**10.1) Develop Pig Latin scripts to group, join, project, and filter the data**

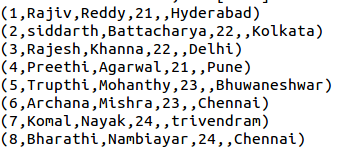
**1) LOAD:**

**Command:**

****

****

**Ouput:**

****

**2) FILTER:**

**Command:**

****

**Ouput:**

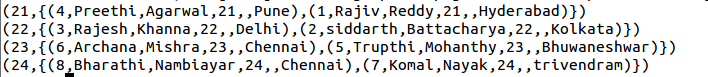
****

**3)GROUP:**

**Command:**

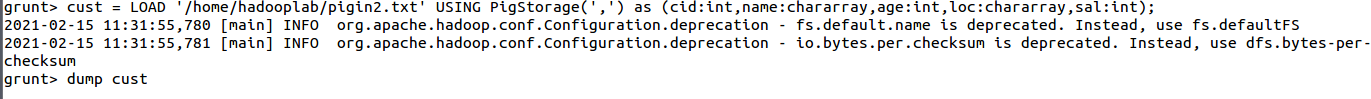
****

**Ouput:**

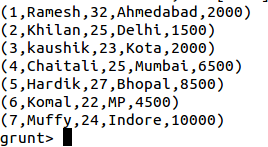
****

**4) INNER JOIN:**

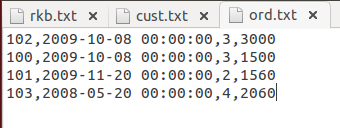
**Command:**

****

**Ouput:**

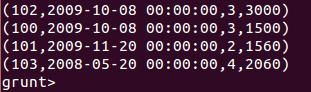
****

**Input file:**



**Command:**

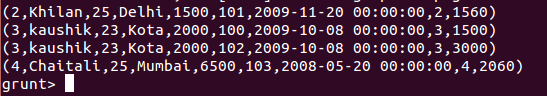
**Ouput:**



**Command:**



**Ouput:**



**11.1) Install Hive Framework**

**Hive Installation on Ubuntu:**

Please follow the below steps to install **Apache Hive**on Ubuntu:

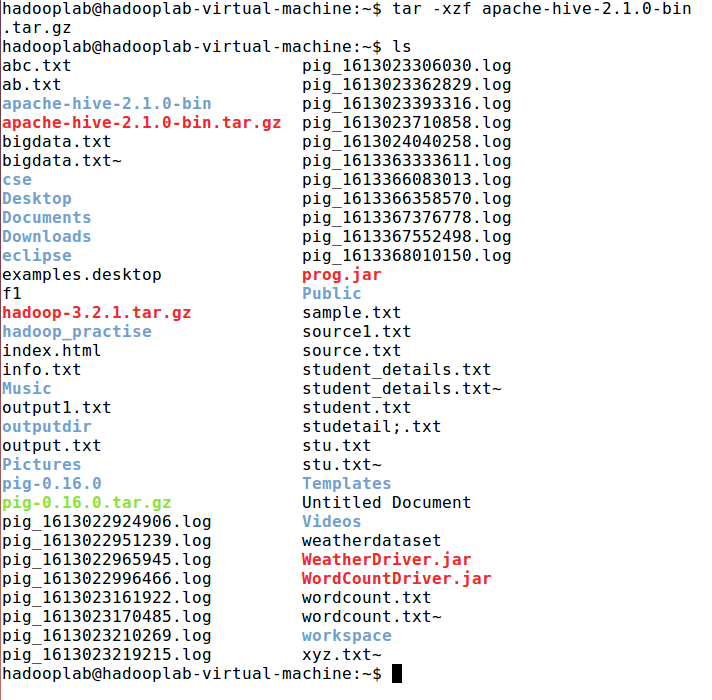
**Step 1:**  Download **Hive tar.**

**Command:**wget http://archive.apache.org/dist/hive/hive-2.1.0/apache-hive-2.1.0-bin.tar.gz

**Step 2:**Extract the **tar** file.

**Command:**tar -xzf apache-hive-2.1.0-bin.tar.gz

**Command:** ls



**Step 3:**Edit the **“.bashrc”** file to update the environment variables for user.

**Command:**sudo gedit .bashrc



Add the following at the end of the file:

***# Set HIVE\_HOME***

***export HIVE\_HOME=/home/hadooplab/apache-hive-2.1.0-bin***  
***export PATH=$PATH:/home/ hadooplab/apache-hive-2.1.0-bin/bin***

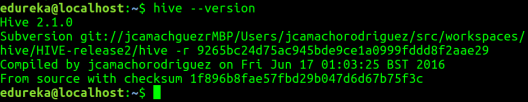
Also, make sure that hadoop path is also set.



Run below command to make the changes work in same terminal.

**Command:** source .bashrc

**Step 4:** Check hive version.



**Step 5:**Create **Hive** directories within**HDFS**. The directory **‘warehouse’** is the location to store the table or data related to hive.

**Command:**

* hdfs dfs -mkdir -p /user/hive/warehouse
* hdfs dfs -mkdir /tmp

**Step 6:**Set read/write permissions for table.

**Command:**

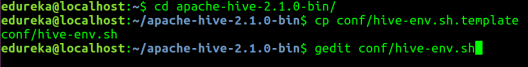
In this command, we are giving write permission to the group:

* hdfs dfs -chmod g+w /user/hive/warehouse
* hdfs dfs -chmod g+w /tmp

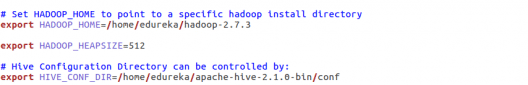
**Step 7:**Set **Hadoop** path in **hive-env.sh**

**Command:**cd apache-hive-2.1.0-bin/

**Command:**gedit conf/hive-env.sh



Set the parameters as shown in the below snapshot.



