

1. Write a servlet application for registration page to store the given details by the user in the database. Use the following query in MySQL for creating the table to store the data. create table registration (name varchar (20), password varchar (10), email varchar (20), mobile bigint, gender varchar (6), country varchar (10));

```

• Connection      Interface      package
com.jdbc.demo.connection;      //4
Connection      interface      public
interface      dbDetails      {      String
CONSTR =
"jdbc:mysql://localhost:3306/cdac_tvm?useSSL=false";
      String DBDDRIVER = "com.mysql.cj.jdbc.Driver";
                        String USERNAME = "root";
      String PASSWORD = "Rushi@123";
}

```

```

-----

• Connection      package
com.jdbc.demo.connection;      // 5
connection implementation import
java.sql.Connection;      import
java.sql.DriverManager;      import
java.sql.SQLException;

public class DbConnection { public static
      Connection getDbConnection() {

      try {
            Class.forName(dbDetails.DBDDRIVER);

            Connection con=
            DriverManager.getConnection(dbDetails.CONSTR,dbDetails.US
ERNAME,dbDetails.PASSWORD);
            return con;
      }
      catch(ClassNotFoundException |SQLException exc) {
            exc.printStackTrace(); return null;
      }
}

}

```

## ❑ EMPLOYEE POJO CLASS

```
package
com.jdbc.demo.pojo; //1
Employee class public
class Employee { private
int id; private String
ename; private int age;
private int salary; public
Employee() {

    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getEname() {
        return ename;
    }
    public void setEname(String ename) {
        this.ename = ename;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    public int getSalary() {
        return salary;
    }
    public void setSalary(int salary) {
        this.salary = salary;
    }
    @Override
    public String toString() { return "Employee [id=" + id +
        ", ename=" + ename + ",
age=" + age + ", salary=" + salary + "]\n";
    }

}
```

• **Employee DAO CLASS** package  
com.jdbc.demo.dao; //2 interface

```

EmployeeDao import java.util.List;
import com.jdbc.demo.pojo.Employee;

public interface EmployeeDao {

    //query Operations
    List<Employee> getAllEmployee();
    Employee searchEmployee(int EmpId);
    //curd
    boolean addNewEmployee(Employee Empmloyee);
    boolean updateEmployee(Employee Employee);
    boolean deleteEmployee(Employee EmpId);

}

```

• IMPLEMENTATION OF EMPLOYEE DAO

```

CLASS package com.jdbc.demo.empImp;
import java.sql.Connection; import
java.sql.PreparedStatement; import
java.sql.ResultSet; import
java.sql.SQLException; import
java.sql.Statement; import
java.util.ArrayList; //3 implement
employeeDao import java.util.List;

import com.jdbc.demo.connection.DbConnection;
import com.jdbc.demo.dao.EmployeeDao; import
com.jdbc.demo.pojo.Employee; public class
EmployeeDaoImp implements EmployeeDao{

    @Override
    public List<Employee> getAllEmployee() {
        List<Employee> lst=new ArrayList<>();
        try(Connection con=DbConnection.getDbConnection()){
            PreparedStatement pst=con.prepareStatement("SELECT *
FROM Employee");
            ResultSet rs=pst.executeQuery();
            while(rs.next()) {
                Employee emp=new Employee();
                emp.setId(rs.getInt("eid"));
                emp.setEname(rs.getString("ename"));
                emp.setAge(rs.getInt("age"));
                emp.setSalary(rs.getInt("salary"));
                lst.add(emp);          }
            return lst;
        }
    }
}

```

```

    }
    catch (NullPointerException | SQLException exc) {
        exc.printStackTrace(); return null;
    }
}

@Override
public Employee searchEmployee(int EmpId) {
    Employee emp=null; try(Connection
con=DbConnection.getDbConnection()){
        PreparedStatement pst=con.prepareStatement("SELECT *
FROM Employee WHERE eid=?");
        //at the place of first ? value of EmpId
parameter must be there pst.setInt(1,EmpId);
        ResultSet rs=pst.executeQuery();
        if(rs.isBeforeFirst()) { rs.next();
        emp=new Employee();
        emp.setId(rs.getInt("eid"));
        emp.setEname(rs.getString("ename"));
        emp.setAge(rs.getInt("age"));
        emp.setSalary(rs.getInt("salary"));
        return
        emp; }
        return
        emp;
    } catch (SQLException|NullPointerException
exc)
    {
        exc.printStackTrace();
        return null;
    }
}

