Querying Data From MySQL Using JDBC

?

In this tutorial, we will show you how to query data from MySQL using JDBC Statement and ResultSet objects.

To query data from MySQL, you first need to establish a connection to MySQL using Connection object.

```
Connection conn =
DriverManager.getConnection(url,username,password);
```

We developed a utility class named MySQLJDBCUtil that open a new connection with database parameters stored in a properties file.

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

After opening the connection, you need to create a Statement object. JDBC provides several kinds of statements such as Statement, PreparedStatement and CallableStatement. For querying data, you just need to use the Statement object as follows:

```
Statement stmt  = conn.createStatement();
```

Once you have a Statement object created, you can use it to execute any valid MySQL query like the following:

```
String sql = "SELECT first_name, last_name, email
 " +
```

```
3               "FROM candidates";
4
ResultSet rs    = stmt.executeQuery(sql)
```

We have called the `executeQuery()` method of the Statement object. This method returns a ResultSet object that contains result of the SQL query. The result is in the form of rows with columns of data based on the SELECT statement.

The ResultSet object provides you with methods to traverse the result and read the data.
The `next()` method returns true and move to the next row in the ResultSet if there are rows available, otherwise it returns false. You must call the `next()` method at least one before reading data because before the first `next()`call, the ResultSet is located before the first row.

To get column data of the current row, you use `getDataType()`method where DataType is the data type of the column e.g., int, string, double, etc., You need to pass the column name or column index to the getDataType() method, for example:

```
String s = rs.getString("column_name");
int id   = rs.getInt(1);
```

To get the data out of the candidates ResultSet, you do it as follows:

```
while (rs.next()) {
   System.out.println(rs.getString("first_name") +
 "\t" +
                      rs.getString("last_name")  +
 "\t" +
                      rs.getString("email"));

}
```

You should always close the ResultSet and Statement objects when you complete traversing the data by calling `close()`method.

```
try{
   rs.close();
   stmt.close();
} catch(SQLException e) {
```

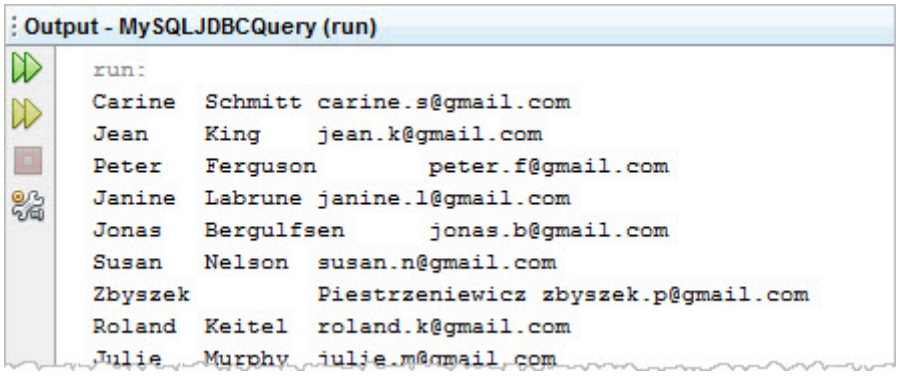
```
    System.out.println(e.getMessage());
}
```

If you use the try-with-resource statement, the `close()`method is automatically called so you don't have to explicitly do it. The following is the complete example of querying data from the `candidates` table in our sample database.

```
package org.mysqltutorial;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

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

    public static void main(String[] args) {
        //
        String sql = "SELECT first_name, last_nam
e, email " +
                     "FROM candidates";

        try (Connection conn = MySQLJDBCUtil.getC
onnection();
             Statement stmt  = conn.createStateme
nt();
             ResultSet rs    = stmt.executeQuery(s
) {

            // loop through the result set
            while (rs.next()) {
                System.out.println(rs.getString("f
st_name") + "\t" +

rs.getString("last_name")  + "\t" +
```

```
33
34  rs.getString("email"));

                }
            } catch (SQLException ex) {
                System.out.println(ex.getMessage());
            }
        }
    }
```

The output of the program is as follows:



In this tutorial, we have shown you how to query data from MySQL using JDBC with simple SQL statement.