```
-------------------pom.xml----------------------
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.ameya</groupId>
  <artifactId>019-jdbcproject</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <properties>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
  </properties>
  <dependencies>
    <dependency>
        <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.26</version>
    </dependency>
  </dependencies>
</project>
---------------------jdbcprops.properties-------------------
CONURL=jdbc:mysql://localhost:3306/sapientdb
DRIVERCLASSNAME=com.mysql.cj.jdbc.Driver
DBUSERNAME=root
DBPASSWORD=root
----------------------Employee.java----------------------
package com.ameya.domain;

public class Employee {

    private int empId;
    private String firstName;
    private String lastName;
    private double salary;
    private int age;
    public Employee() {
        super();
        // TODO Auto-generated constructor stub
    }

    public Employee(int empId, String firstName, String lastName, double salary, int
    age) {
        super();
        this.empId = empId;
        this.firstName = firstName;
        this.lastName = lastName;
        this.salary = salary;
        this.age = age;
    }

    public int getEmpId() {
        return empId;
    }
    public void setEmpId(int empId) {
        this.empId = empId;
    }
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    public double getSalary() {
        return salary;
    }
    public void setSalary(double salary) {
        this.salary = salary;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return "Employee [empId=" + empId + ", firstName=" + firstName + ",
        lastName=" + lastName + ", salary=" + salary
                + ", age=" + age + "]";
    }


}
------------------AppProperties.java---------------------
package com.ameya.domain;

public enum AppProperties {

    CONURL,
    DRIVERCLASSNAME,
    DBUSERNAME,
    DBPASSWORD;
}
-------------------PropertiesHelper.java---------------------
package com.ameya.helpers;

import java.io.IOException;
import java.util.Properties;

public class PropertiesHelper {

    private final Properties dbProps;
    public PropertiesHelper() {
        dbProps=new Properties();
        try {
                dbProps.load(getClass()
                        .getClassLoader()
                        .getResourceAsStream("jdbcprops.properties"));
        }catch(IOException e) {
            e.printStackTrace();
        }
    }
    public String getProperty(String key) {
        return dbProps.getProperty(key);
    }
}
--------------------TestPropertiesHelper.java---------------------
package com.ameya.test;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

import com.ameya.domain.AppProperties;
import com.ameya.helpers.PropertiesHelper;
import java.sql.DatabaseMetaData;

public class TestPropertiesHelper {

    public static void main(String[] args) {
        PropertiesHelper helper=new PropertiesHelper();
        System.out.println(helper.getProperty(AppProperties.CONURL.toString()));

        System.out.println(helper.getProperty(AppProperties.DRIVERCLASSNAME.toStri
        ng()));
```

```java
        System.out.println(helper.getProperty(AppProperties.DBUSERNAME.toString()));
        System.out.println(helper.getProperty(AppProperties.DBPASSWORD.toString()));
        try {
            Class.forName(helper.getProperty(AppProperties.DRIVERCLASSNAME.toString()));
            System.out.println("++++ Driver Loaded ++++");
            Connection con=DriverManager.getConnection(
                    helper.getProperty(AppProperties.CONURL.toString())
                    , helper.getProperty(AppProperties.DBUSERNAME.toString())
                    , helper.getProperty(AppProperties.DBPASSWORD.toString()));
            System.out.println("++++ Connected To DB ++++");
            DatabaseMetaData dmd=con.getMetaData();
            System.out.println("Database Product :: "+dmd.getDatabaseProductName());
            System.out.println("Database Version :: "+dmd.getDatabaseProductVersion());
            System.out.println("Driver :: "+dmd.getDriverName());
            System.out.println("Driver Version :: "+dmd.getDriverVersion());
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

}
---------------------EmployeeDAO.java---------------------
package com.ameya.daos;

import java.util.List;

import com.ameya.domain.Employee;

public interface EmployeeDAO {

    void createConnection();
    void addEmployee(Employee employee);