```

```

@Override
public boolean addNewEmployee(Employee Employee) {
    try(Connection con=DbConnection.getDbConnection()){
        PreparedStatement pst=con.prepareStatement("INSERT
INTO Employee(ename,age,salary)VALUES (?, ?, ?)",
        Statement.RETURN_GENERATED_KEYS);
        pst.setString(1,Employee.getEname());
        pst.setInt(2,Employee.getAge());
        pst.setInt(3, Employee.getSalary()); int
count=pst.executeUpdate(); ResultSet
rs=pst.getGeneratedKeys(); rs.next();
        System.out.println("generated id is"+rs.getInt(1));
        if(count>0) { return true;
        } else { return
        false;
    }
}

```

```

    }
}
catch(SQLException | NullPointerException
exc){ exc.printStackTrace(); return
false;

}
}

@Override
public boolean updateEmployee(Employee Employee) {
    try(Connection con=DbConnection.getDbConnection()){
        PreparedStatement
pst=con.prepareStatement("UPDATE Employee SET
ename=?,age=?,salary=?"
                        + " WHERE eid=?");
        pst.setString(1,Employee.getEname())
        ; pst.setInt(2, Employee.getAge());
        pst.setInt(3, Employee.getSalary());
        pst.setInt(4, Employee.getId()); int
count =pst.executeUpdate();
        if(count>0) { return true;
        } else { return
false;
        }
    }

    catch(SQLException | NullPointerException
exc){ exc.printStackTrace(); return
false;

    }
}

@Override
public boolean deleteEmployee(Employee EmpId) {
    // TODO Auto-generated method stub return
false;
}
}

```

## □ Main class

```
package com.jdbcdemo.main;
```

```

import java.util.List;
import java.util.Scanner;

import com.jdbc.demo.dao.EmployeeDao;
import
com.jdbc.demo.empImp.EmployeeDaoImp;
import com.jdbc.demo.pojo.Employee; public
class AppMain {

    public static void main(String[] args) {

        //ADD NEW ROW
        EmployeeDaoImp daoImp=new EmployeeDaoImp();
        Scanner sc=new Scanner(System.in);

        System.out.println("Enter the name");
        String name=sc.next();

        System.out.println("Enter the age");
        int age=sc.nextInt();

        System.out.println("Enter the Salary");
        int salary=sc.nextInt();

        Employee emp=new Employee();
        emp.setEname(name);

        emp.setAge(age);
        emp.setSalary(salary);
        if(daoImp.addNewEmployee(emp)) {
            System.out.println("Employee Save");
        }
        else
        {
            System.out.println("Employee Not save");
        }

    }

}


```

```
46
47     System.out.println("Enter the name");
48     String name=sc.next();
49
50     System.out.println("Enter the age");
51     int age=sc.nextInt();
52
53     System.out.println("Enter the Salary");
54     int salary=sc.nextInt();
55
56     Employee emp=new Employee();
57     emp.setName(name);
58
59     emp.setAge(age);
60     emp.setSalary(salary);
61     if(daoImp.addNewEmployee(emp)) {
62         System.out.println("Employee Save");
63     }
64     else {
65         System.out.println("Employee Not save");
66     }
}
```

Enter the name  
nikhil  
Enter the age  
48  
Enter the Salary  
4000000  
Generated id is5  
Employee Save

Result Grid				
Filter Rows:				
	eid	ename	age	salary
▶	1	atharva	23	1000
	2	Kshitij	21	500000
	3	pranit	24	2000
	4	RAHUL	33	5000
	5	nikhil	48	4000000
●	NULL	NULL	NULL	NULL

