**Part – I - Installing Apache Spark, Cassandra and Connector for Streaming Data**

In this document I am explaining how to set up a standalone SPARK and Cassandra Instance on Windows 10 and enable a connection using SPARK-CASSANDRA connector for transferring data from SPARK to Cassandra. Since apache spark works well with Ubuntu and Linux systems, the installation and configuration should be approached differently in windows. Versions of each tool is **VERY** important for enabling a trouble free connectivity.

**Tools/Software Used**

OS – Windows 10

Below are the most stable version I find suitable for this set up.

* Apache Spark 1.6
* Cassandra 2.2.8
* Java (JDK 7) 7 version

The version matrix is given below

| **Connector** | **Spark** | **Cassandra** | **Cassandra Java Driver** | **Minimum Java Version** | **Supported Scala Versions** |
| --- | --- | --- | --- | --- | --- |
| 2.0 | 2.0, 2.1 | 2.1.5\*, 2.2, 3.0 | 3.0 | 8 | 2.10, 2.11 |
| 1.6 | 1.6 | 2.1.5\*, 2.2, 3.0 | 3.0 | 7 | 2.10, 2.11 |
| 1.5 | 1.5, 1.6 | 2.1.5\*, 2.2, 3.0 | 3.0 | 7 | 2.10, 2.11 |
| 1.4 | 1.4 | 2.1.5\* | 2.1 | 7 | 2.10, 2.11 |
| 1.3 | 1.3 | 2.1.5\* | 2.1 | 7 | 2.10, 2.11 |
| 1.2 | 1.2 | 2.1, 2.0 | 2.1 | 7 | 2.10, 2.11 |
| 1.1 | 1.1, 1.0 | 2.1, 2.0 | 2.1 | 7 | 2.10, 2.11 |
| 1.0 | 1.0, 0.9 | 2.0 | 2.0 | 7 | 2.10, 2.11 |

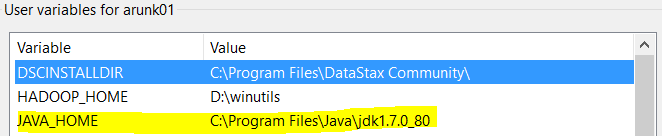
\**Compatible with 2.1.X where X >= 5*

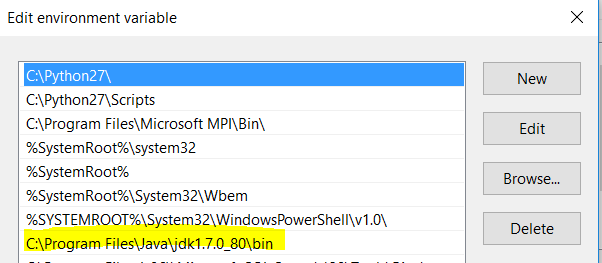
**Step 1 – Installing Prerequisites**

First up we'll need Java installed. Oracle Java 7 is the most stable version to use in this setup. Install JDK from here –

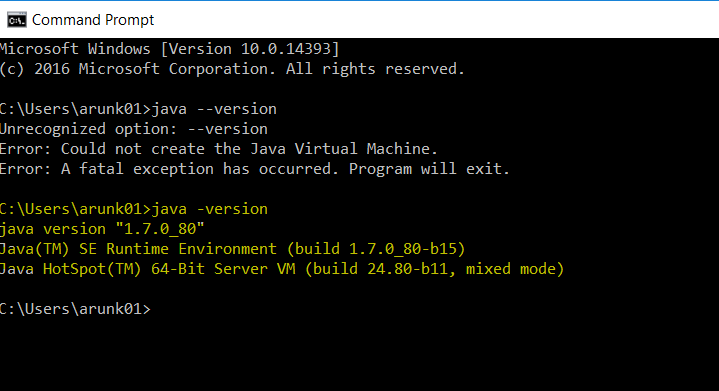
<http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase7-521261.html#jdk-7u80-oth-JPR>

Follow the default path of Installation and create environment variables as shown below –





Open command prompt and check for JAVA version at the root and you should see the below before proceeding.

