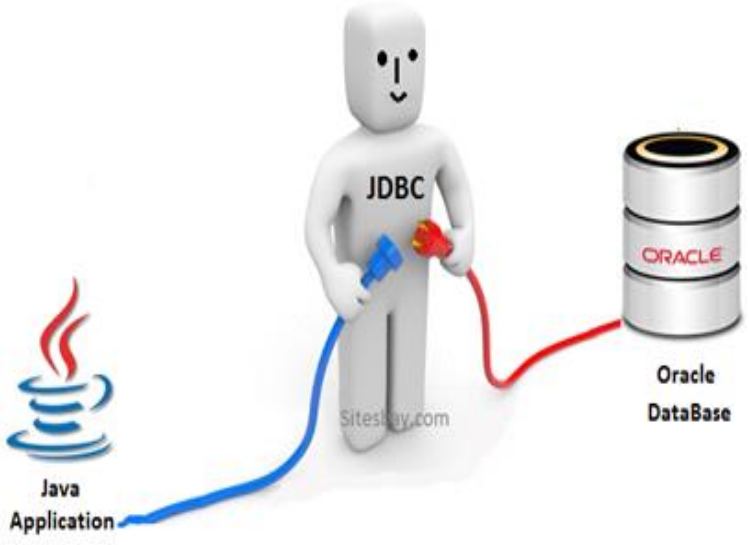
(Formerly ITM University, Gurugram)

Java Database Connectivity (JDBC)

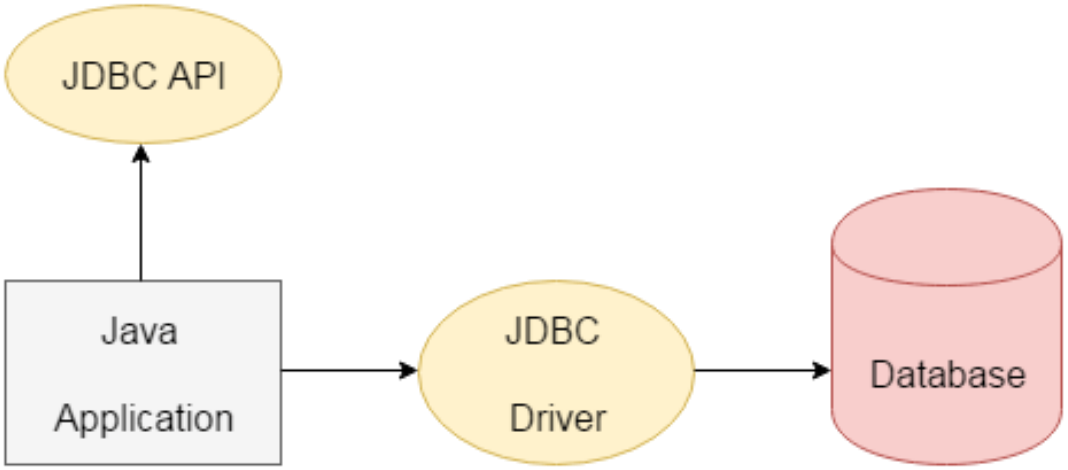
Contents

- Introduction
- Establishing Connection
- JDBC Demo
- JDBC Drivers
- Statements
- ResultSet Class
- Meta Data
- Transaction Management



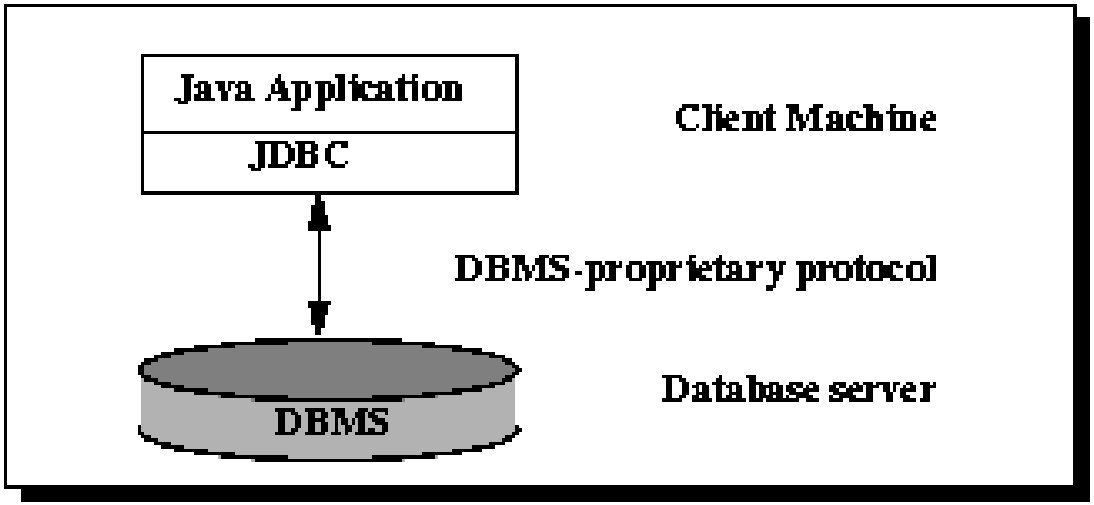
Introduction

- JDBC is a common database programming API that can access any kind of tabular data (i.e., data stored in Relational Databases)
- It is often referred as *"Java Database Connectivity"*, however it is not an acronym (It is just an API)
- With the help of JDBC API we can **C**reate, **U**pdate, **D**elete and **E**xecute (CRUD) records



Introduction

- JDBC Architecture
  - The JDBC API supports 2 tier and 3 tier architecture models for accessing the relational databases
  - In tier-2 model, java application directly communicates with the database



Introduction

- JDBC Architecture
  - In a 3-tier architecture, commands will be sent to *middle-tier* services, which then sends the request to the database source