    Employee getEmployee(int empId);
    List<Employee> getAllEmployees();
    void updateEmployee(Employee employee);
    void deleteEmployee(int empId);
    void closeConnection();

}
------------------EmployeeDAOImpl.java---------------------
package com.ameya.daos.impl;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import com.ameya.daos.EmployeeDAO;
import com.ameya.domain.AppProperties;
import com.ameya.domain.Employee;
import com.ameya.helpers.PropertiesHelper;

public class EmployeeDAOImpl implements EmployeeDAO {

    private Connection con;
    private PreparedStatement ps;

    private static String conUrl;
    private static String dbDriver;
    private static String dbUsername;
    private static String dbPassword;

    static {
        PropertiesHelper helper=new PropertiesHelper();
        conUrl=helper.getProperty(AppProperties.CONURL.toString());
        dbDriver=helper.getProperty(AppProperties.DRIVERCLASSNAME.toString());
        dbUsername=helper.getProperty(AppProperties.DBUSERNAME.toString());
        dbPassword=helper.getProperty(AppProperties.DBPASSWORD.toString());
        try {
            Class.forName(dbDriver);
            System.out.println("++++ MySQL Driver Loaded ++++");
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
    @Override
    public void createConnection() {
        try {
            con=DriverManager.getConnection(conUrl, dbUsername, dbPassword);
            System.out.println("++++ Connected To DB ++++");
        } catch (SQLException e) {
            e.printStackTrace();
        }

    }

    @Override
    public void addEmployee(Employee employee) {
        final String SQL = "insert into employee values(?,?,?,?,?)";
        createConnection();
        try {
            ps = con.prepareStatement(SQL);
            ps.setInt(1, employee.getEmpId());
            ps.setString(2, employee.getFirstName());
            ps.setString(3, employee.getLastName());
            ps.setDouble(4, employee.getSalary());
            ps.setInt(5, employee.getAge());
            int cnt = ps.executeUpdate();
            if (cnt != 0) {
                System.out.println("## Row Inserted Into Employee Table ##");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            closeConnection();
        }
    }

    @Override
    public Employee getEmployee(int empId) {
        Employee employee=null;
        final String SQL="select * from employee where empId = ?";
        createConnection();
        try {
            ps=con.prepareStatement(SQL);
            ps.setInt(1, empId);
            ResultSet rs=ps.executeQuery();
            if(rs.next()) {
                employee=new Employee();
                employee.setEmpId(rs.getInt("empid"));
                employee.setFirstName(rs.getString("firstname"));
                employee.setLastName(rs.getString("lastname"));
                employee.setSalary(rs.getDouble("salary"));
                employee.setAge(rs.getInt("age"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }finally {
            closeConnection();
        }
        return employee;
    }

    @Override
    public List<Employee> getAllEmployees() {
        final String SQL="select * from employee";
        ArrayList<Employee> employees=new ArrayList<Employee>();
        createConnection();
        try {
            ps=con.prepareStatement(SQL);
            ResultSet rs=ps.executeQuery();
            while(rs.next()) {
                employees.add(new Employee(
                        rs.getInt("empid"),
                        rs.getString("firstname"),
                        rs.getString("lastname"),
                        rs.getDouble("salary"),
                        rs.getInt("age")));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }finally {
            closeConnection();
        }
        return employees;
    }

    @Override
    public void updateEmployee(Employee employee) {
        final String SQL="update employee set firstname = ? , lastname = ? , salary
        = ? , age = ? where empid = ?";
        createConnection();
        try {
            ps=con.prepareStatement(SQL);
            ps.setString(1, employee.getFirstName());
            ps.setString(2, employee.getLastName());
            ps.setDouble(3, employee.getSalary());
            ps.setInt(4, employee.getAge());
            ps.setInt(5, employee.getEmpId());

            int cnt=ps.executeUpdate();
            if(cnt!=0) {