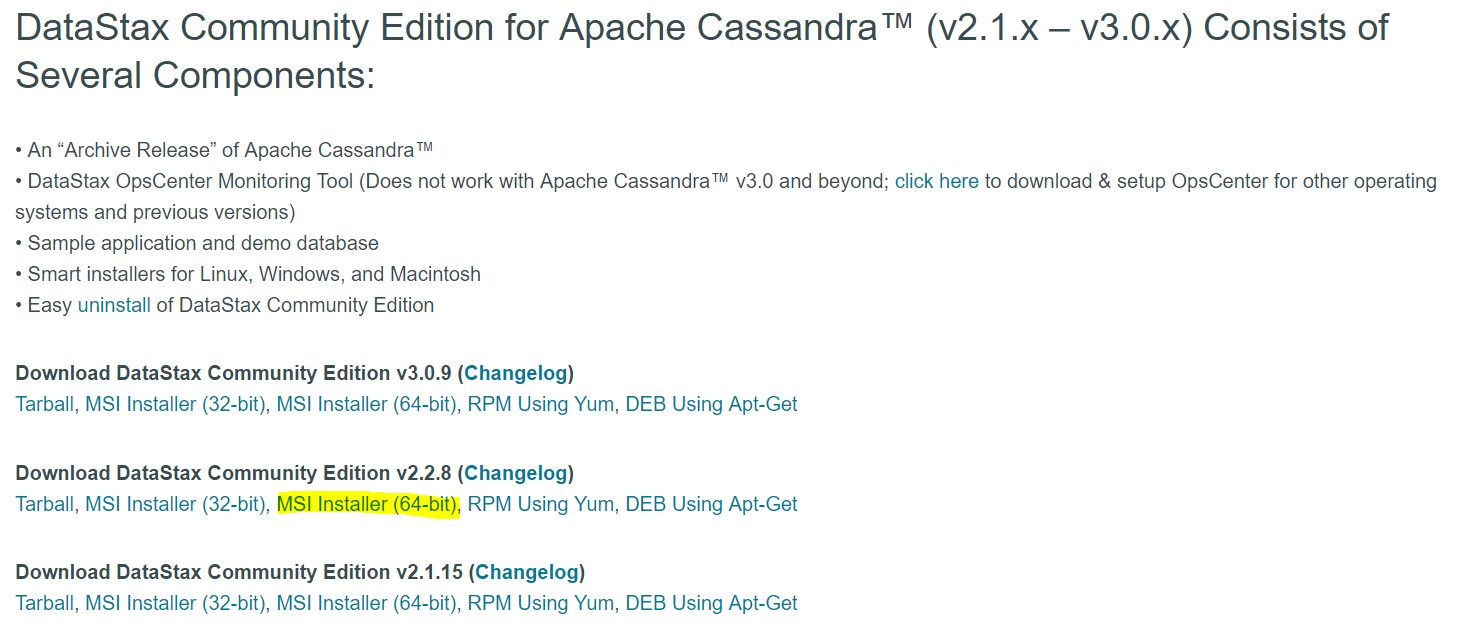


**Step - 2 Installing Cassandra**

There are many ways of doing this, but the easiest way I find is to install Apache Cassandra Community edition that is pre built for Windows and comes with a OPS Center for monitoring.

Go here to download - <https://academy.datastax.com/planet-cassandra//cassandra>

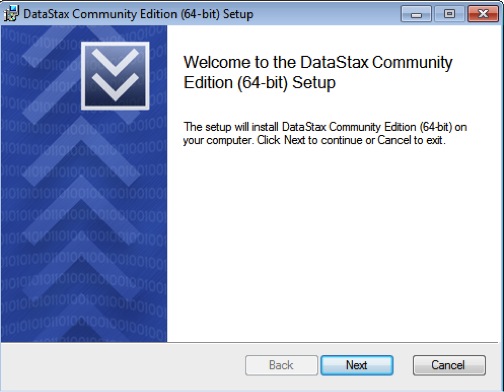
Select the highlighted one below for installation –



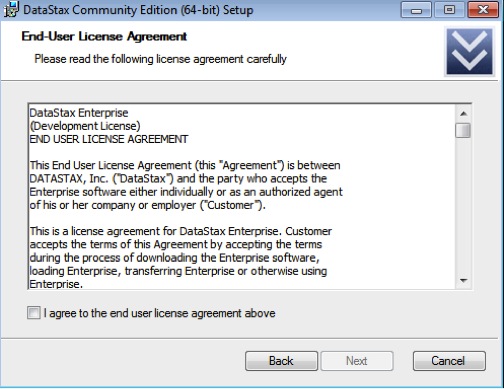
Installation steps as below –

**Using the Windows Installer**

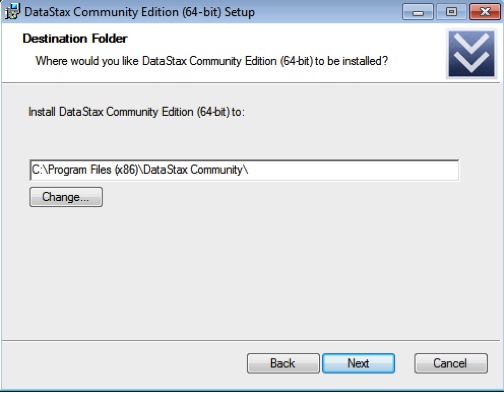
The Windows installer is an MSI package that is run like any other on Windows. When you execute the setup program, you are first presented with an initial welcome panel that identifies your installation package:



Clicking next takes you to the end user license agreement

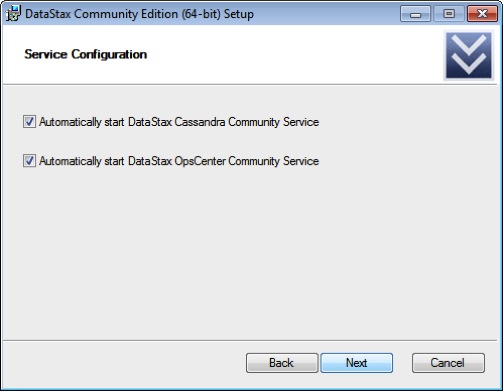


The next panel allows you to specify where the software is to be installed:

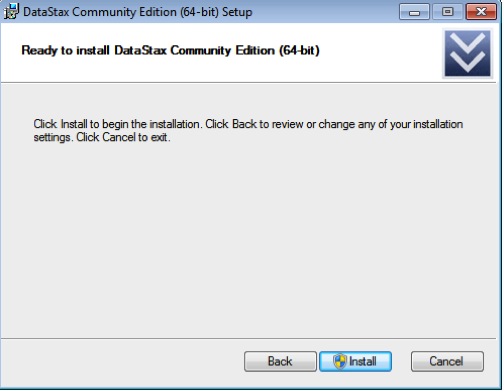


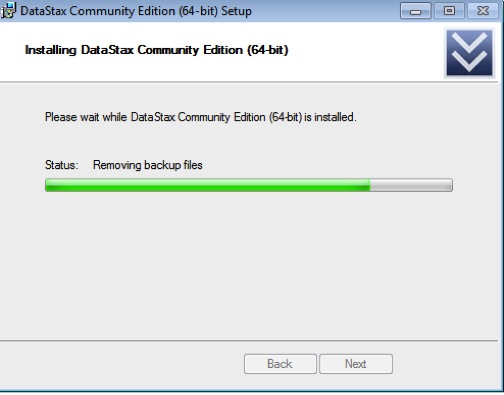
Once the installation directory has been set, the installer will ask how you want to handle the services that will be installed. Installed services include:

* The Apache Cassandra server, labeled DataStax Cassandra Community Service
* The DataStax OpsCenter management tool, which includes both the OpsCenter service used to manage and monitor Cassandra, and the OpsCenter agent, which is used to gather metrics and carry out various tasks on every Cassandra node. Note that the primary OpsCenter service and agent are actually broken out into two distinct services on Windows

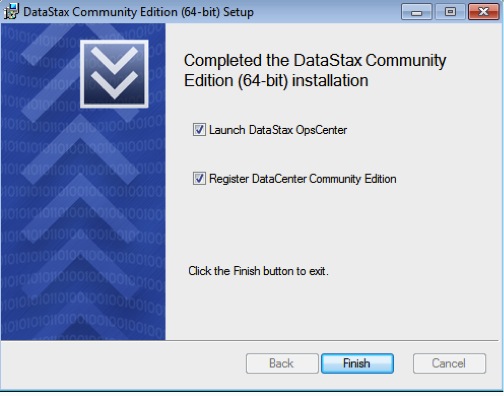


The next panel initiates the installation process:

