MYBATIS - READ OPERATION

We discussed in the last chapter, how to insert values into the STUDENT table using MyBatis by performing CREATE operation. This chapter explains how to read the data in a table using MyBatis.

We have the following STUDENT table in MySQL −

```
CREATE TABLE details.student(
   ID int(10) NOT NULL AUTO_INCREMENT,
   NAME varchar(100) NOT NULL,
   BRANCH varchar(255) NOT NULL,
   PERCENTAGE int(3) NOT NULL,
   PHONE int(11) NOT NULL,
   EMAIL varchar(255) NOT NULL,
   PRIMARY KEY (`ID`)
);
```

Assume, this table has two record as −

```
+----+----------+--------+------------+-----------+--------------------+
| ID |  NAME    | BRANCH | PERCENTAGE |   PHONE   |      EMAIL         |
+----+----------+--------+------------+-----------+--------------------+
| 1  | Mohammad |   It   |     80     | 984803322 | Mohammad@gmail.com |
| 2  | shyam    |   It   |     75     | 984800000 | shyam@gmail.com    |
+----+----------+--------+------------+-----------+--------------------+
```

Student POJO Class

To perform read operation, we would modify the Student class in Student.java as −

```
public class Student {
   private int id;
   private String name;
   private String branch;
   private int percentage;
   private int phone;
   private String email;

   public Student(int id, String name, String branch, int percentage, int phone, String
email) {
      super();
      this.id = id;
      this.name = name;
      this.branch = branch;
      this.percentage = percentage;
      this.phone = phone;
      this.email = email;
   }

   public Student() {}

   public int getId() {
      return id;
   }

   public String getName() {
      return name;
   }

   public int getPhone() {
      return phone;
   }
```

```
   public String getEmail() {
      return email;
   }

   public String getBranch() {
      return branch;
   }

   public int getPercentage() {
      return percentage;
   }

}
```

Student.xml File

To define SQL mapping statement using MyBatis, we would add **<select>** tag in Student.xml file and inside this tag definition, we would define an **"id"** which will be used in mybatisRead.java file for executing SQL SELECT query on database. While reading the records, we can get all the records at once or we can get a particular record using the where clause. In the XML given below, you can observe both the queries.

To retrieve a particular record, we need a unique key to represent that record. Therefore, we have also defined the resultmap "id" *uniquekey* of type Student to map the result of the select query with the variable of Student class.

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

<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace = "Student">

   <resultMap id = "result" type = "Student">
      <result property = "id" column = "ID"/>
   </resultMap>

   <select id = "getAll" resultMap = "result">
      SELECT * FROM STUDENT;
   </select>

   <select id = "getById" parameterType = "int" resultMap = "result">
      SELECT * FROM STUDENT WHERE ID = #{id};
   </select>

</mapper>
```

mybatisRead_ALL.java File

This file has application level logic to read all the records from the Student table. Create and save **mybatisRead_ALL.java** file as shown below −

```
import java.io.IOException;
import java.io.Reader;
import java.util.List;

import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;

public class mybatisRead_ALL {

   public static void main(String args[]) throws IOException{

      Reader reader = Resources.getResourceAsReader("SqlMapConfig.xml");
      SqlSessionFactory sqlSessionFactory = new SqlSessionFactoryBuilder().build(reader);
```

```
      SqlSession session = sqlSessionFactory.openSession();

      //select contact all contacts
      List<Student> student = session.selectList("Student.getAll");

      for(Student st : student ){
         System.out.println(st.getId());
         System.out.println(st.getName());
         System.out.println(st.getBranch());
         System.out.println(st.getPercentage());
         System.out.println(st.getEmail());
         System.out.println(st.getPhone());
      }

      System.out.println("Records Read Successfully ");
      session.commit();
      session.close();
   }
}
```

Compilation and Execution

Here are the steps to compile and run the mybatisRead_ALL file. Make sure, you have set PATH and CLASSPATH appropriately before proceeding for compilation and execution.

- Create Student.xml as shown above.
- Create Student.java as shown above and compile it.
- Create mybatisRead_ALL.java as shown above and compile it.
- Execute mybatisRead_ALL binary to run the program.

You would get all the record of the student table as −

```
++++++++++++++ details of the student who's id is  :1 ++++++++++++++++++
1
Mohammad
It
80
Mohammad@gmail.com
984803322
++++++++++++++ details of the student who's id is  :2
++++++++++++++++++
2
shyam
It
75
shyam@gmail.com
984800000
Records Read Successfully
```

Reading a Particular Record

Copy and save the following program with the name **mybatisRead_byID** −

```
import java.io.IOException;
import java.io.Reader;

import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;

public class mybatisRead_byID {

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

```
      int i = 1;
      Reader reader = Resources.getResourceAsReader("SqlMapConfig.xml");
      SqlSessionFactory sqlSessionFactory = new SqlSessionFactoryBuilder().build(reader);
      SqlSession session = sqlSessionFactory.openSession();

      //select a particular student  by  id
      Student student = (Student) session.selectOne("Student.getById", 2);

      //Print the student details
      System.out.println(student.getId());
      System.out.println(student.getName());
      System.out.println(student.getBranch());
      System.out.println(student.getPercentage());
      System.out.println(student.getEmail());
      System.out.println(student.getPhone());

      session.commit();
      session.close();

   }

}
```

Compilation and Execution

Here are the steps to compile and run the mybatisRead_byID file. Make sure, you have set PATH and CLASSPATH appropriately before proceeding for compilation and execution.

- Create Student.xml as shown above.
- Create SqlMapConfig.xml as shown in the MYBATIS - Configuration XML chapter of this tutorial.
- Create Student.java as shown above and compile it.
- Create mybatisRead_byID.java as shown above and compile it.
- Execute mybatisRead_byID binary to run the program.

You would get the following result, and a record would be read from the Student table as −

```
2
shyam
It
75
shyam@gmail.com
984800000
```