                System.out.println("#### Employee Record Updated ####");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }finally {
            closeConnection();
        }

    }

    @Override
    public void deleteEmployee(int empId) {
        final String SQL="delete from employee where empid = ?";
        createConnection();
        try {
            ps=con.prepareStatement(SQL);
            ps.setInt(1, empId);
            int cnt=ps.executeUpdate();
            if(cnt!=0) {
                System.out.println("#### Employee Record Deleted ####");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }finally {
            closeConnection();
        }

    }

    @Override
    public void closeConnection() {
        if(con!=null) {
            try {
                con.close();
                System.out.println("++++ DB Connection Closed ++++");
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }

}
--------------------EmployeeService.java----------------------
package com.ameya.services;

import java.util.List;

import com.ameya.domain.Employee;

public interface EmployeeService {

    void createEmployee(Employee employee);
    void modifyEmployee(Employee employee);
    void removeEmployee(int empId);
    Employee findEmployeeById(int empId);
    List<Employee> findAll();
}
--------------------EmployeeServiceImpl.java----------------------
package com.ameya.services.impl;
```

```java

import java.util.List;

import com.ameya.daos.EmployeeDAO;
import com.ameya.domain.Employee;
import com.ameya.services.EmployeeService;

public class EmployeeServiceImpl implements EmployeeService {

    private EmployeeDAO employeeDao;
    public EmployeeServiceImpl() {

    }
    public EmployeeServiceImpl(EmployeeDAO employeeDao) {
        this.employeeDao=employeeDao;
    }
    @Override
    public void createEmployee(Employee employee) {
        employeeDao.addEmployee(employee);
    }

    @Override
    public void modifyEmployee(Employee employee) {
        employeeDao.updateEmployee(employee);

    }

    @Override
    public void removeEmployee(int empId) {
        employeeDao.deleteEmployee(empId);

    }

    @Override
    public Employee findEmployeeById(int empId) {
        return employeeDao.getEmployee(empId);
    }

    @Override
    public List<Employee> findAll() {
        return employeeDao.getAllEmployees();
    }

}
----------------------TestJdbc.java----------------------
package com.ameya.test;

import java.util.List;

import com.ameya.daos.impl.EmployeeDAOImpl;
import com.ameya.domain.Employee;
import com.ameya.services.EmployeeService;
import com.ameya.services.impl.EmployeeServiceImpl;

public class TestJdbc {

    public static void main(String[] args) {
        EmployeeService empService=new EmployeeServiceImpl(new EmployeeDAOImpl());
        empService.createEmployee(new Employee(1,"Ameya","Joshi",45000,42));
        empService.createEmployee(new Employee(2,"Amol","Patil",47000,41));
        empService.createEmployee(new Employee(3,"Amit","Shah",55000,43));
        empService.createEmployee(new Employee(4,"Sanjay","Kadam",65000,41));
        empService.createEmployee(new Employee(5,"Rahul","Pawar",55000,42));
        System.out.println("++++++++++++++++++++++++++++++++++++++++++");
        Employee emp=empService.findEmployeeById(3);
        System.out.println(emp);
        System.out.println("+++++++++++++++++++++++++++++++++++++++++++");
        empService.modifyEmployee(new Employee(3,"Pratap","Shah",66000,47));
        System.out.println("++++++++++++++++++++++++++++++++++++++++++++");
        emp=empService.findEmployeeById(3);
        System.out.println(emp);
        System.out.println("+++++++++++++++++++++++++++++++++++++++++++");
        List<Employee> emps=empService.findAll();
        for(Employee e : emps) {
            System.out.println(e);
        }
        System.out.println("+++++++++++++++++++++++++++++++++++++++++++++");
        empService.removeEmployee(3);
        emps=empService.findAll();
        for(Employee e : emps) {
            System.out.println(e);
        }
    }

}

------------------------------Employee table--------------------
CREATE TABLE `employee` (
    `empid` INT(11) NOT NULL,
    `firstname` VARCHAR(30) NOT NULL DEFAULT '' COLLATE 'latin1_swedish_ci',
    `lastname` VARCHAR(30) NOT NULL DEFAULT '' COLLATE 'latin1_swedish_ci',
    `salary` DOUBLE NOT NULL DEFAULT '0',
    `age` INT(11) NOT NULL DEFAULT '0',
    PRIMARY KEY (`empid`) USING BTREE
)
COLLATE='latin1_swedish_ci'
```

```
ENGINE=InnoDB
;
```