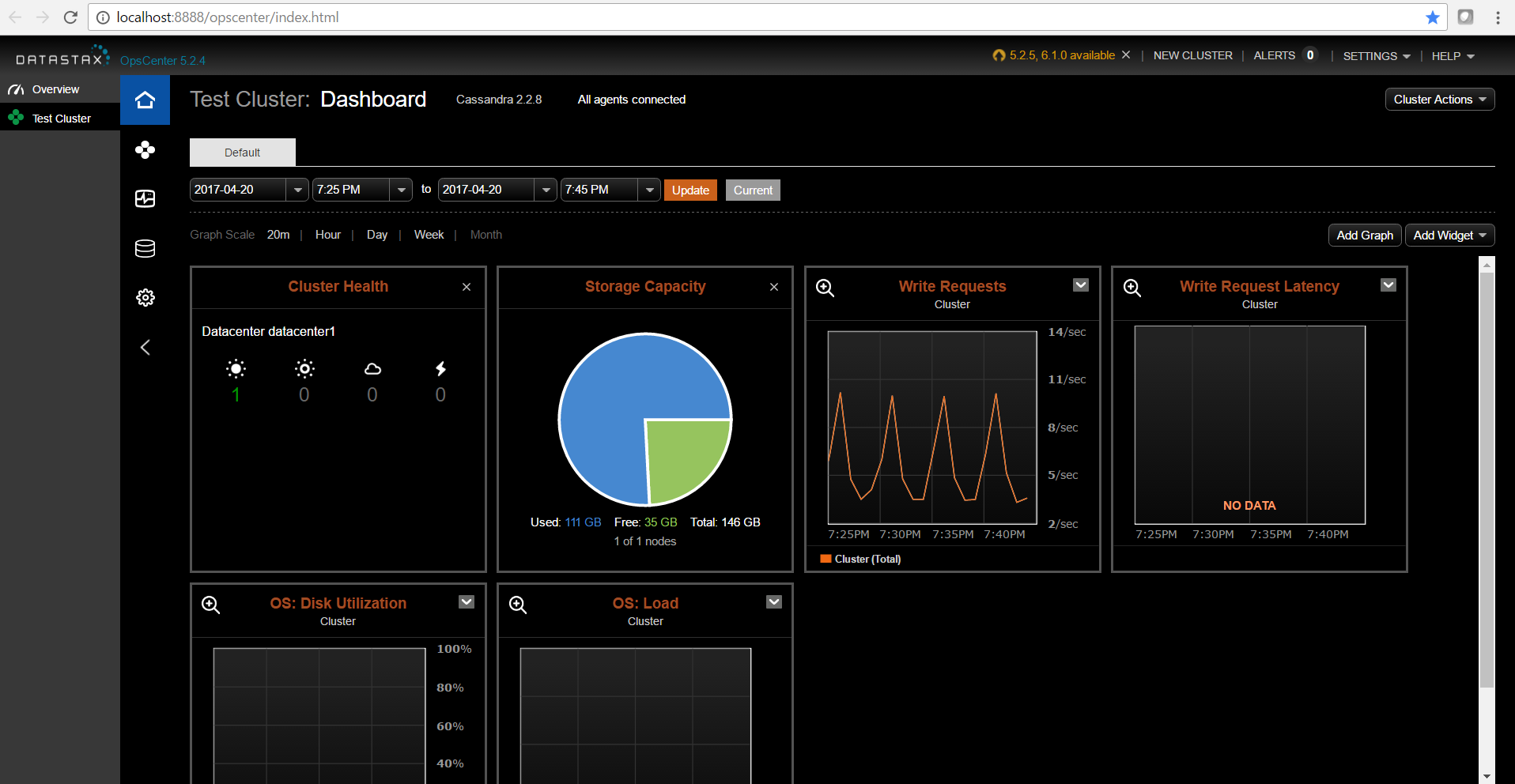




The final panel asks if you would like to launch DataStax OpsCenter in your browser and also register to be updated when new versions of the software become available:



If you choose to execute DataStax OpsCenter (and you have either Google Chrome or Firefox as your default web browser), you’ll be presented with the OpsCenter dashboard as shown below:



Once installation of Cassandra is complete, we have to test it using CQLSH (command line utility). Go to windows programs and open ‘Cassandra CQL Shell’. Create the following in the CQL and verify using the following SQL commands –

create keyspace hr with replication={'class':'SimpleStrategy', 'replication\_factor':1};

Once a keyspace is created, you can create column families (the primary data object in Cassandra), insert data, query data, and more:

use hr;

create table emp

         ... (empid int primary key,

         ... emp\_first varchar,

         ... emp\_last varchar,

         ... emp\_dept varchar);

insert into emp (empid, emp\_first, emp\_last, emp\_dept)

         ... values (1, 'Fred', 'Smith', 'HR');

insert into emp (empid, emp\_first, emp\_last, emp\_dept)

         ... values (2, 'Mary', 'Taylor', 'HR');

insert into emp (empid, emp\_first, emp\_last, emp\_dept)

         ... values (3, 'Bob', 'Jackson', 'Eng');

select \* from emp;

 empid | emp\_dept | emp\_first | emp\_last

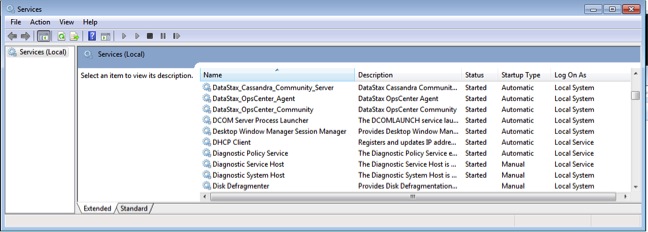
-------+----------+-----------+----------

     1 |       HR |      Fred |    Smith

     2 |       HR |      Mary |   Taylor

     3 |      Eng |       Bob |  Jackson

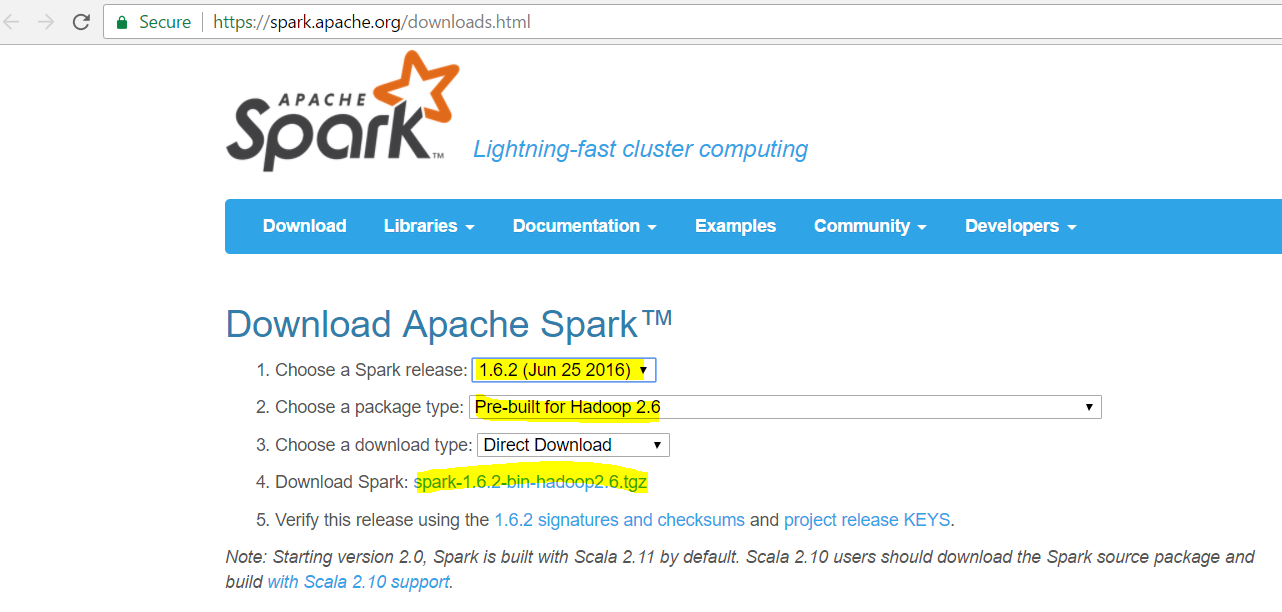
Lastly, you can start, stop, and manage the various services installed via the standard Windows services control panel interface:



