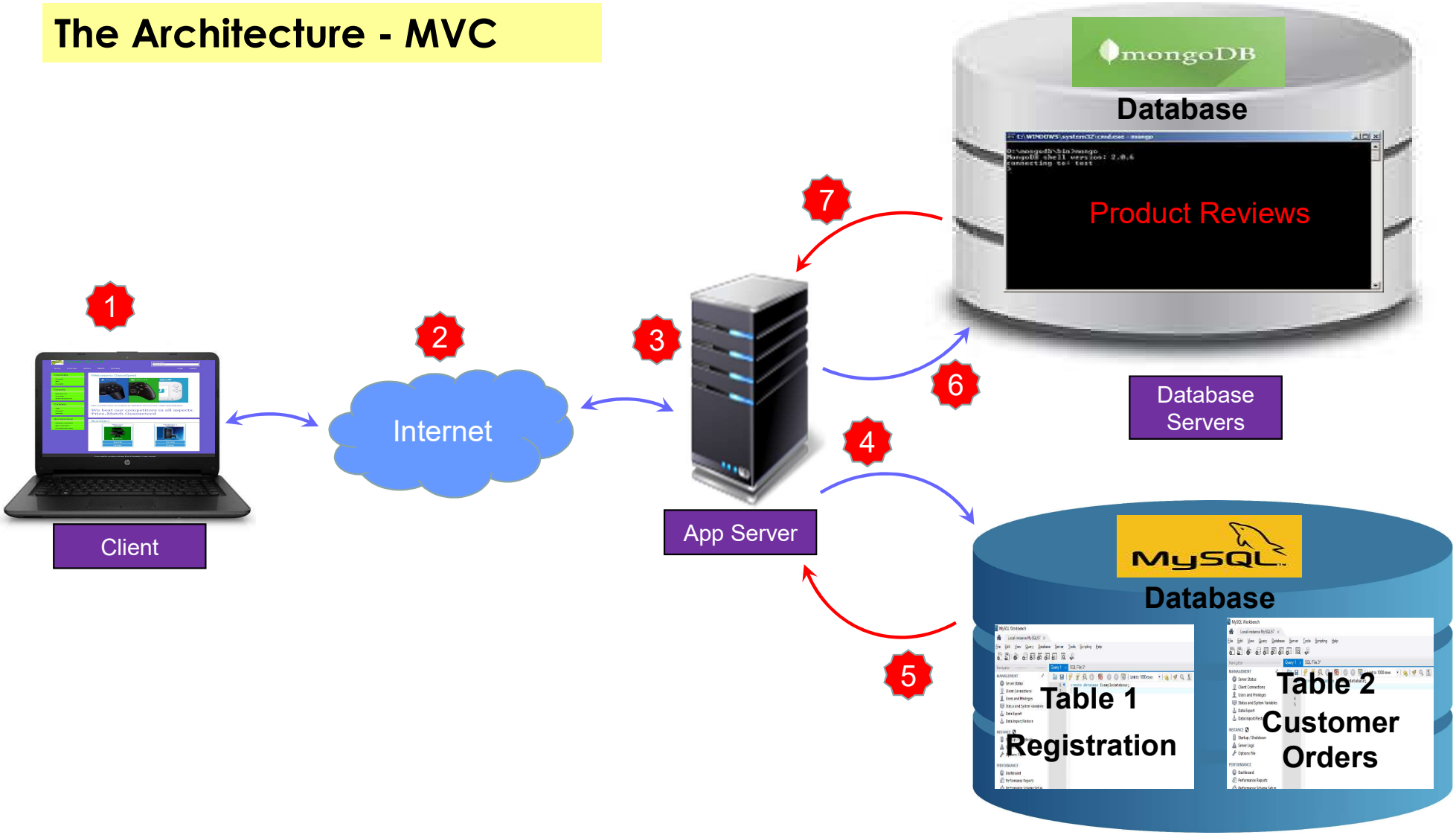
Tutorial #2

MySQL

Datastore

The Architecture - MVC

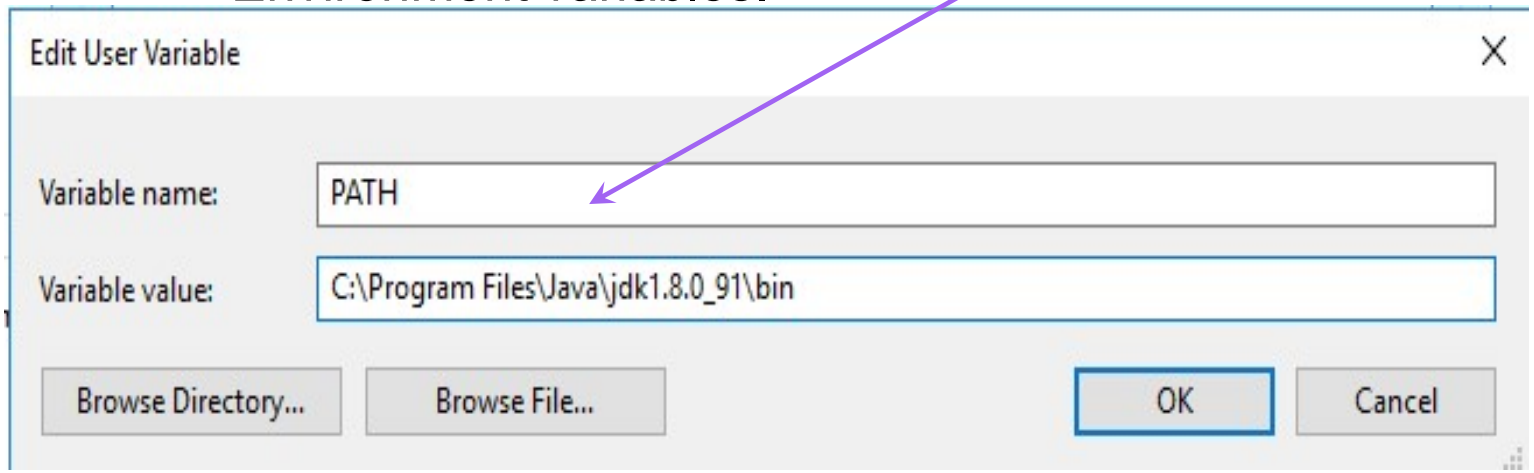


Pre-Requisites:

- Install Java latest version into your system.
- Set the PATH system variable in your local system under

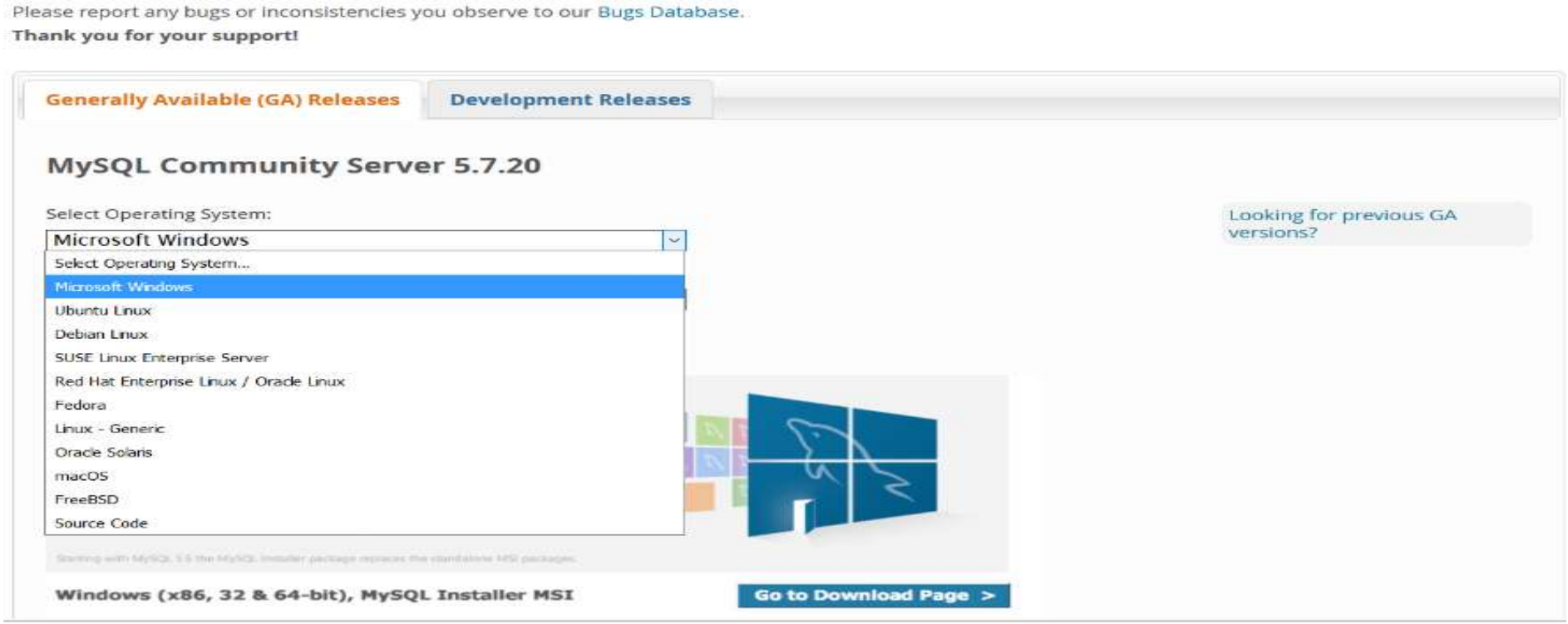
Control Panel –> system -> Advanced system settings ->

Environment variables.



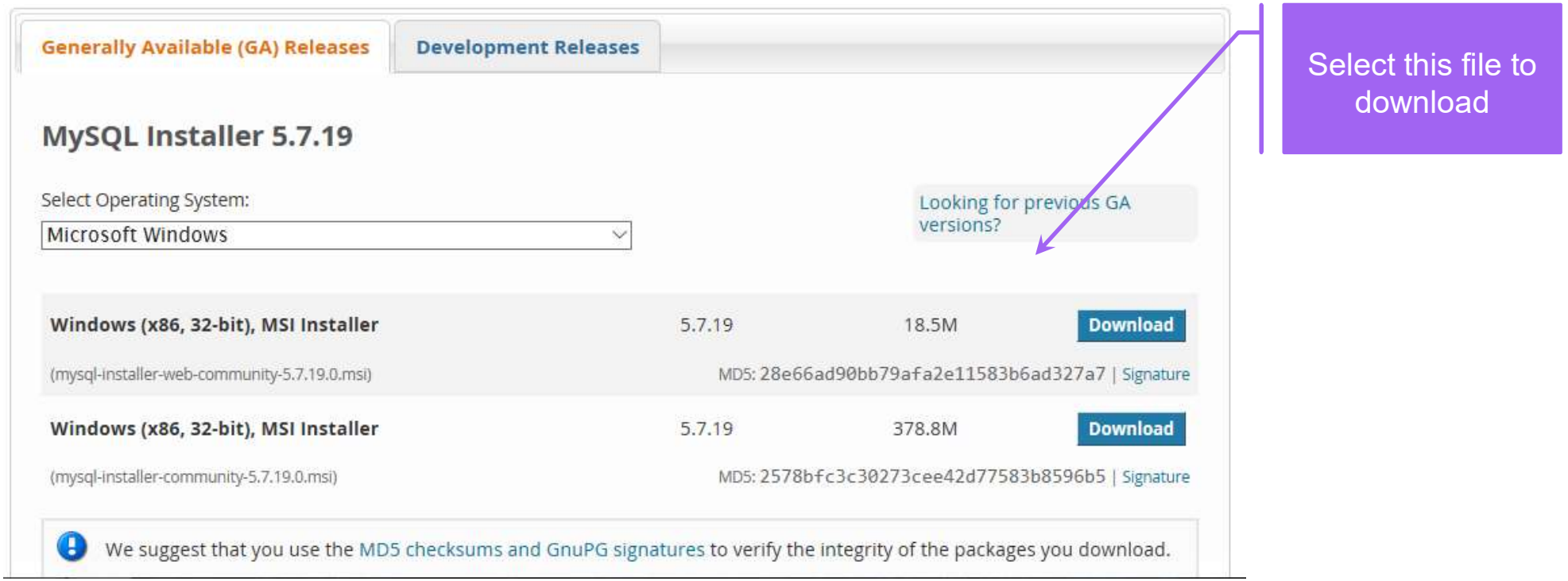
Example PATH variable

Platforms supported in MySQL



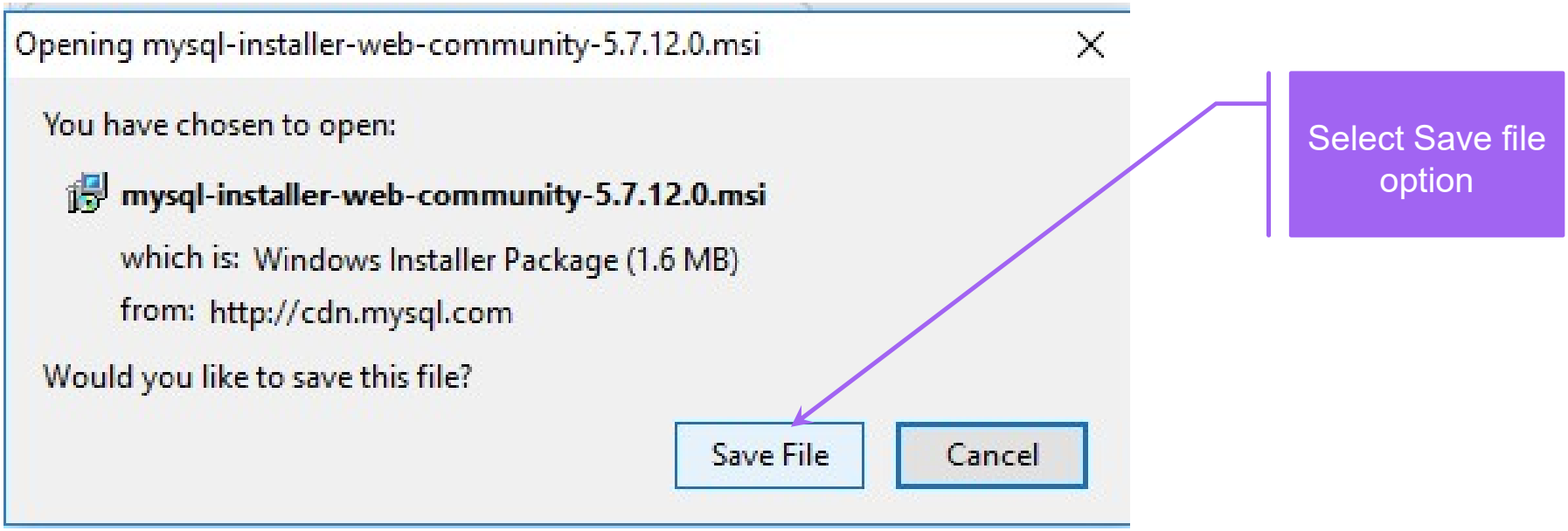
Download and Install MySQL Server

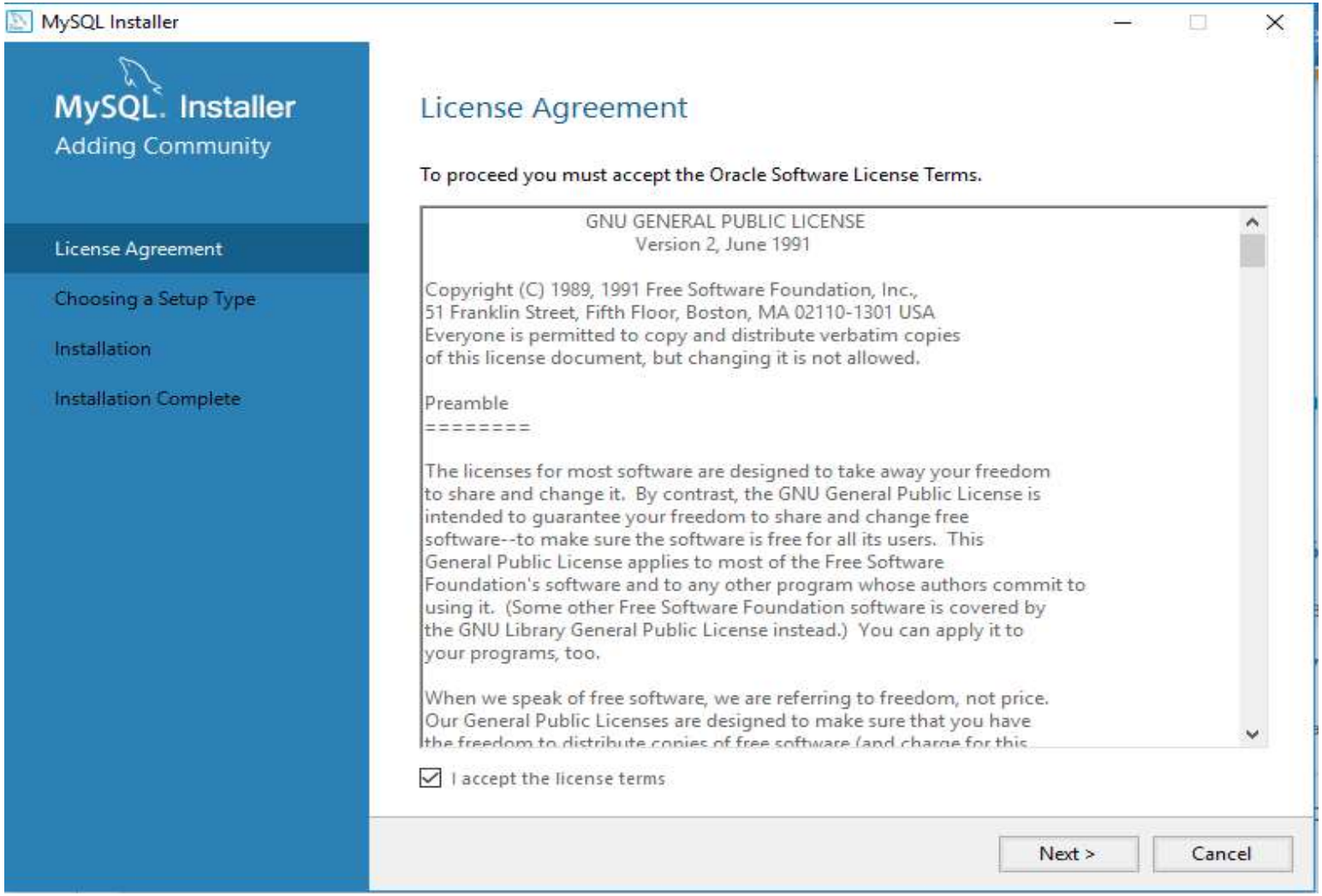
- Download the MySQL installer from http://dev.mysql.com/downloads/installer/ (Choose the my-installer-web-community file)

