

# **CLOUD COMPUTING**

**PROGRAMMING ASSIGNMENT – 2**

**PROJECT REPORT**

## **1. Introduction:**

This programming assignment is basically made to get used to with the Amazon Web Services (AWS) virtual cluster creation, deployment and use. Besides this main aspect, I had setup virtual cluster for hadoop and spark for single as well as multiple nodes (16 nodes). After setting up these environment, created sorting program for hadoop and spark for 10GB and 100GB of data. Next task was to implement the 10GB data sorting algorithm using any sorting method.

## **2. Runtime Environment Setup:**

### **Amazon Web Services EC2**

AMI – Amazon Linux (HVM), SSD Volume Type – ami-3d50120d , 2 x 16 GB

Hadoop – 2.7.2

Spark - 1.6.1

Java - 1.7.0\_95

### **Shared-Memory:**

EC2 instance type : c3-large

### **Hadoop part :**

EC2 instance type : (master node) c3.4xlarge, (slaves node)c3.large

### **Spark part:**

EC2 instance type : (master node) c3.4xlarge, (slaves node)c3.large

### **➤ Details of EC2 Instances Used :**

1. Instance type : c3.large Linux

RAM : 3.75

Cores : 2

Storage : 32 GB

2. Instance type : c3.4xlarge Linux

RAM : 30

Cores : 16

Storage : 320 GB

### **3. Programming and Designing:**

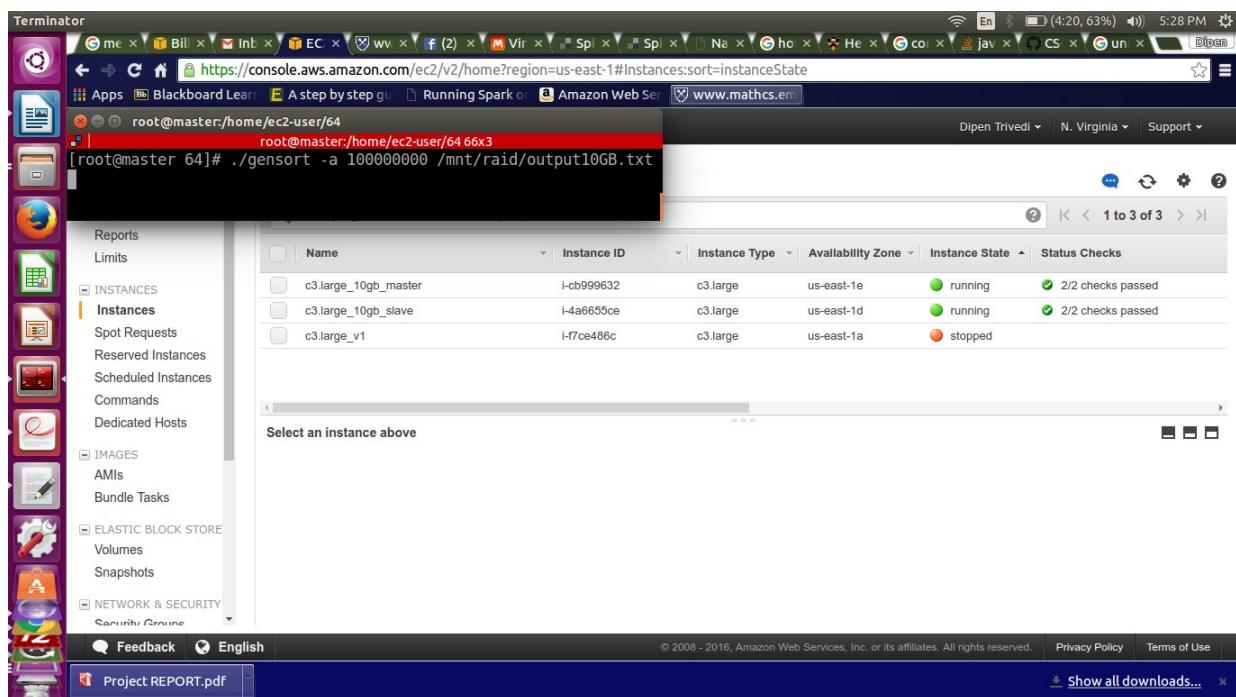
For programming and building, JAVA and Python are used for hadoop and spark respectively. As JAVA provides libraries to perform the map-reduce functionality easily on hadoop cluster and Python is good for the scripting, both were good fit for the hadoop and spark. Working in the Linux Environment made the work more reliable and efficient, as the terminal commands and methodology are same on the EC2 instances also. Used c3.large and c3.4xlarge instances to perform the tasks. Used Gensort to generate the 10GB/100GB data and valsor to check the sorted data.