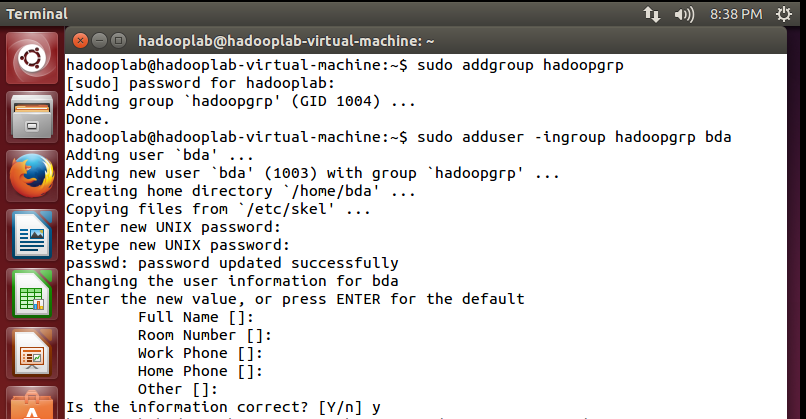
**3) Hadoop Standalone mode Installation**

**3.1) Perform setting up and Installing Hadoop in the Standalone mode**

**Creating a User in Ubuntu:**

At the beginning, it is recommended to create a separate user for Hadoop to isolate Hadoop file system from Unix file system.

In addition to this, as we need to prepare cluster, first create group and then a user in that group.

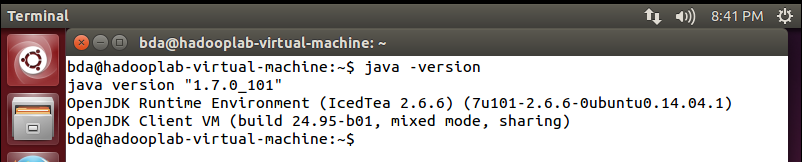


Logout and switched to the new user account currently created(bda)

Pwd:admin

**I. Local Standalone mode**

STEP 1.To Check if the java is already installed or not



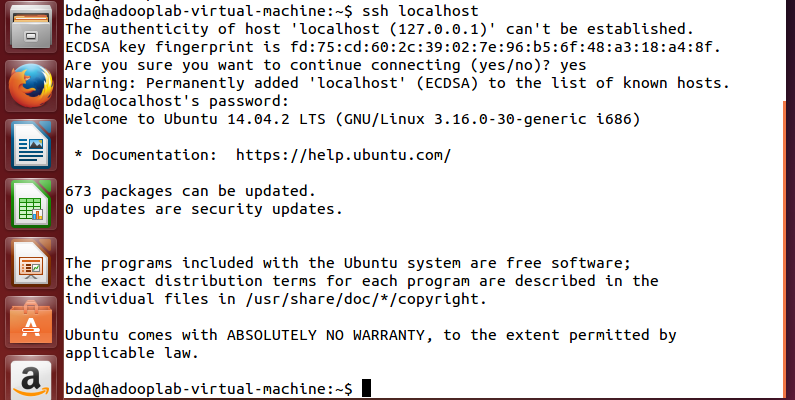
IF NOT java is installed by following commands we can install java

$sudo apt-get install default –jre

$sudo apt-get install default –jdk

**STEP 2:SSH Setup and key Generation in Ubuntu**

SSH setup is required to do different operations on a cluster such as starting, stopping, distributed daemon shell operations. To authenticate different users of Hadoop, it is required to provide public/private key pair for a Hadoop user and share it with different users. $ sshlocalhostAfter Typing this you must be able to connect to localhost and able to see the following output.

****

Otherwise use the following commands

$ sudo apt-get install openssh-server

$ sudo apt-get install vsftpd

$ sshlocalhost

The following commands are used for generating

1. A key value pair using SSH.
2. Copy the public keys from id\_rsa.pub to authorized\_keys and
3. Provide the owner with read and write permissions to authorized\_keys file respectively.

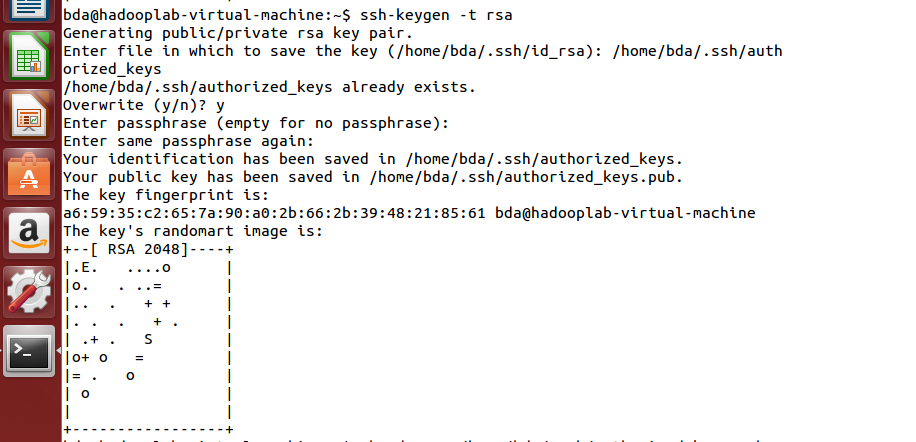
$ ssh-keygen -t rsa

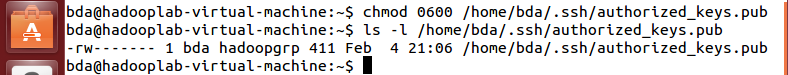
$ cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys

$ chmod 0600 ~/.ssh/authorized\_keysssh-keygen -t rsa

$ cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys

$ chmod 0600 ~/.ssh/authorized\_keys

****

****

**STEP 3: INSTALLING JAVA**

$ uname –i

As Java is already installed in your computer, You will get either

* + 1. x86\_64 or
    2. i686 or
    3. something else

**Then open a text editor using search in ubuntu.Create a file abc.txt**

**Those who got** x86\_64, type these two lines into abc.txt

export JAVA\_HOME=/usr/lib/jvm/java-7-openjdk-amd64

export PATH=$PATH:$JAVA\_HOME/bin

**Those who got** other than x86\_64, type these two lines into abc.txt

export JAVA\_HOME=/usr/lib/jvm/java-7-openjdk-i386

export PATH=$PATH:$JAVA\_HOME/bin

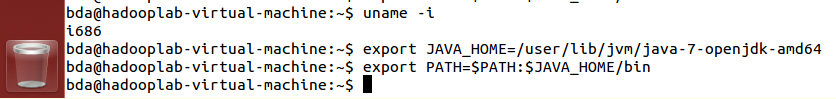
**STEP 4:DOWNLOAD HADOOP**

As Hadoop is already downloaded in one computer in this lab, you need to copy that downloaded Hadoop into your computer. So follow the following commands to do this.

