HADOOP-ECO Systems

1. Install, configure and run Hadoop Eco Systems– Hadoop,Pig, Hive ,Hbase

a) Hadoop :

Hadoop is an open-source software framework for storing data and running applications on clusters of commodity hardware. It provides massive storage for any kind of data, enormous processing power and the ability to handle virtually limitless concurrent tasks or jobs.

Install OpenJDK on Ubuntu

The Hadoop framework is written in Java, and its services require a compatible Java Runtime Environment (JRE) and Java Development Kit (JDK). Use the following command to update your system before initiating a new installation:

a) Sudo apt update

Type the following command in your terminal to install OpenJDK 8:

b) sudo apt install openjdk-8-jdk –y

Once the installation process is complete, verify the current Java version:

c) Java -version

Set Up a Non-Root User for Hadoop Environment

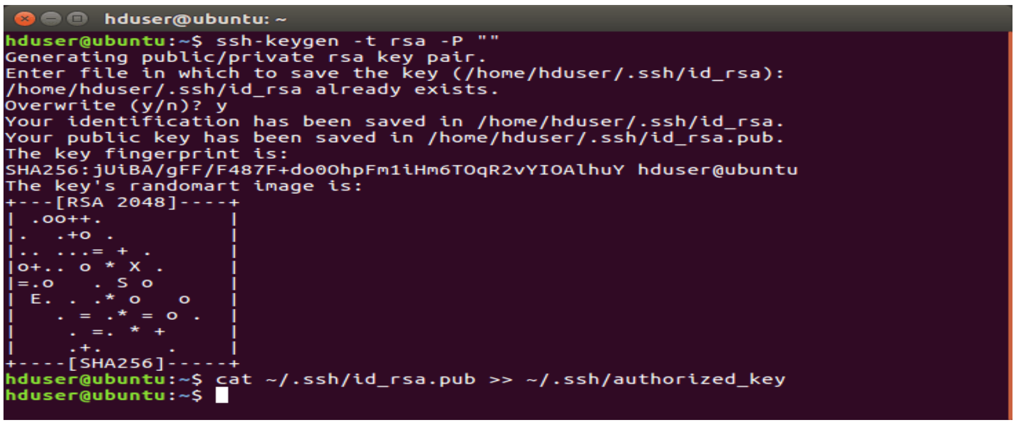
a) Install Open SSH Server and Open SSH Client

sudo apt-get install openssh-server openssh-client

1. Setup passwordless ssh

a) Install Open SSH Server and Open SSH Client

We will now setup the passwordless ssh client with the following command.
sudo apt-get install openssh-server openssh-client



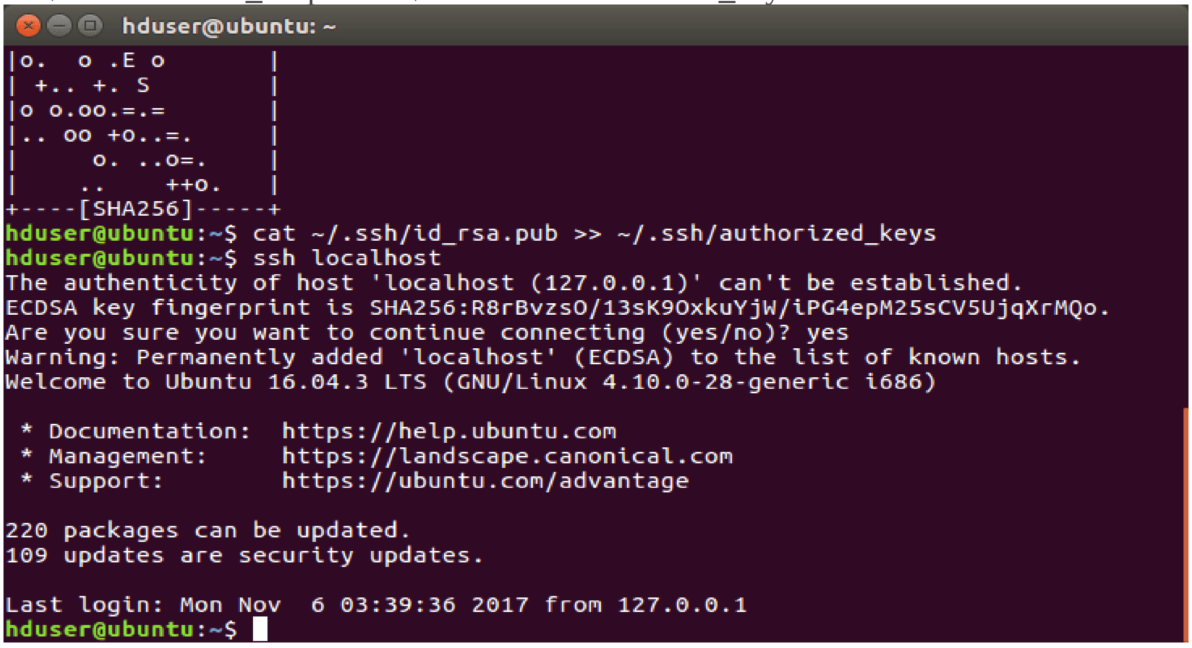
```
hduser@ubuntu: ~
hduser@ubuntu:~$ ssh-keygen -t rsa -P ""
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hduser/.ssh/id_rsa):
/home/hduser/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Your identification has been saved in /home/hduser/.ssh/id_rsa.
Your public key has been saved in /home/hduser/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:jUiBA/gFF/F487F+do0OhpFm1iHm6TOqR2vYIOAlhuY hduser@ubuntu
The key's randomart image is:
+---[RSA 2048]----+
|  .oo++.         |
|.   .+o .        |
|..  ...= + .     |
|o+.. o * X .     |
|=.o    . S o     |
| E. .  .* o    o |
|    . = .* = o . |
|     . =. * +    |
|     .+.       . |
+----[SHA256]-----+
hduser@ubuntu:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_key
hduser@ubuntu:~$ 
```

b) Generate Public & Private Key Pairs

ssh-keygen -t rsa -P ""
The terminal will prompt the user for entering the file name. Press enter and proceed. The location of a file will be in the home directory. Moreover, the extension will be the .ssh file.

c) Configure password-less SSH

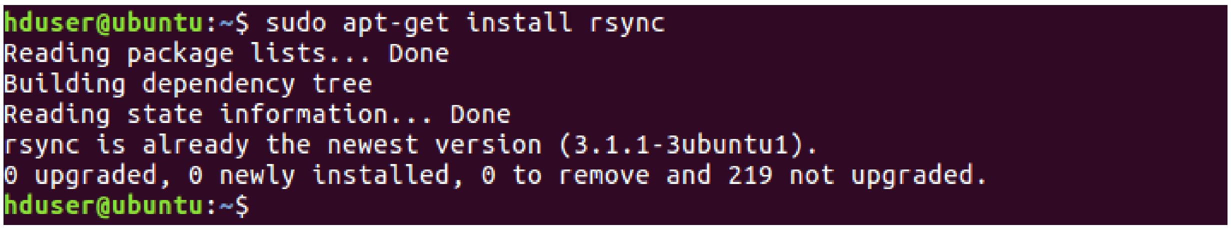
The below command will add the public ssh-key to authorized_keys. Moreover, it will configure the passwordless ssh.
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys


hduser@ubuntu: ~
```
|o.   o .E o        |
| +.. +. S          |
|o o.oo.=.=         |
|.. oo +o..=.       |
|     o. ..o=.      |
|     ..    ++o.    |
+----[SHA256]-----+
hduser@ubuntu:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
hduser@ubuntu:~$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:R8rBvzsO/13sK9OxkuYjW/iPG4epM25sCV5UjqXrMQo.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.10.0-28-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

220 packages can be updated.
109 updates are security updates.

Last login: Mon Nov  6 03:39:36 2017 from 127.0.0.1
hduser@ubuntu:~$ 
```

d) Now verify the working of password-less ssh

As we type the “ssh localhost” it will prompt us to connect with it. Type ‘yes’ and press enter to proceed.
e) Now install rsync with command

$ sudo apt-get install rsync


```
hduser@ubuntu:~$ sudo apt-get install rsync
Reading package lists... Done
Building dependency tree
Reading state information... Done
rsync is already the newest version (3.1.1-3ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 219 not upgraded.
hduser@ubuntu:~$
```

2. Configure and Setup Hadoop

a) Download the Hadoop package 2.8.x

Use the link given below to download the Hadoop 2.8.x from Apache mirrors.
http://www-eu.apache.org/dist/hadoop/common/hadoop-2.8.2/

b) Untar the Tarball

Then we will extract the Hadoop into the home directory
$tar xzf hadoop-2.8.2.tar.gz

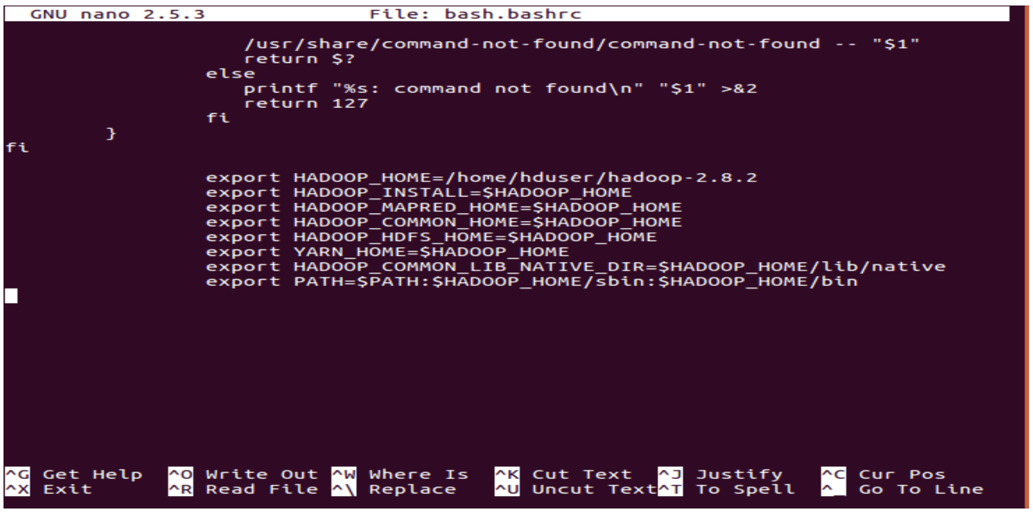
3. Setup Configuration

We can add only the minimum property in the Hadoop configuration. The user can add more properties to it.
a) Setting Up the environment variables

- **Edit .bashrc-** Edit the bashrc and therefore add hadoop in a path:

nano bash.bashrc
And add the following path variables in it

```
export HADOOP_HOME=/home/hduser/hadoop-2.8.2
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
```



```
  GNU nano 2.5.3                File: bash.bashrc

                   /usr/share/command-not-found/command-not-found -- "$1"
                   return $?
                else
                   printf "%s: command not found\n" "$1" >&2
                   return 127
                fi
        }
fi


                export HADOOP_HOME=/home/hduser/hadoop-2.8.2
                export HADOOP_INSTALL=$HADOOP_HOME
                export HADOOP_MAPRED_HOME=$HADOOP_HOME
                export HADOOP_COMMON_HOME=$HADOOP_HOME
                export HADOOP_HDFS_HOME=$HADOOP_HOME
                export YARN_HOME=$HADOOP_HOME
                export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
                export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
```

Hadoop Installation – Edit .bashrc

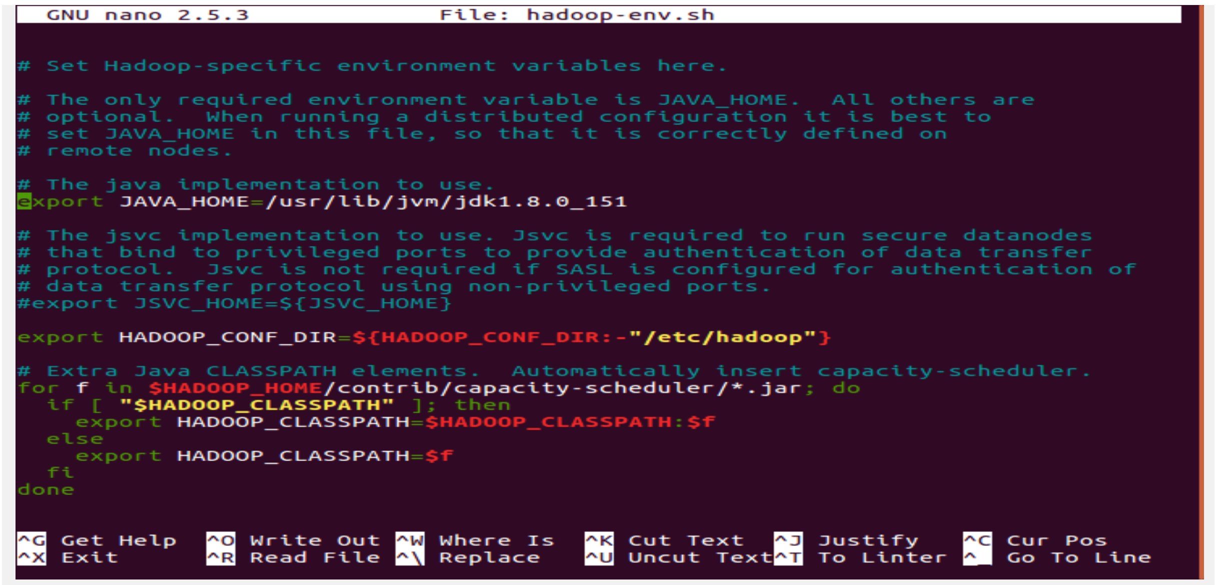
- **Source .bashrc in current login session in terminal**

source ~/.bashrc

b) Hadoop configuration file changes

- **Edit hadoop-env.sh**

Edit hadoop-env.sh file which is in etc/hadoop inside the Hadoop installation directory. The user can set JAVA_HOME:
export JAVA_HOME=<root directory of Java-installation> (eg: /usr/lib/jvm/jdk1.8.0_151/)



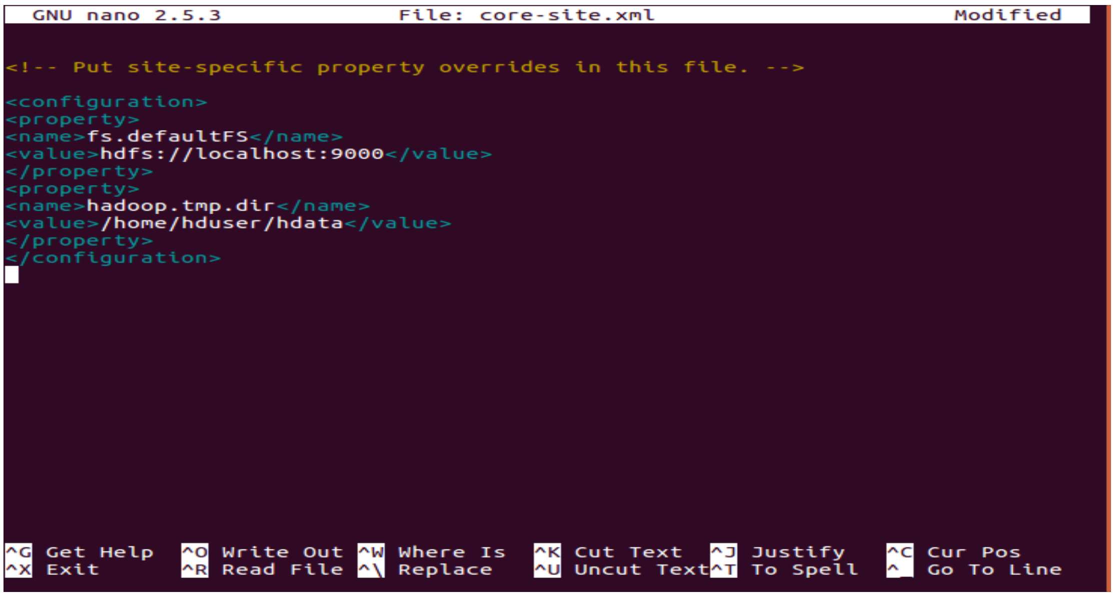
```
  GNU nano 2.5.3                 File: hadoop-env.sh

# Set Hadoop-specific environment variables here.

# The only required environment variable is JAVA_HOME.  All others are
# optional.  When running a distributed configuration it is best to
# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The java implementation to use.
export JAVA_HOME=/usr/lib/jvm/jdk1.8.0_151

# The jsvc implementation to use. Jsvc is required to run secure datanodes
# that bind to privileged ports to provide authentication of data transfer
# protocol.  Jsvc is not required if SASL is configured for authentication of
# data transfer protocol using non-privileged ports.
#export JSVC_HOME=${JSVC_HOME}

export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:-"/etc/hadoop"}

# Extra Java CLASSPATH elements.  Automatically insert capacity-scheduler.
for f in $HADOOP_HOME/contrib/capacity-scheduler/*.jar; do
  if [ "$HADOOP_CLASSPATH" ]; then
    export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$f
  else
    export HADOOP_CLASSPATH=$f
  fi
done

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Linter  ^_ Go To Line
```

Edit hadoop-env.sh

- **Edit core-site.xml**

Edit the core-site.xml with “nano core-site.xml”. The file is in the etc/hadoop inside Hadoop directory. Then we will add following entries.

```
<configuration>
<property>
<name>fs.defaultFS</name>
<value>hdfs://localhost:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/home/hdadmin/hdata</value>
</property>
</configuration>
```



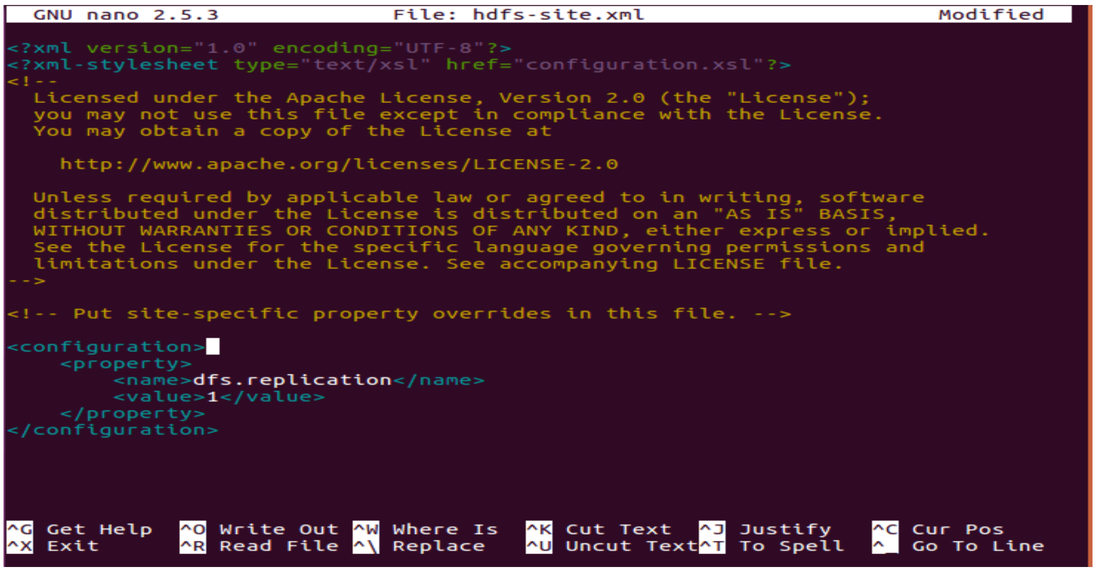
```
  GNU nano 2.5.3               File: core-site.xml                       Modified

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

<configuration>
<property>
<name>fs.defaultFS</name>
<value>hdfs://localhost:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/home/hduser/hdata</value>
</property>
</configuration>

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```

Edit core-site.xml

Then we will edit the hdfs-site.xml with “nano hdfs-site.xml”. This file is actually located in etc/hadoop inside Hadoop installation directory. We will add the following entries:

- **Edit hdfs-site.xml**

1. <configuration>
2. <property>
3. <name>dfs.replication</name>
4. <value>1</value>
5. </property>
6. </configuration>



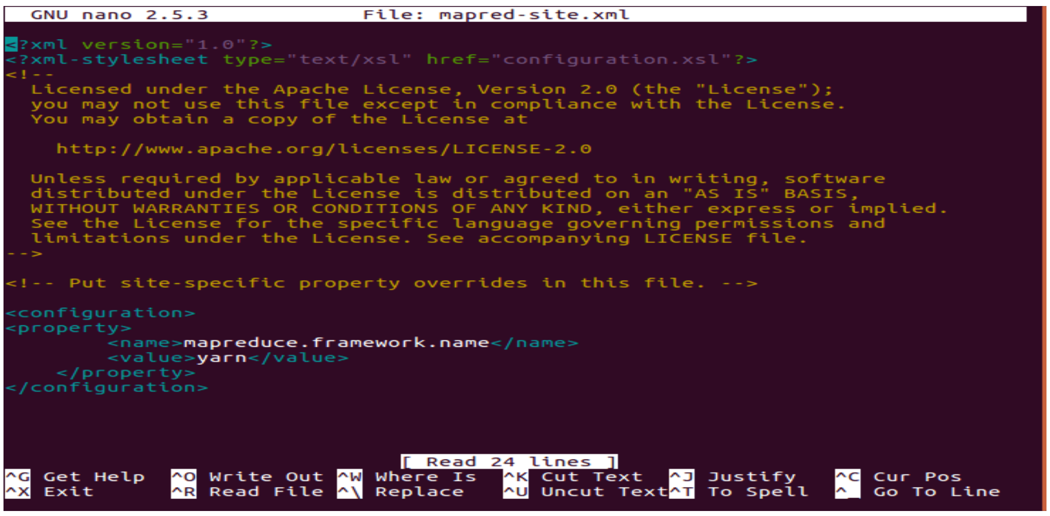
```
  GNU nano 2.5.3              File: hdfs-site.xml                      Modified

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

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

<configuration>
    <property>
        <name>dfs.replication</name>
        <value>1</value>
    </property>
</configuration>

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell  ^_ Go To Line
```

Edit hdfs-site.xml

- **Edit mapred-site.xml**

We will create a copy of mapred-site.xml from mapred-site.xml.template using cp command (cp mapred-site.xml.template mapred-site.xml). Now edit the mapred-site.xml with “nano command”. This file is also located in etc/hadoop inside Hadoop directory. We will copy the file with same name mapred-site.xml. This will add following entries:

1. <configuration>
2. <property>
3. <name>mapreduce.framework.name</name>
4. <value>yarn</value>
5. </property>
6. </configuration>

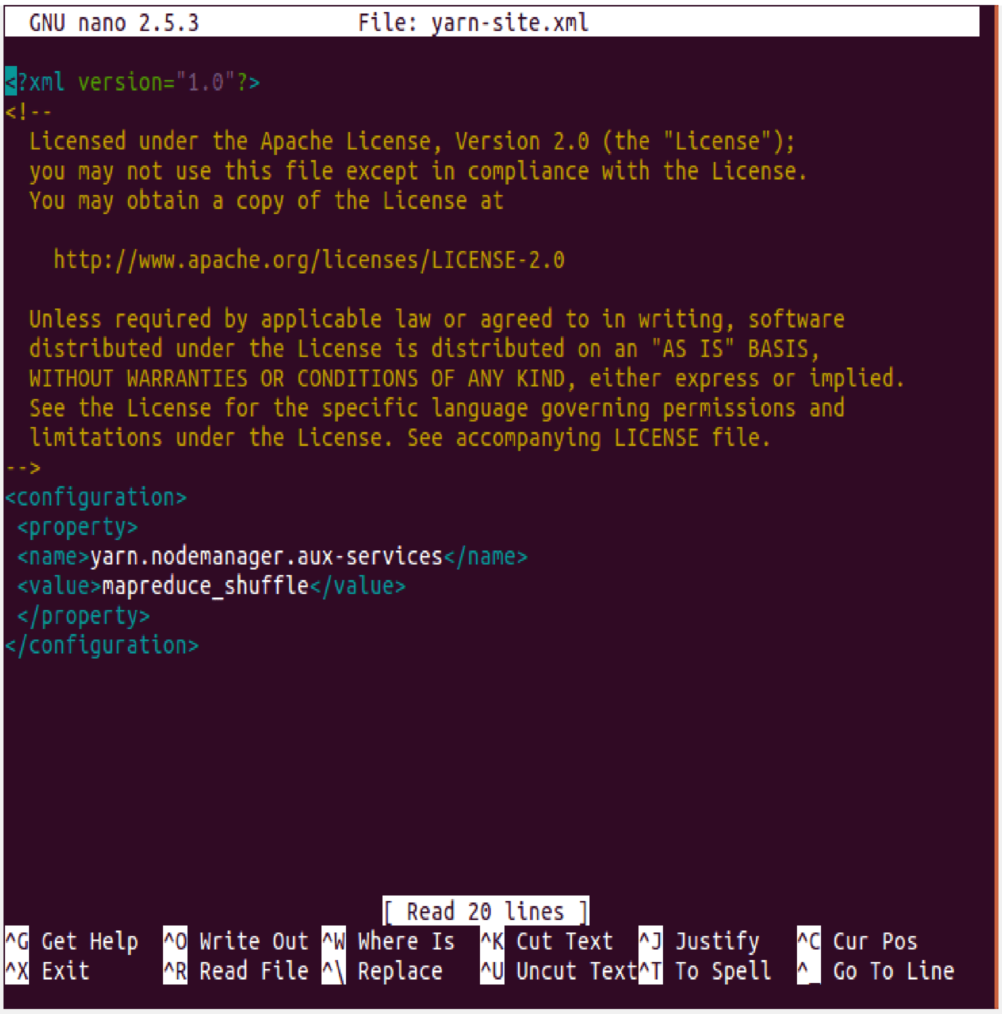


```
  GNU nano 2.5.3              File: mapred-site.xml

<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

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

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

                              [ Read 24 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell  ^_ Go To Line
```

- **Edit yarn-site.xml**

We will now edit yarn-site.xml with “nano yarn-site.xml”. It is in etc/hadoop inside Hadoop installation directory. Finally we add following entries:

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



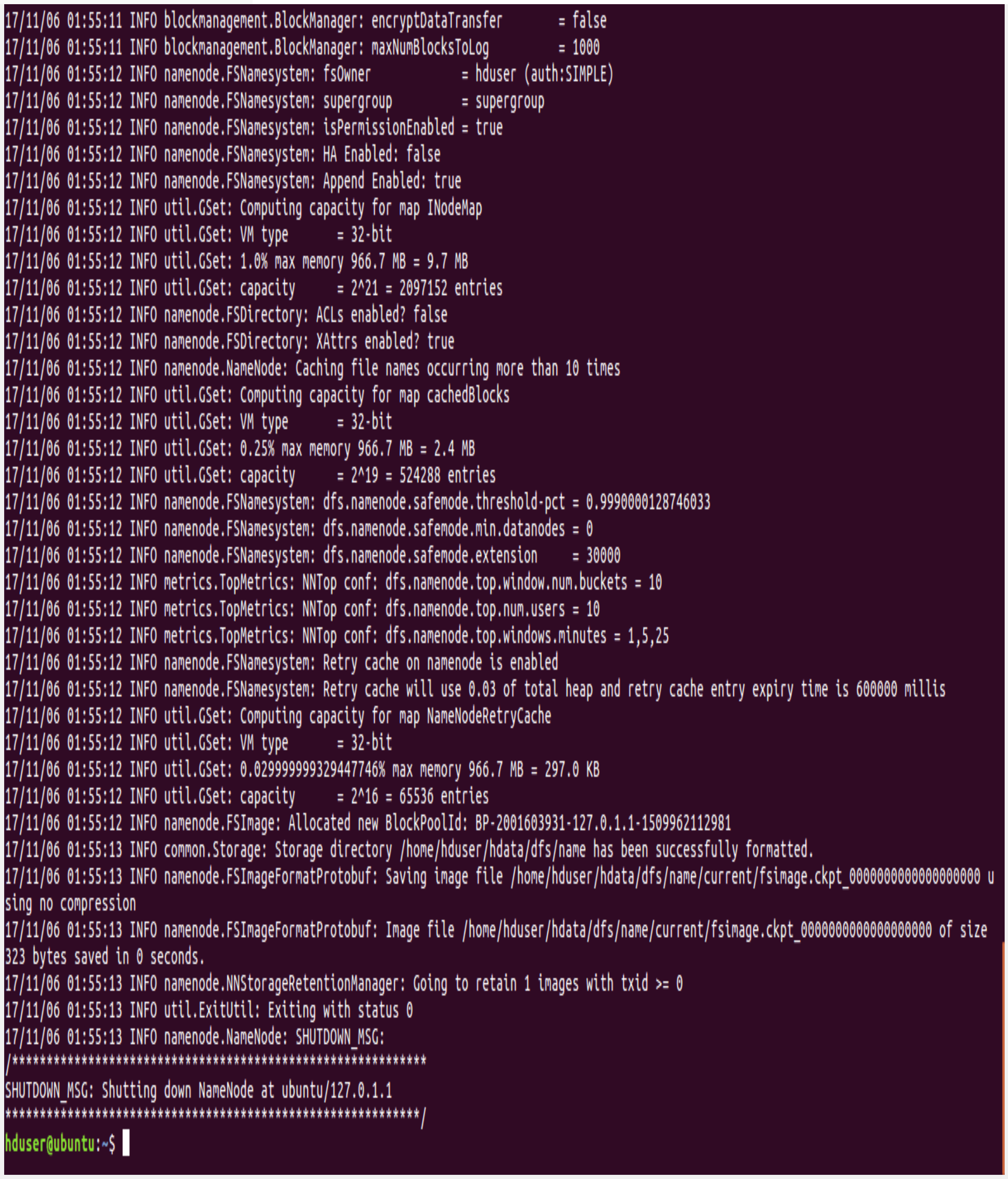
```
  GNU nano 2.5.3              File: yarn-site.xml

<?xml version="1.0"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->
<configuration>
 <property>
 <name>yarn.nodemanager.aux-services</name>
 <value>mapreduce_shuffle</value>
 </property>
</configuration>

                              [ Read 20 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text^T To Spell    ^_ Go To Line
```

Hadoop Installation – Edit yarn-site.xml file.

4. Start the cluster

We will now start the single node cluster with the following commands.
a) Format the namenode

Moreover, we will format the namenode before using it the first time.
hdfs namenode -format

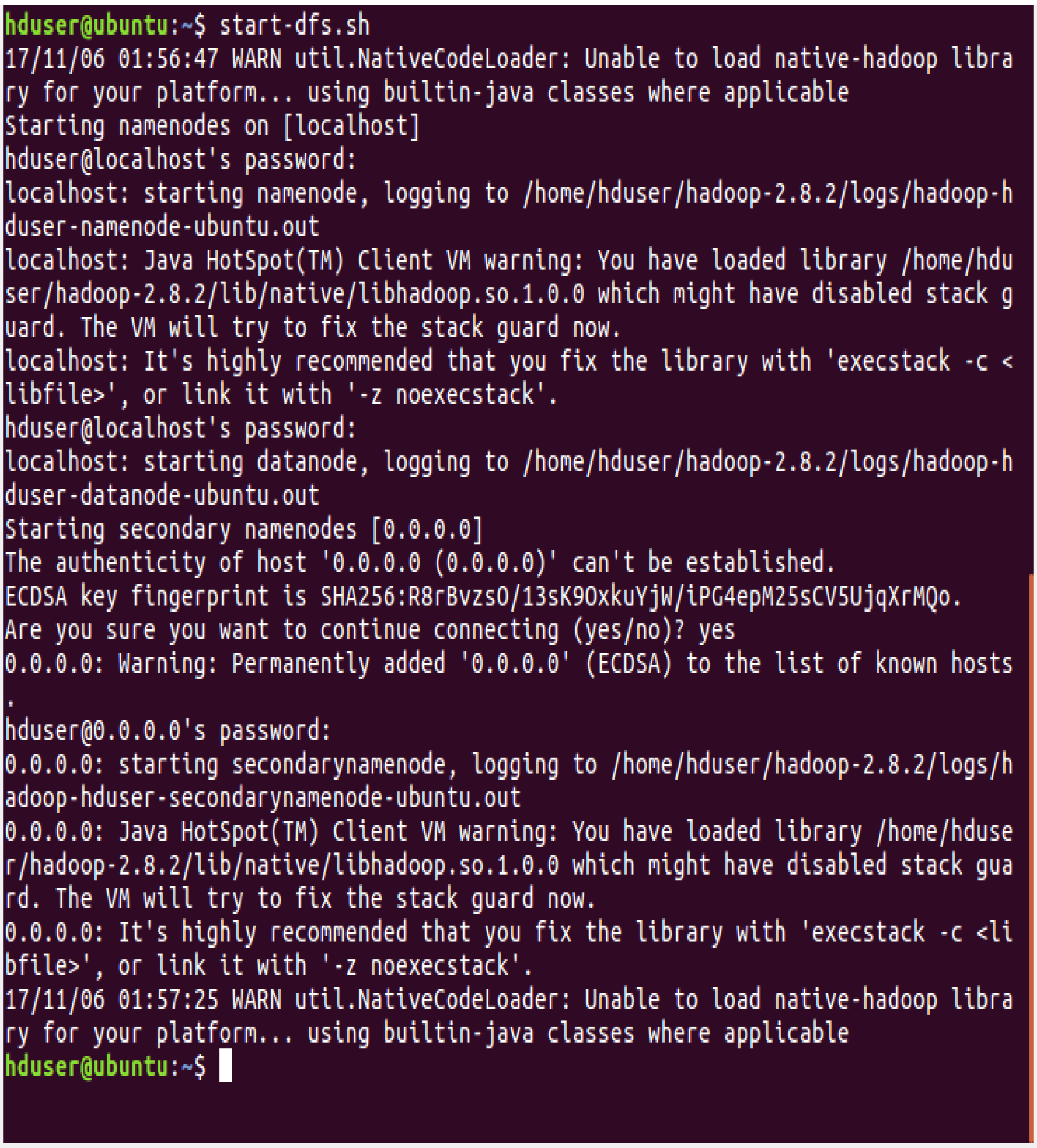


