	MALLA REDDY COLLEGE OF ENGINEERING (Approved by AICTE, Permanently Affiliated to JNTUH) Recognised under Section 2(f) & 12(B) of the UGC Act 1956, An ISO 9001:2015 Certified Institution. Maisammaguda, Dhulapally, post via Kompally, Secunderabad - 500100

DEPARTMENT OF CSE-DS &AI&DS

To, 8-01-24

The Principal,

MRCE.

Respected Sir,

Sub: Request to conduct WORKSHOP on **"BIG DATA ANALYTICS"**-reg

Greetings!!!

Our department of CSE-DS & AI&DS has planned to organize an WORKSHOP on **"BIG DATA ANALYTICS"** dated on 8-01-24 TO 11-01-24 & 18-01-24 TO 19-01-24 @9.30 AM to 4PM. Here with I attached the budget proposal for your kind attention.

Kindly accept the same and do the needful.

SL.NO	NAME OF THE EVENT	TITLE	TARGET AUDIENCE	DATE
1	WORKSHOP	**"BIG DATA ANALYTICS LAB"**	III YEAR (AI & DS)STUDENTS	8-01-24 TO 11-01-24 & 18-01-24 TO 19-01-24 @9.30 AM to 4PM AT 211 LAB

Thanking you,

Your's Truly,

[illegible]

MALLA REDDY COLLEGE OF ENGINEERING

(Approved by AICTE, Permanently Affiliated to JNTUH)
Recognised under Section 2(f) & 12(B) of the UGC Act 1956, An ISO 9001:2015 Certified Institution.
Maisammaguda, Dhulapally, post via Kompally, Secunderabad - 500100

MRCE R&D UPDATES

INVITATION

A workshop

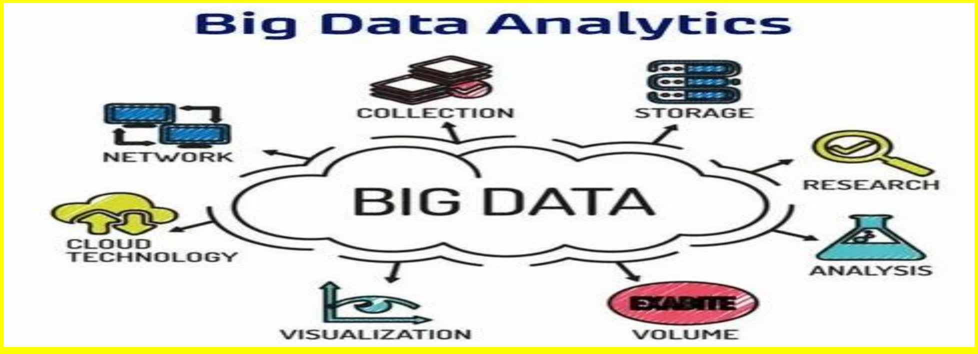
on

"BIG DATA ANALYTICS"

DATE: 8-01-24 TO 11-01-24 &18-01-24 TO 19-01-24

TIME : 9:30AM TO 4PM (OFFLINE)

PARTICIPANTS: STUDENTS& FACULTY MEMBERS



Mrs.K.PAARIJATHA	**Dr.J.Britto**	**Dr.M.Ashok**
FACULTY COORDINATORS	**HoD/CSE-DS & DEAN(R&D)**	**PRINCIPAL**

"Hard work beats talent when talent doesn't work hard."

HALL TICKET NO	NAME OF THE CANDIDATE
21Q91A7201	ALA SHIVA KUMAR REDDY
21Q91A7202	BADAM SIVA RAMI REDDY
21Q91A7203	BEKKARI HEMAVARDHAN
21Q91A7204	BODDU THARUN KUMAR
21Q91A7205	CHENNARAM SANTOSH KUMAR REDDY
21Q91A7206	CHITTIREDDY NAVEEN
21Q91A7207	DESU SAI CHARAN
21Q91A7208	DHAGNAPUR AKSHAYA
21Q91A7209	DIDDI CHANDANA SREE
21Q91A7210	GAJULA SATHWIK
21Q91A7211	GANGONI SAMANTH
21Q91A7212	GOLI VARSHA
21Q91A7213	GOVURI HARSHINI
21Q91A7214	GUGULOTH GOPALAKRISHNA
21Q91A7215	GUGULOTH RAHUL JADAV
21Q91A7216	GUVVA PRASHANTHI
21Q91A7217	K VENKAT NARAYANA
21Q91A7218	K VIVEK
21Q91A7219	KACHARLA SANTHOSH
21Q91A7220	KONERI NAVEEN SHA
21Q91A7221	KRISHNA KUSHWAH
21Q91A7222	KUCHIPUDI ARAVIND KUMAR
21Q91A7223	KUKKAKLA KEERTHAN RAJ
21Q91A7224	KUMMARI SAICHARAN
21Q91A7225	MAALE PRANATHI
21Q91A7226	MADHUKUNTE NITHIN REDDY
21Q91A7227	MALLEBOINA SHIVA KUMAR
21Q91A7230	NALLAN CHAKRAVATHULA YASHASWINI
21Q91A7231	P BHAVANI
21Q91A7232	PALLE VENKATA ASWITHADEVI
21Q91A7233	PAMMI VENKATA KOUSIK
21Q91A7234	PAMU SAI GOUTHAM

21Q91A7236	PASHAM ABHINAV REDDY
21Q91A7237	PASULA VAMSHI RAJ
21Q91A7238	PASUNUTI HARIKA
21Q91A7241	SYED MINHAJ UDDIN
21Q91A7242	TELU VANI
21Q91A7243	VINAY GOPALAPURAM
22Q95A7202	POTHAKAMOORI SAI KIRAN
22Q95A7203	BEEGARI NIKHIL
22Q95A7204	GUDURU VANKATA KOUSHIK
22Q95A7205	PADILAM THARUN
22Q95A7206	VARA BHARGAV RAJ

MALLA REDDY COLLEGE OF ENGINEERING

Approved by AICTE-New Delhi, Permanently Affiliated to JNTUH, Recognized under section 2 (f) & 12(B) of UGC Act 1956 Accredited by NBA & NAAC

DEPT.OF CSE-DS &AI&DS

(True success is all about working towards meaningful goals and dreams)

Report

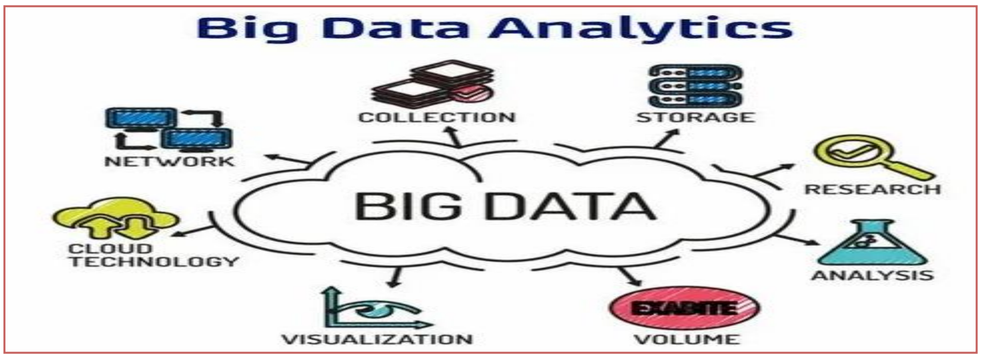
AN ONE WEEK WORKSHOP(OFFLINE) ON

BIG DATA ANALYTICS

08-01-24 TO 11-01-24 &18-01-24 TO 19-01-24

PARTICIPANTS

III AIDS STUDENTS



ALL ARE CORDIALLY INVITED !!!

BIG DATA

Big Data and Its Challenges

Big Data refers to the massive amount of data that cannot be stored, processed, and analyzed using traditional ways.

The main elements of Big Data are:

- Volume - There is a massive amount of data generated every second.
- Velocity - The speed at which data is generated, collected, and analyzed
- Variety - The different types of data: structured, semi-structured, unstructured
- Value - The ability to turn data into useful insights for your business
- Veracity - Trustworthiness in terms of quality and accuracy

The main challenges that Big Data faced and the solutions for each are listed below:

Why Is Hadoop Important?

Hadoop is a beneficial technology for data analysts. There are many essential features in Hadoop which make it so important and user-friendly.

1. The system is able to store and process enormous amounts of data at an extremely fast rate. A semi-structured, structured and unstructured data set can differ depending on how the data is structured.
2. Enhances operational decision-making and batch workloads for historical analysis by supporting real-time analytics.
3. Data can be stored by organisations, and it can be filtered for specific analytical uses by processors as needed.
4. A large number of nodes can be added to Hadoop as it is scalable, so organisations will be able to pick up more data.
5. A protection mechanism prevents applications and data processing from being harmed by hardware failures. Nodes that are down are automatically redirected to other nodes, allowing applications to run without interruption.

Based on the above reasons, we can say that Hadoop is important.

Who Uses Hadoop?

Hadoop is a popular big data tool, used by many companies worldwide. Here's a brief sample of successful Hadoop users:

- British Airways
- Uber
- The Bank of Scotland
- Netflix
- The National Security Agency (NSA), of the United States
- The UK's Royal Mail system
- Expedia
- Twitter

Now that we have some idea of Hadoop's popularity, it's time for a closer look at its components to gain an understanding of what is Hadoop.

Components of Hadoop

Hadoop is a framework that uses distributed storage and parallel processing to store and manage Big Data. It is the most commonly used software to handle Big Data. There are three components of Hadoop.

1. Hadoop HDFS - Hadoop Distributed File System (HDFS) is the storage unit of Hadoop.
2. Hadoop MapReduce - Hadoop MapReduce is the processing unit of Hadoop.
3. Hadoop YARN - Hadoop YARN is a resource management unit of Hadoop.

Let us take a detailed look at Hadoop HDFS in this part of the What is Hadoop article.

Hadoop HDFS

Data is stored in a distributed manner in HDFS. There are two components of HDFS - name node and data node. While there is only one name node, there can be multiple data nodes.

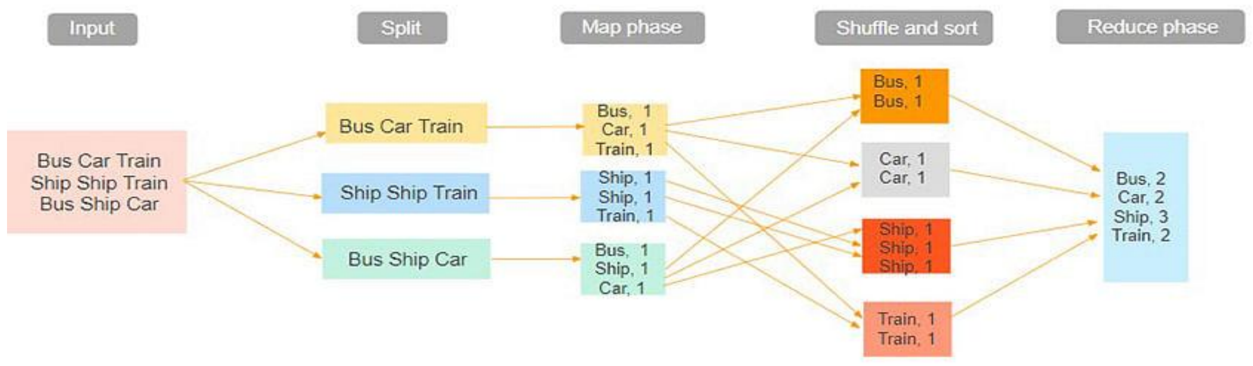
Features of HDFS

- Provides distributed storage
- Can be implemented on commodity hardware
- Provides data security
- Highly fault-tolerant - If one machine goes down, the data from that machine goes to the next machine

Hadoop MapReduce

Hadoop MapReduce is the processing unit of Hadoop. In the MapReduce approach, the processing is done at the slave nodes, and the final result is sent to the master node.

A data containing code is used to process the entire data. This coded data is usually very small in comparison to the data itself. You only need to send a few kilobytes worth of code to perform a heavy-duty process on computers.



The input dataset is first split into chunks of data. In this example, the input has three lines of text with three separate entities - “bus car train,” “ship ship train,” “bus ship car.” The dataset is then split into three chunks, based on these entities, and processed parallelly.

In the map phase, the data is assigned a key and a value of 1. In this case, we have one bus, one car, one ship, and one train.

These key-value pairs are then shuffled and sorted together based on their keys. At the reduce phase, the aggregation takes place, and the final output is obtained.

Hadoop YARN is the next concept we shall focus on in the What is Hadoop article.

Hadoop YARN

Hadoop YARN stands for Yet Another Resource Negotiator. It is the resource management unit of Hadoop and is available as a component of Hadoop version 2.

- Hadoop YARN acts like an OS to Hadoop. It is a file system that is built on top of HDFS.
- It is responsible for managing cluster resources to make sure you don't overload one machine.
- It performs job scheduling to make sure that the jobs are scheduled in the right place

How Does Hadoop Work?

The primary function of Hadoop is to process the data in an organised manner among the cluster of commodity software. The client should submit the data or program that needs to be processed. Hadoop HDFS stores the data. YARN, MapReduce divides the resources and assigns the tasks to the data. Let's know the working of Hadoop in detail.

- The client input data is divided into 128 MB blocks by HDFS. Blocks are replicated according to the replication factor: various DataNodes house the unions and their duplicates.
- The user can process the data once all blocks have been put on HDFS DataNodes.
- The client sends Hadoop the MapReduce programme to process the data.
- The user-submitted software was then scheduled by ResourceManager on particular cluster nodes.
- The output is written back to the HDFS once processing has been completed by all nodes.

Hadoop Distributed File System:

HDFS is known as the Hadoop distributed file system. It is the allocated File System. It is the primary data storage system in Hadoop Applications. It is the storage system of Hadoop that is spread all over the system. In HDFS, the data is once written on the server, and it will continuously be used many times according to the need. The targets of HDFS are as follows.

- The ability to recover from hardware failures in a timely manner
- Access to Streaming Data
- Accommodation of Large data sets
- Portability

How Hadoop Improves on Traditional Databases

- Understanding what is Hadoop requires further understanding on how it differs from traditional databases.
- Hadoop uses the HDFS (Hadoop Data File System) to divide the massive data amounts into manageable smaller pieces, then saved on clusters of community servers. This offers scalability and economy.
- Furthermore, Hadoop employs MapReduce to run parallel processings, which both stores and retrieves data faster than information residing on a traditional database. Traditional databases are great for handling predictable and constant workflows; otherwise, you need Hadoop's power of scalable infrastructure.

Advantages of Hadoop for Big Data

Hadoop was created to deal with big data, so it's hardly surprising that it offers so many benefits. The five main benefits are:

- Speed. Hadoop’s concurrent processing, MapReduce model, and HDFS lets users run complex queries in just a few seconds.
- Diversity. Hadoop’s HDFS can store different data formats, like structured, semi-structured, and unstructured.
- Cost-Effective. Hadoop is an open-source data framework.
- Resilient. Data stored in a node is replicated in other cluster nodes, ensuring fault tolerance.
- Scalable. Since Hadoop functions in a distributed environment, you can easily add more servers.

