Q1. Write a program to insert, update and delete records from the given table.

Command Prompt - mysql -u root -p

```
Microsoft Windows [Version 10.0.19045.2364]
(c) Microsoft Corporation. All rights reserved.

C:\Users\admin>mysql -u root -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.31 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

Command Prompt - mysql -u root -p

```
mysql> create database mydb;
Query OK, 1 row affected (0.09 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mydb               |
| mysql              |
| performance_schema |
| sakila             |
| sys                |
| world              |
+--------------------+
7 rows in set (0.00 sec)
```

Command Prompt - mysql -u root -p

```
mysql> use mydb
Database changed
mysql> show tables;
Empty set (0.02 sec)
```

```
Command Prompt - mysql  -u root -p
mysql> create table employee( id int,  name varchar(20),  salary varchar(10));
Query OK, 0 rows affected (0.19 sec)

mysql> show tables;
+----------------+
| Tables_in_mydb |
+----------------+
| employee       |
+----------------+
1 row in set (0.00 sec)
```

Employee.java

```
package com.jdbc;


public class Employee {
	private int id;
	private String name;
	private float salary;
	public int getId() {
		return id;
	}
	public Employee(){}
	public Employee(int id, String name, float salary) {
		super();
		this.id = id;
		this.name = name;
		this.salary = salary;
	}
```

```
        public void setId(int id) {
                this.id = id;
        }
        public String getName() {
                return name;
        }
        public void setName(String name) {
                this.name = name;
        }
        public float getSalary() {
                return salary;
        }
        public void setSalary(float salary) {
                this.salary = salary;
        }

}
```

EmployeeDao.java

```
package com.jdbc;

import java.sql.PreparedStatement;
import java.sql.SQLException;

import org.springframework.dao.DataAccessException;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.PreparedStatementCallback;

public class EmployeeDao {

private  JdbcTemplate jdbcTemplate;

public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
   this.jdbcTemplate = jdbcTemplate;
}

public int saveEmployee(Employee e){
   String query="insert into employee
values('"+e.getId()+"','"+e.getName()+"','"+e.getSalary()+"')";
   return jdbcTemplate.update(query);
}
public int updateEmployee(Employee e){
   String query="update employee set
name='"+e.getName()+"',salary='"+e.getSalary()+"' where id='"+e.getId()+"' ";
```

```
    return jdbcTemplate.update(query);
}
public int deleteEmployee(Employee e){
    String query="delete from employee where id='"+e.getId()+"' ";
    return jdbcTemplate.update(query);
}

Boolean saveEmployeebyPrepared(Employee e) {

        String query="insert into employee values(?,?,?)";
        return jdbcTemplate.execute(query, new
PreparedStatementCallback<Boolean>() {

                @Override
                public Boolean doInPreparedStatement(PreparedStatement ps)
throws SQLException, DataAccessException {
                        ps.setInt(1,e.getId());
            ps.setString(2,e.getName());
            ps.setFloat(3, e.getSalary());
            return ps.execute();
                }



        });
}
}
```

Config.xml

```
package com.jdbc;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;


public class App {

	public static void main(String [] args) {

		ApplicationContext context=new
ClassPathXmlApplicationContext("Config.xml");

		EmployeeDao dao=(EmployeeDao)context.getBean("e123");

		int status=dao.saveEmployee(new Employee(102,"Satya",190000));

		System.out.println(status);


		Employee e=new Employee();

		e.setId(108);

		e.setName("Satya");

		e.setSalary(200000);

		int result= dao.saveEmployee(e);

		System.out.println(result);

		//Deletion

//		e.setId(546);

		System.out.println("101 delete operation" +dao.deleteEmployee(e));

		System.out.println("102 Update operation"
+dao.updateEmployee(new Employee(102,"Sandeep",2000000)));

	}

}
```

App.java

```
package com.jdbc;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;


public class App {

public static void main(String [] args) {

	ApplicationContext context=new
ClassPathXmlApplicationContext("Config.xml");

	EmployeeDao dao=(EmployeeDao)context.getBean("e123");

	int status=dao.saveEmployee(new Employee(102,"Satya",190000));