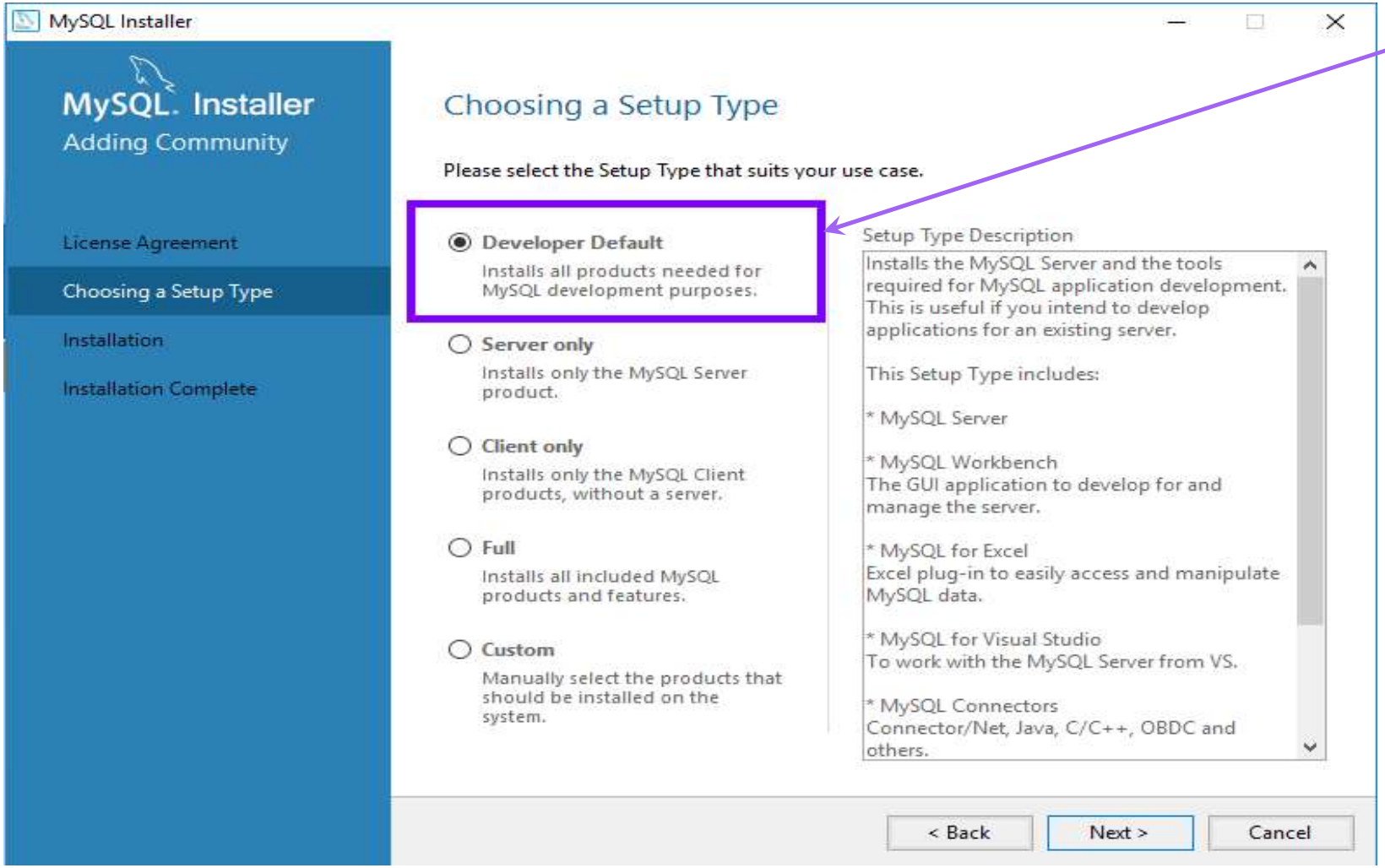


Open file and follow the steps :

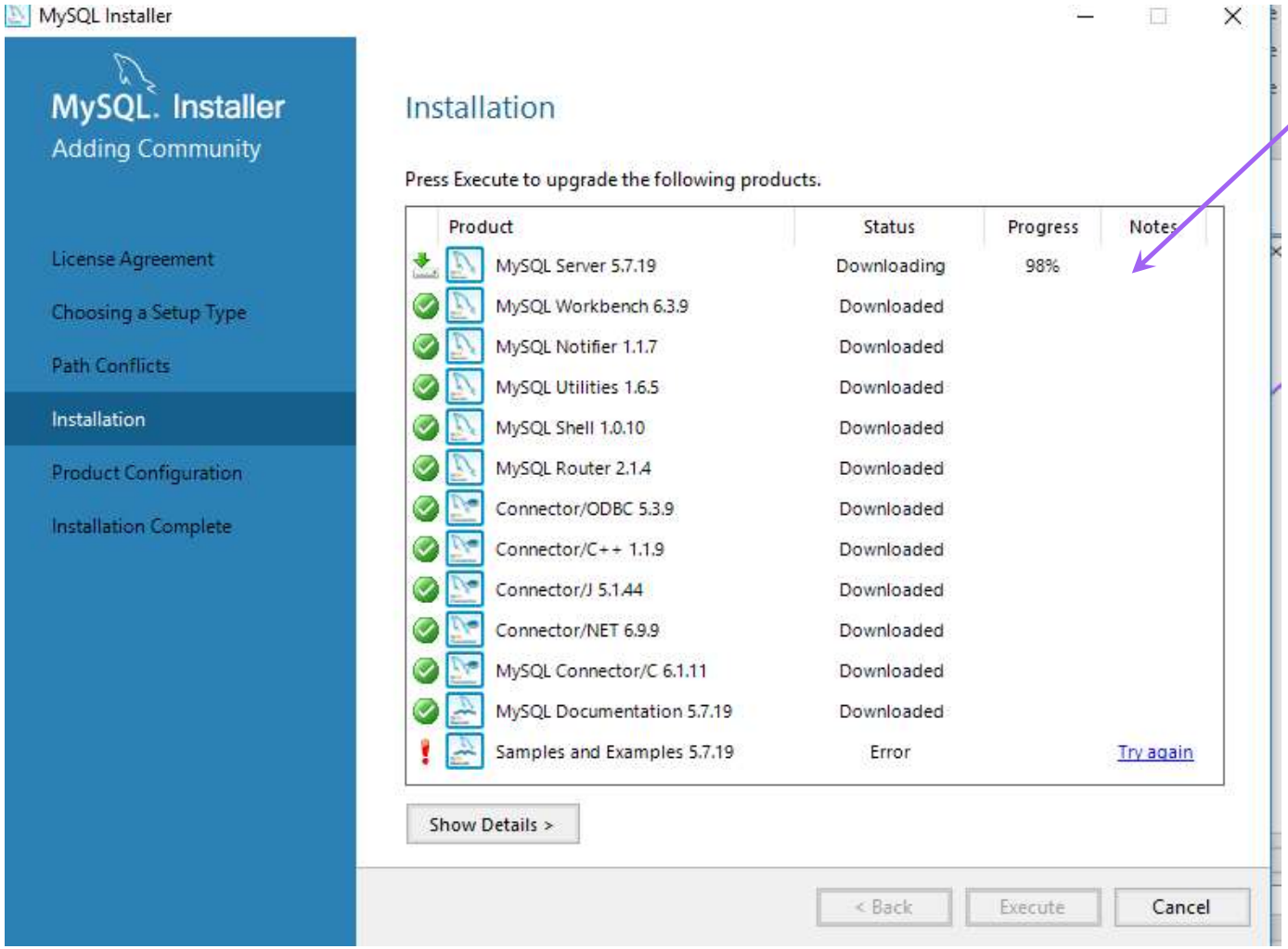
Step 1: Save file to download the installer.



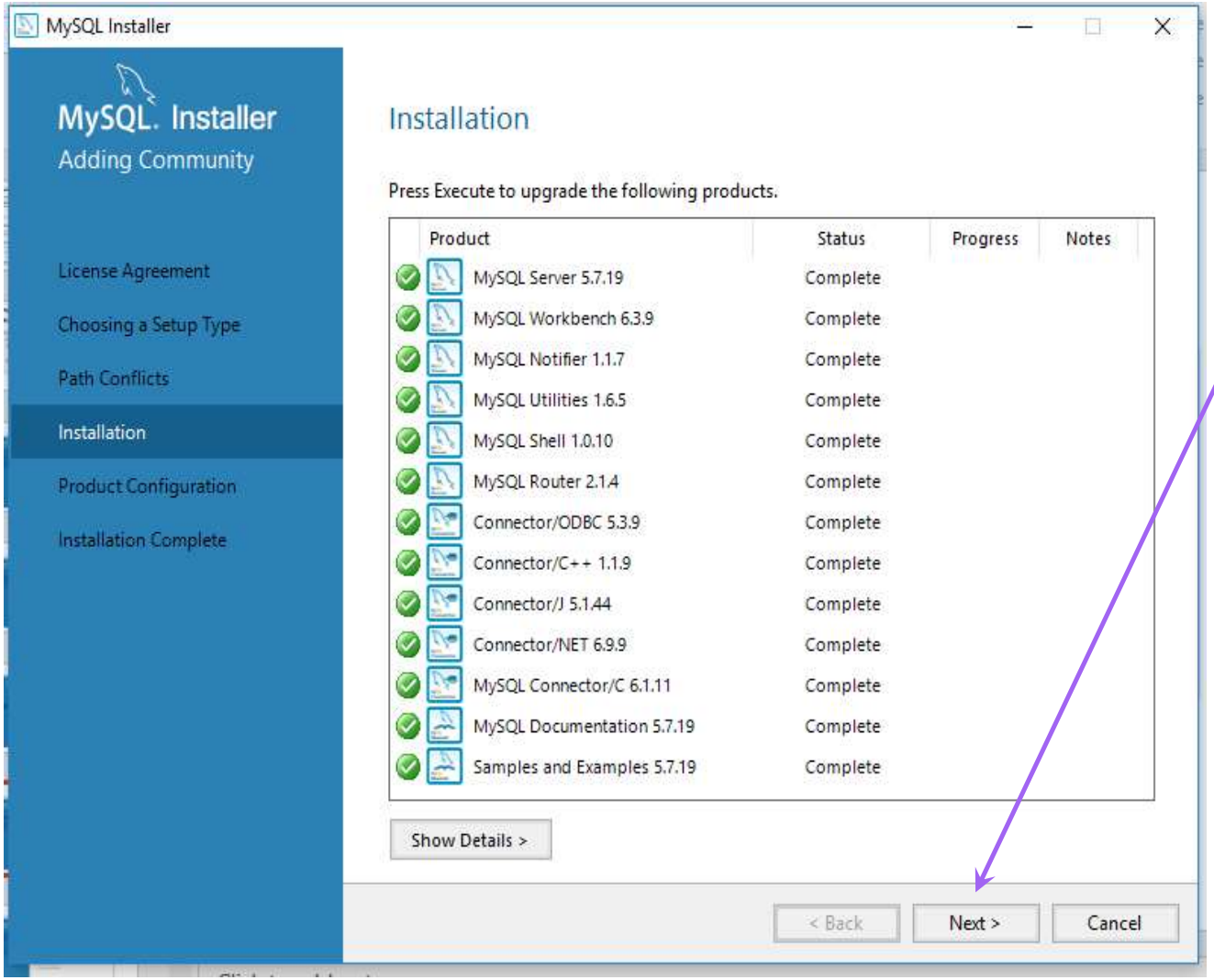

MySQL Installer
MySQL. Installer
Adding Community
License Agreement
Choosing a Setup Type
Installation
Installation Complete
License Agreement
To proceed you must accept the Oracle Software License Terms.
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The licenses for most software are designed to take away your freedom
to share and change it. By contrast, the GNU General Public License is
intended to guarantee your freedom to share and change free
software--to make sure the software is free for all its users. This
General Public License applies to most of the Free Software
Foundation's software and to any other program whose authors commit to
using it. (Some other Free Software Foundation software is covered by
the GNU Library General Public License instead.) You can apply it to
your programs, too.
When we speak of free software, we are referring to freedom, not price.
Our General Public Licenses are designed to make sure that you have
I accept the license terms
Next >
Cancel



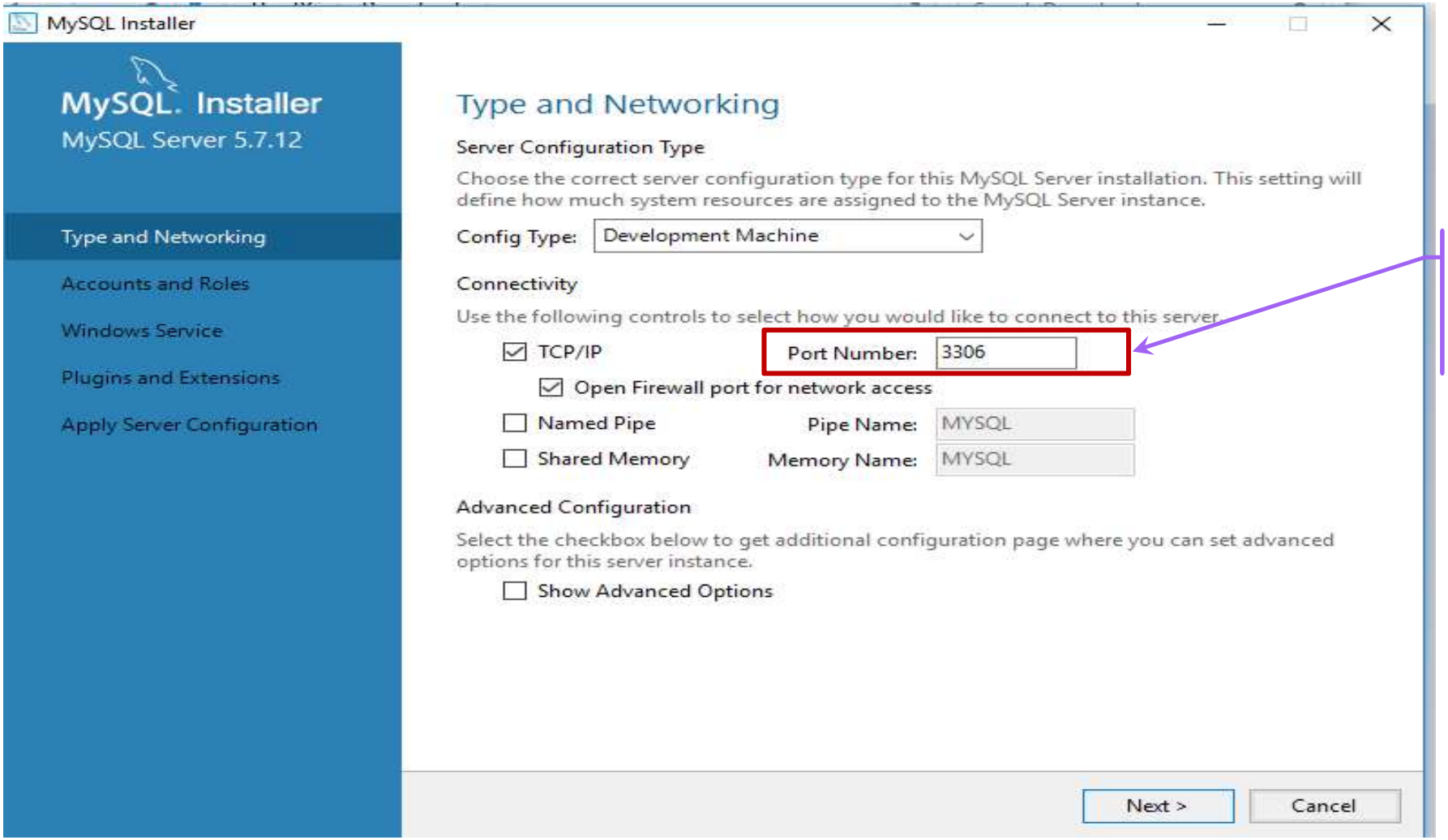
Select the setup type as " Developer Default"