List of Experiments:

1. Implement a simple map-reduce job that builds an inverted index on the set of input documents (Hadoop)

2. Process big data in HBase

3. Store and retrieve data in Pig

4. Perform Social media analysis using Cassandra

5. Buyer event analytics using Cassandra on suitable product sales data.

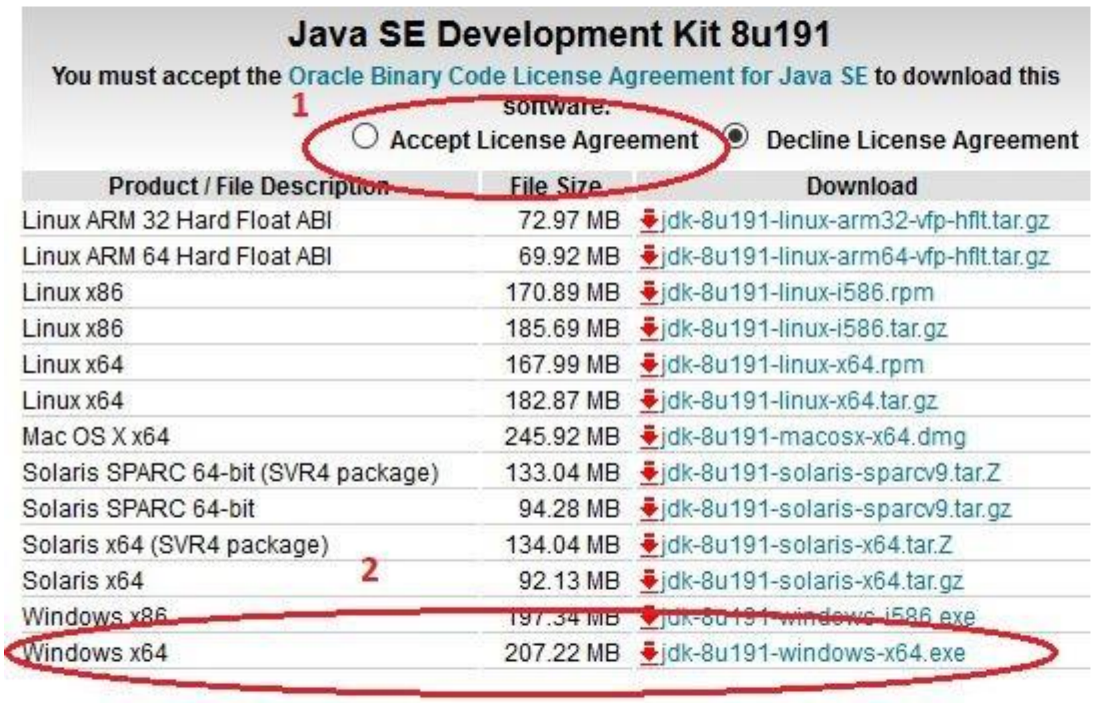
INSTALLING HADOOP IN WINDOWS 10

Preparations

A. Make sure that you are using Windows 10 and are logged in as admin.

B. Download Java jdk1.8.0 from https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

C. Accept Licence Agreement [**1**] and download the exe-file [**2**]



Java SE Development Kit 8u191

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

1

○ Accept License Agreement ● Decline License Agreement

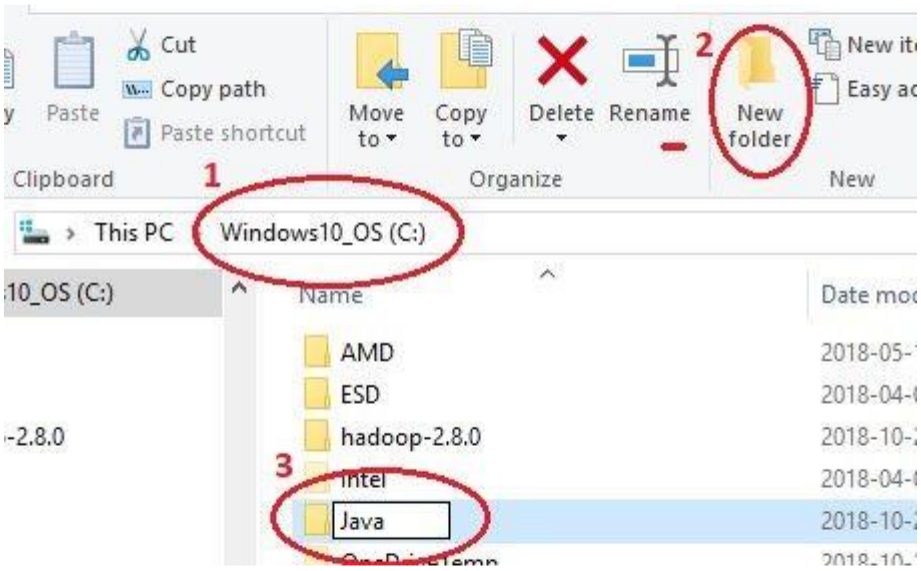
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	72.97 MB	jdk-8u191-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	69.92 MB	jdk-8u191-linux-arm64-vfp-hflt.tar.gz
Linux x86	170.89 MB	jdk-8u191-linux-i586.rpm
Linux x86	185.69 MB	jdk-8u191-linux-i586.tar.gz
Linux x64	167.99 MB	jdk-8u191-linux-x64.rpm
Linux x64	182.87 MB	jdk-8u191-linux-x64.tar.gz
Mac OS X x64	245.92 MB	jdk-8u191-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	133.04 MB	jdk-8u191-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	94.28 MB	jdk-8u191-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	134.04 MB	jdk-8u191-solaris-x64.tar.Z
Solaris x64	92.13 MB	jdk-8u191-solaris-x64.tar.gz
Windows x86	197.34 MB	jdk-8u191-windows-i586.exe
Windows x64	207.22 MB	jdk-8u191-windows-x64.exe

2

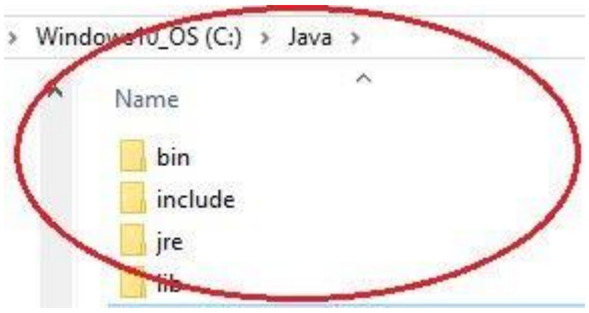
D. Download Hadoop 2.8.0 from
http://archive.apache.org/dist/hadoop/core//hadoop-2.8.0/hadoop-2.8.0.tar.gz

E. Download Notepad++ from
https://notepad-plus-plus.org (current version for Windows)

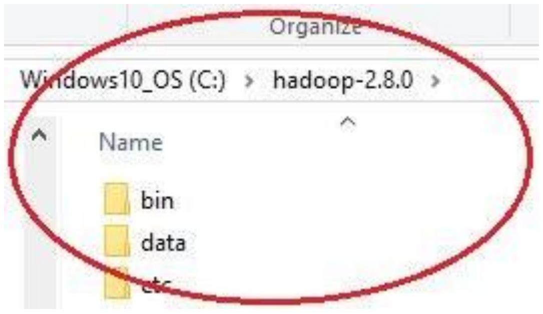
F. Navigate to C:\ [1], make a New folder [2] and name it Java [3]



G. Run the Java installation file **jdk-8u191-windows-x64**. Install direct in the folder **C:\Java**, or move the items from the folder **jdk1.8.0** to the folder **C:\Java**. It should look like this:



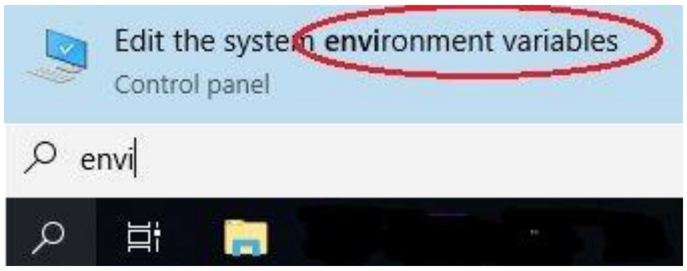
H. Install Hadoop 2.8.0 right under C:\ like this:



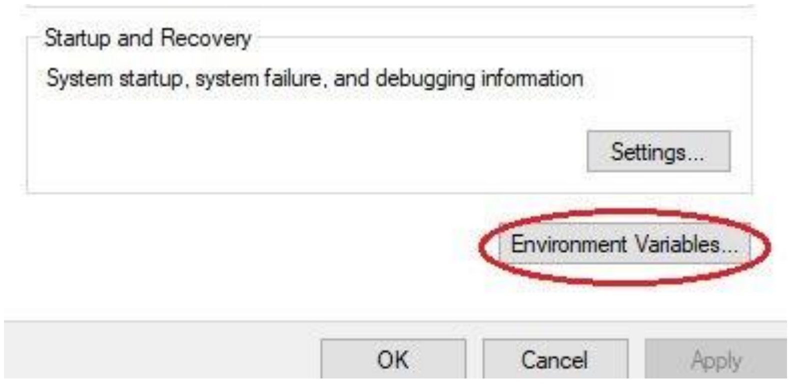
I. Install Notepad++ anywhere

Setup Environment variables

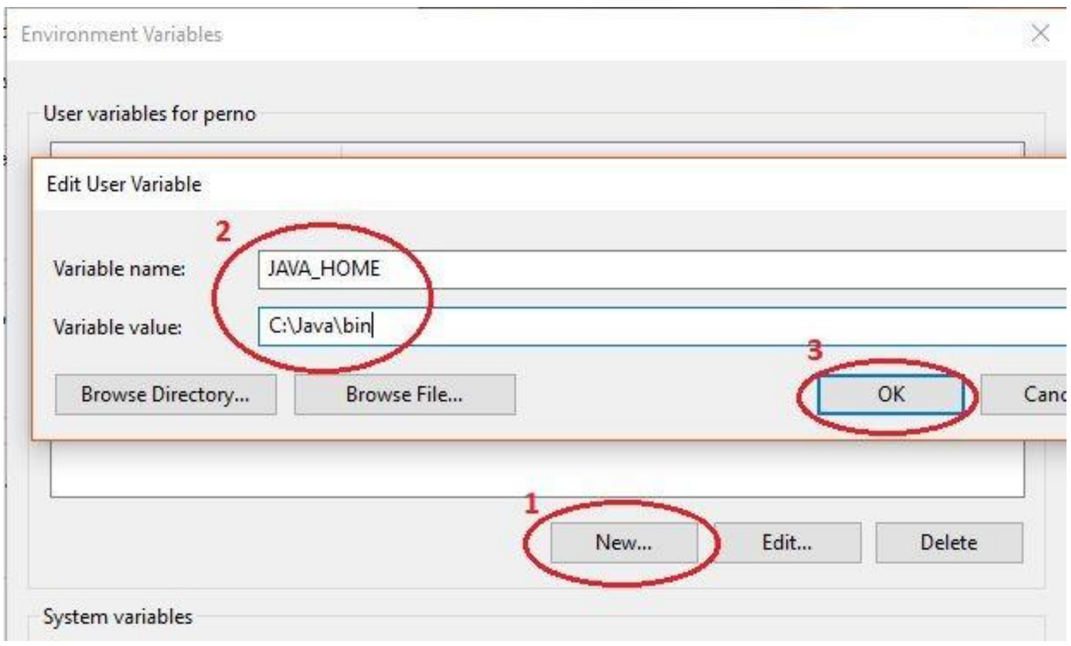
A. Use the search-function to find the environment variables.



In System properties, click the button Environment Variables...

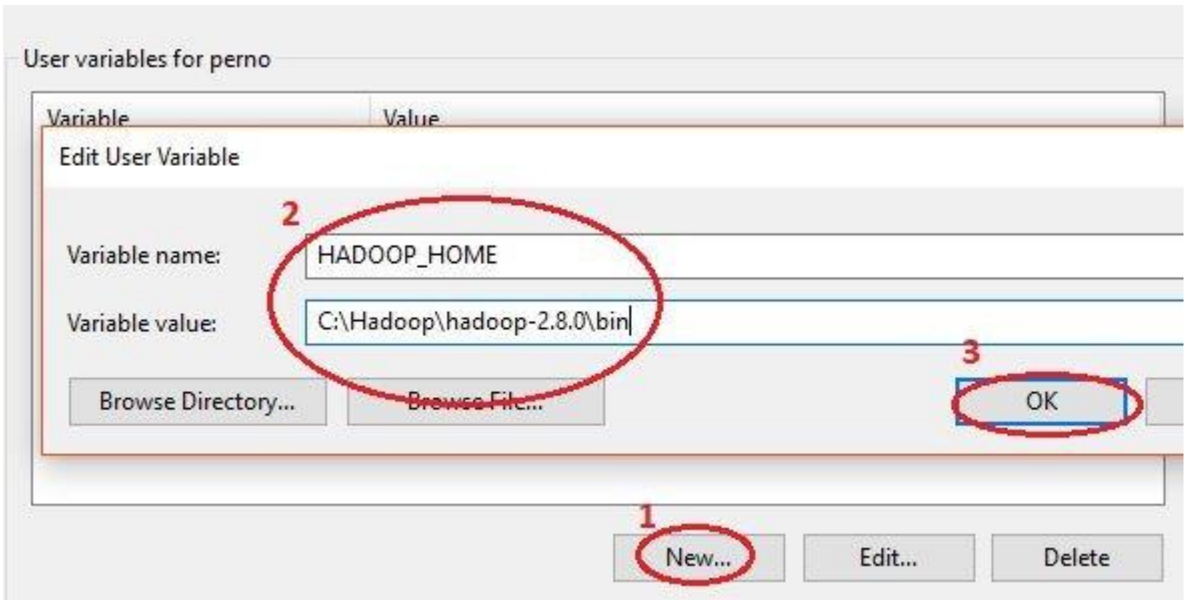


A new window will open with two tables and buttons. The upper table is for User variablesand the lower for System variables.



