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 utilized the utility class `MySQLJDBCUtil` that we developed inthe previous tutorial.

```
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 the second argument to the method. The second argument instructs JDBC to give the inserted ID back.

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


```

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

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

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

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

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:

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

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

```
package org.mysqltutorial;
```

```
import java.sql.Connection;
import java.sql.Date;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

/**
 *
 * @author mysqltutorial.org
 */
public class Main {

    /**
     * Insert a new candidate
     * @param firstName
     * @param lastName
     * @param dob
     * @param email
     * @param phone
     * @return
     */
    public static int insertCandidate(String firstName,String lastName,Date dob,
                                      String email, String phone) {
        // for insert a new candidate
        ResultSet rs = null;
        int candidateId = 0;

        String sql = "INSERT INTO candidates(first_name,last_name,dob,phone,email) "
                   + "VALUES(?,?,?,?,?)";

        try (Connection conn = MySQLJDBCUtil.getConnection();
             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              if(rowAffected == 1)
50              {
51                  // get candidate id
52                  rs = pstmt.getGeneratedKeys();
53                  if(rs.next())
54                      candidateId = rs.getInt(1);
55
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      public static void main(String[] args) {
72           // insert a new candidate
73           int id = insertCandidate("Bush", "Lily",
    Date.valueOf("1980-01-04"),
74
```

```
                        "bush.l@yahoo.com", "(408) 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.