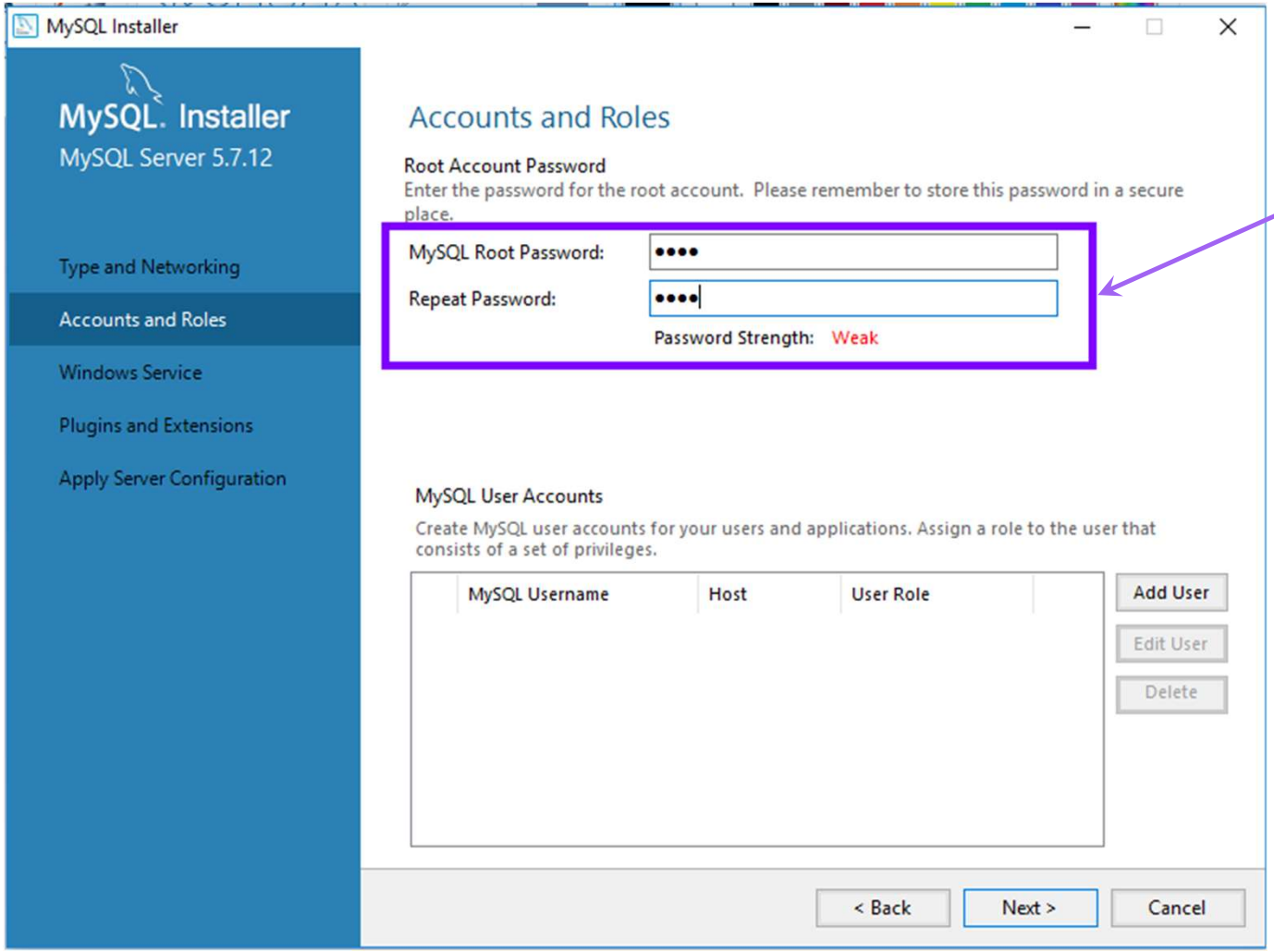
If you get a “Try Again “ option here instead, then click on the try again option to download the products.



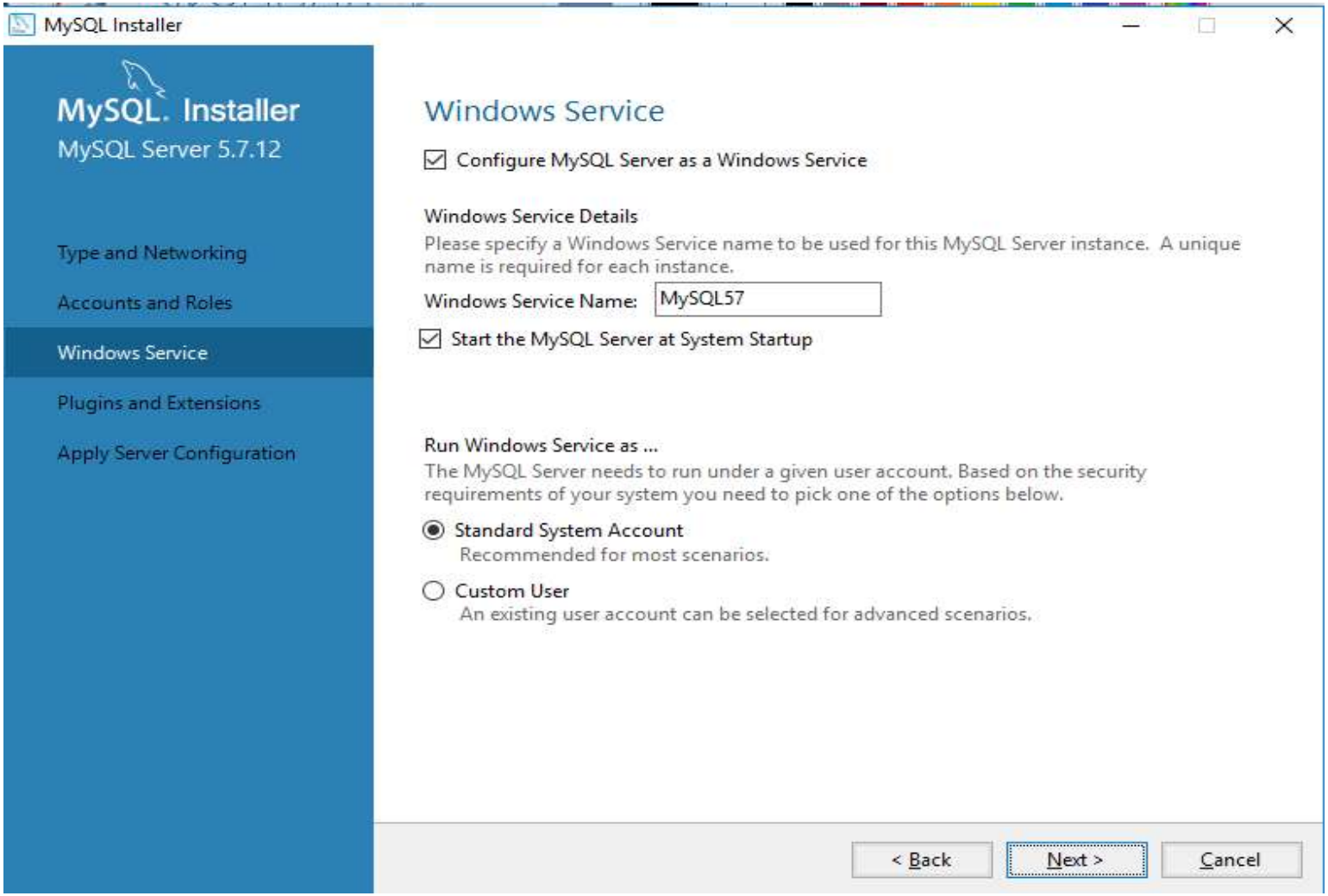
Once all the products are installed and the status is shown as “complete” proceed with the next option .

