

Practical 5

Aim: Demonstrate applications using JDBC with Spring

Theory:

Spring JDBC:

Spring JDBC is a module of the Spring framework that provides templates and APIs for simplifying the JDBC (Java Database Connectivity) operations. It reduces the amount of code needed for database operations and makes it easier to work with databases.

DAO (Data Access Object):

DAO is a design pattern that provides an abstract interface for accessing data from a database. It separates the data access logic from the business logic and makes it easier to test and maintain the code. In Spring, you can create DAO classes that implement the DAO interface and perform database operations.

RowMapper:

RowMapper is an interface in Spring JDBC that maps a single row of a result set to a Java object. It provides a way to map the result set to a custom object instead of using the result set directly. In Spring, you can create a custom implementation of the RowMapper interface to map result set rows to custom objects.

PreparedStatement:

A PreparedStatement is an interface in JDBC that represents a precompiled SQL statement. It provides a way to execute SQL statements with variables. PreparedStatements are faster and more secure than regular Statement objects. In Spring, you can use the JdbcTemplate's query methods that accept a PreparedStatementCreator or a CallableStatementCreator to execute a PreparedStatement.

ResultExtractor:

ResultExtractor is an interface in Spring JDBC that provides a way to extract the results of a query into a single value or a custom object. It can be used to extract the results of a query that returns a single row or multiple rows. In Spring, you can create a custom implementation of the ResultExtractor interface to extract the results of a query into a custom object.

Questions:

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

App.java

```
package finny.vesit.jdbc;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.jdbc.core.JdbcTemplate;

public class App {
    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext("config.xml");
```

```

JdbcTemplate jdbcTemplate = context.getBean("JdbcTemplate", JdbcTemplate.class);
EmployeeDAO employeeDAO = new EmployeeDAO(jdbcTemplate);
Employee employee1 = new Employee(1, "Prathamesh Ingale", 60000.00);
Employee employee2 = new Employee(2, "Clark Kent", 25000.00);
Employee employee3 = new Employee(3, "Chloe Sullivan", 50000.00);
Employee employee4 = new Employee(4, "Lois Lane", 75000.00);
System.out.println("Employee Inserted:
"+employeeDAO.insertEmployee(employee1)+employee1);
System.out.println("Employee Inserted:
"+employeeDAO.insertEmployee(employee2)+employee2);
System.out.println("Employee Inserted:
"+employeeDAO.insertEmployee(employee3)+employee3);
System.out.println("Employee Inserted:
"+employeeDAO.insertEmployee(employee4)+employee4);
// Update
System.out.println("Increment of Employee 1");
Employee updatedEmployee = new Employee(employee1.getId(), employee1.getName(),
80000.00);
System.out.println("Employee Updated: "
+employeeDAO.updateEmployee(employee1,updatedEmployee)
+employee1);
// deletion
System.out.println("Employee Deleted: "
+employeeDAO.deleteEmployee(employee3)
+employee3);
}

```

Employee.java

```
package finny.vesit.jdbc;
```

```

public class Employee {
    private int id;
    private String name;
    private Double salary;
    public Employee(int id, String name, Double salary) {
        this.id = id;
        this.name = name;
        this.salary = salary;
    }
    public Employee() {
    }
    @Override
    public String toString() {
        return "Employee{" +
            "id=" + id +
            ", name=" + name + "\" +
            ", 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 Double getSalary() {
    return salary;
}
public void setSalary(Double salary) {
    this.salary = salary;
}
}
```

EmployeeDAO.java

```
package finny.vesit.jdbc;
```

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

```
import java.sql.ResultSet;
```

```
public class EmployeeDAO {
    JdbcTemplate jdbcTemplate;
```

```
    public EmployeeDAO(JdbcTemplate jdbcTemplate) {
        this.jdbcTemplate = jdbcTemplate;
    }
```

```
    public JdbcTemplate getJdbcTemplate() {
        return jdbcTemplate;
    }
```

```
    public int insertEmployee(Employee e){
        String query = "INSERT INTO employee (id, name, salary) values (?, ?, ?)";
        int result = this.jdbcTemplate.update(query,
            e.getId(), e.getName(), e.getSalary());
        return result;
    }
```

```
    public int updateEmployee(Employee eOriginal, Employee eNew){
        String query = "UPDATE employee SET name=?, salary=? WHERE id=?";
        int result = this.jdbcTemplate.update(query,
            eNew.getName(), eNew.getSalary(), eOriginal.getId());
        return result;
    }
```

```
    public int deleteEmployee(Employee employee){
        String query = "DELETE FROM employee WHERE id=?";
        int result = this.jdbcTemplate.update(query, employee.getId());
        return result;
    }
}
```

config.xml

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

<beans xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns="http://www.springframework.org/schema/beans"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-3.0.xsd">
    <!--      xmlns:p="http://www.springframework.org/schema/p"-->