### **4. Data Generation and Validation :**

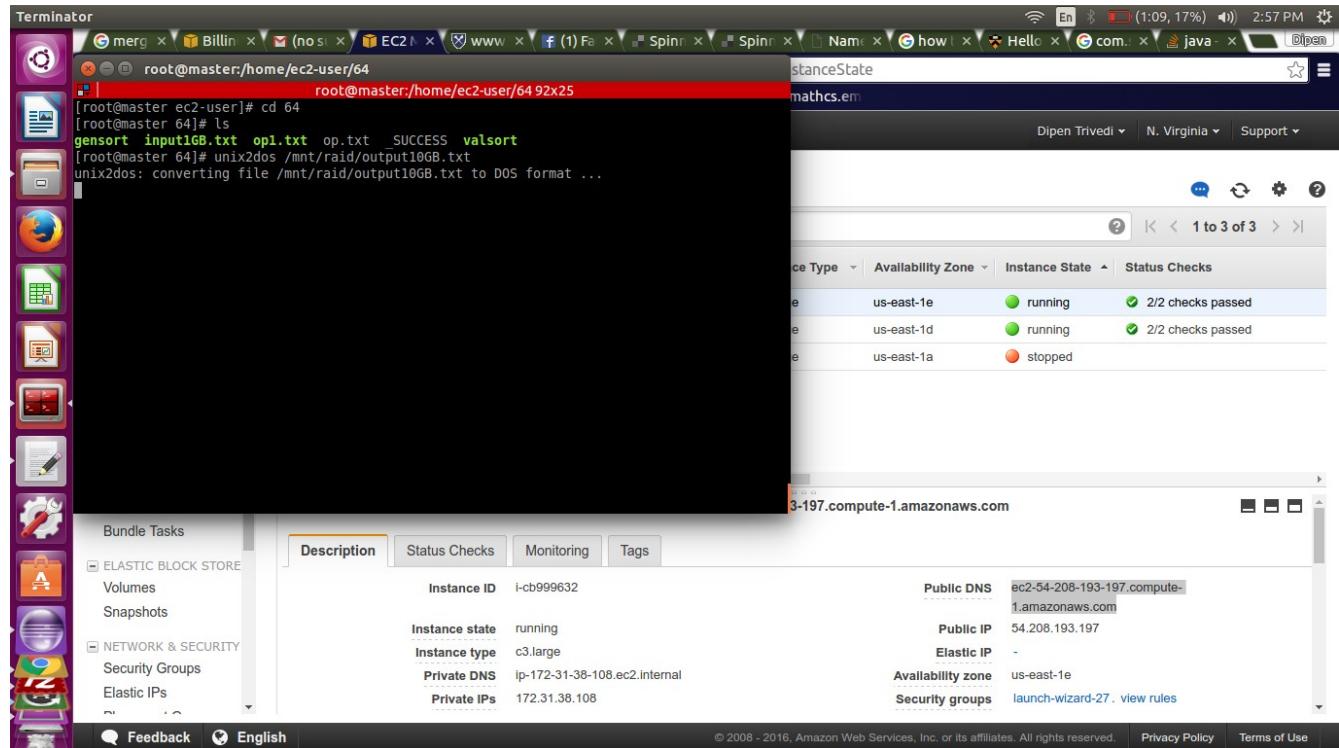
To generate 10GB and 100GB data, Gensort is used. Gensort generates the data as per the input value of number of line and the format of the file. '-a' creates the ascii file.

To validate the sorted data Valsort is used. To make the data operated by valsor, first step is to convert it the dos format as valsor only accept the input in dos format. Then, just run the valsor on the output file.

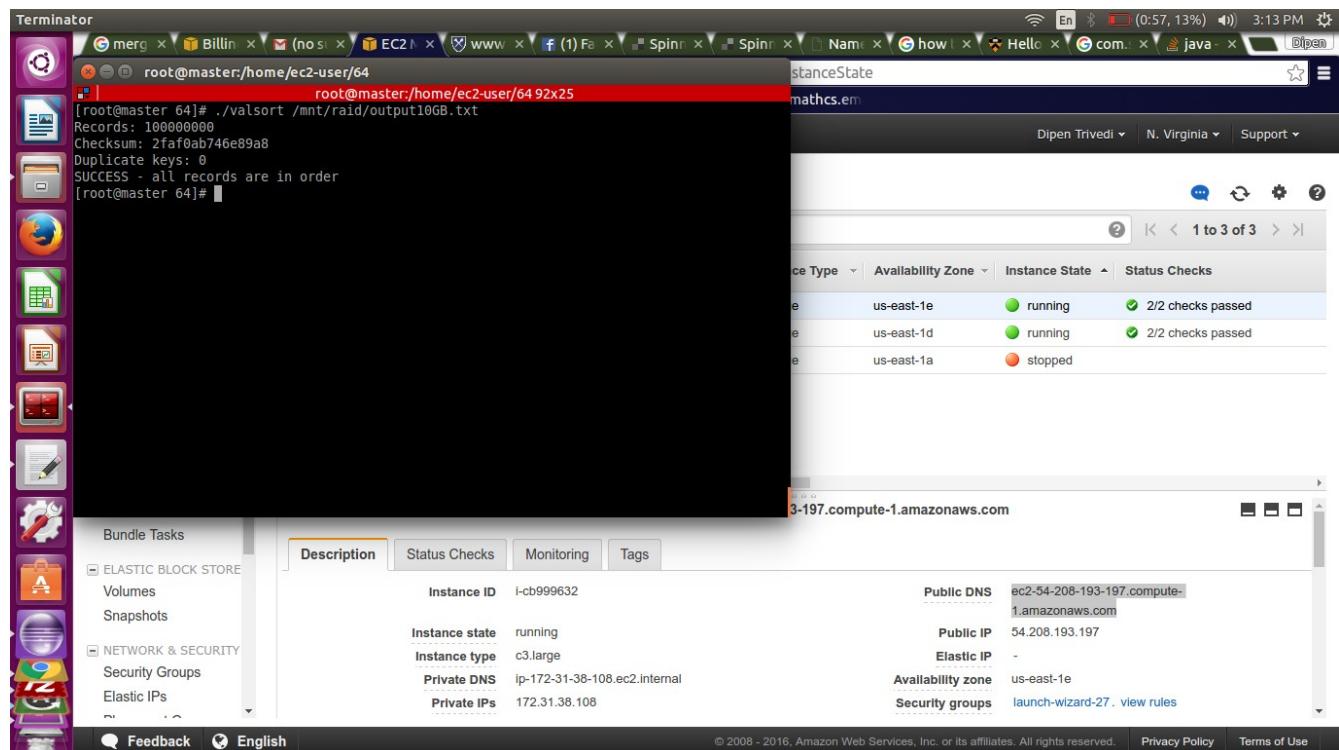
#### **◆ 10GB data generation with Gensort :**



## ◆ 10GB data unix2dos :



## ◆ 10GB data validation with valsrt :



## 5. Virtual Cluster Launch (1-node) :

- Go to Instances in the EC2 Instances tab.