$ pwd (It must be nitw\_viper\_user)

$ wget ftp://172.168.10.168/Downloads/hadoop-2.7.2.tar.gz --user=lenova --password=nitw

By doing this, the download Hadoop software which is in the tar form will be copied into ‘nitw\_viper\_user’.

****

**Note:tar.gz =zip format**

**STEP 5: Untar (un ziping) the Hadoop**

To untar (like unzipping) the Hadoop in nitw\_cvhd\_user follow the following commands.

$ tar zxf hadoop-2.7.2.tar.gz

$ ls –lrt

**STEP 6: Setting the Hadoop path**

Just like your Java path, Now we are doing it for Hadoop path.Open the file abc.txt, where you have already typed two lines in that previouslyType the following two lines into abc.txt

export HADOOP\_HOME=/home/acet\_viper\_user/hadoop-2.7.2

export PATH=$PATH:$HADOOP\_HOME/bin

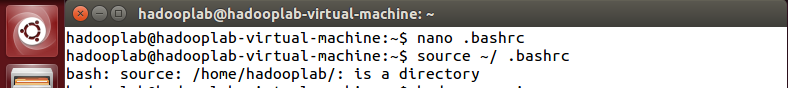
**STEP 7: Updating the bashrc**

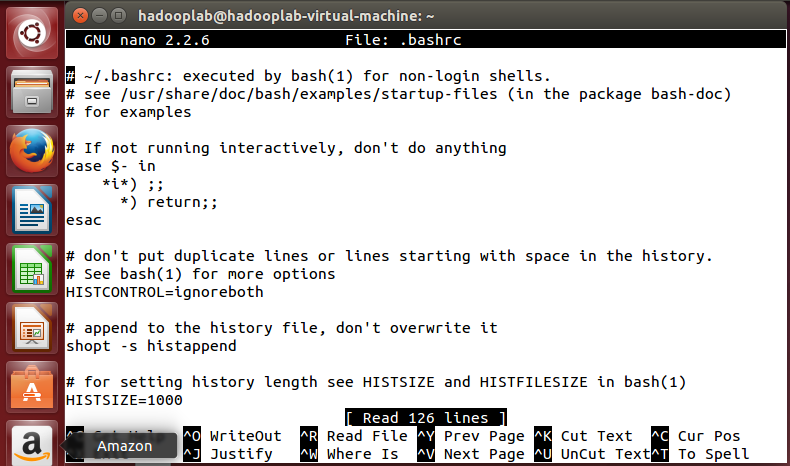
To set the environmental changes, i.e. to keep the java path and hadoop path to work correctly in all ‘terminals’, we have to save the changes to ‘bashrc file’ of the current user. Actually, bashrc is a hidden file. So to open this file, we have to use .bashrc.Use the following commands to do this.

$ nano .bashrc

bashrc will be opened,

* go to the end of the file
* paste the 4-lines from abc.txt into this bashrc file
* Save the bashrc file

****

****

**Step-8:Now use the command**

$ source ~/.bashrc

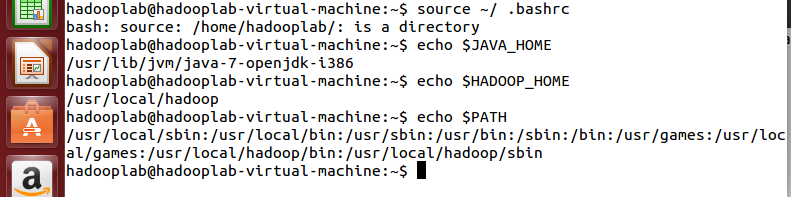
(This command is to refresh the terminal with updated bashrc)

**Step-9: Just verify, whether everything is done properly or not , to check the path settings are reflected or not.**

$ echo $JAVA\_HOME

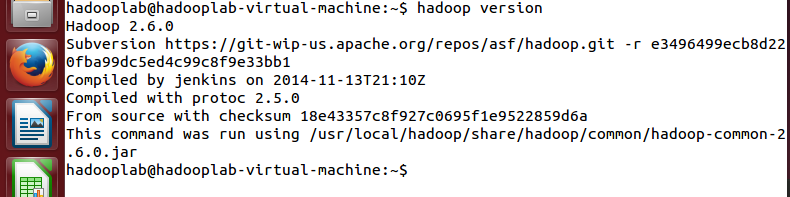
$ echo $HADOOP\_HOME

$ echo $PATH (It will show both Java path and Hadoop Path)

****

**Step-10: Now, Check whether Hadoop is working or not. (Just like Java -version)**

$ hadoop version

****

Now Hadoop on Local Standalone mode is Ready

**Now, We run a program on Hadoop from the already existing examples given.**

Let's have an input directory where we will push a few files and our requirement is to count the total number of words in those files. To calculate the total number of words, we do not need to write our MapReduce, provided the .jar file contains the implementation for word count. You can try other examples using the same .jar file; just issue the following commands to check supported MapReduce functional programs by hadoop-mapreduce-examples-2.7.1.jar file.

$ cp $HADOOP\_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar .