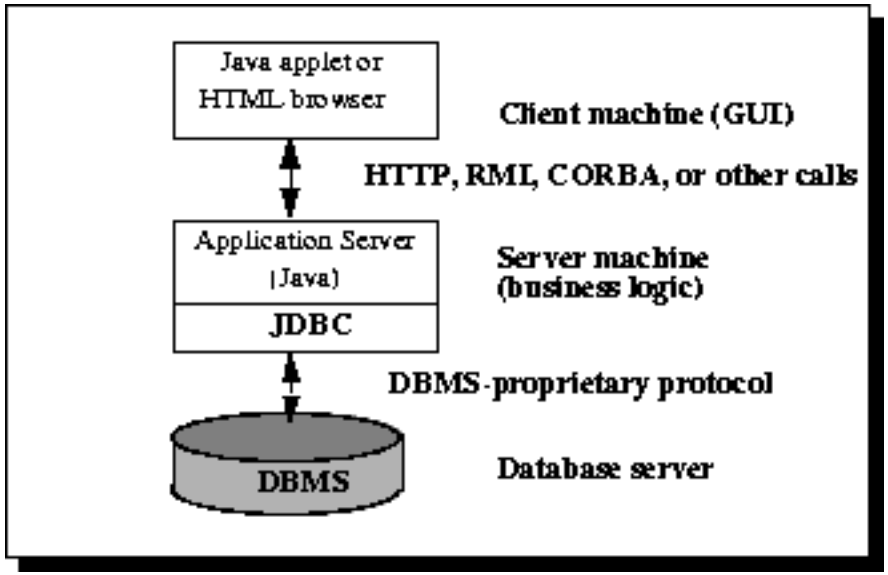


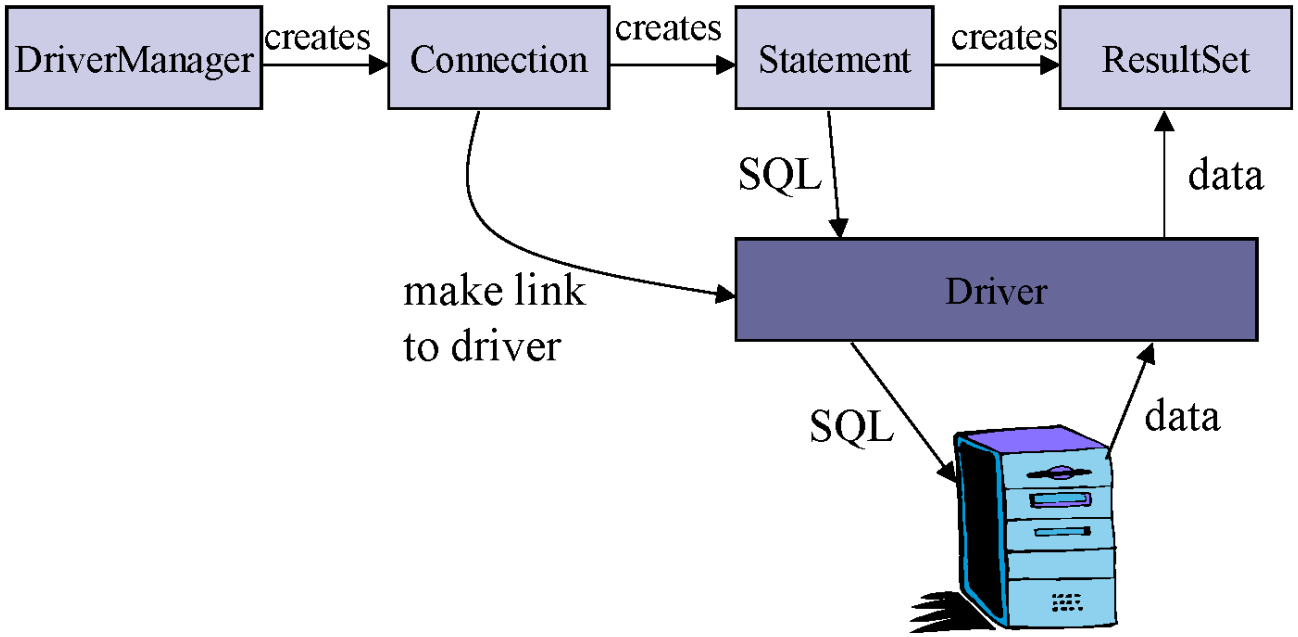
*Image taken from
https://docs.oracle.com/javase/tutorial/jdbc/overview/index.html

Introduction

- **JDBC Components**
  - The JDBC API includes the following components
    - **I.** **DriverManager**: This class manages the JDBC drivers. It manages the connection established between the underlying java application and the relational database
    - **II.** **Drivers**: It is a software component that enables different java applications to interact with the database. The *DriverManager* class is responsible for managing JDBC Drivers
    - **III.** **Connection**: It is an *interface* that contain all the methods for establishing a connection with the database. All communication with the database will go through Connection interface
    - **IV.** **Statement**: This interface contain methods to execute queries with the database
    - **V.** **ResultSet**: The object of *ResultSet* class holds the data that is retrieved from the *Statement* interface. It acts as an *iterator* that points to the first row of the table
    - **VI.** **SQL Exceptions:** This class handles exceptions generated while interacting with the database

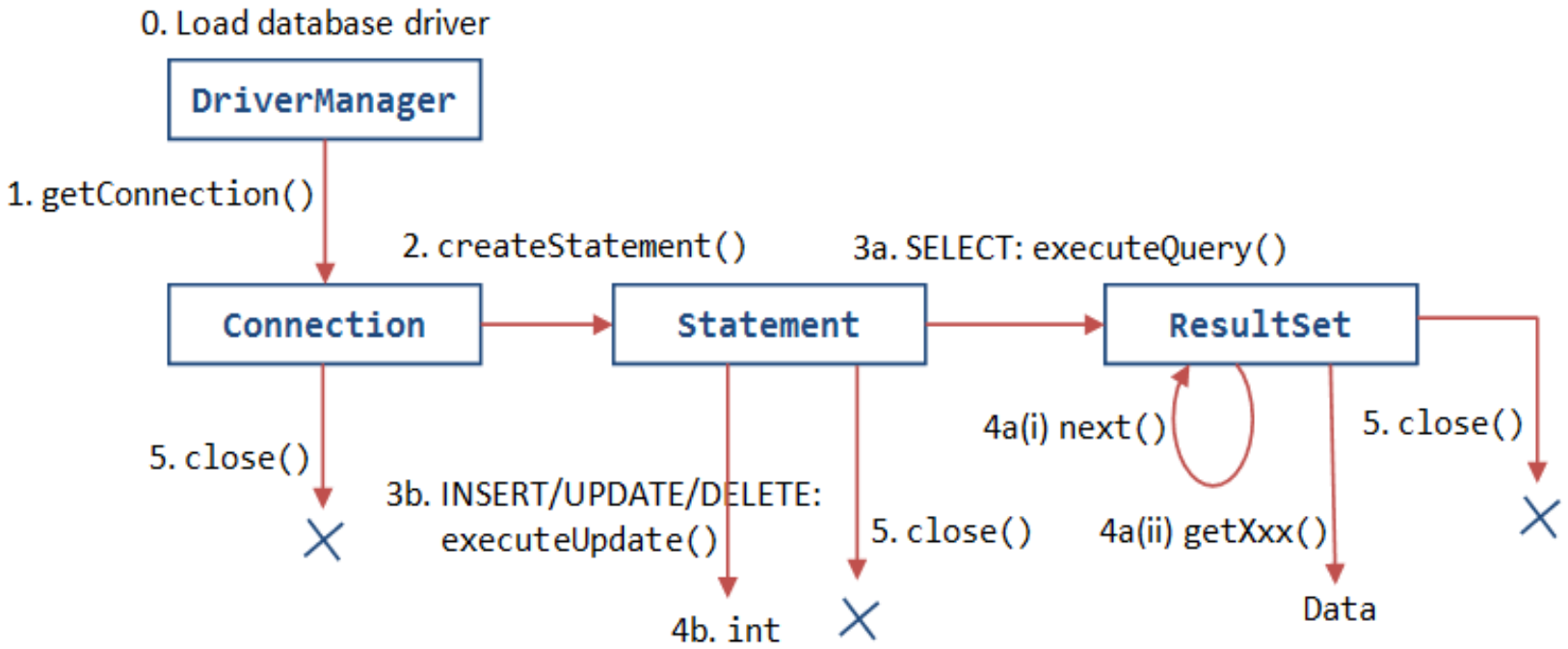
Introduction

- JDBC Components cont.…



Introduction

- Steps



Establishing Connection

- Following 6 steps are involved for establishing a database connection:

1) **Importing the necessary packages**: Import the *sql* package
2) **Register the JDBC driver class**: using *forName()* method of the Class class
3) **Open a Connection**: using *getConnection()* method of the DriverManager class
4) **Create a Statement**: using *createStatement()* method of the Statement interface
5) **Execute a Query**: using *executeQuery()* method of the Statement interface which returns an object of ResultSet class
6) **Clean up the Environment**: using *close()* method of the Connection interface

Establishing Connection

- Steps:
  - Set up a database server on your computer (*Oracle, MySql, PostgreSql, MS Access* etc.)
  - Get a JDBC driver for the installed database
    - *The latest JDK makes most Open Database Connectivity (ODBC) driver available to users through the JDBC API*
  - Import the library
    - `Import java.sql.*;`
  - Specify the URL of the database server
    - `String URL = "jdbc:mysql://localhost/DB_Name";` //case of *MySql* Database

Establishing Connection

Register the Driver	Register the JDBC driver class • Class.forName("com.mysql.cj.jdbc.Driver"); //mysql driver
Open DB Connection	Open a database connection • Connection con = DriverManager.getConnection(URL,"Username","Password");
Create Statement	Create a Statement • Statement stmt = con.createStatement();

Establishing Connection

Execute Query	Execute SQL Query • *String query = "Select id, first, last, age FROM Employees";* • *ResultSet rs = stmt.executeQuery(query);*

Process the Result	`While(rs.next()) {`
	`int id = rs.getInt("id");`
	`String first = rs.getString("first");`
	`String last = rs.getString("last");`
	`int age = rs.getInt("age");`
	`}`