```
17/11/06 01:55:11 INFO blockmanagement.BlockManager: encryptDataTransfer        = false
17/11/06 01:55:11 INFO blockmanagement.BlockManager: maxNumBlocksToLog          = 1000
17/11/06 01:55:12 INFO namenode.FSNamesystem: fsOwner             = hduser (auth:SIMPLE)
17/11/06 01:55:12 INFO namenode.FSNamesystem: supergroup          = supergroup
17/11/06 01:55:12 INFO namenode.FSNamesystem: isPermissionEnabled = true
17/11/06 01:55:12 INFO namenode.FSNamesystem: HA Enabled: false
17/11/06 01:55:12 INFO namenode.FSNamesystem: Append Enabled: true
17/11/06 01:55:12 INFO util.GSet: Computing capacity for map INodeMap
17/11/06 01:55:12 INFO util.GSet: VM type       = 32-bit
17/11/06 01:55:12 INFO util.GSet: 1.0% max memory 966.7 MB = 9.7 MB
17/11/06 01:55:12 INFO util.GSet: capacity      = 2^21 = 2097152 entries
17/11/06 01:55:12 INFO namenode.FSDirectory: ACLs enabled? false
17/11/06 01:55:12 INFO namenode.FSDirectory: XAttrs enabled? true
17/11/06 01:55:12 INFO namenode.NameNode: Caching file names occurring more than 10 times
17/11/06 01:55:12 INFO util.GSet: Computing capacity for map cachedBlocks
17/11/06 01:55:12 INFO util.GSet: VM type       = 32-bit
17/11/06 01:55:12 INFO util.GSet: 0.25% max memory 966.7 MB = 2.4 MB
17/11/06 01:55:12 INFO util.GSet: capacity      = 2^19 = 524288 entries
17/11/06 01:55:12 INFO namenode.FSNamesystem: dfs.namenode.safemode.threshold-pct = 0.9990000128746033
17/11/06 01:55:12 INFO namenode.FSNamesystem: dfs.namenode.safemode.min.datanodes = 0
17/11/06 01:55:12 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension     = 30000
17/11/06 01:55:12 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
17/11/06 01:55:12 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
17/11/06 01:55:12 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
17/11/06 01:55:12 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
17/11/06 01:55:12 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
17/11/06 01:55:12 INFO util.GSet: Computing capacity for map NameNodeRetryCache
17/11/06 01:55:12 INFO util.GSet: VM type       = 32-bit
17/11/06 01:55:12 INFO util.GSet: 0.029999999329447746% max memory 966.7 MB = 297.0 KB
17/11/06 01:55:12 INFO util.GSet: capacity      = 2^16 = 65536 entries
17/11/06 01:55:12 INFO namenode.FSImage: Allocated new BlockPoolId: BP-2001603931-127.0.1.1-1509962112981
17/11/06 01:55:13 INFO common.Storage: Storage directory /home/hduser/hdata/dfs/name has been successfully formatted.
17/11/06 01:55:13 INFO namenode.FSImageFormatProtobuf: Saving image file /home/hduser/hdata/dfs/name/current/fsimage.ckpt_0000000000000000000 u
sing no compression
17/11/06 01:55:13 INFO namenode.FSImageFormatProtobuf: Image file /home/hduser/hdata/dfs/name/current/fsimage.ckpt_0000000000000000000 of size
323 bytes saved in 0 seconds.
17/11/06 01:55:13 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
17/11/06 01:55:13 INFO util.ExitUtil: Exiting with status 0
17/11/06 01:55:13 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at ubuntu/127.0.1.1
************************************************************/
hduser@ubuntu:~$ 
```

Install Hadoop 2 – Format the Namenode

b) Start the HDFS

We will start the hadoop cluster using the hadoop start-up script.
start-dfs.sh

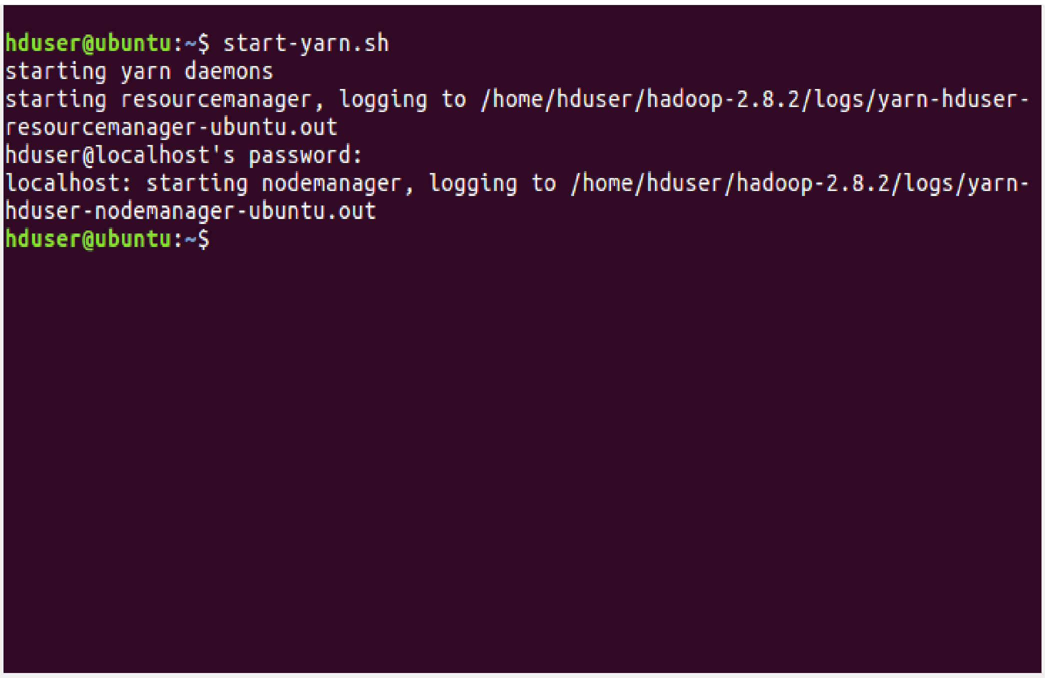


```
hduser@ubuntu:~$ start-dfs.sh
17/11/06 01:56:47 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
hduser@localhost's password:
localhost: starting namenode, logging to /home/hduser/hadoop-2.8.2/logs/hadoop-h
duser-namenode-ubuntu.out
localhost: Java HotSpot(TM) Client VM warning: You have loaded library /home/hdu
ser/hadoop-2.8.2/lib/native/libhadoop.so.1.0.0 which might have disabled stack g
uard. The VM will try to fix the stack guard now.
localhost: It's highly recommended that you fix the library with 'execstack -c <
libfile>', or link it with '-z noexecstack'.
hduser@localhost's password:
localhost: starting datanode, logging to /home/hduser/hadoop-2.8.2/logs/hadoop-h
duser-datanode-ubuntu.out
Starting secondary namenodes [0.0.0.0]
The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.
ECDSA key fingerprint is SHA256:R8rBvzsO/13sK9OxkuYjW/iPG4epM25sCV5UjqXrMQo.
Are you sure you want to continue connecting (yes/no)? yes
0.0.0.0: Warning: Permanently added '0.0.0.0' (ECDSA) to the list of known hosts
.
hduser@0.0.0.0's password:
0.0.0.0: starting secondarynamenode, logging to /home/hduser/hadoop-2.8.2/logs/h
adoop-hduser-secondarynamenode-ubuntu.out
0.0.0.0: Java HotSpot(TM) Client VM warning: You have loaded library /home/hduse
r/hadoop-2.8.2/lib/native/libhadoop.so.1.0.0 which might have disabled stack gua
rd. The VM will try to fix the stack guard now.
0.0.0.0: It's highly recommended that you fix the library with 'execstack -c <li
bfile>', or link it with '-z noexecstack'.
17/11/06 01:57:25 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
hduser@ubuntu:~$
```

Hadoop start-up script

c) Starting the YARN services

For starting the YARN we use
start-yarn.sh

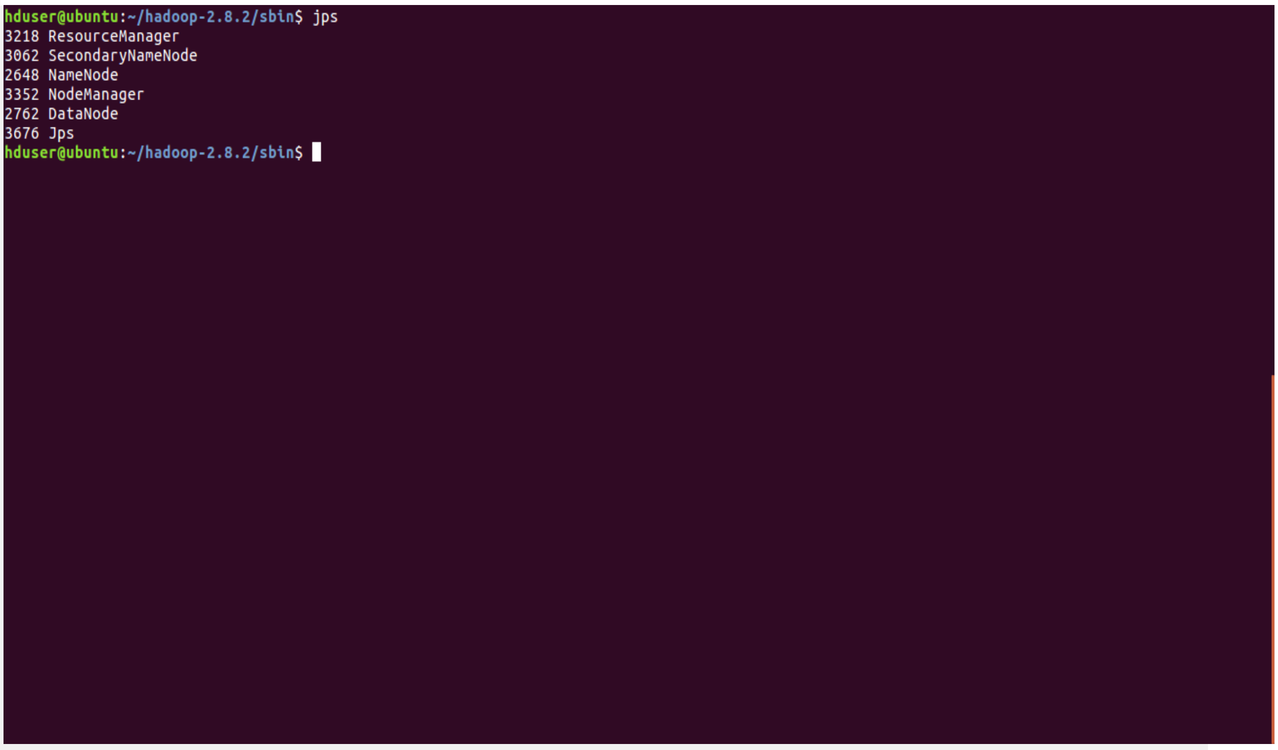


```
hduser@ubuntu:~$ start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/hduser/hadoop-2.8.2/logs/yarn-hduser-
resourcemanager-ubuntu.out
hduser@localhost's password:
localhost: starting nodemanager, logging to /home/hduser/hadoop-2.8.2/logs/yarn-
hduser-nodemanager-ubuntu.out
hduser@ubuntu:~$
```

Apache Hadoop Installation on Ubuntu – Starting the YARN services

d) Verify if all process started

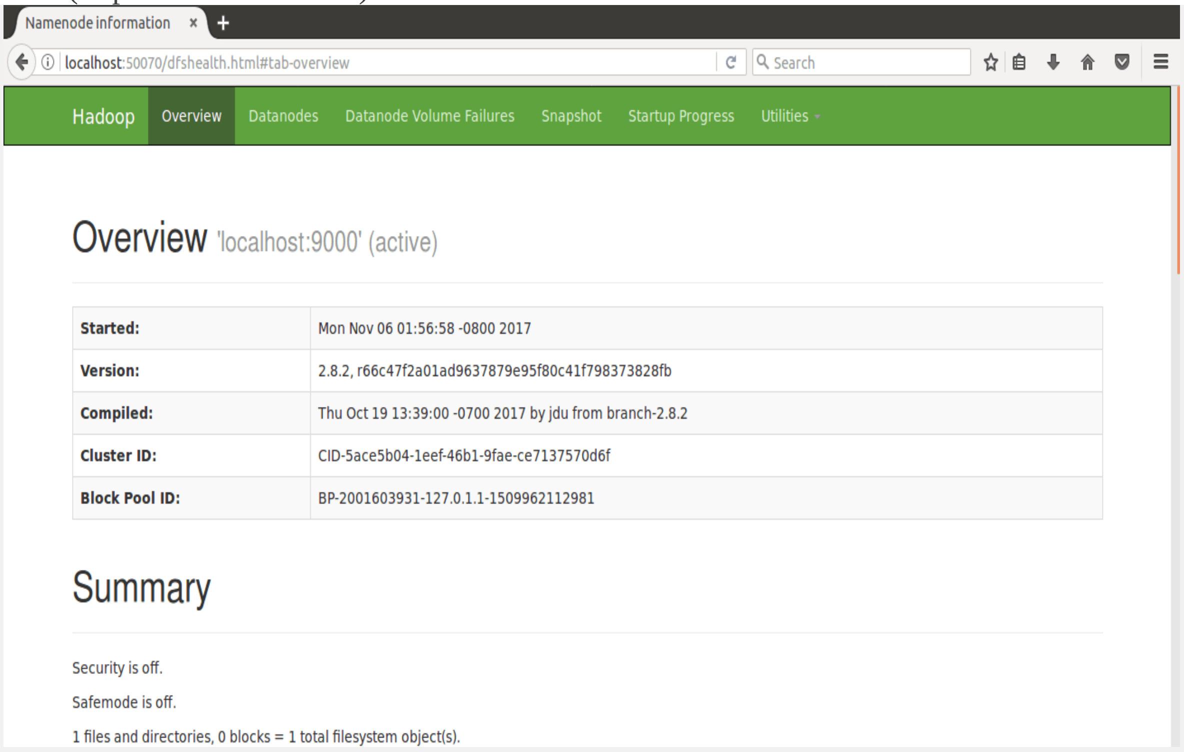
1. jps
2. 6775 DataNode
3. 7209 ResourceManager
4. 7017 SecondaryNameNode
5. 6651 NameNode
6. 7339 NodeManager
7. 7663 Jps



```
hduser@ubuntu:~/hadoop-2.8.2/sbin$ jps
3218 ResourceManager
3062 SecondaryNameNode
2648 NameNode
3352 NodeManager
2762 DataNode
3676 Jps
hduser@ubuntu:~/hadoop-2.8.2/sbin$
```

e) Web interface-For viewing Web UI of NameNode

visit : (http://localhost:50070)



Overview 'localhost:9000' (active)

Started:	Mon Nov 06 01:56:58 -0800 2017
Version:	2.8.2, r66c47f2a01ad9637879e95f80c41f798373828fb
Compiled:	Thu Oct 19 13:39:00 -0700 2017 by jdu from branch-2.8.2
Cluster ID:	CID-5ace5b04-1eef-46b1-9fae-ce7137570d6f
Block Pool ID:	BP-2001603931-127.0.1.1-1509962112981

Summary

Security is off.

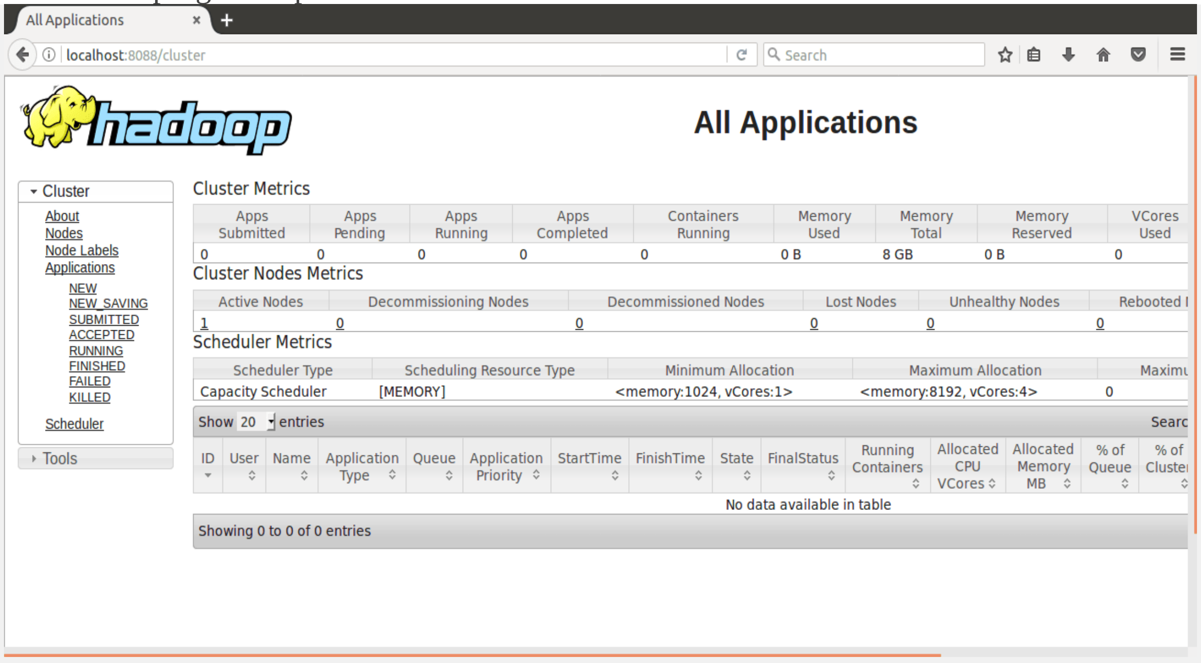
Safemode is off.

1 files and directories, 0 blocks = 1 total filesystem object(s).

Web interface-For viewing Web UI of NameNode

f) Resource Manager UI (http://localhost:8088)

The web interface will display all running jobs on cluster information. Hence, this will help monitor the progress report.



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used
0	0	0	0	0	0 B	8 GB	0 B	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted I
1	0	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximu
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Show 20 entries Searc

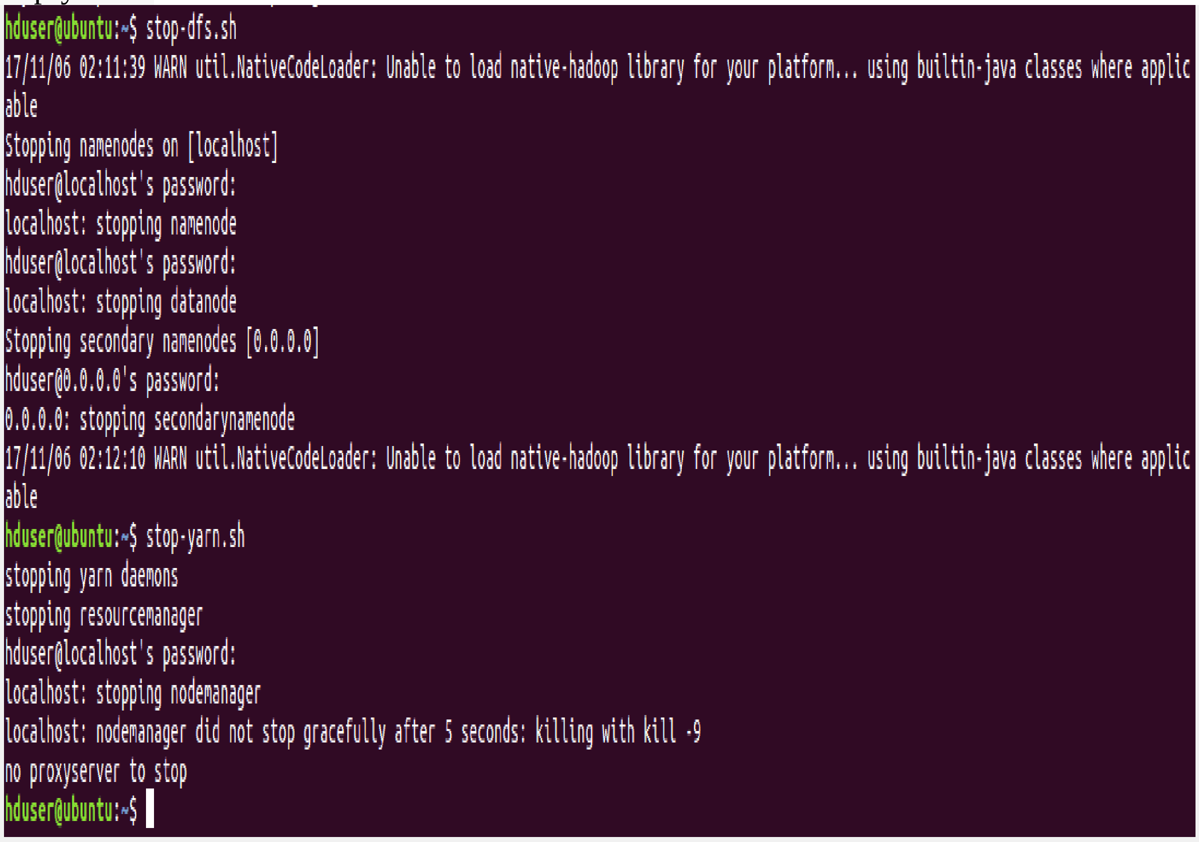
ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	% of Queue	% of Cluste
No data available in table														

Showing 0 to 0 of 0 entries

Hadoop Installation – Resource Manager UI

5. Stopping the clusters

To Stop the HDFS Services we use stop-dfs.sh. To Stop YARN Services we use stop-yarn.sh



```
hduser@ubuntu:~$ stop-dfs.sh
17/11/06 02:11:39 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
Stopping namenodes on [localhost]
hduser@localhost's password:
localhost: stopping namenode
hduser@localhost's password:
localhost: stopping datanode
Stopping secondary namenodes [0.0.0.0]
hduser@0.0.0.0's password:
0.0.0.0: stopping secondarynamenode
17/11/06 02:12:10 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
hduser@ubuntu:~$ stop-yarn.sh
stopping yarn daemons
stopping resourcemanager
hduser@localhost's password:
localhost: stopping nodemanager
localhost: nodemanager did not stop gracefully after 5 seconds: killing with kill -9
no proxyserver to stop
hduser@ubuntu:~$
```

Install Hadoop on Ubuntu – Stopping the clusters

You have successfully installed Hadoop 2.8.x on Ubuntu.

b) PIG INSTALLATION

b. PIG: A high-level platform for creating programs that run on Hadoop, Apache Pig can execute its Hadoop jobs in MapReduce, Apache Tez, or Apache Spark. Pig Latin is the language used for this platform, which can be extended using user-defined functions (UDFs) that the users can write in Java, Python, JavaScript, Ruby, or Groovy.

i. Pre-Requisite to Install Pig

You must have Hadoop and Java JDK installed on your system. Hence, before installing Pig you should install Hadoop and Java by following the steps given in this installation guide.
ii. Downloading Pig

You can download Pig file from the below link:
https://archive.cloudera.com/cdh5/cdh/5/
hadoop-2.5.0-cdh5.3.2 is already installed on the system hence the supported pig version will be downloaded from here which is pig-0.12.0-cdh5.3.2.

iii. Installing Pig

The steps for Apache Pig installation are given below:

Step 1:

Move the downloaded pig-0.12.0-cdh5.3.2.tar file from Downloads folder to the Directory where you had installed Hadoop.

Step 2:

Untar pig-0.12.0-cdh5.3.2.tar file by executing the below command on your terminal:

```
dataflair@ubuntu:~$ tar zxvf pig-0.12.0-cdh5.3.2.tar
```

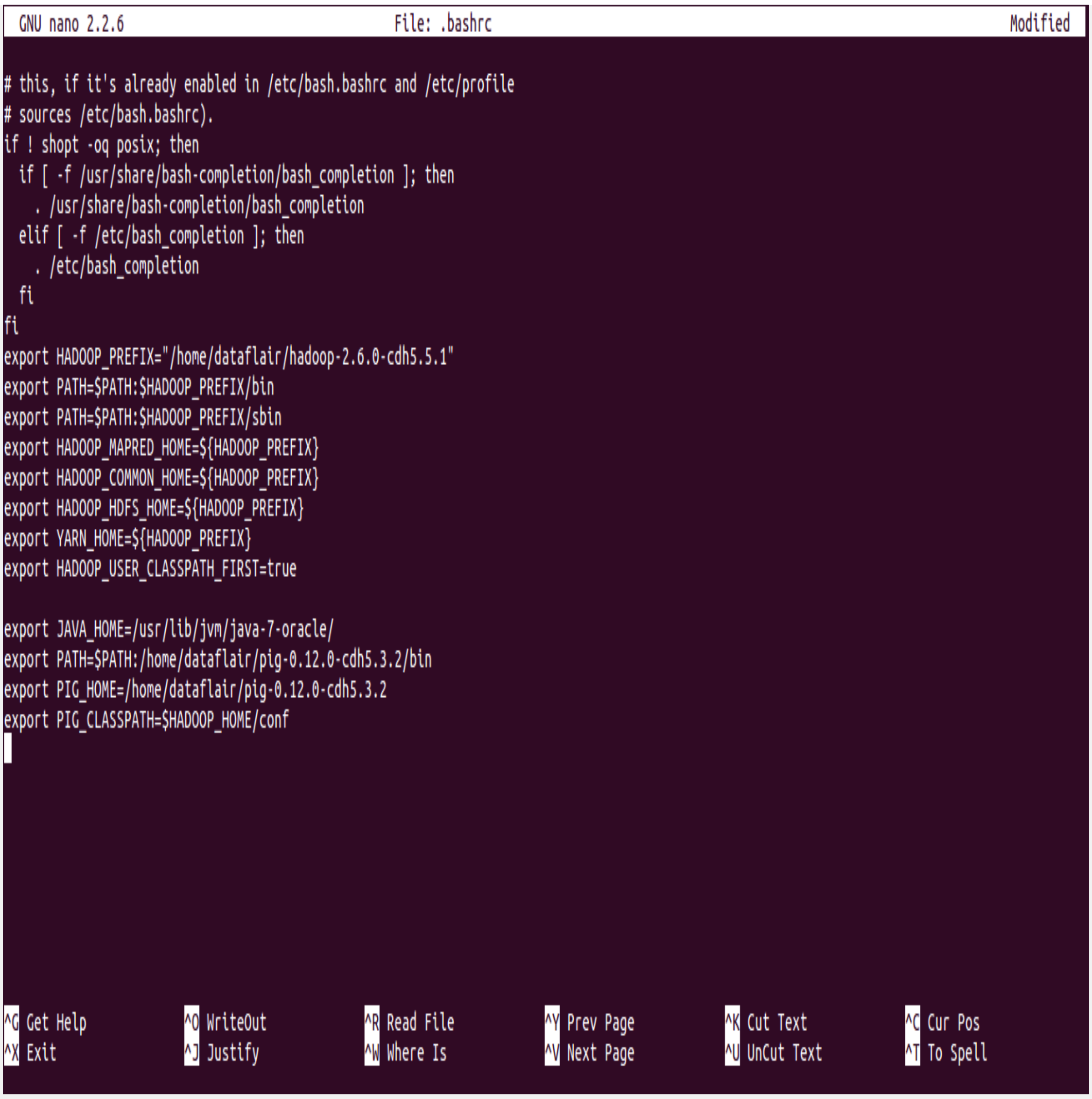
Step 3:

Now we need to configure pig. In order to configure pig, we need to edit ".bashrc" file. To edit this file execute below command:

```
dataflair@ubuntu:~$ nano .bashrc
```

And in this file we need to add the following:

```
export PATH=$PATH:/home/dataflair/pig-0.12.0-cdh5.3.2/bin
export PIG_HOME=/home/dataflair/pig-0.12.0-cdh5.3.2
export PIG_CLASSPATH=$HADOOP_HOME/conf
```



```
GNU nano 2.2.6                      File: .bashrc                                Modified

# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi
export HADOOP_PREFIX="/home/dataflair/hadoop-2.6.0-cdh5.5.1"
export PATH=$PATH:$HADOOP_PREFIX/bin
export PATH=$PATH:$HADOOP_PREFIX/sbin
export HADOOP_MAPRED_HOME=${HADOOP_PREFIX}
export HADOOP_COMMON_HOME=${HADOOP_PREFIX}
export HADOOP_HDFS_HOME=${HADOOP_PREFIX}
export YARN_HOME=${HADOOP_PREFIX}
export HADOOP_USER_CLASSPATH_FIRST=true

export JAVA_HOME=/usr/lib/jvm/java-7-oracle/
export PATH=$PATH:/home/dataflair/pig-0.12.0-cdh5.3.2/bin
export PIG_HOME=/home/dataflair/pig-0.12.0-cdh5.3.2
export PIG_CLASSPATH=$HADOOP_HOME/conf
```

Apache Pig Installation_Bashrc File

After adding the above parameters save this file by using "CTRL+X" and then "Y" on your keyboard.

Step 4:

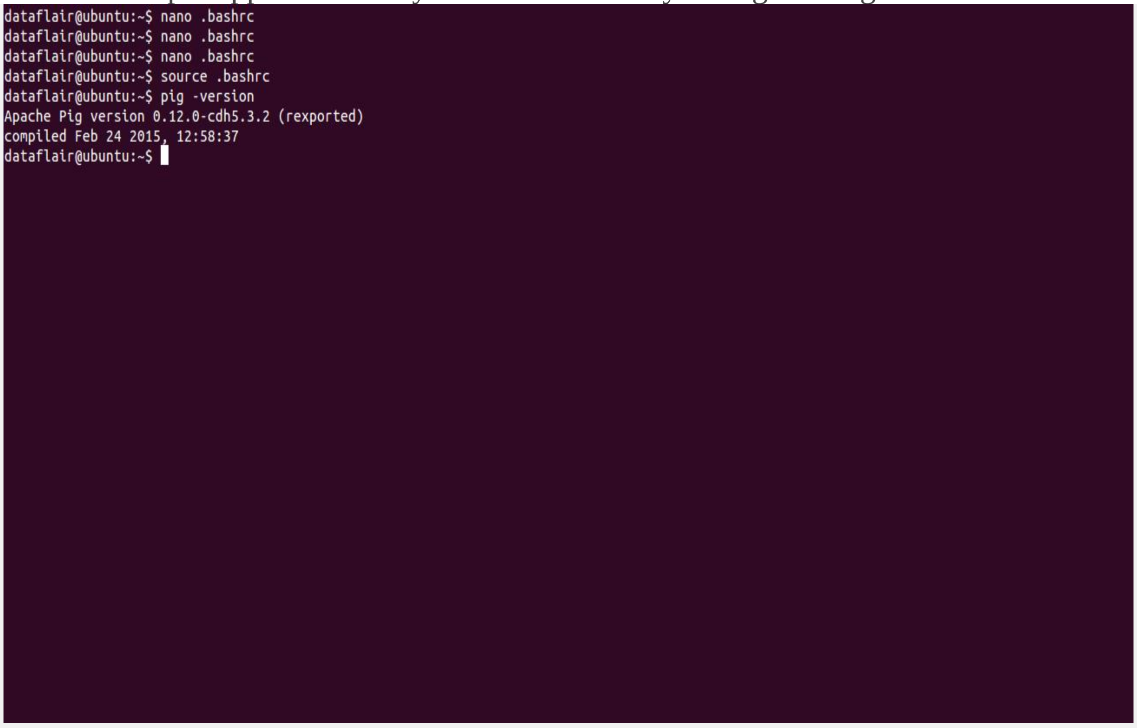
Update .bashrc file by executing below command:

```
dataflair@ubuntu:~$ source .bashrc
```

After refreshing the .bashrc file Pig gets successfully installed. In order to check the version of your Pig file execute the below command:

```
dataflair@ubuntu:~$ pig -version
```

If the below output appears means you had successfully configured Pig.