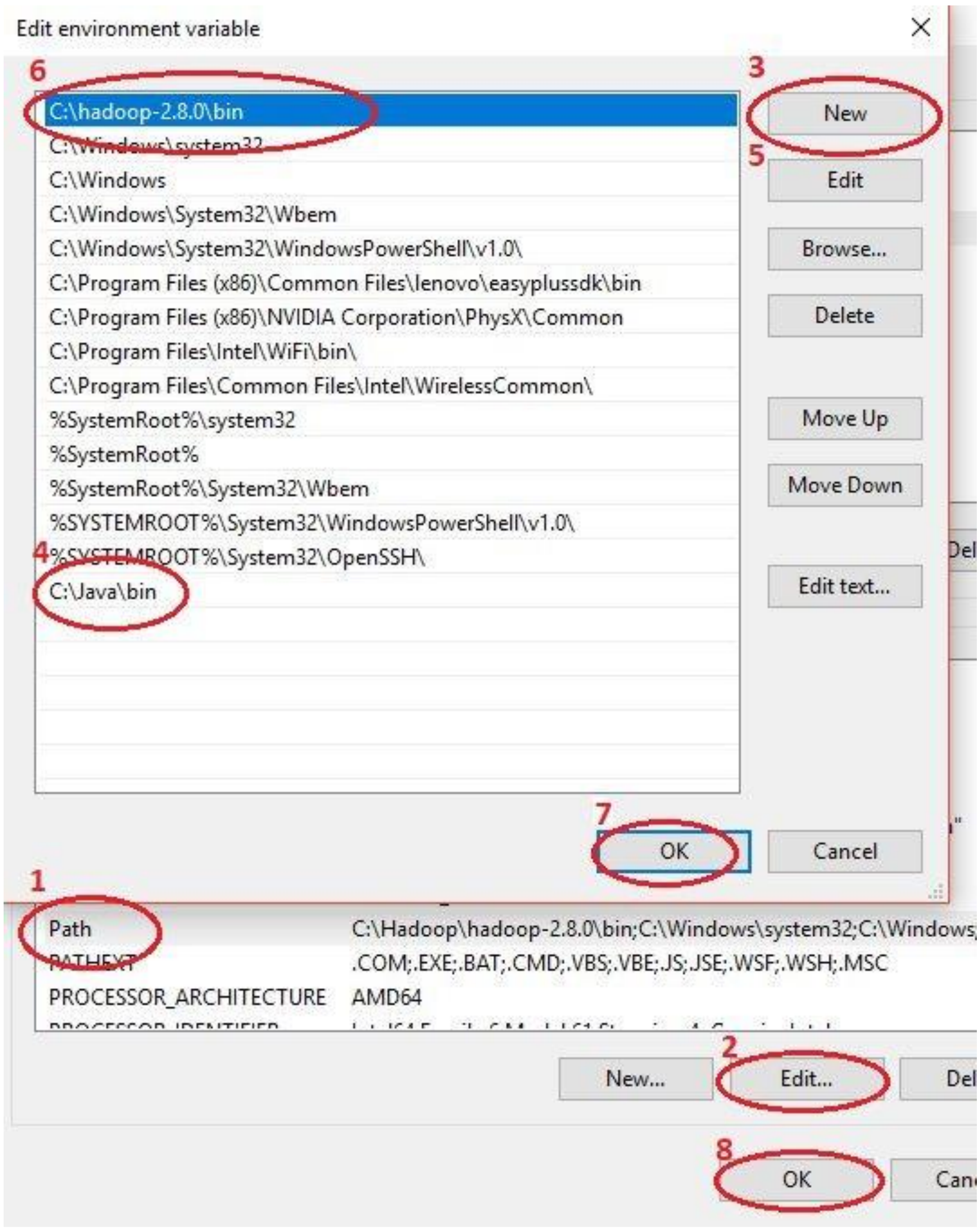
Variable name: JAVA_HOME
Variable value: C:\Java\bin

B. Make another New User variable [1]. Name it HADOOP_HOME and set it to the hadoop-2.8.0 bin-folder [2]. Click OK [3].



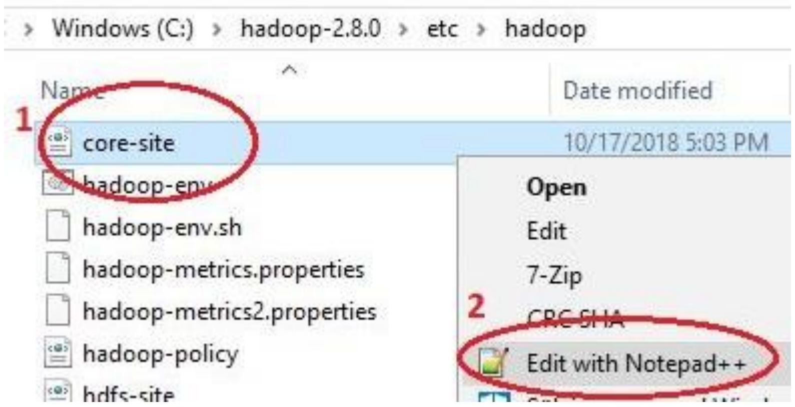
Variable name: HADOOP_HOME
Variable value: C:\Hadoop\hadoop-2.8.0\bin

C. Now add Java and Hadoop to System variables path: Go to path [1] and click edit [2]. The editor window opens. Chose New [3] and add the address C:\Java\bin [4]. Chose New again [5] and add the address C:\hadoop-2.8.0\bin [6]. Click OK [7] in the editor window and OK [8] to change the System variables.


Edit environment variable
C:\hadoop-2.8.0\bin
C:\Windows\system32
C:\Windows
C:\Windows\System32\Wbem
C:\Windows\System32\WindowsPowerShell\v1.0\
C:\Program Files (x86)\Common Files\lenovo\easyplussdk\bin
C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common
C:\Program Files\Intel\WiFi\bin\
C:\Program Files\Common Files\Intel\WirelessCommon\
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Java\bin
New
Edit
Browse...
Delete
Move Up
Move Down
Edit text...
OK
Cancel
Path
C:\Hadoop\hadoop-2.8.0\bin;C:\Windows\system32;C:\Windows;
PATHEXT
.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE
AMD64
New...
Edit...
OK
1
2
3
4
5
6
7
8

Configuration

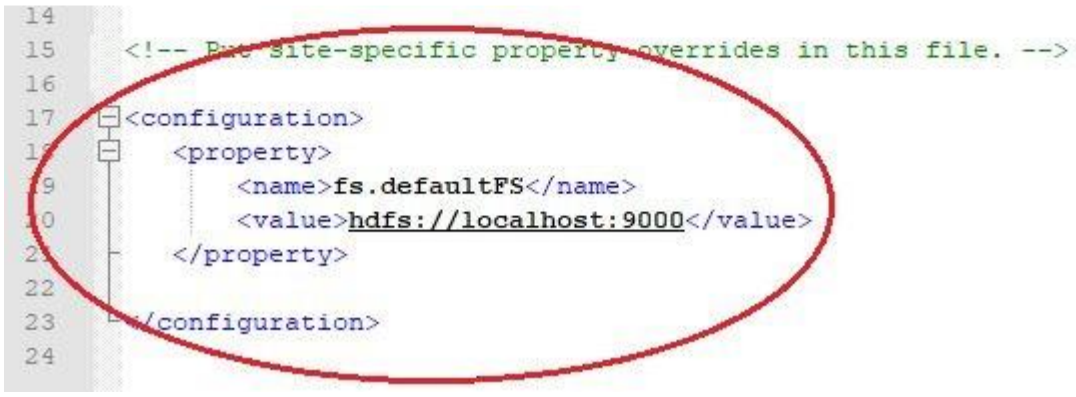
A. Go to the file **C:\Hadoop\Hadoop-2.8.0\etc\hadoop\core-site.xml** [1]. Right-click on the file and edit with Notepad++ [2].



B. In the end of the file you have two configuration tags.
`<configuration>`
`</configuration>`
Paste the following code between the two tags and save (spacing doesn't matter):

```
<property>
   <name>fs.defaultFS</name>
   <value>hdfs://localhost:9000</value>
</property>
```

It should look like this in Notepad++:

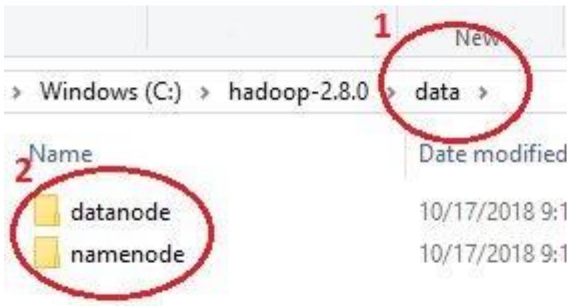


```
<!-- Put site-specific property overrides in this file. -->

<configuration>
   <property>
      <name>fs.defaultFS</name>
      <value>hdfs://localhost:9000</value>
   </property>

</configuration>
```

C. Rename **C:\Hadoop-2.8.0\etc\hadoop\mapred-site.xml.template** to **mapred-site.xml** and edit this file with Notepad++. Paste the following code between the configuration tags and save:

```
<property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
</property>
```

D. Under C:\Hadoop-2.8.0 create a folder named data [1] with two subfolders, “datanode” and “namenode” [2].



E. Edit the file **C:\Hadoop-2.8.0\etc\hadoop/hdfs-site.xml** with Notepad++. Paste the following code between the configuration tags and save:

```
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>C:\hadoop-2.8.0\data\namenode</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>C:\hadoop-2.8.0\data\datanode</value>
</property>
```

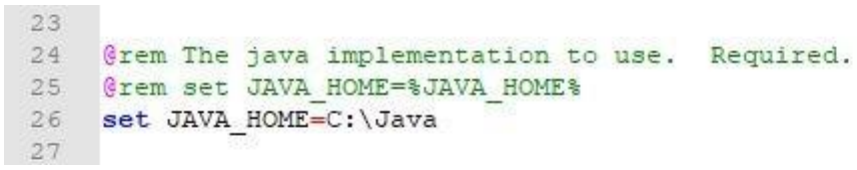
F. Edit the file **C:\Hadoop-2.8.0\etc\hadoop\yarn-site.xml** with Notepad++. Paste the following code between the configuration tags and save:

```
<property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
</property>
<property>
        <name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>
        <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
```

G. Edit the file **C:\Hadoop-2.8.0\etc/hadoop\hadoop-env.cmd** with Notepad++.
Write **@rem** in front of "set JAVA_HOME=%JAVA_HOME%".

Write **set JAVA_HOME=C:\Java** at the next row.

It should look like this is Notepad++:



```

@rem The java implementation to use.  Required.
@rem set JAVA_HOME=%JAVA_HOME%
set JAVA_HOME=C:\Java

```

Don't forget to save.

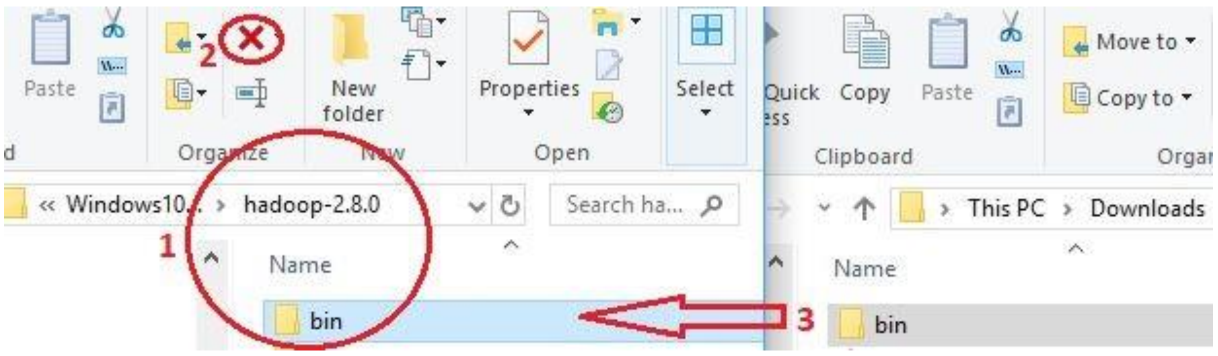
Bravo, configuration done!

Replace the bin folder

Before we can start testing we must exchange a folder in Hadoop.

A. Download **Hadoop Configuration.zip**
[https://github.com/MuhammadBilalYar/HADOOP-INSTALLATION-ON-WINDOW-10/blob /master/Hadoop%20Configuration.zip](https://github.com/MuhammadBilalYar/HADOOP-INSTALLATION-ON-WINDOW-10/blob/master/Hadoop%20Configuration.zip) Unzip the bin file.

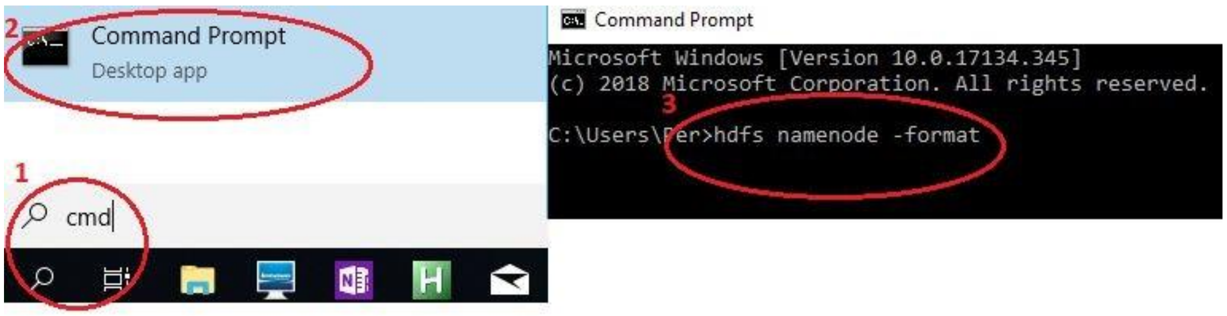
B. Delete the bin file **C:\Hadoop\Hadoop-2.8.0\bin** [1, 2] and replace it with the new bin-folder from Hadoop Configuration.zip. [3].



Testing

A. Search for cmd [1] and open the Command Prompt [2]. Write

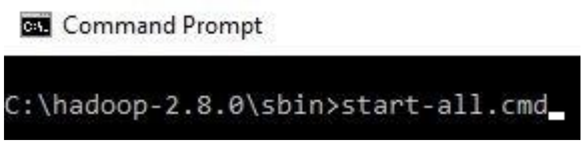
hdfs namenode –format [3] and push enter.



```
Microsoft Windows [Version 10.0.17134.345]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Per>hdfs namenode -format
```

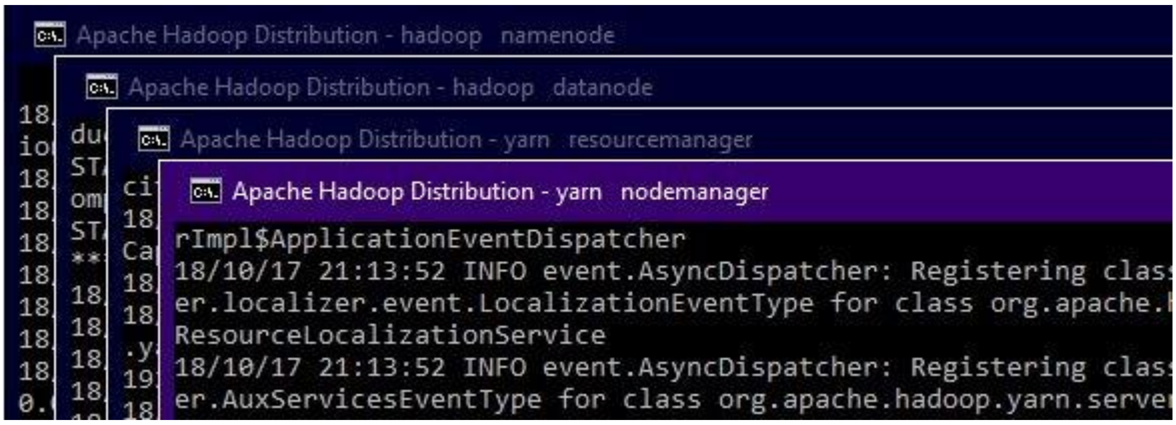
If this first test works the Command Prompt will run a lot of information. It is a good sign!

B. Now you must change directory in the Command Prompt. Write **cd C:\hadoop-2.8.0\sbin** and push enter. In the sbin folder, write **start-all.cmd** and push enter.