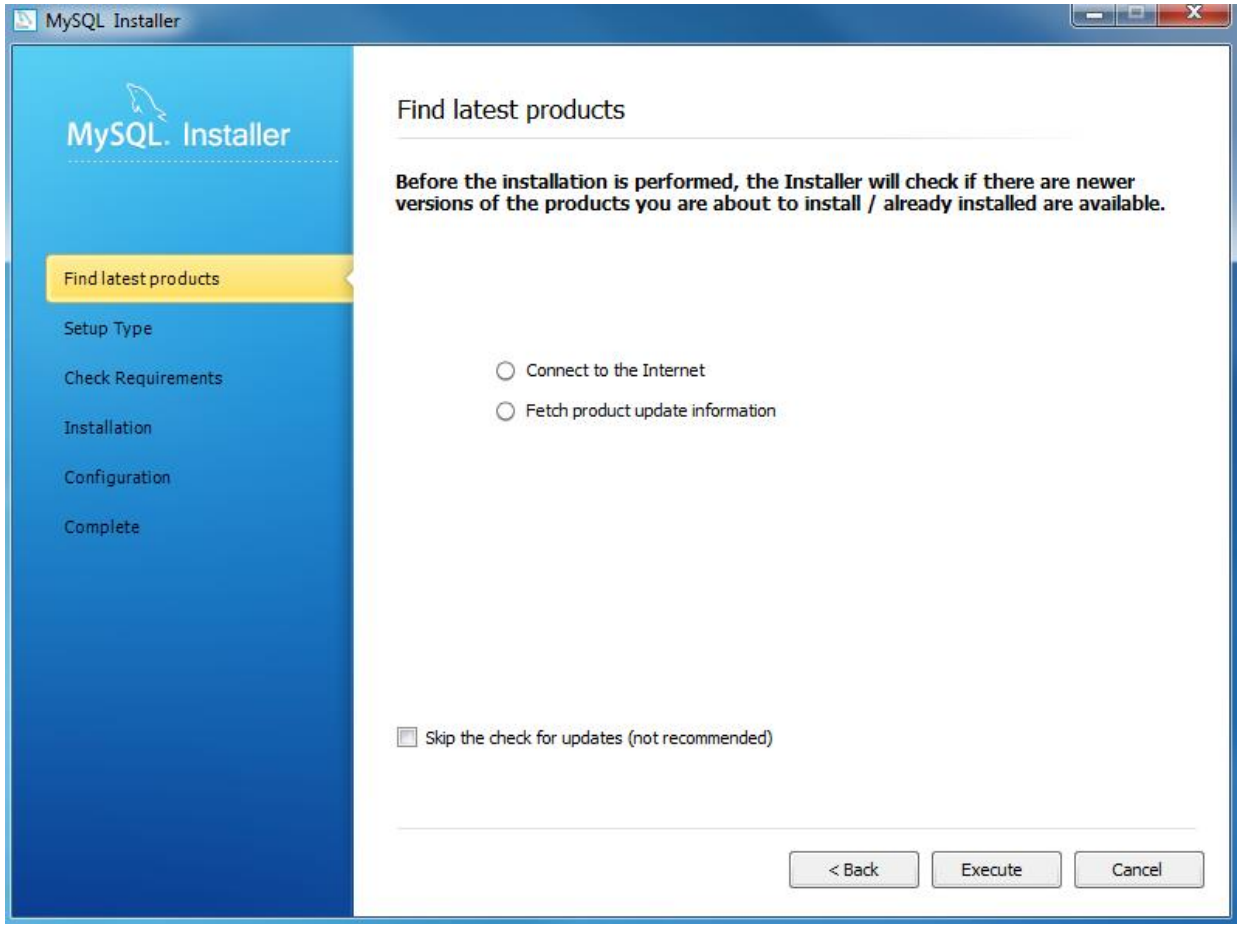
Close the Connection	Clean up the environment `rs.close(); | stmt.close(); | con.close();`

JDBC Demo – MySQL Installation

- Download any open source relational Database software:
  - To download *MySql,* follow the given link: https://dev.mysql.com/downloads/
  - To install MySQL using the MySQL installer, double-click on the MySQL installer file and follow the steps below:
    - Install MySQL Step 1: Windows configures MySQL Installer.. Click on Install MySQL Products

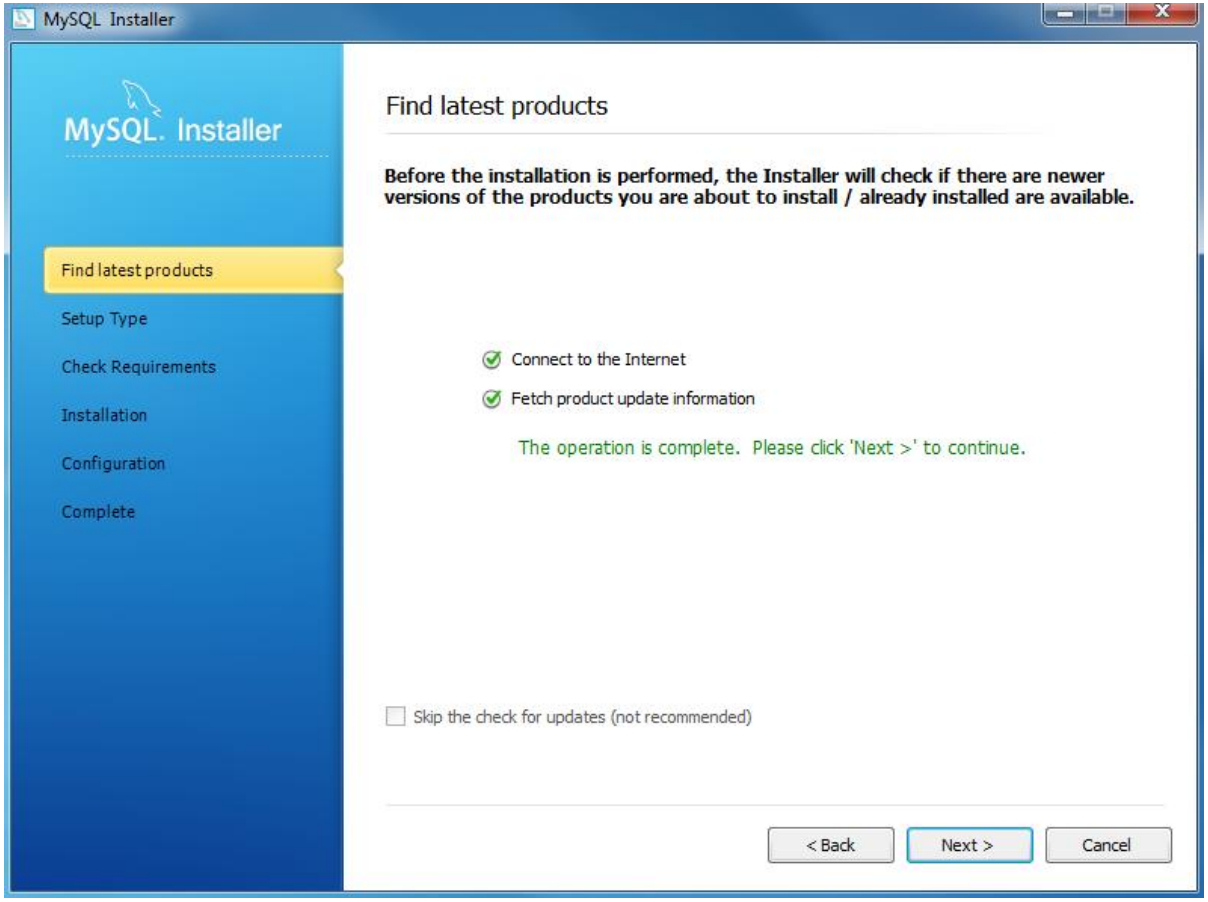
MySQL Installation

- Install MySQL Step 2 – Download the latest MySQL products: MySQL installer checks and downloads the latest MySQL products including MySQL server, MySQL Workbench, etc