The screenshot shows the AWS EC2 Management Console in Google Chrome. The left sidebar has 'Instances' selected under 'EC2 Dashboard'. The main area shows summary statistics: 2 Running Instances, 1 Elastic IP, 15 Snapshots, 17 Volumes, 0 Load Balancers, 3 Key Pairs, and 0 Placement Groups. Below this is a 'Create Instance' section with a 'Launch Instance' button. The right side includes sections for 'Account Attributes' (Supported Platforms: VPC; Default VPC: vpc-471a5023), 'Additional Information' (Getting Started Guide, Documentation, All EC2 Resources, Forums, Pricing, Contact Us), and 'AWS Marketplace' (Find free software trial products). The URL in the address bar is https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#instances:.

- Select launch instance. Select any instance(c3.large here) and then configure it.

The screenshot shows the 'Choose an Instance Type' step of the AWS EC2 Launch Instance Wizard. The 'Compute optimized' c3.large instance is selected. The table below lists other instance types:

Instance Type	Memory (GiB)	Processor Speed (GHz)	Storage Options	Network Options	Support
m3.2xlarge	8	30	2 x 80 (SSD)	Yes	High
c4.large	2	3.75	EBS only	Yes	Moderate
c4.xlarge	4	7.5	EBS only	Yes	High
c4.2xlarge	8	15	EBS only	Yes	High
c4.4xlarge	16	30	EBS only	Yes	High
c4.8xlarge	36	60	EBS only	Yes	10 Gigabit
<b>c3.large</b>	<b>2</b>	<b>3.75</b>	<b>2 x 16 (SSD)</b>	-	<b>Moderate</b>
c3.xlarge	4	7.5	2 x 40 (SSD)	Yes	Moderate
c3.2xlarge	8	15	2 x 80 (SSD)	Yes	High
c3.4xlarge	16	30	2 x 160 (SSD)	Yes	High
c3.8xlarge	32	60	2 x 320 (SSD)	-	10 Gigabit

At the bottom are 'Cancel', 'Previous', 'Review and Launch' (disabled), and 'Next: Configure Instance Details'.

➤ Select instance on-demand or on-spot.

**Step 3: Configure Instance Details**

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

**Number of instances:** 1 **Purchasing option:** Request Spot instances **Network:** vpc-471a5023 (172.31.0.0/16) (default) **Subnet:** No preference (default subnet in any Availability Zone) **Auto-assign Public IP:** Use subnet setting (Enable) **Placement group:** No placement group **IAM role:** None **Shutdown behavior:** Stop **Enable termination protection:** Protect against accidental termination **Monitoring:** Enable CloudWatch detailed monitoring

**Buttons:** Cancel, Previous, Review and Launch, Next: Add Storage

**Step 4: Add Storage**

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination	Encrypted
Root	/dev/xvda	snap-12c47a84	8	General Purpose SSD (GP2)	24 / 3000	<input checked="" type="checkbox"/>	Not Encrypted
EBS	/dev/sdb	Search (case-insensit)	30	General Purpose SSD (GP2)	90 / 3000	<input type="checkbox"/>	<a href="#">X</a>

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

**Buttons:** Cancel, Previous, Review and Launch, Next: Tag Instance

- After give label to the instance.

**Step 5: Tag Instance**

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. [Learn more](#) about tagging your Amazon EC2 resources.

<b>Key</b> (127 characters maximum)	<b>Value</b> (255 characters maximum)
Name	c3.large_1node

**Create Tag** (Up to 10 tags maximum)

**Buttons:** Cancel, Previous, Review and Launch, Next: Configure Security Group

- Configure the security group to allow different traffic in network. Here. All TCP, All ICMP, SSH and All traffic rules are given to the instance.

**Step 6: Configure Security Group**

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group:  Create a new security group  
 Select an existing security group

Security group name: launch-wizard-28

Description: launch-wizard-28 created 2016-03-28T19:50:22.531-05:00

Type	Protocol	Port Range	Source
SSH	TCP	22	Anywhere 0.0.0.0/0
All TCP	TCP	0 - 65535	Anywhere 0.0.0.0/0
All ICMP	ICMP	0 - 65535	Anywhere 0.0.0.0/0
All traffic	All	0 - 65535	Anywhere 0.0.0.0/0

**Buttons:** Add Rule, Cancel, Previous, Review and Launch

➤ Review and launch the instance.