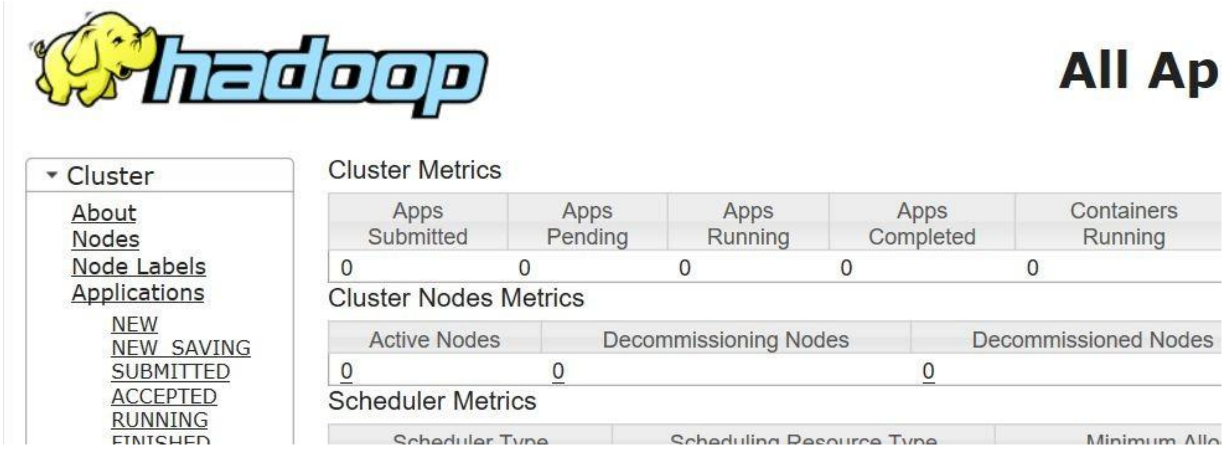
```
Command Prompt

C:\hadoop-2.8.0\sbin>start-all.cmd_
```

If the configuration is right, four apps will start running and it will look something like this:



```
Apache Hadoop Distribution - hadoop   namenode
Apache Hadoop Distribution - hadoop   datanode
Apache Hadoop Distribution - yarn   resourcemanager
Apache Hadoop Distribution - yarn   nodemanager
rImpl$ApplicationEventDispatcher
18/10/17 21:13:52 INFO event.AsyncDispatcher: Registering clas
er.localizer.event.LocalizationEventType for class org.apache.
ResourceLocalizationService
18/10/17 21:13:52 INFO event.AsyncDispatcher: Registering clas
er.AuxServicesEventType for class org.apache.hadoop.yarn.serve
```

C. Now open a browser and write in the address field **localhost:8088** and push enter.
Can you see the little hadoop elephant? Then you have made a really good work!



Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running
0	0	0	0	0

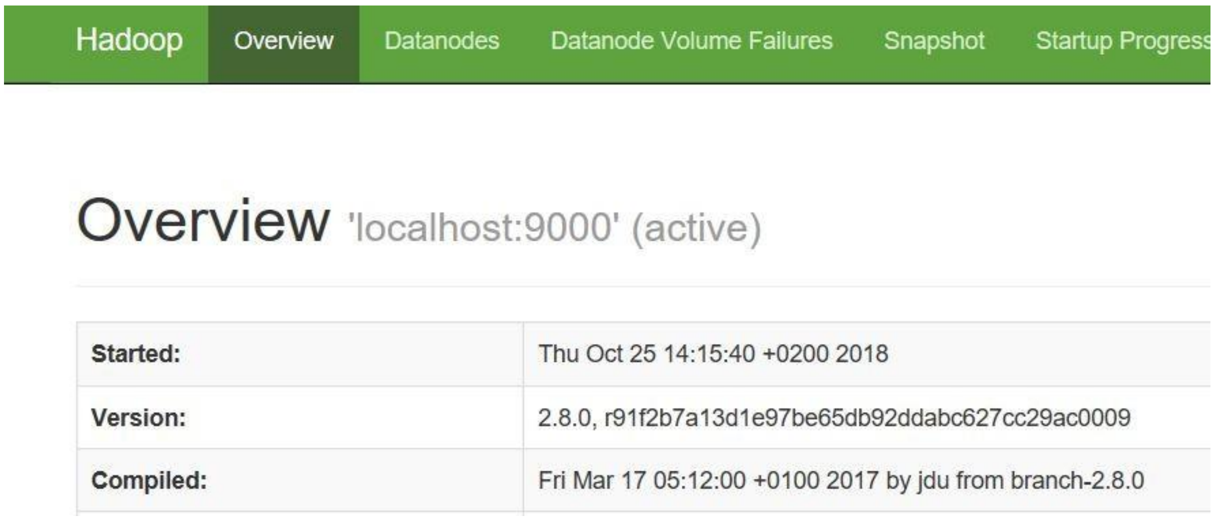
Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes
0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allo

D. Last test - try to write **localhost:50070** instead.



Overview 'localhost:9000' (active)

Started:	Thu Oct 25 14:15:40 +0200 2018
Version:	2.8.0, r91f2b7a13d1e97be65db92ddabc627cc29ac0009
Compiled:	Fri Mar 17 05:12:00 +0100 2017 by jdu from branch-2.8.0

If you can see the overview you have implemented Hadoop on your PC.

Congratulations, you did it!!!

****

To close the running programs, run "stop-all.cmd" in tho command prompt

Program 1:

1. Write the steps about hadoop installation in windows 10

The step-by-step method

The step-by-step method uses headlines and letters to keep track of the successive flow. It have five main steps with substeps labeled by letters. The substeps are sometimes illustrated, and the illustrations are often numbered in order to show the linear process in detail.

Preparations

A. Check that you are logged in as admin.
B. Download Java jdk1.8.0
C. Download Hadoop 2.8.0
D. Download Notepad++
E. Create a folder C:\Java
F. Install Java jdk1.8.0 in the folder C:\Java
G. Install Hadoop 2.8.0 right under C:\
H. Install Notepad++
I. If the Windows firewall is activated open ports 8088 and 50070

Setup Environment variables

A. Set the JAVA_HOME Environment variable to Java bin-folder.
B. Set the HADOOP_HOME Environment variable to Hadoop bin-folder.
C. Add Java and Hadoop to the bin directory path.

Configuration

A. Edit the file C:\Hadoop\Hadoop-2.8.0\etc\hadoop\core-site.xml.
B. Rename C:\Hadoop-2.8.0\etc/hadoop\mapred-site.xml.template to mapred-site.xml and edit.
C. C:\Hadoop-2.8.0 create a folder named "data" with subfolders "datanode" and "namenode".
D. Edit the file C:\Hadoop-2.8.0\etc\hadoop/hdfs-site.xml.
E. Edit the file C:\Hadoop-2.8.0\etc\hadoop\yarn-site.xml.
F. Edit the file C:\Hadoop-2.8.0\etc/hadoop\hadoop-env.cmd

Replace the Bin library

A. Download Hadoop Configuration.zip and extract the bin file.
B. Replace C:\Hadoop\Hadoop-2.8.0\bin with the bin-file from Hadoop

Configuration.zip.

Testing

A. Run the cmd "hdfs namenode –format".
B. Change directory in cmd: cd C:\Hadoop\Hadoop-2.8.0\sbin and run start-all.cmd.
C. Open browser and write localhost:8088

D. Open browser and write localhost:50070

Program2:

Write a program for simple Word count MapReduce with hadoop using Java in Windows 10

Aim: To perform a program for simple Word count MapReduce with hadoop using Java in Windows 10

Prerequisites:

This program deals with the basic MapReduce program using the Apache Hadoop framework in windows computers. The Hadoop framework installed is Pseudo Distributed (Single node).

This tutorial deals with running the MapReduce program on windows. Hadoop single node framework and JDK with eclipse are already installed.

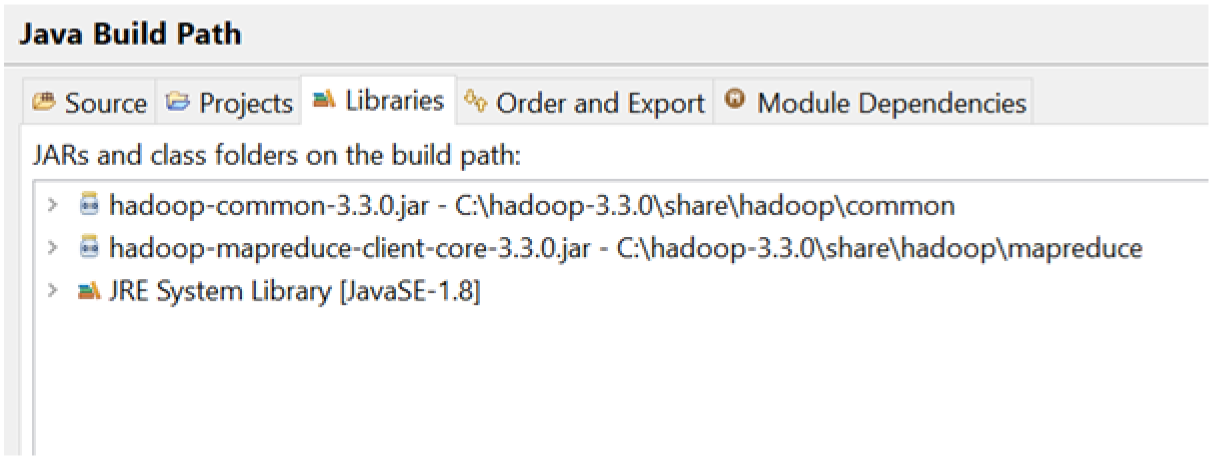
Java JDK version "1.8.0_291" and Hadoop version "3.3.0" are used here; the operations resemble similar to other versions also.

Create text file with some content for word count.

Procedure:

Initially, open eclipse IDE and create a new project. Here the project name is map_reduce_example.

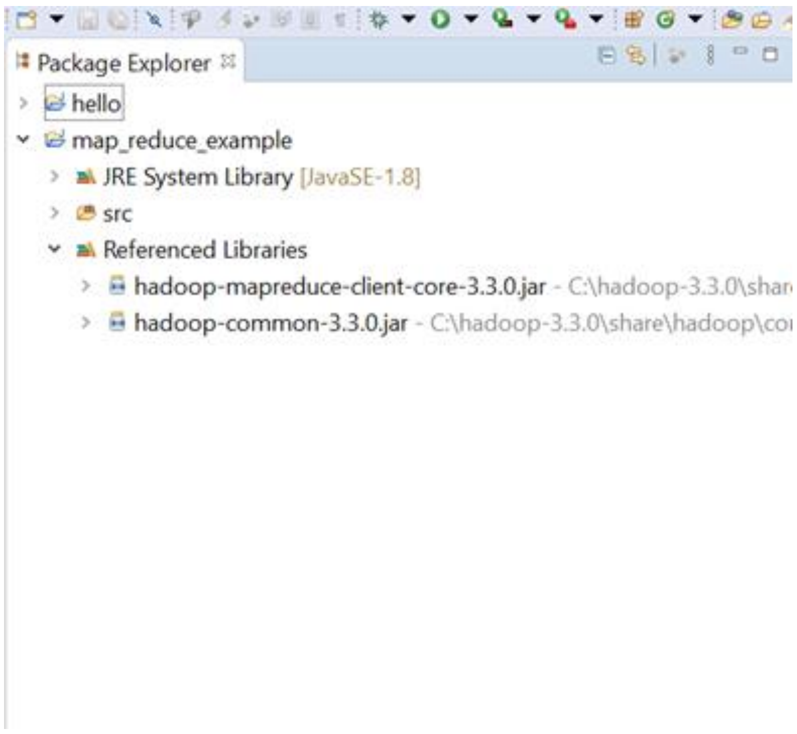
Now use configure build path in your project and add the following external Hadoop jar files to the project as shown below.


Java Build Path

JARs and class folders on the build path:

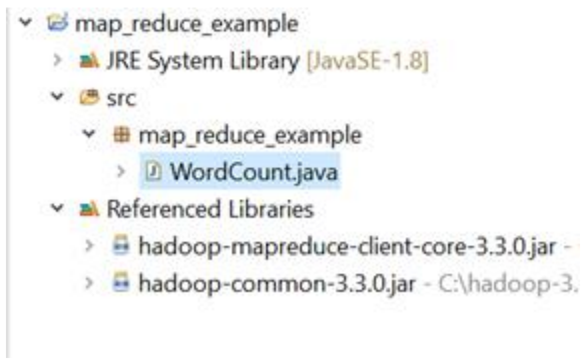
- hadoop-common-3.3.0.jar - C:\hadoop-3.3.0\share\hadoop\common
- hadoop-mapreduce-client-core-3.3.0.jar - C:\hadoop-3.3.0\share\hadoop\mapreduce
- JRE System Library [JavaSE-1.8]

After successfully adding the jar files, the eclipse will show items in referenced libraries, as shown below.



```
> hello
v map_reduce_example
  > JRE System Library [JavaSE-1.8]
  > src
  v Referenced Libraries
    > hadoop-mapreduce-client-core-3.3.0.jar - C:\hadoop-3.3.0\shar
    > hadoop-common-3.3.0.jar - C:\hadoop-3.3.0\share\hadoop\co
```

Next, add source code for the word-count program.


map_reduce_example
JRE System Library [JavaSE-1.8]
src
map_reduce_example
WordCount.java
Referenced Libraries
hadoop-mapreduce-client-core-3.3.0.jar -
hadoop-common-3.3.0.jar - C:\hadoop-3.

copy the below code in the WordCount.java class

Program:

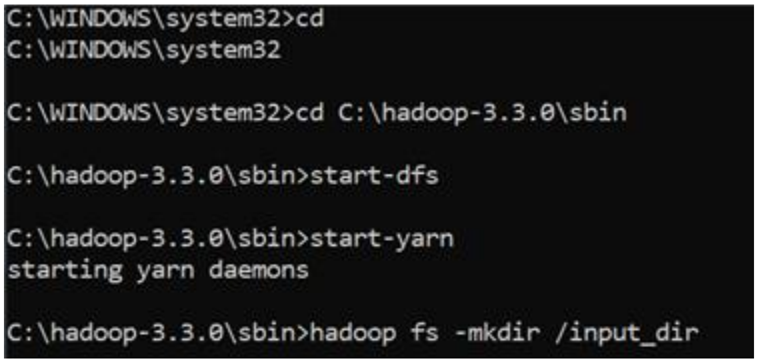
```
import java.io.IOException;

import java.util.StringTokenizer;



import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;



public class WordCount {
```

```
public static class TokenizerMapper

     extends Mapper<Object, Text, Text, IntWritable>{


  private final static IntWritable one = new IntWritable(1);

  private Text word = new Text();


  public void map(Object key, Text value, Context context

                  ) throws IOException, InterruptedException {

    StringTokenizer itr = new StringTokenizer(value.toString());

    while (itr.hasMoreTokens()) {

      word.set(itr.nextToken());

      context.write(word, one);

    }

  }

}


public static class IntSumReducer

     extends Reducer<Text,IntWritable,Text,IntWritable> {

  private IntWritable result = new IntWritable();
```

```
    public void reduce(Text key, Iterable values,
                       Context context
                       ) throws IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) {
        sum += val.get();
      }
      result.set(sum);
      context.write(key, result);
    }
  }

  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));

    FileOutputFormat.setOutputPath(job, new Path(args[1]));

    System.exit(job.waitForCompletion(true) ? 0 : 1);

  }

}
```

After adding source code, create a jar file of map_reduce_example using the export option in eclipse IDE.