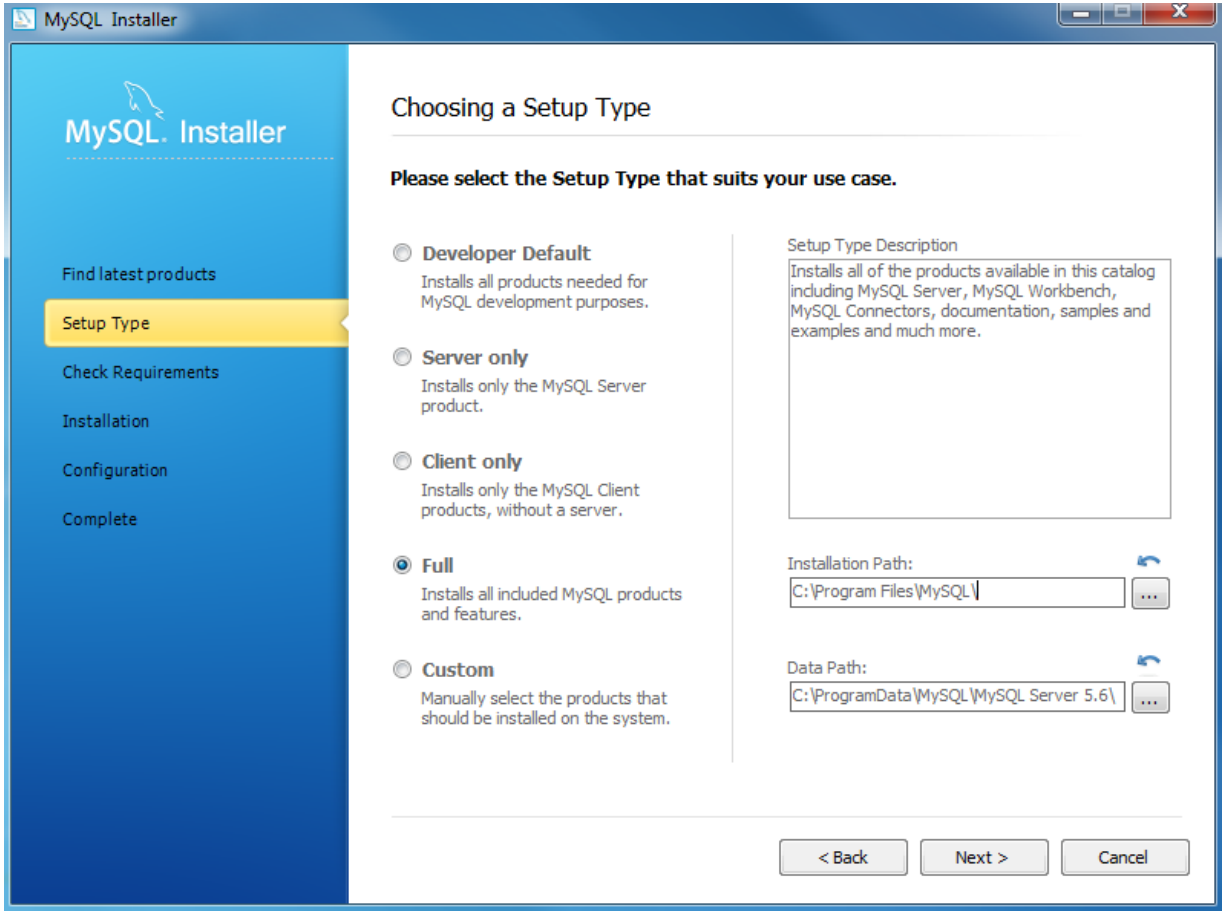
MySQL Installation

- Install MySQL Step 3 – Download the latest MySQL products: MySQL installer checks and downloads the latest MySQL products including MySQL server, MySQL Workbench, etc.



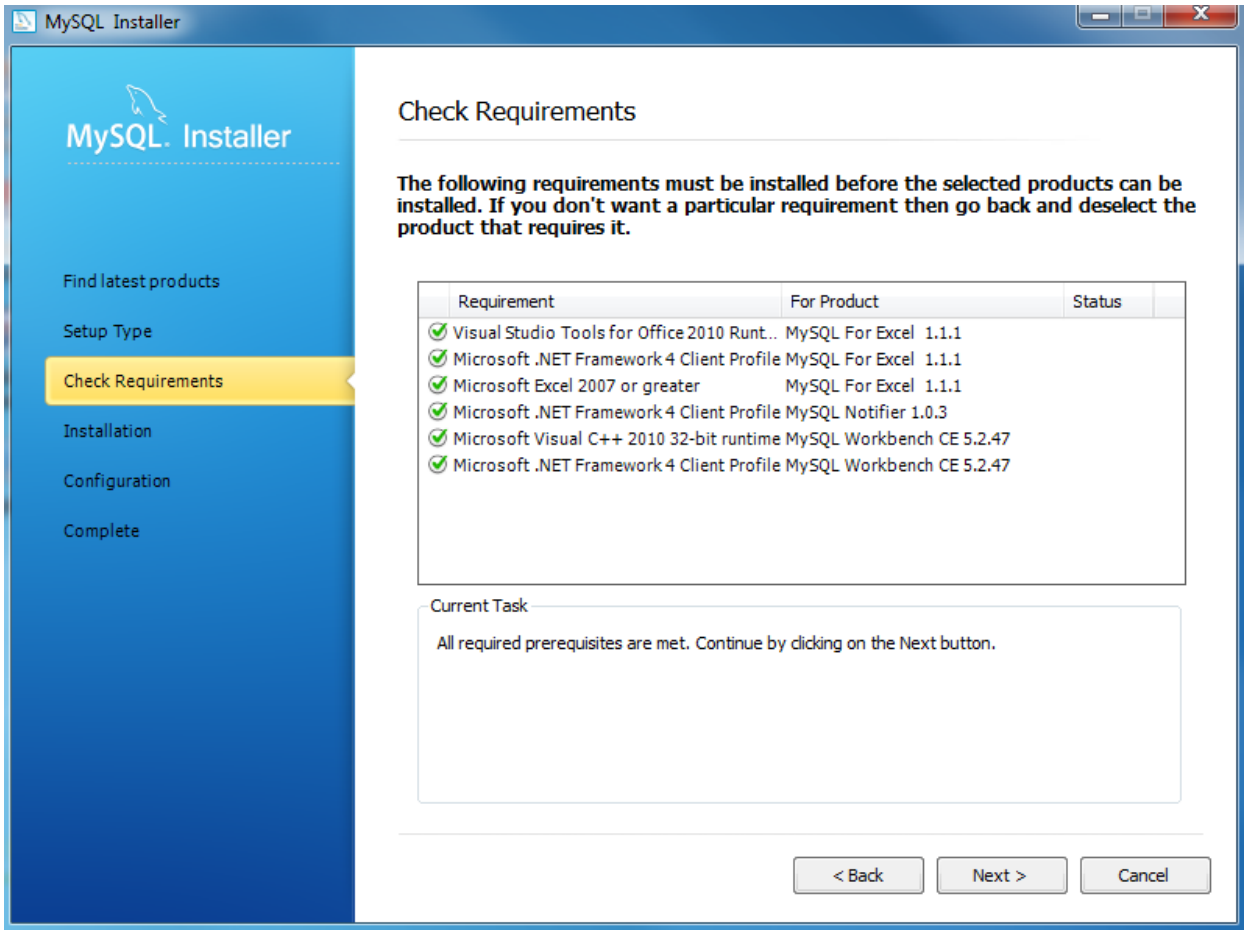
MySQL Installation

- Install MySQL Step 5 – Choosing a Setup Type: there are several setup types available. Choose the Full option to install all MySQL products and features.