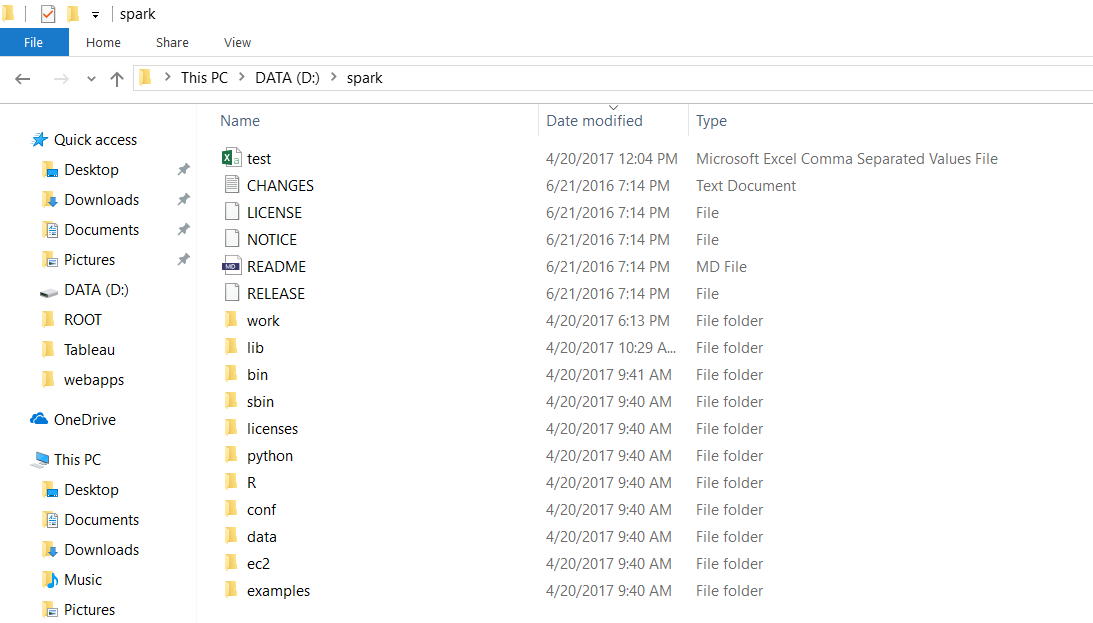
**Step – 3 – Installing SPARK**

Download SPARK from here - <https://spark.apache.org/downloads.html>

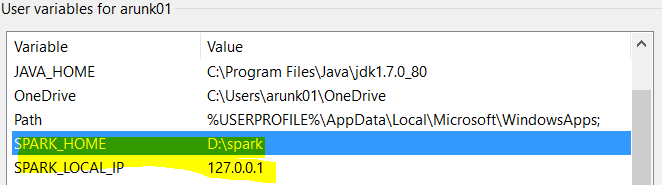
Follow the installation as shown in the below screen shots



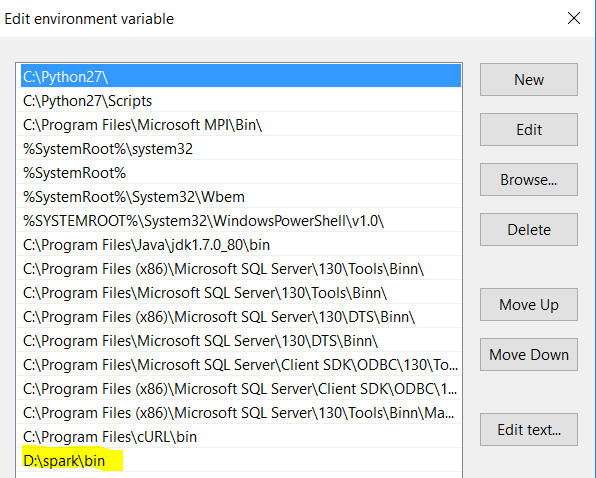
Decompress it using 7-zip and keep it in the folder D:\spark for easy access –



After installing spark, environment variables need to be setup on the computer as shown in below screen shot–



\*SPARK\_LOCAL\_IP set up is optional if the installation is standalone on a single PC.



You can move the SPARK folder to wherever you like. Now let's test it out. Open up a spark shell on command prompt by doing a cd into your spark directory and then:

bin/spark-shell

This will take a few seconds and there will be a lot of log output. You'll eventually get presented with a Scala prompt. Now let's get spark to do a calculation for us:

SCALA > sc.parallelize( 1 to 50 ).sum()

This will eventually output the result 1275.

When you run for the first time there will be couple of warnings, exceptions and errors. To avoid those exceptions, warnings and errors following should be set up.