	System.out.println(status);


	Employee e=new Employee();

	e.setId(108);

	e.setName("Satya");

	e.setSalary(200000);

	int result= dao.saveEmployee(e);

	System.out.println(result);

	//Deletion

	e.setId(546);

	System.out.println("101 delete operation" +dao.deleteEmployee(e));

	System.out.println("102 Update operation" +dao.updateEmployee(new
Employee(102,"Sandeep",2000000)));

}
```

}

Output

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> App (1) [Java Application] C:\Users\satya\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86
Feb 10, 2023 8:23:54 PM org.springframework.beans.factory.xml.XmlBeanDefinitionRe
INFO: Loading XML bean definitions from class path resource [Config.xml]
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class i
Feb 10, 2023 8:23:54 PM org.springframework.jdbc.datasource.DriverManagerDataSour
INFO: Loaded JDBC driver: com.mysql.jdbc.Driver
1
1
101 delete operation1
102 Update operation1
```

```
mysql> select * from employee;
+------+---------+-----------+
| id   | name    | salary    |
+------+---------+-----------+
|  102 | Sandeep | 2000000.0 |
+------+---------+-----------+
1 row in set (0.00 sec)
```

Q2. Write a program to demonstrate PreparedStatement in Spring JdbcTemplate

App.java

```
package com.jdbc;

import java.util.List;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;


public class App {

	public static void main(String[] args) {
```

```
ApplicationContext context = new ClassPathXmlApplicationContext("config.xml");
EmployeeDao dao = (EmployeeDao)context.getBean("e123");
int status = dao.saveEmployee(new Employee(106,"Vivek","10000"));
System.out.println(status);
Employee e= new Employee();
e.setId(99);
status = dao.deleteEmployee(e);
System.out.println(status);
// Q2
// PreparedStatement Callback Interface
Employee e1 = new Employee(2,"Abhishek","600000");
System.out.println(dao.saveEmployeePrepared(new Employee(68,"James","35000")));
}
}
```

EmployeeDao.java

```
package com.jdbc;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.dao.DataAccessException;
```

```
import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.jdbc.core.PreparedStatementCallback;

import org.springframework.jdbc.core.ResultSetExtractor;


public class EmployeeDao {

	private JdbcTemplate jdbcTemplate;

	public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {

	this.jdbcTemplate = jdbcTemplate;

	}

	public int saveEmployee(Employee e){

	String query="insert into employee
values('"+e.getId()+"','"+e.getName()+"','"+e.getSalary()+"')";

	return jdbcTemplate.update(query);

	}

	public int updateEmployee(Employee e){

	String query="update employee set
name='"+e.getName()+"',salary='"+e.getSalary()+"' where id='"+e.getId()+"' ";

	return jdbcTemplate.update(query);

	}

	public int deleteEmployee(Employee e){

	String query="delete from employee where id='"+e.getId()+"' ";

	return jdbcTemplate.update(query);

	}

	Boolean saveEmployeePrepared(Employee e) {

		String query = "insert into employee values(?,?,?)";

		return jdbcTemplate.execute(query,new
PreparedStatementCallback<Boolean>() {
```

```
@Override
public Boolean doInPreparedStatement(PreparedStatement ps)
throws SQLException,DataAccessException{
ps.setInt(1, e.getId());
ps.setString(2, e.getName());
ps.setString(3,e.getSalary());
return ps.execute();
}
});
}
}
```

Employee.java

```
package com.jdbc;


public class Employee {
	private int id;
	public Employee(int id, String name, String salary) {
	super();
	this.id = id;
	this.name = name;
	this.salary = salary;
	}
	public int getId() {
	return id;
	}
	public void setId(int id) {
```

```
this.id = id;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public String getSalary() {
return salary;
}
public void setSalary(String salary) {
this.salary = salary;
}
private String name;
private String salary;
public Employee() {}
}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> App (1) [Java Application] C:\Users\satya\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.wi
INFO: Refreshing org.springframework.context.support.ClassPathXmlApplication
Feb 10, 2023 9:13:25 PM org.springframework.beans.factory.xml.XmlBeanDefinit
INFO: Loading XML bean definitions from class path resource [config.xml]
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver cl
Feb 10, 2023 9:13:26 PM org.springframework.jdbc.datasource.DriverManagerDat
INFO: Loaded JDBC driver: com.mysql.jdbc.Driver
1
0
false
```