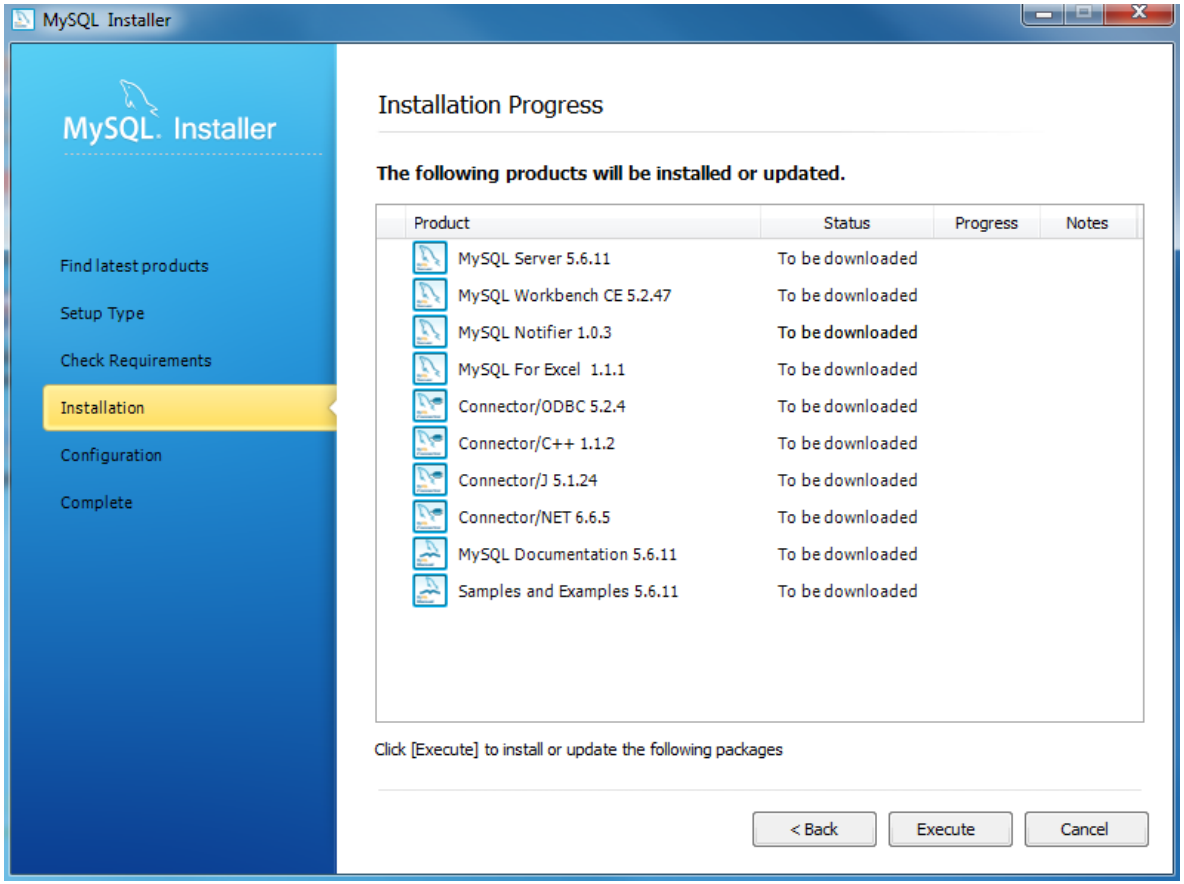
MySQL Installation

- Install MySQL Step 6 – Checking Requirements



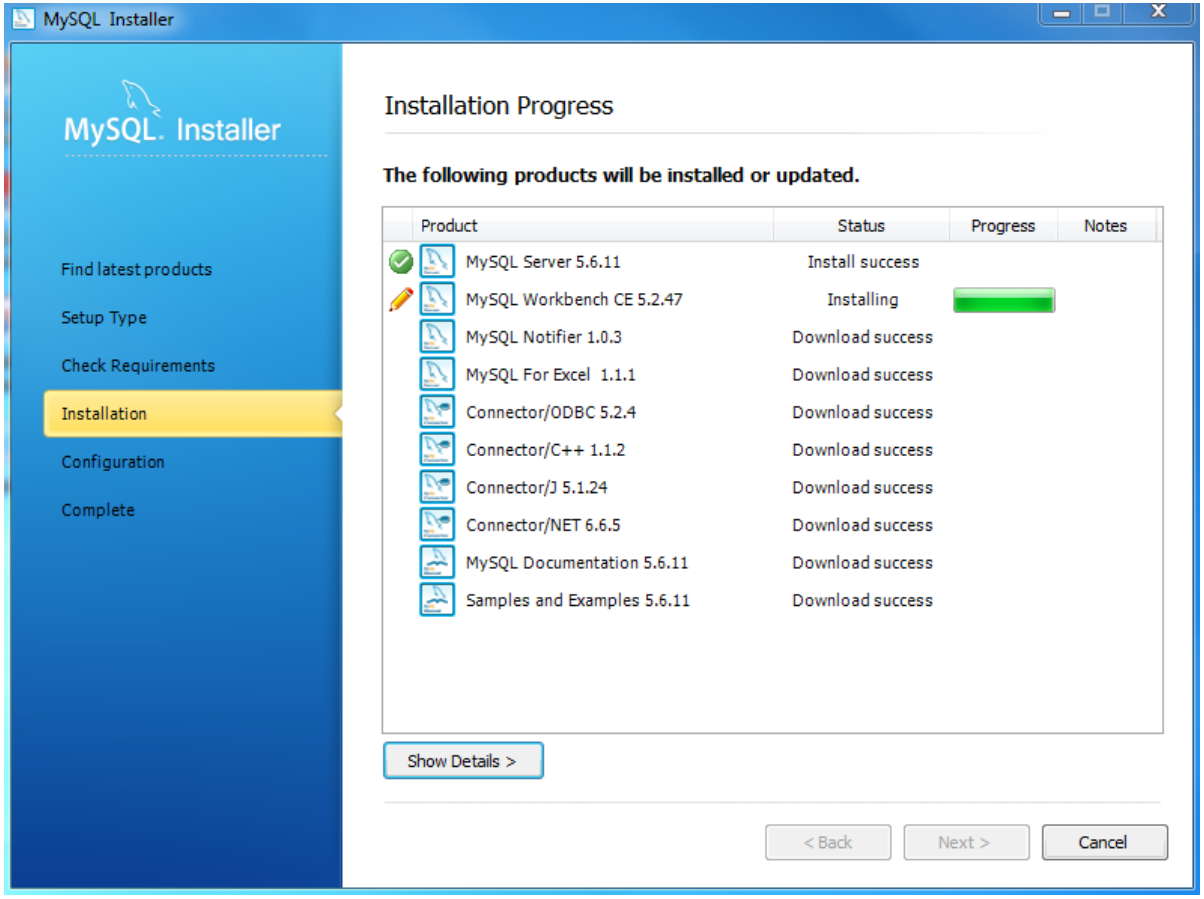
MySQL Installation

- Install MySQL Step 7 – Installation Progress: MySQL Installer downloads all selected products. It will take a while, depending on which products you selected and the speed of your internet connection



