

**CHANDIGARH**  
**UNIVERSITY**

Discover. Learn. Empower.

## **UNIVERSITY INSTITUTE OF ENGINEERING**

**Project Based Learning in Java**

**Experiment 7**

**23CSP-304**

**Submitted To:**

**Faculty Name: Er. Deep Prakash**

**Submitted By:**

**Name: Aryan Srivastav  
UID: 23BCS10135  
Section: KRG - 2B  
Semester: 5th**



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Experiment 7: JDBC-Based CRUD Operations with Transaction Handling

### Aim

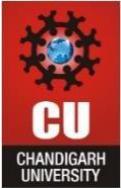
To build a Java program that performs CRUD (Create, Read, Update, Delete) operations on a MySQL database table named **Product**, using JDBC with proper transaction handling to ensure data integrity.

### Objectives

- Create a **Product** table in MySQL with columns: ProductID, ProductName, Price, and Quantity.
- Establish a **JDBC connection** between Java and MySQL.
- Implement **menu-driven CRUD operations** (Add, View, Update, Delete).
- Use **PreparedStatement** to prevent SQL injection.
- Implement **transaction handling** using commit() and rollback().
- Close all JDBC resources properly in the finally block.

### Code Implementation

```
import java.sql.*;  
import java.util.Scanner;  
  
public class ProductCRUD {  
    private static final String URL = "jdbc:mysql://localhost:3306/your_database_name";  
    private static final String USER = "root";  
    private static final String PASSWORD = "your_password";  
  
    public static void main(String[] args) {  
        Connection conn = null;  
        Scanner sc = new Scanner(System.in);  
  
        try {  
            // Step 1: Establish connection  
            conn = DriverManager.getConnection(URL, USER, PASSWORD);  
            conn.setAutoCommit(false); // Transaction Handling  
  
            while (true) {  
                System.out.println("\n--- Product Management Menu ---");  
                System.out.println("1. Add Product");  
                System.out.println("2. View All Products");  
                System.out.println("3. Update Product");  
                System.out.println("4. Delete Product");  
                System.out.println("5. Exit");  
                System.out.print("Enter your choice: ");  
                int choice = sc.nextInt();  
                if (choice == 1) {  
                    addProduct(conn);  
                } else if (choice == 2) {  
                    viewProducts(conn);  
                } else if (choice == 3) {  
                    updateProduct(conn);  
                } else if (choice == 4) {  
                    deleteProduct(conn);  
                } else if (choice == 5) {  
                    break;  
                }  
            }  
            conn.close();  
        } catch (SQLException e) {  
            e.printStackTrace();  
        }  
    }  
}  
  
// Methods for CRUD operations (omitted for brevity)
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
switch (choice) {  
    case 1:  
        addProduct(conn, sc);  
        break;  
    case 2:  
        viewProducts(conn);  
        break;  
    case 3:  
        updateProduct(conn, sc);  
        break;  
    case 4:  
        deleteProduct(conn, sc);  
        break;  
    case 5:  
        System.out.println("Exiting...");  
        conn.close();  
        System.exit(0);  
    default:  
        System.out.println("Invalid choice! Try again.");  
    }  
}  
}  
}  
} catch (Exception e) {  
    e.printStackTrace();  
} finally {  
    try {  
        if (conn != null) conn.close();  
        sc.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}  
}  
}  
  
// CREATE Operation  
private static void addProduct(Connection conn, Scanner sc) {  
    try {  
        System.out.print("Enter Product Name: ");  
        sc.nextLine(); // Consume newline  
        String name = sc.nextLine();  
        System.out.print("Enter Price: ");  
        double price = sc.nextDouble();  
        System.out.print("Enter Quantity: ");  
        int qty = sc.nextInt();  
        String sql = "INSERT INTO Product (ProductName, Price, Quantity) VALUES (?, ?, ?)";  
    }  
}
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
PreparedStatement ps = conn.prepareStatement(sql);
ps.setString(1, name);
ps.setDouble(2, price);
ps.setInt(3, qty);

ps.executeUpdate();
conn.commit();
System.out.println("Product added successfully!");

} catch (Exception e) {
try {
conn.rollback();
System.out.println("Transaction rolled back due to an error.");
} catch (SQLException ex) {
ex.printStackTrace();
}
}
}

// READ Operation
private static void viewProducts(Connection conn) {
try {
String sql = "SELECT * FROM Product";
Statement st = conn.createStatement();
ResultSet rs = st.executeQuery(sql);

System.out.println("\n--- Product List ---");
while (rs.next()) {
System.out.println(rs.getInt("ProductID") + " | " +
rs.getString("ProductName") + " | ₹" +
rs.getDouble("Price") + " | Qty: " +
rs.getInt("Quantity"));
}
} catch (SQLException e) {
e.printStackTrace();
}
}

// UPDATE Operation
private static void updateProduct(Connection conn, Scanner sc) {
try {
System.out.print("Enter Product ID to update: ");
int id = sc.nextInt();
System.out.print("Enter new Price: ");
double price = sc.nextDouble();
System.out.print("Enter new Quantity: ");
}
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
int qty = sc.nextInt();
String sql = "UPDATE Product SET Price = ?, Quantity = ? WHERE ProductID = ?";
PreparedStatement ps = conn.prepareStatement(sql);
ps.setDouble(1, price);
ps.setInt(2, qty);
ps.setInt(3, id);

int rows = ps.executeUpdate();
if (rows > 0) {
    conn.commit();
    System.out.println("Product updated successfully!");
} else {
    System.out.println("Product not found!");
}

} catch (Exception e) {
try {
    conn.rollback();
    System.out.println("Transaction rolled back due to an error.");
} catch (SQLException ex) {
    ex.printStackTrace();
}
}
}

// DELETE Operation
private static void deleteProduct(Connection conn, Scanner sc) {
try {
    System.out.print("Enter Product ID to delete: ");
    int id = sc.nextInt();

    String sql = "DELETE FROM Product WHERE ProductID = ?";
    PreparedStatement ps = conn.prepareStatement(sql);
    ps.setInt(1, id);

    int rows = ps.executeUpdate();
    if (rows > 0) {
        conn.commit();
        System.out.println("Product deleted successfully!");
    } else {
        System.out.println("Product not found!");
    }

} catch (Exception e) {
try {
    conn.rollback();
}
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
System.out.println("Transaction rolled back due to an error.");
} catch (SQLException ex) {
ex.printStackTrace();

}
}
}
}
```

## Output

```
--- Product Management Menu ---
1. Add Product
2. View All Products
3. Update Product
4. Delete Product
5. Exit

Enter your choice: 1
Enter Product Name: Laptop
Enter Price: 80000
Enter Quantity: 10
Product added successfully!
```

## Learning Outcomes

- Understood how to connect Java with MySQL using **JDBC API**.
- Learned to perform **CRUD operations** using PreparedStatement.
- Implemented **transaction management** using commit() and rollback().
- Ensured **data integrity** and prevented **SQL injection**.
- Gained experience in writing **menu-driven console applications** with JDBC.