    <!-- Initialization for data source -->
    <bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="com.mysql.cj.jdbc.Driver"/>
        <property name="url" value="jdbc:mysql://localhost:3306/jdbc"/>
        <property name="username" value="root"/>
        <property name="password" value="root"/>
    </bean>
    <!-- Definition for studentJdbcTemplate bean -->
    <bean id="JdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
        <property name="dataSource" ref="dataSource"/>
    </bean>
</beans>
```

Output :-

```
<terminated> App (1) [Java Application] C:\Users\LENOVO\.p2\pool\plugins\org.eclipse.justj.o
Hello World!
Employee Inserted: 1Employee{id=1, name='Finny', salary=60000.0}
Employee Inserted: 1Employee{id=2, name='Sreenivas', salary=25000.0}
Employee Inserted: 1Employee{id=3, name='Sunny', salary=50000.0}
Employee Inserted: 1Employee{id=4, name='Omkar', salary=75000.0}
Increment of Employee 1
Employee Updated: 1Employee{id=1, name='Finny', salary=60000.0}
Employee Deleted: 1Employee{id=1, name='Finny', salary=60000.0}
```

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

App.java

```
package finny.vesit.jdbc;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.jdbc.core.JdbcTemplate;

import java.util.List;
import java.util.ListIterator;

public class App {
    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext("config.xml");
```

```
JdbcTemplate jdbcTemplate = context.getBean("JdbcTemplate", JdbcTemplate.class);
EmployeeDAO employeeDAO = new EmployeeDAO(jdbcTemplate);
List<Employee> employeeList = employeeDAO.getAllEmployees();
ListIterator<Employee> employeeListIterator = employeeList.listIterator();
System.out.println("Employees: ");
while (employeeListIterator.hasNext()){
    System.out.println(employeeListIterator.next());
}
}
```

Employee.java

```
package finny.vesit.jdbc;

public class Employee {
    private int id;
    private String name;
    private Double salary;

    public Employee(int id, String name, Double salary) {
        this.id = id;
        this.name = name;
        this.salary = salary;
    }

    public Employee() {
    }

    @Override
    public String toString() {
        return "Employee{" +
            "id=" + id +
            ", name=" + name + "\" +
            ", 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 Double getSalary() {  
        return salary;  
    }  
  
    public void setSalary(Double salary) {  
        this.salary = salary;  
    }  
}
```

EmployeeDAO.java

```
package finny.vesit.jdbc;  
  
import org.springframework.jdbc.core.JdbcTemplate;  
import java.util.List;  
  
public class EmployeeDAO {  
    JdbcTemplate jdbcTemplate;  
    public EmployeeDAO(JdbcTemplate jdbcTemplate) {  
        this.jdbcTemplate = jdbcTemplate;  
    }  
  
    public JdbcTemplate getJdbcTemplate() {  
        return jdbcTemplate;  
    }  
    public List<Employee> getAllEmployees(){  
        return jdbcTemplate.query("SELECT * FROM employee", new EmployeeResultSetExtractor());  
    }  
}
```

EmployeeRowMapper.java

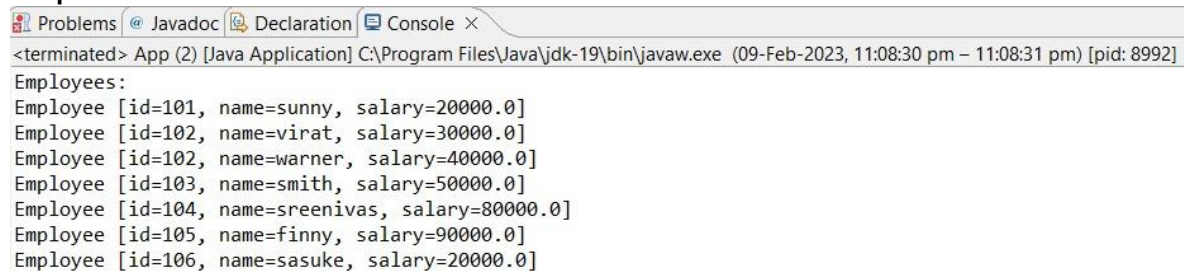
```
package paxy.p5q2;  
  
import org.springframework.jdbc.core.RowMapper;  
  
import java.sql.ResultSet;  
import java.sql.SQLException;  
  