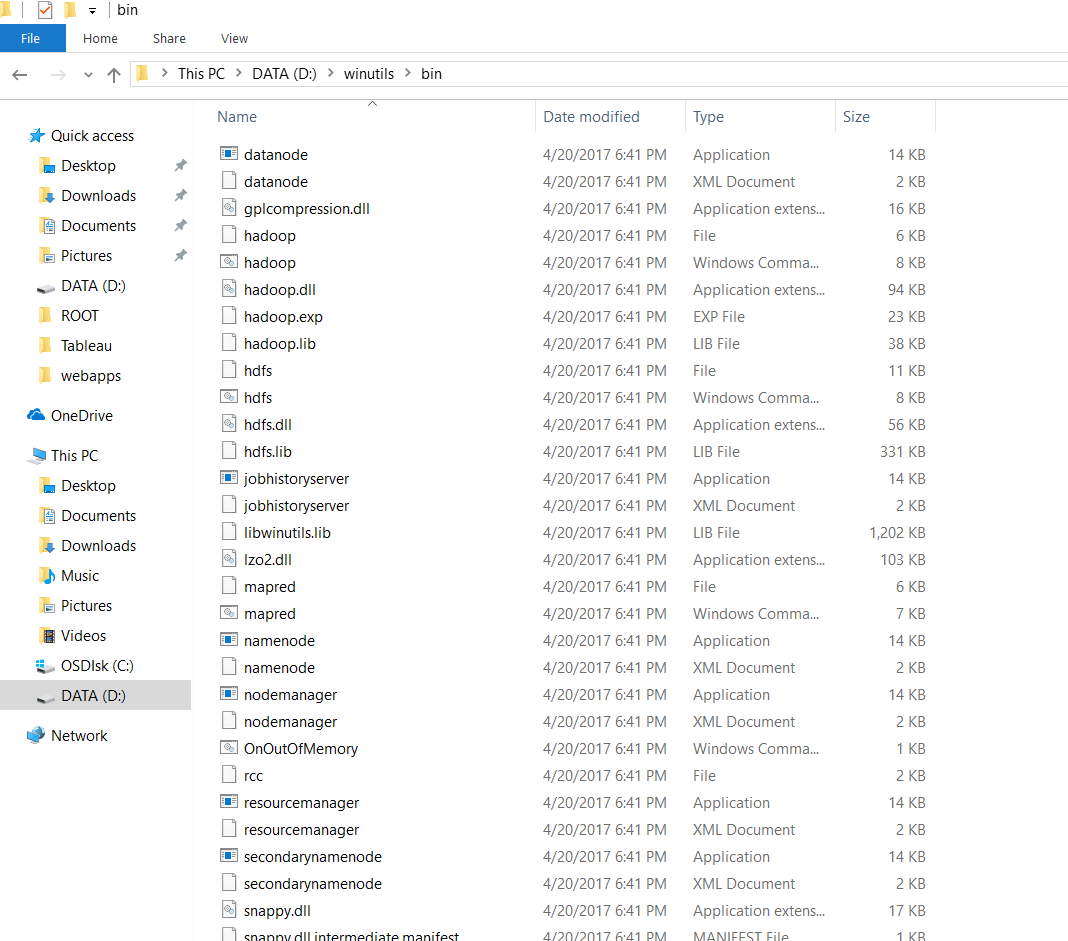
1. Error - Winutils.exe - Spark depends on Hadoop libraries, if Winutils are not set up the Cassandra connectivity do not work well as Spark depends on Java libraries of running.
2. Error - Setting up a master and Worker node – A master and worker node is required for running an application like connecting to Cassandra.
3. Exception - Write permission on tmp/Hive – write permission should be set up on tmp/Hive directory. One should always run the command prompt on administrator mode.

**Setting up Winutils.exe**

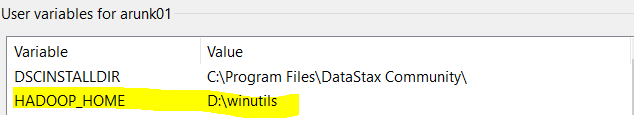
Winutils should be cloned or downloaded from here - <https://github.com/steveloughran/winutils>

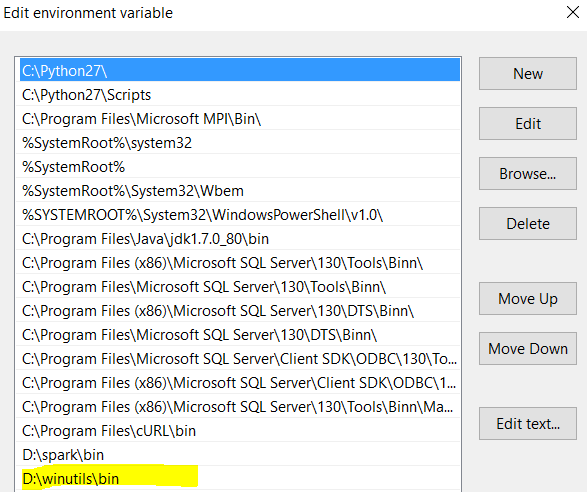
Unzip the downloaded folder and select the ‘bin’ component of winutils for hadoop2.6. Create a folder in D drive and place the ‘bin’ part in the path as below