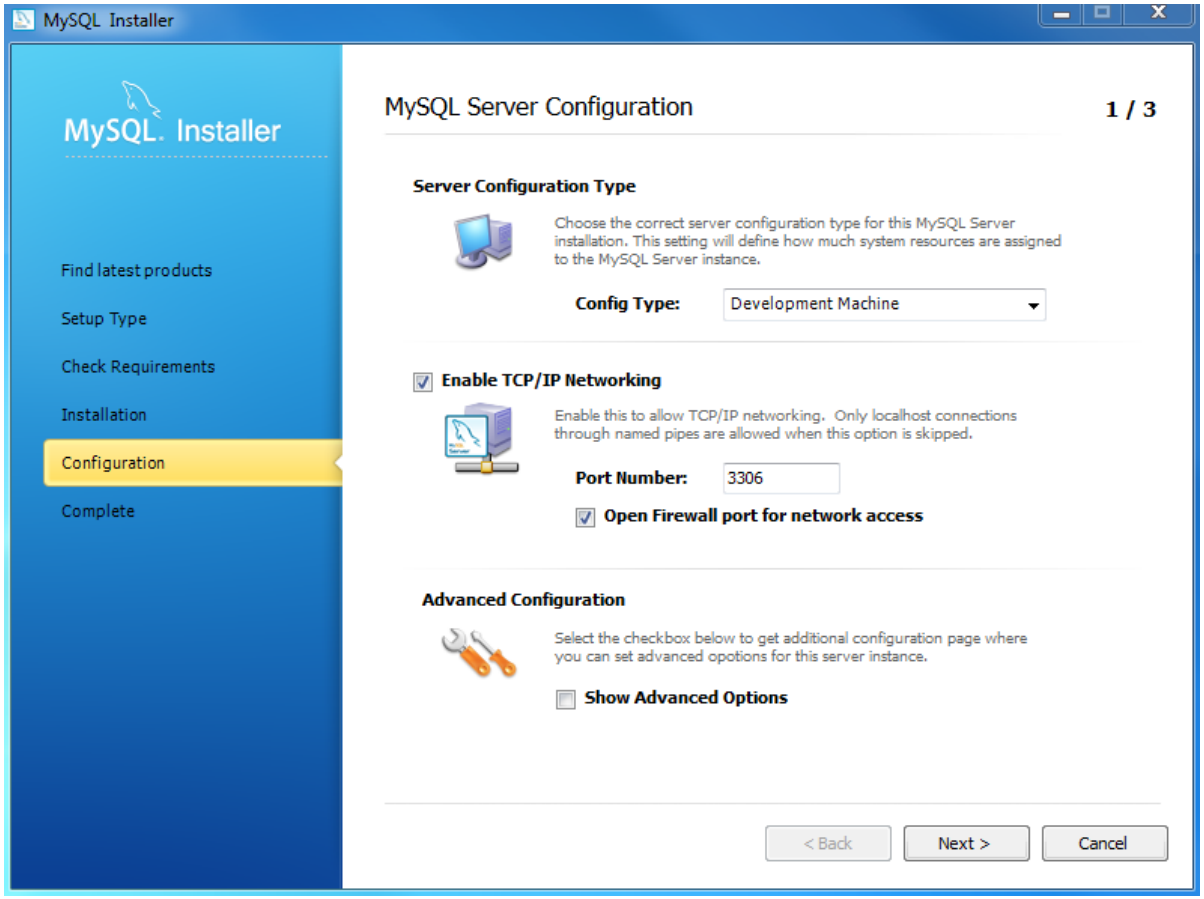
MySQL Installation

- Install MySQL Step 7 – Installation Progress: downloading Products in progress… Click Next to continue



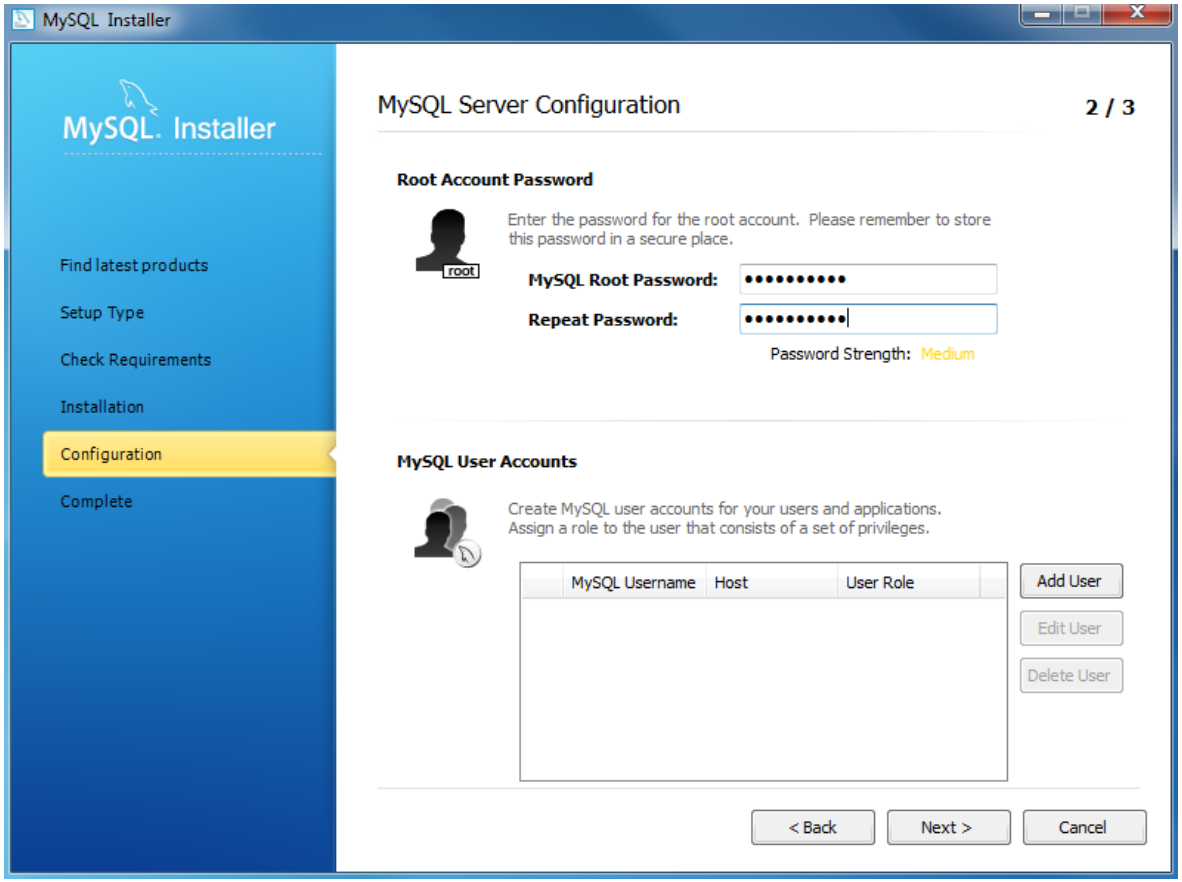
MySQL Installation

- Install MySQL Step 8 – MySQL Server Configuration: choose Config Type and MySQL port (3006 by default) and click Next button to continue.



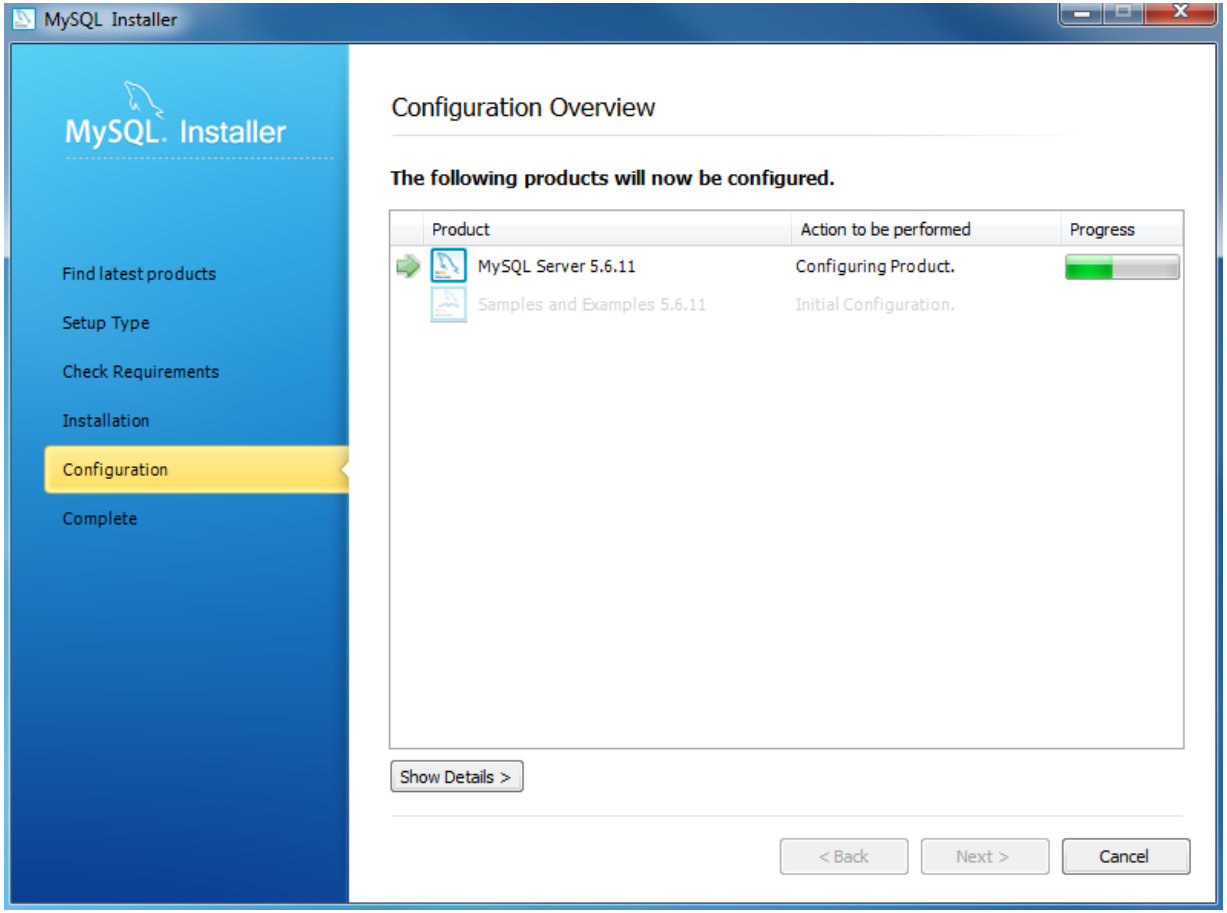
MySQL Installation

- Install MySQL Step 8.1 – MySQL Server Configuration: choose a password for the root account. Please note the password download and keep it securely if you are installing MySQL database server on a production server. If you want to add a more MySQL user, you can do it in this step.