(This dot is must, i.e. to copy into present working director)

This command copies the examples into present working directory i.e. nitw\_viper\_user

$ ls –lrt

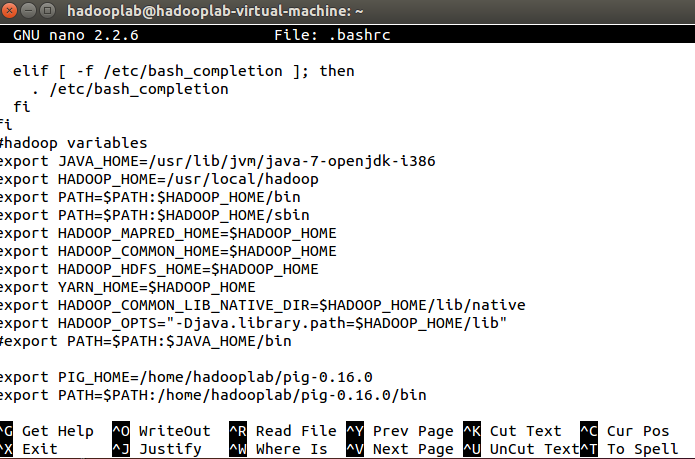
**4) Hadoop Pseudo distributed mode Installation**

**4.1) Perform setting up and Installing Hadoop in the Pseudo distributed mode.**

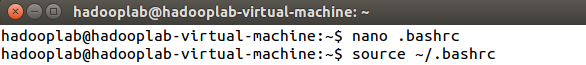
**STEP 1:** open bash file using nano command set Hadoop environment variables by appending the following commands to **.bashrc and then save**

****

**STEP 2:** check the following commands are exist in bashrc file or not if not append in the bash file

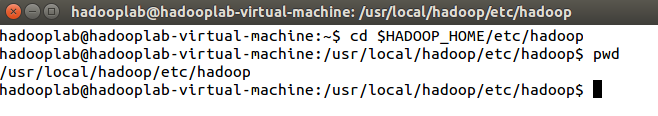
****

**STEP 3:** To exit that bashrc file ctrl+x is press.This command is to refresh the terminal with updated bashrc

****

**STEP 4:**You can find all the Hadoop configuration files in the location “$HADOOP\_HOME/etc/hadoop”.

It is required to make changes in those configuration files according to your Hadoop infrastructure**.** Now successfully change the directory

****

**STEP 5:** In order to develop Hadoop programs in java, you have to reset the java environment

variables in hadoop-env.sh file by replacing JAVA\_HOME value with the location of java in your system.

$ nano hadoop-env.sh Those who got x86\_64 (uname -i), type &export JAVA\_HOME=/usr/lib/jvm/java-7-openjdk-amd64 & Write this statement at the end of the file

Those who got other than x86\_64, type

export JAVA\_HOME=/usr/lib/jvm/java-7-openjdk-i386 & after this we exist the file by pressing ctrl+x

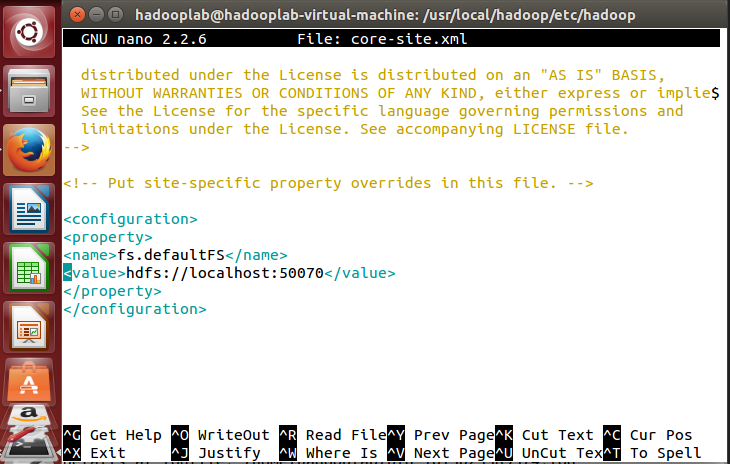




**STEP 6:** You need to configure the following files also

core-site.xml, hdfs-site.xml, yarn-site-xml, mapred-site.xml Now Configuring core-site.xmlThe core-site.xml file contains information such as the port number used forHadoop instance, memory allocated for the file system, memory limit for storingthe data, and size of Read/Write buffers. Open the core-site.xml and add the following properties in between

<configuration>, </configuration> tags.

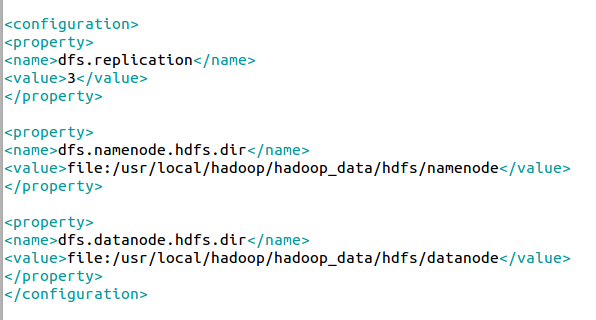


**Step 7: Configuring hdfs-site.xml :**Using $ nano hdfs-site.xmlThe hdfs-site.xml file contains information such as the value of replication data, namenode

path, and datanode paths of your local file systems. It means the place where you want to

store the Hadoop infrastructure.



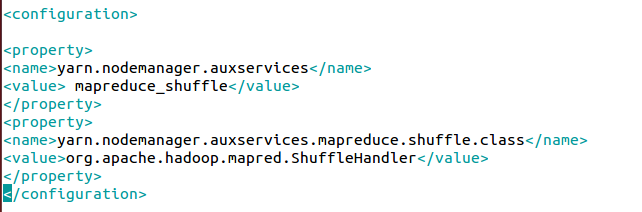


**Step 8: Configuring yarn-site.xml :**Using $ nano yarn-site.xml

This file is used to configure yarn into Hadoop. Open the yarn-site.xml file and add the following properties in between the

<configuration>, </configuration> tags in this file.

