

Inserting Data Into Table Using JDBC PreparedStatement

?

In this tutorial, you will learn how to use PreparedStatement object to insert data into MySQL table.

In the previous tutorial, we have shown you [how to use the PreparedStatement object to update data](#). When you call the `executeUpdate()` method, you get the number of rows affected. When you insert a record into a table, you may want to get the inserted ID back to the program for further processing. Let's see how we can do it.

First, as always, you [open a new connection to MySQL](#). You can utilize the utility class `MySQLJDBCUtil` that we developed in [the previous tutorial](#).

```
1 Connection conn = MySQLJDBCUtil.getConnection();
```

Then, you construct an INSERT statement with placeholders and create a new PreparedStatement object by calling the `prepareStatement()` method of the Connection object. You pass the [INSERT statement](#) as the first argument and an integer with value `Statement.RETURN_GENERATED_KEYS` as the second argument to the method. The second argument instructs JDBC to give the inserted ID back.

```
1 String sql = "INSERT INTO candidates(first_name, la
2 st_name, dob, phone, email) "
3         + "VALUES (?, ?, ?, ?, ?) ";
4
5
```

```
PreparedStatement pstmt = conn.prepareStatement(sq  
1,  
  
Statement.RETURN_GEN  
ERATED_KEYS);
```

Next, you supply values for placeholders by calling `setYYY()` method of the `PreparedStatement` object.

```
1 // set parameters for statement  
2 pstmt.setString(1, firstName);  
3 pstmt.setString(2, lastName);  
4 pstmt.setDate(3, dob);  
5 pstmt.setString(4, phone);  
6 pstmt.setString(5, email);
```

After that, you call the `executeUpdate()` method to execute the `INSERT` statement. This method returns the number of rows affected. We check the return value to see if the record has been inserted successfully.

```
1 int rowAffected = pstmt.executeUpdate();  
2 if(rowAffected == 1)  
3 {  
4     // process further here  
5 }
```

Finally, to get the inserted id, you call the `getGeneratedKeys()` method of the `PreparedStatement` object. The method returns a `ResultSet`. You just need to get data out of this `ResultSet` as follows:

```
1 // get candidate id  
2 int candidateId = 0;  
3 ResultSet rs = pstmt.getGeneratedKeys();  
4 if(rs.next())  
5     candidateId = rs.getInt(1);
```

The following is the complete example of inserting data into the `candidates` table and get the inserted ID back.

```
1 package org.mysqltutorial;
```

```

2
3 import java.sql.Connection;
4 import java.sql.Date;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import java.sql.SQLException;
8 import java.sql.Statement;
9
10 /**
11  *
12  * @author mysqltutorial.org
13  */
14 public class Main {
15
16     /**
17      * Insert a new candidate
18      * @param firstName
19      * @param lastName
20      * @param dob
21      * @param email
22      * @param phone
23      * @return
24      */
25     public static int insertCandidate(String firstName, String lastName, Date dob,
26                                     String email, String phone) {
27         // for insert a new candidate
28         ResultSet rs = null;
29         int candidateId = 0;
30
31         String sql = "INSERT INTO candidates(firstName, last_name, dob, phone, email) "
32             + "VALUES(?, ?, ?, ?, ?)";
33
34         try (Connection conn = MySQLJDBCUtil.getConnection();
35             PreparedStatement pstmt = conn.prepareStatement(sql, Statement.RETURN_GENERATED_KEYS);)

```

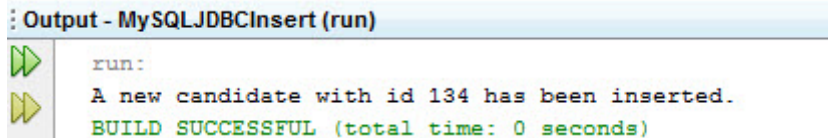
```

38 {
39
40     // set parameters for statement
41     pstmt.setString(1, firstName);
42     pstmt.setString(2, lastName);
43     pstmt.setDate(3, dob);
44     pstmt.setString(4, phone);
45     pstmt.setString(5, email);
46
47     int rowAffected = pstmt.executeUpdate(
48 ;
49
50     if(rowAffected == 1)
51     {
52         // get candidate id
53         rs = pstmt.getGeneratedKeys();
54         if(rs.next())
55             candidateId = rs.getInt(1);
56     }
57 } catch (SQLException ex) {
58     System.out.println(ex.getMessage());
59 } finally {
60     try {
61         if(rs != null) rs.close();
62     } catch (SQLException e) {
63         System.out.println(e.getMessage());
64     }
65 }
66
67     return candidateId;
68 }
69 /**
70  * @param args the command line arguments
71  */
72 public static void main(String[] args) {
73     // insert a new candidate
74     int id = insertCandidate("Bush", "Lily",
Date.valueOf("1980-01-04"),

```

```
75         "bush.l@yahoo.com", "(40  
8) 898-6666");  
  
        System.out.println(String.format("A new  
candidate with id %d has been inserted.",id));  
    }  
}
```

Let's run the program.



```
: Output - MySQLJDBCInsert (run)  
run:  
A new candidate with id 134 has been inserted.  
BUILD SUCCESSFUL (total time: 0 seconds)
```

It shows that you have successfully inserted a new candidate into the candidates table with id 134.

In this tutorial, we have shown you how to use PreparedStatement object to insert a new record into a MySQL table and get the inserted ID back for further processing.