**Step 7: Review Instance Launch**

Instance Type: c3.large

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
c3.large	7	2	3.75	2 x 16	-	Moderate

Security Groups: launch-wizard-28

Type	Protocol	Port Range	Source
SSH	TCP	22	0.0.0.0/0
All TCP	TCP	0 - 65535	0.0.0.0/0
All ICMP	All	N/A	0.0.0.0/0
All traffic	All	All	0.0.0.0/0

Buttons: Cancel, Previous, Launch, Define key pair and launch

**Select an existing key pair or create a new key pair**

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

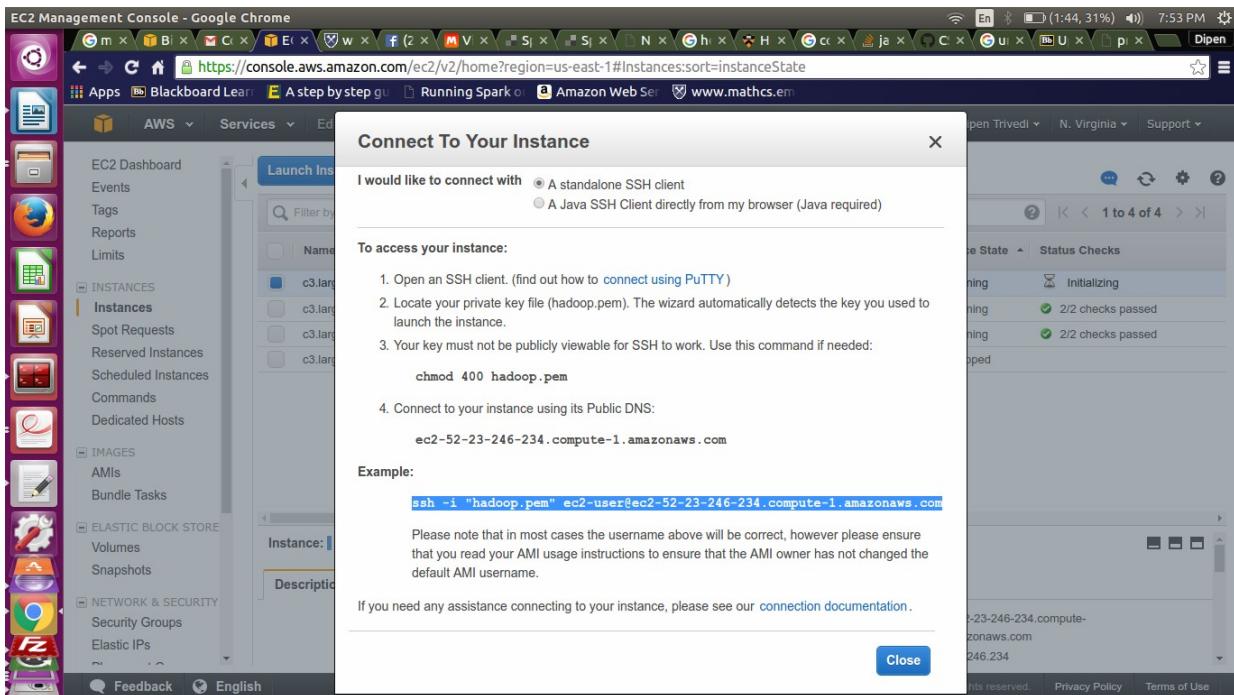
Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

Choose an existing key pair  
Select a key pair

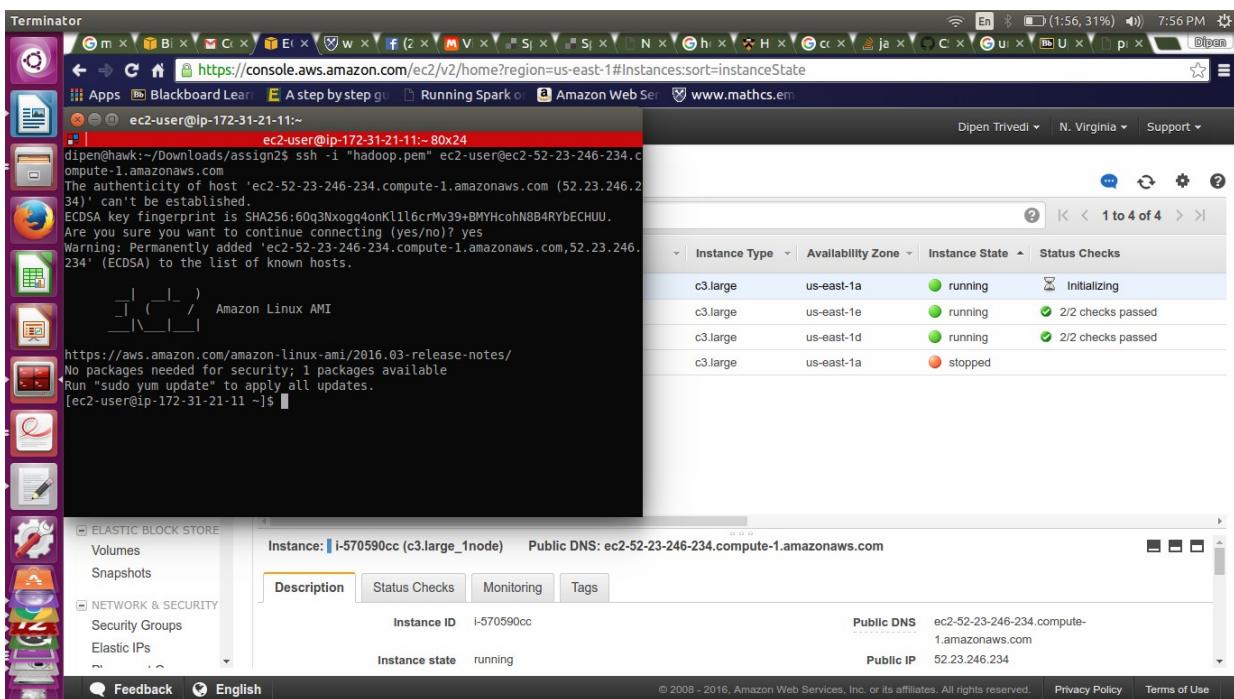
- AWS Key
- hadoop
- spark

Buttons: Cancel, Launch Instances

- To access the instance, take action connect and copy the SSH command to the terminal.