```
dataflair@ubuntu:~$ nano .bashrc
dataflair@ubuntu:~$ nano .bashrc
dataflair@ubuntu:~$ nano .bashrc
dataflair@ubuntu:~$ source .bashrc
dataflair@ubuntu:~$ pig -version
Apache Pig version 0.12.0-cdh5.3.2 (rexported)
compiled Feb 24 2015, 12:58:37
dataflair@ubuntu:~$ 
```

Apache Pig Version

iv. Starting Pig

We can start Pig in one of the following two modes mentioned below:

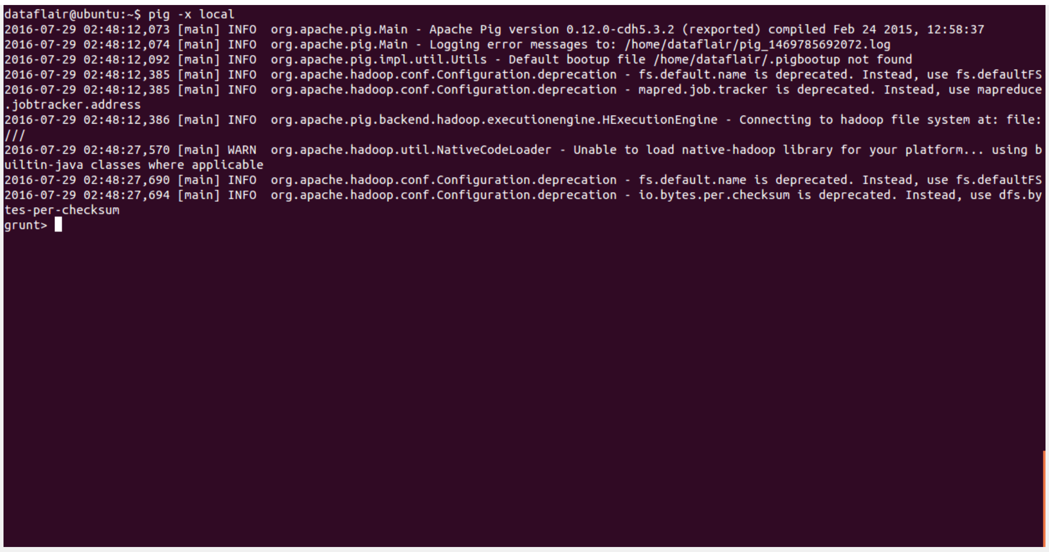
1. Local Mode
2. Cluster Mode

To start using pig in local mode '-x local' option is used whereas while executing only "pig" command without any options, Pig starts in the cluster mode. While running pig in local mode, it can only access files present on the local file system. Whereas, on starting pig in cluster mode pig can access files present in HDFS.

To start Pig in Local Mode execute the below command:

```
dataflair@ubuntu:~$ pig -x local
```

And if you get the below output that means Pig started successfully in Local mode.



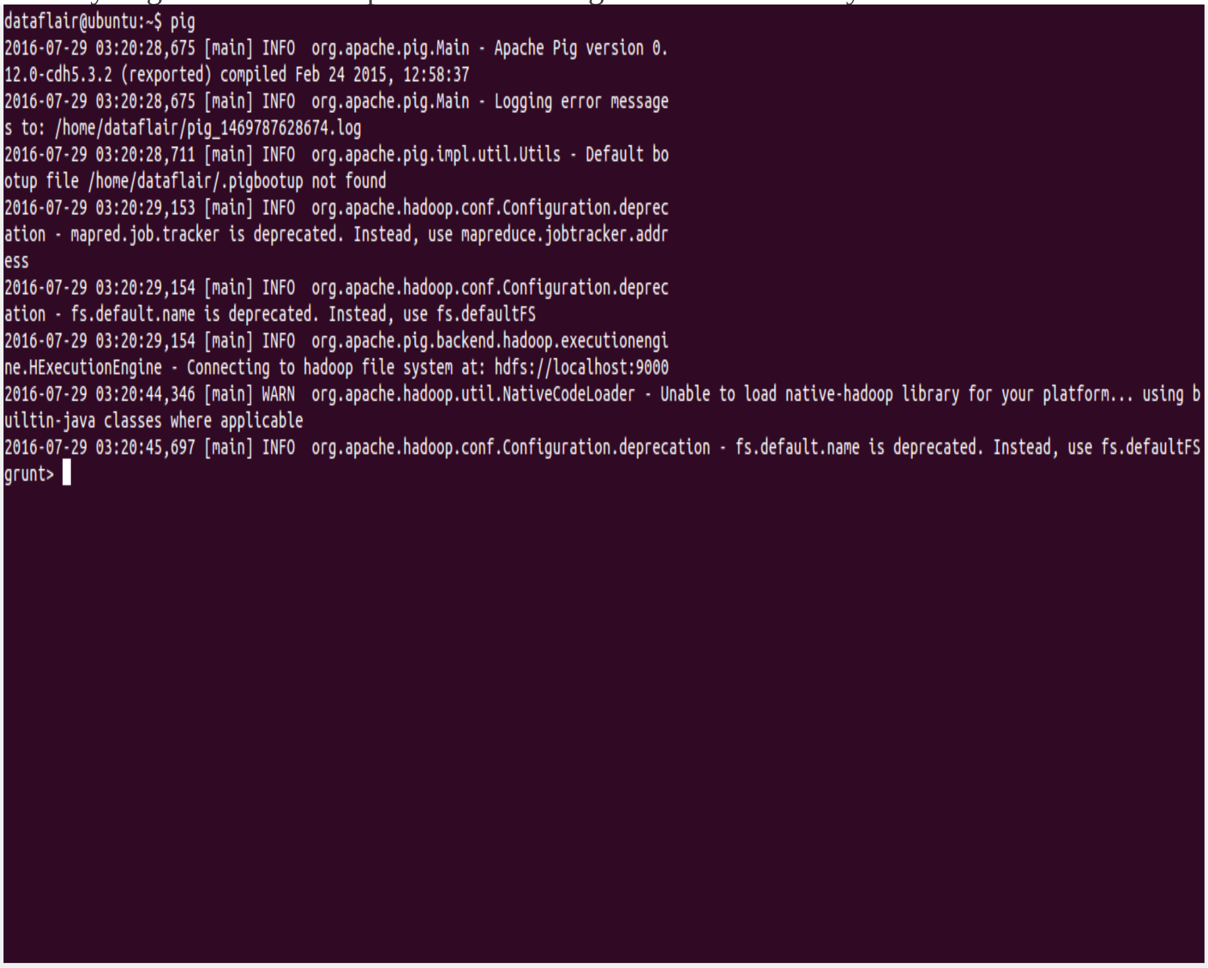
```
dataflair@ubuntu:~$ pig -x local
2016-07-29 02:48:12,073 [main] INFO  org.apache.pig.Main - Apache Pig version 0.12.0-cdh5.3.2 (rexported) compiled Feb 24 2015, 12:58:37
2016-07-29 02:48:12,074 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/dataflair/pig_1469785692072.log
2016-07-29 02:48:12,092 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home/dataflair/.pigbootup not found
2016-07-29 02:48:12,385 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2016-07-29 02:48:12,385 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce
.jobtracker.address
2016-07-29 02:48:12,386 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: file:
///
2016-07-29 02:48:27,570 [main] WARN  org.apache.hadoop.util.NativeCodeLoader - Unable to load native-hadoop library for your platform... using b
uiltin-java classes where applicable
2016-07-29 02:48:27,690 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2016-07-29 02:48:27,694 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.by
tes-per-checksum
grunt> 
```

Running_Pig_Local_mode

To start Pig in Cluster-Mode execute the below command:

dataflair@ubuntu:~$ pig

And if you get the below output that means Pig started successfully in Cluster mode.



```
dataflair@ubuntu:~$ pig
2016-07-29 03:20:28,675 [main] INFO  org.apache.pig.Main - Apache Pig version 0.
12.0-cdh5.3.2 (rexported) compiled Feb 24 2015, 12:58:37
2016-07-29 03:20:28,675 [main] INFO  org.apache.pig.Main - Logging error message
s to: /home/dataflair/pig_1469787628674.log
2016-07-29 03:20:28,711 [main] INFO  org.apache.pig.impl.util.Utils - Default bo
otup file /home/dataflair/.pigbootup not found
2016-07-29 03:20:29,153 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
2016-07-29 03:20:29,154 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2016-07-29 03:20:29,154 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to hadoop file system at: hdfs://localhost:9000
2016-07-29 03:20:44,346 [main] WARN  org.apache.hadoop.util.NativeCodeLoader - Unable to load native-hadoop library for your platform... using b
uiltin-java classes where applicable
2016-07-29 03:20:45,697 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt> 
```

Running_Pig_Cluster_mode

C. HBASE : HBase is a column-oriented non-relational database management system that runs on top of Hadoop Distributed File System (HDFS). HBase provides a fault-tolerant way of storing sparse data sets, which are common in many big data use cases. It is well suited for real- time data processing or random read/write access to large volumes of data.

HBASE INSTALLATION

Step 1: Before installing Hbase, you need to First ensure that java8 is installed:

```
sudo add-apt-repository ppa:webupd8team/java

sudo apt-get update

sudo apt-get install oracle-java8-installer
```

Verify that java is correctly installed:

```
java -version
```

Configuring Java Environment

```
sudo apt-get install oracle-java8-set-default
```

Step 2: Ensure that hadoop installation was successful on your machine

Step 3: Download Apache Hbase

Go to downloads page

Choose hbase file: hbase-1.2.5-bin.tar.gz

Step 4: Complete the installation process

- Move the downloaded file "hbase-1.2.5-bin.tar.gz" to your home (~)
- Compress it :

```
tar -zxvf hbase-1.2.5-bin.tar.gz
```

Edit hbase-env.sh using this command lines:

- cd /usr/local/hbase/conf
- gedit hbase-env.sh

Add this line:

- export JAVA_HOME=/usr/lib/jvm/java-8-oracle

Edit hbase-site.xml using this command lines:

- cd /usr/local/hbase/conf
- gedit hbase-site.xml

Add this lines:

```
<configuration>

   //Here you have to set the path where you want HBase to store its files.

   <property>

      <name>hbase.rootdir</name>

      <value>hdfs://localhost:9000/hbase</value>

   </property>

  //Here you have to set the path where you want HBase to store its built in zookeeper
files.

   <property>

      <name>hbase.zookeeper.property.dataDir</name>

      <value>/home/hadoopworkshop/zookeeper</value>

   </property>

   <property>

       <name>hbase.cluster.distributed</name>

       <value>true</value>

   </property>

 </configuration>
```

Edit bashrc using this command line:

- cd
- sudo gedit .bashrc

Add this lines:

- # - HBASE ENVIRONMENT VARIABLES START -#
- export HBASE_HOME=/usr/local/hbase
- export PATH=$PATH:$HBASE_HOME/bin
- # - HBASE ENVIRONMENT VARIABLES END -#

Execute bashrc:

- . ~/.bashrc

Step 5: Start hbase processes

Start hadoop process first:

- start-hdfs.sh
- start-yarn.sh

Step 6: Stop hbase processes

stop-hbase.sh

Stop hadoop process:

stop-dfs.sh

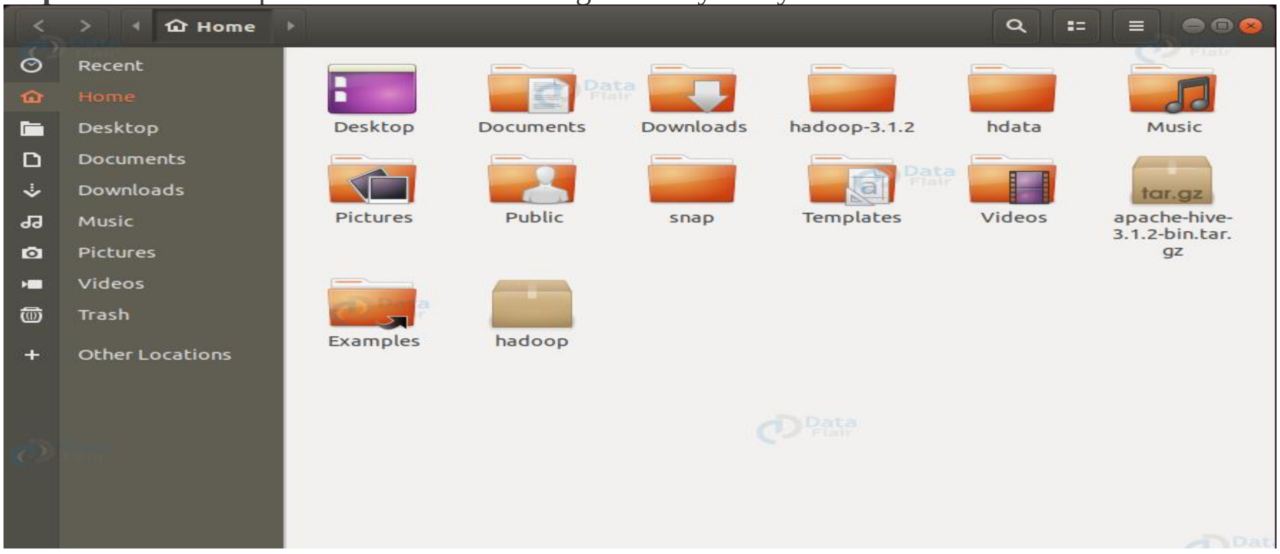
stop-yarn.sh

D) Apache Hive is a data warehouse system for data summarization and analysis and for querying of large data systems in the open-source Hadoop platform. It converts SQL-like queries into MapReduce jobs for easy execution and processing of extremely large volumes of data.

HIVE INSTALLATION

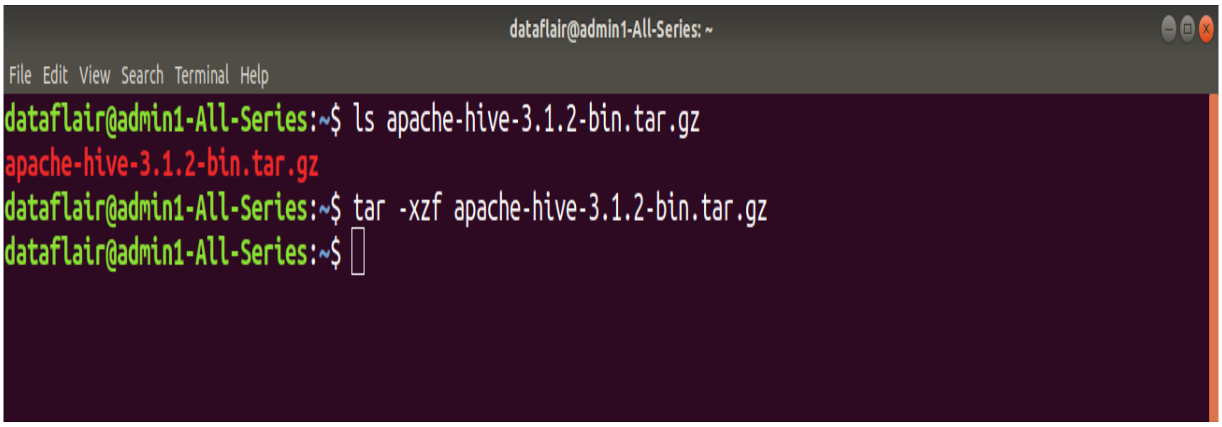
1. Download Hive

Step 1: First, download the Hive 3.1.2 from this **link.**
Step 2: Locate the apache-hive-3.1.2-bin.tar.gz file in your system.

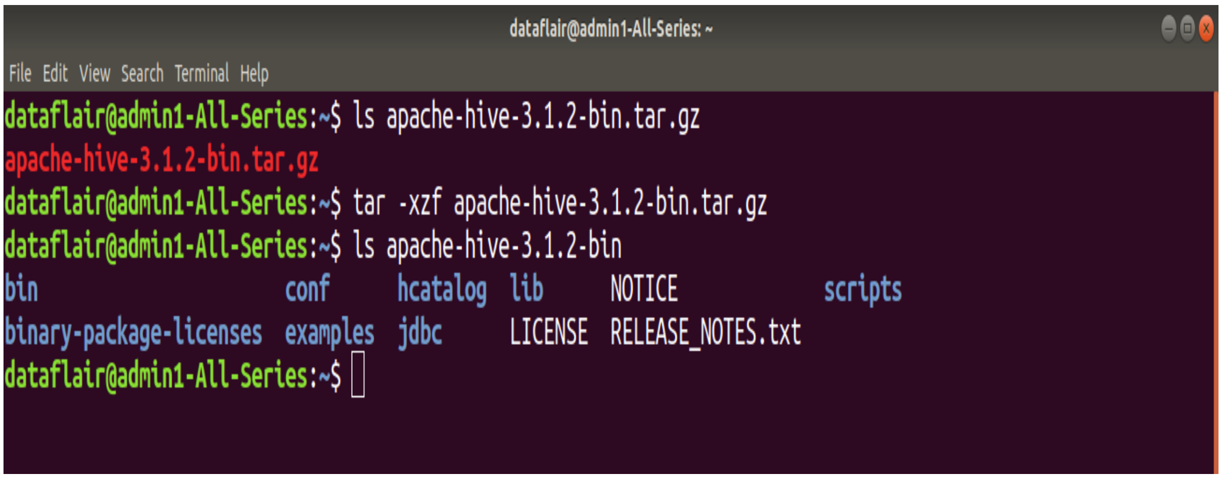


Step 3: Extract this tar file using the below command:

```
1. tar -xzf apache-hive-3.1.2-bin.tar.gz
```



```
dataflair@admin1-All-Series:~$ ls apache-hive-3.1.2-bin.tar.gz
apache-hive-3.1.2-bin.tar.gz
dataflair@admin1-All-Series:~$ tar -xzf apache-hive-3.1.2-bin.tar.gz
dataflair@admin1-All-Series:~$
```

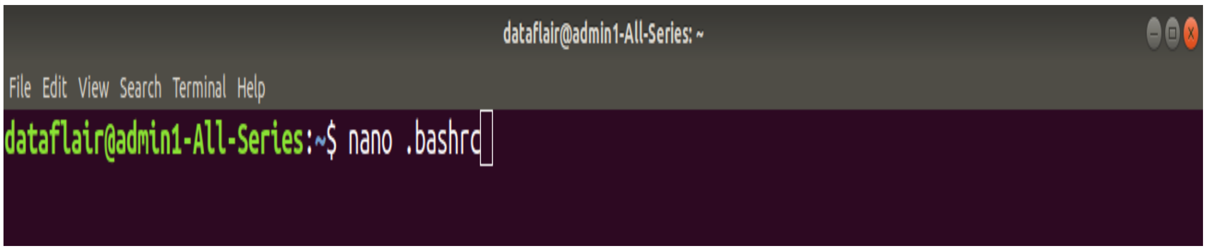


```
dataflair@admin1-All-Series:~$ ls apache-hive-3.1.2-bin.tar.gz
apache-hive-3.1.2-bin.tar.gz
dataflair@admin1-All-Series:~$ tar -xzf apache-hive-3.1.2-bin.tar.gz
dataflair@admin1-All-Series:~$ ls apache-hive-3.1.2-bin
bin                      conf      hcatalog  lib      NOTICE             scripts
binary-package-licenses  examples  jdbc      LICENSE  RELEASE_NOTES.txt
dataflair@admin1-All-Series:~$
```

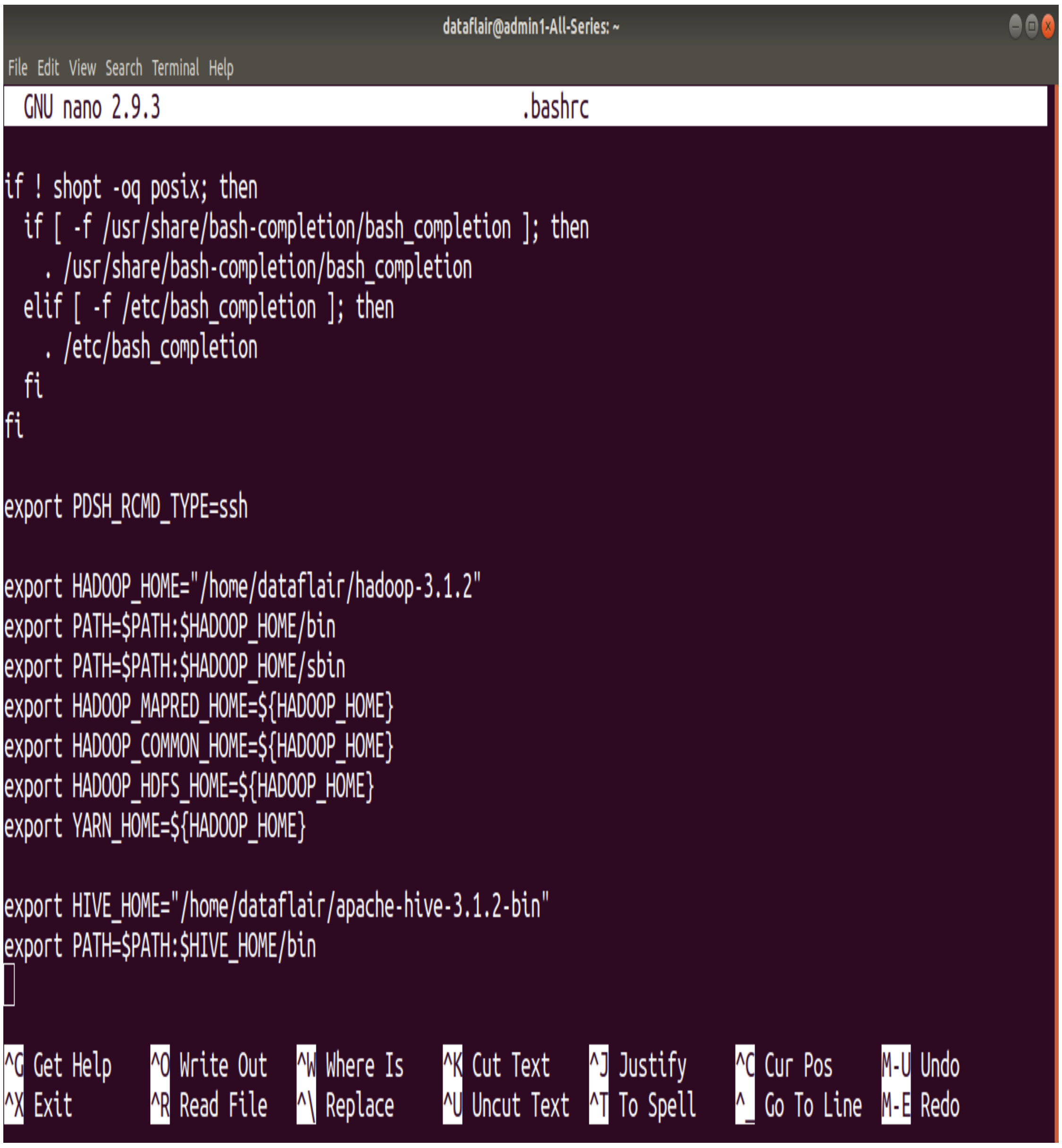
2. Configuring Hive files

Step 4: Now, we have to place the Hive PATH in .bashrc file. For this, open **.bashrc** file in the nano editor and add the following in the .bashrc file.

1. export HIVE_HOME= “home/dataflair/apache-hive-3.1.2-bin”
2. export PATH=$PATH:$HIVE_HOME/bin



```
dataflair@admin1-All-Series:~$ nano .bashrc
```


dataflair@admin1-All-Series: ~
File Edit View Search Terminal Help

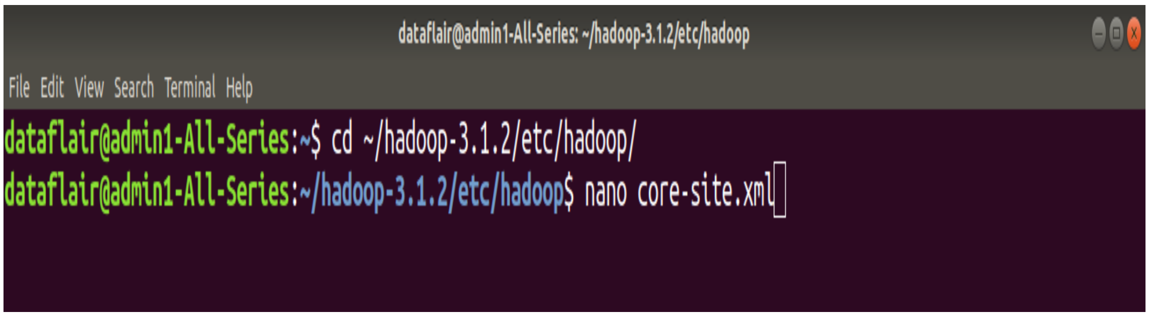
```
GNU nano 2.9.3                         .bashrc

if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export PDSH_RCMD_TYPE=ssh

export HADOOP_HOME="/home/dataflair/hadoop-3.1.2"
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=${HADOOP_HOME}
export HADOOP_COMMON_HOME=${HADOOP_HOME}
export HADOOP_HDFS_HOME=${HADOOP_HOME}
export YARN_HOME=${HADOOP_HOME}

export HIVE_HOME="/home/dataflair/apache-hive-3.1.2-bin"
export PATH=$PATH:$HIVE_HOME/bin


^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line M-E Redo
```

Note: Here enter the correct name & version of your hive and correct path of your Hive File "home/dataflair/apache-hive-3.1.2-bin" this is the path of my Hive File and "apache-hive-3.1.2-bin" is the name of my hive file. So please enter the correct path and name of your Hive file. After adding save this file.
Press **CTRL+O** and enter to save changes. Then press **CTRL+D** to exit the editor.

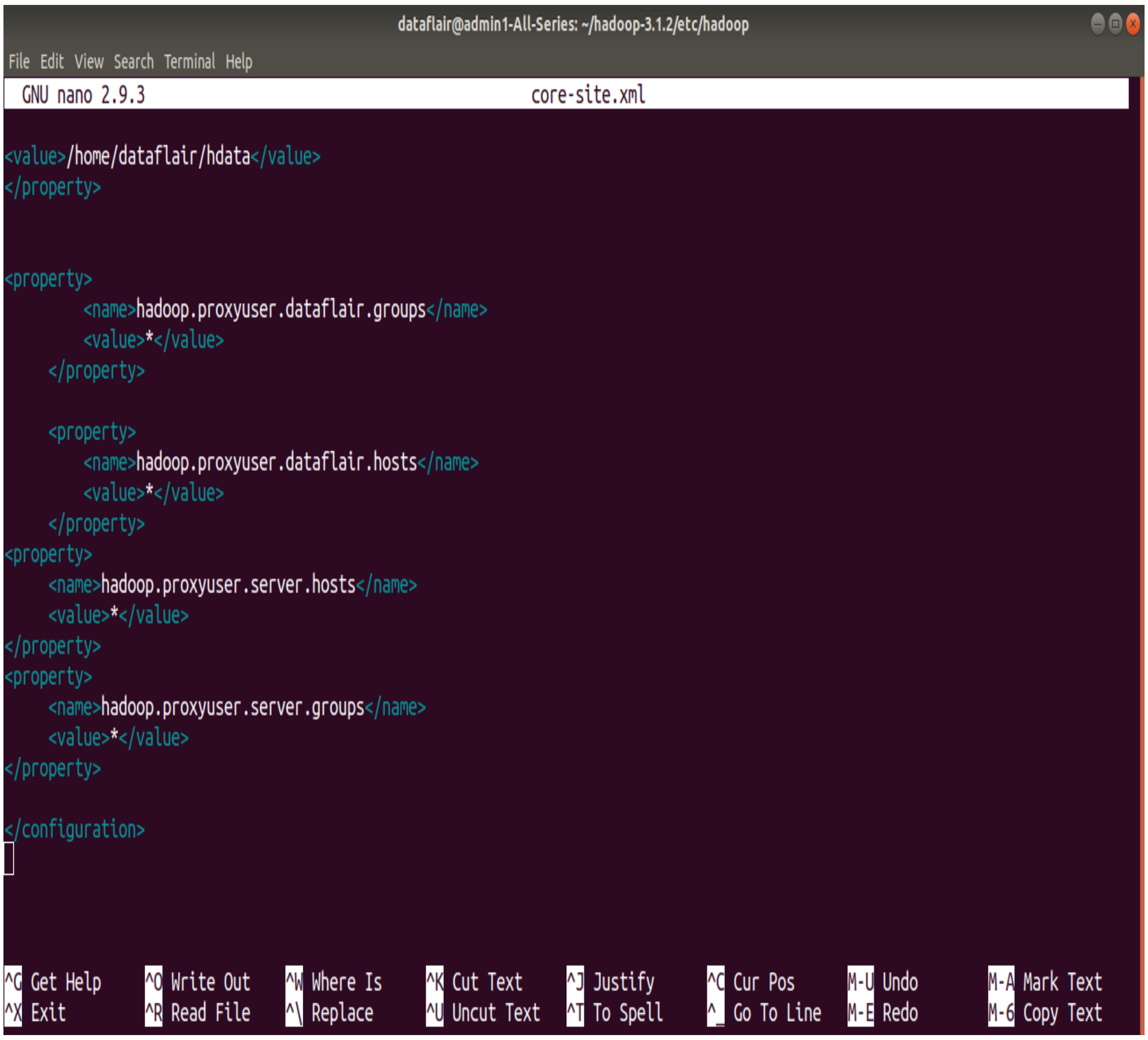
Step 5: Open the **core-site.xml** file in the nano editor. The file is located in **home/hadoop-3.1.2/etc/hadoop/** (Hadoop Configuration Directory).
Add the following configuration property in the core-site.xml file.

