perform CRUD (Create, Read, Update, Delete) operations using JDBC with an H2 database and ArrayList for managing Product objects. This example assumes you have H2 configured and are working with a Product class, which could be something like this:

**1. Product Class**

This class will represent a product with attributes such as ID, name, and price.

public class Product {

private int id;

private String name;

private double price;

public Product(int id, String name, double price) {

this.id = id;

this.name = name;

this.price = price;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

@Override

public String toString() {

return "Product{" +

"id=" + id +

", name='" + name + '\'' +

", price=" + price +

'}';

}

}

**2. H2 Database Setup**

Ensure that you have the H2 database driver in your classpath. You can use Maven or Gradle to include the H2 dependency:

**Maven dependency:**

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<version>2.1.214</version> <!-- Use the latest version -->

</dependency>

**3. JDBC Helper Class**

This class will handle the database connection and CRUD operations.

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class ProductDao {

private static final String URL = "jdbc:h2:~/test;AUTO\_SERVER=TRUE"; // H2 database URL

private static final String USER = "sa";

private static final String PASSWORD = "";

// Establish connection

private Connection getConnection() throws SQLException {

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

}

// Create Product

public void createProduct(Product product) {

String query = "INSERT INTO products (name, price) VALUES (?, ?)";

try (Connection conn = getConnection(); PreparedStatement stmt = conn.prepareStatement(query)) {

stmt.setString(1, product.getName());

stmt.setDouble(2, product.getPrice());

stmt.executeUpdate();

System.out.println("Product created: " + product);

} catch (SQLException e) {

e.printStackTrace();

}

}

// Read all Products

public List<Product> getAllProducts() {

List<Product> products = new ArrayList<>();

String query = "SELECT \* FROM products";

try (Connection conn = getConnection(); Statement stmt = conn.createStatement()) {

ResultSet rs = stmt.executeQuery(query);

while (rs.next()) {

int id = rs.getInt("id");

String name = rs.getString("name");

double price = rs.getDouble("price");

products.add(new Product(id, name, price));

}

} catch (SQLException e) {

e.printStackTrace();

}

return products;

}

// Update Product

public void updateProduct(Product product) {

String query = "UPDATE products SET name = ?, price = ? WHERE id = ?";

try (Connection conn = getConnection(); PreparedStatement stmt = conn.prepareStatement(query)) {

stmt.setString(1, product.getName());

stmt.setDouble(2, product.getPrice());

stmt.setInt(3, product.getId());

stmt.executeUpdate();

System.out.println("Product updated: " + product);

} catch (SQLException e) {

e.printStackTrace();

}

}

// Delete Product

public void deleteProduct(int id) {

String query = "DELETE FROM products WHERE id = ?";

try (Connection conn = getConnection(); PreparedStatement stmt = conn.prepareStatement(query)) {

stmt.setInt(1, id);

stmt.executeUpdate();

System.out.println("Product with ID " + id + " deleted.");

} catch (SQLException e) {

e.printStackTrace();

}

}

// Initialize Database and Table

public void initDatabase() {

String createTableQuery = "CREATE TABLE IF NOT EXISTS products (" +

"id INT AUTO\_INCREMENT PRIMARY KEY, " +

"name VARCHAR(255), " +

"price DOUBLE)";

try (Connection conn = getConnection(); Statement stmt = conn.createStatement()) {

stmt.execute(createTableQuery);

System.out.println("Database initialized and table created.");

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**4. Main Class to Test CRUD Operations**

Here’s how you can use the ProductDao class in your main program.

import java.util.List;

public class Main {

public static void main(String[] args) {

ProductDao productDao = new ProductDao();

// Initialize database and create table

productDao.initDatabase();

// Create Products

Product product1 = new Product(0, "Laptop", 999.99);

Product product2 = new Product(0, "Smartphone", 499.99);

productDao.createProduct(product1);

productDao.createProduct(product2);

// Read all Products

System.out.println("All Products:");

List<Product> products = productDao.getAllProducts();

for (Product product : products) {

System.out.println(product);

}

// Update Product

product1.setPrice(899.99);

productDao.updateProduct(product1);

// Delete Product

productDao.deleteProduct(product2.getId());

// Final product list

System.out.println("Final Product List:");

products = productDao.getAllProducts();

for (Product product : products) {

System.out.println(product);

}

}

}

**5. Sample Output**

Here is what the output could look like when running the program:

Database initialized and table created.

Product created: Product{id=1, name='Laptop', price=999.99}

Product created: Product{id=2, name='Smartphone', price=499.99}

All Products:

Product{id=1, name='Laptop', price=999.99}

Product{id=2, name='Smartphone', price=499.99}

Product updated: Product{id=1, name='Laptop', price=899.99}

Product with ID 2 deleted.

Final Product List:

Product{id=1, name='Laptop', price=899.99}

**6. Notes:**

* **H2 Database:** This is an embedded in-memory database. When you close the application, the data is lost unless you use a file-based database.
* **JDBC Connection:** The getConnection() method establishes a connection to the H2 database.
* **CRUD Operations:** Methods in ProductDao manage CRUD operations on the products table.
* **Initialization:** The initDatabase() method ensures the table is created when the application starts.

Make sure to handle the H2 dependencies properly and adjust any configurations if you are using a different version or setup of H2.