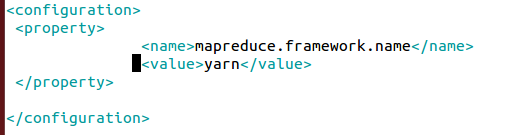




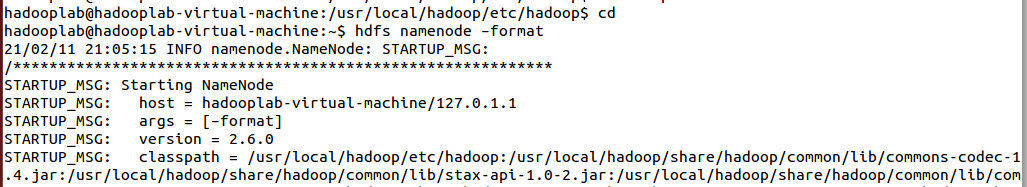
**Step 9: Configuring mapred-site.xml :** This file is used to specify which MapReduce framework we are using. By default, Hadoop contains a template of mapred-site.xml.template First of all, it is required to copy the file from mapred-site,xml.template to mapred-site.xml file using the following command.



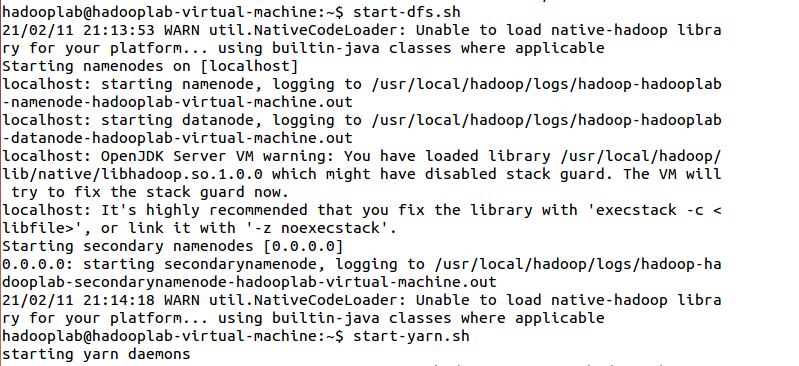


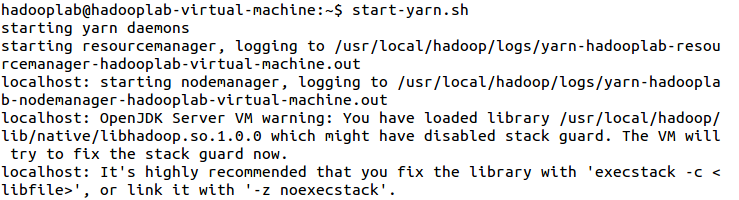


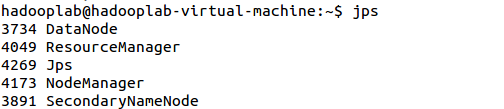
**Step-10:** Verifying HadoopInstallation Goto home directory. Using these two commands $ cd $ hdfsnamenode –format



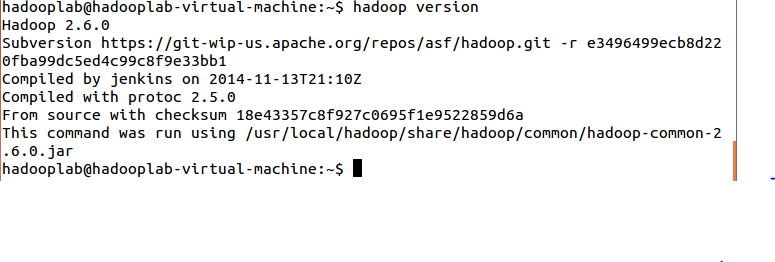
**Step-11:** Verifying HadoopdfsGoto home directory using these three commands $ start–dfs.sh ,$ start–yarn.sh , $ jps







**STEP -12 :**Before proceeding further, you need to make sure that Hadoop is working fine. Just issue the following command $ hadoopversion If everything is fine with your setup, then you should see the following result: It means your Hadoop's pseudo distributed mode setup is working fine. By default, Hadoop is configured to run in a non-distributed mode on a single machine.

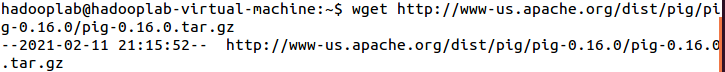


**9) PigLatin Script**

**9.1) Install Pig Latin Software**

**Step 1:**  Download **Pig tar** file

**Command:**wget http://www-us.apache.org/dist/pig/pig-0.16.0/pig-0.16.0.tar.gz

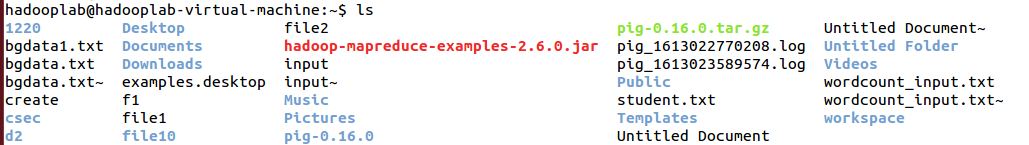


**Step 2:**Extract the **tar** file using tar command. In below tar command, **x** means extract an archive file,**z** means filter an archive through gzip,**f** means filename of an archive file. And check ls command also.

**Command:**tar -xzf pig-0.16.0.tar.gz

**Command:** ls





**Step 3:**Edit the “**.bashrc**” file to update the environment variables of Apache Pig. We are setting it so that we can access pig from any directory, we need not go to pig directory to execute pig commands. Also, if any other application is looking for Pig, it will get to know the path of Apache Pig from this file.

