

## **Worksheet -3**

**Student Name: Rahul Saxena**

**UID: 24MCI10204**

**Branch: MCA (AI & ML)**

**Section/Group: 3-B**

**Semester: 2nd semester**

**Date of Performance: 20/02/2025**

**Subject: Advanced Internet Programming**

**Subject code: 24CAP-652**

### **Aim/Overview of the practical:**

To develop a Java EE application that manages employee data using ORM (Hibernate) for database operations. The application will use Java Persistence Query Language (JPQL) to interact with the database and JNDI (Java Naming and Directory Interface) for establishing Java Database Connectivity (JDBC), ensuring efficient and scalable data management.

### **The task to be done:**

- 1. Set Up the Environment:** Configure the Java EE environment with necessary dependencies for Hibernate, JPA, and JNDI.
- 2. Create the Database:** Design and set up an Employee table in the database.
- 3. Develop the Entity Class:** Create a Hibernate entity to map the Employee table using JPA annotations.
- 4. Implement CRUD Operations:** Use Java Persistence Query Language (JPQL) to perform Create, Read, Update, and Delete (CRUD) operations.
- 5. Configure JNDI:** Set up JNDI to manage JDBC resources for database connectivity.
- 6. Deploy and Test:** Deploy the application on a Java EE server, test the CRUD functionality, and verify data persistence in the database.

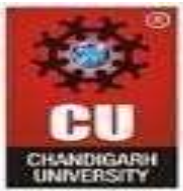
### **2. Algorithm/Flowchart:**

- 1. Start**
- 2. Set up Environment:** Configure **Hibernate**, **JPA**, and **JNDI**.
- 3. Create Database and Entity:** Design the **Employee** table and map it using **JPA annotations**.
- 4. Configure JNDI:** Set up **JNDI** for **database connectivity**.
- 5. Perform CRUD Operations:** Use **JPQL** to **Create**, **Read**, **Update**, and **Delete** employee records.
- 6. Deploy and Test:** Deploy the application and verify database operations.
- 7. End**

### 3-Code for experiment/practical and output:

#### MAIN CODE:

```
package com.mycompany.experiment_3; import
java.sql.*;
/**
 * @author Rahul
 */
public class Experiment_3 {
    public static void main(String[] args) {
        // Database connection details
        String url = "jdbc:mysql://localhost:3306/employee";
        String user = "root";
        String password = "root123";
        try {
            // Step 1: Load MySQL JDBC Driver
            Class.forName("com.mysql.cj.jdbc.Driver");
            // Step 2: Establish connection
            Connection con = DriverManager.getConnection(url, user, password);
            // Step 3: Create a Statement
            Statement stmt = con.createStatement();
            // Step 4: Execute Query
            ResultSet rs = stmt.executeQuery("SELECT * FROM emp_details");
            // Step 5: Print Employee Details
            System.out.println("Employee Details:");
            System.out.println("-----");
            while (rs.next()) {
                System.out.printf("ID: %s | Name: %s | Department: %s | Salary: ₹%.2f%n",
rs.getString("emp_id"),          rs.getString("emp_name"),
rs.getString("dept"),
                rs.getDouble("salary"));
            }
            System.out.println("-----");
            // Step 6: Close resources
            rs.close();
            stmt.close();
            con.close();        } catch
(Exception e) {
e.printStackTrace();
        }
    }
}
```



## POM.XML:

```
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchemaInstance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven4.0.0.xsd">

    <modelVersion>4.0.0</modelVersion>

    <groupId>com.mycompany</groupId>

    <artifactId>experiment_3</artifactId>

    <version>1.0-SNAPSHOT</version>

    <packaging>jar</packaging>

    <properties>

        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

        <maven.compiler.release>23</maven.compiler.release>

        <exec.mainClass>com.mycompany.experiment_3.Experiment_3</exec.mainClass>

    </properties>

    <dependencies>

        <dependency>

            <groupId>com.mysql</groupId>

            <artifactId>mysql-connector-j</artifactId>

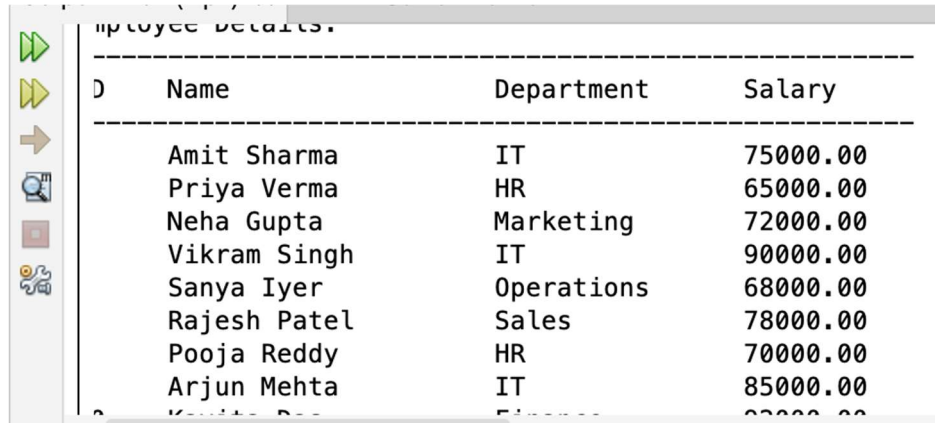
            <version>8.3.0</version>

        </dependency>

    </dependencies>

</project>
```

## Output:



The screenshot shows a Java Swing application window titled "Employee Details." with a standard Mac OS X-style title bar (green, yellow, and red buttons). The window contains a table with the following data:

| ID | Name         | Department | Salary   |
|----|--------------|------------|----------|
|    | Amit Sharma  | IT         | 75000.00 |
|    | Priya Verma  | HR         | 65000.00 |
|    | Neha Gupta   | Marketing  | 72000.00 |
|    | Vikram Singh | IT         | 90000.00 |
|    | Sanya Iyer   | Operations | 68000.00 |
|    | Rajesh Patel | Sales      | 78000.00 |
|    | Pooja Reddy  | HR         | 70000.00 |
|    | Arjun Mehta  | IT         | 85000.00 |
|    | Kavita Rao   | Finance    | 82000.00 |

## Learning outcomes:

1. Understanding Hibernate ORM and JPA for managing relational databases in Java EE applications.
2. Implementing CRUD operations using Java Persistence Query Language (JPQL).
3. Configuring and utilizing JNDI for database connectivity.
4. Developing a scalable and efficient enterprise application using Java EE standards.
5. Deploying and testing a Java EE application on an enterprise server.