

NAME: Khushal Mishra, **UID:** 22BCS11931, **GROUP/SECTION:** 619 - B

EXPERIMENT – 5 (Project Based Learning with Java)

CODE JAVA

```
package name;

import java.sql.*;

public class Mydb {
    private static final String URL = "jdbc:mysql://localhost:3306/exp4.db";
    private static final String USER = "root";
    private static final String PASSWORD = "Scnzcoming@1";
    private Connection connection;

    public Mydb() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            connection = DriverManager.getConnection(URL, USER, PASSWORD);
            System.out.println(" Database Connected Successfully!");
        } catch (ClassNotFoundException e) {
            System.err.println(" JDBC Driver not found! Add MySQL Connector JAR.");
        } catch (SQLException e) {
            System.err.println(" Connection failed! Check credentials. Error: " + e.getMessage());
        }
    }

    public void getAllEmployees() {
        String query = "SELECT * FROM employee";
        try (Statement stmt = connection.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {
            System.out.println("\n Employee List:");
            while (rs.next()) {
                System.out.println("ID: " + rs.getInt("id") +
                                   ", Name: " + rs.getString("name") +
                                   ", Salary: " + rs.getDouble("salary"));
            }
        } catch (SQLException e) {
            System.err.println("Error fetching employees: " + e.getMessage());
        }
    }

    public void addEmployee(int id, String name, double salary) {
        String query = "INSERT INTO employee (id, name, salary) VALUES (?, ?, ?)";
        try (PreparedStatement stmt = connection.prepareStatement(query)) {
            stmt.setInt(1, id);
            stmt.setString(2, name);
            stmt.setDouble(3, salary);
            stmt.executeUpdate();
            System.out.println("Employee added successfully: " + name);
        } catch (SQLException e) {
            System.err.println(" Error adding employee: " + e.getMessage());
        }
    }
}
```

```
public void removeDuplicateEmployees() {
    String query = "DELETE e1 FROM employee e1 " +
        "INNER JOIN employee e2 " +
        "ON e1.name = e2.name AND e1.salary = e2.salary " +
        "WHERE e1.id > e2.id";
    try (Statement stmt = connection.createStatement()) {
        int rowsDeleted = stmt.executeUpdate(query);
        System.out.println(" " + rowsDeleted + " duplicate records removed.");
    } catch (SQLException e) {
        System.err.println(" Error removing duplicates: " + e.getMessage());
    }
}

public void closeConnection() {
    if (connection != null) {
        try {
            connection.close();
            System.out.println("Database Connection Closed.");
        } catch (SQLException e) {
            System.err.println(" Error closing connection: " + e.getMessage());
        }
    }
}

public static void main(String[] args) {
    Mydb db = new Mydb();

    db.addEmployee(109,"James", 5000);
    db.addEmployee(110,"Harry Potter", 6000);
    db.addEmployee(111,"John cena", 5500);
    db.addEmployee(112,"Rahul", 8000);
    db.addEmployee(113,"Ram", 5200);
    db.addEmployee(140,"Devi", 6000);

    db.getAllEmployees();

    db.removeDuplicateEmployees();

    db.getAllEmployees();

    db.closeConnection();
}
```

OUTPUT

```

ID: 107, Name: John Doe, Salary: 5000.0
ID: 108, Name: Krishna1, Salary: 6000.0
ID: 109, Name: Harsh1, Salary: 5500.0
ID: 110, Name: Sartha1k, Salary: 8000.0
ID: 111, Name: John cena, Salary: 5500.0
ID: 112, Name: Rahul, Salary: 8000.0
ID: 113, Name: Ram, Salary: 5200.0
ID: 140, Name: Devi, Salary: 6000.0
ID: 160, Name: Aman1, Salary: 5200.0
☒ 0 duplicate records removed.

❖ Employee List:
ID: 101, Name: John Doe, Salary: 50000.0
ID: 102, Name: Krishna, Salary: 60000.0
ID: 103, Name: Harsh, Salary: 55000.0
ID: 104, Name: Sarthak, Salary: 58000.0
ID: 105, Name: Aman, Salary: 52000.0
ID: 107, Name: John1 Doe, Salary: 5000.0
ID: 108, Name: Krishna1, Salary: 6000.0
ID: 109, Name: Harsh1, Salary: 5500.0
ID: 110, Name: Sartha1k, Salary: 8000.0
ID: 111, Name: John cena, Salary: 5500.0
ID: 112, Name: Rahul, Salary: 8000.0
ID: 113, Name: Ram, Salary: 5200.0
ID: 140, Name: Devi, Salary: 6000.0
ID: 160, Name: Aman1, Salary: 5200.0

```

DATABASE

The screenshot shows the MySQL Workbench application. The Navigator pane on the left displays the database schema, including the 'exp4' schema which contains tables like 'employee' and 'employees'. The central SQL editor pane contains Java code for managing employees, including methods for adding employees, removing duplicates, and closing connections. The Output pane at the bottom shows the execution log with various SQL statements and their execution details.

```

public void removeDuplicateEmployees() {
    String query = "DELETE e1 FROM employee e1 " +
        "INNER JOIN employee e2 " +
        "ON e1.name = e2.name AND e1.salary = e2.salary " +
        "WHERE e1.id > e2.id";
    try (Statement stmt = connection.createStatement()) {
        int rowsDeleted = stmt.executeUpdate(query);
        System.out.println("☒ " + rowsDeleted + " duplicates removed");
    } catch (SQLException e) {
        System.err.println("Error removing duplicates: " + e);
    }
}

public void closeConnection() {
    if (connection != null) {
        connection.close();
    }
}

```

#	Time	Action	Message	Duration / Fetch
11	09:43:59	CREATE TABLE employee (id INT PRIMARY KEY auto_increment, name ...)	0 row(s) affected	0.047 sec
12	09:59:19	select * from employee LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
13	09:59:22	select * from employee LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
14	09:59:36	SELECT * FROM amitdb.employee LIMIT 0, 1000	0 row(s) returned	0.016 sec / 0.000 sec
15	10:03:10	SELECT * FROM amitdb.employee LIMIT 0, 1000	0 row(s) returned	0.015 sec / 0.000 sec
16	10:03:13	CREATE TABLE amitdb.employees (id INT PRIMARY KEY auto_incr...	19 rows(s) returned	0.016 sec / 0.000 sec