- Instance is connected now.



## **6. Hadoop Sort:**

### **● Hadoop Sorting Algorithm:**

Created one java program for hadoop file system that sorts the input data. In Hadoop, Map-Reduce function takes the data in <key,value> format. To get key and value, substring the input data line by line, and assign it to the key and value respectively. Sorting is done by the frame work and after data sorting is done, Reduce function comes in the action. The main work of the reduce function is to combine the same key data, and then to add them in one <key,value> format. After that, write the data in same format as the input, so that it will be easy to validate using the valsort.

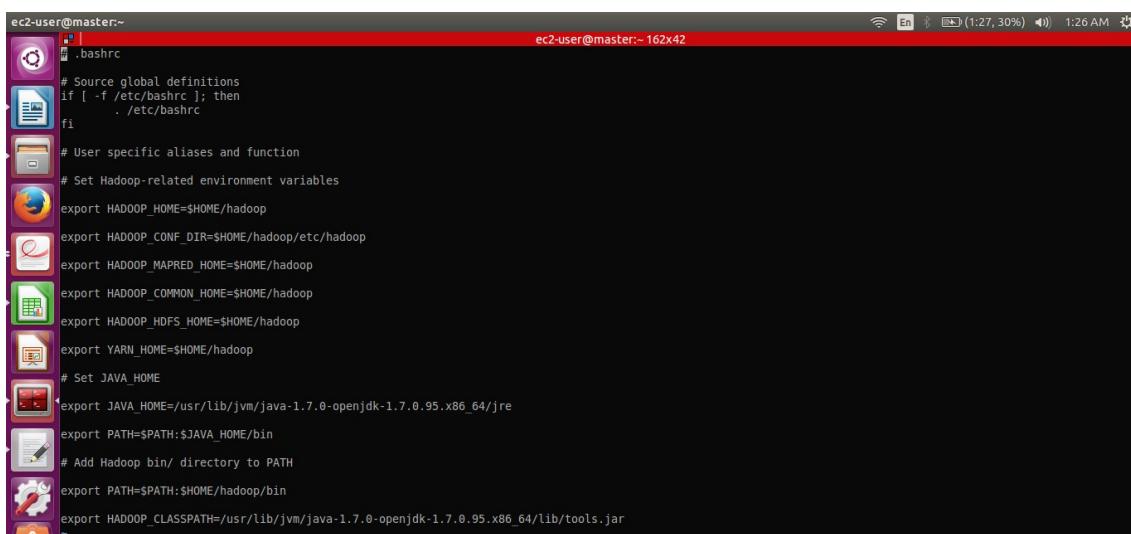
### **● Hadoop Configuration:**

After starting the instance we have to run following commands to install the hadoop in the system:

- ◆ *sudo apt-get update*
- ◆ *sudo apt-get install openjdk-7-jdk*
- ◆ *wget https://archive.apache.org/dicst/hadoop/core/hadoop-2.7.2/hadoop-2.7.2.tar.gz*
- ◆ *sudo tar -xvf hadoop-2.7.2.tar.gz*

Now add hadoop and java environment variables in *.bashrc* file.

- ◆ *Sudo vi .bashrc*



```
ec2-user@master:~/.bashrc
# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# User specific aliases and functions

# Set Hadoop-related environment variables
export HADOOP_HOME=$HOME/hadoop
export HADOOP_CONF_DIR=$HOME/hadoop/etc/hadoop
export HADOOP_MAPRED_HOME=$HOME/hadoop
export HADOOP_COMMON_HOME=$HOME/hadoop
export HADOOP_HDFS_HOME=$HOME/hadoop
export YARN_HOME=$HOME/hadoop

# Set JAVA_HOME
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.95.x86_64/jre
export PATH=$PATH:$JAVA_HOME/bin
# Add Hadoop bin/ directory to PATH
export PATH=$PATH:$HOME/hadoop/bin
export HADOOP_CLASSPATH=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.95.x86_64/lib/tools.jar
```

- ◆ *source .bashrc*

Add java home path in *hadoop-env.sh*

- ◆ *sudo vi /hadoop/etc/hadoop/hadoop-env.sh*

```
ec2-user@master:~/hadoop/etc/hadoop
```

```
ec2-user@master:~/hadoop/etc/hadoop 162x42
```

```
# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements. See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership. The ASF licenses this file
# to you 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.

# 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/java-1.7.0-openjdk-1.7.0.05.x86_64

# 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}

# 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
```

Now, edit core-site.xml, which is configuration file for hadoop Core.