13

Result Grid     Filter Rows: <input type="text"/>   Edit				
	eid	ename	age	salary
▶	1	atharva	23	1000
	2	Kshitij	21	500000
	3	pranit	24	2000
	4	RAHUL	33	5000
•	NULL	NULL	NULL	NULL

c) Selecting rows using parameter in the Where clause  
 ( select \* from emp where age>?)

```

• Connection Interface package
com.jdbc.demo.connection; //4
Connection interface public
interface dbDetails { String
CONSTR =
"jdbc:mysql://localhost:3306/cdac_tvm?useSSL=false";
String DBDDRIVER = "com.mysql.cj.jdbc.Driver";
String USERNAME = "root";
String PASSWORD = "patil123";
}

```

```

• Connection package
com.jdbc.demo.connection; // 5
connection implementation import
java.sql.Connection; import
java.sql.DriverManager; import
java.sql.SQLException;

public class DbConnection { public static
Connection getDbConnection() {

try {
Class.forName(dbDetails.DBDDRIVER);

Connection con=
DriverManager.getConnection(dbDetails.CONSTR,dbDetails.US
ERNAME,dbDetails.PASSWORD);
return con;
}
}

```



```

        catch(ClassNotFoundException | SQLException exc) {
            exc.printStackTrace(); return null;
        }
    }
}

```

### ❑ EMPLOYEE POJO CLASS

```

package
com.jdbc.demo.pojo; //1
Employee class public
class Employee { private
int id; private String
ename; private int age;
private int salary; public
Employee() {

    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getEname() {
        return ename;
    }
    public void setEname(String ename) {
        this.ename = ename;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    public int getSalary() {
        return salary;
    }
    public void setSalary(int salary) {
        this.salary = salary;
    }
    @Override
    public String toString() { return "Employee [id=" + id +
        ", ename=" + ename + ",
age=" + age + ", salary=" + salary + "]\n";
}
}

```

```

    }

}

    • Employee DAO CLASS package
com.jdbc.demo.dao; //2 interface
EmployeeDao import java.util.List;
import com.jdbc.demo.pojo.Employee;
public interface EmployeeDao {

    //query Operations
    List<Employee> getAllEmployee();
    Employee searchEmployee(int EmpId);
    //curd
    boolean addNewEmployee(Employee EmpEmployee);
    boolean updateEmployee(Employee Employee);
    boolean deleteEmployee(Employee EmpId);

}

```

```

    • IMPLEMENTATION OF EMPLOYEE DAO
CLASS package com.jdbc.demo.empImp;
import java.sql.Connection; import
java.sql.PreparedStatement; import
java.sql.ResultSet; import
java.sql.SQLException; import
java.sql.Statement; import
java.util.ArrayList; //3 implement
employeeDao import java.util.List;

import com.jdbc.demo.connection.DbConnection;
import com.jdbc.demo.dao.EmployeeDao; import
com.jdbc.demo.pojo.Employee; public class
EmployeeDaoImp implements EmployeeDao{

    @Override
    public List<Employee> getAllEmployee() {
        List<Employee> lst=new ArrayList<>();
        try(Connection con=DbConnection.getDbConnection()){
            PreparedStatement pst=con.prepareStatement("SELECT *
FROM Employee");
            ResultSet rs=pst.executeQuery();

```

```

        while(rs.next()) {
            Employee emp=new Employee();
            emp.setId(rs.getInt("eid"));
            emp.setEname(rs.getString("ename"));
            emp.setAge(rs.getInt("age"));
            emp.setSalary(rs.getInt("salary"));
            lst.add(emp);
        }
        return lst;
    }

    catch(NullPointerException |SQLException exc) {
        exc.printStackTrace(); return null;
    }
}

@Override
public Employee searchEmployee(int EmpId) {
    Employee emp=null; try(Connection
con=DbConnection.getDbConnection()){
        PreparedStatement pst=con.prepareStatement("SELECT *
FROM Employee WHERE eid=?");
        //at the place of first ? value of EmpId
parameter must be there pst.setInt(1,EmpId);
        ResultSet rs=pst.executeQuery();

        if(rs.isBeforeFirst()) { rs.next();
            emp=new Employee();
            emp.setId(rs.getInt("eid"));
            emp.setEname(rs.getString("ename"));
            emp.setAge(rs.getInt("age"));
            emp.setSalary(rs.getInt("salary"));
            return
            emp; }
        return
        emp;
    } catch(SQLException|NullPointerException
exc)
    {
        exc.printStackTrace();
        return null;
    }
}

@Override
public boolean addNewEmployee(Employee Employee) {
    try(Connection con=DbConnection.getDbConnection()){
        PreparedStatement pst=con.prepareStatement("INSERT
INTO Employee(ename,age,salary)VALUES (?, ?, ?)",