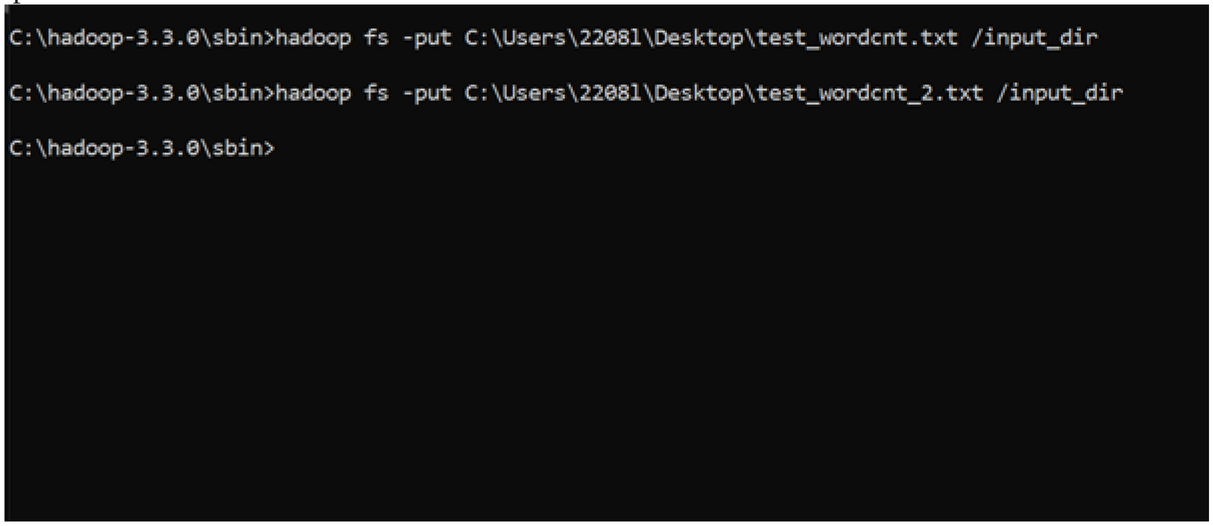
1. Open cmd prompt with administrator status.
2. Move to sbin folder of Hadoop using cd command (***cd C:\hadoop-3.3.0\sbin***)
3. Start the daemons by giving command start-all or better use (***start-dfs*** then ***start-yarn***) for specific initialization



```
C:\WINDOWS\system32>cd
C:\WINDOWS\system32

C:\WINDOWS\system32>cd C:\hadoop-3.3.0\sbin

C:\hadoop-3.3.0\sbin>start-dfs

C:\hadoop-3.3.0\sbin>start-yarn
starting yarn daemons

C:\hadoop-3.3.0\sbin>hadoop fs -mkdir /input_dir
```

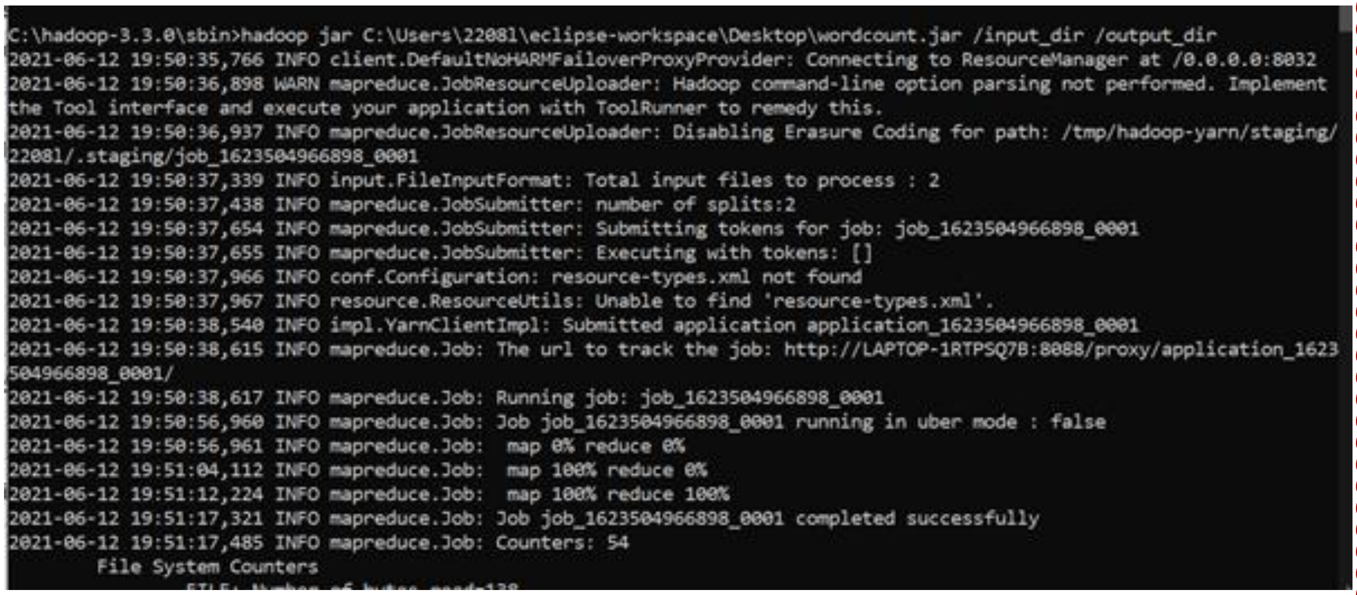
4. After starting, check if all nodes (namenode, datanode, resoucemanager, nodemanager) are working using the command ***(jps)***
5. Make an input directory in the Hadoop cluster using the command (***hadoop fs -mkdir /input_directory)***.
6. Now add the required text files to the input directory using the command (***hadoop fs -put file_path/file_name.txt /input_directory)*** *as shown below.* test_wordcnt and test_wordcnt_2 are my input file with some words with

spaces.



```
C:\hadoop-3.3.0\sbin>hadoop fs -put C:\Users\22081\Desktop\test_wordcnt.txt /input_dir

C:\hadoop-3.3.0\sbin>hadoop fs -put C:\Users\22081\Desktop\test_wordcnt_2.txt /input_dir

C:\hadoop-3.3.0\sbin>
```

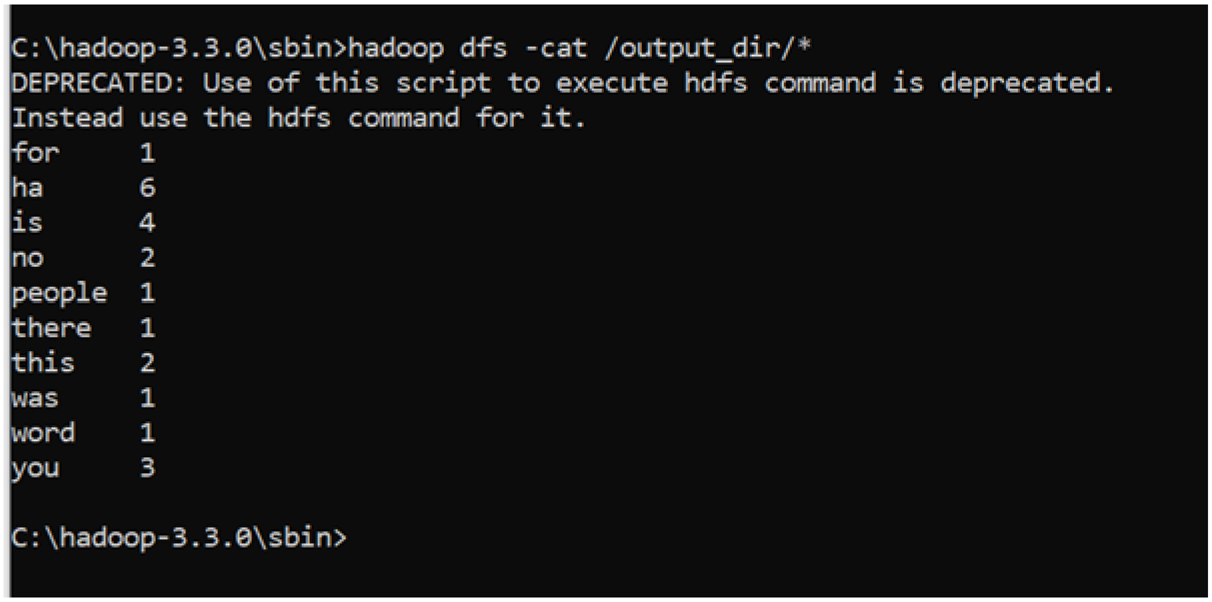
7. Now run the map_reduce jar file exported previously using the Hadoop command (***hadoop jar jarpath/jar_name.txt /input_directory /output_directory)*** --- leave space between input directory and output directory as shown.



```
C:\hadoop-3.3.0\sbin>hadoop jar C:\Users\22081\eclipse-workspace\Desktop\wordcount.jar /input_dir /output_dir
2021-06-12 19:50:35,766 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2021-06-12 19:50:36,898 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement
the Tool interface and execute your application with ToolRunner to remedy this.
2021-06-12 19:50:36,937 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/
22081/.staging/job_1623504966898_0001
2021-06-12 19:50:37,339 INFO input.FileInputFormat: Total input files to process : 2
2021-06-12 19:50:37,438 INFO mapreduce.JobSubmitter: number of splits:2
2021-06-12 19:50:37,654 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1623504966898_0001
2021-06-12 19:50:37,655 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-06-12 19:50:37,966 INFO conf.Configuration: resource-types.xml not found
2021-06-12 19:50:37,967 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-06-12 19:50:38,540 INFO impl.YarnClientImpl: Submitted application application_1623504966898_0001
2021-06-12 19:50:38,615 INFO mapreduce.Job: The url to track the job: http://LAPTOP-1RTPSQ7B:8088/proxy/application_1623
504966898_0001/
2021-06-12 19:50:38,617 INFO mapreduce.Job: Running job: job_1623504966898_0001
2021-06-12 19:50:56,960 INFO mapreduce.Job: Job job_1623504966898_0001 running in uber mode : false
2021-06-12 19:50:56,961 INFO mapreduce.Job:  map 0% reduce 0%
2021-06-12 19:51:04,112 INFO mapreduce.Job:  map 100% reduce 0%
2021-06-12 19:51:12,224 INFO mapreduce.Job:  map 100% reduce 100%
2021-06-12 19:51:17,321 INFO mapreduce.Job: Job job_1623504966898_0001 completed successfully
2021-06-12 19:51:17,485 INFO mapreduce.Job: Counters: 54
        File System Counters
```

8. Finally check the output using command (***hadoop dfs -cat /output_dir/****). By this, we have successfully executed the word count MapReduce program in windows.

Output:



```
C:\hadoop-3.3.0\sbin>hadoop dfs -cat /output_dir/*
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.
for     1
ha      6
is      4
no      2
people  1
there   1
this    2
was     1
word    1
you     3

C:\hadoop-3.3.0\sbin>
```

Program :3 Process big data in HBase

Aim: To perform big data in H-base

Installation of h-base

HBase is a data format equivalent to Google's big table that allows for speedy random access to massive volumes of structured data. HBase is a Java-based open-source, multidimensional, distributed, and scalable NoSQL database that works on top of HDFS (Hadoop Distributed File System).

It is intended to hold a massive number of sparse data sets. It enables users to obtain information in real-time.HBase is a column-oriented database that stores data in tables. Only column families are defined in the HBase table structure. The HBase database is divided into families, and each family can contain an infinite number of columns. The column values are successively saved on a disc. Each table cell has a timestamp.

In this blog, we will learn how to set up Hbase in the window operating system.

Prerequisite

>Install Java JDK - You can download it from
this link. (https://www.oracle.com/java/technologies/downloads/)

>The Java Development Kit (JDK) is a cross-platform software development environment that includes tools and libraries for creating Java-based software applications and applets.

Download Hbase - Download Apache Hbase from this link.
(https://hbase.apache.org/downloads.html)

Steps

Step-1 (Extraction of files)

Extract all the files in C drive

Step-2 (Creating Folder)

Create folders named "hbase" and "zookeeper."

Step-3 (Deleting line in HBase.cmd)

Open hbase.cmd in any text editor.

Search for line %HEAP_SETTINGS% and remove it.

Step-4 (Add lines in hbase-env.cmd)

Now open hbase-env.cmd, which is in the conf folder in any text editor.

Add the below lines in the file after the comment session.

```
set JAVA_HOME=%JAVA_HOME%
set HBASE_CLASSPATH=%HBASE_HOME%\lib\client-facing-thirdparty\*
set HBASE_HEAPSIZE=8000
set HBASE_OPTS="-XX:+UseConcMarkSweepGC" "-Djava.net.preferIPv4Stack=true"
set SERVER_GC_OPTS="-verbose:gc" "-XX:+PrintGCDetails" "-XX:+PrintGCDateStamps"
%HBASE_GC_OPTS%
set HBASE_USE_GC_LOGFILE=true

set HBASE_JMX_BASE="-Dcom.sun.management.jmxremote.ssl=false" "-
Dcom.sun.management.jmxremote.authenticate=false"

set HBASE_MASTER_OPTS=%HBASE_JMX_BASE% "-
Dcom.sun.management.jmxremote.port=10101"
set HBASE_REGIONSERVER_OPTS=%HBASE_JMX_BASE% "-
Dcom.sun.management.jmxremote.port=10102"
set HBASE_THRIFT_OPTS=%HBASE_JMX_BASE% "-
Dcom.sun.management.jmxremote.port=10103"
set HBASE_ZOOKEEPER_OPTS=%HBASE_JMX_BASE% -
Dcom.sun.management.jmxremote.port=10104"
set HBASE_REGIONSERVERS=%HBASE_HOME%\conf\regionservers
set HBASE_LOG_DIR=%HBASE_HOME%\logs
set HBASE_IDENT_STRING=%USERNAME%
set HBASE_MANAGES_ZK=true
```

Step-5 (Add the line in Hbase-site.xml)

Open hbase-site.xml, which is in the conf folder in any text editor.

Add the lines inside the <configuration> tag.

A distributed HBase entirely relies on Zookeeper (for cluster configuration and management). ZooKeeper coordinates, communicates and distributes state between the Masters and RegionServers in Apache HBase.

HBase's design strategy is to use ZooKeeper solely for transient data (that is, for coordination and state communication). Thus, removing HBase's ZooKeeper data affects only temporary operations – data can continue to be written and retrieved to/from HBase.

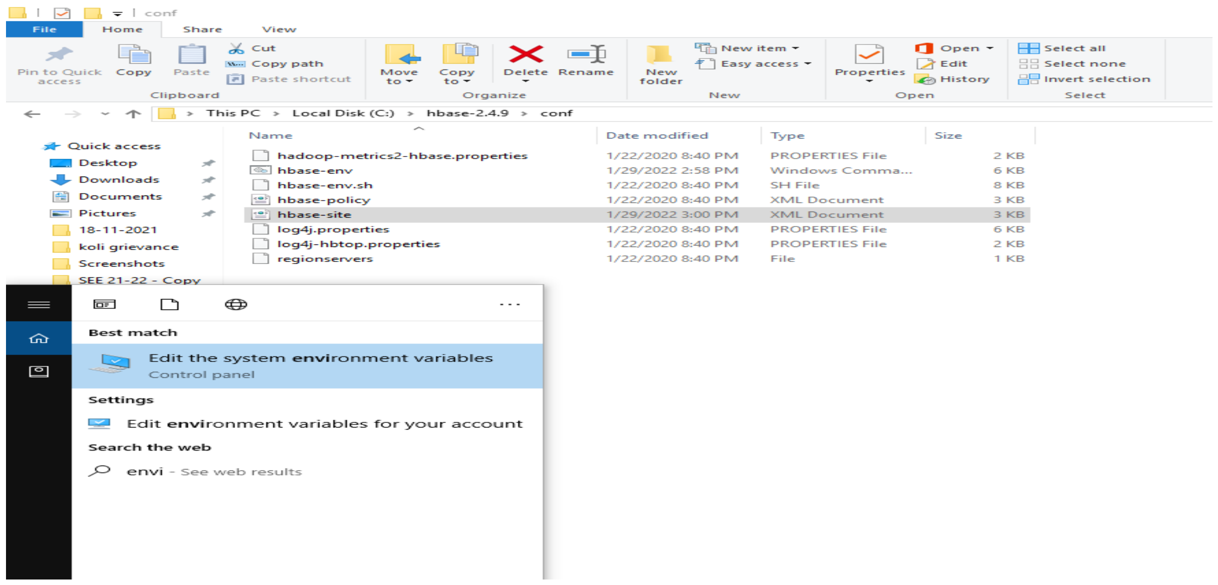
```
<property>
   <name>hbase.rootdir</name>
   <value>file:///C:/Documents/hbase-2.2.5/hbase</value>
```

```
</property>
<property>
  <name>hbase.zookeeper.property.dataDir</name>
  <value>/C:/Documents/hbase-2.2.5/zookeeper</value>
</property>
<property>
   <name> hbase.zookeeper.quorum</name>
  <value>localhost</value>
</property>
```

Step-6 (Setting Environment Variables)

Now set up the environment variables.