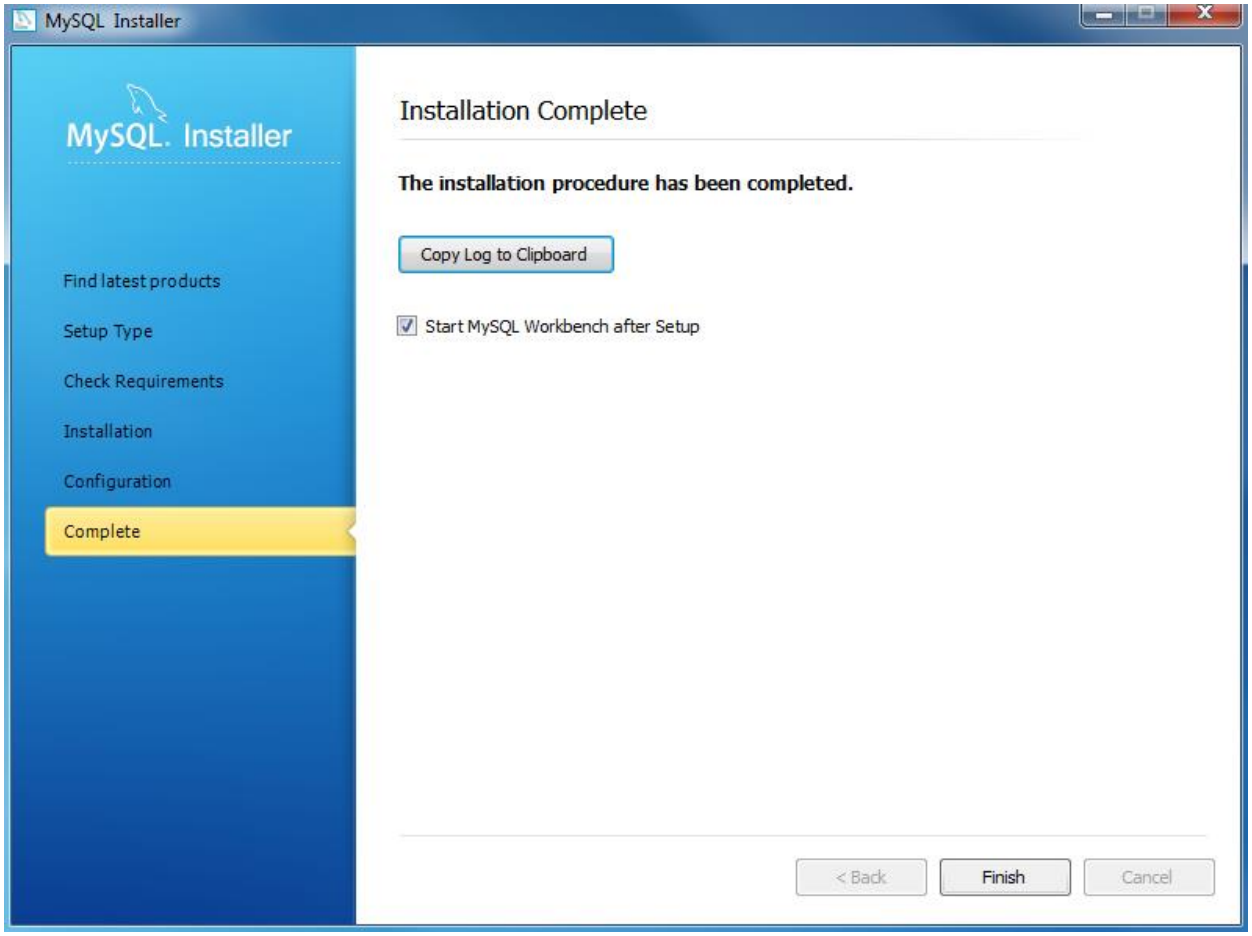
MySQL Installation

- Install MySQL Step 8.1 – MySQL Server Configuration – In Progress: MySQL Installer is configuring MySQL database server. Wait until it is done and click the Next button to continue.



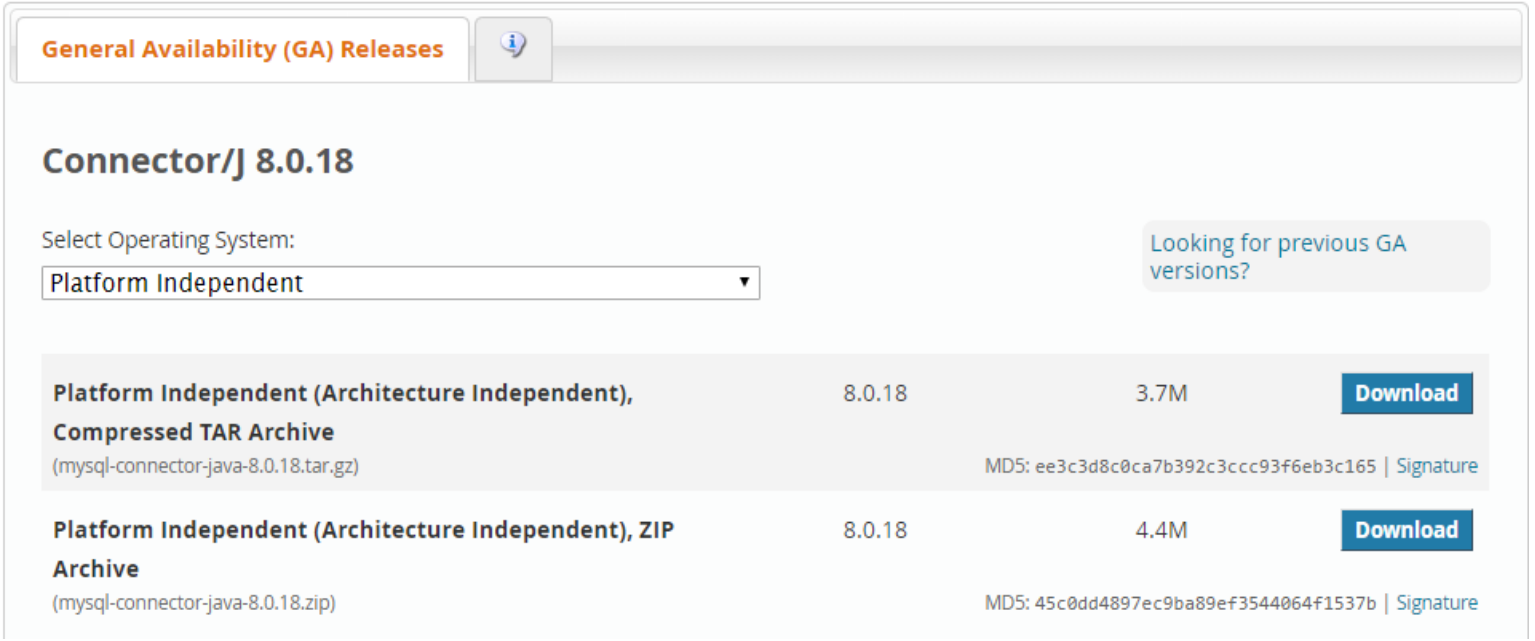
MySQL Installation

- Install MySQL Step 9 – Installation Completes: the installation completes. Click the Finish button to close the installation wizard and launch the MySQL Workbench



