   private boolean disponible;
   // Constructor
   public Tienda(int idProducto, String nombre, Date fechaEnvasado, int unidades, BigDecimal
precio, boolean disponible) {
       this.idProducto = idProducto;
       this.nombre = nombre;
       this.fechaEnvasado = fechaEnvasado;
       this.unidades = unidades;
       this.precio = precio;
       this.disponible = disponible;
   }
   // Getters y setters
   public int getIdProducto() {
       return idProducto;
   }
   public void setIdProducto(int idProducto) {
       this.idProducto = idProducto;
   }
   public String getNombre() {
```

```
        return nombre;
    }
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
    public Date getFechaEnvasado() {
        return fechaEnvasado;
    }
    public void setFechaEnvasado(Date fechaEnvasado) {
        this.fechaEnvasado = fechaEnvasado;
    }
    public int getUnidades() {
        return unidades;
    }
    public void setUnidades(int unidades) {
        this.unidades = unidades;
    }
    public BigDecimal getPrecio() {
        return precio;
    }
    public void setPrecio(BigDecimal precio) {
        this.precio = precio;
    }
    public boolean isDisponible() {
        return disponible;
    }
    public void setDisponible(boolean disponible) {
        this.disponible = disponible;
    }
}
```

Clase TiendaDAO

```
package cuestion3;
        import java.math.BigDecimal;
        import java.sql.Connection;
        import java.sql.Date;
        import java.sql.PreparedStatement;
        import java.sql.ResultSet;
        import java.sql.SQLException;
        import java.util.ArrayList;
        import java.util.List;

        public class TiendaDAO {
            private Connection conexion;
            // Constructor
            public TiendaDAO(Connection conexion) {
                this.conexion = conexion;
            }
            // Métodos CRUD
            //Insertar un producto a la BDD
            public void insertarProducto(Tienda producto) throws SQLException {
                String sql = "INSERT INTO Productos (nombre, fechaEnvasado, unidades,
precio, disponible) VALUES (?, ?, ?, ?, ?)";
                try (PreparedStatement stmt = conexion.prepareStatement(sql)) {
                    stmt.setString(1, producto.getNombre());
                    stmt.setDate(2, producto.getFechaEnvasado());
                    stmt.setInt(3, producto.getUnidades());
                    stmt.setBigDecimal(4, producto.getPrecio());
                    stmt.setBoolean(5, producto.isDisponible());
```

```
                stmt.executeUpdate();
            }
        }

        //Leer los productos de la BDD
        public List<Tienda> recuperarProductos() throws SQLException {
            List<Tienda> productos = new ArrayList<>();
            String sql = "SELECT idProducto, nombre, fechaEnvasado, unidades, precio,
disponible FROM Productos";
            try (PreparedStatement stmt = conexion.prepareStatement(sql)) {
                try (ResultSet rs = stmt.executeQuery()) {
                    while (rs.next()) {
                        int idProducto = rs.getInt("idProducto");
                        String nombre = rs.getString("nombre");
                        Date fechaEnvasado = rs.getDate("fechaEnvasado");
                        int unidades = rs.getInt("unidades");
                        BigDecimal precio = rs.getBigDecimal("precio");
                        boolean disponible = rs.getBoolean("disponible");
                        Tienda producto = new Tienda(idProducto, nombre, fechaEnvasado,
unidades, precio, disponible);
                        productos.add(producto);
                    }
                }
            }
            return productos;
        }

        //Actualizar un producto de las BDD
        public void actualizarProducto(Tienda producto) throws SQLException {
            String sql = "UPDATE Productos SET nombre = ?, fechaEnvasado = ?,
unidades = ?, precio = ?, disponible = ? WHERE idProducto = ?";
            try (PreparedStatement stmt = conexion.prepareStatement(sql)) {
                stmt.setString(1, producto.getNombre());
                stmt.setDate(2, producto.getFechaEnvasado());
                stmt.setInt(3, producto.getUnidades());
                stmt.setBigDecimal(4, producto.getPrecio());
                stmt.setBoolean(5, producto.isDisponible());
                stmt.setInt(6, producto.getIdProducto());
                stmt.executeUpdate();
            }
        }
        //Eliminar un producto de la BDD
        public void eliminarProducto(int idProducto) throws SQLException {
            String sql = "DELETE FROM Productos WHERE idProducto = ?";
            try (PreparedStatement stmt = conexion.prepareStatement(sql)) {
                stmt.setInt(1, idProducto);
                stmt.executeUpdate();
            }
        }
    }
```

Clase Main