```

```

        Statement.RETURN_GENERATED_KEYS);
        pst.setString(1,Employee.getEname());
        ; pst.setInt(2,Employee.getAge());
        pst.setInt(3, Employee.getSalary());
        int count=pst.executeUpdate();
        ResultSet rs=pst.getGeneratedKeys();
        rs.next();
        System.out.println("generated id
        is"+rs.getInt(1)); if(count>0) { return true;
        } else { return
        false;
        }
    }
    catch(SQLException | NullPointerException exc){
        exc.printStackTrace(); return false;

    }
}

@Override
public boolean updateEmployee(Employee Employee) {
    try(Connection con=DbConnection.getDbConnection()){
        PreparedStatement
pst=con.prepareStatement("UPDATE Employee SET
ename=?,age=?,salary=?"
                        + " WHERE eid=?");
        pst.setString(1,Employee.getEname())
        ; pst.setInt(2, Employee.getAge());
        pst.setInt(3, Employee.getSalary());
        pst.setInt(4, Employee.getId()); int
count =pst.executeUpdate();
        if(count>0) { return true;
        } else { return
        false;
        }
    }

    catch(SQLException | NullPointerException
exc){ exc.printStackTrace(); return
false;

    }

}

@Override
public boolean deleteEmployee(Employee EmpId) {
    // TODO Auto-generated method stub
    return false;
}

```

```

@Override public List<Employee> PrintSelectStmt(int
    Age) { List<Employee> lst=new ArrayList<>();
        try(Connection con=DbConnection.getDbConnection()){
            PreparedStatement
pst=con.prepareStatement("SELECT * FROM Employee WHERE
            age>?"); pst.setInt(1,Age);
            ResultSet rs=pst.executeQuery();
            while(rs.next()) {
                Employee emp=new Employee();
                emp.setId(rs.getInt("eid"));
                emp.setEname(rs.getString("ename"));
                emp.setAge(rs.getInt("age"));
                emp.setSalary(rs.getInt("salary"));
                lst.add(emp);

//                lst.add(new Employee(rs.getInt(1),
rs.getString(2), rs.getInt(3),rs.getInt(4)));
            } return
            lst;

        }
        catch(NullPointerException |SQLException exc) {
            exc.printStackTrace(); return null;

        }
    }
}

```

### □ Main

```

package com.jdbcdemo.main;

import java.util.List;
import java.util.Scanner;
import
com.jdbc.demo.dao.Employeeed
ao; import
com.jdbc.demo.empImp.Emplay
eeDaoImp; import

```

```

com.jdbc.demo.pojo.Employee
; public class AppMain {
public static void
main(String[] args) {

    //Select Query For age
    Scanner sc=new Scanner(System.in);
    EmployeeDaoImp daoImp=new EmployeeDaoImp();
    System.out.println("Enter the age: ");
    int age=sc.nextInt();

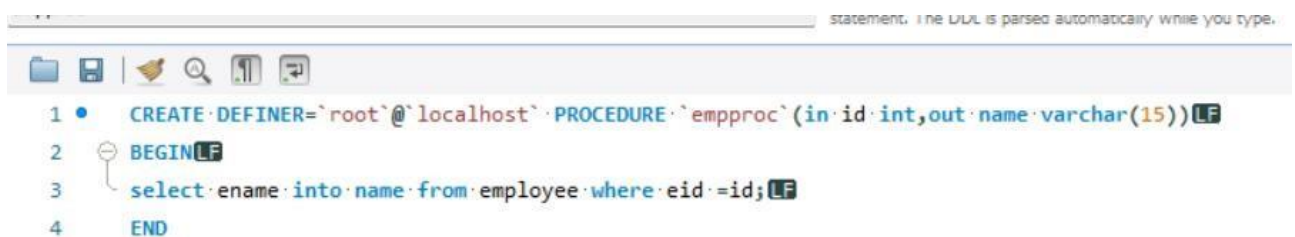
    List
    <Employee>lst=daoImp.PrintSelectStmt(age);

    if(lst.size() > 0) {
        System.out.println("AGE OF employe greater
then : "+age);
        lst.forEach(System.out::println)
        ;
    }
    else
        System.out.println("no employee found");
    }
}