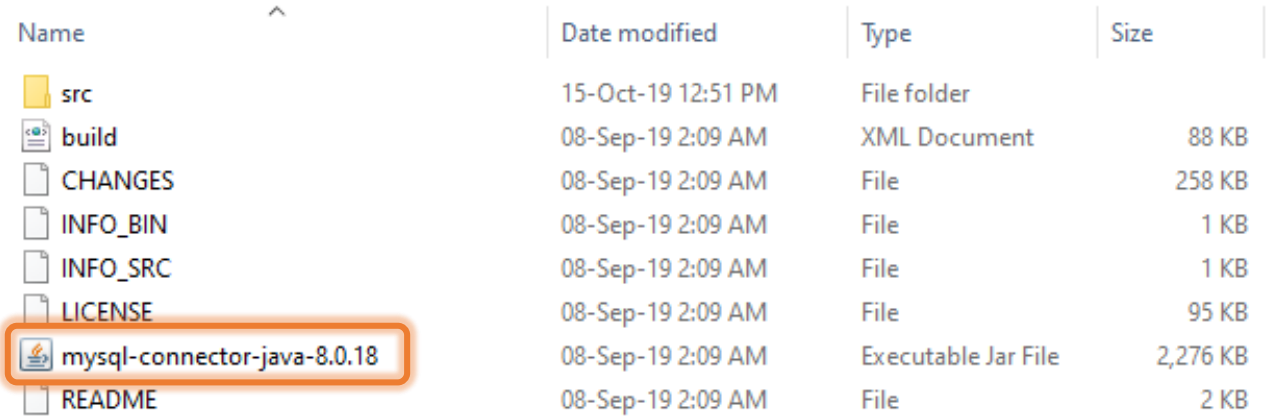
Download MySQL Connector

- Download *MySql* connector from: https://dev.mysql.com/downloads/connector/j/
  - Select Platform Independent → Download the zip file



JDBC Demo - Eclipse

- Extract the downloaded zip file

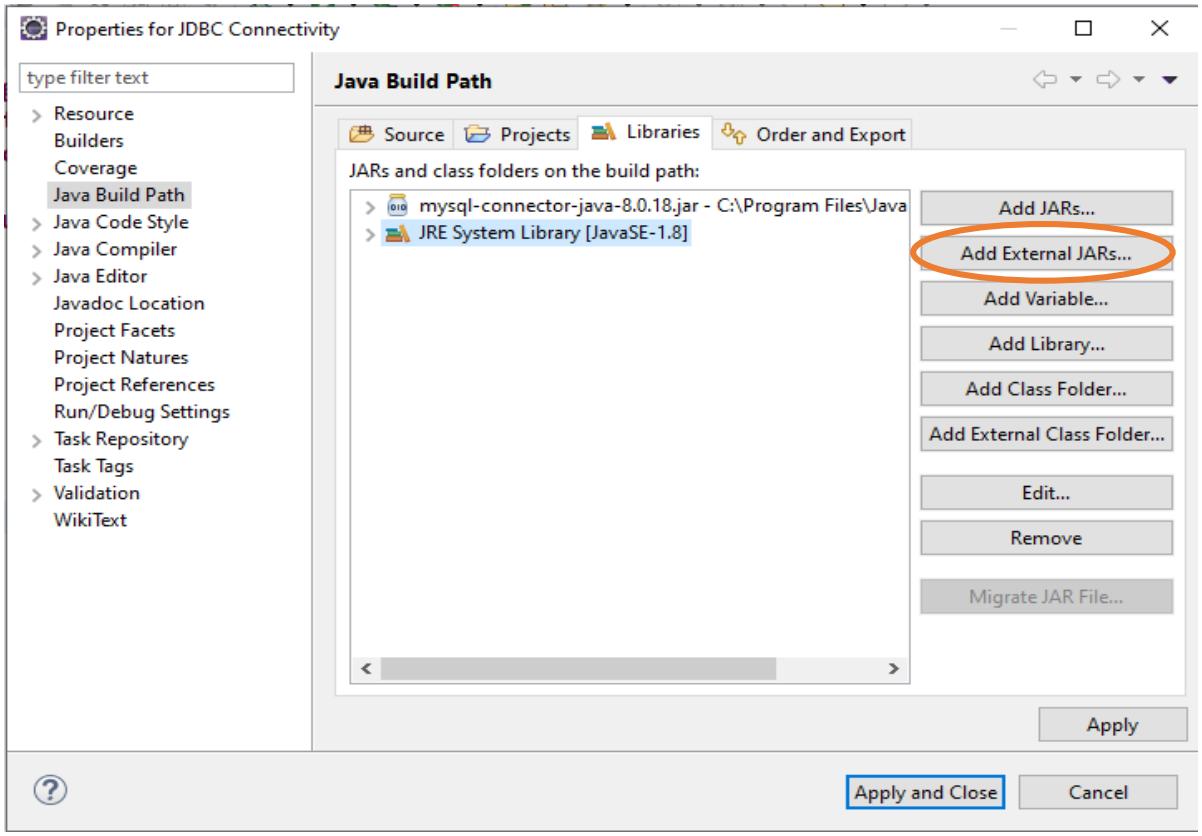


Name	Date modified	Type	Size
src	15-Oct-19 12:51 PM	File folder	
build	08-Sep-19 2:09 AM	XML Document	88 KB
CHANGES	08-Sep-19 2:09 AM	File	258 KB
INFO_BIN	08-Sep-19 2:09 AM	File	1 KB
INFO_SRC	08-Sep-19 2:09 AM	File	1 KB
LICENSE	08-Sep-19 2:09 AM	File	95 KB
mysql-connector-java-8.0.18	08-Sep-19 2:09 AM	Executable Jar File	2,276 KB
README	08-Sep-19 2:09 AM	File	2 KB

- Next step is to add the downloaded MySql connector (JAR) in Eclipse Project library
  - Right click on the Java Project and select "*Build Path" → "Configure Build Path"*
    - A Property window opens up

JDBC Demo - Eclipse

- Select "*Add External JARs*" and add path of the downloaded MySql Connector (JAR)



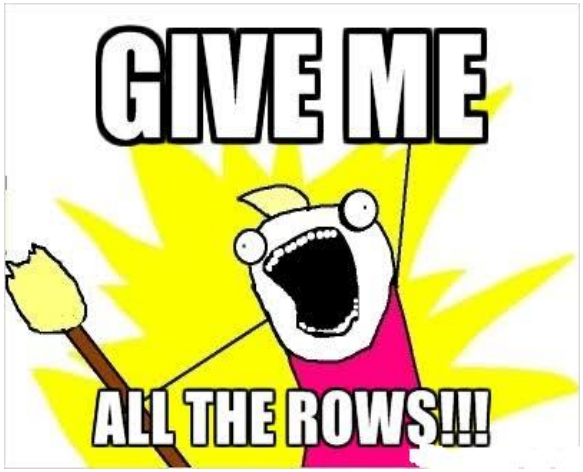
JDBC Demo

```
import java.sql.*;
Public class DemoExample
{
        public static void main(String [] args)
        {
                try {
                        // Step 1: Register the driver
                        Class.forName("com.mysql.cj.jdbc.Driver");
                        String URL = "jdbc:mysql://localhost/EMP";
                        String DB_User = "John";
                        String DB_Pass = "john";

                        // Step 2: Open a database connection
                        Connection con = DriverManager.getConnection(URL, DB_User, DB_Pass);

                        // Step 3: Create a statement
                        Statement stmt = con.createStatement();
                        String query = "Select Emp_id, Emp_Name, Emp_Age FROM Employees";
```

JDBC Demo

```
// Step 4: Execute a query
ResultSet rs = stmt.executeQuery(query);

// Step 5: Extract data from ResultSet
While(rs.next()) {
        int id = rs.getInt("Emp_Id");
        String name = rs.getString("Emp_Name");
        int age = rs.getInt("Emp_Age");
        //Display values
        System.out.println("Emp ID: "+id);
        System.out.println("Emp Name: "+name);
        System.out.println("Emp Age: "+age);
```

Returns object of type ResultSet that is used to walk through the query result, one row at a time.



JDBC Demo

```
                        // Step 6: Cleaning up the environment
                        rs.close();
                        stmt.close();
                        con.close();
                }
                catch(ClassNotFoundException e1)
                {
                        System.out.println(e1);
                }
                catch(SQLException e2)
                {
                        System.out.println(e2);
                }
        }
}
```

Database Table

	Emp_Id	Emp_Name	Emp_Age
▶	10	John	54
	11	Maria	27
	12	Ram	32
	13	Ryan	43
*	NULL	NULL	NULL

NCU
THE NORTHCAP UNIVERSITY
NAAC ACCREDITED
(Formerly ITM University, Gurugram)

Thank You