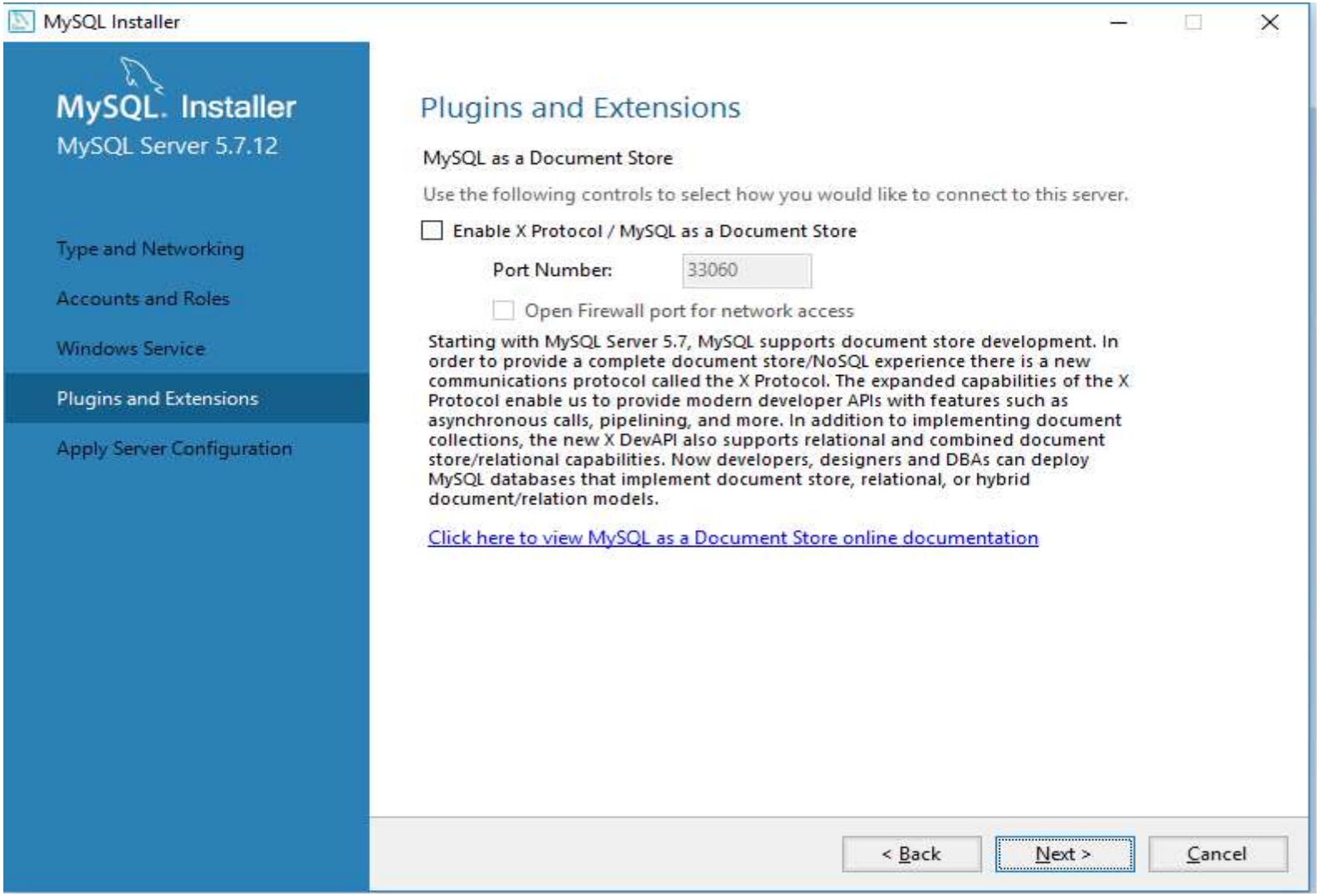


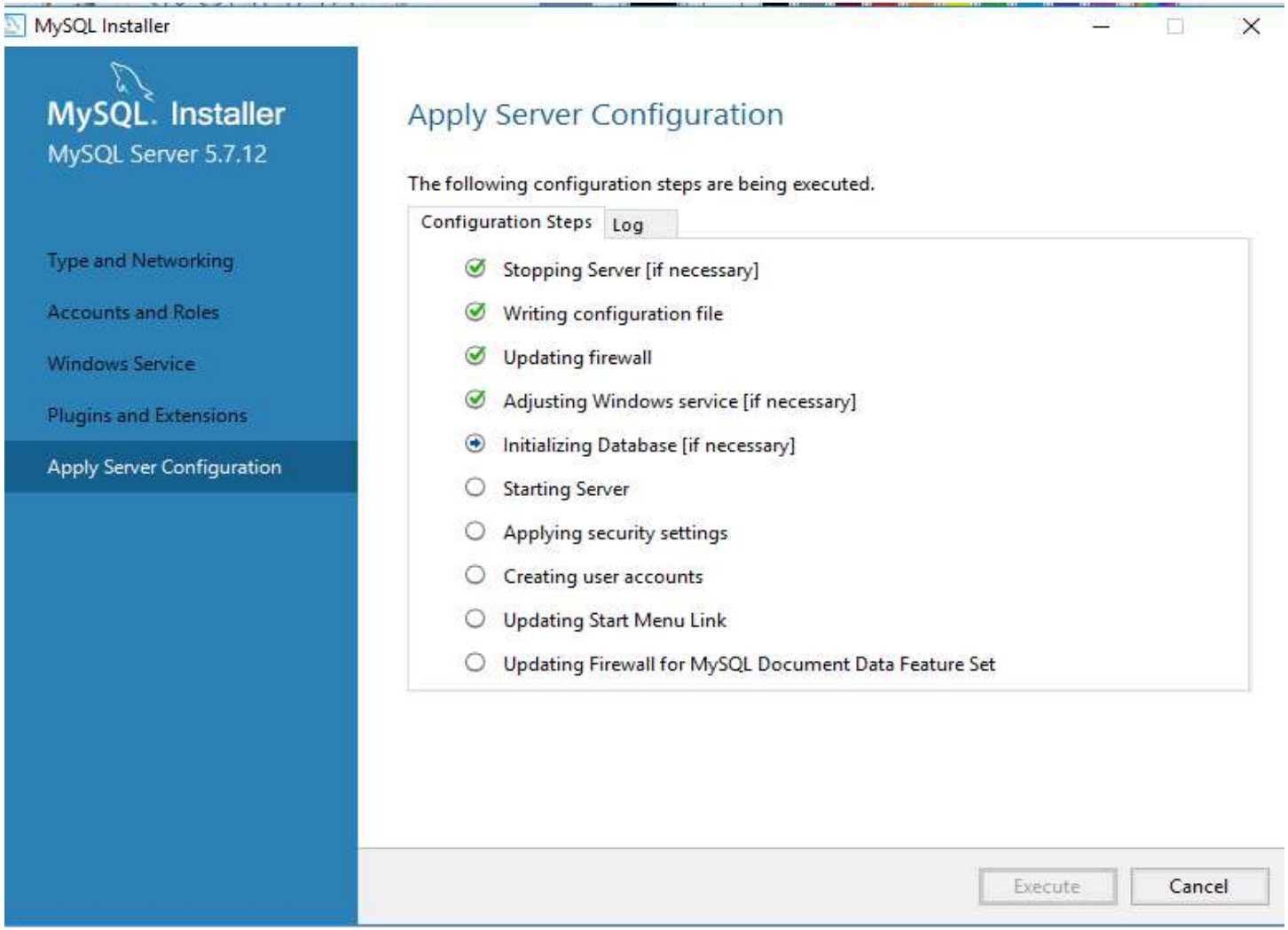
Port to connect to MYSQL

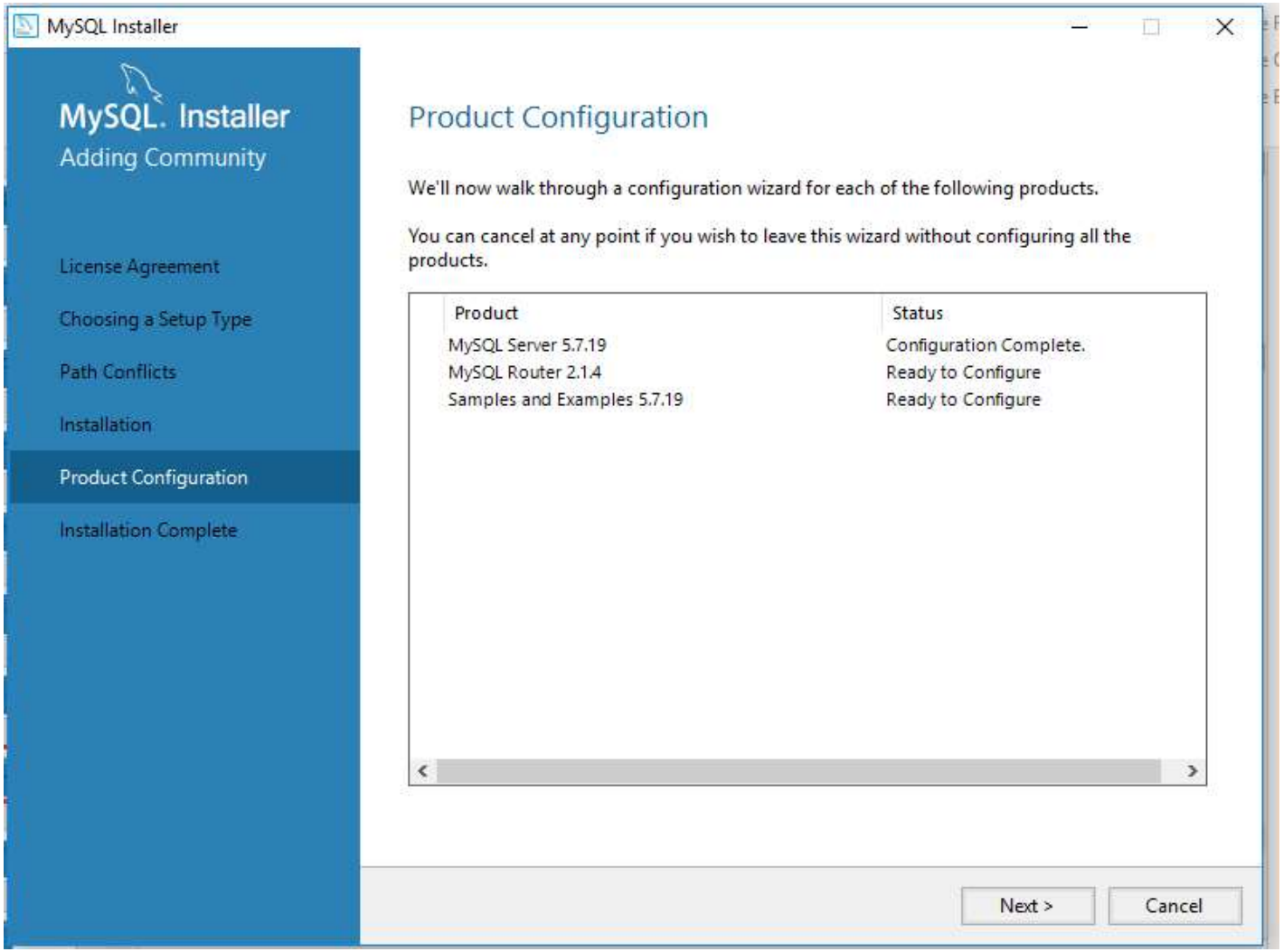


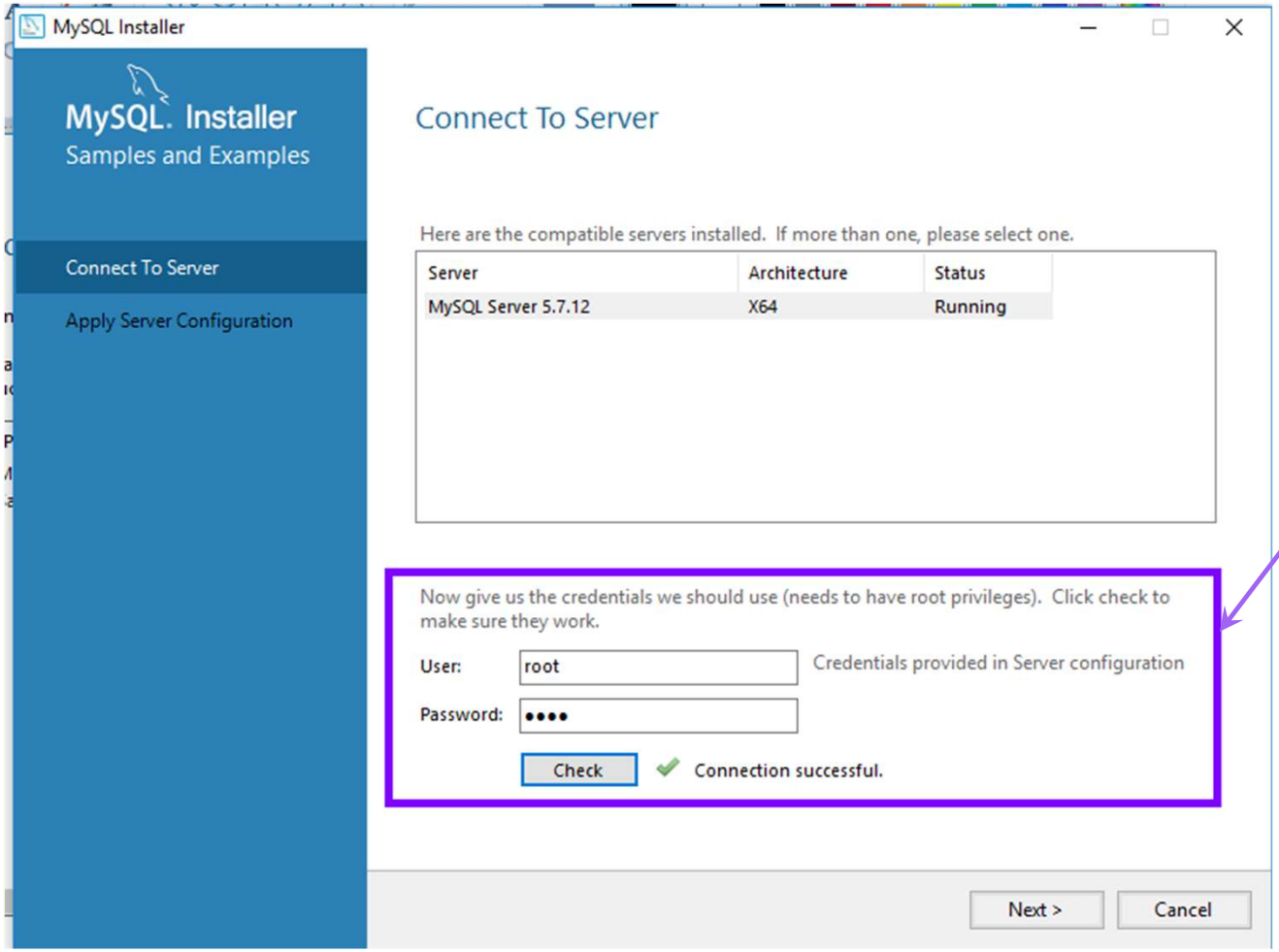
Enter the password. You will need to store these credentials as you need them to access MySQL.


MySQL Installer
MySQL. Installer
MySQL Server 5.7.12
Type and Networking
Accounts and Roles
Windows Service
Plugins and Extensions
Apply Server Configuration
Windows Service
Configure MySQL Server as a Windows Service
Windows Service Details
Please specify a Windows Service name to be used for this MySQL Server instance. A unique name is required for each instance.
Windows Service Name: MySQL57
Start the MySQL Server at System Startup
Run Windows Service as ...
The MySQL Server needs to run under a given user account. Based on the security requirements of your system you need to pick one of the options below.
Standard System Account
Recommended for most scenarios.
Custom User
An existing user account can be selected for advanced scenarios.
< Back
Next >
Cancel

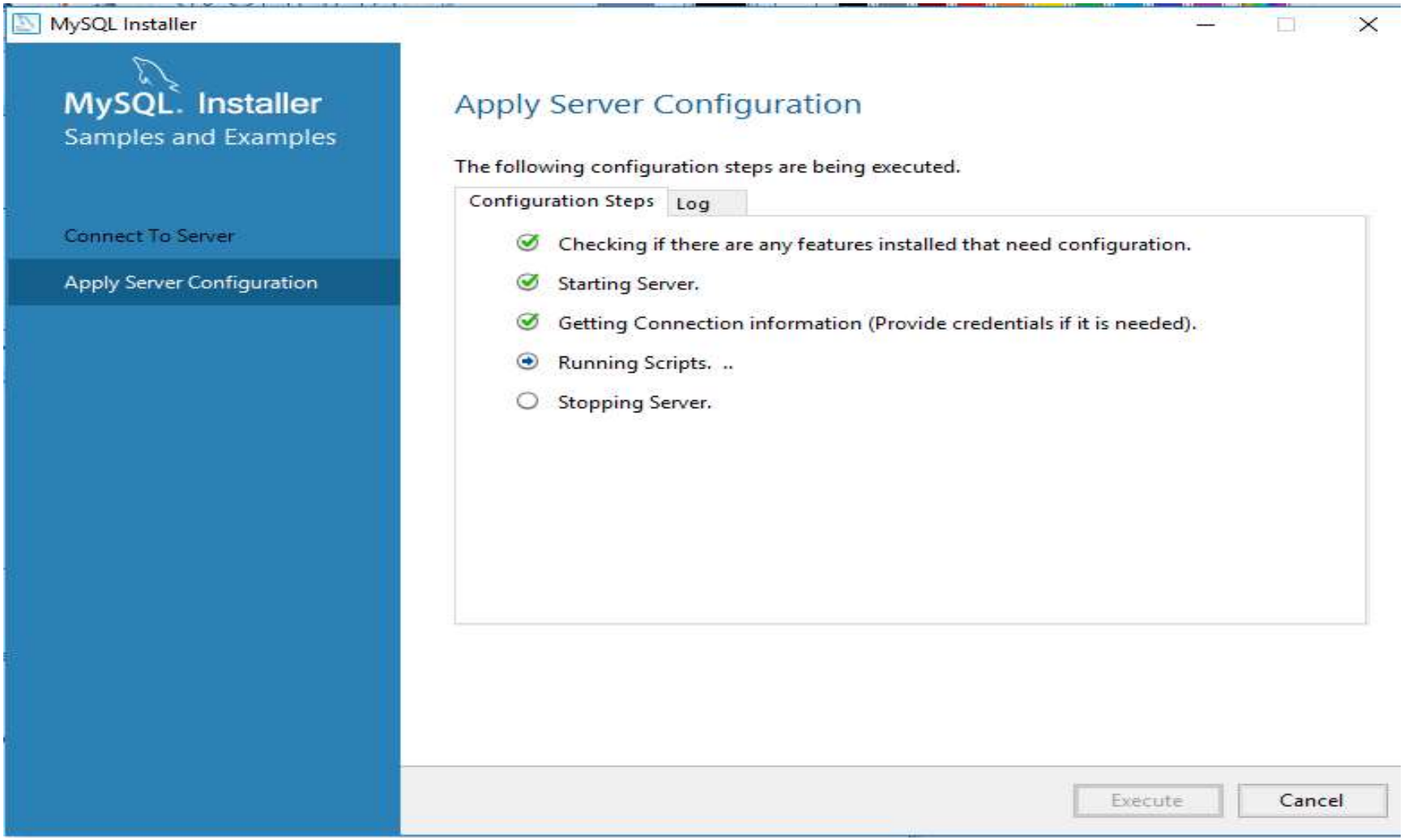

MySQL Installer
MySQL. Installer
MySQL Server 5.7.12
Type and Networking
Accounts and Roles
Windows Service
Plugins and Extensions
Apply Server Configuration
Plugins and Extensions
MySQL as a Document Store
Use the following controls to select how you would like to connect to this server.
Enable X Protocol / MySQL as a Document Store
Port Number:
33060
Open Firewall port for network access
Starting with MySQL Server 5.7, MySQL supports document store development. In order to provide a complete document store/NoSQL experience there is a new communications protocol called the X Protocol. The expanded capabilities of the X Protocol enable us to provide modern developer APIs with features such as asynchronous calls, pipelining, and more. In addition to implementing document collections, the new X DevAPI also supports relational and combined document store/relational capabilities. Now developers, designers and DBAs can deploy MySQL databases that implement document store, relational, or hybrid document/relation models.
Click here to view MySQL as a Document Store online documentation
< Back
Next >
Cancel

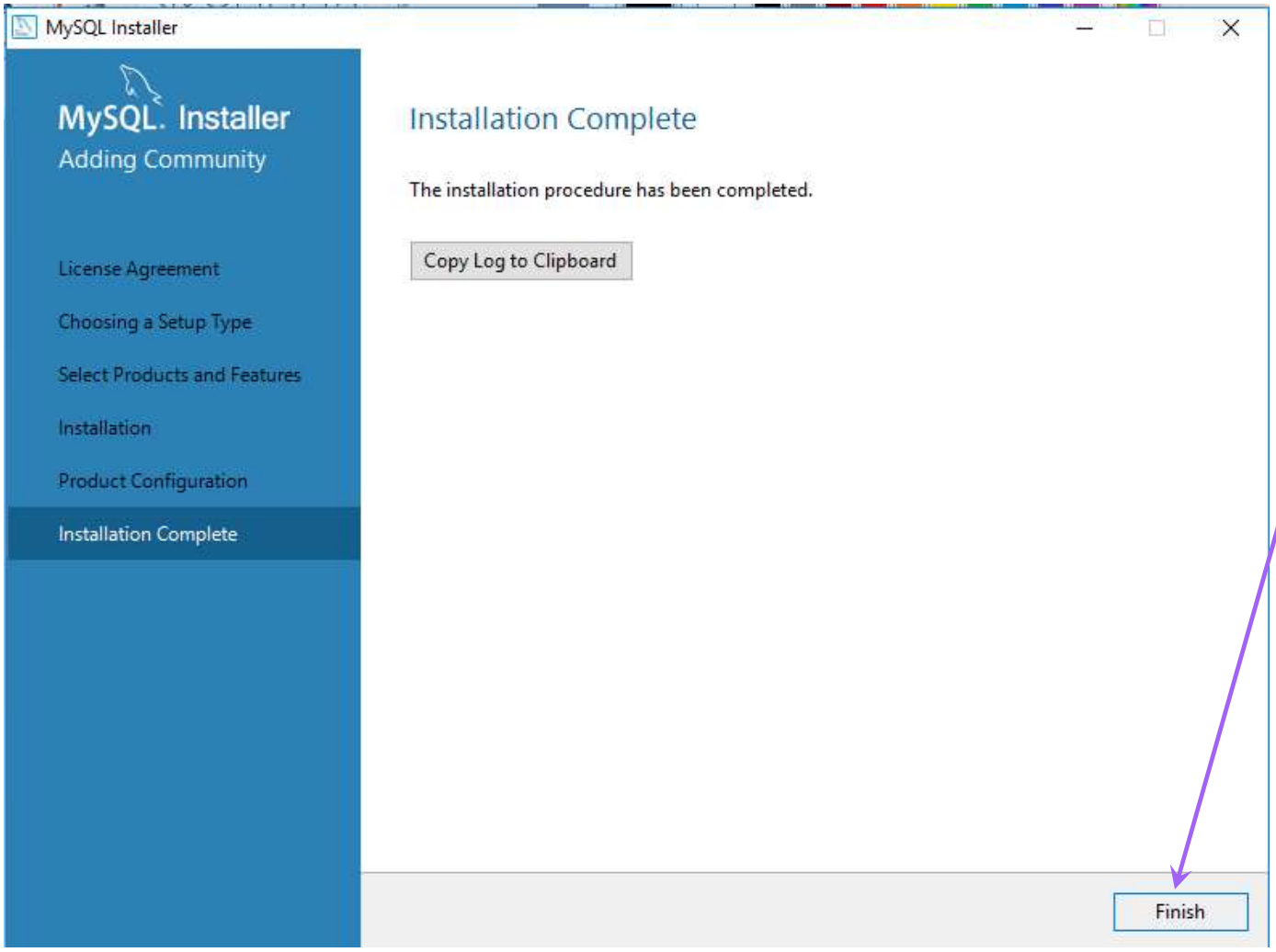

MySQL Installer
MySQL. Installer
MySQL Server 5.7.12
Type and Networking
Accounts and Roles
Windows Service
Plugins and Extensions
Apply Server Configuration
Apply Server Configuration
The following configuration steps are being executed.
Configuration Steps
Log
Stopping Server [if necessary]
Writing configuration file
Updating firewall
Adjusting Windows service [if necessary]
Initializing Database [if necessary]
Starting Server
Applying security settings
Creating user accounts
Updating Start Menu Link
Updating Firewall for MySQL Document Data Feature Set
Execute
Cancel


MySQL Installer
MySQL. Installer
Adding Community
License Agreement
Choosing a Setup Type
Path Conflicts
Installation
Product Configuration
Installation Complete
Product Configuration
We'll now walk through a configuration wizard for each of the following products.
You can cancel at any point if you wish to leave this wizard without configuring all the products.
Product
Status
MySQL Server 5.7.19
Configuration Complete.
MySQL Router 2.1.4
Ready to Configure
Samples and Examples 5.7.19
Ready to Configure
Next >
Cancel



Before proceeding, make sure you check the connection by providing required credentials.

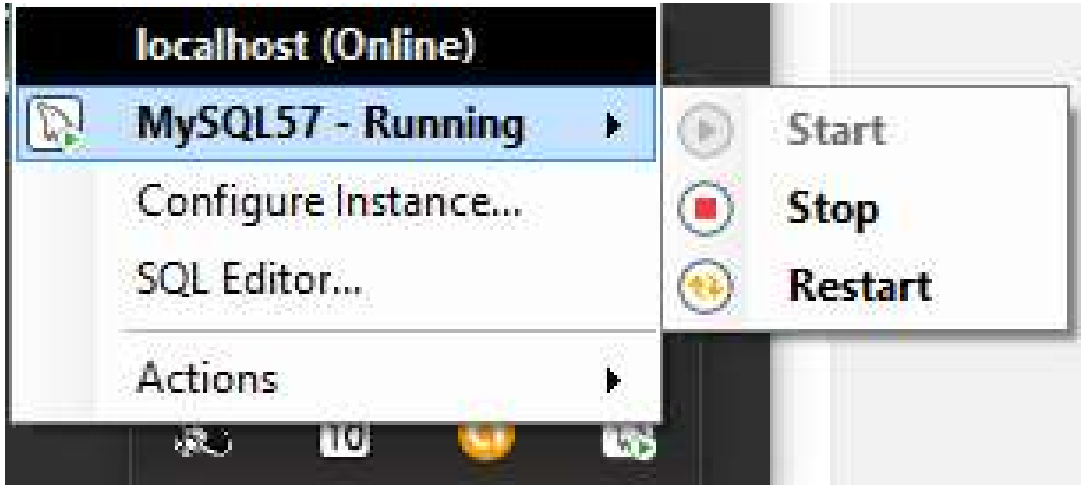

MySQL Installer
MySQL. Installer
Samples and Examples
Connect To Server
Apply Server Configuration
Apply Server Configuration
The following configuration steps are being executed.
Configuration Steps
Log
Checking if there are any features installed that need configuration.
Starting Server.
Getting Connection information (Provide credentials if it is needed).
Running Scripts. ..
Stopping Server.
Execute
Cancel



Now, the installation is completed.
.