**Command:**sudogedit .bashrc

Add the following at the end of the file:

***# Set PIG\_HOME***

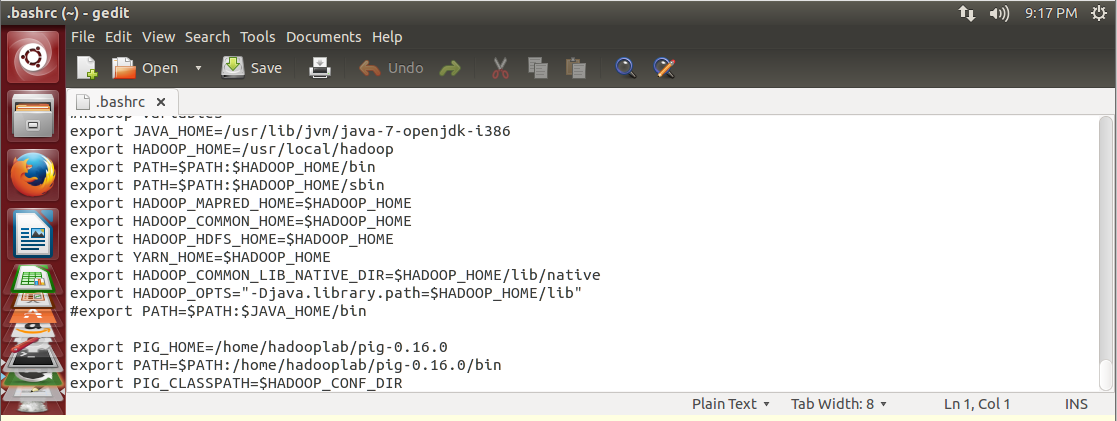
***export PIG\_HOME=/home/hadooplab/pig-0.16.0***  
***export PATH=$PATH:/home/hadooplab/pig-0.16.0/bin***  
***export PIG\_CLASSPATH=$HADOOP\_CONF\_DIR***

Also, make sure that hadoop path is also set.

Run below command to make the changes get updated in same terminal.

**Command:** source .bashrc







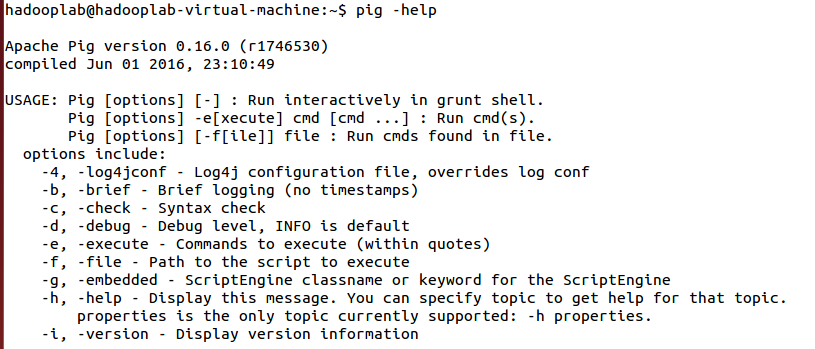
**Step 4:** Check pig version. This is to test that Apache Pig got installed correctly. In case, you don’t get the Apache Pig version, you need to verify if you have followed the above steps correctly.

**Command:** pig -version

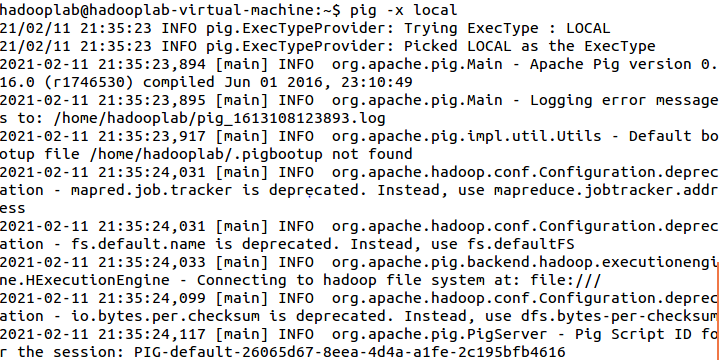


**Step 5**: Check pig help to see all the pig command options.

**Command:**pig -help

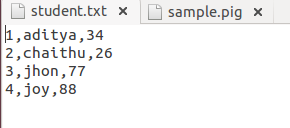
**Step 6**: Run Pig to start the grunt shell. Grunt shell is used to run Pig Latin scripts.

**Command:** pig –x local

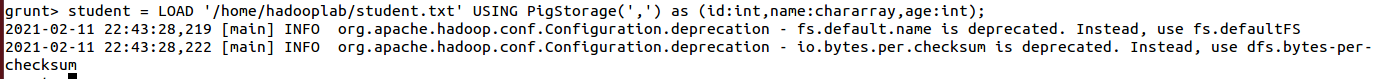


**9.2)Develop Pig Latin scripts to load the data and sort your data.**

**Step 1:**Arrange input file input using gedit filename is student.txt and type the data as bag format example is shown below.

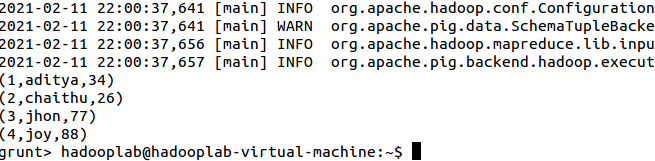


**Step 2:** load input file into hadoop file sytem using load command.



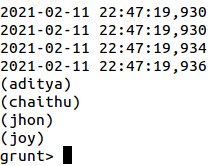
**Step 3:**  Now diplay the input file using dump command. Ouput is shown below





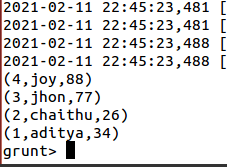
**Step 4:** Now display only names field from bag which is in input file all these done using foreachcommand .output is shown below.





**Step 5:** Now display sorted bag by age field which is in input file all these done using order command . output is shown below.