public class EmployeeRowMapper implements RowMapper<Employee> {  
  
    public Employee mapRow(ResultSet rs, int rowNum) throws SQLException {  
        return new Employee(rs.getInt(1),  
            rs.getString(2),  
            rs.getDouble(3));  
    }  
}
```

config.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
  
<beans xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xmlns:context="http://www.springframework.org/schema/context"
xmlns="http://www.springframework.org/schema/beans"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-3.0.xsd ">
<!--      xmlns:p="http://www.springframework.org/schema/p"-->

<!-- Initialization for data source -->
<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
  <property name="driverClassName" value="com.mysql.cj.jdbc.Driver"/>
  <property name="url" value="jdbc:mysql://localhost:3306/jdbc"/>
  <property name="username" value="root"/>
  <property name="password" value="root"/>
</bean>
<!-- Definition for studentJdbcTemplate bean -->
<bean id="JdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
  <property name="dataSource" ref="dataSource"/>
</bean>
</beans>
```

Output:-

```
<terminated> App (2) [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (09-Feb-2023, 11:08:30 pm – 11:08:31 pm) [pid: 8992]
Employees:
Employee [id=101, name=sunny, salary=20000.0]
Employee [id=102, name=virat, salary=30000.0]
Employee [id=102, name=warner, salary=40000.0]
Employee [id=103, name=smith, salary=50000.0]
Employee [id=104, name=sreenivas, salary=80000.0]
Employee [id=105, name=finny, salary=90000.0]
Employee [id=106, name=sasuke, salary=20000.0]
```

3. Write a program to demonstrate PreparedStatement in Spring JdbcTemplate**App.java**

```
package finny.vesit.jdbc;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.jdbc.core.JdbcTemplate;

public class App {
    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext("config.xml");

        JdbcTemplate jdbcTemplate = context.getBean("JdbcTemplate", JdbcTemplate.class);
        EmployeeDAO employeeDAO = new EmployeeDAO(jdbcTemplate);
        System.out.println("Employee: "+employeeDAO.getEmployees("Prathamesh Ingale"));
        System.out.println("Employee: "+employeeDAO.getEmployees("Clark Kent"));
    }
}
```

Employee.java

```
package finny.vesit.jdbc;
```

```
public class Employee {
    private int id;
    private String name;
    private Double salary;

    public Employee(int id, String name, Double salary) {
        this.id = id;
        this.name = name;
        this.salary = salary;
    }

    public Employee() {
    }

    @Override
    public String toString() {
        return "Employee{" +
            "id=" + id +
            ", name=" + name + "\" +
            ", 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 Double getSalary() {
        return salary;
    }

    public void setSalary(Double salary) {
        this.salary = salary;
    }
}
```

EmployeeDAO.java

```
package finny.vesit.jdbc;
```



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

import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.util.List;

public class EmployeeDAO {
    JdbcTemplate jdbcTemplate;
    public EmployeeDAO(JdbcTemplate jdbcTemplate) {
        this.jdbcTemplate = jdbcTemplate;
    }
    public JdbcTemplate getJdbcTemplate() {
        return jdbcTemplate;
    }
    public List<Employee> getEmployees(final String name){
        String query = "SELECT * FROM employee WHERE name=?";
        return jdbcTemplate.query(query, new PreparedStatementSetter() {
            public void setValues(PreparedStatement ps) throws SQLException {
                ps.setString(1,name);
            }
        },new EmployeeRowMapper());
    }
}
```

EmployeeRowMapper.java

```
package paxy.p5q3;

import org.springframework.jdbc.core.RowMapper;

import java.sql.ResultSet;
import java.sql.SQLException;

public class EmployeeRowMapper implements RowMapper<Employee> {

    public Employee mapRow(ResultSet rs, int rowNum) throws SQLException {
        return new Employee(rs.getInt(1),
            rs.getString(2),
            rs.getDouble(3));
    }
}
```

config.xml

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

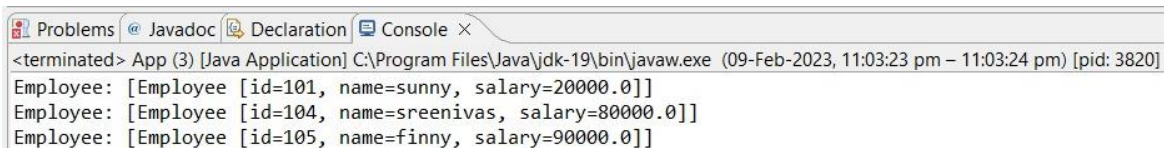
<beans xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns="http://www.springframework.org/schema/beans"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-3.0.xsd ">
```

```

<!-- xmlns:p="http://www.springframework.org/schema/p"-->

<!-- Initialization for data source -->
<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
    <property name="driverClassName" value="com.mysql.cj.jdbc.Driver"/>
    <property name="url" value="jdbc:mysql://localhost:3307/jdbc"/>
    <property name="username" value="root"/>
    <property name="password" value="root"/>
</bean>
<!-- Definition for studentJdbcTemplate bean -->
<bean id="JdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
    <property name="dataSource" ref="dataSource"/>
</bean>
</beans>