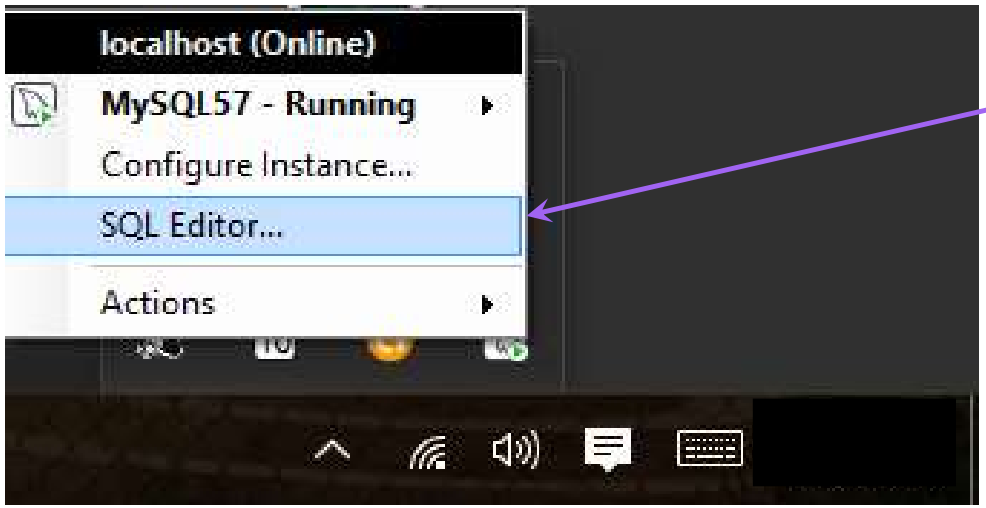
Post Installation…

After the installation, we can monitor MySQL server by accessing MySQL notifier from the task bar.



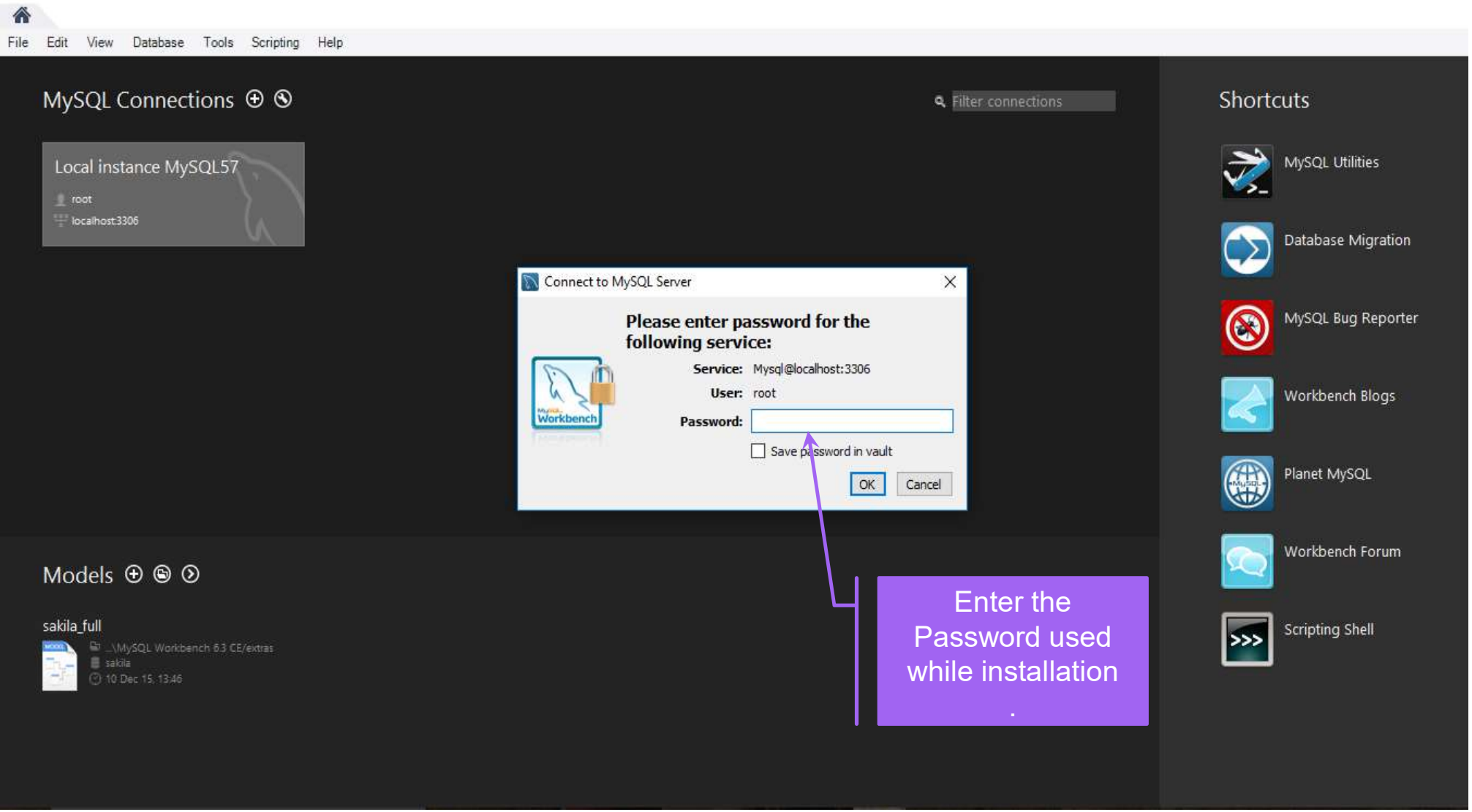
Through the MySQL notifier, we could start, stop or restart MySQL components.

To open the Workbench, where we write and execute several queries, we should click on “SQL Editor”.



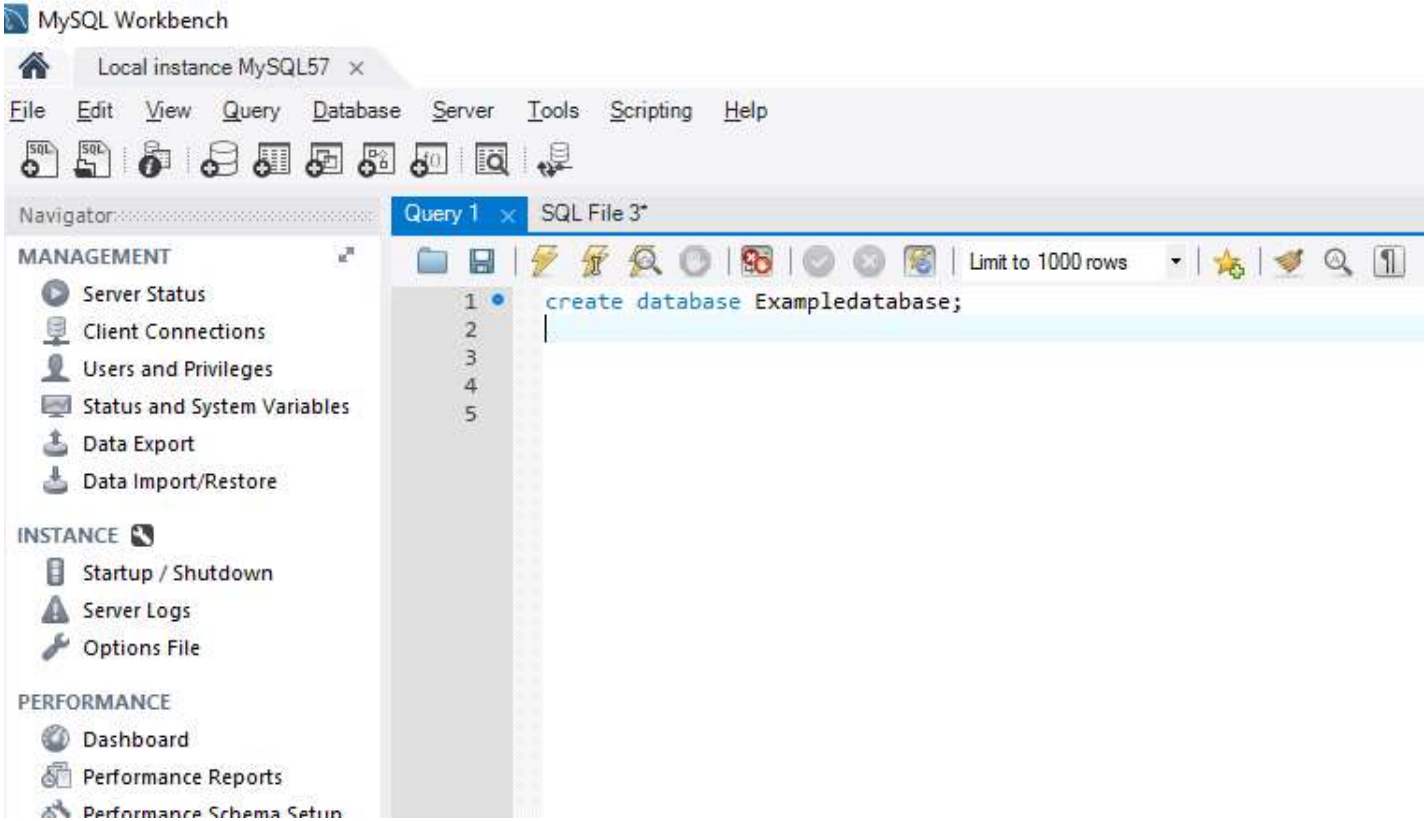
Now, the installation is completed.
.

Once the workbench is opened, we need to enter the credentials (used during installation) to access workbench. Following screenshot exhibits the same.


File Edit View Database Tools Scripting Help
MySQL Connections
Filter connections
Local instance MySQL57
root
localhost:3306
Connect to MySQL Server
Please enter password for the following service:
Service: Mysql@localhost:3306
User: root
Password:
Save password in vault
OK
Cancel
Shortcuts
MySQL Utilities
Database Migration
MySQL Bug Reporter
Workbench Blogs
Planet MySQL
Workbench Forum
Scripting Shell
Models
sakila_full
...\MySQL Workbench 6.3 CE/extras
sakila
10 Dec 15, 13:46
Enter the Password used while installation .

Executing queries in MySQL Workbench

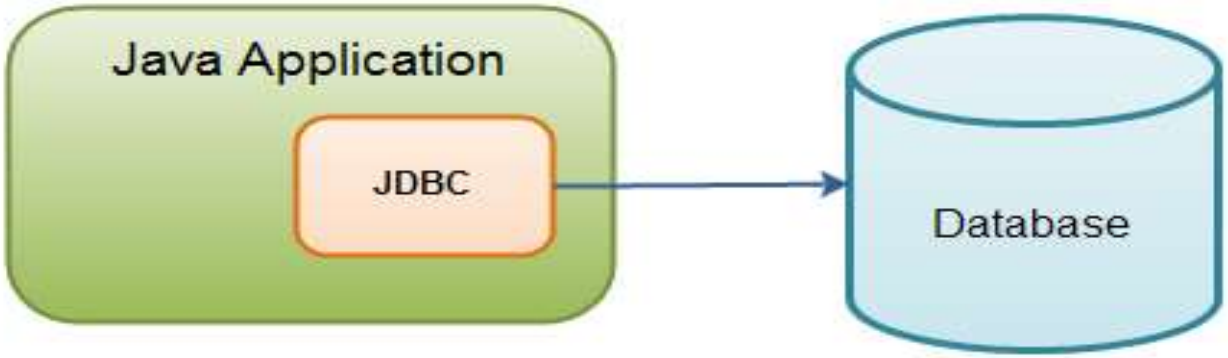
Goto File and select “New Query Tab “ to write and execute the queries.



Executing queries using a Java Application

Now let us write and execute the SQL queries using a Java Application.

How do we do it?
-> By using a JDBC Driver. JDBC stands for Java Database Connectivity.



JDBC Driver Types

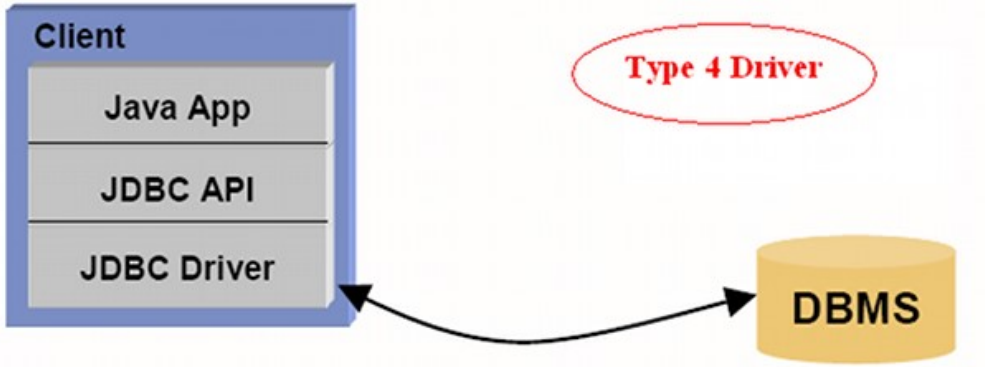
Type 1: JDBC-ODBC Bridge driver (Bridge)

Type 2: Native-API/partly Java driver (Native)

Type 3: All Java/Net-protocol driver (Middleware)

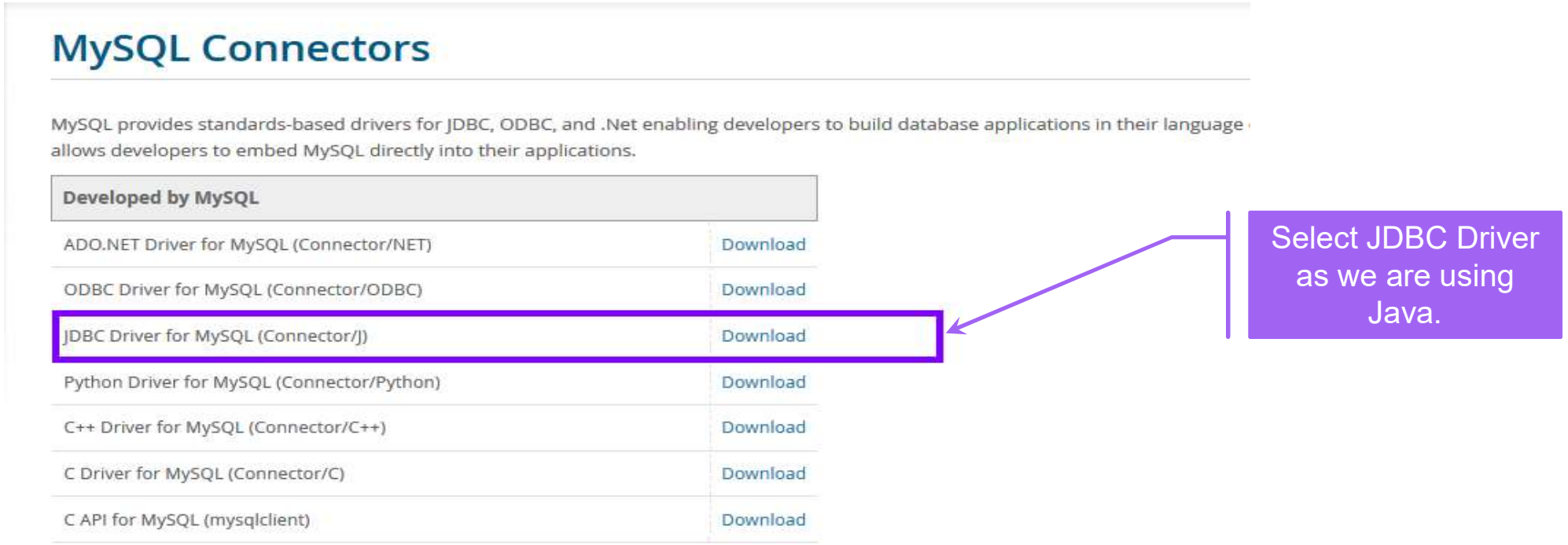
Type 4: All Java/Native-protocol driver (Pure)

We will be using the Type 4 driver , as we use libraries to communicate directly with the database server.



Download the Jar file:

We need a jar file mysql-connector-java-5.1.39-bin.jar to compile and execute a application. If you want to compile the examples from the command line, go to the site http://www.mysql.com/products/connector/ and download the MySQL connector for the Java language.



Steps to Execute the application

Option 1: Executing the Program by adding the jar file path in a bat file .