- ◆ `sudo vi /hadoop/etc/hadoop/core-site.xml`

The screenshot shows a terminal window with the following details:

- Title Bar:** Terminator
- URL:** https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:sort=desc:tag:Name
- Terminal Prompt:** ec2-user@master:~/hadoop/etc/hadoop
- Sub-Prompt:** ec2-user@master:~/hadoop/etc/hadoop 80x24
- Content:**
  - HTTP header: http://www.apache.org/licenses/LICENSE-2.0
  - Text: 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.
  - Text: -->
  - Text: <!-- Put site-specific property overrides in this file. -->
  - Configuration code:

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://master:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/mnt/raid/temp/hadoop-ec2-user</value>
</property>
</configuration>
```

To the right of the terminal, there is a table showing the status of four Amazon EC2 instances:

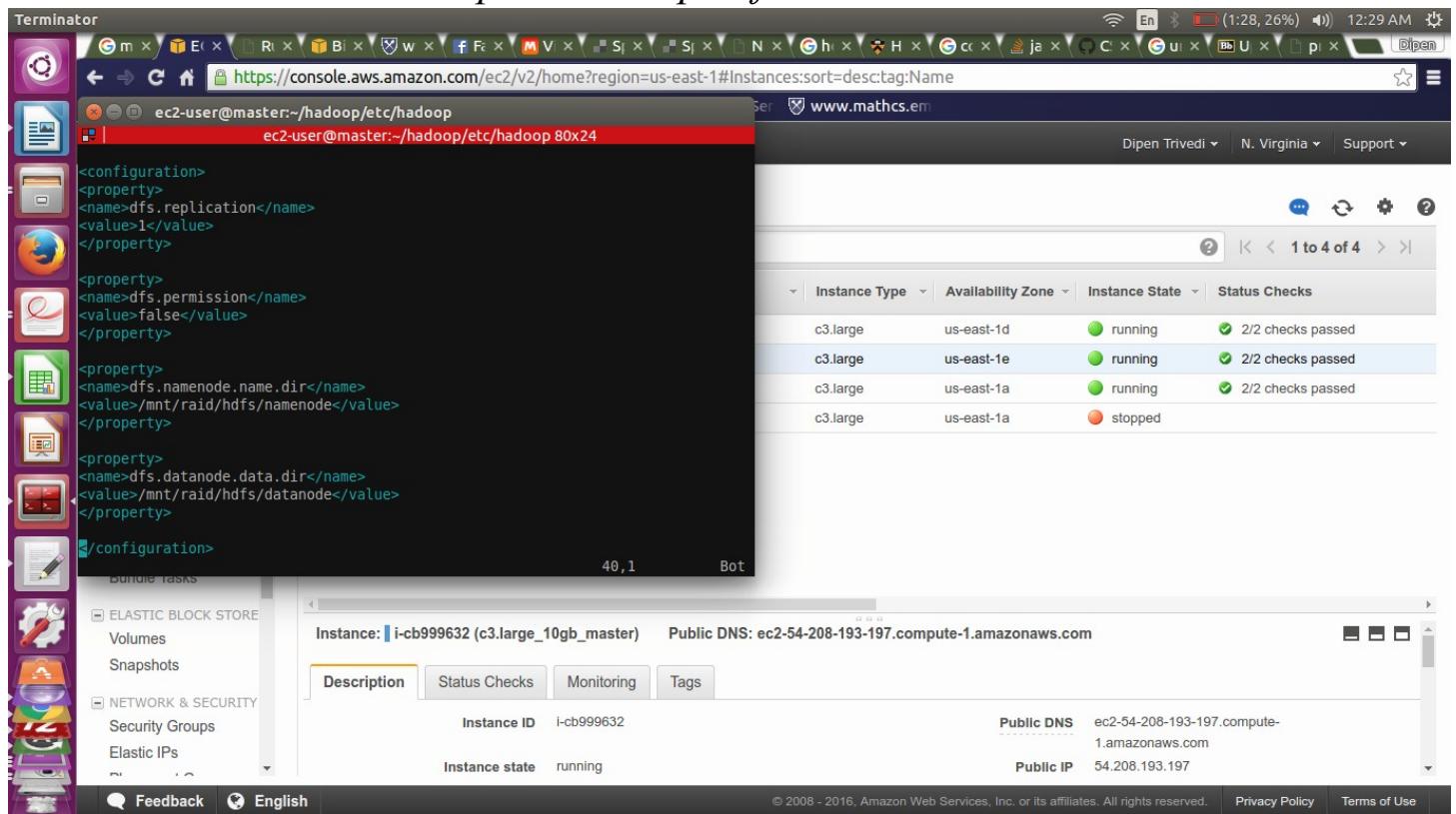
Instance Type	Availability Zone	Instance State	Status Checks
