|  |  |  |  |
| --- | --- | --- | --- |
| **Name of Student: Pushkar Sane** | | | |
| **Roll Number: 45** | | **Lab Assignment Number: 7** | |
| **Title of Lab Assignment: Assignment based on Spring JDBC.** | | | |
| **DOP: 24-10-2023** | | **DOS: 28-10-2023** | |
| **CO Mapped:**  **CO5** | **PO Mapped:**  **PO1, PO2, PO3, PO11, PSO1** | | **Signature:** |

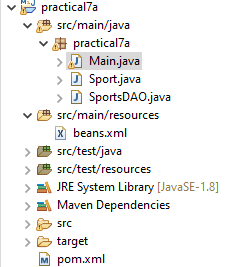
**Practical No. 7**

**Aim:**

1. Create table sports (name, type, no of players). Use spring jdbc to insert 3 records in

the sports table. Delete one record from the table using the same concept.

**File Structure:**

****

**Code:**

**Main.java**

package practical7a;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

public static void main(String[] args) {

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

SportsDAO sportsDAO = (SportsDAO) context.getBean("sportsDAO");

System.out.println("inserting 3 records \n" + "Football " + "Team " + 11 + "\n" + "Basketball " + "Team " + 5

+ "\n" + "Tennis " + "Singles " + 2);

// Insert records

sportsDAO.insertSportsRecord("Football", "Team", 11);

sportsDAO.insertSportsRecord("Basketball", "Team", 5);

sportsDAO.insertSportsRecord("Tennis", "Singles", 2);

// Display records after inserting

System.out.println("\ndisplay table after inserting 3 records");

sportsDAO.displayAllSportsRecords();

// Update a record

sportsDAO.updateSportsRecord(1, "Cricket", "Team", 11);

// Display records after updating

System.out.println("\ndisplay table after updating Football -> Cricket");

sportsDAO.displayAllSportsRecords();

// Delete a record

sportsDAO.deleteSportsRecord(1);

// Display records after deleting

System.out.println("\ndisplay table after deleting id = 1 ");

sportsDAO.displayAllSportsRecords();

}

}

**SportsDAO.java**

package practical7a;

import org.springframework.jdbc.core.JdbcTemplate;

import java.util.List;

import org.springframework.jdbc.core.BeanPropertyRowMapper;

public class SportsDAO {

private JdbcTemplate jdbcTemplate;

public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {

this.jdbcTemplate = jdbcTemplate;

}

public void insertSportsRecord(String name, String type, int noOfPlayers) {

String sql = "INSERT INTO sports (name, type, no\_of\_players) VALUES (?, ?, ?)";

jdbcTemplate.update(sql, name, type, noOfPlayers);

}

public void deleteSportsRecord(int id) {

String sql = "DELETE FROM sports WHERE id = ?";

jdbcTemplate.update(sql, id);

}

public void updateSportsRecord(int id, String name, String type, int noOfPlayers) {

String sql = "UPDATE sports SET name = ?, type = ?, no\_of\_players = ? WHERE id = ?";

jdbcTemplate.update(sql, name, type, noOfPlayers, id);

}

public List<Sport> getAllSportsRecords() {

String sql = "SELECT \* FROM sports";

return jdbcTemplate.query(sql, new BeanPropertyRowMapper<>(Sport.class));

}

public void displayAllSportsRecords() {

List<Sport> sports = this.getAllSportsRecords();

for (Sport sport : sports) {

System.out.println("Sport " + sport.getId() + " Name: " + sport.getName() + " Type: " + sport.getType()

+ " Number of players " + sport.getNoOfPlayers());

}

}

}

**Sport.java**

package practical7a;

public class Sport {

private int id;

private String name;

private String type;

private int noOfPlayers;

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 getType() {

return type;

}

public void setType(String type) {

this.type = type;

}

public int getNoOfPlayers() {

return noOfPlayers;

}

public void setNoOfPlayers(int noOfPlayers) {

this.noOfPlayers = noOfPlayers;

}

}

**beans.xml**

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

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:context="http://www.springframework.org/schema/context"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">

<!-- bean definitions here -->

<!-- Define the DataSource bean for MySQL -->

<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/student" />

<property name="username" value="root" />

<property name="password" value="Root@123" />

</bean>

<!-- Define the JdbcTemplate bean that uses the DataSource -->

<bean id="jdbcTemplate"

class="org.springframework.jdbc.core.JdbcTemplate">

<property name="dataSource" ref="dataSource" />

</bean>

<bean id="sportsDAO" class="practical7a.SportsDAO">

<property name="jdbcTemplate" ref="jdbcTemplate" />

</bean>

</beans>

**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>practical7a</groupId>

<artifactId>practical7a</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.9</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>5.3.9</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>5.3.9</version>

</dependency>

<!-- MySQL Connector -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.26</version>

</dependency>

</dependencies>

</project>

**SQL Query:**

use student;