D:\winutils\bin



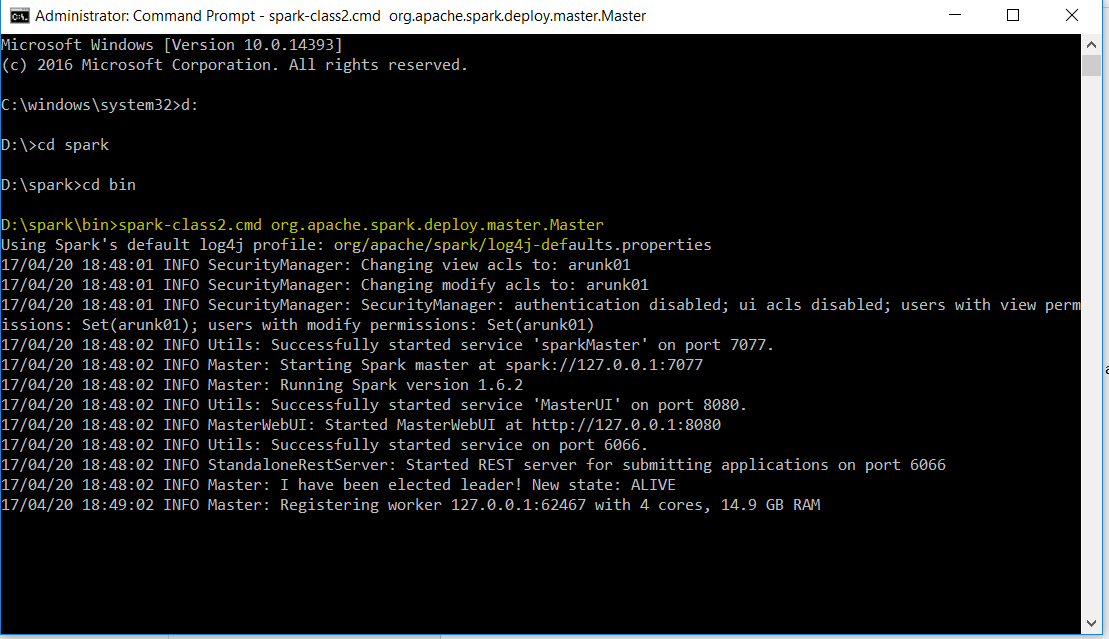
Set up %HADOOP\_HOME% in the environment variables pointing to the winutils directory as shown below –

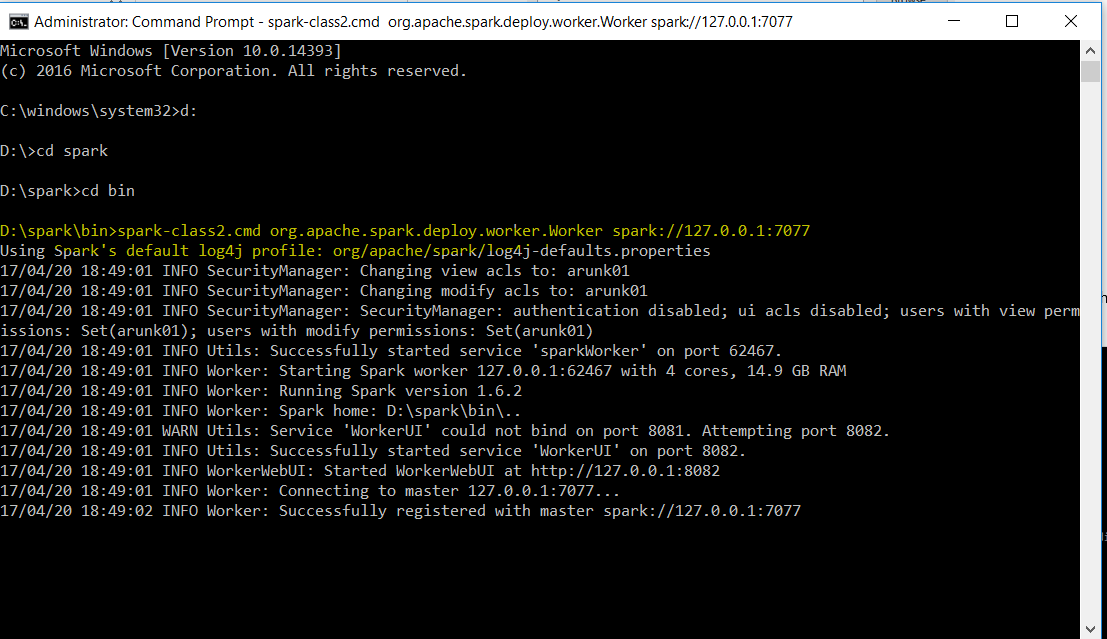




**Setting up Master and Worker node**

After setting up the winutils, we have to set up master and worker node as shown below screen shot.





Master and Worker node should be running at the time of SPARK – Cassandra activity.

**Write permissions on /tmp directory**

You have to provide write permissions on the directory that is created at root while running the SPARK. The directory path is /tmp/hive.

Open CMD in admin mode and at root type the following –

Winutils.exe chmod 777 /tmp/hive

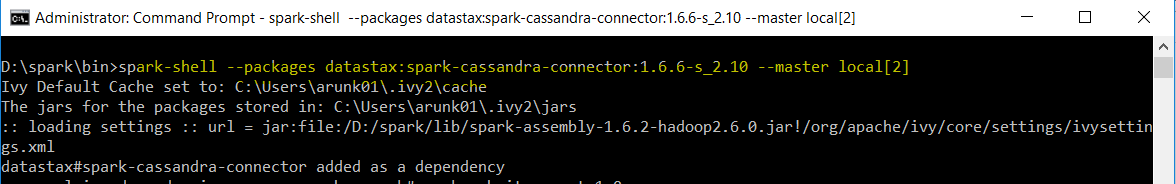
One can also see the SPARK working in a browser by typing <http://localhost:4040>

**Step – 4 – Connecting SPARK and Cassandra with a Connector and extracting data from Cassandra table**

SPARK-Cassandra connector can be set up in many different ways. In Linux and Ubuntu systems, it can be built using SBT or Maven or Eclipse. Whereas in Windows it is complicated to build your own Connector Package using SBT. It is better to download a pre-built connector package and use it with SPARK-SHELL.