c3.large	us-east-1d	running	2/2 checks passed
c3.large	us-east-1e	running	2/2 checks passed
c3.large	us-east-1a	running	2/2 checks passed
c3.large	us-east-1a	stopped	

Create namenode and datanode:

- ◆ `mkdir -p /hadoop/hdfs/namenode`
- ◆ `mkdir -p /hadoop/hdfs/datanode`

Now, edit `hdfs-site.xml`, which contains the configuration settings for HDFS daemons.

- ◆ `Vi /hadoop/etc/hadoop/hdfs-site.xml`

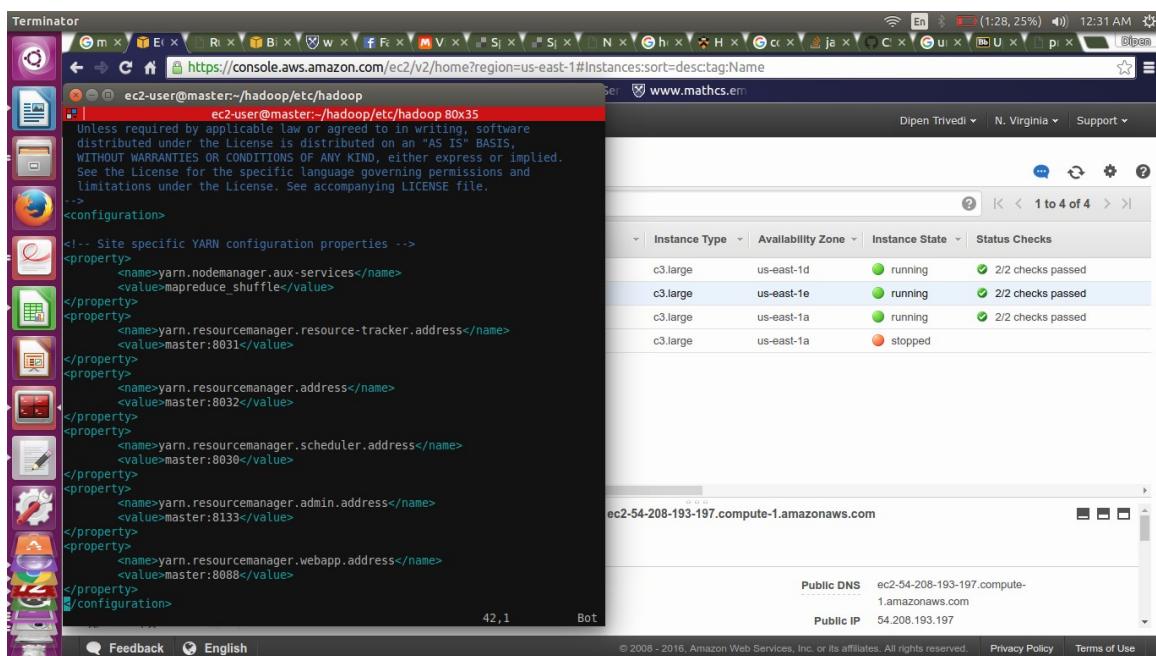


Edit mapred-site.xml, which contains the configuration settings for Map-Reduce daemons.

- ◆ *cp /hadoop/etc/hadoop/mapred-site.xml.template /hadoop/etc/hadoop/mapred-site.xml*
- ◆ *vi /hadoop/etc/hadoop/mapred-site.xml*

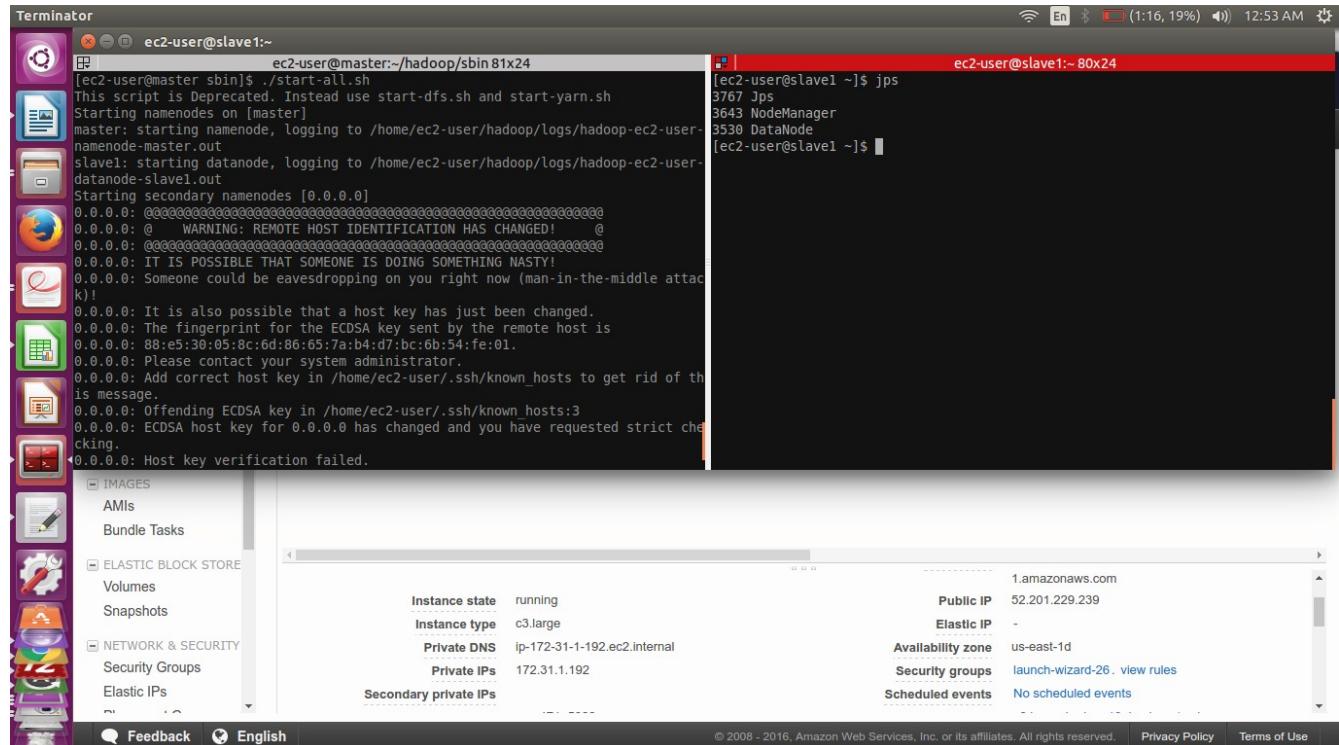
Edit yarn-site.xml, which contains settings for YARN.

- ◆ *vi /hadoop/etc/hadoop/yarn-site.xml*



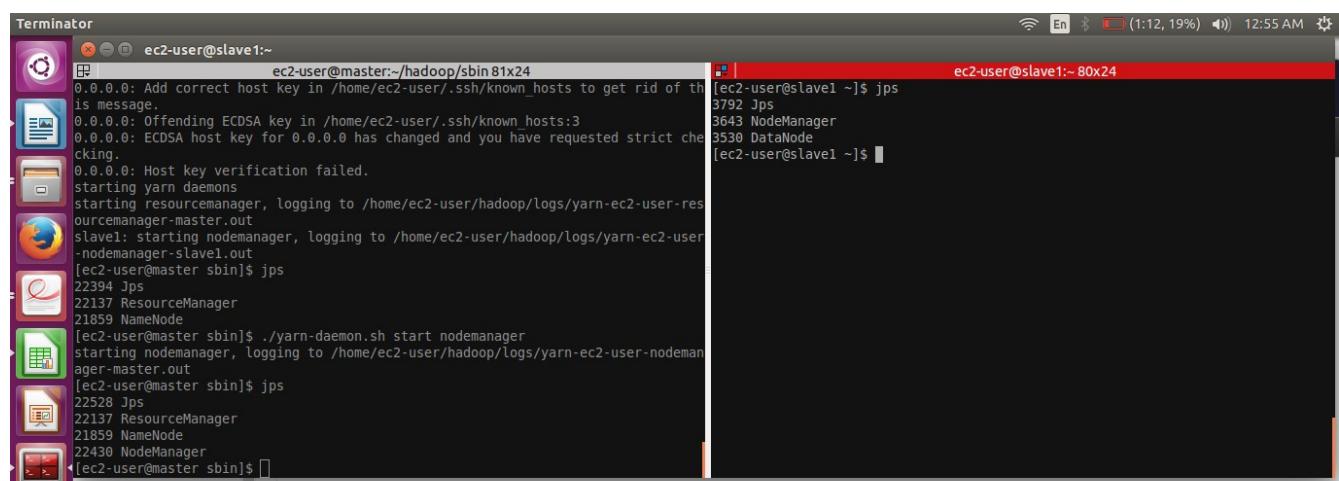
start all the daemons:

- ◆ *hadoop namenode -format*
- ◆ *./hadoop/sbin/start-all.sh*