Search "System environment variables."



Name	Date modified	Type	Size
hadoop-metrics2-hbase.properties	1/22/2020 8:40 PM	PROPERTIES File	2 KB
hbase-env	1/29/2022 2:58 PM	Windows Comma...	6 KB
hbase-env.sh	1/22/2020 8:40 PM	SH File	8 KB
hbase-policy	1/22/2020 8:40 PM	XML Document	3 KB
hbase-site	1/29/2022 3:00 PM	XML Document	3 KB
log4j.properties	1/22/2020 8:40 PM	PROPERTIES File	6 KB
log4j-hbtop.properties	1/22/2020 8:40 PM	PROPERTIES File	2 KB
regionservers	1/22/2020 8:40 PM	File	1 KB

Best match
Edit the system environment variables
Control panel
Settings
Edit environment variables for your account
Search the web
envi - See web results

Now click on " Environment Variables."

Then click on "New."

Variable name: HBASE_HOME

Variable Value: Put the path of the Hbase folder.

We have completed the HBase Setup on Windows 10 procedure.

Procedure:

- Now we are all set to run HBase, to start HBase execute the command below from the bin folder.
- Open Command Prompt and cd to Hbase' bin directory
- Run start-hbase.cmd
- Look for any errors
- Test the installation using HBase shell
- Open command prompt and cd to HBase' bin directory
- Run hbase shell [should connect to the HBase server]
- Try creating a table
- create 'emp','p'
- list [Table name should get printed]
- put 'emp','emp01','p:fn','First Name'
- scan 'emp' [The row content should get printed]

Program 4:

Store and retrieve data in Pig

Aim: To perform storing and retrieving data in pig

Definition

Pig is an open-source high level data flow system. It provides a simple language called Pig Latin, for queries and data manipulation, which are then compiled in to MapReduce jobs that run on Hadoop.

Pig is important as companies like Yahoo, Google and Microsoft are collecting huge amounts of data sets in the form of click streams, search logs and web crawls. Pig is also used in some form of ad-hoc processing and analysis of all the information.

Why Do you Need Pig?

- It's easy to learn, especially if you're familiar with SQL.

- Pig's multi-query approach reduces the number of times data is scanned. This means 1/20th the lines of code and 1/16th the development time when compared to writing raw MapReduce.
- Performance of Pig is in par with raw MapReduce
- Pig provides data operations like filters, joins, ordering, etc. and nested data types like tuples, bags, and maps, that are missing from MapReduce.
- Pig Latin is easy to write and read.

Installation:

1. Prerequisites

1. Hadoop Cluster Installation

Apache Pig is a platform build on the top of Hadoop. You can refer to our previously published article to install a Hadoop single node cluster on Windows 10.

Note that the Apache Pig latest version 0.17.0 supports Hadoop 2.x versions and still facing some compatibility issues with Hadoop 3.x. In this article, we will only illustrate the installation since we are working with Hadoop 3.2.1

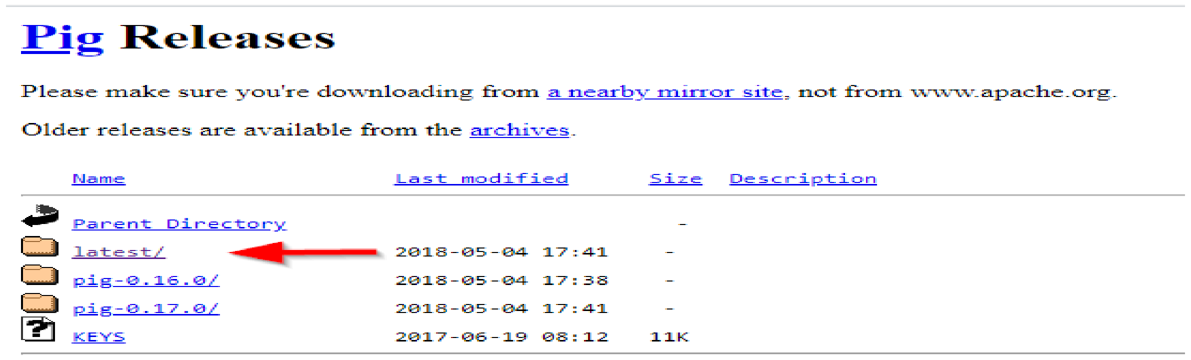
2. 7zip

7zip is needed to extract .tar.gz archives we will be downloading in this guide.

2. Downloading Apache Pig

To download the Apache Pig, you should go to the following link:

- https://downloads.apache.org/pig/



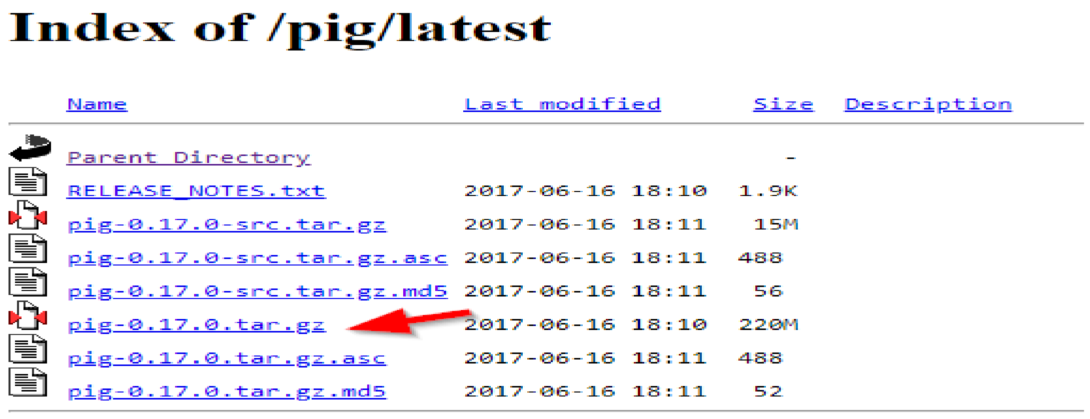
Pig Releases

Please make sure you're downloading from a nearby mirror site, not from www.apache.org.

Older releases are available from the archives.

```
Name                                Last modified      Size  Description

Parent Directory                                        -
latest/                             2018-05-04 17:41    -
pig-0.16.0/                         2018-05-04 17:38    -
pig-0.17.0/                         2018-05-04 17:41    -
KEYS                                2017-06-19 08:12   11K
```

If you are looking for the latest version, navigate to "latest" directory, then download the pig-x.xx.x.tar.gz file.



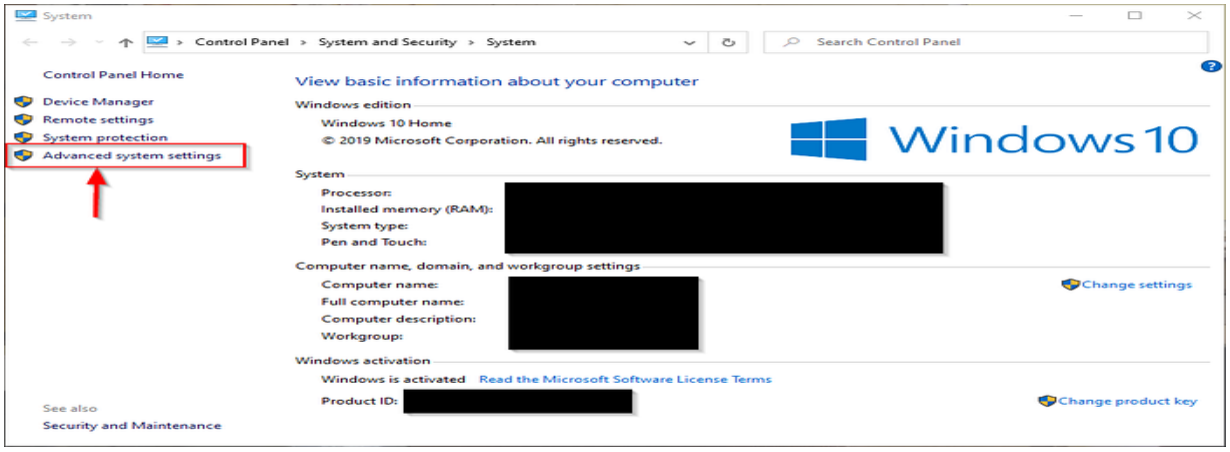
Index of /pig/latest

```
Name                             Last modified      Size  Description

Parent Directory                                     -
RELEASE_NOTES.txt                2017-06-16 18:10  1.9K
pig-0.17.0-src.tar.gz            2017-06-16 18:11   15M
pig-0.17.0-src.tar.gz.asc        2017-06-16 18:11  488
pig-0.17.0-src.tar.gz.md5        2017-06-16 18:11   56
pig-0.17.0.tar.gz                2017-06-16 18:10  220M
pig-0.17.0.tar.gz.asc            2017-06-16 18:11  488
pig-0.17.0.tar.gz.md5            2017-06-16 18:11   52
```

After the file is downloaded, we should extract it twice using 7zip *(using 7zip: the first time we extract the .tar.gz file, the second time we extract the .tar file)*. We will extract the Pig folder into "E:\hadoop-env" directory as used in the previous articles.

3. Setting Environment Variables

After extracting Derby and Hive archives, we should go to Control Panel > System and Security > System. Then Click on "Advanced system settings".



View basic information about your computer

Windows edition

Windows 10 Home

© 2019 Microsoft Corporation. All rights reserved.

System

Processor:
Installed memory (RAM):
System type:
Pen and Touch:
[redacted]

Computer name, domain, and workgroup settings

Computer name:
Full computer name:
Computer description:
Workgroup:
[redacted]

Change settings

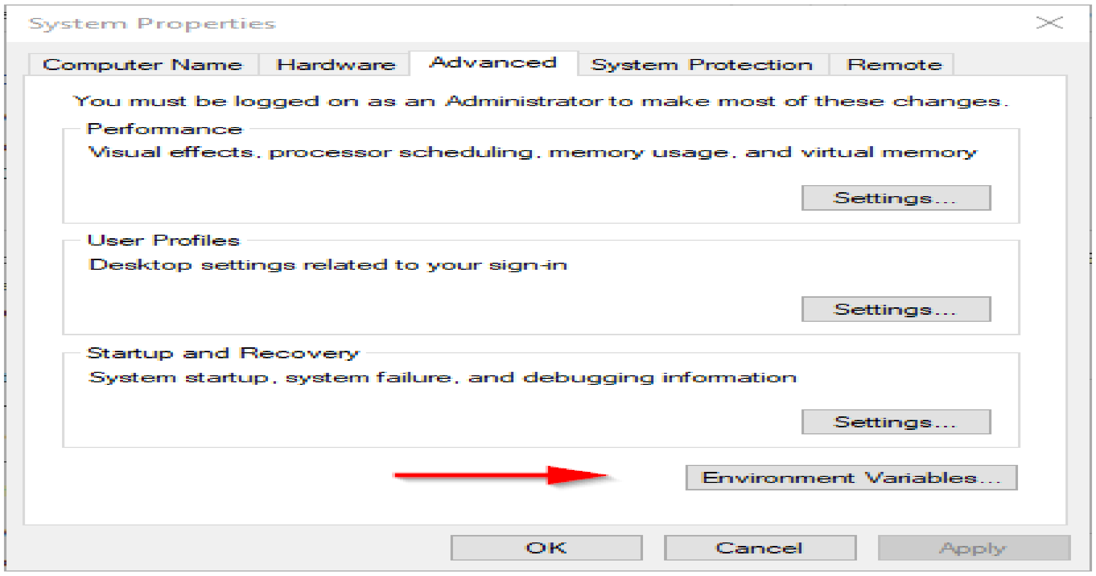
Windows activation

Windows is activated Read the Microsoft Software License Terms

Product ID: [redacted]

Change product key

In the advanced system settings dialog, click on “Environment variables” button.



You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles

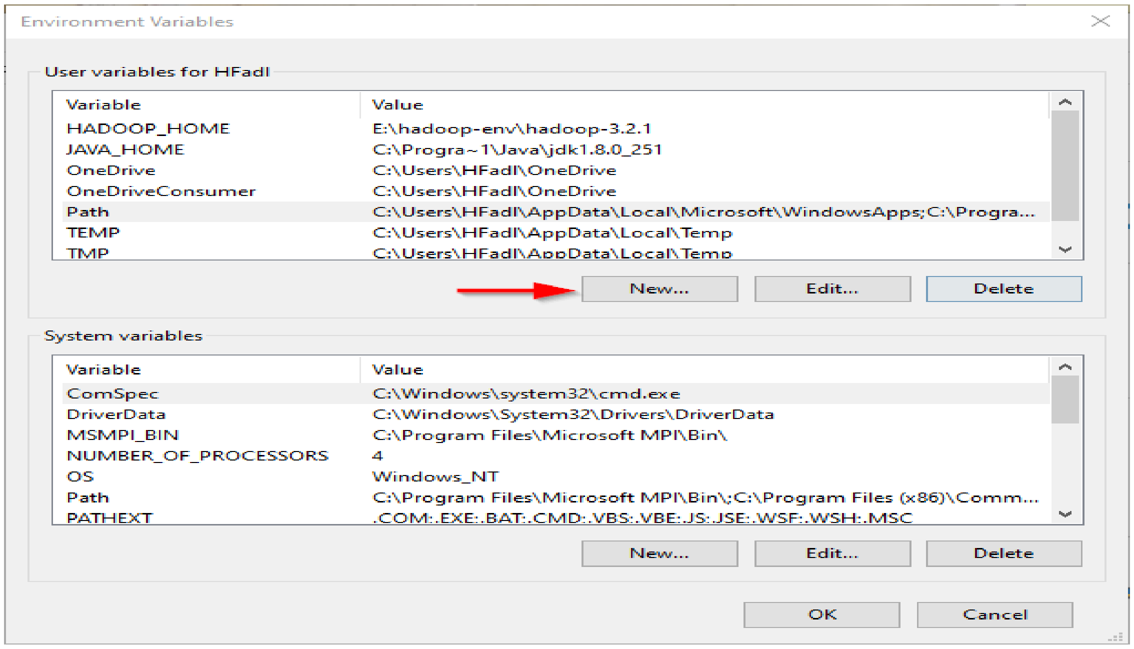
Desktop settings related to your sign-in

Startup and Recovery

System startup, system failure, and debugging information

Figure 4 — Opening environment variables editor

Now we should add the following user variables:



Environment Variables

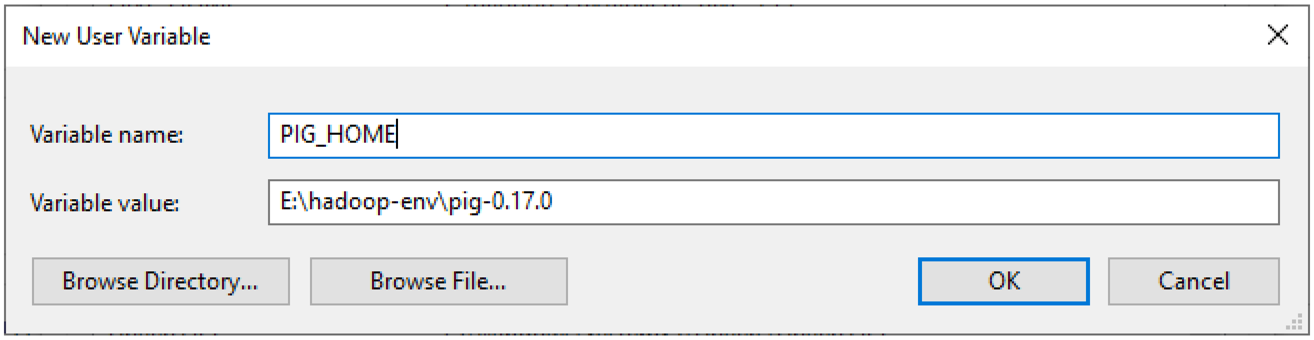
User variables for HFadl

Variable	Value
HADOOP_HOME	E:\hadoop-env\hadoop-3.2.1
JAVA_HOME	C:\Progra~1\Java\jdk1.8.0_251
OneDrive	C:\Users\HFadl\OneDrive
OneDriveConsumer	C:\Users\HFadl\OneDrive
Path	C:\Users\HFadl\AppData\Local\Microsoft\WindowsApps;C:\Progra...
TEMP	C:\Users\HFadl\AppData\Local\Temp
TMP	C:\Users\HFadl\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
MSMPI_BIN	C:\Program Files\Microsoft MPI\Bin\
NUMBER_OF_PROCESSORS	4
OS	Windows_NT
Path	C:\Program Files\Microsoft MPI\Bin\;C:\Program Files (x86)\Comm...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC

- PIG_HOME: “E:\hadoop-env\pig-0.17.0”



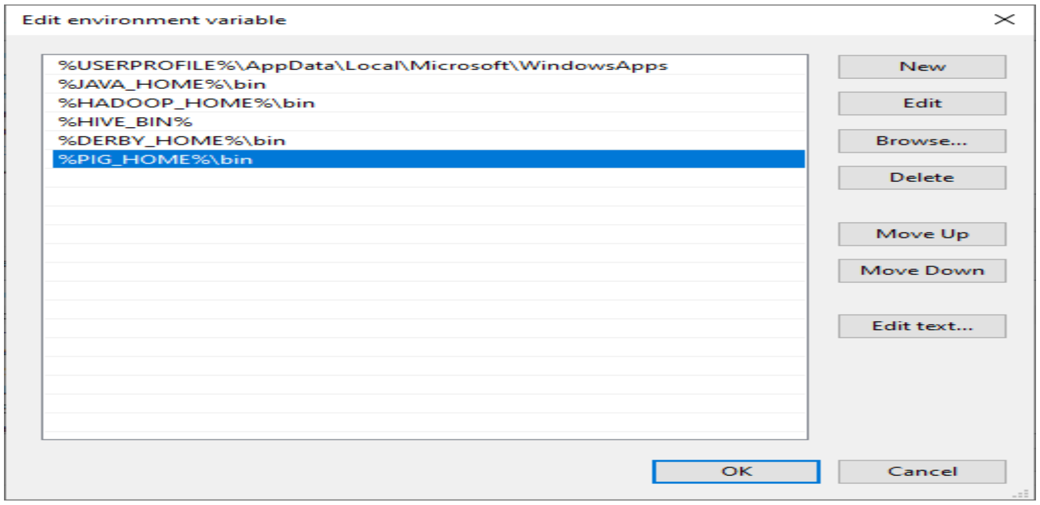
New User Variable

Variable name: PIG_HOME

Variable value: E:\hadoop-env\pig-0.17.0

Now, we should edit the Path user variable to add the following paths:

- %PIG_HOME%\bin



```
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
%JAVA_HOME%\bin
%HADOOP_HOME%\bin
%HIVE_BIN%
%DERBY_HOME%\bin
%PIG_HOME%\bin
```

4. Starting Apache Pig

After setting environment variables, let's try to run Apache Pig.

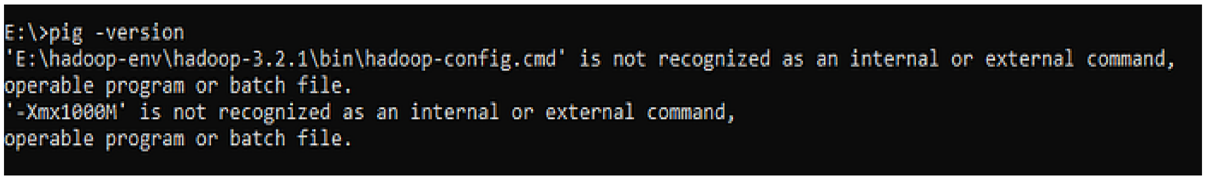
***Note**: Hadoop Services must be running*

Open a command prompt as administrator, and execute the following command

pig –version

You will receive the following exception:

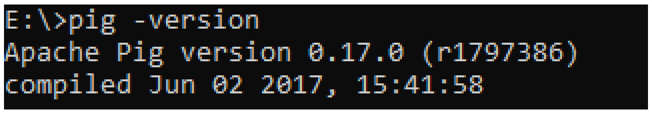
```
'E:\hadoop-env\hadoop-3.2.1\bin\hadoop-config.cmd' is not recognized as an internal or external command,
operable program or batch file.
'-Xmx1000M' is not recognized as an internal or external command,
operable program or batch file.
```



```
E:\>pig -version
'E:\hadoop-env\hadoop-3.2.1\bin\hadoop-config.cmd' is not recognized as an internal or external command,
operable program or batch file.
'-Xmx1000M' is not recognized as an internal or external command,
operable program or batch file.
```

To fix this error, we should edit the pig.cmd file located in the “pig-0.17.0\bin” directory by changing the HADOOP_BIN_PATH value from “%HADOOP_HOME%\bin” to “%HADOOP_HOME%\libexec”.

Now, let's try to run the “pig -version” command again:



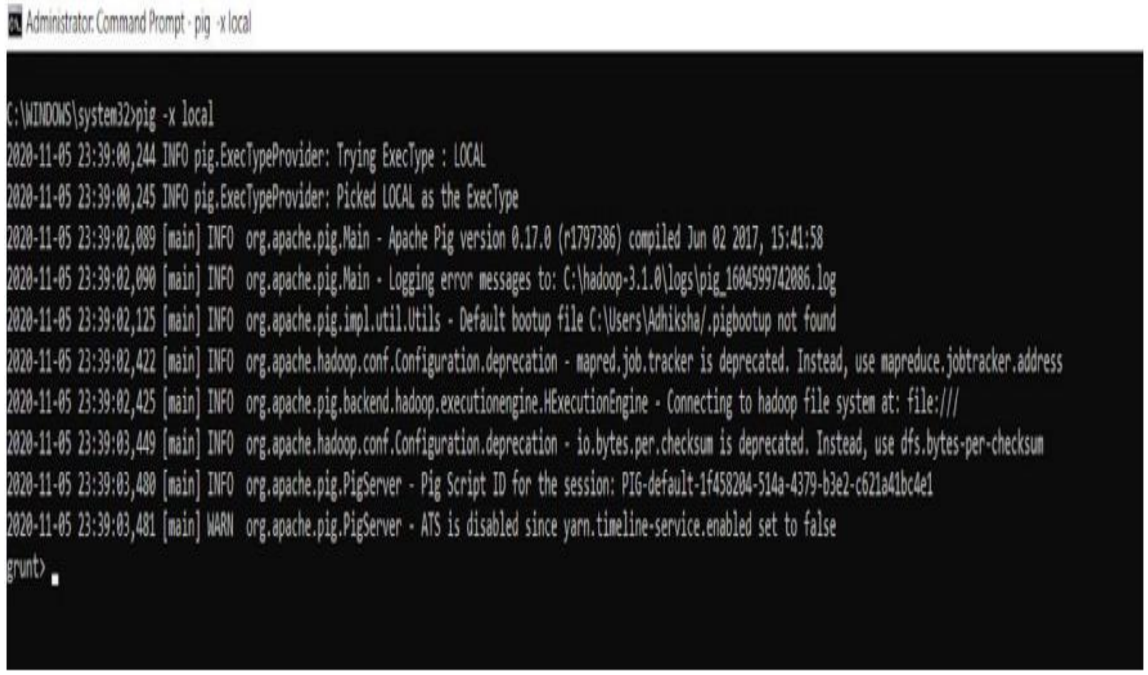
```
E:\>pig -version
Apache Pig version 0.17.0 (r1797386)
compiled Jun 02 2017, 15:41:58
```

The simplest way to write PigLatin statements is using Grunt shell which is an interactive tool where we write a statement and get the desired output.

There are 2 Ways of Invoking the grunt shell:

Local Mode: All the files are installed, accessed, and run in the local machine itself. No need to use HDFS. The command for running Pig in local mode is as follows.

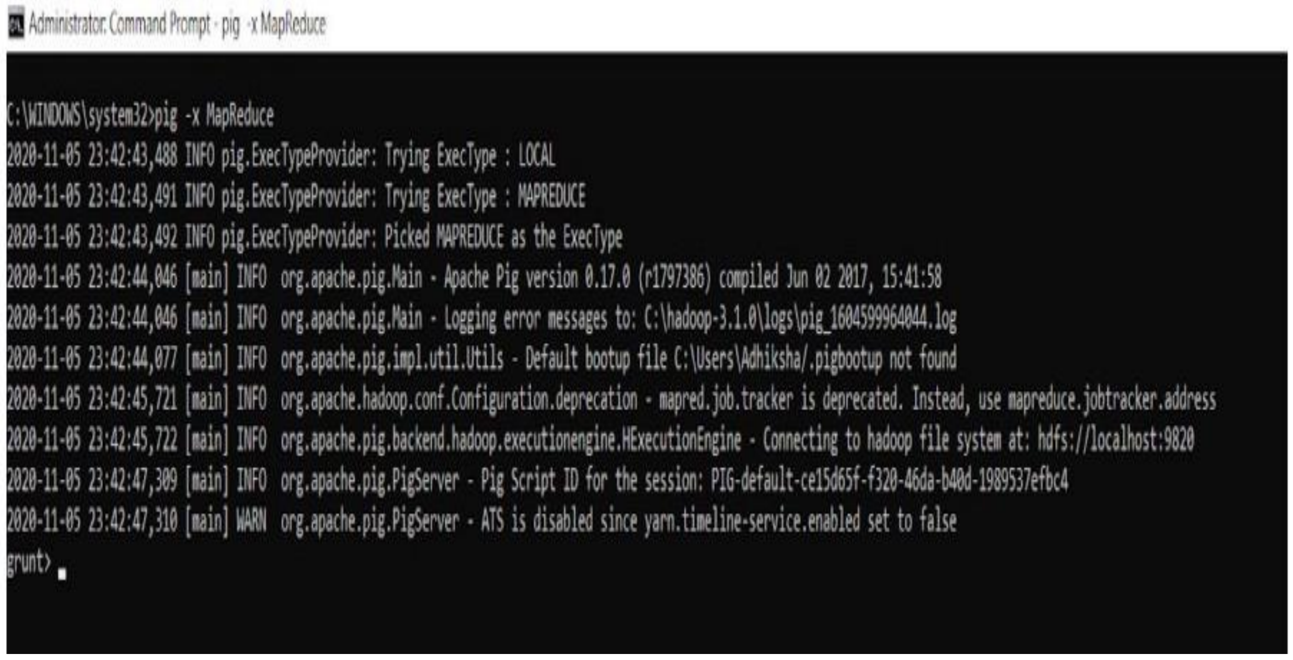
pig -x local



```
Administrator: Command Prompt - pig -x local

C:\WINDOWS\system32>pig -x local
2020-11-05 23:39:00,244 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
2020-11-05 23:39:00,245 INFO pig.ExecTypeProvider: Picked LOCAL as the ExecType
2020-11-05 23:39:02,089 [main] INFO  org.apache.pig.Main - Apache Pig version 0.17.0 (r1797386) compiled Jun 02 2017, 15:41:58
2020-11-05 23:39:02,090 [main] INFO  org.apache.pig.Main - Logging error messages to: C:\hadoop-3.1.0\logs\pig_1604599742086.log
2020-11-05 23:39:02,125 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file C:\Users\Adhiksha/.pigbootup not found
2020-11-05 23:39:02,422 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2020-11-05 23:39:02,425 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: file:///
2020-11-05 23:39:03,449 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2020-11-05 23:39:03,480 [main] INFO  org.apache.pig.PigServer - Pig Script ID for the session: PIG-default-1f458204-514a-4379-b3e2-c621a41bc4e1
2020-11-05 23:39:03,481 [main] WARN  org.apache.pig.PigServer - ATS is disabled since yarn.timeline-service.enabled set to false
grunt>
```

MapReduce Mode: The files are all present on the HDFS . We need to load this data to process it. The command for running Pig in MapReduce/HDFS Mode is as follows.

pig -x mapreduce



Administrator: Command Prompt - pig -x MapReduce

```
C:\WINDOWS\system32>pig -x MapReduce
2020-11-05 23:42:43,488 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
2020-11-05 23:42:43,491 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
2020-11-05 23:42:43,492 INFO pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2020-11-05 23:42:44,046 [main] INFO  org.apache.pig.Main - Apache Pig version 0.17.0 (r1797386) compiled Jun 02 2017, 15:41:58
2020-11-05 23:42:44,046 [main] INFO  org.apache.pig.Main - Logging error messages to: C:\hadoop-3.1.0\logs\pig_1604599964044.log
2020-11-05 23:42:44,077 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file C:\Users\Adhiksha/.pigbootup not found
2020-11-05 23:42:45,721 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2020-11-05 23:42:45,722 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: hdfs://localhost:9820
2020-11-05 23:42:47,309 [main] INFO  org.apache.pig.PigServer - Pig Script ID for the session: PIG-default-ce15d65f-f320-46da-b40d-1989537efbc4
2020-11-05 23:42:47,310 [main] WARN  org.apache.pig.PigServer - ATS is disabled since yarn.timeline-service.enabled set to false
grunt>
```

Program:

Download the dataset containing the Agriculture related data about crops in various regions and their area and produce. The link for dataset –https://www.kaggle.com/abhinand05/crop-production-in-india The dataset contains 7 columns namely as follows.