CREATE TABLE sports (

id INT AUTO\_INCREMENT PRIMARY KEY,

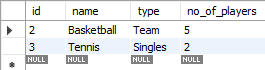
name VARCHAR(255),

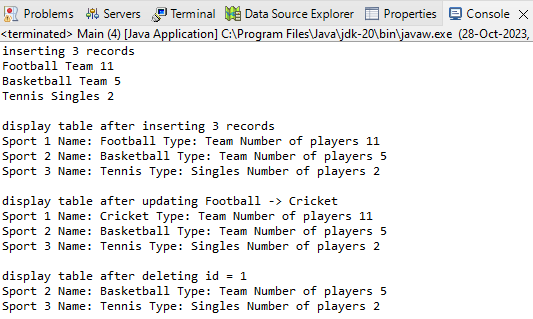
type VARCHAR(255),

no\_of\_players INT

);

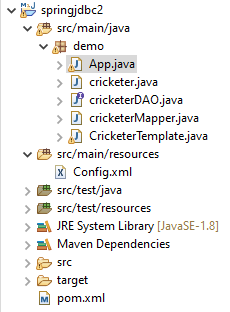
**Output:**





1. Create a table cricketer (name, runs, best score), insert 5 records from the backend. Use spring jdbc concept to display all 5 records. Use Rowmapper Interface.

**File Structure:**

****

**Codes:**

**App.java**

package demo;

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");

CricketerTemplate cricketerTemplate = (CricketerTemplate) context.getBean("cricketerTemplate");

System.out.println("inserting 5 records");

cricketerTemplate.insert("Pushkar", 50, 100);

cricketerTemplate.insert("Prasad", 45, 100);

cricketerTemplate.insert("Anish", 70, 110);

cricketerTemplate.insert("Shreya", 30, 200);

cricketerTemplate.insert("Mrudula", 60, 80);

System.out.println("Listing Records...");

List<cricketer> cricketers = cricketerTemplate.listCricketers();

for (cricketer record : cricketers) {

System.out.print("Name : " + record.getName());

System.out.print(", Runs : " + record.getRuns());

System.out.println(", Best score : " + record.getBestRuns());

}

}

}

**cricketer.java**

package demo;

public class cricketer {

String name;

Integer runs;

Integer bestRuns;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Integer getRuns() {

return runs;

}

public void setRuns(Integer runs) {

this.runs = runs;

}

public Integer getBestRuns() {

return bestRuns;

}

public void setBestRuns(Integer bestRuns) {

this.bestRuns = bestRuns;

}

}

**cricketerDAO.java**

package demo;

import java.util.List;

import javax.sql.DataSource;

public interface cricketerDAO {

public void setDataSource(DataSource ds);

public void insert(String name, Integer runs, Integer bestRuns);

public List<cricketer> listCricketers();

}

**cricketerMapper.java**

package demo;

import java.sql.ResultSet;

import java.sql.SQLException;

import org.springframework.jdbc.core.RowMapper;

public class cricketerMapper implements RowMapper<cricketer> {

@Override

public cricketer mapRow(ResultSet rs, int rowNum) throws SQLException {

cricketer c = new cricketer();

c.setName(rs.getString("name"));

c.setRuns(rs.getInt("runs"));

c.setBestRuns(rs.getInt("bestScore"));

return c;

}

}

**CricketerTemplate.java**

package demo;

import java.util.List;

import javax.sql.DataSource;

import org.springframework.jdbc.core.JdbcTemplate;

public class CricketerTemplate implements cricketerDAO {

private DataSource ds;

private JdbcTemplate jdbcTemplate;

@Override

public void setDataSource(DataSource ds) {

this.ds = ds;

this.jdbcTemplate = new JdbcTemplate(ds);

}

@Override

public void insert(String name, Integer runs, Integer bestRuns) {

String SQL = "INSERT INTO cricketer(name, runs, bestScore) VALUES(?,?,?)";

jdbcTemplate.update(SQL, name, runs, bestRuns);

System.out.println("Created Record Name = " + name + " runs = " + runs + " Best Score = " + bestRuns);

}

@Override

public List<cricketer> listCricketers() {

String SQL = "SELECT \* FROM cricketer";

List<cricketer> cricketers = jdbcTemplate.query(SQL, new cricketerMapper());

return cricketers;

}

}

**config.xml**

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

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-3.0.xsd ">

<bean id="dataSource"

class="org.springframework.jdbc.datasource.DriverManagerDataSource">

<property name="driverClassName"

value="com.mysql.jdbc.Driver" />

<property name="url"

value="jdbc:mysql://localhost:3306/student" />

<property name="username" value="root" />

<property name="password" value="Root@123" />

</bean>

<bean id="cricketerTemplate" class="demo.CricketerTemplate">

<property name="dataSource" ref="dataSource" />

</bean>

</beans>

**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 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>springjdbc</groupId>

<artifactId>springjdbc</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>springjdbc</name>

<url>http://maven.apache.org</url>

<properties>

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

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>5.3.5</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.18</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.5</version>

</dependency>

</dependencies>

</project>

**Output:**