Add the whole path for jar file in CLASSPATH inside your env-setup.bat file

The location of the JAR files highlighted will differ based on where they are present on your computer
Please make sure you do the changes accordingly

env-setup - Notepad

File Edit Format View Help

```
set JAVA_HOME=C:\Program Files\Java\jdk1.7.0_80

set PATH="C:\Program Files\Java\jdk1.7.0_80\bin";%PATH%

set CLASSPATH=c:\apache-tomcat-7.0.34\lib\servlet-api.jar;
              c:\apache-tomcat-7.0.34\lib\jsp-api.jar;
              c:\apache-tomcat-7.0.34\lib\mongo-java-driver-3.1.0-20150911.172057-81.jar;
              c:\apache-tomcat-7.0.34\lib\mysql-connector-java-5.1.39-bin.jar;

set ANT_HOME=C:\apache-tomcat-7.0.34

set TOMCAT_HOME=C:\apache-tomcat-7.0.34

set CATALINA_HOME=C:\apache-tomcat-7.0.34
```

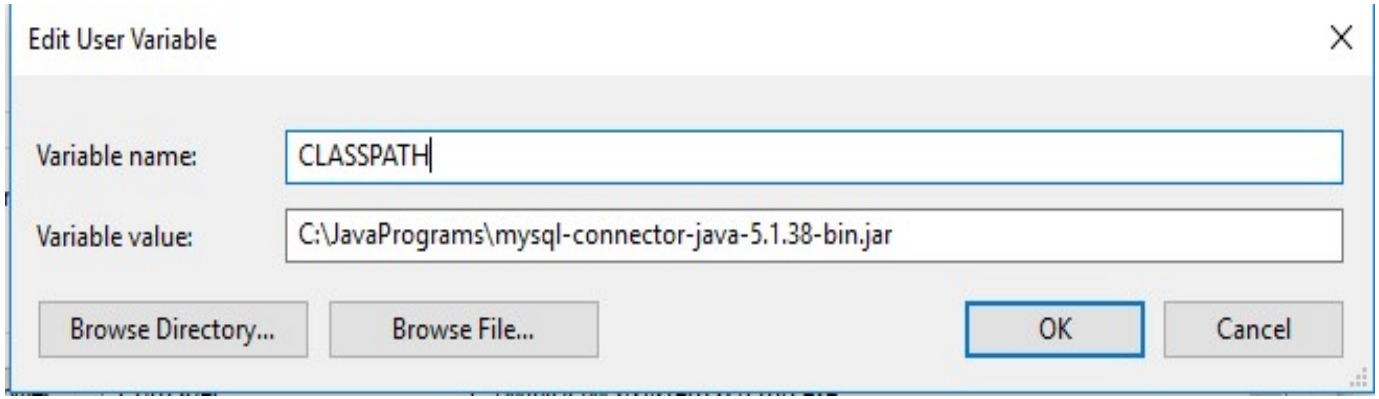
Executing by adding jar file in class path in env-setup.bat file

Option 2 : By setting a classpath variable.

Steps:

- Goto Control Panel -> system -> Advanced system settings -> Environment Variables.
- Under User variables, choose new, and create a new variable called

  Classpath = “The full pathname where jar file is present in the system “
- Click on save.
- Then OK.



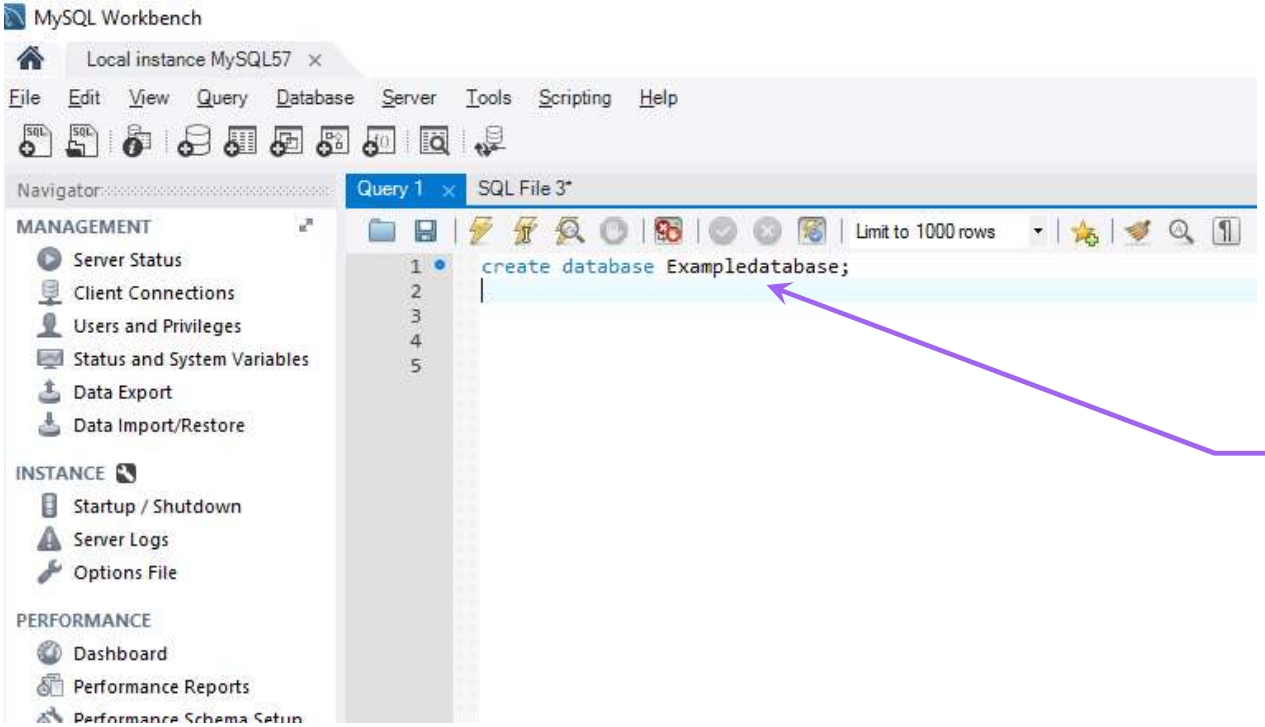
A walkthrough example

In this example,

- First, we will create a database called “exampledatabase” using MySQL workbench.
- Secondly, we will create a table “Registration” within the exampledatabase and store customer login details
- Third, we will create a table “CustomerOrders” within the exampledatabase and store order details for game speed application.

A walkthrough example

- ***Step 1***: Create a database called “exampledatabase” in the workbench space, and execute the SQL command.

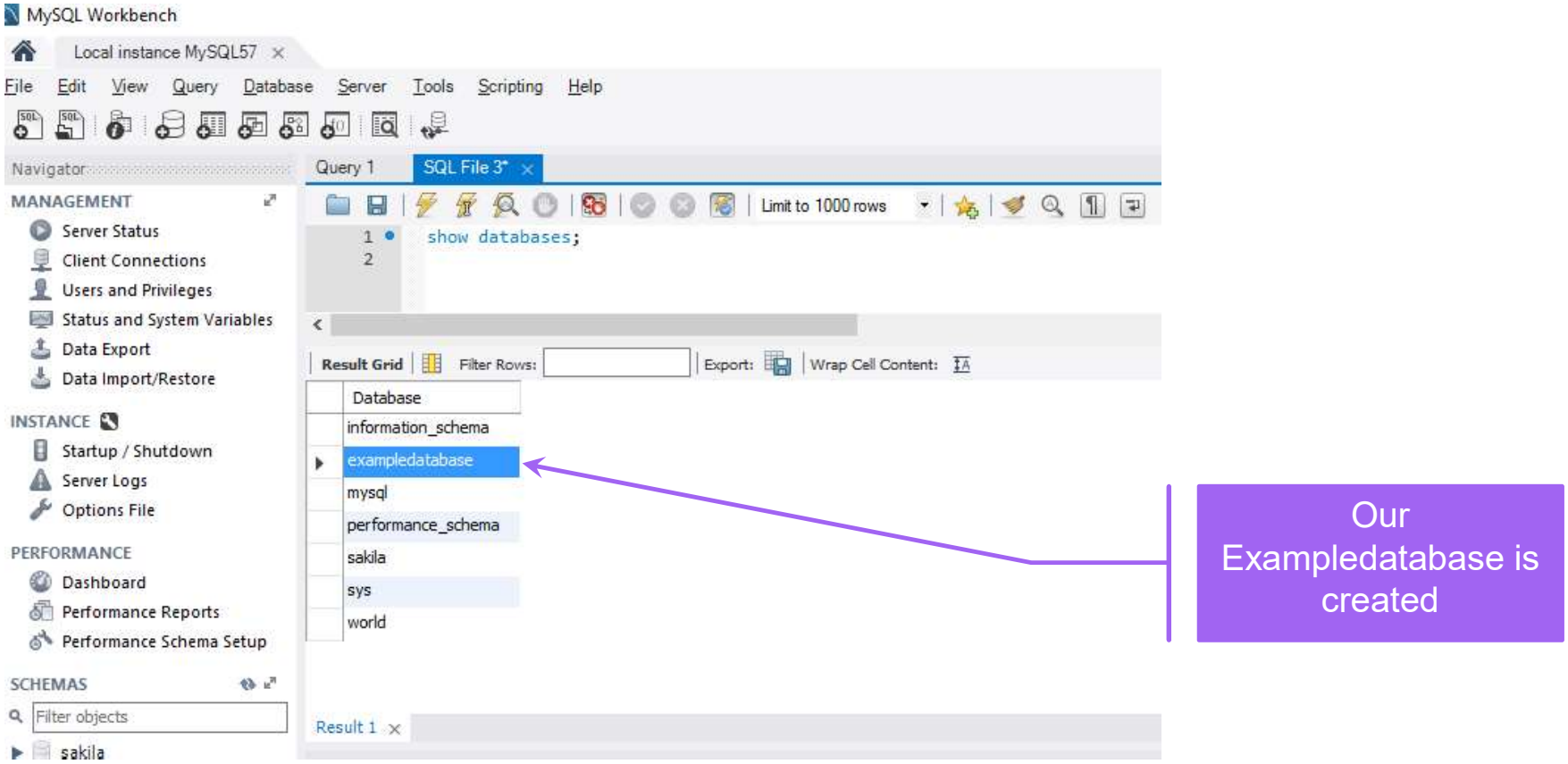


Query to create a new database

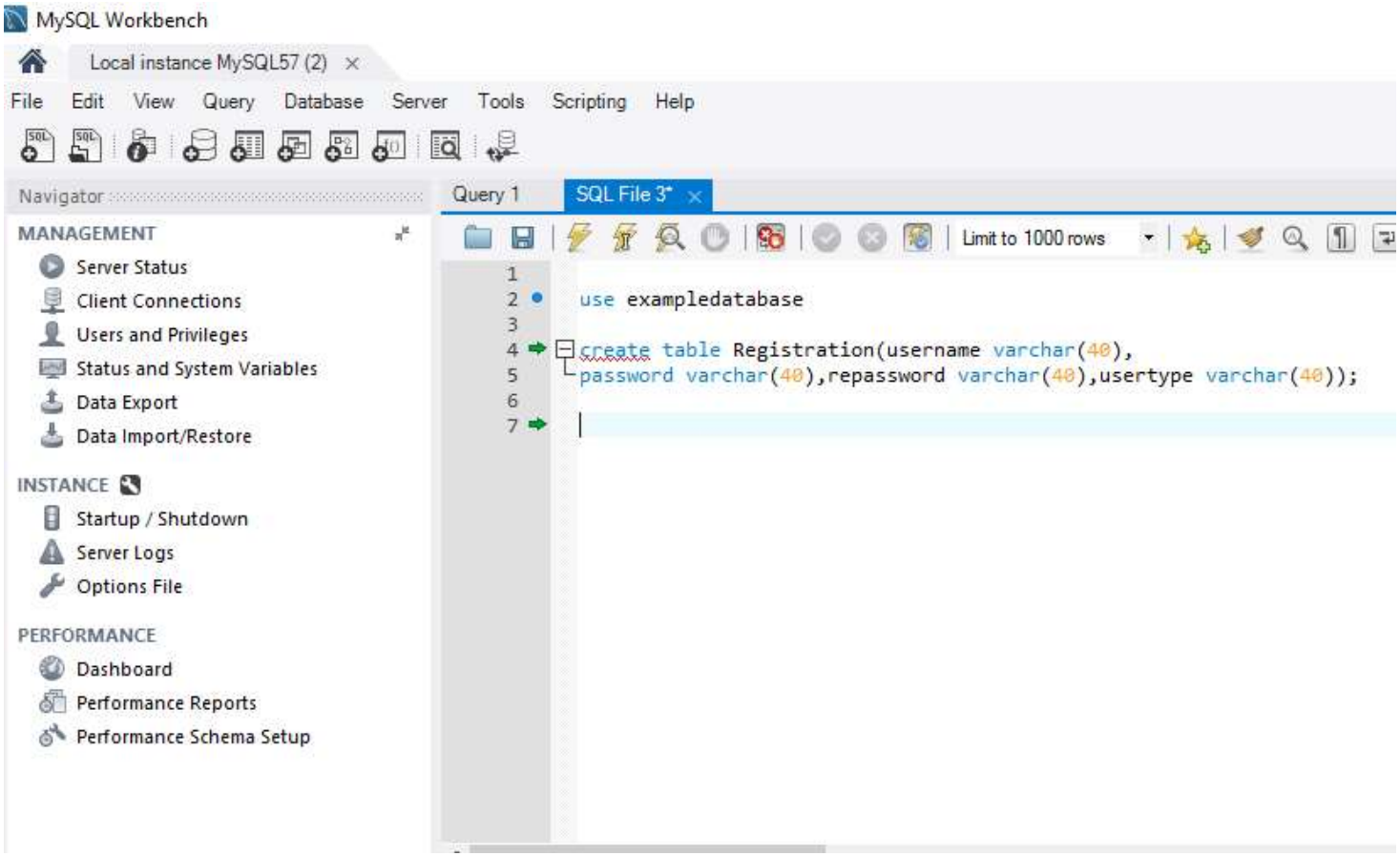
A walkthrough example

Write and execute the following commands to check if the database is created.

-> show databases;

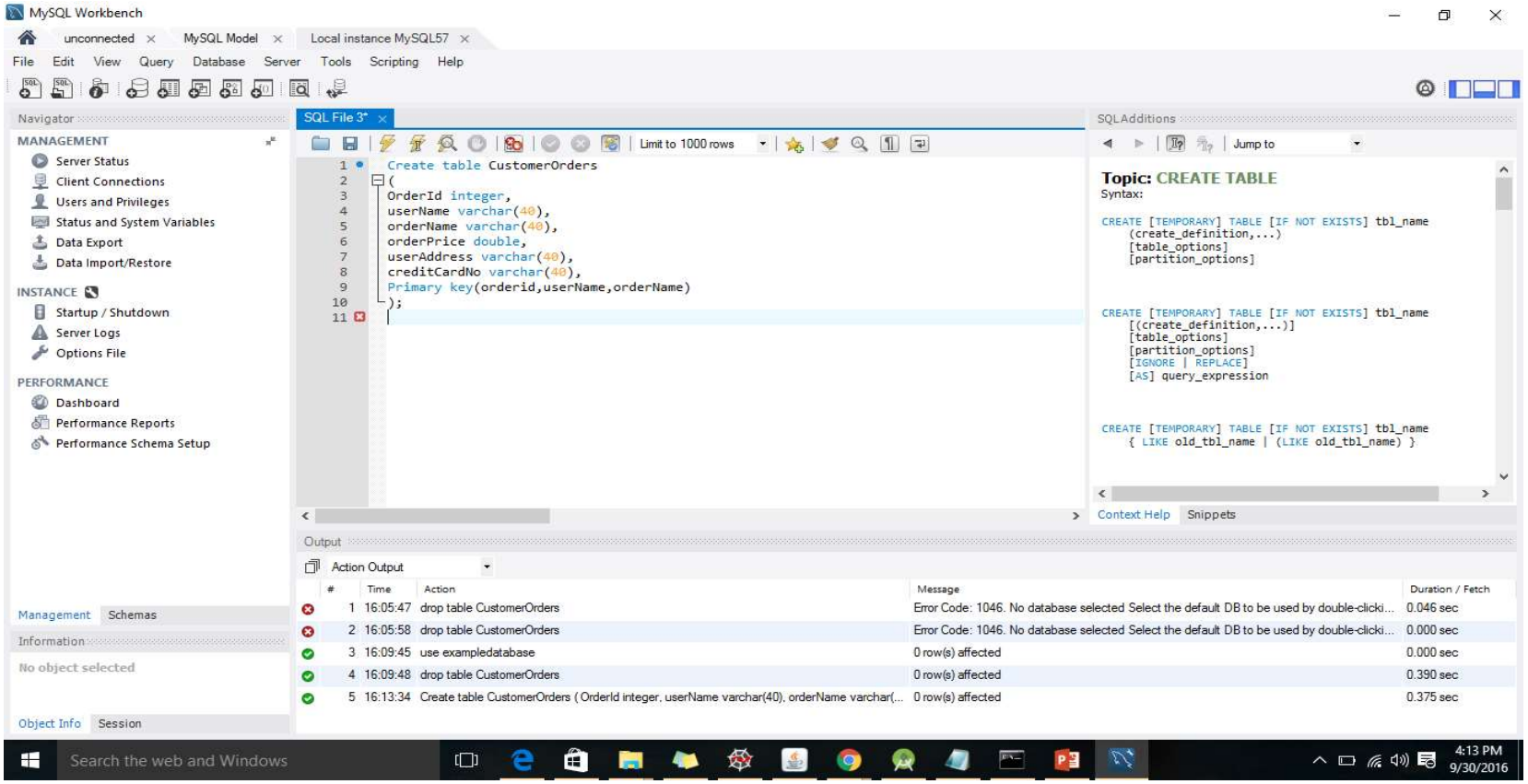


Step 2: Create a table "Registration" in the workbench space, and execute the SQL command.