```
<configuration>
<property>
<name>hadoop.proxyuser.dataflair.groups</name>
<value>*</value>
</property>
<property>
<name>hadoop.proxyuser.dataflair.hosts</name>
```

```
<value>*</value>
</property>
<property>
<name>hadoop.proxyuser.server.hosts</name>
<value>*</value>
</property>
<property>
<name>hadoop.proxyuser.server.groups</name>
<value>*</value>
</property>
</configuration>
```



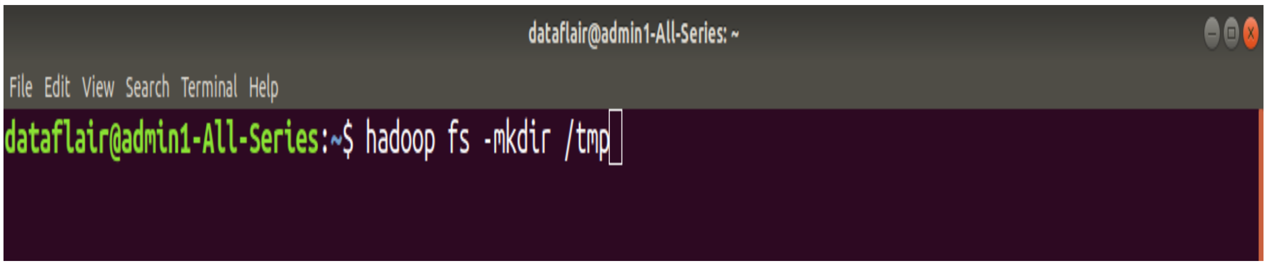
```
dataflair@admin1-All-Series:~$ cd ~/hadoop-3.1.2/etc/hadoop/
dataflair@admin1-All-Series:~/hadoop-3.1.2/etc/hadoop$ nano core-site.xml
```



```
  GNU nano 2.9.3                          core-site.xml

<value>/home/dataflair/hdata</value>
</property>


<property>
         <name>hadoop.proxyuser.dataflair.groups</name>
         <value>*</value>
     </property>

     <property>
         <name>hadoop.proxyuser.dataflair.hosts</name>
         <value>*</value>
     </property>
<property>
     <name>hadoop.proxyuser.server.hosts</name>
     <value>*</value>
</property>
<property>
     <name>hadoop.proxyuser.server.groups</name>
     <value>*</value>
</property>

</configuration>
```

Press **CTRL+O** and enter to save changes. Then press **CTRL+D** to exit the editor.
Step 6: Make a directory 'tmp' in HDFS using the below command:

1. hadoop fs -mkdir /tmp

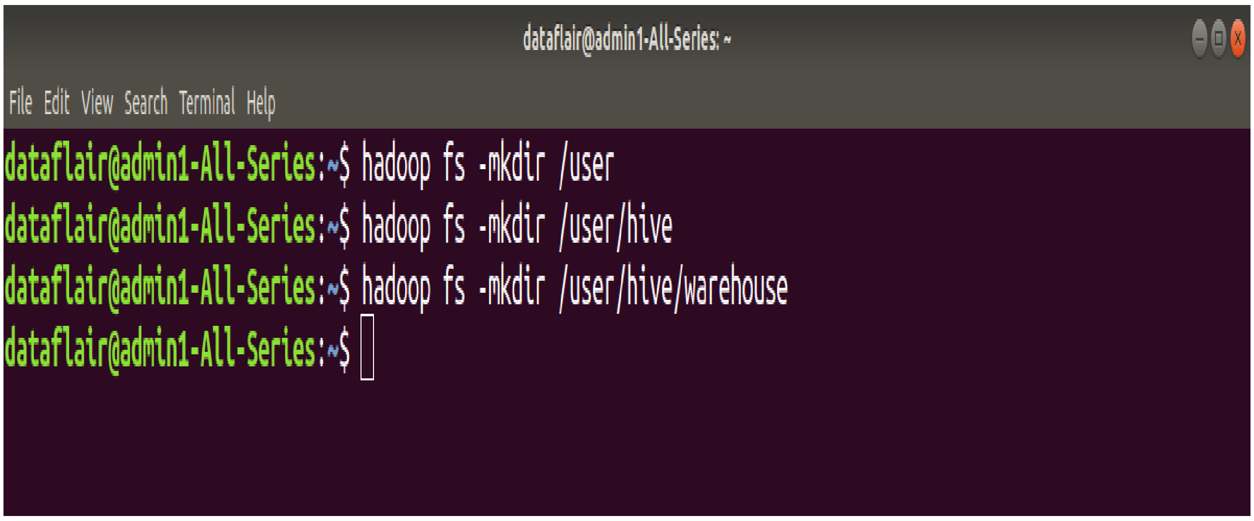


dataflair@admin1-All-Series: ~
File Edit View Search Terminal Help

```
dataflair@admin1-All-Series:~$ hadoop fs -mkdir /tmp
```

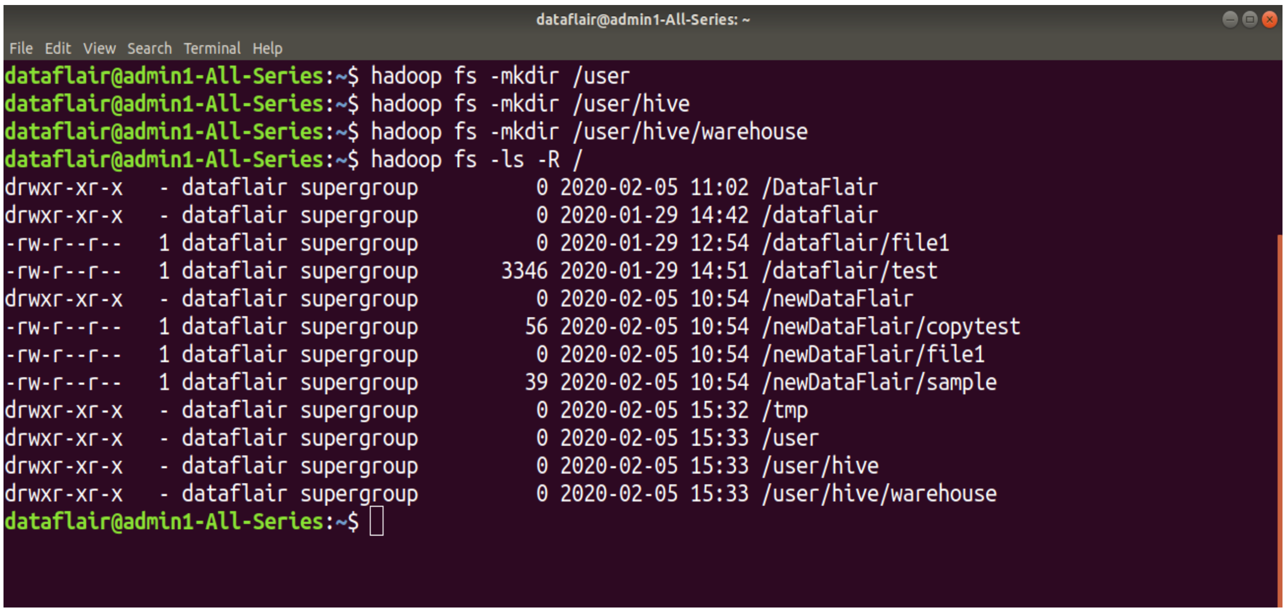
Step 6: Use the below commands to create a directory 'warehouse' inside 'hive' directory, which resides in 'user' directory. The warehouse is the location to store data or tables related to Hive.

1. hadoop fs -mkdir /user
2. hadoop fs -mkdir /user/hive
3. hadoop fs -mkdir /user/hive/warehouse



dataflair@admin1-All-Series: ~
File Edit View Search Terminal Help

```
dataflair@admin1-All-Series:~$ hadoop fs -mkdir /user
dataflair@admin1-All-Series:~$ hadoop fs -mkdir /user/hive
dataflair@admin1-All-Series:~$ hadoop fs -mkdir /user/hive/warehouse
dataflair@admin1-All-Series:~$
```

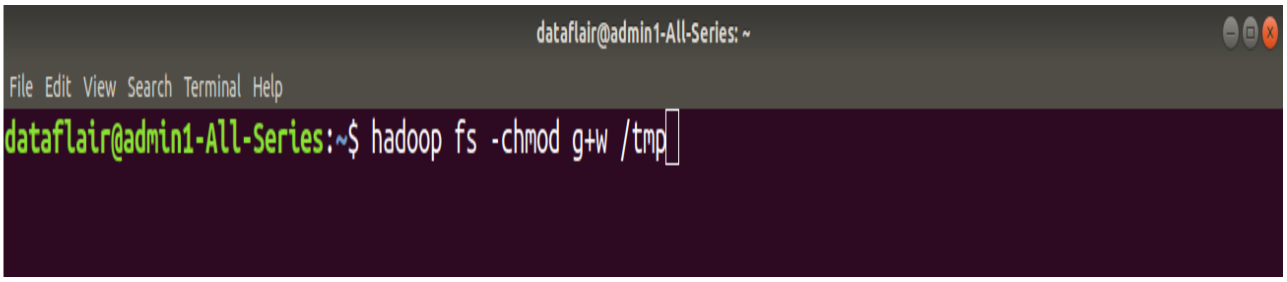


dataflair@admin1-All-Series: ~
File Edit View Search Terminal Help

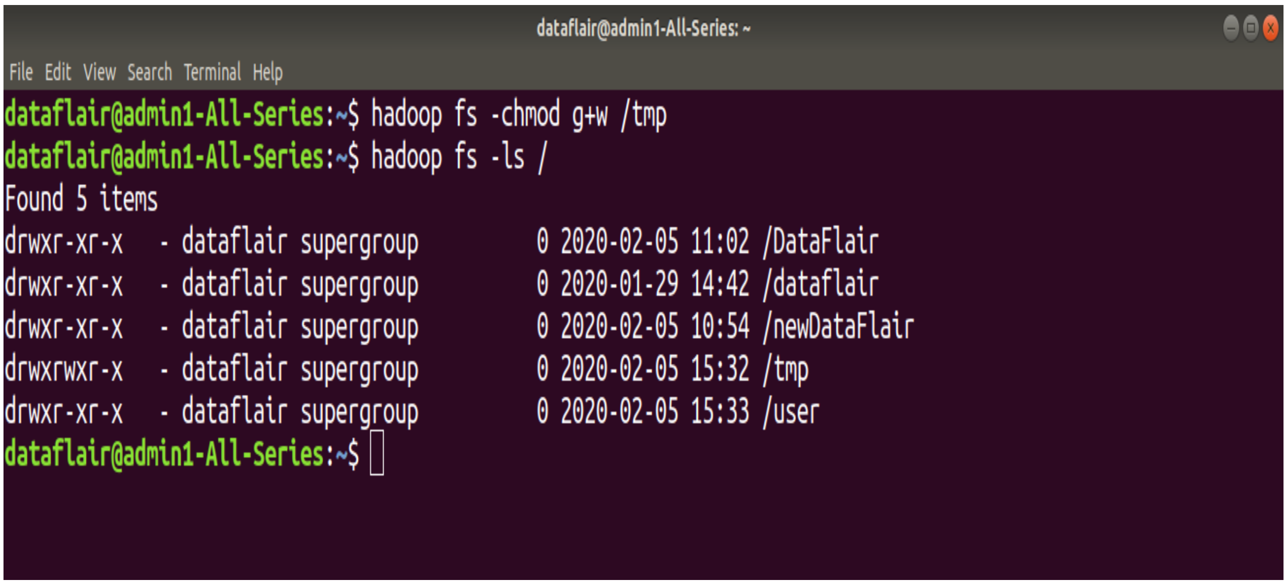
```
dataflair@admin1-All-Series:~$ hadoop fs -mkdir /user
dataflair@admin1-All-Series:~$ hadoop fs -mkdir /user/hive
dataflair@admin1-All-Series:~$ hadoop fs -mkdir /user/hive/warehouse
dataflair@admin1-All-Series:~$ hadoop fs -ls -R /
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 11:02 /DataFlair
drwxr-xr-x   - dataflair supergroup          0 2020-01-29 14:42 /dataflair
-rw-r--r--   1 dataflair supergroup          0 2020-01-29 12:54 /dataflair/file1
-rw-r--r--   1 dataflair supergroup       3346 2020-01-29 14:51 /dataflair/test
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 10:54 /newDataFlair
-rw-r--r--   1 dataflair supergroup         56 2020-02-05 10:54 /newDataFlair/copytest
-rw-r--r--   1 dataflair supergroup          0 2020-02-05 10:54 /newDataFlair/file1
-rw-r--r--   1 dataflair supergroup         39 2020-02-05 10:54 /newDataFlair/sample
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 15:32 /tmp
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 15:33 /user
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 15:33 /user/hive
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 15:33 /user/hive/warehouse
dataflair@admin1-All-Series:~$
```

Step 7: Give the write permission to the members of the 'tmp' file group using command:

1. hadoop fs -chmod g+w /tmp


dataflair@admin1-All-Series: ~
File Edit View Search Terminal Help

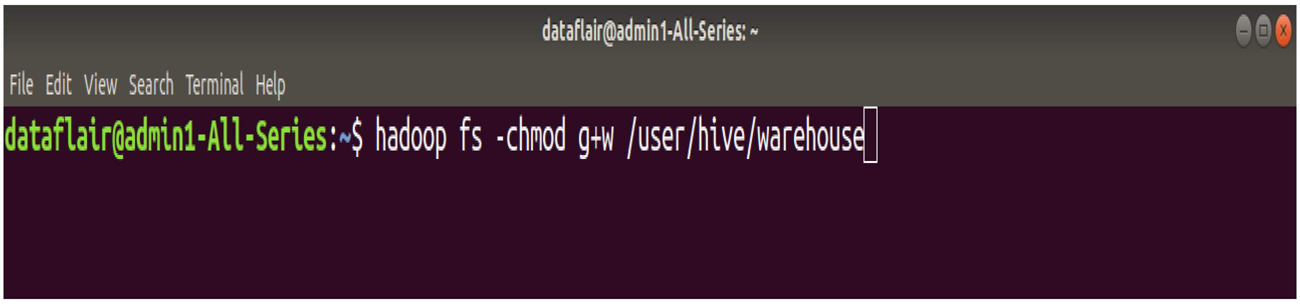
```
dataflair@admin1-All-Series:~$ hadoop fs -chmod g+w /tmp
```


dataflair@admin1-All-Series: ~
File Edit View Search Terminal Help

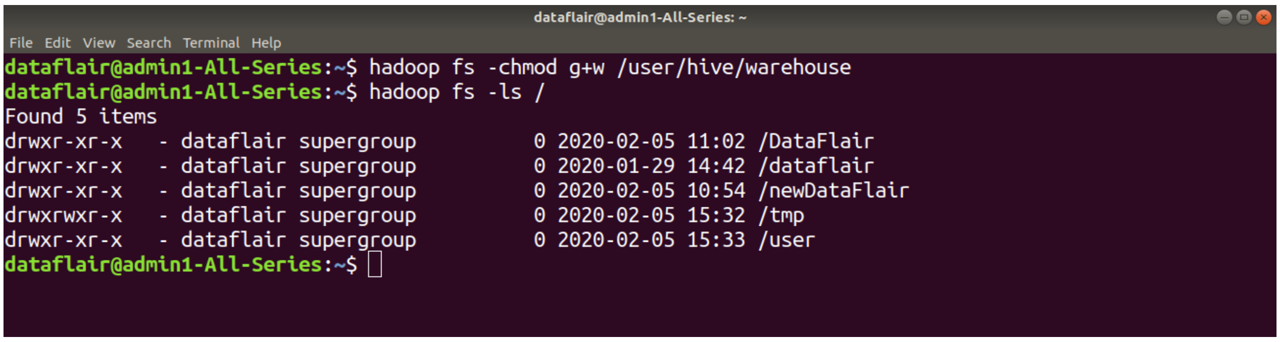
```
dataflair@admin1-All-Series:~$ hadoop fs -chmod g+w /tmp
dataflair@admin1-All-Series:~$ hadoop fs -ls /
Found 5 items
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 11:02 /DataFlair
drwxr-xr-x   - dataflair supergroup          0 2020-01-29 14:42 /dataflair
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 10:54 /newDataFlair
drwxrwxr-x   - dataflair supergroup          0 2020-02-05 15:32 /tmp
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 15:33 /user
dataflair@admin1-All-Series:~$
```

Step 8: Now give write permission to the warehouse directory using the command:

1. hadoop fs -chmod g+w /user/hive/warehouse


dataflair@admin1-All-Series: ~
File Edit View Search Terminal Help

```
dataflair@admin1-All-Series:~$ hadoop fs -chmod g+w /user/hive/warehouse
```

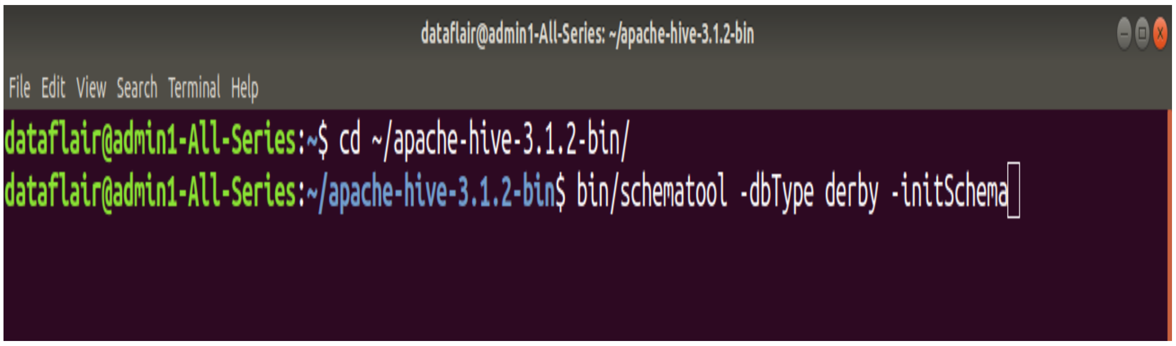

dataflair@admin1-All-Series: ~
File Edit View Search Terminal Help

```
dataflair@admin1-All-Series:~$ hadoop fs -chmod g+w /user/hive/warehouse
dataflair@admin1-All-Series:~$ hadoop fs -ls /
Found 5 items
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 11:02 /DataFlair
drwxr-xr-x   - dataflair supergroup          0 2020-01-29 14:42 /dataflair
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 10:54 /newDataFlair
drwxrwxr-x   - dataflair supergroup          0 2020-02-05 15:32 /tmp
drwxr-xr-x   - dataflair supergroup          0 2020-02-05 15:33 /user
dataflair@admin1-All-Series:~$
```

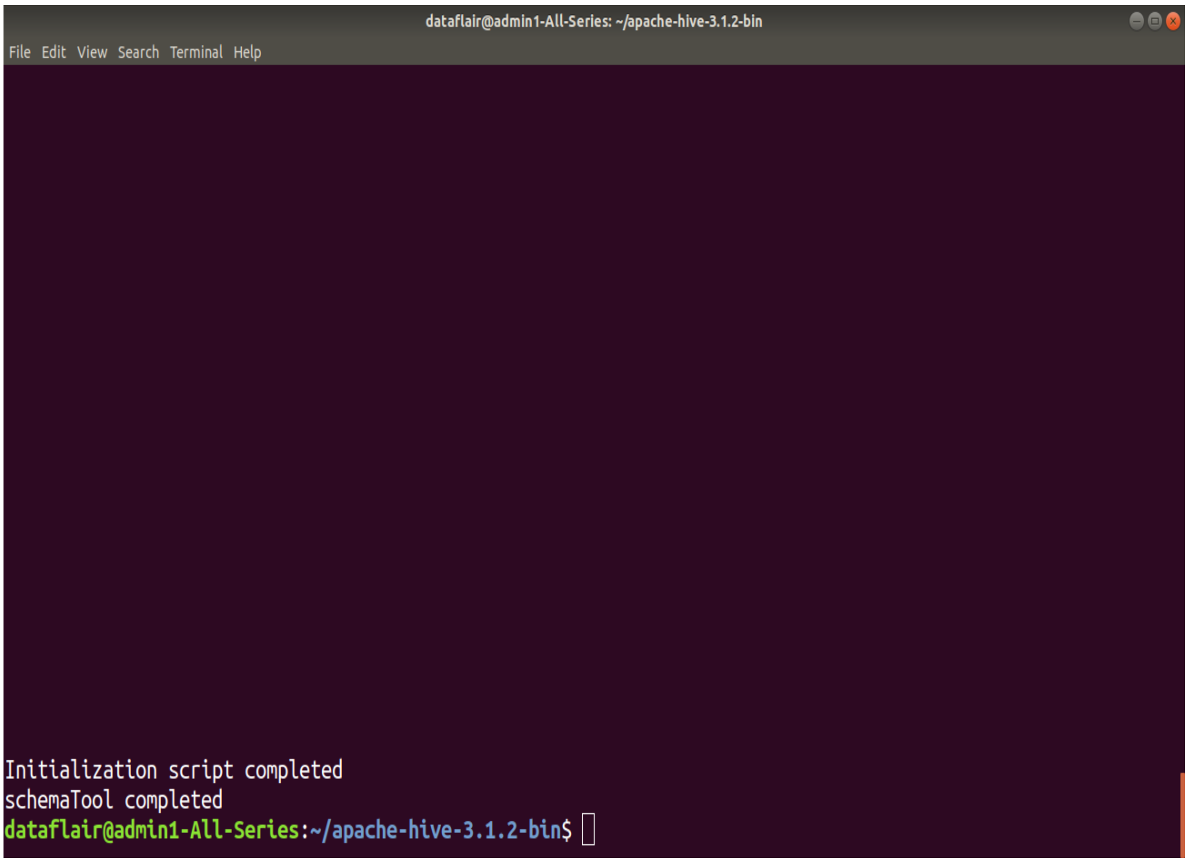
3. Initialize Derby database

Step 9: Hive by default uses **Derby database**. Use the below command to initialize the Derby database.

1. bin/schematool -dbType derby -initSchema



```
dataflair@admin1-All-Series:~$ cd ~/apache-hive-3.1.2-bin/
dataflair@admin1-All-Series:~/apache-hive-3.1.2-bin$ bin/schematool -dbType derby -initSchema
```



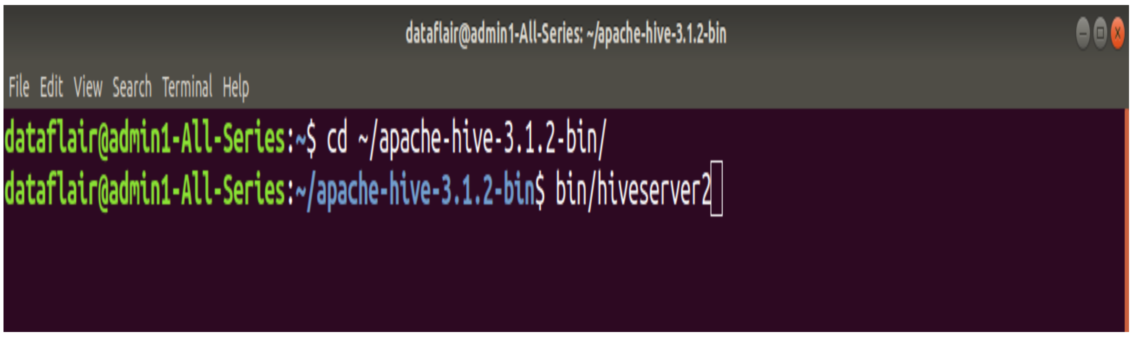
```
Initialization script completed
schemaTool completed
dataflair@admin1-All-Series:~/apache-hive-3.1.2-bin$
```

4. Launching Hive

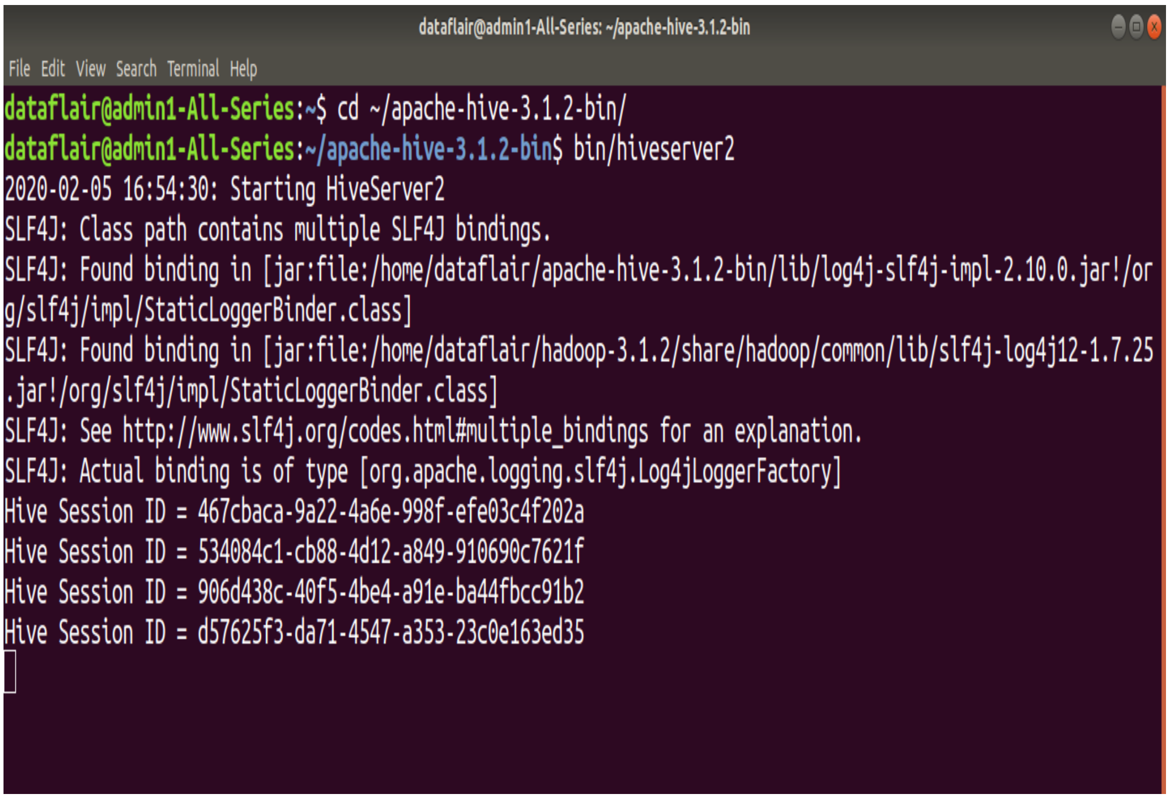
Step 10: Now start the HiveServer2 using the below command:

1. bin/hiveserver2

[For this first move to the ~/apache-hive-3.1.2-bin/]


dataflair@admin1-All-Series: ~/apache-hive-3.1.2-bin
File Edit View Search Terminal Help

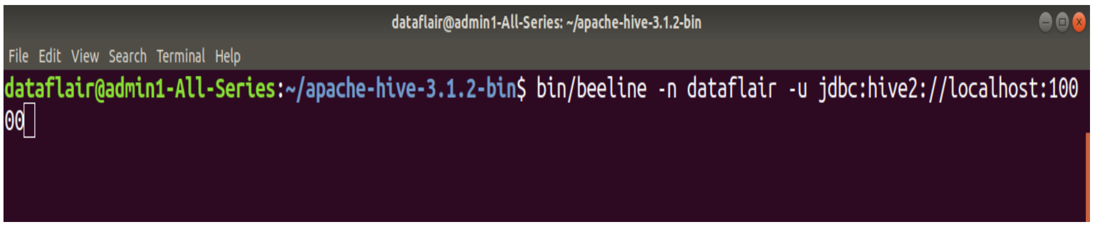
```
dataflair@admin1-All-Series:~$ cd ~/apache-hive-3.1.2-bin/
dataflair@admin1-All-Series:~/apache-hive-3.1.2-bin$ bin/hiveserver2
```


dataflair@admin1-All-Series: ~/apache-hive-3.1.2-bin
File Edit View Search Terminal Help

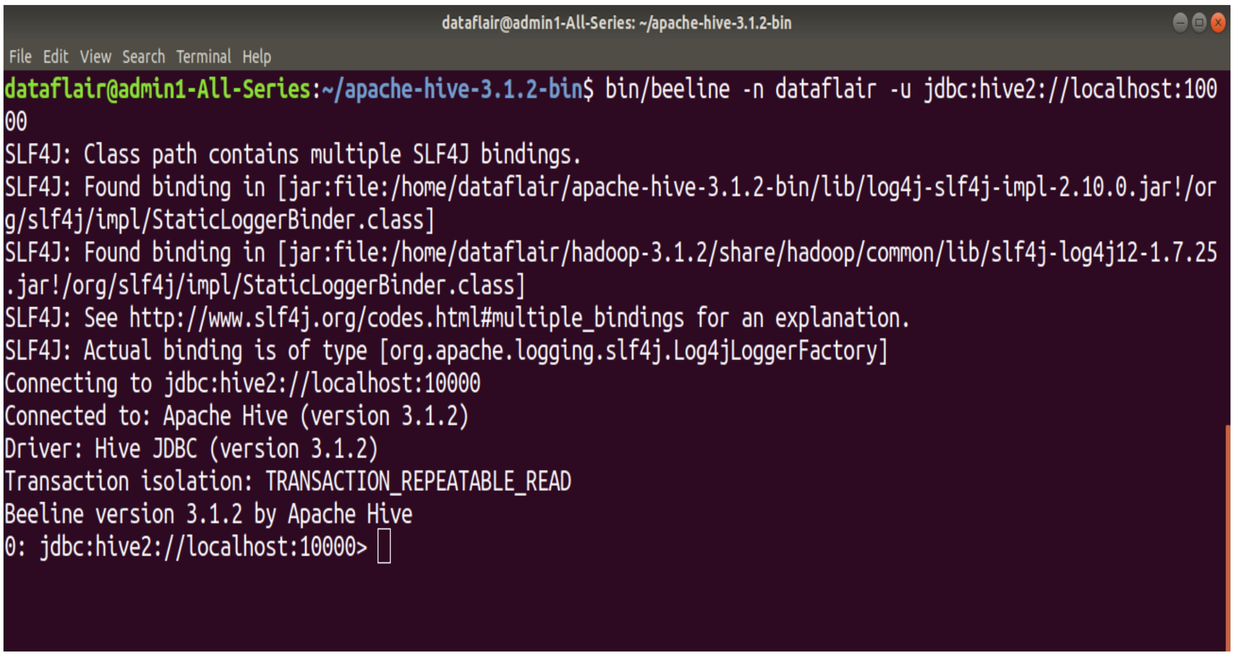
```
dataflair@admin1-All-Series:~$ cd ~/apache-hive-3.1.2-bin/
dataflair@admin1-All-Series:~/apache-hive-3.1.2-bin$ bin/hiveserver2
2020-02-05 16:54:30: Starting HiveServer2
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/dataflair/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/or
g/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/dataflair/hadoop-3.1.2/share/hadoop/common/lib/slf4j-log4j12-1.7.25
.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 467cbaca-9a22-4a6e-998f-efe03c4f202a
Hive Session ID = 534084c1-cb88-4d12-a849-910690c7621f
Hive Session ID = 906d438c-40f5-4be4-a91e-ba44fbcc91b2
Hive Session ID = d57625f3-da71-4547-a353-23c0e163ed35
```

Step 11: On the **different tab**, type the below command to launch the beeline command shell.