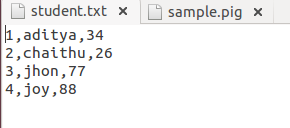




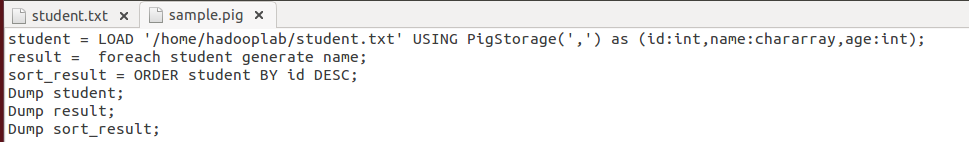
**Step 6:** Now all these done pig script means in single file. Input file, script file and output is shown below. To run the pig script following command is used in grunt shell

grunt>run /home/hadooplab/sample.pig

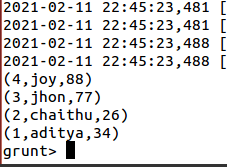
**INPUT FILE:**



**PIG SCRIPT FILE:**

****

**OUTPUT**:



**7) MapReduce-WordCount;**

**STEPS TO EXCUTE:**

1. WordCountDriver.java

2.Mapper.java

3.Reducer.java

IDE:Eclipse IDE –New Project-class- WordCountDriver.java

Installing Eclipse on ubantu:

Java Installation:

Check whether java is installed or not

1.$java -version

$sudo apt-get install openjdk-8-jdk

Password:

2.In Google,Eclipse IDE

Opens home page of Eclipse Community and there is a Download option click on it.

3.Download64bit option click on it.It will download eclipse-inst-linux64.tar.gz

4. Extract eclipse folder from .tar.gz

5. After Extracting, go to installer folder,then select eclipse-inst,right click and choose run

6. Select required Developer->Eclipse IDE for Java Developer,then click on next

7.Click on Install(Progress Bar displays)

8.Select directory as a Workspace(C:\ProgramFiles\EclipseIDE\....)

How to Execute Map Reduce Program:

1.Create a Project-Wordcount-src-class-

WordCountDriver.java

2.Type 3 classes Mapper,Reducer,Driver classes & Save .

3.To create a jar file,go to Project,click on Export

4.A window JAR EXPORT-select the export destination JAR file:

5.Select the resources to export-classpath,project(deselect)

6.Specify the manifest file

MainClass: WordCountDriver.class

Execute WordCount Program in Hadoop(command prompt):

$hadoop fs –mkdir /hadoopfiles

$hadoop fs –put /home/hadooplab/hadoop.txt /hadoopfiles/source.txt

$hadoop jar WordCountDriver.jar /hadoopfiles/source.txt /hadoopfiles/output.txt

To get output.txt from hadoop system to local system:

$hadoop fs –get /hadoopfiles/output.txt /home/hadooplab/outputdir

$hadoop fs –get /hadoopfiles/output.txt /home/hadooplab/outputdir

OUTPUT:

Source.txt:

HDFS is a storage unit of hadoop

MapReduce is a Processing tool of hadoop

To view outputfile, type following command:

$hadoop fs –cat / outputdir /part–00000

HDFS 1

Hadoop 2

MapReduce 1

a 2

is 2

of 2

processing 1

storage 1

tool 1

unit

**8.1)WRITE A MAP REDUCE APPLICATION TO FIND THE MAXIMUM TEMPERATURE IN THE WEATHER DATASET.**

**WEATHERDRIVER : PROGRAM**

importjava.io.IOException;

importorg.apache.hadoop.conf.Configuration;

importorg.apache.hadoop.fs.Path;

importorg.apache.hadoop.io.IntWritable;

importorg.apache.hadoop.io.Text;

importorg.apache.hadoop.mapred.FileOutputFormat;

importorg.apache.hadoop.mapred.JobConf;

importorg.apache.hadoop.mapred.TextOutputFormat;

importorg.apache.hadoop.mapreduce.Job;

importorg.apache.hadoop.mapreduce.lib.input.FileInputFormat;

publicclassWeatherDriver {

publicstaticvoid main(String args[]) {

JobConfconf=newJobConf();

Job job;

try {

job=new~~Job~~(conf,"WeatherDataExtraction");

job.setJobName("WeatherDataExtraction");

job.setMapperClass(WeatherMapper.class);

job.setReducerClass(WeatherReducer.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

FileInputFormat.*addInputPath*(job, new Path(args[0]));

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

try {

job.waitForCompletion(true); }

catch (ClassNotFoundException | IOException | InterruptedException e) {

e.printStackTrace();} }

catch (IOException e) {

e.printStackTrace();

} } }

**WEATHERMAPPER : PROGRAM:**

importjava.io.IOException;

importorg.apache.hadoop.io.IntWritable;

importorg.apache.hadoop.io.LongWritable;

importorg.apache.hadoop.io.Text;

importorg.apache.hadoop.mapreduce.Mapper;

public class WeatherMapper extends Mapper<LongWritable,Text,Text,IntWritable>{

public void map(LongWritablekey,Textvalue,Context context) throws IOException, InterruptedException{

String line=value.toString();

String year=line.substring(15,19);

intairTemperature;

if (line.charAt(87)=='+'){

airTemperature=Integer.*parseInt*(line.substring(41,45));

}

else{

airTemperature=Integer.*parseInt*(line.substring(40,45));

}

String quality=line.substring(45,46);

if(quality.matches("[01459]")){

context.write(new Text(year), new IntWritable(airTemperature));

}}}

**WEATHERREDUCER : PROGRAM:**

importjava.io.IOException;

importorg.apache.hadoop.io.IntWritable;

importorg.apache.hadoop.io.Text;

importorg.apache.hadoop.mapreduce.Reducer;

publicclassWeatherReducerextends Reducer<Text,IntWritable,Text,IntWritable>{

publicvoid reduce(Text key,Iterable<IntWritable>values,Context context) throwsIOException,InterruptedException{

Integer max=newInteger(0);

for(IntWritableval:values){

if(val.get()>max.intValue()) {max=val.get();}

}

context.write(key, newIntWritable(max.intValue()));

} }

**OUTPUT:**

$hadoop fs –mkdir /hadoopfiles

$hadoop fs –put /home/hadooplab/hadoop.txt /hadoopfiles/weather–source.txt

$hadoop jar WeatherDriver.jar /hadoopfiles/ weather–source.txt /hadoopfiles/output.txt

$hadoop fs –cat /hadoopfiles/weather–source.txt

011990-99999-1950.gz

011990-99999-1950.gz

...