Q3. Write a program in Spring JDBC to demonstrate ResultSetExtractor Interface

App.java

```
package com.jdbc;

import java.util.List;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class App {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("config.xml");

EmployeeDao dao = (EmployeeDao)context.getBean("e123");

int status = dao.saveEmployee(new Employee(102,"Sandeep","10000"));

System.out.println(status);

Employee e= new Employee();

e.setId(99);

status = dao.deleteEmployee(e);

System.out.println(status);

// PreparedStatement Callback Interface

Employee e1 = new Employee(03,"Abhishek","600000");
```

```
System.out.println(dao.saveEmployeePrepared(new
Employee(006,"Suraj","35000")));

//Resultextractor to display data

System.out.println("Employee Data by ResultExtractor: ");

List<Employee> list=dao.getAllEmployees();

for (Employee display:list) {

System.out.println(""+ display.getId());

System.out.println(""+ display.getName());

System.out.println(""+ display.getSalary());

System.out.println();

System.out.println("----------------------");

}

}

}
```

Employee.java

```
package com.jdbc;

public class Employee {

private int id;

public Employee(int id, String name, String salary) {

super();

this.id = id;

this.name = name;

this.salary = salary;

}

public int getId() {

return id;
```

```
}
public void setId(int id) {
this.id = id;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public String getSalary() {
return salary;
}
public void setSalary(String salary) {
this.salary = salary;
}
private String name;
private String salary;
public Employee() {}
}
```

EmployeeDao.java

```
package com.jdbc;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
```

```
import java.util.ArrayList;
import java.util.List;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.dao.DataAccessException;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.PreparedStatementCallback;
import org.springframework.jdbc.core.ResultSetExtractor;
public class EmployeeDao {
private JdbcTemplate jdbcTemplate;
public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
this.jdbcTemplate = jdbcTemplate;
}
//Q1
public int saveEmployee(Employee e) {
String query = "insert into employee values"
+ "('" + e.getId()+"','"+e.getName()+"','"+e.getSalary()+"')";
return jdbcTemplate.update(query);
}
public int updateEmployee(Employee e){
String query="update employee set
name='"+e.getName()+"',salary='"+e.getSalary()+"' where id='"+e.getId()+"' ";
return jdbcTemplate.update(query);
}
public int deleteEmployee(Employee e){
String query="delete from employee where id='"+e.getId()+"' ";
return jdbcTemplate.update(query);
```

```
}
//Q2
Boolean saveEmployeePrepared(Employee e) {
String query = "insert into employee values(?,?,?)";
return jdbcTemplate.execute(query,new PreparedStatementCallback<Boolean>() {
@Override
public Boolean doInPreparedStatement(PreparedStatement ps) throws
SQLException,DataAccessException{
ps.setInt(1, e.getId());
ps.setString(2, e.getName());
ps.setString(3,e.getSalary());
return ps.execute();
}
});
}
//Q3
public List<Employee> getAllEmployees(){
return jdbcTemplate.query("select * from employee",new
ResultSetExtractor<List<Employee>>() {
@Override
public List<Employee> extractData(ResultSet rs) throws
SQLException,DataAccessException{
List<Employee> list = new ArrayList<Employee>();
while(rs.next()) {
Employee e = new Employee();
e.setId(rs.getInt(1));
e.setName(rs.getString(2));
```

```
e.setSalary(rs.getString(3));
list.add(e);
}
return list;
}
}
);
}
}
```

Output

```
1
0
false
Employee Data by ResultExtractor:
102
Sandeep
2000000.0

----------------------
102
Sandeep
10000

----------------------
6
Suraj
35000

----------------------
```

Q4. Write a program to demonstrate RowMapper interface to fetch the records from the database.