```

**3.Create a stored procedure 'empproc' in the database from MySQL command prompt**

**Using the command: create procedure empproc(In eid int , out  
ename varchar(15)) begin  
select name into ename from emp where id =eid;  
end**



```

5 • call empproc(2,@name);
6 • select @name;
7

```

Result Grid | Filter Rows:  | Export: | Wrap Cell Content:

@name
Kshitij

Write a java application which calls the above procedure

### □ Interface of DbConnection

```

package com.jdbc.demo.connection;
//4 Connection interface public
interface dBDetails { String
CONSTR =
"jdbc:mysql://localhost:3306/cdac_tvm?useSSL=false";
    String DBDDRIVER = "com.mysql.cj.jdbc.Driver";
    String USERNAME = "root";
    String PASSWORD = "patil123";
}
//allowPublicKeyRetrieval=true&

```

### □ Implement Employee DbConnection

```

package com.jdbc.demo.connection; //
5 connection implementation import
java.sql.Connection; import
java.sql.DriverManager;
import java.sql.SQLException;

public class DbConnection { public static
    Connection getDbConnection() {

        try {
            Class.forName(dBDetails.DBDDRIVER);

            Connection con=
                DriverManager.getConnection(dBDetails.CONSTR,dBDetails.US
ERNAME,dBDetails.PASSWORD);
            return con;
        }
        catch(ClassNotFoundException |SQLException exc)
            { exc.printStackTrace(); return null;
            }
    }
}

```

### • Interface class of EmployeeDao

```
package    com.jdbc.demo.dao;    //2
interface    EmployeeDao    import
java.util.List;                import
com.jdbc.demo.pojo.Employee; public
interface EmployeeDao {

    String callProcedure(int Empid);

}
```

### • Implementing of employeeDao package

```
com.jdbc.demo.empImp; import
java.sql.CallableStatement; import
java.sql.Connection; import
java.sql.PreparedStatement; import
java.sql.ResultSet; import
java.sql.SQLException; import
java.sql.Statement; import java.sql.Types;
import java.util.ArrayList; //3 implement
employeeDao
import java.util.List;

import com.jdbc.demo.connection.DbConnection;
import com.jdbc.demo.dao.EmployeeDao; import
com.jdbc.demo.pojo.Employee; import
com.mysql.cj.jdbc.CallableStatement.CallableStatementParamInfo;

public class EmployeeDaoImp implements EmployeeDao{

    @Override
    public String callProcedure(int Empid) { try(Connection
        con=DbConnection.getDbConnection()){
        CallableStatement cs=con.prepareCall("{call
empproc(?,?)}");
        cs.setInt(1,Empid);

        cs.registerOutParameter(2, Types.CHAR);
        cs.execute();
        String result = cs.getString(2);

        return result;
    }
}
```



```

    }
    catch (NullPointerException|SQLException
        exc){ exc.printStackTrace(); return
        null;
    }
}

```

```

}

```

### □ Main

```

package com.jdbcdemo.main;

import java.util.List;
import java.util.Scanner;

import com.jdbc.demo.dao.EmployeeDao;
import
com.jdbc.demo.empImp.EmployeeDaoImp;
import com.jdbc.demo.pojo.Employee; public
class AppMain { public static void
main(String[] args) //Call procedure

    EmployeeDaoImp daoImp=new EmployeeDaoImp();
    Scanner sc=new Scanner(System.in);
    System.out.println("Enter the eId: ");
    int id=sc.nextInt();

    String name=daoImp.callProcedure(id);
    System.out.println(name);
    }

}

```

```
Markers Properties Servers Data Source Explorer Snippets Terminal Console X
<terminated> AppMain [Java Application] C:\Users\91998\Desktop\Eclip\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.4.v20220903-1038\jre\bin\javaw.exe (20-Dec-2022, 1
Enter the eId:
2
Kshitij
```