011990-99999-1950.gz

$ hadoop fs -cat result/part-r-000001901 317 1902 244

1903 289

1904 256

1905 283

1906 294

1907 283

1908 289

1909 278

1910 294

1911 306

1912 322

1913 300

1914 333

1915 294

1916 278

1917 317

1918 322

1919 378

1920 294

**Library Management System:**

1. **Usecase diagram:**



**Library Management System:**

1. **Class diagram:**



**Library Management System:**

1. **Sequence Diagram:**



**Library Management System:**

1. **Collaboration Diagram:**



1. **State Diagram:**



**Library Management System:**

1. **Activity Diagram:**



**Library Management System:**

1. **Component diagram**:



1. **Deployment diagram:**



**Online book Shop:**

1. **Usecase diagram:**



**Online book Shop:**

1. **Class diagram:**



**Online book Shop:**

1. **Activity diagram:**



**Online book Shop:**

1. **State chart diagram:**



1. **Component diagram:**



**Online book Shop:**

1. **Sequence Diagram**:

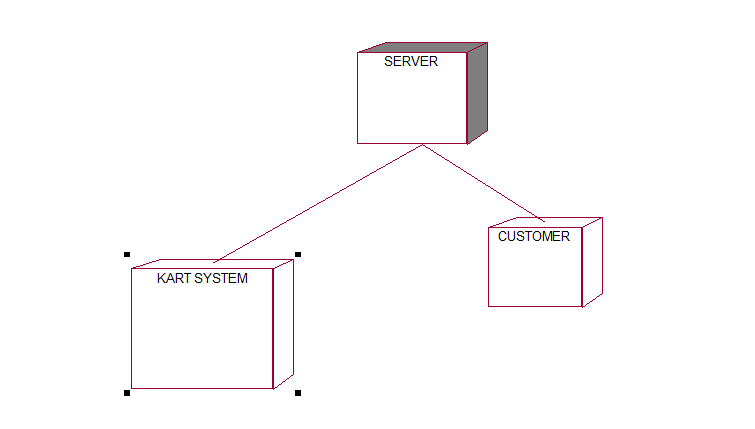


**Online book Shop:**

1. **Collaboration Diagram:**



**8)Deployment diagram:**



**ATM SYSTEM :**

1. **Usecase diagram:**



**ATM SYSTEM:**

1. **Class diagram:**



**ATM SYSTEM:**

1. **Sequence diagram:**



**ATM SYSTEM:**

1. **Collaboration diagram:**



**5)State chart diagram:**



**ATM SYSTEM :**

1. **Activity diagram**:



**ATM SYSTEM:**

1. **Component diagram:**



1. **Deployment diagram:**

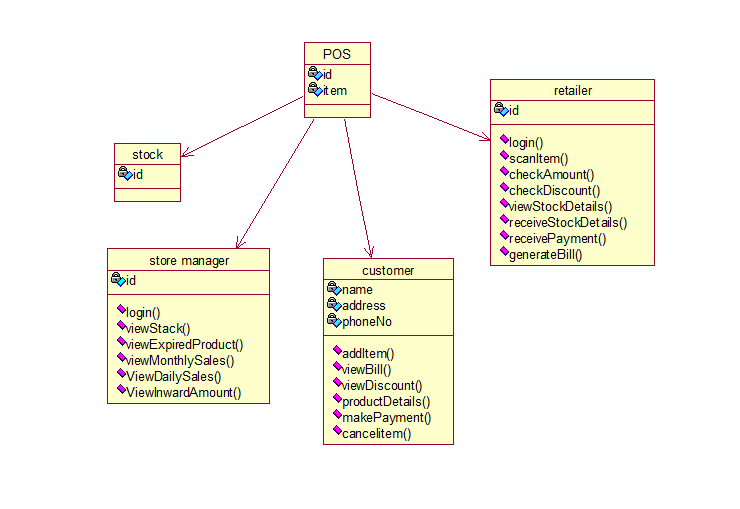


**POS:  
1) Usecase diagram:**



**POS**

**2)Class diagram:**

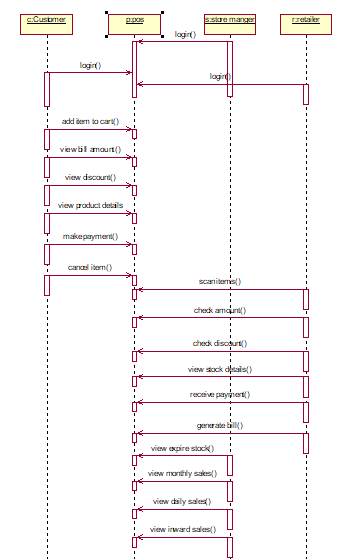


**POS:**

**3)State chart diagram:**



**POS:  
4)Sequence diagram:**

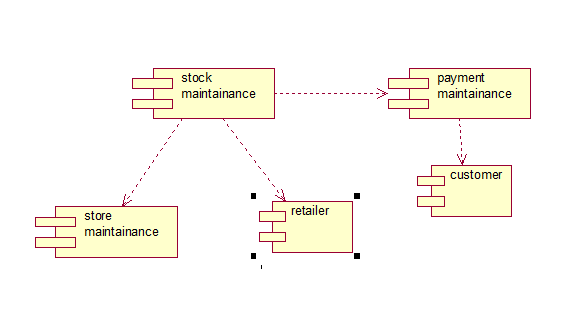


**POS:**

**6)State chart diagram:**



7)**Component diagram:**



**POS:**

**8)Deployment diagram:**