```
1. bin/beeline -n dataflair -u jdbc:hive2://localhost:10000
```


dataflair@admin1-All-Series: ~/apache-hive-3.1.2-bin
File Edit View Search Terminal Help

```
dataflair@admin1-All-Series:~/apache-hive-3.1.2-bin$ bin/beeline -n dataflair -u jdbc:hive2://localhost:100
00
```

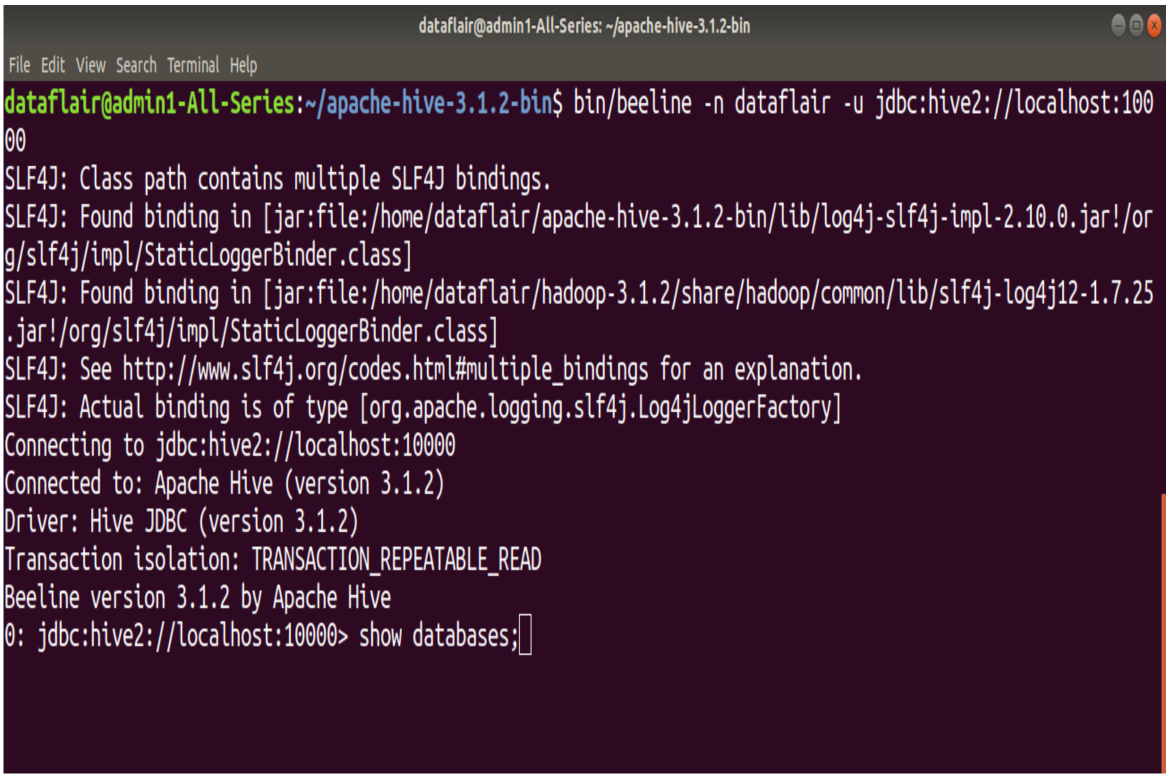

dataflair@admin1-All-Series: ~/apache-hive-3.1.2-bin
File Edit View Search Terminal Help

```
dataflair@admin1-All-Series:~/apache-hive-3.1.2-bin$ bin/beeline -n dataflair -u jdbc:hive2://localhost:100
00
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/dataflair/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/or
g/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/dataflair/hadoop-3.1.2/share/hadoop/common/lib/slf4j-log4j12-1.7.25
.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Connecting to jdbc:hive2://localhost:10000
Connected to: Apache Hive (version 3.1.2)
Driver: Hive JDBC (version 3.1.2)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 3.1.2 by Apache Hive
0: jdbc:hive2://localhost:10000>
```

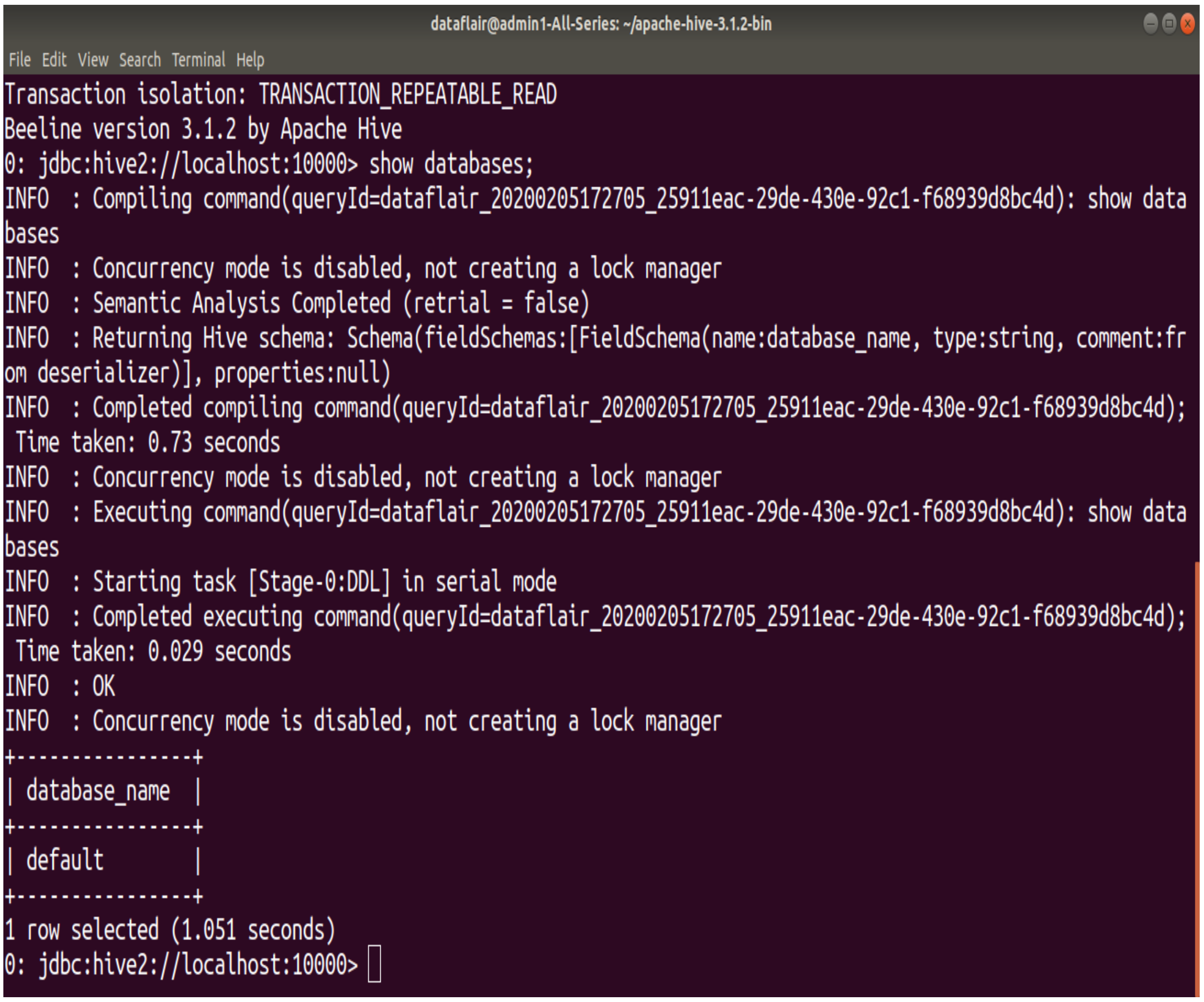
Congratulations!! We have successfully installed Hive 3.1.2 on Ubuntu.
Now, you can type SQL queries in the Beeline command shell to interact with the Hive system.

5. Verifying Hive Installation

For example, in the below image, I am using **show databases** query to list out the database in the Hive warehouse.


dataflair@admin1-All-Series: ~/apache-hive-3.1.2-bin
File Edit View Search Terminal Help

```
dataflair@admin1-All-Series:~/apache-hive-3.1.2-bin$ bin/beeline -n dataflair -u jdbc:hive2://localhost:100
00
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/dataflair/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/or
g/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/dataflair/hadoop-3.1.2/share/hadoop/common/lib/slf4j-log4j12-1.7.25
.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Connecting to jdbc:hive2://localhost:10000
Connected to: Apache Hive (version 3.1.2)
Driver: Hive JDBC (version 3.1.2)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 3.1.2 by Apache Hive
0: jdbc:hive2://localhost:10000> show databases;
```


dataflair@admin1-All-Series: ~/apache-hive-3.1.2-bin
File Edit View Search Terminal Help

```
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 3.1.2 by Apache Hive
0: jdbc:hive2://localhost:10000> show databases;
INFO  : Compiling command(queryId=dataflair_20200205172705_25911eac-29de-430e-92c1-f68939d8bc4d): show data
bases
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:database_name, type:string, comment:fr
om deserializer)], properties:null)
INFO  : Completed compiling command(queryId=dataflair_20200205172705_25911eac-29de-430e-92c1-f68939d8bc4d);
 Time taken: 0.73 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=dataflair_20200205172705_25911eac-29de-430e-92c1-f68939d8bc4d): show data
bases
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=dataflair_20200205172705_25911eac-29de-430e-92c1-f68939d8bc4d);
 Time taken: 0.029 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+----------------+
| database_name  |
+----------------+
| default        |
+----------------+
1 row selected (1.051 seconds)
0: jdbc:hive2://localhost:10000>
```

We have successfully installed Apache Hive 3.1.2 on Hadoop 3.1.2 on Ubuntu.

VIVA QUESTIONS :

1. What is Hadoop? What is Hadoop example?

2. What is Big Data?What is Hadoop in Big Data?

3. List out Hadoop’s three configuration files?

4. What is Apache pig?

5. What is Hive ?

6. What is Apache Hbase

2. Implement a MapReduce program to find word count in the given input file

Aim: MR used to find the word count in the given input file

MapReduce Word Count Example

In MapReduce word count example, we find out the frequency of each word. Here, the role of Mapper is to map the keys to the existing values and the role of Reducer is to aggregate the keys of common values. So, everything is represented in the form of Key-value pair.

Create a text file in your local machine and write some text into it.

```
$ nano data.txt
```

Check the text written in the data.txt file.

```
$ cat data.txt
```

- Create a directory in HDFS, where to kept text file.
  ```
  $ hdfs dfs -mkdir /test
  ```
- Upload the data.txt file on HDFS in the specific directory.
  ```
  $ hdfs dfs -put /home/codegyani/data.txt /test
  ```

File: WC_Mapper.java

```java
package com.cmrit;
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class WC_Mapper extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWritable>{
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();
    public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
            Reporter reporter) throws IOException{
        String line = value.toString();
        StringTokenizer  tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()){
```

```
            word.set(tokenizer.nextToken());
            output.collect(word, one);
        }
    }

}
```

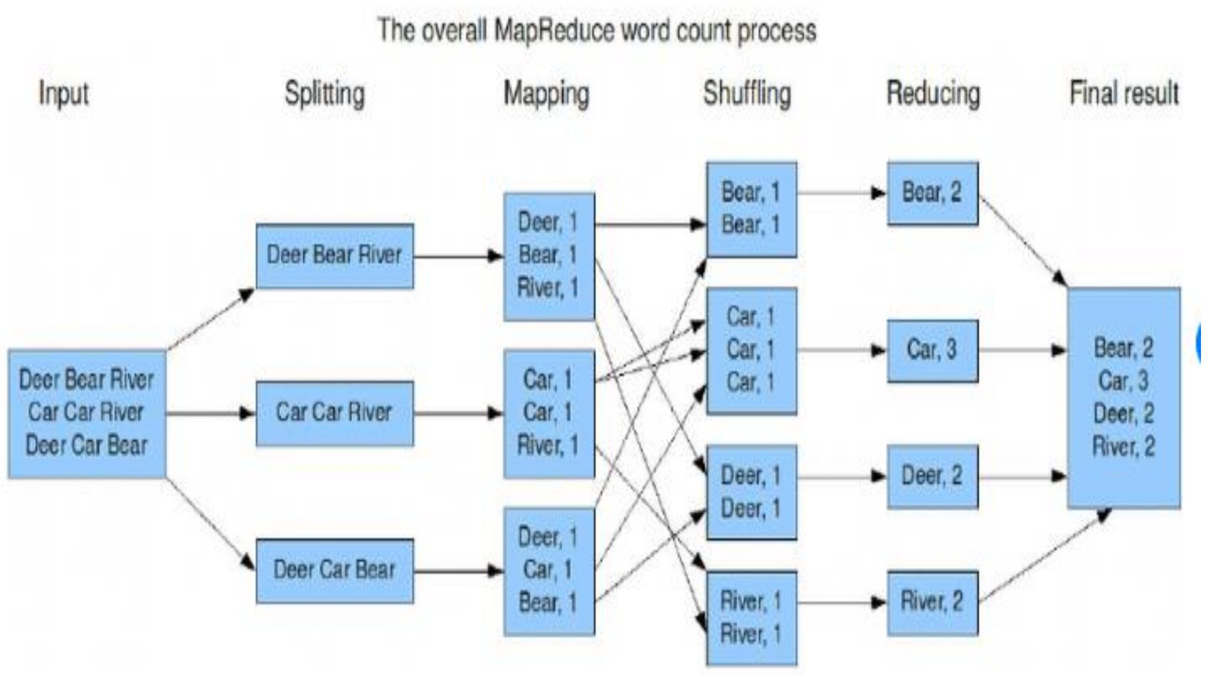
File: WC_Reducer.java

```
package com.cmrit;
    import java.io.IOException;
    import java.util.Iterator;
    import org.apache.hadoop.io.IntWritable;
    import org.apache.hadoop.io.Text;
    import org.apache.hadoop.mapred.MapReduceBase;
    import org.apache.hadoop.mapred.OutputCollector;
    import org.apache.hadoop.mapred.Reducer;
    import org.apache.hadoop.mapred.Reporter;

    public class WC_Reducer  extends MapReduceBase implements Reducer<Text,IntWritable,Text,IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,IntWritable> output,
     Reporter reporter) throws IOException {
    int sum=0;
    while (values.hasNext()) {
    sum+=values.next().get();
    }
    output.collect(key,new IntWritable(sum));
    }
    }
```

File: WC_Runner.java

```
package com.javatpoint;

    import java.io.IOException;
    import org.apache.hadoop.fs.Path;
    import org.apache.hadoop.io.IntWritable;
    import org.apache.hadoop.io.Text;
    import org.apache.hadoop.mapred.FileInputFormat;
    import org.apache.hadoop.mapred.FileOutputFormat;
    import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.TextOutputFormat;
public class WC_Runner {
    public static void main(String[] args) throws IOException{
        JobConf conf = new JobConf(WC_Runner.class);
        conf.setJobName("WordCount");
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        conf.setMapperClass(WC_Mapper.class);
        conf.setCombinerClass(WC_Reducer.class);
        conf.setReducerClass(WC_Reducer.class);
        conf.setInputFormat(TextInputFormat.class);
        conf.setOutputFormat(TextOutputFormat.class);
        FileInputFormat.setInputPaths(conf,new Path(args[0]));
        FileOutputFormat.setOutputPath(conf,new Path(args[1]));
        JobClient.runJob(conf);
    }
}
```

OUTPUT



: Word count program flow executed with MapReduce.

OUTPUT

VIVA QUESTIONS :

1. What is MapReduce?

2. What is Hadoop Map Reduce?

3. Mention what are the data components, core components and data storage components used by Hadoop?

4. What is Shuffling and Sorting in MapReduce?

5. Explain what is “map” and what is "reducer" in Hadoop?

6. What are the main components of MapReduce Job?

3. Implement a MapReduce program to find the Length of the word in the given input file.

Aim: MR used to find Length of the word in the given Input File

Hadoop MapReduce is the core Hadoop ecosystem component which provides data processing. MapReduce is a software framework for easily writing applications that process the vast amount of structured and unstructured data stored in the Hadoop Distributed File system. MapReduce programs are parallel in nature, thus are very useful for performing large-scale data analysis using multiple machines in the cluster. Thus, it improves the speed and reliability of cluster this parallel processing.

Working of MapReduce

Hadoop Ecosystem component ‘MapReduce’ works by breaking the processing into two phases:

- Map phase
- Reduce phase

Each phase has key-value pairs as input and output. In addition, programmer also specifies two functions: map function and reduce function

Map function takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). Read Mapper in detail.

Reduce function takes the output from the Map as an input and combines those data tuples based on the key and accordingly modifies the value of the key. Read Reducer in detail.

Features of MapReduce

- **Simplicity** – MapReduce jobs are easy to run. Applications can be written in any language such as java, C++, and python.

- **Scalability** – MapReduce can process petabytes of data.

- **Speed** – By means of parallel processing problems that take days to solve, it is solved in hours and minutes by MapReduce.

- **Fault Tolerance** – MapReduce takes care of failures. If one copy of data is unavailable, another machine has a copy of the same key pair which can be used for solving the same subtask.

Mapper 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;
import org.apache.hadoop.util.GenericOptionsParser;

public class lengthofword {
 public static class TokenizerMapper extends Mapper<Object, Text, Text, IntWritable> {
  public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
   String line = value.toString();
   String[] letters = line.split(" ");
   for (String letter : letters) {
    context.write(new Text(letter), new IntWritable(letter.length()));
   }
  }
 }
```

Reducer Program

```
 public static class IntSumReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
  public void reduce(Text key, IntWritable values, Context context) throws IOException, InterruptedException {

   context.write(key, values);
  }
 }
```

Driver Program

```
 public static void main(String[] args) throws Exception {
  Configuration conf = new Configuration();
  String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
  if (otherArgs.length != 2) {
   System.err.println("Usage: WordCount <in> <out>");
   System.exit(2);
  }
```

```java
Job job = new Job(conf, "word count by www.ssaik.com");
job.setJarByClass(lengthofword.class);
job.setMapperClass(TokenizerMapper.class);
job.setCombinerClass(IntSumReducer.class);
// filesystem fs = filesystem.get(conf);
job.setReducerClass(IntSumReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
// if(fs.exists(outputDir))
// fs.delete(outputDir,true);
 }
}
```

Sample Input and Output

Input File

```
siva
ram
som
phani
sasi
manik
sreenu
```

Desired output:--

```
3     ram,som
4      siva,sasi
5      phani,manik
6      sreenu
```

OUTPUT:

VIVA QUESTIONS :

1. Explain what are the basic parameters of a Mapper and reducer?

2. What is MapReduce example?

3. How Hadoop MapReduce works?

4. Explain what is the function of MapReduce partitioner?

5. What is the key- value pair in Hadoop MapReduce?

6. Where sorting is done in Hadoop MapReduce Job?

4. Implement a MR/Hive/Pig/HBase program that processes the Weather dataset to find the maximum temperature.

Aim: MR used to find the Maximum Temperature in the given dataset

Weather Dataset

Weather sensors collect data every hour at many locations across the globe and gather a large volume of log data. This data is very much suitable for analysis with MapReduce because it is semi- structured and recordoriented.

Mapper Program

Mapper for the maximum temperature example

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class MaxTemperatureMapper extends Mapper
{
 private static final int MISSING = 9999;
@Override
public void map(LongWritable key, Text value, Context context)throws IOException, InterruptedException {
String line = value.toString();
String year = line.substring(15, 19);
int airTemperature;
if (line.charAt(87) == '+') { // parseInt doesn't like leading plus signs airTemperature = Integer.parseInt(line.substring(88, 92));
} else {
airTemperature = Integer.parseInt(line.substring(87, 92));
}
String quality = line.substring(92, 93);
if (airTemperature != MISSING && quality.matches("[01459]")) {
context.write(new Text(year), new IntWritable(airTemperature));
}
}
}
```

Reducer Program

Reducerfor the maximum temperature example

```
 import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
public class MaxTemperatureReducer extends Reducer
{
@Override
public void reduce(Text key, Iterable values, Context context)
throws IOException, InterruptedException
{
int maxValue = Integer.MIN_VALUE;
for (IntWritable value : values)
{
maxValue = Math.max(maxValue, value.get());
}
context.write(key, new IntWritable(maxValue));
} }
```

Driver program

Application to find the maximum temperature in the weather dataset

```
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.lib.input.FileInputFormat;

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

public class MaxTemperature

 {

 public static void main(String[] args) throws Exception

{

if (args.length != 2)

 {

 System.err.println("Usage: MaxTemperature <input path><outputpath”);

System.exit(-1);

}

Job job = new Job();

job.setJarByClass(MaxTemperature.class);

job.setJobName("Max temperature");

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

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

job.setMapperClass(MaxTemperatureMapper.class);

job.setReducerClass(MaxTemperatureReducer.class);

job.setOutputKeyClass(Text.class);

 job.setOutputValueClass(IntWritable.class);

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

} }
```

OUTPUT:

VIVA QUESTIONS :

1. **What is NameNode, Task Tracker and Job Tracker and its port numbers?**

2. **Mention when to use Map reduce mode?**

3. **What is the sequence of execution of map, reduce, recordreader, split, combiner, partitioner?**

4. **What are the various InputFormats in Hadoop?**

5. **What are the various OutputFormats in Hadoop?**

6. **What is KeyValueTextInputFormat in Hadoop MapReduce?**

5. Implement a WIKI DataMining Program to find the page counts by using Pig/Hive.

Hint: (take 3 fields like language(en,us), searchengine name(yahoo,google), pageclicks)

AIM: PIG Program is used to fine the page counts in the given WIKI File

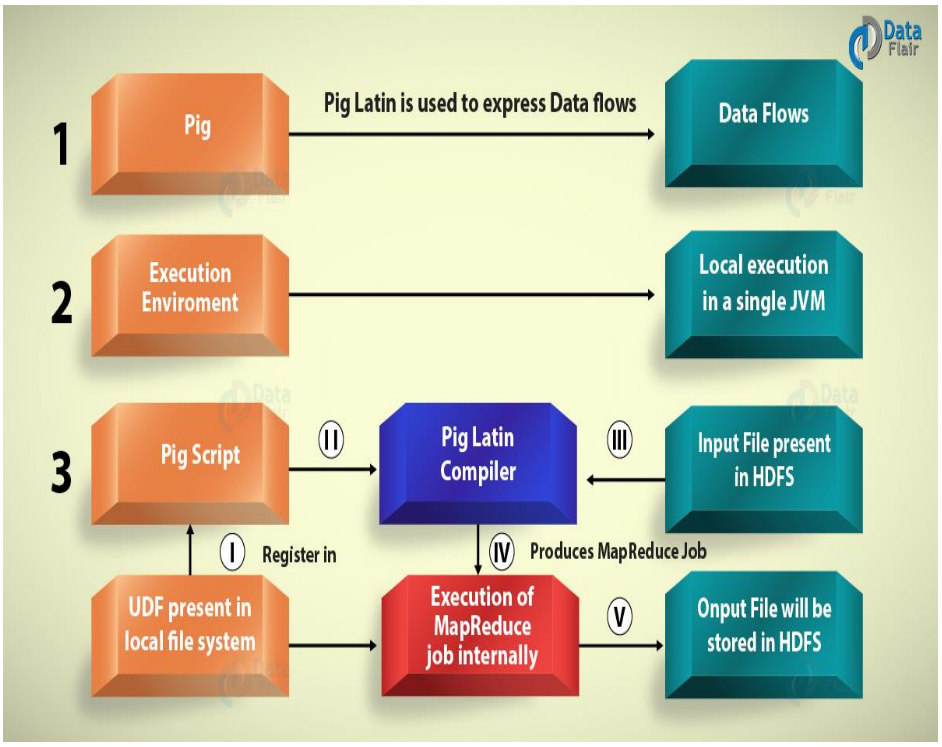
Pig: Apache Pig is a high-level language platform for analyzing and querying huge dataset that are stored in HDFS. Pig as a component of Hadoop Ecosystem uses PigLatin language. It is very similar to SQL. It loads the data, applies the required filters and dumps the data in the required format. For Programs execution, pig requires Java runtime environment.

Features of Apache Pig:

•**Extensibility** – For carrying out special purpose processing, users can create their own function.

•**Optimization opportunities** – Pig allows the system to optimize automatic execution. This allows the user to pay attention to semantics instead of efficiency.

•**Handles all kinds of data** – Pig analyzes both structured as well as unstructured.


1
Pig
Pig Latin is used to express Data flows
Data Flows
2
Execution Enviroment
Local execution in a single JVM
3
Pig Script
II
Pig Latin Compiler
III
Input File present in HDFS
I
Register in
IV
Produces MapReduce Job
UDF present in local file system
Execution of MapReduce job internally
V
Onput File will be stored in HDFS

a). PIG-ProgramCode: myprogram.pig

```
records = LOAD 'wiki-input' using PigStorage(' ') as
(projectname:chararray,pagename:chararray,clicks:int,pagesize:int);
```

```
en google 30 30000
en yahoo 50 50000
us yahoo 60 60000
en google 70 70000
```

fil_records = FILTER records by projectname == 'en';

```
en google 30 30000
en yahoo 50 50000
en google 70 70000
```

grp_records = GROUP fil_records by pagename;

```
google{(en google 30
30000)(en google 70 70000)}
yahoo{(en yahoo 50 50000)}
```

results = FOREACH grp_records generate group,SUM(fil_records.clicks);

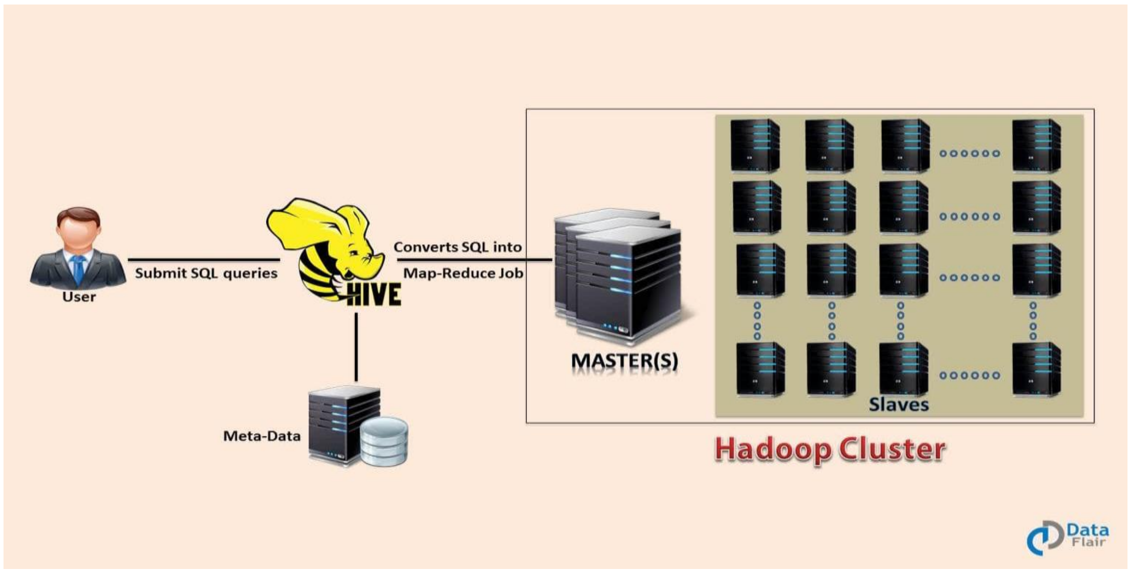
```
google 100
yahoo 50
```

sorted_results = ORDER results by $0 ASC;
STORE sorted_results into **'wiki-output'**;

Execution:--
create a file in linux with pig extension
grunt:/> run /home/XXX/myprogram.pig

b)Hive WIKI program

Hive: The Hadoop ecosystem component, Apache Hive, is an open source data warehouse system for querying and analyzing large datasets stored in Hadoop files. Hive do three main functions: data summarization, query, and analysis.Hive use language called HiveQL (HQL), which is similar to SQL. HiveQL automatically translates SQL-like queries into MapReduce jobs which will execute on Hadoop.



Main parts of Hive are:
•**Metastore** – It stores the metadata.
•**Driver** – Manage the lifecycle of a HiveQL statement.
•**Query compiler** – Compiles HiveQL into Directed Acyclic Graph(DAG).
•**Hive server** – Provide a thrift interface and JDBC/ODBC server.

```
hive> create external table samata (projname string, pagename string,
clicks int, pagesize int)
    > row format delimited fields terminated by ' '
    > lines terminated by '\n'
    > stored as textfile;
OK
Time taken: 0.102 seconds

hive> load data LOcal inpath '/home/sbkt/wiki' into table samata;
Copying data from file:/home/sbkt/wiki
Copying file: file:/home/sbkt/wiki
Loading data to table default.samata
OK
Time taken: 0.195 seconds

hive> select * from samata;
OK
en     google      30    3000
en     yahoo 60    6000
en     google      70    7000
us     yahoo 50    5000
```

```
Time taken: 0.106 seconds
hive> create table latha (pagename string, totalclicks int);
OK

hive> insert overwrite table latha
    > select pagename,sum(clicks) from samata
    > where projname='en' group by pagename;

MapReduce Total cumulative CPU time: 1 seconds 610 msec
Ended Job = job_201602272332_0004
Loading data to table default.latha
Deleted hdfs://localhost:54310/user/sbkt/warehouse/latha
Table default.latha stats: [num_partitions: 0, num_files: 1, num_rows:
0, total_size: 20, raw_data_size: 0]
2 Rows loaded to latha
MapReduce Jobs Launched:
Job 0: Map: 1  Reduce: 1   Cumulative CPU: 1.61 sec   HDFS Read: 278
HDFS Write: 20 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 610 msec
OK
Time taken: 31.831 seconds


hive> select * from latha;
OK
google      100
yahoo        60
Time taken: 0.119 seconds
hive> exit
    > ;
```

Sample Input File and Output

Input File: wiki-input.pig

en google 30 30000
en yahoo 50 50000
us yahoo 60 60000
en google 70 70000

Output File: WIKI-Output.pig

google 100
yahoo 50

Result:

VIVA QUESTIONS :

1. **What kind of applications is supported by Apache Hive?**

2. **What are the different types of tables available in HIve?**

3. **What are the three different modes in which hive can be run?**

4. **hat is a metastore in Hive?**

5. **When should we use SORT BY instead of ORDER BY?**

6. **What is a partition in Hive?**

6. Implement a Hive/MR/Pig/HBase program to find the Length of the Voter data with city .

Aim: PIG program is used to find the Length of the Voter data with city.
Apache Pig is a high-level language platform for analyzing and querying huge dataset that are stored in HDFS. Pig as a component of Hadoop Ecosystem uses PigLatin language. It is very similar to SQL. It loads the data, applies the required filters and dumps the data in the required format. For Programs execution, pig requires Java runtime environment.

PIG program

Step 1: voterrecords = LOAD 'voterIN' using PigStorage(' ') as (id:int, name:chararray, state:chararray, city:chararray, gender:chararray);

```
(10101,sasidhar,AP,nlr,M)
(20134,Nagaraj,KA,rcr,M)
(20564,Imthiyaz,KA,glb,M)
(20189,Manjunath,KA,rcr,M)
(30178,Andal,TN,mdr,F)
(28145,Lakshmi,KA,rcr,F)
(14567,Srinivas,AP,tpty,M)
(35678,Murugan,TN,chi,M)
```

Step 2: group_records = GROUP voterrecords by (state,gender);

```
((AP,M),{(10101,sasidhar,AP,nlr,M),(14567,Srinivas,AP,tpty,M)})
((KA,F),{(28145,Lakshmi,KA,rcr,F)})
((KA,M),{(20134,Nagaraj,KA,rcr,M),(20564,Imthiyaz,KA,glb,M),(20189,Manjunath,KA,rcr,M)})
((TN,F),{(30178,Andal,TN,mdr,F)})
((TN,M),{(35678,Murugan,TN,chi,M)})
```

Step 3: result = FOREACH group_records generate group, COUNT(voterrecords.gender);

```
((AP,M),2)
((KA,F),1)
((KA,M),3)
((TN,F),1)
((TN,M),1)
```

Step 4: STORE result into 'PigVoterOP';

6b).To find the Length of the Voter data with city using Map Reducer

<u>Driver program</u>

```
    package voterlist;
     import org.apache.hadoop.conf.Configuration;
     import org.apache.hadoop.conf.Configured;
     import org.apache.hadoop.fs.Path;
     import org.apache.hadoop.io.IntWritable;
     import org.apache.hadoop.io.LongWritable;
     import org.apache.hadoop.io.Text;
     import org.apache.hadoop.mapreduce.Job;
     import org.apache.hadoop.mapreduce.Reducer;
     import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
     import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
     import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
     import org.apache.hadoop.util.Tool;
     import java.io.IOException;

     public class voterdriver{
      public static void main(String[] args) throws
IOException,ClassNotFoundException, InterruptedException {
             Job job=new Job(new Configuration());
             job.setJobName("voter list");
             job.setJarByClass(voterdriver.class);
             job.setInputFormatClass(TextInputFormat.class);
             job.setMapOutputKeyClass(Text.class);
             job.setMapOutputValueClass(LongWritable.class);
             job.setOutputKeyClass(Text.class);
             job.setOutputValueClass(IntWritable.class);
             job.setMapperClass(votermapper.class);
            job.setReducerClass(voterreducer.class);
             FileInputFormat.addInputPath(job, new Path(args[0]));
             FileOutputFormat.setOutputPath(job, new Path(args[1]));
              job.waitForCompletion(true);
                       }}
```

Mapper Program

```
 package voterlist;
 import java.io.IOException;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Mapper.Context;

public class votermapper extends Mapper<LongWritable, Text,Text, LongWritable>
{
   public void map(LongWritable key, Text value, Context output) throws
IOException, InterruptedException
   {
      String Line= value.toString(); //for storing into string
      String a[]=Line.split(" ");   // for split based on space
      output.write(new Text(a[2] +"   "+ a[3]  +"  "+a[4] ), new LongWritable(1));
   }
    }
```

Reducer program

