

JDBC - CREATE TABLES EXAMPLE

<http://www.tutorialspoint.com/jdbc/jdbc-create-tables.htm>

Copyright © tutorialspoint.com

This chapter provides an example on how to create a table using JDBC application. Before executing the following example, make sure you have the following in place –

- To execute the following example you can replace the *username* and *password* with your actual user name and password.
- Your MySQL or whatever database you are using is up and running.

Required Steps

The following steps are required to create a new Database using JDBC application –

- **Import the packages:** Requires that you include the packages containing the JDBC classes needed for database programming. Most often, using *import java.sql.** will suffice.
- **Register the JDBC driver:** Requires that you initialize a driver so you can open a communications channel with the database.
- **Open a connection:** Requires using the *DriverManager.getConnection* method to create a Connection object, which represents a physical connection with a database server.
- **Execute a query:** Requires using an object of type Statement for building and submitting an SQL statement to create a table in a selected database.
- **Clean up the environment:** Requires explicitly closing all database resources versus relying on the JVM's garbage collection.

Sample Code

Copy and paste the following example in JDBCExample.java, compile and run as follows –

```
//STEP 1. Import required packages
import java.sql.*;

public class JDBCExample {
    // JDBC driver name and database URL
    static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
    static final String DB_URL = "jdbc:mysql://localhost/STUDENTS";

    // Database credentials
    static final String USER = "username";
    static final String PASS = "password";

    public static void main(String[] args) {
        Connection conn = null;
        Statement stmt = null;
        try{
            //STEP 2: Register JDBC driver
            Class.forName("com.mysql.jdbc.Driver");

            //STEP 3: Open a connection
            System.out.println("Connecting to a selected database...");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            System.out.println("Connected database successfully...");

            //STEP 4: Execute a query
            System.out.println("Creating table in given database...");
            stmt = conn.createStatement();

            String sql = "CREATE TABLE REGISTRATION " +
                "(id INTEGER not NULL, " +
                " first VARCHAR(255), " +
```

```

        " last VARCHAR(255), " +
        " age INTEGER, " +
        " PRIMARY KEY ( id )");

    stmt.executeUpdate(sql);
    System.out.println("Created table in given database...");
} catch (SQLException se) {
    //Handle errors for JDBC
    se.printStackTrace();
} catch (Exception e) {
    //Handle errors for Class.forName
    e.printStackTrace();
} finally {
    //finally block used to close resources
    try {
        if (stmt != null)
            conn.close();
    } catch (SQLException se) {
    } // do nothing
    try {
        if (conn != null)
            conn.close();
    } catch (SQLException se) {
        se.printStackTrace();
    } //end finally try
} //end try
System.out.println("Goodbye!");
} //end main
} //end JDBCExample

```

Now, let us compile the above example as follows –

```

C:\>javac JDBCExample.java
C:\>

```

When you run **JDBCExample**, it produces the following result –

```

C:\>java JDBCExample
Connecting to a selected database...
Connected database successfully...
Creating table in given database...
Created table in given database...
Goodbye!
C:\>

```

Loading [Mathjax]/jax/output/HTML-CSS/fonts/TeX/fontdata.js