Step 3: Create a table called “CustomerOrder” in the workbench space, and execute the SQL command.

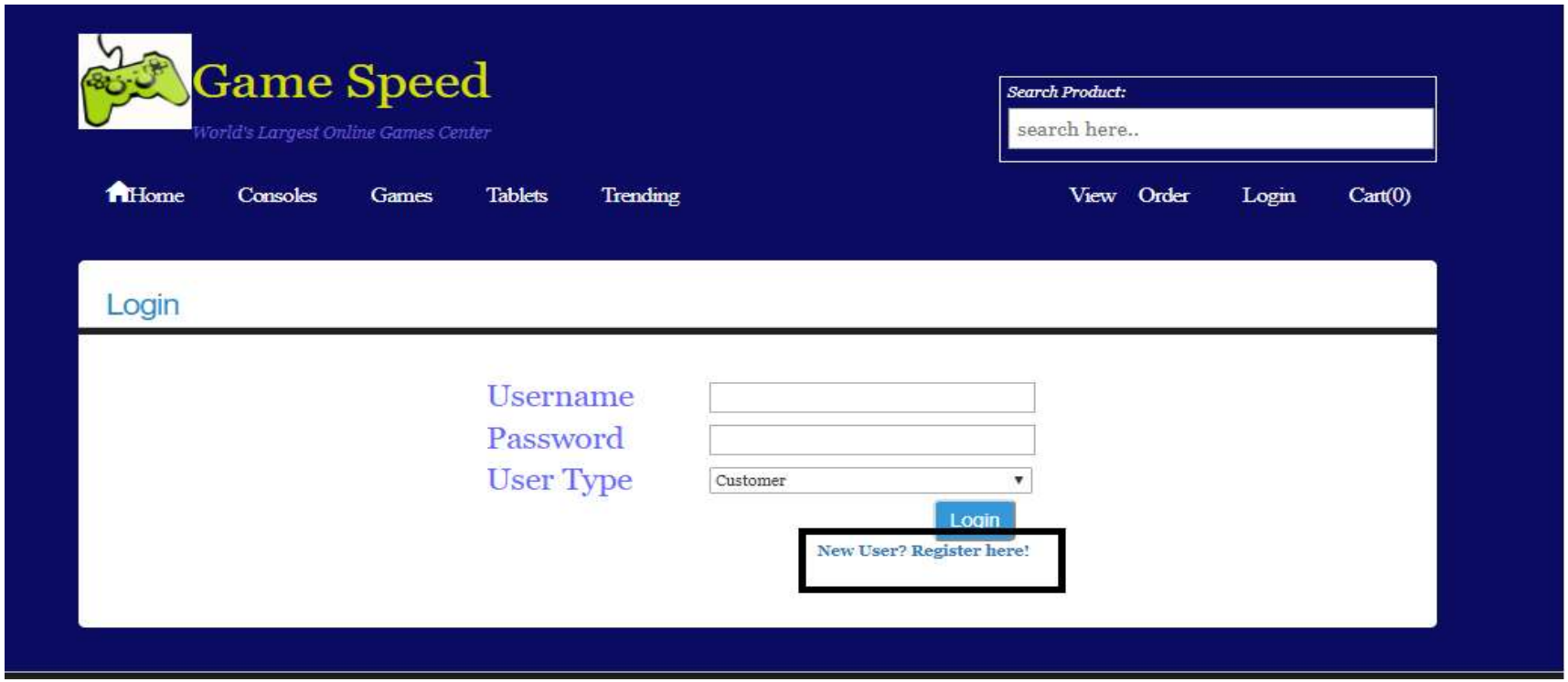
Specify the id as primary key for table



Example –Registration

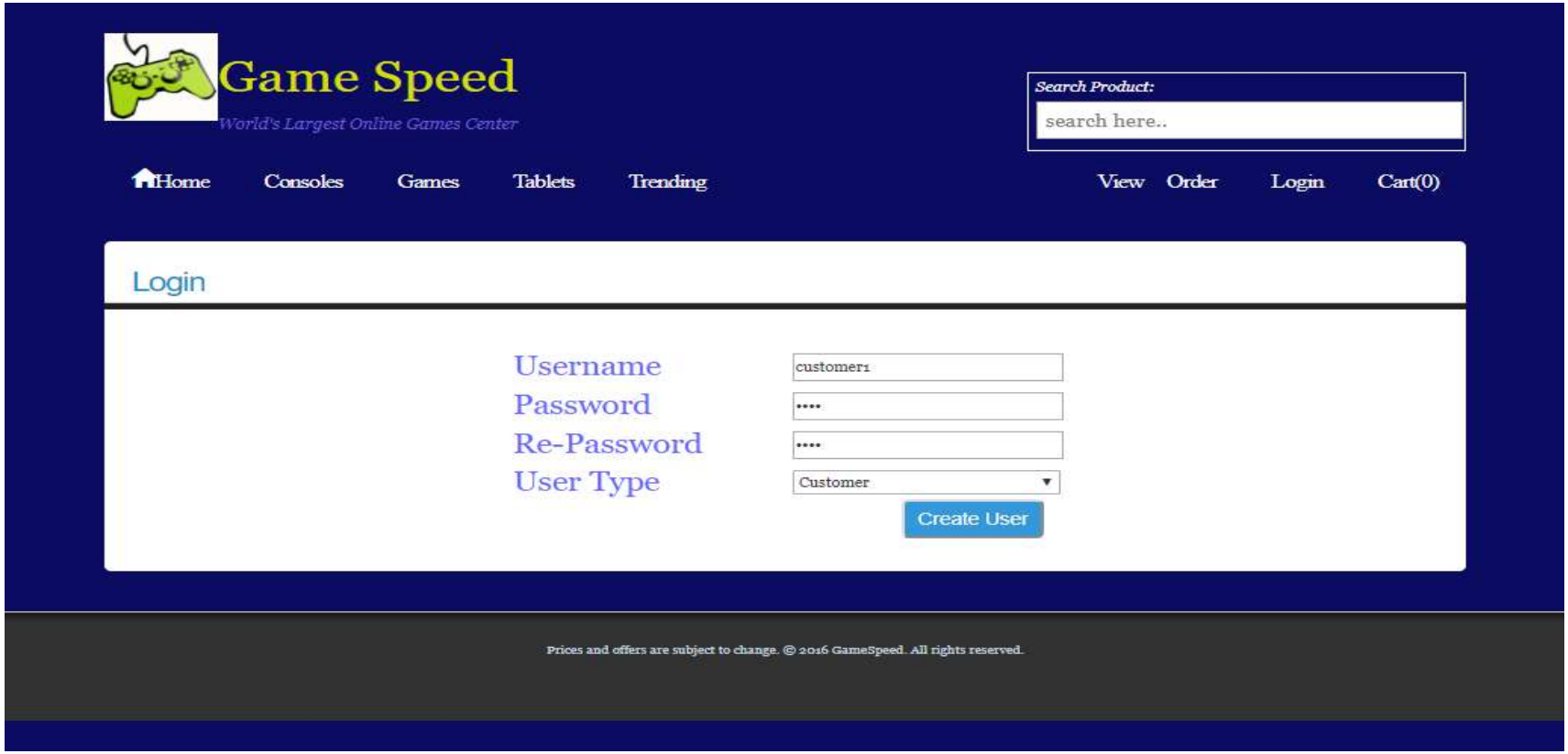
New User can Register into Website by Clicking on the Register here link

On clicking the button user is directed to Registration page

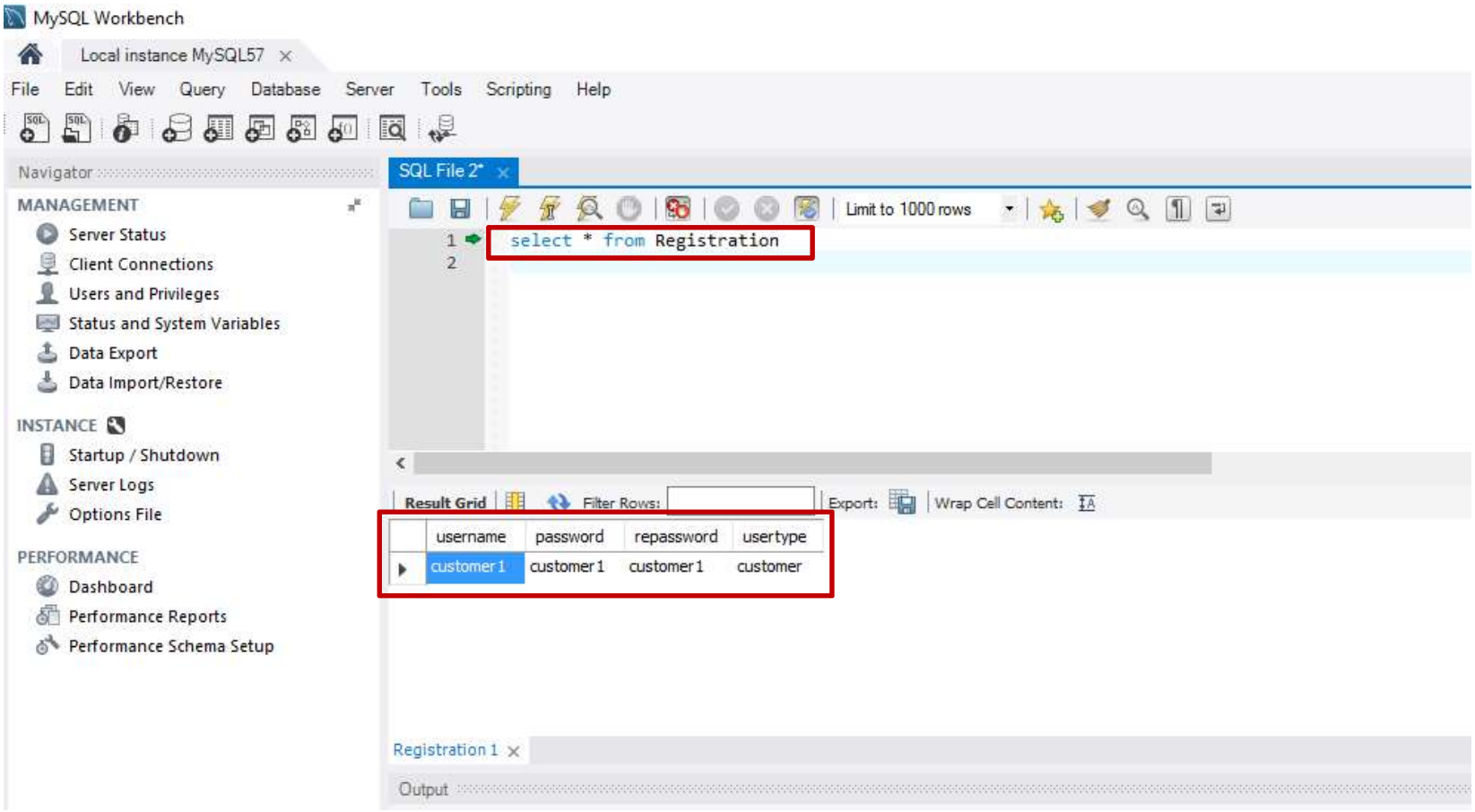


User provides the login information

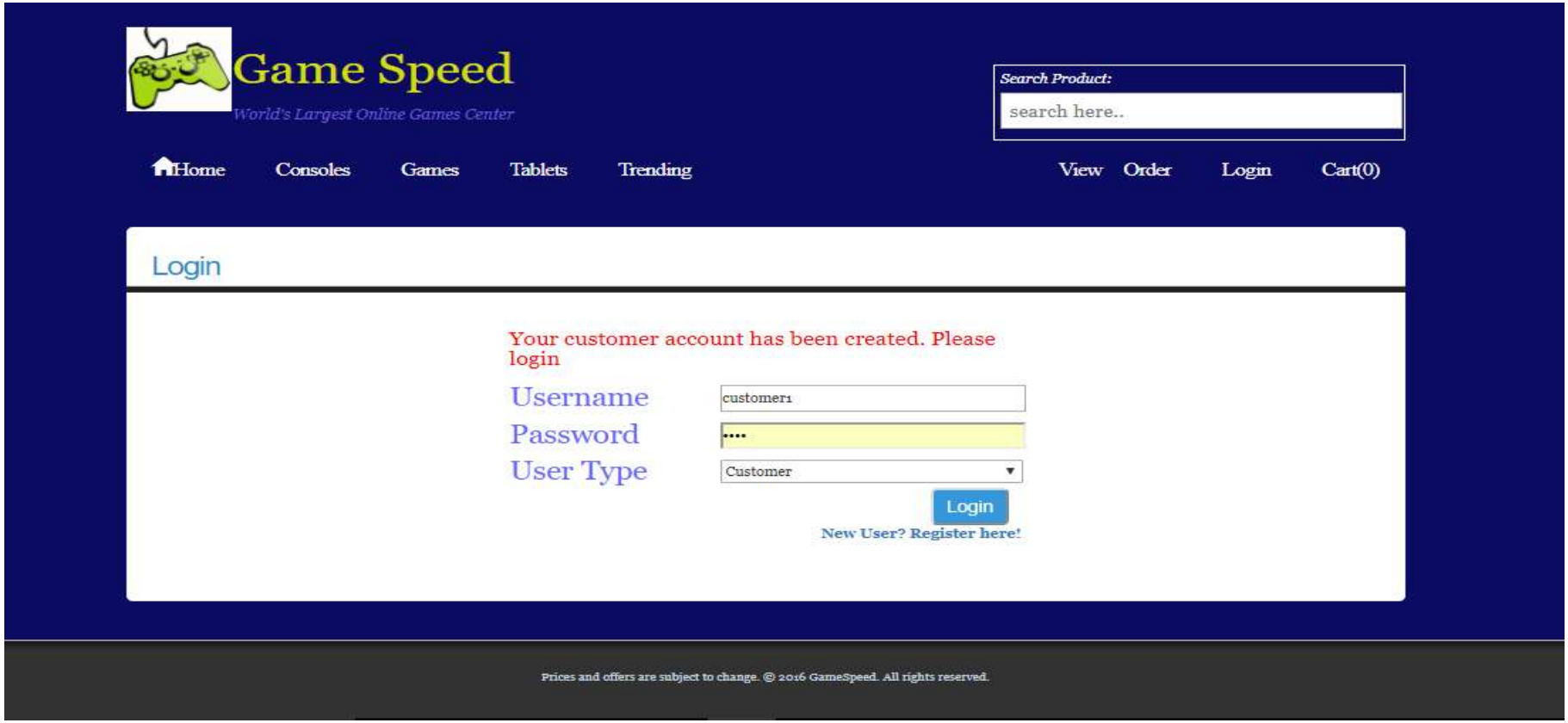
On clicking create user button data is stored in Registration Table of My sql



User data is stored in Registration table you can check using the select query in workbench to check if all the column values are stored properly



After an Account is created for user in Registration table user can login into website with the credentials

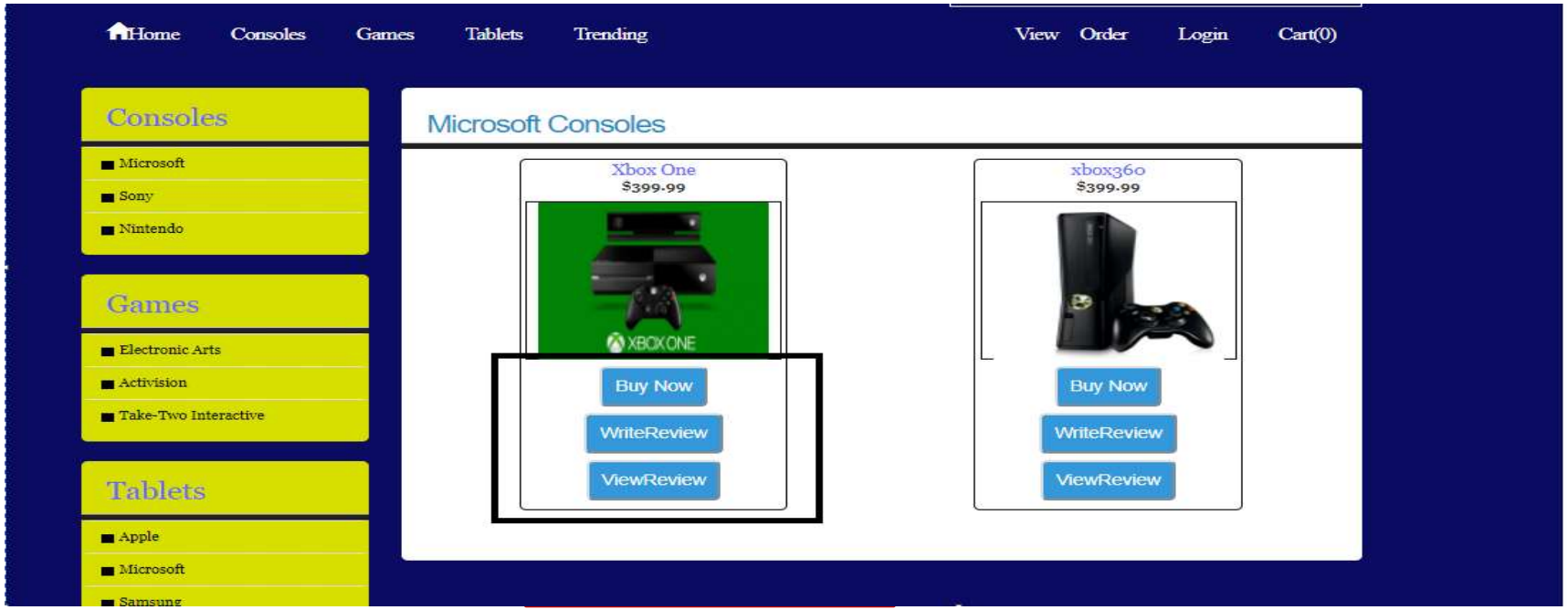


Example –Place Order

- Click on the products available in the navigation bar
- You can also select the products from the left navigation bar

Example – Place Order:

- Clicking on a product type will take you to the product page
- You have different options available such as buy a product , write reviews.

