Calling MySQL Stored Procedures from JDBC

In this tutorial, you will learn how to call MySQL stored procedures from JDBC using CallableStatement object.

Before you start

For the sake of demonstration, we will create a new stored procedure named get_candidate_skill that accepts candidate_id as the IN parameter and returns a result set that contains the skills of the candidate.

DELIMITER $$

CREATE PROCEDURE get_candidate_skill(IN candidate_id INT)

BEGIN

    SELECT candidates.id, first_name,last_name, skills.name AS skill

    FROM candidates

    INNER JOIN candidate_skills ON candidates.id = candidate_skills.candidate_id

    INNER JOIN skills ON skills.id = candidate_skills.skill_id

    WHERE candidates.id = candidate_id;

  END$$
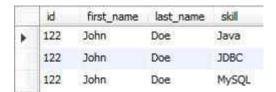
DELIMITER ;

Code language: SQL (Structured Query Language) (sql)

Let's call this stored procedure for candidate id with value 122.

CALL get_candidate_skill(122);

Code language: SQL (Structured Query Language) (sql)

id	first_name	last_name	skill
122	John	Doe	Java
122	John	Doe	JDBC
122	John	Doe	MySQL

Introducing to CallableStatement and stored procedure call syntax

To call stored procedures or stored functions in MySQL from JDBC, you use CallableStatement object, which inherits from PreparedStatement object. The general syntax of calling a stored procedure is as follows:

```sql
{?= call procedure_name(param1,param2,...)}
```

Code language: SQL (Structured Query Language) (sql)

You wrap the stored procedure call within braces ({}). If the stored procedure returns a value, you need to add the question mark and equal (?=) before the call keyword. If a stored procedure does not return any values, you just omit the ?= sign. In case the stored procedure accepts any parameters, you list them within the opening and closing parentheses after the stored procedure's name.

The following are examples of using the syntax for calling stored procedures in different contexts:

Syntax	Stores Procedures
{ call procedure_name() }	Accept no parameters and return no value
{ call procedure_name(?,?) }	Accept two parameters and return no value
{?= call procedure_name() }	Accept no parameter and return value
{?= call procedure_name(?) }	Accept one parameter and return value

Notice that question mark placeholder (?) can be used for both IN ,OUT, and INOUT parameters. For detailed information on different parameter types in stored procedures, check it out [MySQL stored procedure parameters tutorial](#).

JDBC MySQL stored procedure example

First, open a connection to MySQL server by creating a new Connection object.

```java
Connection conn = DriverManager.getConnection();
```

Code language: Java (java)

Then, prepare a stored procedure call and create a CallableStatement object by calling prepareCall() method of the Connection object.

```java
String query = "{CALL get_candidate_skill(?)}";

CallableStatement stmt = conn.prepareCall(query)
```

Code language: Java (java)

Next, pass all the parameters to the stored procedure. In this case, the get_candidate_skill stored procedure accepts only one IN parameter.

```java
stmt.setInt(1, candidateId);
```

Code language: Java (java)

After that, execute the stored procedure by calling the executeQuery() method of the CallableStatement object. It returns a result set in this case.

ResultSet rs = stmt.executeQuery();

Code language: Java (java)

Finally, traverse the ResultSet to display the results.

```java
while (rs.next()) {

  System.out.println(String.format("%s - %s",

              rs.getString("first_name") + " "

              + rs.getString("last_name"),

              rs.getString("skill")));

}
```

Code language: Java (java)

The following is the complete example of calling the MySQL stored procedure from JDBC.

```java
package org.mysqltutorial;


import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.CallableStatement;


/**

 *

 * @author mysqltutorial.org

 */

public class Main {
```

```java
/**
 * Get skills by candidate id
 *
 * @param candidateId
 */
public static void getSkills(int candidateId) {
    //
    String query = "{ call get_candidate_skill(?) }";
    ResultSet rs;

    try (Connection conn = MySQLJDBCUtil.getConnection();
         CallableStatement stmt = conn.prepareCall(query)) {

        stmt.setInt(1, candidateId);

        rs = stmt.executeQuery();
        while (rs.next()) {
            System.out.println(String.format("%s - %s",
                    rs.getString("first_name") + " "
                    + rs.getString("last_name"),
                    rs.getString("skill")));
        }
    } catch (SQLException ex) {
        System.out.println(ex.getMessage());
    }
```

```java
    }



    /**
     *
     * @param args
     */
    public static void main(String[] args) {

        getSkills(122);

    }

}
```

Code language: Java (java)

Let's run the program.

The program works as expected.

In this tutorial, we have shown you how to call a stored procedure in MySQL database from a Java program using JDBC CallableStatement object.