```
 package voterlist;
import java.io.IOException;
 import org.apache.hadoop.io.IntWritable;
 import org.apache.hadoop.io.LongWritable;
 import org.apache.hadoop.io.Text;
 import org.apache.hadoop.mapreduce.Reducer;
 import org.apache.hadoop.mapreduce.Reducer.Context;

 public class voterreducer extends Reducer<Text, LongWritable, Text, IntWritable>
 {
public void reduce(Text key, Iterable<LongWritable> value, Context output) throws IOException,
InterruptedException
   {
      int sum=0;
       for(LongWritable val:value){
          sum +=val.get();
      }
      output.write(key, new IntWritable(sum));
   }

 }
```

Sample Input and Output

Input:--

```
10101 sasidhar AP nlr M
20134 nagaraj KA rcr M
20564 imthiyaz KA glb M
20189 manjunath KA rcr M
30178 andal TN mdr F
28145 lakshmi KA rcr F
14567 srinivas AP tpty M
35678 murugan TN chi M
```

output:--

```
AP nlr M 1
AP tpty M 1
KA rcr F 1
KA glb M 1
KA rcr M 2
TN mdr F 1
TN chi M 1
```

Text input format

mapper :-

```
input key - offset value -data
                    001 10101 sasidhar AP nlr M
                    ---------------------
output key -- state+gender  value - 1
          output.write(new Text(a[2]+" "+a[4]), new...
                    AP M 1
                    AP M 1
                    KA M 1
                    KA M 1
                    KA M 1
                    TN F 1
                    TN F 1
```

GROUPING: Reducer:--

```
input :--  AP M (1,1)
          KA M (1,1,1)
           TN F(1,1)
output:- sum=sum+values;
          AP M 1
          KA  M  3
          TN  F    2
```

OUTPUT

VIVA QUESTIONS :

1. **What are the components of Pig Execution Environment?**

2. **What are the data types of Pig Latin?**

3. **List the relational operators in Pig.**

4. **How Apache Pig deals with the schema and schema-less data?**

5. **What is UDF?**

6. **What are the limitations of the Pig?**

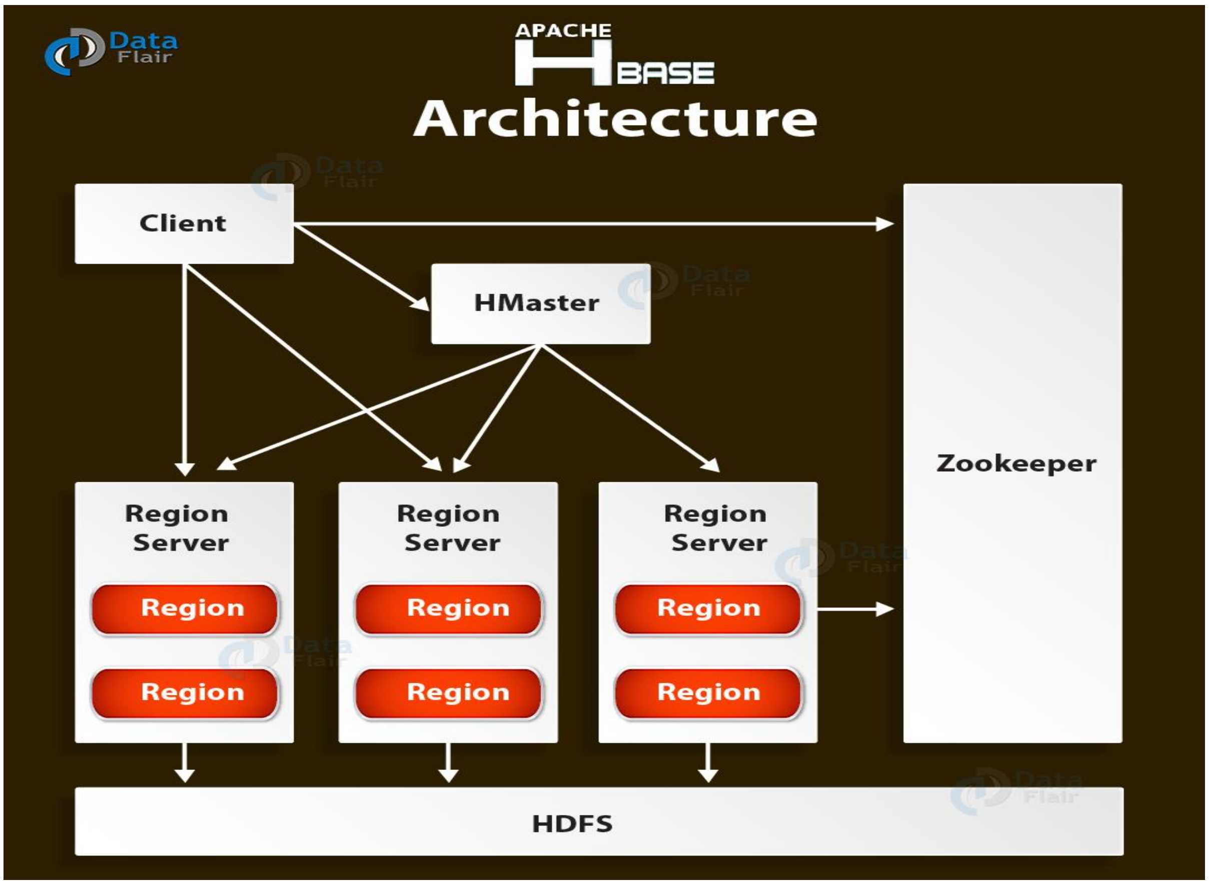
7. Implement a Hive/Hbase/MR/Pig program by taking 6 subjects marks of the students in the EXCEL sheet to find the followings

a)find the max and min marks in the each subject

b)find the count of pass and fail students in each the subject

HBase:Apache HBase is a Hadoop ecosystem component which is a distributed database that was designed to store structured data in tables that could have billions of row and millions of columns. HBase is scalable, distributed, and NoSQL database that is built on top of HDFS. HBase, provide real-time access to read or write data in HDFS.

HBase Diagram



There are two HBase Components namely- HBase Master and RegionServer.

i. HBase Master

It is not part of the actual data storage but negotiates load balancing across allRegionServer.

- Maintain and monitor the Hadoop cluster.
- Performs administration (interface for creating, updating and deleting tables.)
- Controls the failover.
- HMaster handles DDL operation.

ii. RegionServer

It is the worker node which handles read, writes, updates and delete requests from clients. Region server process runs on every node in Hadoop cluster. Region server runs on HDFS DateNode.

<u>HBaseProgram</u>

7a)find the max and min marks in the each subject

```
1.CREATE TABLE Student
        (StudentID int, StudentName varchar(6), Details varchar(1));

INSERT INTO Student
        (StudentID, StudentName, Details)
VALUES
        (1, 'John', 'X'),
        (2, 'Paul', 'X'),
        (3, 'George', 'X'),
        (4, 'Paul', 'X');
2.CREATE TABLE Subject
        (SubjectID varchar(1), SubjectName varchar(7));
INSERT INTO Subject
        (SubjectID, SubjectName)
VALUES
        ('M', 'Math'),
        ('E', 'English'),
        ('H', 'History');
3.CREATE TABLE Mark
        (StudentID int, SubjectID varchar(1), MarkRate int);

INSERT INTO Mark
        (StudentID, SubjectID, MarkRate)
VALUES
        (1, 'M', 90),
        (1, 'E', 100),
        (2, 'M', 95),
        (2, 'E', 70),
        (3, 'E', 95),
        (3, 'H', 98),
        (4, 'H', 90),
        (4, 'E', 100);
```

Maximum marks

****

```
select sb.SubjectName, m.MarkRate, st.StudentName
   from Mark as m
   join Student as st on st.StudentID = m.StudentID
   join Subject as sb on sb.SubjectID = m.SubjectID
      join
    (
    select SubjectID, max(MarkRate) as MaxMarkRate
    from Mark
    group by SubjectID
    ) as x on m.SubjectID = x.SubjectID AND m.MarkRate = x.MaxMarkRate
   order by sb.SubjectName, st.StudentName
```

subjectname	markrate	studentname
English	100	John
English	100	Paul
History	98	George
Math	95	Paul

****

MinimumMarks

```
select sb.SubjectName, m.MarkRate, st.StudentName
   from Mark as m
   join Student as st on st.StudentID = m.StudentID
   join Subject as sb on sb.SubjectID = m.SubjectID
      join
    (
    select SubjectID, Min(MarkRate) as MinMarkRate
    from Mark
    group by SubjectID
    ) as x on m.SubjectID = x.SubjectID AND m.MarkRate = x.MinMarkRate
```

```
order by sb.SubjectName, st.StudentName
```

subjectname	markrate	studentname
English	70	Paul
History	90	Paul
Math	90	John

****

7b) find the count of pass and fail students in each the subject

```
CREATE TABLE students
        (id int not null, studentName varchar(6), Marks int not null);

INSERT INTO students
        (id, studentName, Marks)
VALUES
        (1, 'John', 50 ),
        (2, 'Paul', 35 ),
        (3, 'George', 60 ),
        (4, 'Paul', 35 );

 select s.id, s.studentName, s.Marks,
     case when s.Marks >= 50 then 'Pass' else 'Fail' end as Status,
     case when s.Marks >= 50 then t.TotalPass else t.TotalFail end as TotalCount
   from Students s,
      (select SUM(case when Marks >= 50 then 1 else 0 end) as TotalPass,
           SUM(case when Marks < 50 then 1 else 0 end) as TotalFail
         from Students) t
```

id	studentname	marks	status	totalcount
1	John	50	Pass	2
2	Paul	35	Fail	2
3	George	60	Pass	2
4	Paul	35	Fail	2

Output

VIVA QUESTIONS :

1. **What is the use of Apache HBase?**

2. **Give the name of the key components of HBase**

3. **What is HBase Bloom Filter?**

4. **Explain the data model of HBase.**

5. **What are the data manipulation commands of HBase?**

6. **Define catalog tables and compaction in HBase?**

7. **Can you explain data versioning?**

8. **Customer ATM transactions with different Bank Debit cards data load into either in the notepad/ excel sheet with following fields like (DeditCard no, customer name, date, time, bank name, withdraw amount, status). Write a program by using Big data frameworks like MapReduce/Pig/Hive/HBase to evaluate the following Bank transactions.**
    a. **Bank wise success transactions list (bank name, times using card, amount with drawn)**
    b. **Account wise how many times the card has been used in the current month.**
    c. **Bank wise failed transactions list with detail**
    d. **Maximum transaction for each bank for last month**
    e. **Minimum transaction for each bank for last month**
    f. **Average transaction for each bank for last month**
    g. **in last month, who used this card more than 2 times.**
    h. **All the banks all the card member details required except ICICI Bank.**

a) **Bank wise success transactions list (bank name, times using card, amount with drawn)**

Driver program

```
package com.TransList;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import java.io.IOException;

public class TransListDriver{
 public static void main(String[] args) throws IOException,ClassNotFoundException,
InterruptedException {
        Job job=new Job(new Configuration());
        job.setJobName("Transaction List");
        job.setJarByClass(TransListDriver.class);
        job.setInputFormatClass(TextInputFormat.class);
        job.setMapOutputKeyClass(Text.class);
```

```
                job.setMapOutputValueClass(LongWritable.class);
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(Text.class);
                job.setMapperClass(TransListMapper.class);
                job.setReducerClass(TransListReducer.class);
                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new Path(args[1]));
                job.waitForCompletion(true);
            }
        }
```

Mapper program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Mapper.Context;

public class TransListMapper extends Mapper<LongWritable, Text, Text,LongWritable>
{
        public void map(LongWritable key, Text value, Context output)throws IOException, 
InterruptedException
        {
                String Line = value.toString();
                String BankData[] = Line.split(" ");
                String DateTime[] = BankData[3].split("-");
                String Date[] = DateTime[0].split("/");
                if(BankData[6].equalsIgnoreCase("success")){
output.write(new Text(BankData[4].toString()), new LongWritable(Integer.parseInt(BankData[5])));
```

Reducer program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;

public class TransListReducer extends Reducer<Text, LongWritable, Text, Text>
{
        public void reduce(Text key, Iterable<LongWritable> value, Context output) throws
IOException, InterruptedException
  {
                int sum=0;
                int count=0;
      for (LongWritable val:value){
        sum+=val.get();
        count++;
      }
      output.write(key, new Text(count+"  "+sum));
  }
}
```

b) Account wise how many times the card has been used in the current month.

Driver program

```
package com.TransList;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import java.io.IOException;

public class TransListDriver{
 public static void main(String[] args) throws IOException,ClassNotFoundException,
InterruptedException {
      Job job=new Job(new Configuration());
     job.setJobName("Transaction List");
     job.setJarByClass(TransListDriver.class);
     job.setInputFormatClass(TextInputFormat.class);
     job.setMapOutputKeyClass(Text.class);
     job.setMapOutputValueClass(LongWritable.class);
     job.setOutputKeyClass(Text.class);
     job.setOutputValueClass(Text.class);
```

```
                job.setMapperClass(TransListMapper.class);
                job.setReducerClass(TransListReducer.class);
                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new Path(args[1]));
                job.waitForCompletion(true);
            }
        }
```

Mapper program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Mapper.Context;

public class TransListMapper extends Mapper<LongWritable, Text, Text,LongWritable>
{
        public void map(LongWritable key, Text value, Context output)throws IOException,
InterruptedException
        {
                String Line = value.toString();
                String BankData[] = Line.split(" ");
                String DateTime[] = BankData[3].split("-");
                String Date[] = DateTime[0].split("/");
                if(BankData[6].equalsIgnoreCase("success")){
output.write(new Text(BankData[1].toString()), new LongWritable(Integer.parseInt(BankData[5])));
```

Reducer program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;

public class TransListReducer extends Reducer<Text, LongWritable, Text, Text>
{
        public void reduce(Text key, Iterable<LongWritable> value, Context output) throws
IOException, InterruptedException
   {
```

```
                int count=0;
                int sum =0;
        for (LongWritable val:value){
                count++;
                  sum += val.get();
        }
        output.write(key, new Text(count+ " " + sum));
    }
}
```

c) Bankwise failed transactions list with detail

Driver program

```
package com.TransList;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import java.io.IOException;

public class TransListDriver{
 public static void main(String[] args) throws IOException,ClassNotFoundException,
InterruptedException {
        Job job=new Job(new Configuration());
        job.setJobName("Transaction List");
        job.setJarByClass(TransListDriver.class);
        job.setInputFormatClass(TextInputFormat.class);
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(LongWritable.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);
        job.setMapperClass(TransListMapper.class);
        job.setReducerClass(TransListReducer.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
```

```
                FileOutputFormat.setOutputPath(job, new Path(args[1]));
                job.waitForCompletion(true);
             }
          }
```

Mapper program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Mapper.Context;

public class TransListMapper extends Mapper<LongWritable, Text, Text,LongWritable>
{
          public void map(LongWritable key, Text value, Context output)throws IOException,
InterruptedException
          {
                    String Line = value.toString();
                    String BankData[] = Line.split(" ");
                    String DateTime[] = BankData[3].split("-");
                    String Date[] = DateTime[0].split("/");
                    if(BankData[6].equalsIgnoreCase("error")){
output.write(new Text(BankData[4].toString()), new LongWritable(Integer.parseInt(BankData[5])));
```

Reducer program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;

public class TransListReducer extends Reducer<Text, LongWritable, Text, Text>
{
          public void reduce(Text key, Iterable<LongWritable> value, Context output) throws
IOException, InterruptedException
   {
                    int sum=0;
                    int count=0;
      for (LongWritable val:value){
          sum+=val.get();
          count++;
      }
      output.write(key, new Text(count+"  "+sum));
   }
}
```

d).Maximum transaction for each bank for last month

e)Minimum transaction for each bank for last month

<u>Driver program</u>

```
package com.MaxMinTrans;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import java.io.IOException;

public class MaxMinTransDriver{
 public static void main(String[] args) throws IOException,ClassNotFoundException,
InterruptedException {
       Job job=new Job(new Configuration());
       job.setJobName("Max Min Transaction");
       job.setJarByClass(MaxMinTransDriver.class);
       job.setInputFormatClass(TextInputFormat.class);
       job.setMapOutputKeyClass(Text.class);
       job.setMapOutputValueClass(LongWritable.class);
       job.setOutputKeyClass(Text.class);
       job.setOutputValueClass(LongWritable.class);
       job.setMapperClass(MaxMinTransMapper.class);
       job.setReducerClass(MaxMinTransReducer.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
       FileOutputFormat.setOutputPath(job, new Path(args[1]));
       job.waitForCompletion(true);
   }
}
```

Mapper program:

```
package com.MaxMinTrans;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Mapper.Context;

public class MaxMinTransMapper extends Mapper<LongWritable, Text, Text,LongWritable>
{
	public void map(LongWritable key, Text value, Context output)throws IOException,
InterruptedException
	{
		String Line = value.toString();
		String BankData[] = Line.split(" ");
		String DateTime[] = BankData[3].split("-");
		String Date[] = DateTime[0].split("/");
		if(BankData[6].equalsIgnoreCase("success")){
output.write(new Text(BankData[4].toString()), new LongWritable(Integer.parseInt(BankData[5])));
		}
	}
}
```

Reducer program

```
package com.MaxMinTrans;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;

public class MaxMinTransReducer extends Reducer<Text, LongWritable, Text, LongWritable>
{
	public void reduce (Text key, Iterable<LongWritable> value, Context output) throws
IOException, InterruptedException
   {
		int max=0;
		int min=0;
		boolean flag = true;
```

```
                for(LongWritable val:value){
                        if(val.get()>max){
                                if(flag==true){
                                        if(val.get()>=min){
                                                min = (int)val.get();
                                                flag=false;
                                        }
                                }
                                max=(int)val.get();
                        }
                        else if(val.get()<min){
                                min = (int)val.get();
                        }
                }
                if(max==min) min=0;
     output.write(new Text("Minimum Transaction of "+key), new LongWritable(min));
     output.write(new Text("Maximum Transaction of "+key), new LongWritable(max));
  }
}
```

f)Average transaction for each bank for last month

Driver program

```
package com.TransList;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import java.io.IOException;

public class TransListDriver{
 public static void main(String[] args) throws IOException,ClassNotFoundException,
InterruptedException {
      Job job=new Job(new Configuration());
      job.setJobName("Transaction List");
      job.setJarByClass(TransListDriver.class);
```

```
                job.setInputFormatClass(TextInputFormat.class);
                job.setMapOutputKeyClass(Text.class);
                job.setMapOutputValueClass(LongWritable.class);
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(Text.class);
                job.setMapperClass(TransListMapper.class);
                job.setReducerClass(TransListReducer.class);
                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new Path(args[1]));
                job.waitForCompletion(true);
             }
          }
```

Mapper program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Mapper.Context;

public class TransListMapper extends Mapper<LongWritable, Text, Text,LongWritable>
{
          public void map(LongWritable key, Text value, Context output)throws IOException,
InterruptedException
          {
                    String Line = value.toString();
                    String BankData[] = Line.split(" ");
                    String DateTime[] = BankData[3].split("-");
                    String Date[] = DateTime[0].split("/");
                    if(BankData[6].equalsIgnoreCase("success")){
output.write(new Text(BankData[4].toString()), new LongWritable(Integer.parseInt(BankData[5])));
```

Reducer program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;
```

```
public class TransListReducer extends Reducer<Text, LongWritable, Text, Text>
{
        public void reduce(Text key, Iterable<LongWritable> value, Context output) throws
IOException, InterruptedException
  {
                int sum=0;
                int count=0;
                int average = 0;
      for (LongWritable val:value){
        sum+=val.get();
        count++;
      }
      average = sum / count;
      count =  Math.ceil(count/2);
      output.write(key, new Text(count+"  "+average));
  }
}
```

g)in last month, who used this card more than 2 times.

<u>Driver program</u>

```
package com.TransList;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import java.io.IOException;

public class TransListDriver{
 public static void main(String[] args) throws IOException,ClassNotFoundException,
InterruptedException {
      Job job=new Job(new Configuration());
      job.setJobName("Transaction List");
      job.setJarByClass(TransListDriver.class);
```

```
        job.setInputFormatClass(TextInputFormat.class);
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(LongWritable.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);
        job.setMapperClass(TransListMapper.class);
        job.setReducerClass(TransListReducer.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        job.waitForCompletion(true);
      }
    }
```

Mapper program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Mapper.Context;

public class TransListMapper extends Mapper<LongWritable, Text, Text,LongWritable>
{
    public void map(LongWritable key, Text value, Context output)throws IOException,
InterruptedException
    {
        String Line = value.toString();
        String BankData[] = Line.split(" ");
        String DateTime[] = BankData[3].split("-");
        String Date[] = DateTime[0].split("/");
        if(BankData[6].equalsIgnoreCase("success")){
output.write(new Text(BankData[4].toString()), new LongWritable(Integer.parseInt(BankData[5])));
```

Reducer program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;
```

```
public class TransListReducer extends Reducer<Text, LongWritable, Text, Text>
{
        public void reduce(Text key, Iterable<LongWritable> value, Context output) throws
IOException, InterruptedException
   {
                int sum=0;
                int count=0;
      for (LongWritable val:value){
         sum+=val.get();
         count++;
      }
      If (count >= 2)
      output.write(key, new Text(count+"  "+sum));
   }
}
```

h) All the banks card member details required except ICICI Bank.

<u>Driver program</u>

```
package com.TransList;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import java.io.IOException;

public class TransListDriver{
 public static void main(String[] args) throws IOException,ClassNotFoundException,
InterruptedException {
       Job job=new Job(new Configuration());
      job.setJobName("Transaction List");
      job.setJarByClass(TransListDriver.class);
```

```
                job.setInputFormatClass(TextInputFormat.class);
                job.setMapOutputKeyClass(Text.class);
                job.setMapOutputValueClass(LongWritable.class);
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(Text.class);
                job.setMapperClass(TransListMapper.class);
                job.setReducerClass(TransListReducer.class);
                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new Path(args[1]));
                job.waitForCompletion(true);
            }
        }
```

Mapper program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Mapper.Context;

public class TransListMapper extends Mapper<LongWritable, Text, Text,LongWritable>
{
        public void map(LongWritable key, Text value, Context output)throws IOException,
InterruptedException
        {
                String Line = value.toString();
                String BankData[] = Line.split(" ");
                String DateTime[] = BankData[3].split("-");
                String Date[] = DateTime[0].split("/");
If(BankData[6].equalsIgnoreCase("success") && !(BankData[4].equalsIgnoreCase("ICICI")
output.write(new Text(BankData.toString()), new LongWritable(Integer.parseInt(BankData[5])));
}}
```

Reducer program

```
package com.TransList;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;
```

```
public class TransListReducer extends Reducer<Text, LongWritable, Text, Text>
{
        public void reduce(Text key, Iterable<LongWritable> value, Context output) throws
IOException, InterruptedException
  {
                int sum=0;
                int count=0;
      for (LongWritable val:value){
        sum+=val.get();
        count++;
      }
      output.write(key, new Text(count+"  "+sum));
  }
}
```

INPUT DATA:

```
1 10101 james 01/01/2020-10:10 HDFC 45000 success
2 10201 hari 01/02/2020-9:10 SBI 6000 error
3 20456 dwaraka 01/03/2020-19:10 SBI 6000 success
4 45678 venkat 01/04/2020-21:10 ICIC 400 success
5 10201 hari 01/05/2020-9:13 SBI 6000 error
6 10209 hari 01/06/2020-9:16 syndicate 5000 error
7 45678 venkat 01/07/2020-21:10 ICIC 800 success
```

OUTPUT:

a)transactions list-- bank wise (bank name, times using card, amount with drwan)

output

```
HDFC 1 45000
SBI 1 6000
ICICI 2 1200
```

b)one bank, account wise howmany times card used in current month
For example SBI list Account num, howmany times card used

output

```
30101 3
40104 4
```

c)failed transactions list

output

```
SBI 10201 12000
syndicate 10209 5000
```

failed transactions list with detail

output

```
SBI 10201 6000 01/02/2020-9:10
SBI 10201 6000 01/02/2014-9:13
syndicate 10209 5000 01/02/2014-9:16
```

d)Maximum transaction for each bank for last month

e)Minimum transaction for each bank for last month

OUTPUT:

```
Maximum HDFC 45000
Minimum HDFC 0

Maximum SBI 6000
Minimum SBI 0

Maximum ICIC 800
Minimum ICIC 400
```

f)Average transaction for each bank for last month

```
HDFC 1 45000
SBI 1 6000
ICICI 2 1200  ----> ICICI   1 600
```

g)in last month, who used this card more than 2 times.

```
ICICI 2 1200
```

g)All the banks card member details required except ICICI Bank

```
1 10101 james 01/01/2020-10:10 HDFC 45000 success
2 20456 dwaraka 01/03/2020-19:10 SBI 6000 success
3 45678 venkat 01/04/2020-21:10 ICIC 400 success
4 10209 hari 01/06/2020-9:16 syndicate 5000 error
```

VIVA QUESTIONS :

1. What is Apache Pig and why do we need it?

2. What is HBase used for?

3. What is Apache Hive used for?

4. **Compare HBase with Hive?**

5. **What is the difference between Pig, Hive and HBase?**

6. **What is the use of ZooKeeper?**

9. **Big Data Case Study (any one)**
    1. **Healthcare Data**
    2. **Web Click stream Data**
    3. **Social Media Data [RSS, Tweets]**
    4. **Educational Data**

OUTPUT

OUTPUT

R-Programming

10. Implement Linear and logistic Regression with sample dataset using R-Programming

10.a) Linear Regression: The general mathematical equation for a linear regression is y = ax + b

Following is the description of the parameters used −

- **y** is the response variable.
- **x** is the predictor variable.
- **a** and **b** are constants which are called the coefficients.

Steps to Establish a Regression

A simple example of regression is predicting weight of a person when his height is known. To do this we need to have the relationship between height and weight of a person.

The steps to create the relationship is −

- Carry out the experiment of gathering a sample of observed values of height and corresponding weight.
- Create a relationship model using the **lm()** functions in R.
- Find the coefficients from the model created and create the mathematical equation
- Get a summary of the relationship model to know the average error in prediction. Also called **residuals**.
- To predict the weight of new persons, use the **predict()** function in R.

Sample Input Data

Below is the sample data representing the observations −

```
# Values of height
151, 174, 138, 186, 128, 136, 179, 163, 152, 131

# Values of weight.
63, 81, 56, 91, 47, 57, 76, 72, 62, 48
```

lm() Function

This function creates the relationship model between the predictor and the response variable.

Syntax

The basic syntax for **lm()** function in linear regression is − **lm(formula,data)**
Following is the description of the parameters used −

- **formula** is a symbol presenting the relation between x and y.
- **data** is the vector on which the formula will be applied.

Create Relationship Model & get the Coefficients

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.
relation <- lm(y~x)
print(relation)
```

OUTPUT:

```
Call:lm(formula = y ~ x)

Coefficients:
(Intercept)            x
  -38.4551      0.6746
```

```
#Get the Summary of the Relationship
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.
relation <- lm(y~x)
print(summary(relation))
```

OUTPUT:

```
Call: lm(formula = y ~ x)

Residuals:
    Min       1Q     Median       3Q      Max
-6.3002    -1.6629  0.0412    1.8944  3.9775

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -38.45509    8.04901  -4.778  0.00139 **
x             0.67461    0.05191  12.997 1.16e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.253 on 8 degrees of freedom
Multiple R-squared:  0.9548,    Adjusted R-squared:  0.9491
F-statistic: 168.9 on 1 and 8 DF,  p-value: 1.164e-06
```

predict() Function

Syntax

The basic syntax for predict() in linear regression is −**predict(object, newdata)**

Following is the description of the parameters used −

- **object** is the formula which is already created using the lm() function.
- **newdata** is the vector containing the new value for predictor variable.

Predict the weight of new persons

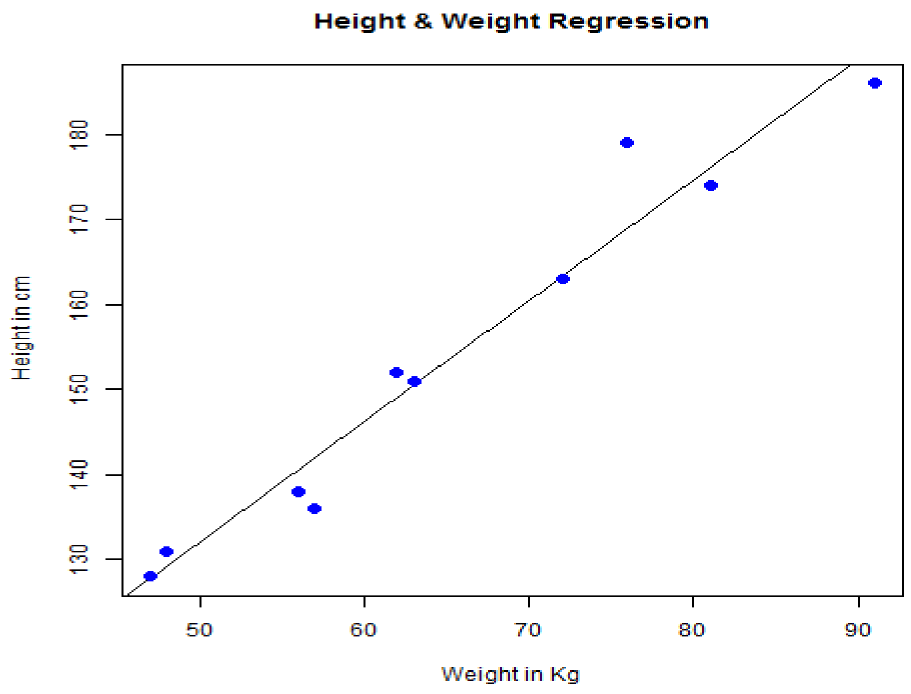
```
# The predictor vector.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
# The resposne vector.
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
# Apply the lm() function.
relation <- lm(y~x)
# Find weight of a person with height 170.
a <- data.frame(x = 170)
result <-  predict(relation,a)
print(result)
```

Output

```
       1    76.22869
```

Visualize the Regression Graphically

```
# Create the predictor and response variable.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
relation <- lm(y~x)
# Give the chart file a name.
png(file = "linearregression.png")
# Plot the chart.
plot(y,x,col = "blue",main = "Height & Weight Regression",
abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in
cm")
# Save the file.
dev.off()
```



OUTPUT

10.b) Logistic Regression : The Logistic Regression is a regression model in which the response variable (dependent variable) has categorical values such as True/False or 0/1. It actually measures the probability of a binary response as the value of response variable based on the mathematical equation relating it with the predictor variables.

The general mathematical equation for logistic regression is −

```
y = 1/(1+e^-(a+b1x1+b2x2+b3x3+...))
```

Following is the description of the parameters used −

- **y** is the response variable.
- **x** is the predictor variable.
- **a** and **b** are the coefficients which are numeric constants.

Syntax

The function used to create the regression model is the **glm()** function.
The basic syntax for **glm()** function in logistic regression is −**glm(formula,data,family)**
Following is the description of the parameters used −

- **formula** is the symbol presenting the relationship between the variables.
- **data** is the data set giving the values of these variables.
- **family** is R object to specify the details of the model. It's value is binomial for logistic regression.

Input data

The in-built data set "mtcars" describes different models of a car with their various engine specifications. In "mtcars" data set, the transmission mode (automatic or manual) is described by the column am which is a binary value (0 or 1). We can create a logistic regression model between the columns "am" and 3 other columns - hp, wt and cyl.

```
# Select some columns form mtcars.
input <- mtcars[,c("am","cyl","hp","wt")]
print(head(input))
```

OUTPUT:

```
                  am   cyl  hp    wt
Mazda RX4          1   6    110   2.620
Mazda RX4 Wag      1   6    110   2.875
Datsun 710         1   4     93   2.320
Hornet 4 Drive     0   6    110   3.215
Hornet Sportabout  0   8    175   3.440
Valiant            0   6    105   3.460
```

Program: To Create Regression Model

We use the **glm()** function to create the regression model and get its summary for analysis.