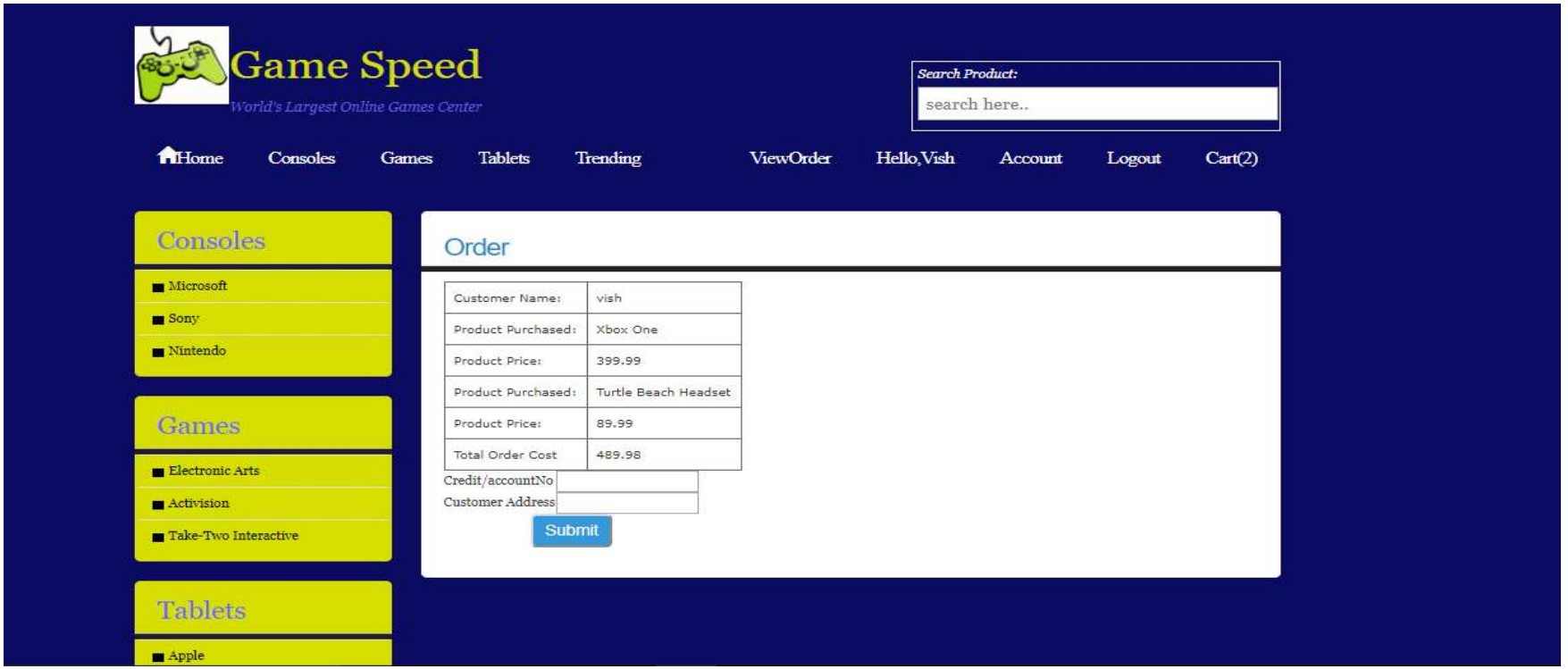


Example – Place Order:

- Click on ‘Buy’ button on the products page to purchase the product
- This should take you to a new page (Cart Servlet) where you can purchase the product
- Click on ‘Check Out’ to place the order for the selected product.

Example – Place Order:

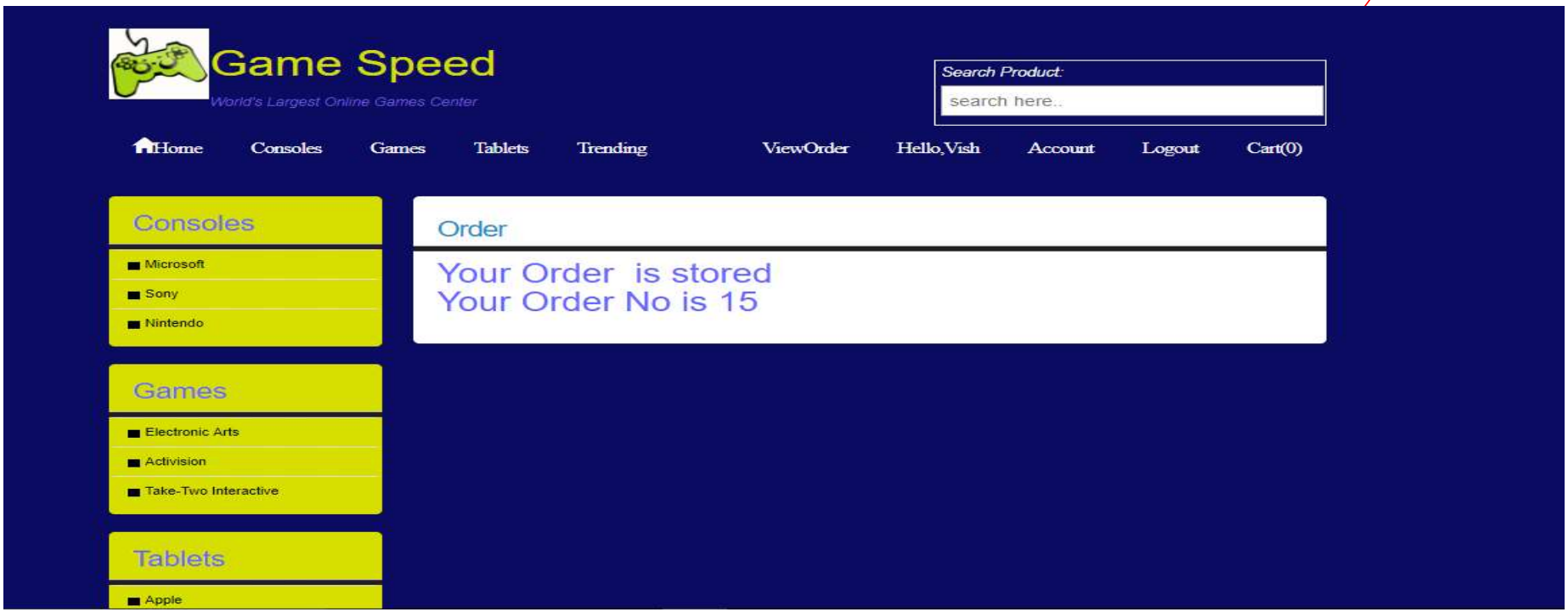
- Clicking on CheckOut Button will take you to the CheckOut webpage where you have to provide your credit card no and address information.



Example – Place Order:

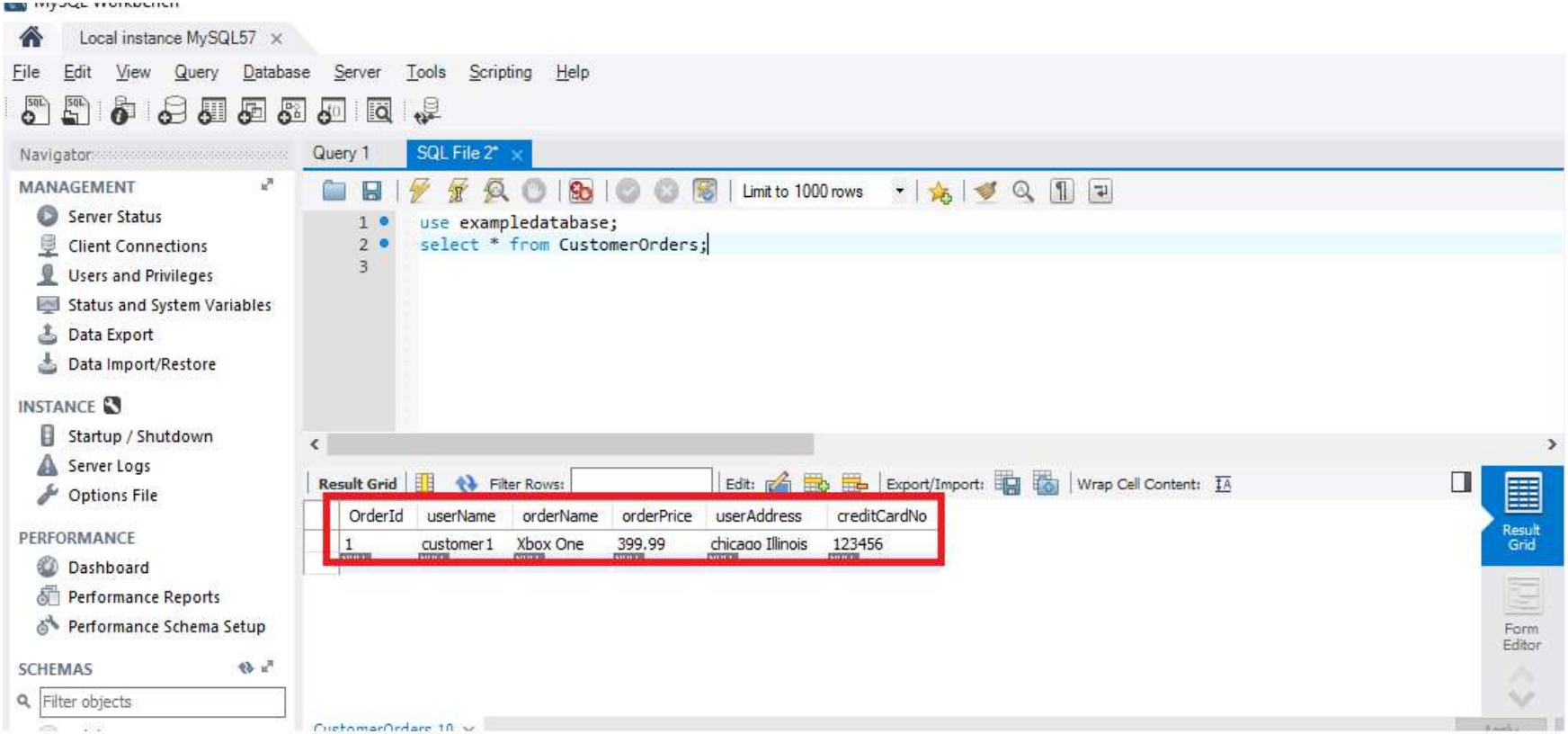
- On clicking the submit button from check out page order will be stored in My Sql database and order no will be generated

Order is Deleted from the cart



Example – Place Order:

You can Check if the order Stored by executing select Query in database using sql workbench



Servlets MySql Connection

We will be using com.mysql.jdbc.Driver for connecting mysql from servlets

Syntax:

Connection conn=

Class.forName("com.mysql.jdbc.Driver").newInstance();

DriverManager.*getConnection() method is used to connect to my sql database*

Specify the database url , user name and password as parameter to the getConnection() method

```
conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/exampledatabase","root",“root");
```

Prepared Statement Execution

- Prepared Statement are used to generate Sql statement for a Query String in java
- Syntax:

  Prepared Statement ps=conn.prepareStatement(“select * from Registration where username=? And usertype=?”)

- Specify the Query String as parameter inside conn.prepareStatement() to perform insert or select into database from java
- ? Are place holder where we need to provide the value for a particular query
- In the next line ? We will replace with actual parameter value As

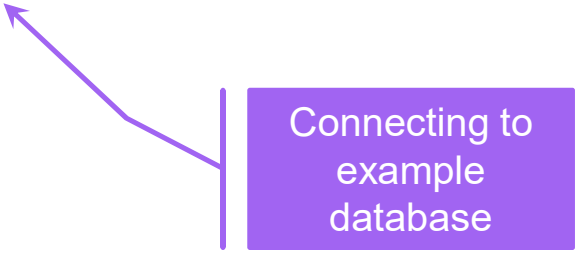
  ps.setString(1,”customer1”) –1 denotes the first ? Place

  ps.setString(2,”customer”) – 2 denotes the second ? Place

Walkthrough to get connect to Database from Servlet

MySqlDataStoreUtilities class to connect Database from Servlet

```
public class MySqlDataStoreUtilities
{
Connection conn = null;
public void getConnection()
{
    try
    {
    Class.forName("com.mysql.jdbc.Driver").newInstance();
    conn=
DriverManager.getConnection("jdbc:mysql://localhost:3306/exampledatabase"
              ,"root","root");
    }
    catch(Exception e)
    {}
}
```



Walkthrough for User Registration Code Snippet

User Registration Sample Code

```
HashMap<String, User> hm=new HashMap<String, User>();
        try
        {
        hm=MySqlDataStoreUtilities.selectUser();
        }
        catch(Exception e){}
        if(hm.containsKey(username))
        error_msg = "Username already exist as " + usertype;
        else
        {
        User user = new User(username,password,usertype);
        hm.put(username, user);
        MySqlDataStoreUtilities.insertUser(…);
        }
```

Calling utility function to select data from database and storing orders in hashmap

Calling utility function to insert user details in database

Utility Function For Registration

```
public static void insertUser(String username,String password,String
usertype){
try{
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        conn =
        DriverManager.getConnection("jdbc:mysql://localhost:3306/exampleda
        tabase“,”root”,”root”)

        String insertIntoCustomerRegisterQuery = "INSERT INTO
        Registration(username,password,usertype) "
                + "VALUES (?,?,?);";
        PreparedStatement pst =
        conn.prepareStatement(insertIntoCustomerRegisterQuery);
                pst.setString(1,username);
                pst.setString(2,password);
                pst.setString(3,usertype);
                pst.execute();
    }
    catch(Exception e){}
}
```

Connecting to example database

Query to insert data to table

Setting Value for Each Parameter

Execute method will insert data into database

Walkthrough for Inserting Order Code Snippet

Storing Order Payments

```
public void storePayment(int orderId,String orderName,double orderPrice,String
userAddress,String creditCardNo){
HashMap<Integer, ArrayList<OrderPayment>> orderPayments= new HashMap<Integer,
ArrayList<OrderPayment>>(); try
{orderPayments= MySqlDataStoreUtilities.selectOrder();
}
catch(Exception e){}
if(!orderPayments.containsKey(orderId)){
ArrayList<OrderPayment> arr = new ArrayList<OrderPayment>();
orderPayments.put(orderId, arr);
}
ArrayList<OrderPayment> listOrderPayment = orderPayments.get(orderId)
OrderPayment orderpayment = new OrderPayment (…);
listOrderPayment.add(orderpayment);
try
{MySqlDataStoreUtilities.insertOrder(…);
}}
```

Calling utility function to select data from database and storing orders in hashmap

Calling utility function to inserting orders in database

Utility Function for Select Order into hashmap

```
public static HashMap<Integer, ArrayList<OrderPayment>> selectOrder()
{
 HashMap<Integer,ArrayList<OrderPayment>> orderPayments=new
       HashMap<Integer,ArrayList<OrderPayment>>();
 try{ getConnection();
     String selectOrderQuery ="select * from customerorders";
     PreparedStatement pst = conn.prepareStatement(selectOrderQuery);
     ResultSet rs = pst.executeQuery();
     ArrayList<OrderPayment> orderList=new ArrayList<OrderPayment>();
     while(rs.next())
     {if(!orderPayments.containsKey(rs.getInt("OrderId")))
        {ArrayList<OrderPayment> arr = new ArrayList<OrderPayment>();
     orderPayments.put(rs.getInt("orderId"), arr);}
     ArrayList<OrderPayment> listOrderPayment =orderPayments.get(rs.getInt("OrderId"));
     OrderPayment order= new
         OrderPayment(rs.getInt("OrderId"),rs.getString("userName"),rs.getString("orderName"));
     listOrderPayment.add(order);
     }
   }catch(…){}
       return orderPayments;}
```

ResultSet used to store table data obtained from database in servlet

Iterate through ResultSet and Store each order into class object

Utility Function for storing orders

```
public static void insertOrder(int orderId,String userName,String orderName)
{       try
        {
                Class.forName("com.mysql.jdbc.Driver").newInstance();
        conn =
        DriverManager.getConnection("jdbc:mysql://localhost:3306/exampledatabase"
                ,"root","root");
        String insertIntoCustomerOrderQuery = "INSERT INTO
        customerOrders(OrderId,UserName,OrderName) " + "VALUES (?,?,?);";
        PreparedStatement pst =
        conn.prepareStatement(insertIntoCustomerOrderQuery);
                pst.setInt(1,orderId);
                pst.setString(2,userName);
                pst.setString(3,orderName);
                pst.execute();
        }
                catch(Exception e){}
}
```