The pre-built connector package can be downloaded from here - <https://spark-packages.org/package/datastax/spark-cassandra-connector>

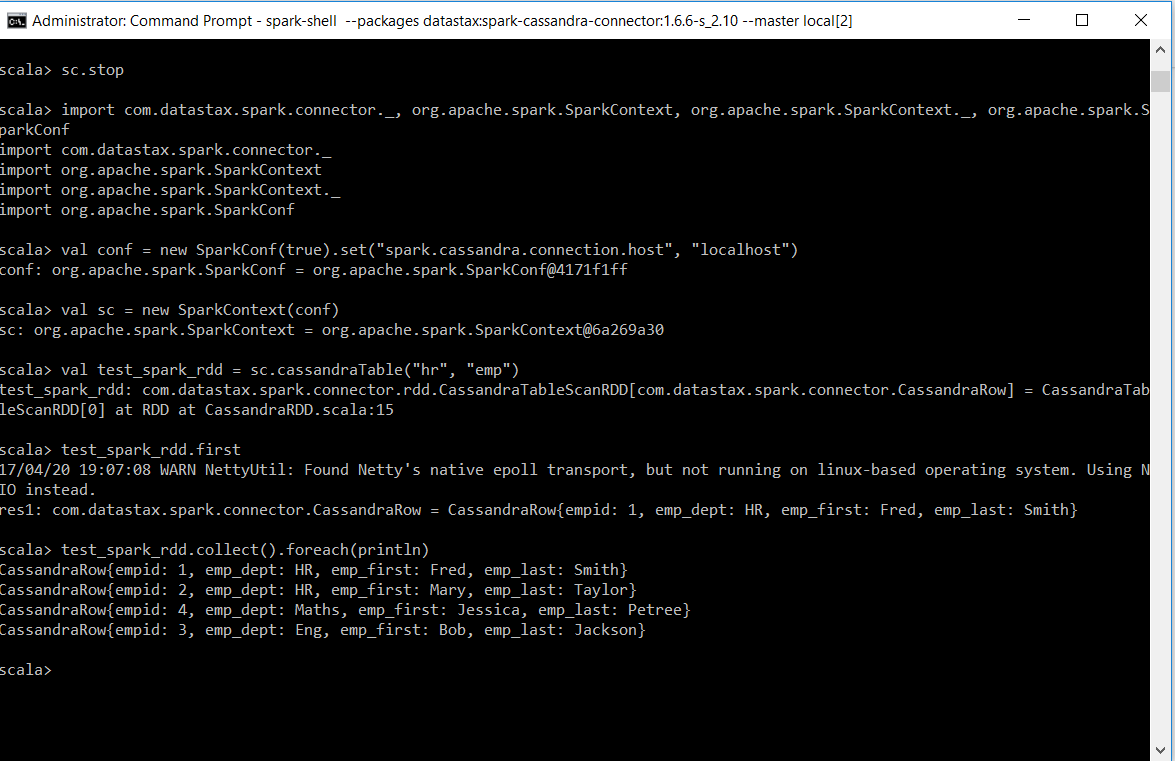
You can download and put it in the root directory or run the package from the command prompt as shown in the screen shot below –



This will run the SPARK and you will get a SCALA Prompt. In the SCALA prompt type the following

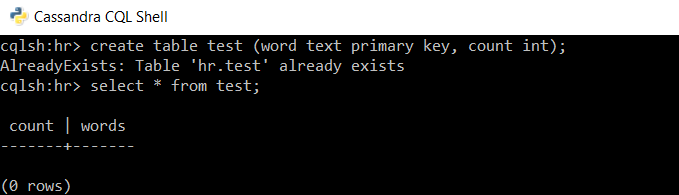
Scala > sc.stop

Follow the screen shot below that shows extraction of data from Cassandra –

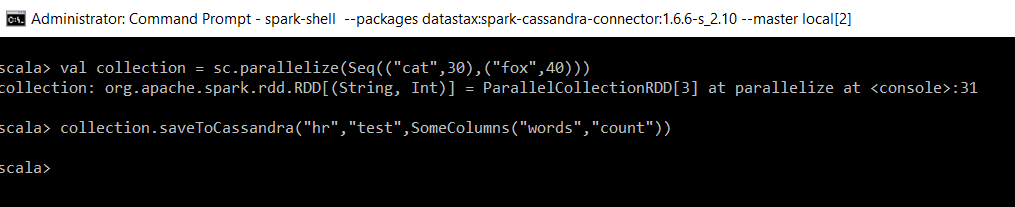


**Step – 5 – Connecting SPARK and Cassandra with a Connector and inserting data into a Cassandra table from SPARK**

Create an empty table in Cassandra in Keyspace ‘hr’ as shown below –



Run as below in SPARK to enter data in columns ‘Count’ and ‘Words’ in Cassandra Table ‘test’



Check in Cassandra CQL to show the data in the table –