```

Output :-


```

<terminated> App (3) [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (09-Feb-2023, 11:03:23 pm - 11:03:24 pm) [pid: 3820]
Employee: [Employee [id=101, name=sunny, salary=20000.0]]
Employee: [Employee [id=104, name=sreenivas, salary=80000.0]]
Employee: [Employee [id=105, name=finny, salary=90000.0]]

```

4. Write a program in Spring JDBC to demonstrate ResultSetExtractor Interface**App.java**

```

package finny.vesit.jdbc;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.jdbc.core.JdbcTemplate;
import java.util.List;
import java.util.ListIterator;

public class App {
    public static void main(String[] args) {
        ApplicationContext context = new ClassPathXmlApplicationContext("config.xml");

        JdbcTemplate jdbcTemplate = context.getBean("JdbcTemplate", JdbcTemplate.class);
        EmployeeDAO employeeDAO = new EmployeeDAO(jdbcTemplate);
        List<paxy.p5q4.Employee> employeeList = employeeDAO.getAllEmployees();
        ListIterator<Employee> employeeListIterator = employeeList.listIterator();
        System.out.println("Employees: ");
        while (employeeListIterator.hasNext()){
            System.out.println(employeeListIterator.next());
        }
    }
}

```

Employee.java

```

package paxy.p5q4;

```

```
public class Employee {
    private int id;
    private String name;
    private Double salary;

    public Employee(int id, String name, Double salary) {
        this.id = id;
        this.name = name;
        this.salary = salary;
    }

    public Employee() {
    }

    @Override
    public String toString() {
        return "Employee{" +
            "id=" + id +
            ", name=" + name + "\" +
            ", 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 Double getSalary() {
        return salary;
    }

    public void setSalary(Double salary) {
        this.salary = salary;
    }
}
```

EmployeeDAO.java

```
package finny.vesit.jdbc;
```

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

```
import java.util.List;

public class EmployeeDAO {
    JdbcTemplate jdbcTemplate;
    public EmployeeDAO(JdbcTemplate jdbcTemplate) {
        this.jdbcTemplate = jdbcTemplate;
    }

    public JdbcTemplate getJdbcTemplate() {
        return jdbcTemplate;
    }
    public List<Employee> getAllEmployees(){
        return jdbcTemplate.query("SELECT * FROM employee", new EmployeeResultSetExtractor());
    }
}
```

EmployeeResultSetExtractor.java

```
package finny.vesit.jdbc;

import org.springframework.dao.DataAccessException;
import org.springframework.jdbc.core.ResultSetExtractor;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

public class EmployeeResultSetExtractor implements ResultSetExtractor<List<Employee>> {
    public List<Employee> extractData(ResultSet rs) throws SQLException, DataAccessException {
        List<Employee> employeeList = new ArrayList<Employee>();
        while (rs.next()){
            Employee employee = new Employee(rs.getInt(1),
                rs.getString(2),
                rs.getDouble(3));
            employeeList.add(employee);
        }
        return employeeList;
    }
}
```

config.xml

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

<beans xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns="http://www.springframework.org/schema/beans"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-3.0.xsd ">
    <!-- xmlns:p="http://www.springframework.org/schema/p"-->
```

```
<!-- Initialization for data source -->
<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
  <property name="driverClassName" value="com.mysql.cj.jdbc.Driver"/>
  <property name="url" value="jdbc:mysql://localhost:3306/jdbc"/>
  <property name="username" value="root"/>
  <property name="password" value="root"/>
</bean>
<!-- Definition for studentJdbcTemplate bean -->
<bean id="JdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
  <property name="dataSource" ref="dataSource"/>
</bean>
</beans>
```

Output:-

```
<terminated> App (4) [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (09-Feb-2023, 11:01:42 pm – 11:01:43 pm) [pid: 8812]
Employees:
Employee [id=101, name=sunny, salary=20000.0]
Employee [id=102, name=virat, salary=30000.0]
Employee [id=102, name=warner, salary=40000.0]
Employee [id=103, name=smith, salary=50000.0]
Employee [id=104, name=sreenivas, salary=80000.0]
Employee [id=105, name=finny, salary=90000.0]
```

Conclusion:

I have successfully understood how to use spring JDBC for Database Operations.