MYBATIS - CREATE OPERATION

http://www.tutorialspoint.com/mybatis/mybatis_create_operation.htm Copyright © tutorialspoint.com

To perform any Create, Read, Update, and Delete *CRUD* operation using MyBATIS, you would need to create a Plain Old Java Objects *POJO* class corresponding to the table. This class describes the objects that will "model" database table rows.

The POJO class would have implementation for all the methods required to perform desired operations.

Create the STUDENT table in MySQL database as shown below −

```
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`)
   ->
);
Query OK, 0 rows affected (0.37 sec)
```

Student POJO Class

Create a STUDENT class in STUDENT.java file as

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

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

}
```

You can define methods to set individual fields in the table. The next chapter explains how to get the values of individual fields.

Student.xml File

To define SQL mapping statement using MyBatis, we would use **<insert>** tag. Inside this tag definition, we would define an **"id."** Further, the ‘id’ will be used in the mybatisInsert.java file for executing SQL INSERT query on database. Create student.xml file as shown below −

```
<?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">

   <insert id = "insert" parameterType = "Student">
```

```
      INSERT INTO STUDENT (NAME, BRANCH, PERCENTAGE, PHONE, EMAIL ) VALUES (#{name},
#{branch}, #{percentage}, #{phone}, #{email});

      <selectKey keyProperty = "id" resultType = "int" order = "AFTER">
         select last_insert_id() as id
      </selectKey>

   </insert>

</mapper>
```

Here, **parameteType** − could take a value as *string, int, float, double*, or any class *object* based on requirement. In this example, we would pass Student object as a parameter, while calling *insert* method of **SqlSession** class.

If your database table uses an IDENTITY, AUTO_INCREMENT, or SERIAL column, or you have defined a SEQUENCE/GENERATOR, you can use the **<selectKey>** element in an **<insert>** statement to use or return that database-generated value.

mybatisInsert.java File

This file would have application level logic to insert records in the Student table. Create and save **mybatisInsert.java** file as shown below −

```
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 mybatisInsert {

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

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

      //Create a new student object
      Student student = new Student("Mohammad","It", 80, 984803322, "Mohammad@gmail.com"
);

      //Insert student data
      session.insert("Student.insert", student);
      System.out.println("record inserted successfully");
      session.commit();
      session.close();

   }

}
```

Compilation and Execution

Here are the steps to compile and run the mybatisInsert.java 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 mybatisInsert.java as shown above and compile it.
- Execute mybatisInsert binary to run the program.

You would get the following result, and a record would be created in the STUDENT table.

```
$java mybatisInsert
Record Inserted Successfully
```

If you check the STUDENT table, it should display the following result −

```
mysql> select * from student;
+----+----------+--------+------------+-----------+--------------------+
| ID |   NAME   | BRANCH | PERCENTAGE |   PHONE   |       EMAIL        |
+----+----------+--------+------------+-----------+--------------------+
|  1 | Mohammad |   It   |     80     | 984803322 | Mohammad@gmail.com |
+----+----------+--------+------------+-----------+--------------------+
1 row in set (0.00 sec)
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js