State_Name : chararray ;

District_Name : chararray ;

Crop_Year : int ;

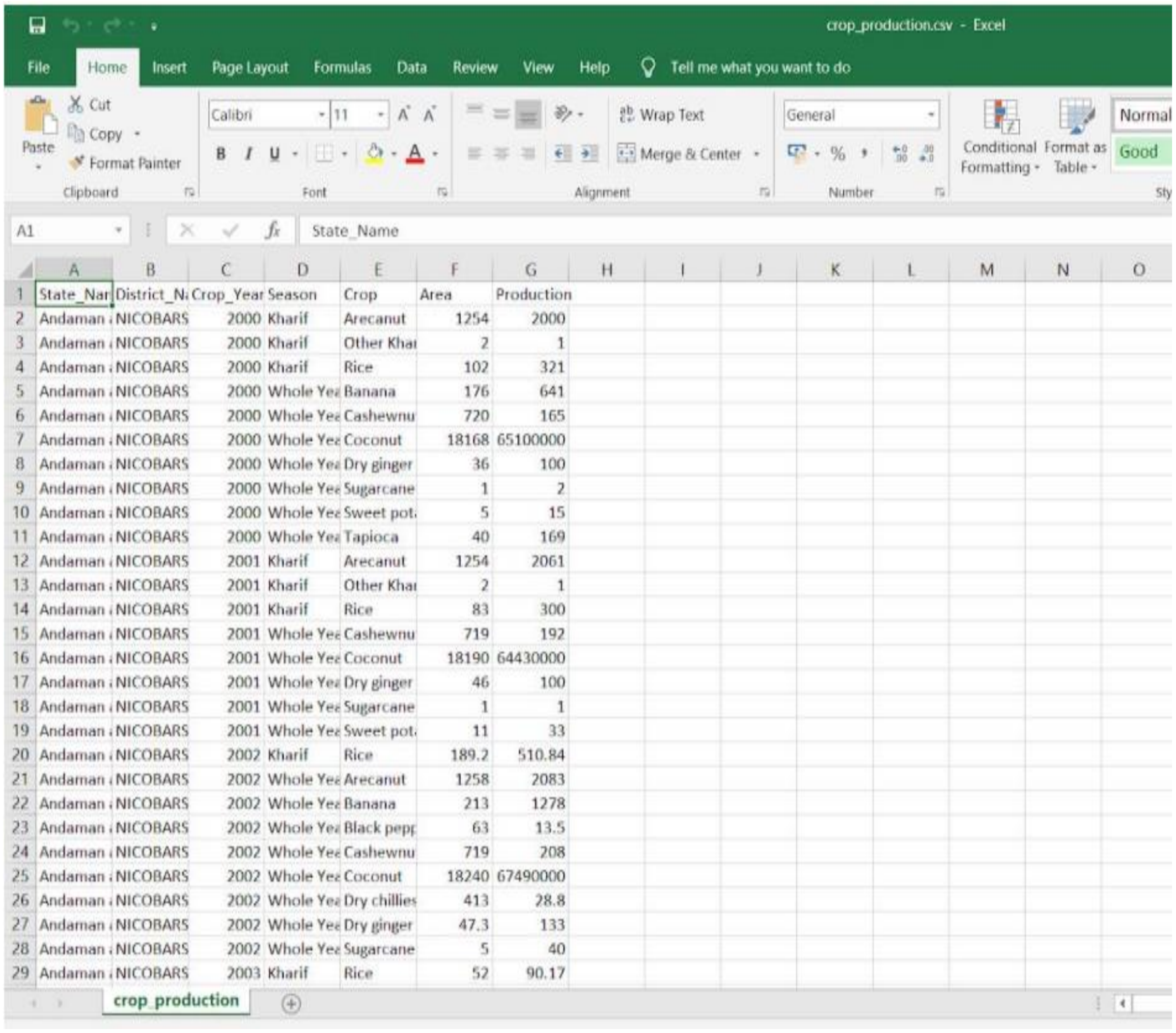
Season : chararray ;

Crop : chararray ;

Area : int ;

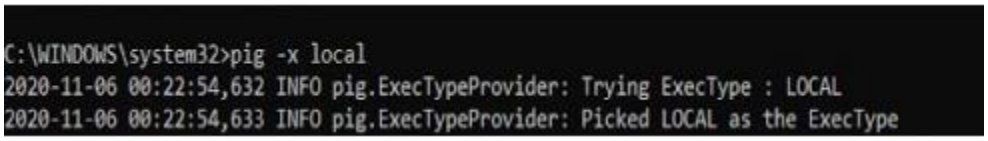
Production : int

No of rows: 246092
No of columns: 7



	A	B	C	D	E	F	G
1	State_Nar	District_N	Crop_Year	Season	Crop	Area	Production
2	Andaman	NICOBARS	2000	Kharif	Arecanut	1254	2000
3	Andaman	NICOBARS	2000	Kharif	Other Khar	2	1
4	Andaman	NICOBARS	2000	Kharif	Rice	102	321
5	Andaman	NICOBARS	2000	Whole Yea	Banana	176	641
6	Andaman	NICOBARS	2000	Whole Yea	Cashewnu	720	165
7	Andaman	NICOBARS	2000	Whole Yea	Coconut	18168	65100000
8	Andaman	NICOBARS	2000	Whole Yea	Dry ginger	36	100
9	Andaman	NICOBARS	2000	Whole Yea	Sugarcane	1	2
10	Andaman	NICOBARS	2000	Whole Yea	Sweet pot	5	15
11	Andaman	NICOBARS	2000	Whole Yea	Tapioca	40	169
12	Andaman	NICOBARS	2001	Kharif	Arecanut	1254	2061
13	Andaman	NICOBARS	2001	Kharif	Other Khar	2	1
14	Andaman	NICOBARS	2001	Kharif	Rice	83	300
15	Andaman	NICOBARS	2001	Whole Yea	Cashewnu	719	192
16	Andaman	NICOBARS	2001	Whole Yea	Coconut	18190	64430000
17	Andaman	NICOBARS	2001	Whole Yea	Dry ginger	46	100
18	Andaman	NICOBARS	2001	Whole Yea	Sugarcane	1	1
19	Andaman	NICOBARS	2001	Whole Yea	Sweet pot	11	33
20	Andaman	NICOBARS	2002	Kharif	Rice	189.2	510.84
21	Andaman	NICOBARS	2002	Whole Yea	Arecanut	1258	2083
22	Andaman	NICOBARS	2002	Whole Yea	Banana	213	1278
23	Andaman	NICOBARS	2002	Whole Yea	Black pepp	63	13.5
24	Andaman	NICOBARS	2002	Whole Yea	Cashewnu	719	208
25	Andaman	NICOBARS	2002	Whole Yea	Coconut	18240	67490000
26	Andaman	NICOBARS	2002	Whole Yea	Dry chillies	413	28.8
27	Andaman	NICOBARS	2002	Whole Yea	Dry ginger	47.3	133
28	Andaman	NICOBARS	2002	Whole Yea	Sugarcane	5	40
29	Andaman	NICOBARS	2003	Kharif	Rice	52	90.17

2. Enter pig local mode using
grunt > pig -x local

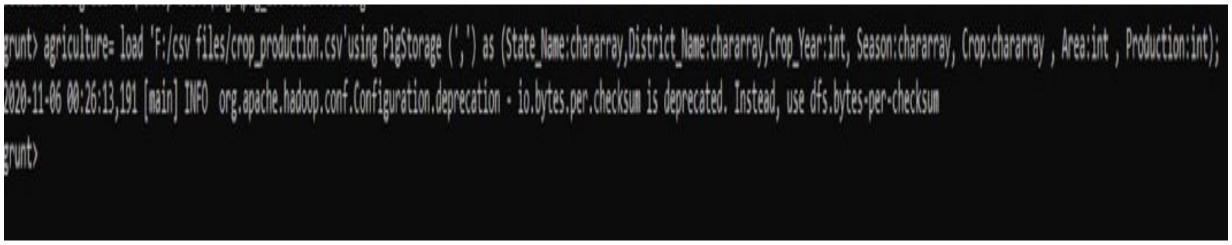


```
C:\WINDOWS\system32>pig -x local
2020-11-06 00:22:54,632 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
2020-11-06 00:22:54,633 INFO pig.ExecTypeProvider: Picked LOCAL as the ExecType
```

3. Load the dataset in the local mode
grunt > agriculture= LOAD 'F:/csv files/crop_production.csv' using PigStorage (',')

as (State_Name:chararray , District_Name:chararray , Crop_Year:int ,

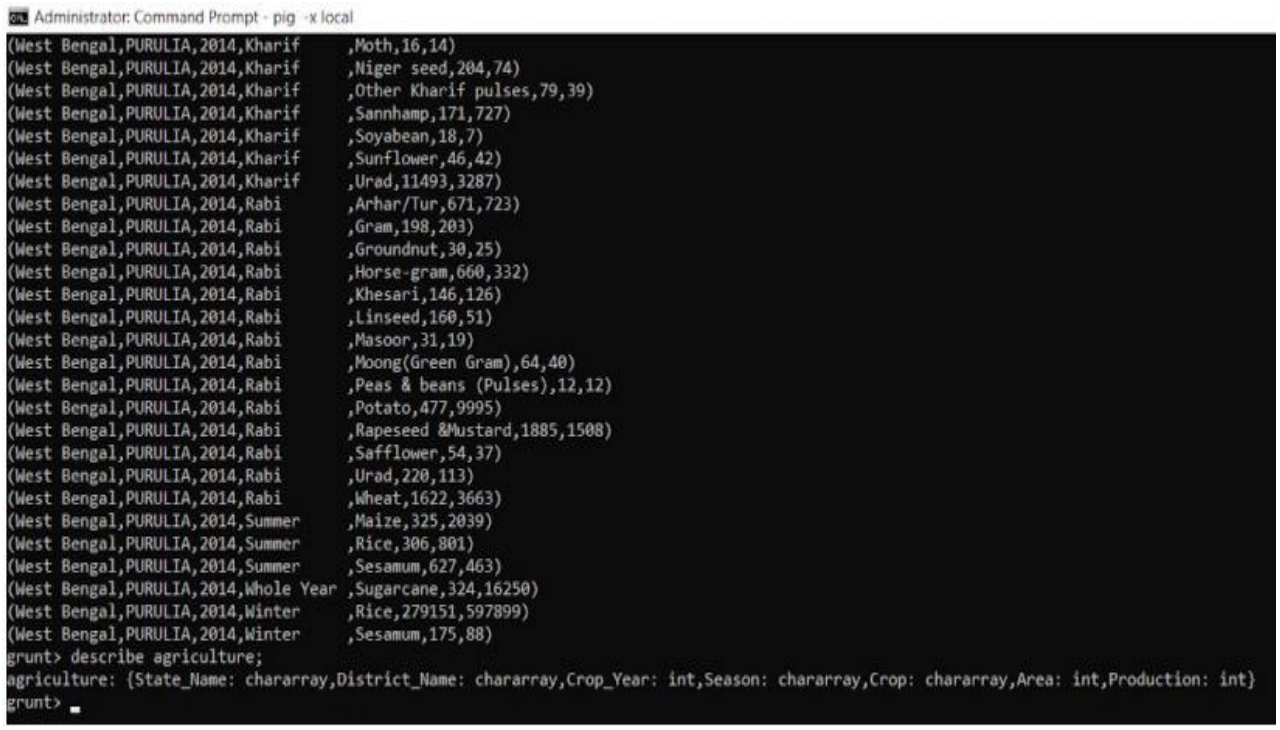
Season:chararray , Crop:chararray , Area:int , Production:int) ;



```
grunt> agriculture= load 'F:/csv files/crop_production.csv'using PigStorage (',') as (State_Name:chararray,District_Name:chararray,Crop_Year:int, Season:chararray, Crop:chararray , Area:int , Production:int);
2020-11-06 00:26:13,191 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
grunt>
```

4. Dump and describe the data set agriculture using
grunt > dump agriculture;

grunt > describe agriculture;



```
Administrator: Command Prompt - pig -x local
(West Bengal,PURULIA,2014,Kharif     ,Moth,16,14)
(West Bengal,PURULIA,2014,Kharif     ,Niger seed,204,74)
(West Bengal,PURULIA,2014,Kharif     ,Other Kharif pulses,79,39)
(West Bengal,PURULIA,2014,Kharif     ,Sannhamp,171,727)
(West Bengal,PURULIA,2014,Kharif     ,Soyabean,18,7)
(West Bengal,PURULIA,2014,Kharif     ,Sunflower,46,42)
(West Bengal,PURULIA,2014,Kharif     ,Urad,11493,3287)
(West Bengal,PURULIA,2014,Rabi       ,Arhar/Tur,671,723)
(West Bengal,PURULIA,2014,Rabi       ,Gram,198,203)
(West Bengal,PURULIA,2014,Rabi       ,Groundnut,30,25)
(West Bengal,PURULIA,2014,Rabi       ,Horse-gram,660,332)
(West Bengal,PURULIA,2014,Rabi       ,Khesari,146,126)
(West Bengal,PURULIA,2014,Rabi       ,Linseed,160,51)
(West Bengal,PURULIA,2014,Rabi       ,Masoor,31,19)
(West Bengal,PURULIA,2014,Rabi       ,Moong(Green Gram),64,40)
(West Bengal,PURULIA,2014,Rabi       ,Peas & beans (Pulses),12,12)
(West Bengal,PURULIA,2014,Rabi       ,Potato,477,9995)
(West Bengal,PURULIA,2014,Rabi       ,Rapeseed &Mustard,1885,1508)
(West Bengal,PURULIA,2014,Rabi       ,Safflower,54,37)
(West Bengal,PURULIA,2014,Rabi       ,Urad,220,113)
(West Bengal,PURULIA,2014,Rabi       ,Wheat,1622,3663)
(West Bengal,PURULIA,2014,Summer     ,Maize,325,2039)
(West Bengal,PURULIA,2014,Summer     ,Rice,306,801)
(West Bengal,PURULIA,2014,Summer     ,Sesamum,627,463)
(West Bengal,PURULIA,2014,Whole Year ,Sugarcane,324,16250)
(West Bengal,PURULIA,2014,Winter     ,Rice,279151,597899)
(West Bengal,PURULIA,2014,Winter     ,Sesamum,175,88)
grunt> describe agriculture;
agriculture: {State_Name: chararray,District_Name: chararray,Crop_Year: int,Season: chararray,Crop: chararray,Area: int,Production: int}
grunt>
```

Program:4

Perform Social media analysis using Cassandra

Aim: To Perform Social media analysis using Cassandra

Objective:

Apache Cassandra is a highly scalable, high-performance distributed database designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. It is a type of NoSQL database. Let us first understand what a NoSQL database does.

NoSQLDatabase

A NoSQL database (sometimes called as Not Only SQL) is a database that provides a mechanism to store and retrieve data other than the tabular relations used in relational databases. These databases are schema-free, support easy replication, have simple API, eventually consistent, and can handle huge amounts of data.

The primary objective of a NoSQL database is to have

- simplicity of design,
- horizontal scaling, and
- finer control over availability.

NoSql databases use different data structures compared to relational databases. It makes some operations faster in NoSQL. The suitability of a given NoSQL database depends on the problem it must solve.

Cassandra Query Language

Cassandra introduced the Cassandra Query Language (CQL). CQL is a simple interface for accessing Cassandra, as an alternative to the traditional Structured Query Language (SQL). CQL adds an abstraction layer that hides implementation details of this structure and provides native syntaxes for collections and other common encodings. Language drivers are available for Java (JDBC), Python (DBAPI2), Node.JS (Datastax), Go (gocql) and C++.

The keyspace in Cassandra is a namespace that defines data replication across nodes. Therefore, replication is defined at the keyspace level. Below an example of keyspace creation, including a column family in CQL 3.0

Example program

```
CREATE KEYSPACE MyKeySpace
  WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : 3 };

USE MyKeySpace;

CREATE COLUMNFAMILY MyColumns (id text, lastName text, firstName text, PRIMARY
KEY(id));

INSERT INTO MyColumns (id, lastName, firstName) VALUES ('1', 'Doe', 'John');

SELECT * FROM MyColumns;
```

Which gives:

```
 id | lastName | firstName
```

```
----+----------+----------
  1 | Doe      | John

(1 rows)
```

DEPARTMENT OF
INFORMATION TECHNOLOGY
DEPARTMENT OF
INFORMATION TECHNOLOGY