App.java

```
package com.jdbc;

import java.util.List;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class App {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("config.xml");

EmployeeDao dao = (EmployeeDao)context.getBean("e123");

int status = dao.saveEmployee(new Employee(102,"Sandeep","10000"));

System.out.println(status);

Employee e= new Employee();

e.setId(99);

status = dao.deleteEmployee(e);

System.out.println(status);

// PreparedStatement Callback Interface

Employee e1 = new Employee(03,"Abhishek","600000");

System.out.println(dao.saveEmployeePrepared(new
Employee(006,"Suraj","35000")));

//Resultextractor to display data

System.out.println("Employee Data by ResultExtractor: ");

List<Employee> list=dao.getAllEmployees();

for (Employee display:list) {

System.out.println(""+ display.getId());

System.out.println(""+ display.getName());

System.out.println(""+ display.getSalary());
```

```
System.out.println();
System.out.println("----------------------");
}
//RowMapper
System.out.println("Employee Data by rowmapper: ");
System.out.println("");
List<Employee> list2=dao.getAllEmployeesRowMapper();
for (Employee display:list) {
System.out.println(""+ display.getId());
System.out.println(""+ display.getName());
System.out.println(""+ display.getSalary());
System.out.println();
System.out.println("----------------------");
}
}
}
```

Employee.java

```
package com.jdbc;
public class Employee {
private int id;
public Employee(int id, String name, String salary) {
super();
this.id = id;
this.name = name;
this.salary = salary;
```

```
}
public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public String getSalary() {
return salary;
}
public void setSalary(String salary) {
this.salary = salary;
}
private String name;
private String salary;
public Employee() {}
}
```

EmployeeDao.java

```
package com.jdbc;
```

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.dao.DataAccessException;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.PreparedStatementCallback;
import org.springframework.jdbc.core.ResultSetExtractor;
public class EmployeeDao {
private JdbcTemplate jdbcTemplate;
public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
this.jdbcTemplate = jdbcTemplate;
}

public int saveEmployee(Employee e) {
String query = "insert into employee values"
+ "('" + e.getId()+"','"+e.getName()+"','"+e.getSalary()+"')";
return jdbcTemplate.update(query);
}
public int updateEmployee(Employee e){
String query="update employee set
name='"+e.getName()+"',salary='"+e.getSalary()+"' where id='"+e.getId()+"' ";
return jdbcTemplate.update(query);
}
```

```
public int deleteEmployee(Employee e){
String query="delete from employee where id='"+e.getId()+"' ";
return jdbcTemplate.update(query);
}

Boolean saveEmployeePrepared(Employee e) {
String query = "insert into employee values(?,?,?)";
return jdbcTemplate.execute(query,new PreparedStatementCallback<Boolean>() {
@Override
public Boolean doInPreparedStatement(PreparedStatement ps) throws
SQLException,DataAccessException{
ps.setInt(1, e.getId());
ps.setString(2, e.getName());
ps.setString(3,e.getSalary());
return ps.execute();
}
});
}

public List<Employee> getAllEmployees(){
return jdbcTemplate.query("select * from employee",new
ResultSetExtractor<List<Employee>>() {
@Override
public List<Employee> extractData(ResultSet rs) throws
SQLException,DataAccessException{
List<Employee> list = new ArrayList<Employee>();
while(rs.next()) {
```

```
Employee e = new Employee();
e.setId(rs.getInt(1));
e.setName(rs.getString(2));
e.setSalary(rs.getString(3));
list.add(e);
}
return list;
}
}
);
}
public List<Employee> getAllEmployeesRowMapper(){
return jdbcTemplate.query("select * from employee", new
RowMapper<Employee>() {
public Employee mapRow(ResultSet rs,int rownumber) throws SQLException{
Employee e = new Employee();
e.setId(rs.getInt(1));
e.setName(rs.getNString(2));
e.setSalary(rs.getNString(3));
return e;
}
});
}
}
```

Output

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> App [Java Application] C:\Users\satya\.p2\pool\plugins\org
Employee Data by rowmapper:

102
Sandeep
2000000.0

----------------------
102
Sandeep
10000

----------------------
6
Suraj
35000

----------------------
102
Sandeep
10000

----------------------
6
Suraj
35000

----------------------
```