```
package cuestion3;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Scanner;
import java.sql.Date;
import java.math.BigDecimal;
public class Main {
```

```
    public static void main(String[] args) {
        // Configuración de la conexión
        String url = "jdbc:mysql://localhost:3306/tienda";
        String usuario = "root";
        String contraseña = "curso";
        try (Connection conexion = DriverManager.getConnection(url, usuario, contraseña)) {
            TiendaDAO tiendaDAO = new TiendaDAO(conexion);
            Scanner scanner = new Scanner(System.in);
            // Menú interactivo
            boolean salir = false;
            while (!salir) {
                System.out.println("\nMenú:");
                System.out.println("1. Agregar producto");
                System.out.println("2. Mostrar productos");
                System.out.println("3. Actualizar producto");
                System.out.println("4. Eliminar producto");
                System.out.println("5. Salir");
                System.out.print("Ingrese su opción: ");
                int opcion = scanner.nextInt();
                scanner.nextLine(); // Consumir el salto de línea después de nextInt()
                switch (opcion) {
                    case 1:
                        // Agregar producto
                        System.out.print("Nombre: ");
                        String nombre = scanner.nextLine();
                        System.out.print("Fecha de envasado (AAAA-MM-DD): ");
                        String fechaEnvasadoStr = scanner.nextLine();
                        System.out.print("Unidades: ");
                        int unidades = scanner.nextInt();
                        System.out.print("Precio: ");
                        double precio = scanner.nextDouble();
                        System.out.print("Disponible (true/false): ");
                        boolean disponible = scanner.nextBoolean();
                        scanner.nextLine(); // Consumir el carácter de nueva línea
                        Tienda nuevoProducto = new Tienda(0, nombre,
Date.valueOf(fechaEnvasadoStr), unidades, BigDecimal.valueOf(precio), disponible);
                        tiendaDAO.insertarProducto(nuevoProducto);
                        System.out.println("Producto agregado correctamente.");
                        break;
                    case 2:
                        // Mostrar productos
                        System.out.println("Listado de productos:");
                        for (Tienda producto : tiendaDAO.recuperarProductos()) {
                            System.out.println(producto.getIdProducto() + ". " +
producto.getNombre() + " - " + producto.getFechaEnvasado() + " - " +
producto.getUnidades() + " - " + producto.getPrecio() + " - " + producto.isDisponible());
                        }
                        break;
                    case 3:
                        // Actualizar producto
                        System.out.print("Ingrese el ID del producto a actualizar: ");
                        int idActualizar = scanner.nextInt();
                        scanner.nextLine(); // Consumir el salto de línea después de nextInt()
                        System.out.print("Nuevo nombre: ");
                        String nuevoNombre = scanner.nextLine();
                        System.out.print("Nueva fecha de envasado (AAAA-MM-DD): ");
                        String nuevaFechaEnvasadoStr = scanner.nextLine();
                        System.out.print("Nuevas unidades: ");
                        int nuevasUnidades = scanner.nextInt();
                        System.out.print("Nuevo precio: ");
```

```
                    double nuevoPrecio = scanner.nextDouble();
                    System.out.print("Disponible (true/false): ");
                    boolean nuevaDisponible = scanner.nextBoolean();
                    scanner.nextLine(); // Consumir el carácter de nueva línea
                    Tienda productoActualizado = new Tienda(idActualizar, nuevoNombre,
Date.valueOf(nuevaFechaEnvasadoStr), nuevasUnidades,
BigDecimal.valueOf(nuevoPrecio), nuevaDisponible);
                    tiendaDAO.actualizarProducto(productoActualizado);
                    System.out.println("Producto actualizado correctamente.");
                    break;
                case 4:
                    // Eliminar producto
                    System.out.print("Ingrese el ID del producto a eliminar: ");
                    int idEliminar = scanner.nextInt();
                    tiendaDAO.eliminarProducto(idEliminar);
                    System.out.println("Producto eliminado correctamente.");
                    break;
                case 5:
                    // Salir del programa
                    salir = true;
                    break;
                default:
                    System.out.println("Opción inválida. Por favor, ingrese una opción
válida.");
                }
            }
        } catch (SQLException e) {
            System.out.println("Error de conexión: " + e.getMessage());
        }
    }
}
```