```
input <- mtcars[,c("am","cyl","hp","wt")]
am.data = glm(formula = am ~ cyl + hp + wt, data = input, family =
binomial)
print(summary(am.data))
```

Output

```
Call:
glm(formula = am ~ cyl + hp + wt, family = binomial,
data = input)

Deviance Residuals:
     Min         1Q      Median        3Q       Max
-2.17272     -0.14907  -0.01464     0.14116   1.27641

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 19.70288    8.11637   2.428   0.0152 *
cyl          0.48760    1.07162   0.455   0.6491
hp           0.03259    0.01886   1.728   0.0840 .
wt          -9.14947    4.15332  -2.203   0.0276 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1

(Dispersion parameter for binomial family taken to be
1)

    Null deviance: 43.2297  on 31  degrees of freedom
Residual deviance:  9.8415  on 28  degrees of freedom
AIC: 17.841
```

Conclusion

In the summary as the p-value in the last column is more than 0.05 for the variables "cyl" and "hp", we consider them to be insignificant in contributing to the value of the variable "am". Only weight (wt) impacts the "am" value in this regression model.

OUTPUT

Viva Questions

1. What is R?

2. What are some advantages of R?

3. What are the disadvantages of R?

4. What is Linear Regression?

5. What is Logistic Regresstion?

6. What are the similarities and differences between R and Python?

11. Implement Decision tree classification with Sample dataset and visualize graphically Using R Programming.

Decision tree : Decision tree is a graph to represent choices and their results in form of a tree. The nodes in the graph represent an event or choice and the edges of the graph represent the decision rules or conditions. It is mostly used in Machine Learning and Data Mining applications using R.

Examples of use of decision tress is –

- predicting an email as spam or not spam,
- predicting of a tumor is cancerous or predicting a loan as a good or bad credit risk based on the factors in each of these.

Steps to Create a Model

- Generally, a model is created with observed data also called training data.
- Then a set of validation data is used to verify and improve the model.
- R has packages which are used to create and visualize decision trees.
- For new set of predictor variable, we use this model to arrive at a decision on the category (yes/No, spam/not spam) of the data.

Install R Package

- The R package **"party"** is used to create decision trees.
- Use the below command in R console to install the package.
- `install.packages("party")`
- The package "party" has the function **ctree()** which is used to create and analyze decison tree.

Syntax

The basic syntax for creating a decision tree in R is − **ctree(formula, data)**
Following is the description of the parameters used −

- **formula** is a formula describing the predictor and response variables.
- **data** is the name of the data set used.

Input Data

We will use the R in-built data set named **readingSkills** to create a decision tree. It describes the score of someone's readingSkills if we know the variables "age","shoesize","score" and whether the person is a native speaker or not.

```
# Load the party package. It will automatically load other dependent
packages.
library(party)
# Print some records from data set readingSkills.
print(head(readingSkills))
```

OUTPUT:

```
  nativeSpeaker   age   shoeSize      score
1           yes     5   24.83189   32.29385
2           yes     6   25.95238   36.63105
3            no    11   30.42170   49.60593
4           yes     7   28.66450   40.28456
5           yes    11   31.88207   55.46085
6           yes    10   30.07843   52.83124
Loading required package: methods
Loading required package: grid
...............................
...............................
```

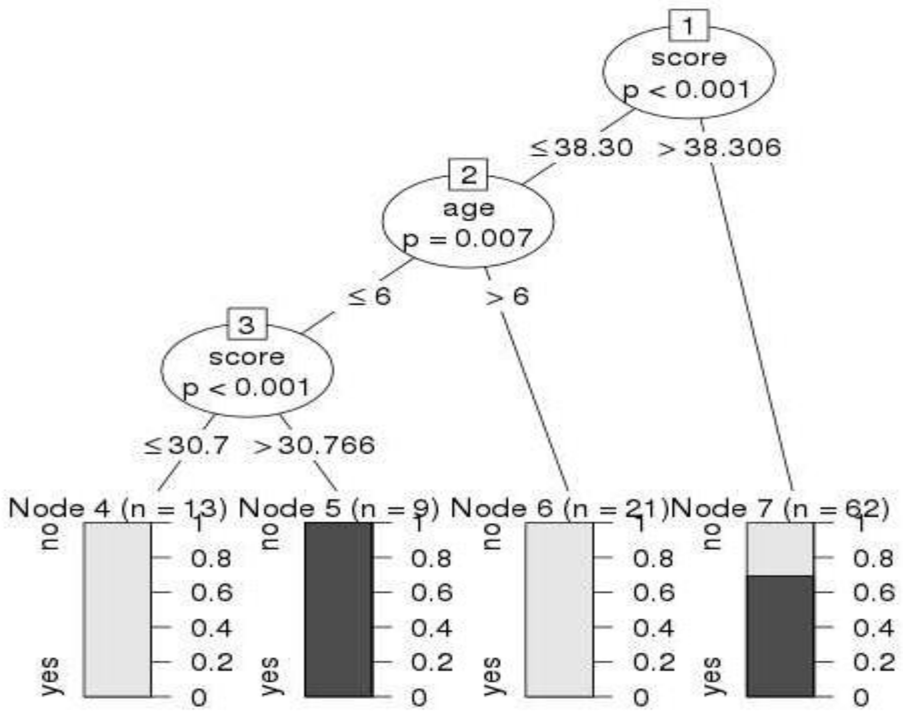
Program

We will use the **ctree()** function to create the decision tree and see its graph.

```
# Load the party package. It will automatically load other dependent
packages.
library(party)

# Create the input data frame.
input.dat <- readingSkills[c(1:105),]

# Give the chart file a name.
png(file = "decision_tree.png")

# Create the tree.
  output.tree <- ctree(
  nativeSpeaker ~ age + shoeSize + score,
  data = input.dat)

# Plot the tree.
plot(output.tree)

# Save the file.
dev.off()
```

OUTPUT:

```
null device
          1
Loading required package: methods
Loading required package: grid
Loading required package: mvtnorm
Loading required package: modeltools
Loading required package: stats4
Loading required package: strucchange
Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':
    as.Date, as.Date.numeric
```



Conclusion

From the decision tree shown above we can conclude that anyone whose readingSkills score is less than 38.3 and age is more than 6 is not a native Speaker.

OUTPUT

VIVA QUESTIONS

1. **How do you assign a variable in R?**

2. **What are the different data types/objects in R?**

3. **Write a custom function in R**

4. **What is classification?**

5. **What is Clustering?**

6. **Define Decision Tree**

12. Practical Implementation of Support vector machine(SVM) In R Programming

In machine learning, Support vector machine(SVM) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. It is mostly used in classification problems. In this algorithm, each data item is plotted as a point in n-dimensional space (where n is number of features), with the value of each feature being the value of a particular coordinate. Then, classification is performed by finding the hyper-plane that best differentiates the two classes.

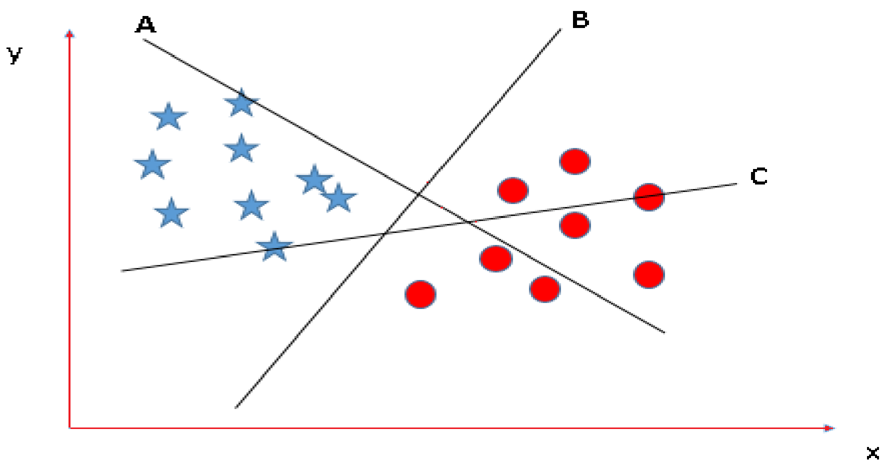
In addition to performing linear classification, SVMs can efficiently perform a non-linear classification, implicitly mapping their inputs into high-dimensional feature spaces.

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.

The most important question that arise while using SVM is how to decide right hyper plane. Consider the following scenarios:

Scenario 1:

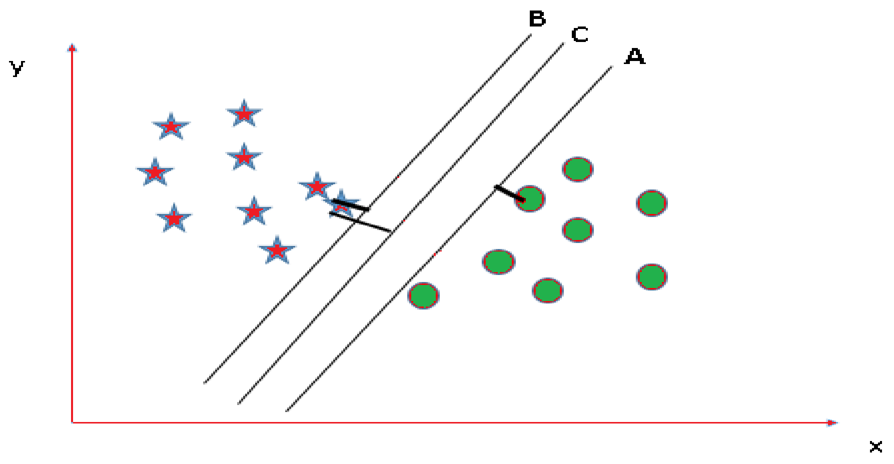
In this scenario there are three hyper planes called A,B,C. Now the problem is to identify the right hyper-plane which best differentiates the stars and the circles.



The thumb rule to be known, before finding the right hyper plane, to classify star and circle is that the hyper plane should be selected which segregate two classes better.In this case B classify star and circle better, hence it is right hyper plane.

Scenario 2:

Now take another Scenario where all three planes are segregating classes well. Now the question arises how to identify the right plane in this situation.



In such scenarios, calculate the margin which is the distance between nearest data point and hyper-plane. The plane having the maximum distance will be considered as the right hyper plane to classify the classes better.Here C is having the maximum margin and hence it will be considered as right hyper plane.

Importing the dataset

```
# Importing the dataset
dataset = read.csv('Social_Network_Ads.csv')
dataset = dataset[3:5]
```

#Selecting columns 3-5

#This is done for the ease of computation and implementation (to keep the example simple).

```
# Taking columns 3-5
dataset = dataset[3:5]
>describe(dataset)
# Encoding the target feature as factor
dataset$Purchased = factor(dataset$Purchased, levels = c(0, 1))
# Splitting the dataset into the Training set and Test set
install.packages('caTools')
library(caTools)
set.seed(123)
split = sample.split(dataset$Purchased, SplitRatio = 0.75)
>split
training_set = subset(dataset, split == TRUE)
> training_set
test_set = subset(dataset, split == FALSE)
> test_set
# Feature Scaling
training_set[-3] = scale(training_set[-3])
> training_set[-3
test_set[-3] = scale(test_set[-3])
> test_set[-3]
# Fitting SVM to the Training set
install.packages('e1071')
library(e1071)
classifier = svm(formula = Purchased ~ .,
                 data = training_set,
                 type = 'C-classification',
                 kernel = 'linear')
```

>describe(classifier)

#Classifier in nutshell

>classifier

```
# Predicting the Test set results
y_pred = predict(classifier, newdata = test_set[-3])
```

>y_pred

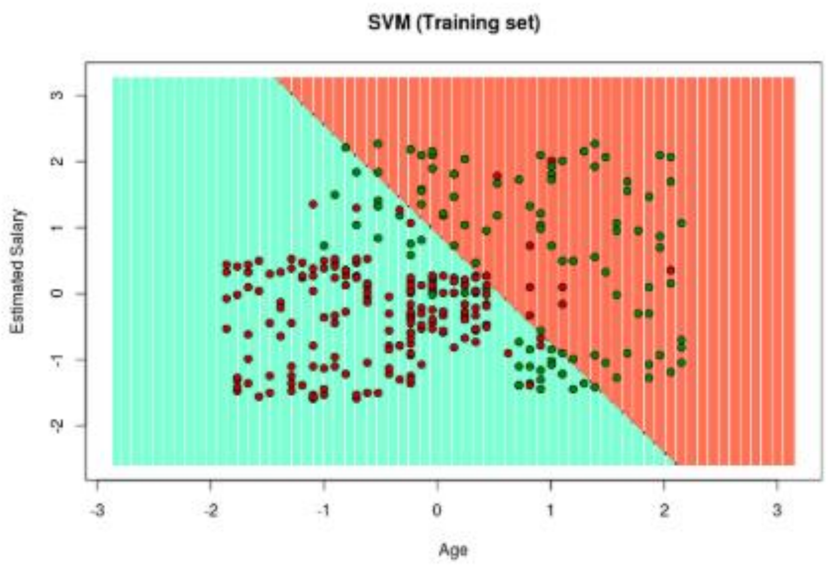
```
# Making the Confusion Matrix
cm = table(test_set[, 3], y_pred)
```

>cm

#Visualizing the Training set results

```
# installing library ElemStatLearn
library(ElemStatLearn)
# Plotting the training data set results
set = training_set
X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)
X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)
grid_set = expand.grid(X1, X2)
colnames(grid_set) = c('Age', 'EstimatedSalary')
y_grid = predict(classifier, newdata = grid_set)
plot(set[, -3],
     main = 'SVM (Training set)',
     xlab = 'Age', ylab = 'Estimated Salary',
     xlim = range(X1), ylim = range(X2))
contour(X1, X2, matrix(as.numeric(y_grid), length(X1), length(X2)), add =
TRUE)
points(grid_set, pch = '.', col = ifelse(y_grid == 1, 'coral1',
'aquamarine'))
points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))
```

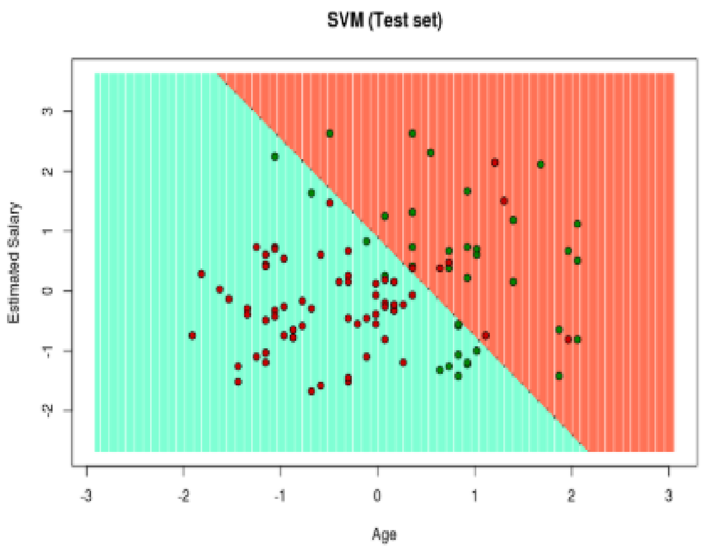
Output:



# Visualizing the Test set results

```
set = test_set
X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)
X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)
grid_set = expand.grid(X1, X2)
colnames(grid_set) = c('Age', 'EstimatedSalary')
y_grid = predict(classifier, newdata = grid_set)
plot(set[, -3], main = 'SVM (Test set)',
     xlab = 'Age', ylab = 'Estimated Salary',
     xlim = range(X1), ylim = range(X2))
contour(X1, X2, matrix(as.numeric(y_grid), length(X1), length(X2)), add =
TRUE)
points(grid_set, pch = '.', col = ifelse(y_grid == 1, 'coral1',
'aquamarine'))
points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))
```

Output:



Since in the result, a hyper-plane has been found in the Training set result and verified to be the best one in the Test set result. Hence, SVM has been successfully implemented in R.

OUTPUT

Viva Questions

1. **How do you import data in R?**

2. **How do you install a package in R?**

3. **What is the use of with() in R?**

4. **What is the use of by() in R?**

5. **When is it appropriate to use mode()?**

6. **What is SVM? Define Types of SVM models**

13. Practical Implementation of Naive Bayes In R Programming

Naive Bayes is a Supervised Machine Learning **algorithm** based on the **Bayes** Theorem that is used to solve **classification** problems by following a probabilistic approach. It is based on the idea that the predictor variables in a Machine Learning model are independent of each other.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Bayes Theorem – Naive Bayes In R – Edureka

In the above equation:

- P(A|B): Conditional probability of event A occurring, given the event B
- P(A): Probability of event A occurring
- P(B): Probability of event B occurring
- P(B|A): Conditional probability of event B occurring, given the event A

Problem Statement: To study a Diabetes data set and build a Machine Learning model that predicts whether or not a person has Diabetes.
Data Set Description: The given data set contains 100s of observations of patients along with their health details. Here's a list of the predictor variables that will help us classify a patient as either Diabetic or Normal:

- Pregnancies: Number of pregnancies so far
- Glucose: Plasma glucose concentration
- BloodPressure: Diastolic blood pressure (mm Hg)
- SkinThickness: Triceps skin fold thickness (mm)
- Insulin: 2-Hour serum insulin (mu U/ml)
- BMI: Body mass index (weight in kg/(height in m)^2)
- DiabetesPedigreeFunction: Diabetes pedigree function
- Age: Age (years)

The response variable or the output variable is:

- Outcome: Class variable (0 or 1)

Logic: To build a Naive Bayes model in order to classify patients as either Diabetic or normal by studying their medical records such as Glucose level, age, BMI, etc. I'll be using the R language in order to build the model.

Step 1: ***Install and load the requires packages***

#Loading required packages

```
install.packages('ggplot2')
library(ggplot2)
install.packages('mice')
library(mice)
install.packages('GGally')
library(GGally)
install.packages ('mice')
library('mice')
install.packages ('missmap')
library('missmap')
install.packages ('e1071')
library('e1071')
```

Step 2: ***Import the data set***

#Reading data into R

```
data<- read.csv("c:/ProgramFiles/R/dataset/diabetes.csv")
```

Before we study the data set let's convert the output variable ('Outcome') into a categorical variable. This is necessary because our output will be in the form of 2 classes, True or False. Where true will denote that a patient has diabetes and false denotes that a person is diabetes free.

#Setting outcome variables as categorical

```
data$Outcome <- factor(data$Outcome, levels = c(0,1), labels = c("False", "True"))
```

Step 3: Studying the Data Set

```
#Studying the structure of the data
 str(data)
#Display the first 6 records
head(data)
#Display the structure of the dataset
describe(data)
```

Step 4: Data Cleaning

While analyzing the structure of the data set, we can see that the minimum values for Glucose, Bloodpressure, Skinthickness, Insulin, and BMI are all zero. This is not ideal since no one can have a value of zero for Glucose, blood pressure, etc. Therefore, such values are treated as missing observations.

```
#Convert '0' values into NA
data[, 2:7][data[, 2:7] == 0] <- NA
```

To check how many missing values we have now, let’s visualize the data:

```
#visualize the missing data
missmap(data)
```

The above illustrations show that our data set has plenty missing values and removing all of them will leave us with an even smaller data set, therefore, **we can perform imputations by using the *mice* package in R.**

```
#Use mice package to predict missing values
mice_mod <- mice(data[ , c("Glucose","BloodPressure","SkinThickness","Insulin","BMI")],
method='rf')
mice_complete <- complete(mice_mod)

> mice_complete
```

output

```
iter imp variable
1 1 Glucose BloodPressure SkinThickness Insulin BMI
1 2 Glucose BloodPressure SkinThickness Insulin BMI
1 3 Glucose BloodPressure SkinThickness Insulin BMI
1 4 Glucose BloodPressure SkinThickness Insulin BMI
1 5 Glucose BloodPressure SkinThickness Insulin BMI
2 1 Glucose BloodPressure SkinThickness Insulin BMI
2 2 Glucose BloodPressure SkinThickness Insulin BMI
2 3 Glucose BloodPressure SkinThickness Insulin BMI
2 4 Glucose BloodPressure SkinThickness Insulin BMI
2 5 Glucose BloodPressure SkinThickness Insulin BMI
```

#Transfer the predicted missing values into the main data set

```
data$Glucose <- mice_complete$Glucose
data$BloodPressure <- mice_complete$BloodPressure
data$SkinThickness <- mice_complete$SkinThickness
data$Insulin<- mice_complete$Insulin
data$BMI <- mice_complete$BMI
```

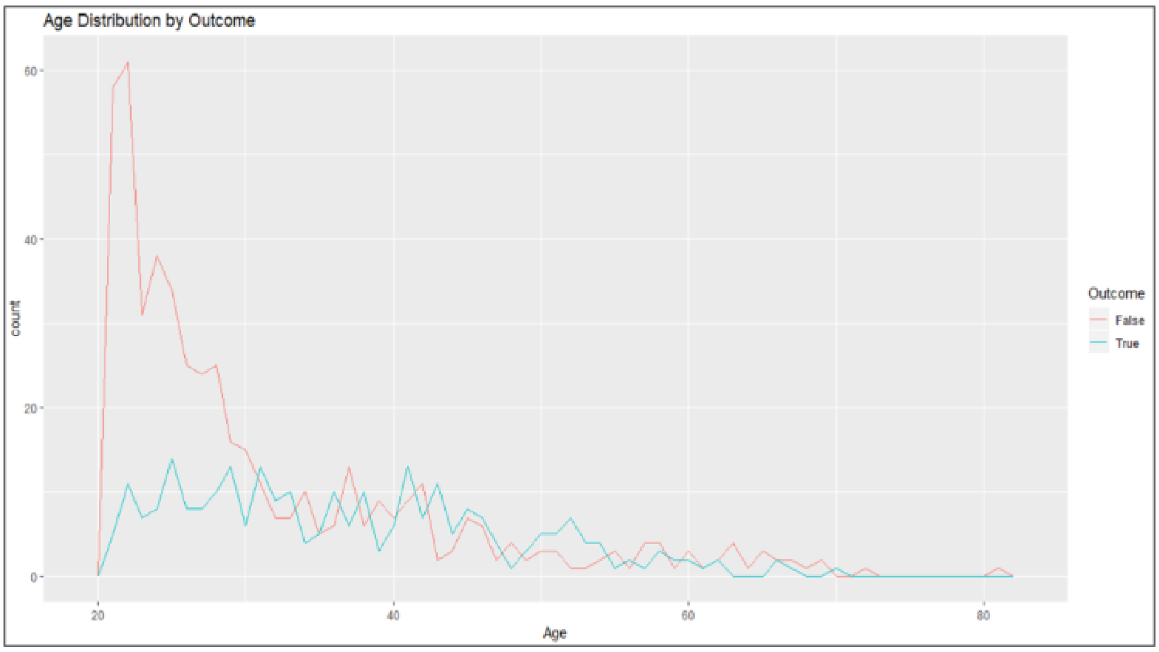
To check if there are still any missing values, let’s use the missmap plot:

```
missmap(data)
```

The output looks good, there is no missing data.

Step 5: *Exploratory Data Analysis*
Now let’s perform a couple of visualizations to take a better look at each variable, this stage is essential to understand the significance of each predictor variable.

```
#Data Visualization
#Visual 1
ggplot(data, aes(Age, colour = Outcome)) +
geom_freqpoly(binwidth = 1) + labs(title="Age Distribution by Outcome")
```



Data Visualization – Naive Bayes In R

#visual 2
c <- ggplot(data, aes(x=Pregnancies, fill=Outcome, color=Outcome)) +
geom_histogram(binwidth = 1) + labs(title="Pregnancy Distribution by Outcome")
c + theme_bw()

#visual 3
P <- ggplot(data, aes(x=BMI, fill=Outcome, color=Outcome)) +
geom_histogram(binwidth = 1) + labs(title="BMI Distribution by Outcome")
P + theme_bw()

#visual 4
ggplot(data, aes(Glucose, colour = Outcome)) +
geom_freqpoly(binwidth = 1) + labs(title="Glucose Distribution by Outcome")
#visual 5
ggpairs(data)

Step 6: *Data Modelling*
This stage begins with a process called Data Splicing, wherein the data set is split into two parts:
Training set: This part of the data set is used to build and train the Machine Learning model.
Testing set: This part of the data set is used to evaluate the efficiency of the model.

#Building a model
#split data into training and test data sets
indxTrain <- createDataPartition(y = data$Outcome,p = 0.75,list = FALSE)
training <- data[indxTrain,]
testing <- data[-indxTrain,]
#Check dimensions of the split
> prop.table(table(data$Outcome)) * 100

```
False True
65.10417 34.89583

> prop.table(table(training$Outcome)) * 100

False True
65.10417 34.89583

> prop.table(table(testing$Outcome)) * 100

False True
65.10417 34.89583
```

For comparing the outcome of the training and testing phase let's create separate variables that store the value of the response variable:

```
#create objects x which holds the predictor variables and y which holds the response variables
x = training[,-9]
y = training$Outcome
```

Now it's time to **load the e1071 package** that **holds the Naive Bayes function**. This is an in-built function provided by R.

```
library(e1071)
```

After loading the package, the below code snippet will create Naive Bayes model by using the training data set:

```
model = train(x,y,'nb',trControl=trainControl(method='cv',number=10))

> model
```

Output:

```
Naive Bayes
 576 samples
8 predictor
2 classes: 'False', 'True'
 No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 518, 518, 519, 518, 519, 518, ...
Resampling results across tuning parameters:
 usekernel Accuracy Kappa
FALSE 0.7413793 0.4224519
TRUE 0.7622505 0.4749285
 Tuning parameter 'fL' was held constant at a value of 0
Tuning parameter 'adjust' was held constant at a value of 1
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were fL = 0, usekernel = TRUE and adjust = 1.
```

We thus created a predictive model by using the Naive Bayes Classifier.

Step 7: ***Model Evaluation***

To check the efficiency of the model, we are now going to run the testing data set on the model, after which we will evaluate the accuracy of the model by using a Confusion matrix.

```
#Model Evaluation
#Predict testing set
Predict <- predict(model,newdata = testing )
#Get the confusion matrix to see accuracy value and other parameter values
 > confusionMatrix(Predict, testing$Outcome )
```

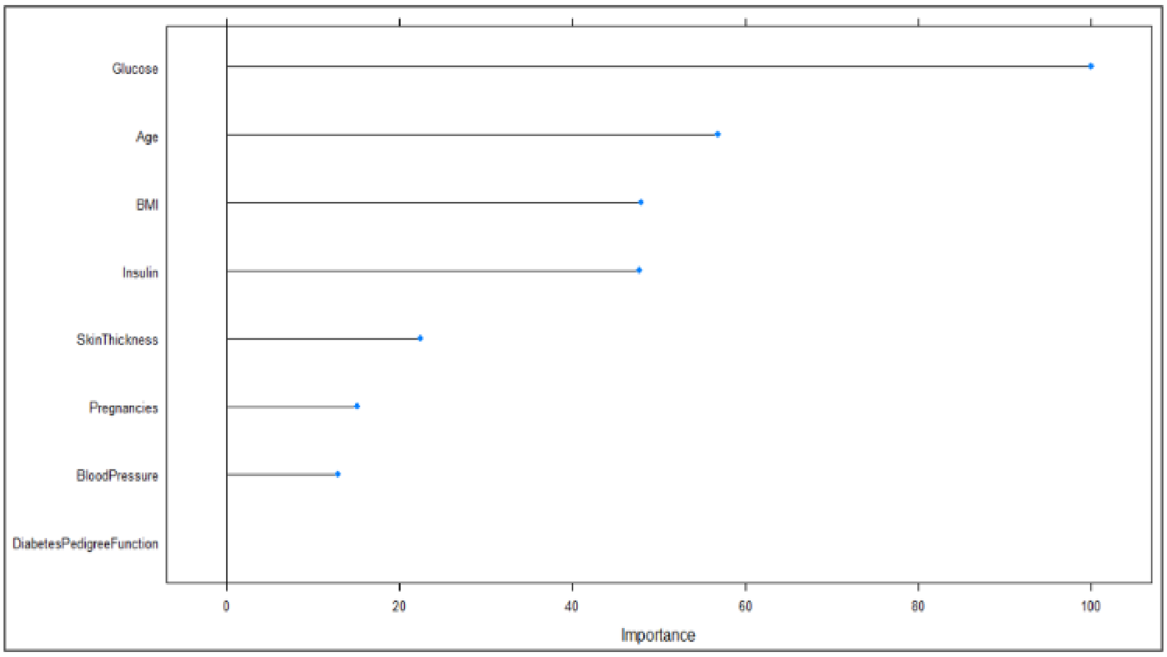
output

```
Confusion Matrix and Statistics
 Reference
Prediction False True
False 91 18
True 34 49
 Accuracy : 0.7292
95% CI : (0.6605, 0.7906)
No Information Rate : 0.651
P-Value [Acc > NIR] : 0.01287
 Kappa : 0.4352
Mcnemar's Test P-Value : 0.03751
 Sensitivity : 0.7280
Specificity : 0.7313
Pos Pred Value : 0.8349
Neg Pred Value : 0.5904
Prevalence : 0.6510
Detection Rate : 0.4740
Detection Prevalence : 0.5677
Balanced Accuracy : 0.7297
'Positive' Class : False
```

The final output shows that we built a Naive Bayes classifier that can predict whether a person is diabetic or not, with an accuracy of approximately 73%.

To summaries the demo, let’s draw a plot that shows how each predictor variable is independently responsible for predicting the outcome.

```
#Plot Variable performance
X <- varImp(model)
plot(X)
```



Variable Performance Plot – Naive Bayes In R

From the above illustration, it is clear that ‘Glucose’ is the most significant variable for predicting the outcome.

output

VIVA QUESTIONS

1. **How do you concatenate strings in R?**

2. **What are 3 sorting algorithms available in R?**

3. **Can you create an R decision tree?**

4. **Why is R useful for data science?**

5. **Describe how R can be used for predictive analysis**

6. **What is NaiveBase? How to create a NaïveBayes in R**

14. Practical Implementation of K-Means clustering in R programming

K Means Clustering in R Programming is an Unsupervised Non-linear algorithm that cluster data based on similarity or similar groups. It seeks to partition the observations into a pre-specified number of clusters. Segmentation of data takes place to assign each training example to a segment called a cluster. In the unsupervised algorithm, high reliance on raw data is given with large expenditure on manual review for review of relevance is given. It is used in a variety of fields like Banking, healthcare, retail, Media, etc.

Theory

K-Means clustering groups the data on similar groups. The algorithm is as follows:

1. Choose the number **K** clusters.
2. Select at random K points, the centroids(Not necessarily from the given data).
3. Assign each data point to closest centroid that forms K clusters.
4. Compute and place the new centroid of each centroid.
5. Reassign each data point to new cluster.

After final reassignment, name the cluster as Final cluster.

The Dataset:IRIS dataset

Iris dataset consists of 50 samples from each of 3 species of Iris(Iris setosa, Iris virginica, Iris versicolor) and a multivariate dataset introduced by British statistician and biologist Ronald Fisher in his 1936 paper The use of multiple measurements in taxonomic problems. Four features were measured from each sample i.e length and width of the sepals and petals and based on the combination of these four features, Fisher developed a linear discriminant model to distinguish the species from each other.

```
# Loading data
data(iris)

# Structure
str(iris)
```

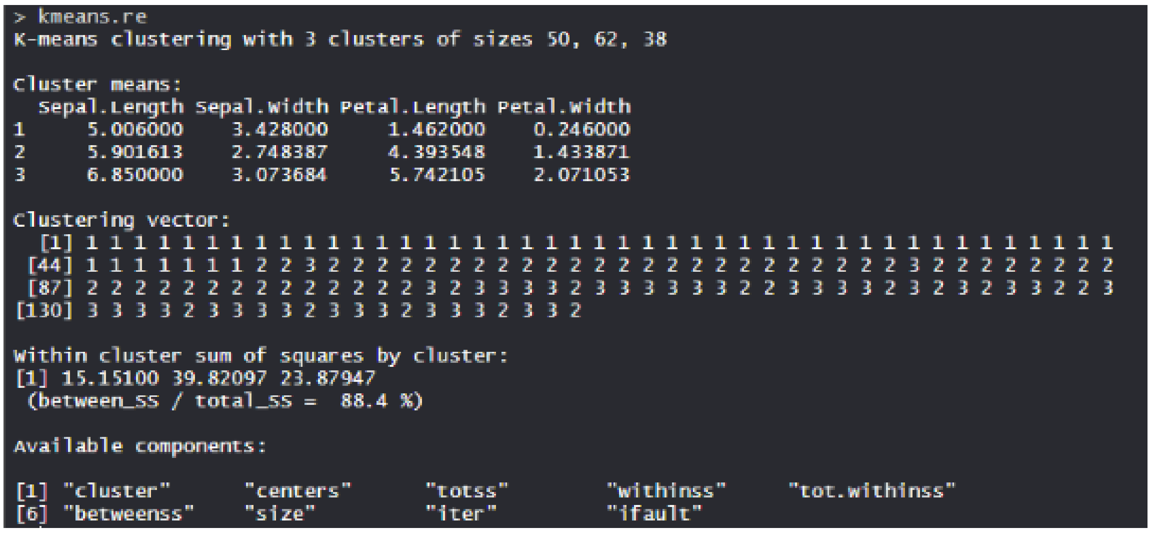
Performing K-Means Clustering on Dataset

Using K-Means Clustering algorithm on the dataset which includes 11 persons and 6 variables or attributes

```
# Installing Packages
install.packages("ClusterR")
install.packages("cluster")

# Loading package
library(ClusterR)
library(cluster)

# Removing initial label of
# Species from original dataset
```

```
iris_1 <- iris[, -5]

# Fitting K-Means clustering Model
# to training dataset
set.seed(240) # Setting seed
kmeans.re <- kmeans(iris_1, centers = 3, nstart = 20)
kmeans.re

# Cluster identification for
# each observation
kmeans.re$cluster

# Confusion Matrix
cm <- table(iris$Species, kmeans.re$cluster)
cm

# Model Evaluation and visualization
plot(iris_1[c("Sepal.Length", "Sepal.Width")])
plot(iris_1[c("Sepal.Length", "Sepal.Width")],
     col = kmeans.re$cluster)
plot(iris_1[c("Sepal.Length", "Sepal.Width")],
     col = kmeans.re$cluster,
     main = "K-means with 3 clusters")

## Plotiing cluster centers
kmeans.re$centers
kmeans.re$centers[, c("Sepal.Length", "Sepal.Width")]

# cex is font size, pch is symbol
points(kmeans.re$centers[, c("Sepal.Length", "Sepal.Width")],
       col = 1:3, pch = 8, cex = 3)

## Visualizing clusters
y_kmeans <- kmeans.re$cluster
clusplot(iris_1[, c("Sepal.Length", "Sepal.Width")],
         y_kmeans,
         lines = 0,
         shade = TRUE,
         color = TRUE,
         labels = 2,
         plotchar = FALSE,
         span = TRUE,
         main = paste("Cluster iris"),
         xlab = 'Sepal.Length',
         ylab = 'Sepal.Width')
```
Output:

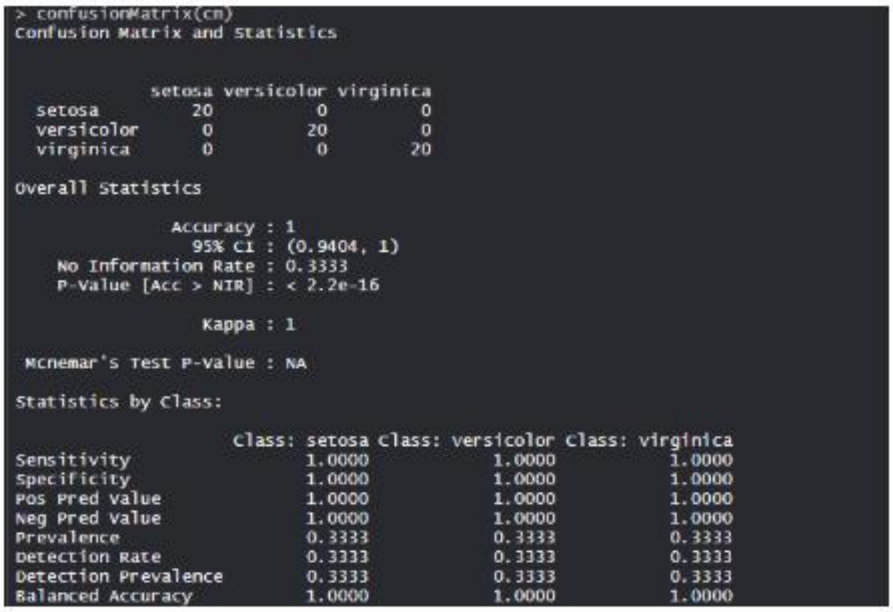
- **Model kmeans_re:**
  - The 3 clusters are made which are of 50, 62, and 38 sizes respectively. Within the cluster, the sum of squares is 88.4%.



```
> kmeans.re
K-means clustering with 3 clusters of sizes 50, 62, 38

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     5.006000    3.428000     1.462000    0.246000
2     5.901613    2.748387     4.393548    1.433871
3     6.850000    3.073684     5.742105    2.071053

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [44] 1 1 1 1 1 1 1 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2
 [87] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3 3 2 3 3 3 3 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 2 2 3
[130] 3 3 3 3 2 3 3 3 3 2 3 3 3 2 3 3 3 2 3 3 2

Within cluster sum of squares by cluster:
[1] 15.15100 39.82097 23.87947
 (between_SS / total_SS =  88.4 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

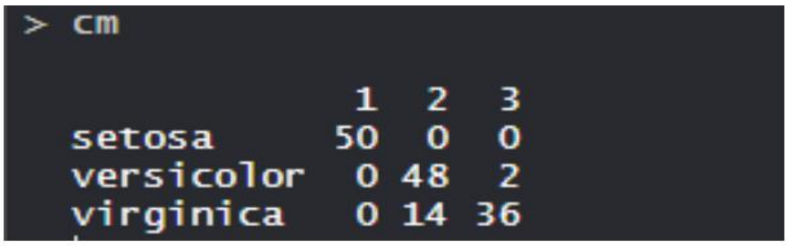
- The 3 clusters are made which are of 50, 62, and 38 sizes respectively. Within the cluster, the sum of squares is 88.4%.

Cluster identification:



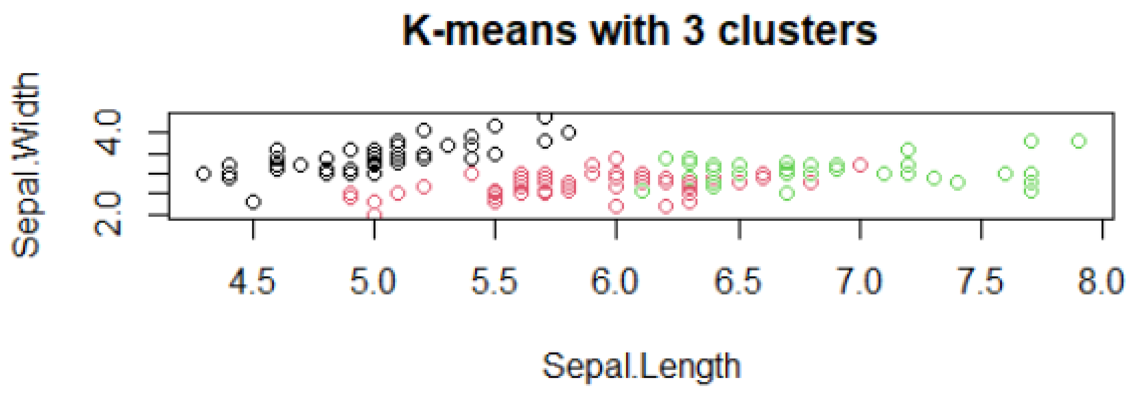
```
> confusionMatrix(cm)
Confusion Matrix and Statistics


            setosa versicolor virginica
  setosa        20          0         0
  versicolor     0         20         0
  virginica      0          0        20

Overall Statistics

               Accuracy : 1
                 95% CI : (0.9404, 1)
    No Information Rate : 0.3333
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 1

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: setosa Class: versicolor Class: virginica
Sensitivity                 1.0000            1.0000           1.0000
Specificity                 1.0000            1.0000           1.0000
Pos Pred Value              1.0000            1.0000           1.0000
Neg Pred Value              1.0000            1.0000           1.0000
Prevalence                  0.3333            0.3333           0.3333
Detection Rate              0.3333            0.3333           0.3333
Detection Prevalence        0.3333            0.3333           0.3333
Balanced Accuracy           1.0000            1.0000           1.0000
```

- The model achieved an accuracy of 100% with a p-value of less than 1. This indicates the model is good.
- **Confusion Matrix:**

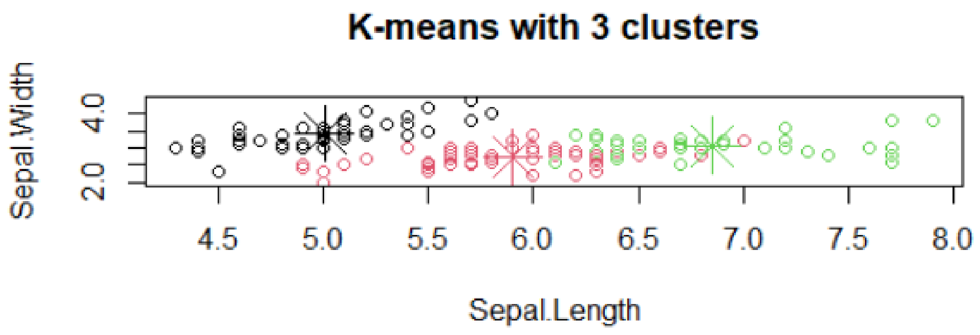


```
> cm

              1  2  3
  setosa     50  0  0
  versicolor  0 48  2
  virginica   0 14 36
```

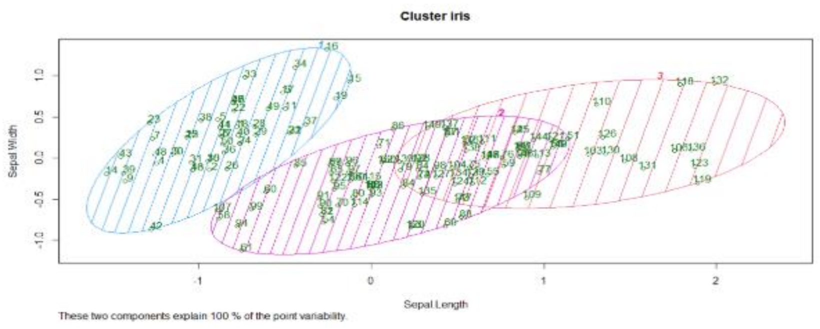
- So, 50 Setosa are correctly classified as Setosa. Out of 62 Versicolor, 48 Versicolor are correctly classified as Versicolor and 14 are classified as virginica. Out of 36 virginica, 19 virginica are correctly classified as virginica and 2 are classified as Versicolor.
- **K-means with 3 clusters plot:**



- The model showed 3 cluster plots with three different colors and with Sepal.length and with Sepal.width.
- **Plotting cluster centers:**



- In the plot, centers of clusters are marked with cross signs with the same color of the cluster.
- **Plot of clusters:**



- So, 3 clusters are formed with varying sepal length and sepal width. Hence, the K-Means clustering algorithm is widely used in the industry.

OUTPUT

VIVA QUESTIONS

1. **How R is used in logistic regression?**

2. **What are different ways to call a function in R?**

3. **What is lazy function evaluation in R?**

4. **What is reshaping of data in R?**

5. **What is clustering? How to define a K-means in R?**

6. **What is the difference between subset() function and sample() function in R?**

15. Visualize data using any plotting framework

Data Visualization Tools

1. **R-Pie Charts**
2. **R-Barcharts**
3. **R-BoxPlots**
4. **R-Histograms**
5. **R-Line Charts**
6. **R-Scatter Plots**

1.R-Pie Charts: R Programming language has numerous libraries to create charts and graphs. A pie-chart is a representation of values as slices of a circle with different colors. The slices are labelled and the numbers corresponding to each slice is also represented in the chart.

In R the pie chart is created using the **pie()** function which takes positive numbers as a vector input. The additional parameters are used to control labels, color, title etc.

Syntax

The basic syntax for creating a pie-chart using the R is:

```
pie(x, labels, radius, main, col, clockwise)
```

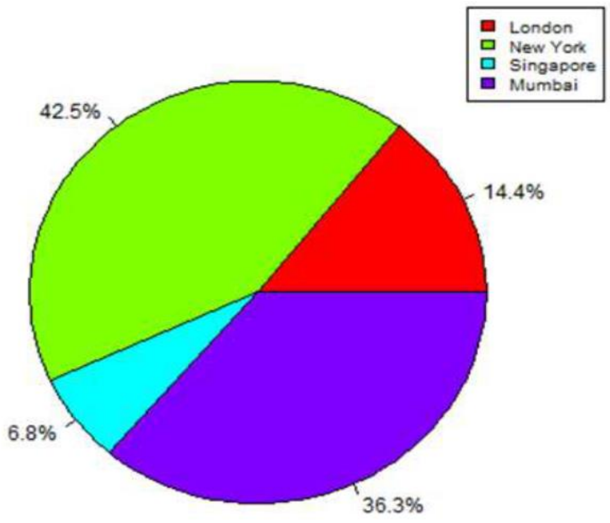
Following is the description of the parameters used:

- **x** is a vector containing the numeric values used in the pie chart.
- **labels** is used to give description to the slices.
- **radius** indicates the radius of the circle of the pie chart.(value between -1 and +1).
- **main** indicates the title of the chart.
- **col** indicates the color palette.
- **clockwise** is a logical value indicating if the slices are drawn clockwise or anti clockwise.

Example

A very simple pie-chart is created using just the input vector and labels. We can add slice percentage and a chart legend by creating additional chart variables.

```
# Create data for the graph.
x <- c(21, 62, 10,53)
labels <- c("London","New York","Singapore","Mumbai")
piepercent<- round(100*x/sum(x), 1)
# Give the chart file a name.
png(file = "city_percentage_legends.jpg")
# Plot the chart.
pie(x, labels=piepercent, main="City pie chart",col=rainbow(length(x)))
legend("topright", c("London","New York","Singapore","Mumbai"), cex=0.8,
fill=rainbow(length(x)))
# Save the file.
dev.off()
```



2.R-Barcharts:A bar chart represents data in rectangular bars with length of the bar proportional to the value of the variable. R uses the function **barplot()** to create bar charts. R can draw both vertical and horizontal bars in the bar chart. In bar chart each of the bars can be given different colors.

Syntax

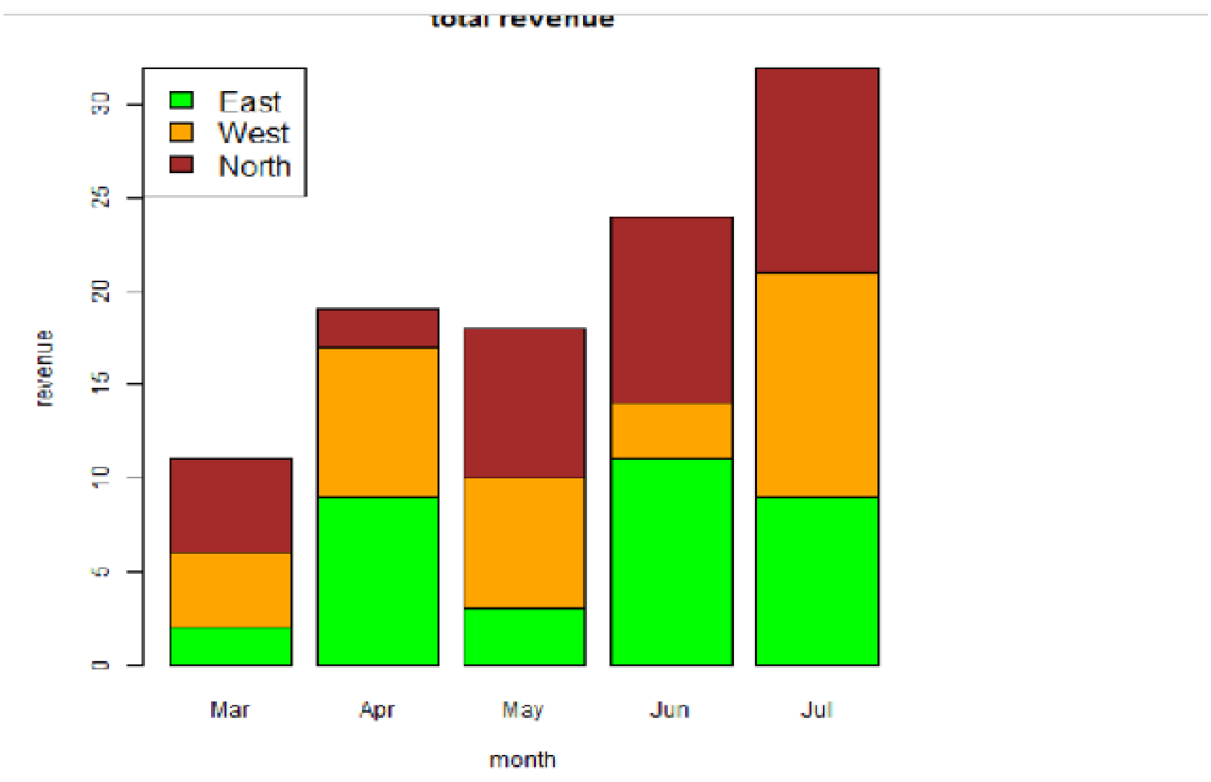
The basic syntax to create a bar-chart in R is:

```
barplot(H,xlab,ylab,main, names.arg,col)
```

Following is the description of the parameters used:

- **H** is a vector or matrix containing numeric values used in bar chart.
- **xlab** is the label for x axis.
- **ylab** is the label for y axis.
- **main** is the title of the bar chart.
- **names.arg** is a vector of names appearing under each bar.
- **col** is used to give colors to the bars in the graph.

```
# Create the input vectors.
colors <- c("green","orange","brown")
months <- c("Mar","Apr","May","Jun","Jul")
regions <- c("East","West","North")

# Create the matrix of the values.
Values <- matrix(c(2,9,3,11,9,4,8,7,3,12,5,2,8,10,11),nrow=3,ncol=5,byrow=TRUE)
# Give the chart file a name.
png(file = "barchart_stacked.png")
# Create the bar chart.
barplot(Values,main="total
revenue",names.arg=months,xlab="month",ylab="revenue",col=colors)
# Add the legend to the chart.
legend("topleft", regions, cex=1.3, fill=colors)
# Save the file.
dev.off()
```



3. Boxplots are a measure of how well distributed is the data in a data set. It divides the data set into three quartiles. This graph represents the minimum, maximum, median, first quartile and third quartile in the data set. It is also useful in comparing the distribution of data across data sets by drawing boxplots for each of them.
Boxplots are created in R by using the **boxplot()** function.

Syntax

The basic syntax to create a boxplot in R is :

```
boxplot(x,data,notch,varwidth,names,main)
```

Following is the description of the parameters used:

- **x** is a vector or a formula.
- **data** is the data frame.
- **notch** is a logical value. Set as TRUE to draw a notch.
- **varwidth** is a logical value. Set as true to draw width of the box proportionate to the sample size.
- **names** are the group labels which will be printed under each boxplot.
- **main** is used to give a title to the graph.

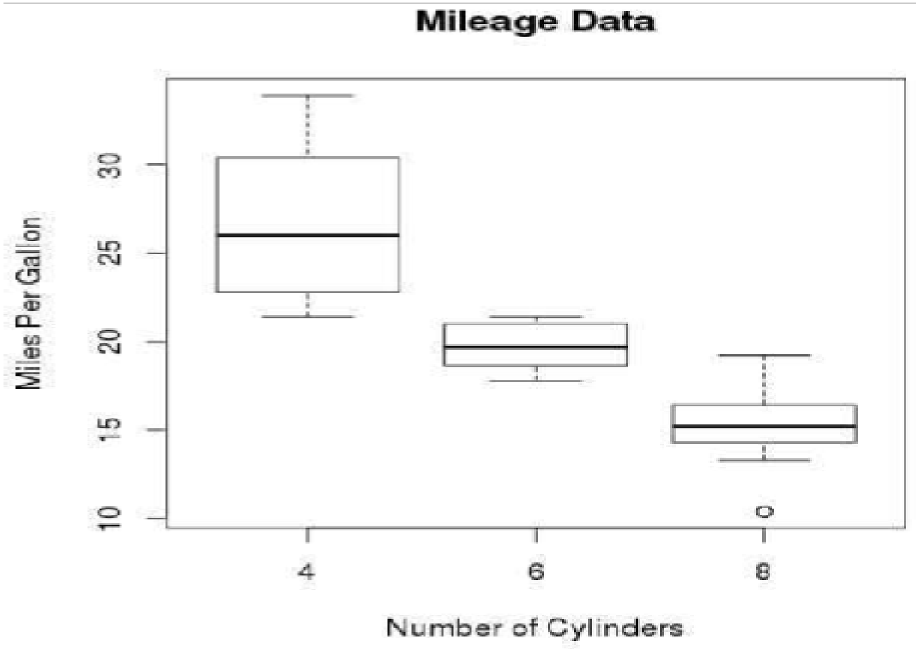
Example

We use the data set "mtcars" available in the R environment to create a basic boxplot.
Let's look at the columns "mpg" and "cyl" in mtcars.

```
input <- mtcars[,c('mpg','cyl')]
print(head(input))
```

The below script will create a boxplot graph for the relation between mpg(miles per gallon)
and cyl (number of cylinders).

```
# Give the chart file a name.
png(file = "boxplot.png")
# Plot the chart.
boxplot(mpg ~ cyl, data=mtcars,
xlab="Number of Cylinders",
ylab="Miles Per Gallon",
main="Mileage Data")
# Save the file.
dev.off()
```



4.R-Histograms: A histogram represents the frequencies of values of a variable bucketed into ranges.Histogram is similar to bar chat but the difference is it groups the values into continuous ranges. Each bar in histogram represents the height of the number of values present in that range.
R creates histogram using **hist**() function. This function takes a vector as an input and uses some more parameters to plot histograms.

Syntax

The basic syntax for creating a histogram using R is:

```
hist(v,main,xlab,xlim,ylim,breaks,col,border)
```

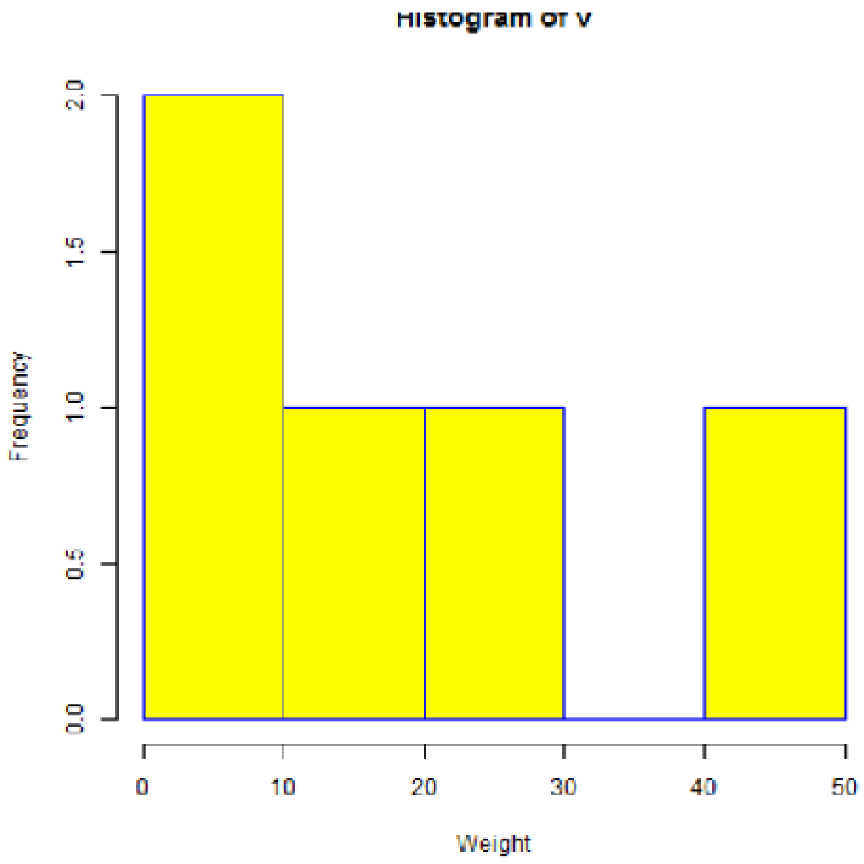
Following is the description of the parameters used:

- **v** is a vector containing numeric values used in histogram.
- **main** indicates title of the chart.
- **col** is used to set color of the bars.
- **border** is used to set border color of each bar.
- **xlab** is used to give description of x-axis.
- **xlim** is used to specify the range of values on the x-axis.
- **ylim** is used to specify the range of values on the y-axis.
- **breaks** is used to mention the width of each bar.

Example

A simple histogram is created using input vector, label, col and border parameters.
The script given below will create and save the histogram in the current R working directory.

```
# Create data for the graph.
v <- c(9,13,21,8,36,22,12,41,31,33,19)
# Give the chart file a name.
png(file = "histogram.png")
# Create the histogram.
hist(v,xlab="Weight",col="yellow",border="blue")

# Save the file.
dev.off()
```



5.R-Line Charts:
A line chart is a graph that connects a series of points by drawing line segments between them. These points are ordered in one of their coordinate (usually the x-coordinate) value.
Line charts are usually used in identifying the trends in data.
The **plot()** function in R is used to create the line graph.

Syntax

The basic syntax to create a line chart in R is:

```
plot(v,type,col,xlab,ylab)
```

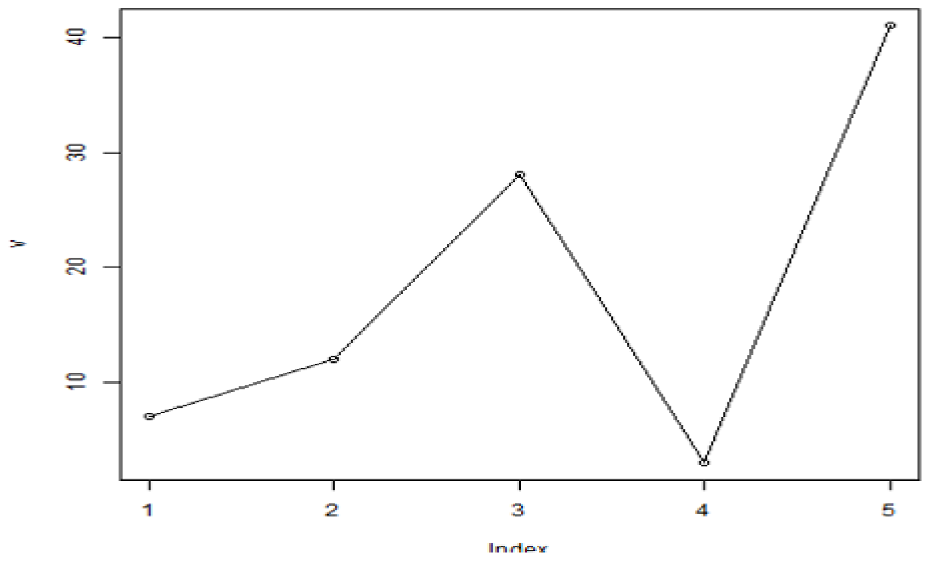
Following is the description of the parameters used:

- **v** is a vector containing the numeric values.
- **type** takes the value "p" to draw only the points, "i" to draw only the lines and "o" to draw both points and lines.
- **xlab** is the label for x axis.
- **ylab** is the label for y axis.
- **main** is the Title of the chart.
- **col** is used to give colors to both the points and lines.

Single line Example

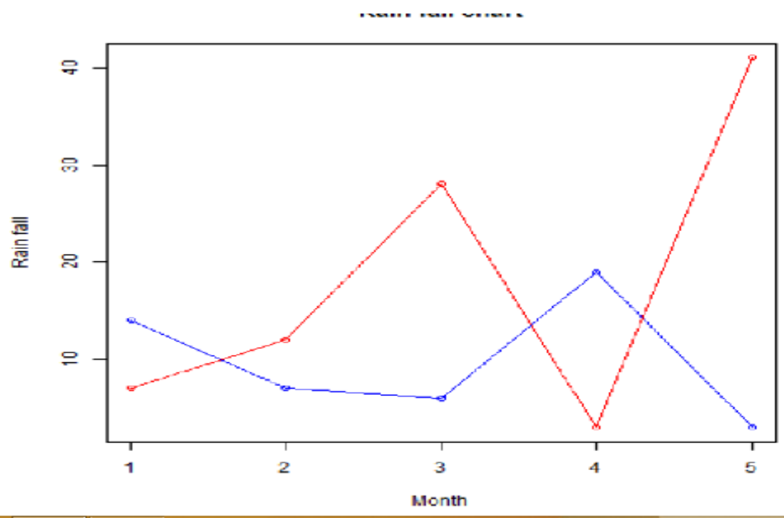
A simple line chart is created using the input vector and the type parameter as "O". The below script will create and save a line chart in the current R working directory.

```
Create the data for the chart.
v <- c(7,12,28,3,41)
# Give the chart file a name.
png(file = "line_chart.jpg")
# Plot the bar chart.
plot(v,type="o")
# Save the file.
dev.off()
```



Multiple lines: More than one line can be drawn on the same chart by using the **lines()**function.After the first line is plotted, the lines() function can use an additional vector as input to draw the second line in the chart,

```
# Create the data for the chart.
v <- c(7,12,28,3,41)
t <- c(14,7,6,19,3)
# Give the chart file a name.
png(file = "line_chart_2_lines.jpg")
# Plot the bar chart.
plot(v,type="o",col="red",xlab="Month",ylab="Rain fall",main="Rain fall chart")
lines(t, type="o", col="blue")

# Save the file.
dev.off()
```



6,R-Scatterplots: Scatterplots show many points plotted in the Cartesian plane. Each point represents the values of two variables. One variable is chosen in the horizontal axis and another in the vertical axis. The simple scatterplot is created using the **plot()** function.

Syntax

The basic syntax for creating scatterplot in R is :

```
plot(x, y, main, xlab, ylab, xlim, ylim, axes)
```

Following is the description of the parameters used:

- **x** is the data set whose values are the horizontal coordinates.
- **y** is the data set whose values are the vertical coordinates.
- **main** is the tile of the graph.
- **xlab** is the label in the horizontal axis.
- **ylab** is the label in the vertical axis.
- **xlim** is the limits of the values of x used for plotting.
- **ylim** is the limits of the values of y used for plotting.
- **axes** indicates whether both axes should be drawn on the plot.

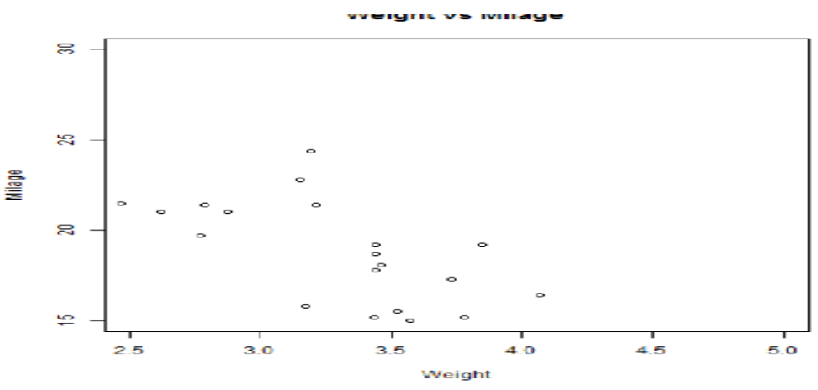
Example

We use the data set **"mtcars"** available in the R environment to create a basic scatterplot.
Let's use the columns "wt" and "mpg" in mtcars.

```
input <- mtcars[,c('wt','mpg')]
print(head(input))
```

#The below script will create a scatterplot graph for the relation between wt(weight) and mpg(miles per gallon).

```
# Get the input values.
input <- mtcars[,c('wt','mpg')]
# Give the chart file a name.
png(file = "scatterplot.png")
# Plot the chart for cars with weight between 2.5 to 5 and mileage between 15
and 30.
plot(x=input$wt,y=input$mpg,
xlab="Weight",
ylab="Milage",
xlim=c(2.5,5),
ylim=c(15,30),
main="Weight vs Milage")
# Save the file.

dev.off()
```



OUTPUT

VIVA QUESTIONS

1. **Give the command to create a Histogram.**

2. **Give the syntax for creating scatterplot matrices.**

3. **Give the command to create Boxplot**

4. **Give the command to create Multiple lines**

5. **Give the command to create Barcharts**

6. **Give the command to create Piecharts**