- ◆ *./hadoop/sbin/yarn-daemon.sh start nodemanager*



check if the all the daemons are running :

- ◆ *jps*



## Mount the unmounted data drive commands :

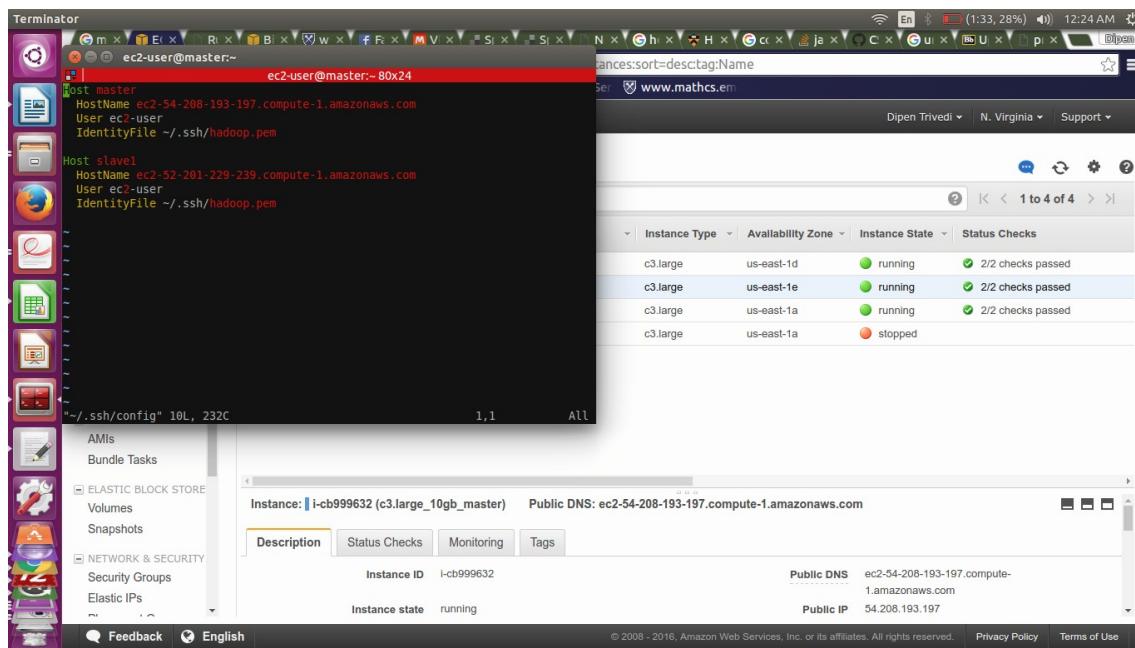
The following is the command to mount the unmounted drives into the hdfs:

- ◆ `sudo mdadm --create --verbose /dev/md0 --level=0 --name=hadoop --raid-devices=3 /dev/xvdb /dev/xvdc /dev/xvdd`
- ◆ `sudo mkfs.ext4 -L hadoop /dev/md0`
- ◆ `sudo mkdir -p /mnt/raid`
- ◆ `sudo mount LABEL=hadoop /mnt/raid`
- ◆ `sudo chmod 777 /mnt/raid`

## SSH Configuration :

To make stop specifying the pem key and host address every time we SSH from local to node, we have to config the `~/.ssh/config` file.

- ◆ `~/.ssh/config`



- ◆ `ssh-keygen -f ~/.ssh/id_rsa -t rsa -P ""`
- ◆ `cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys`
- ◆ `cat ~/.ssh/id_rsa.pub | ssh slave1 'cat >> ~/.ssh/authorized_keys'`

■ Create AMI of this configuration.

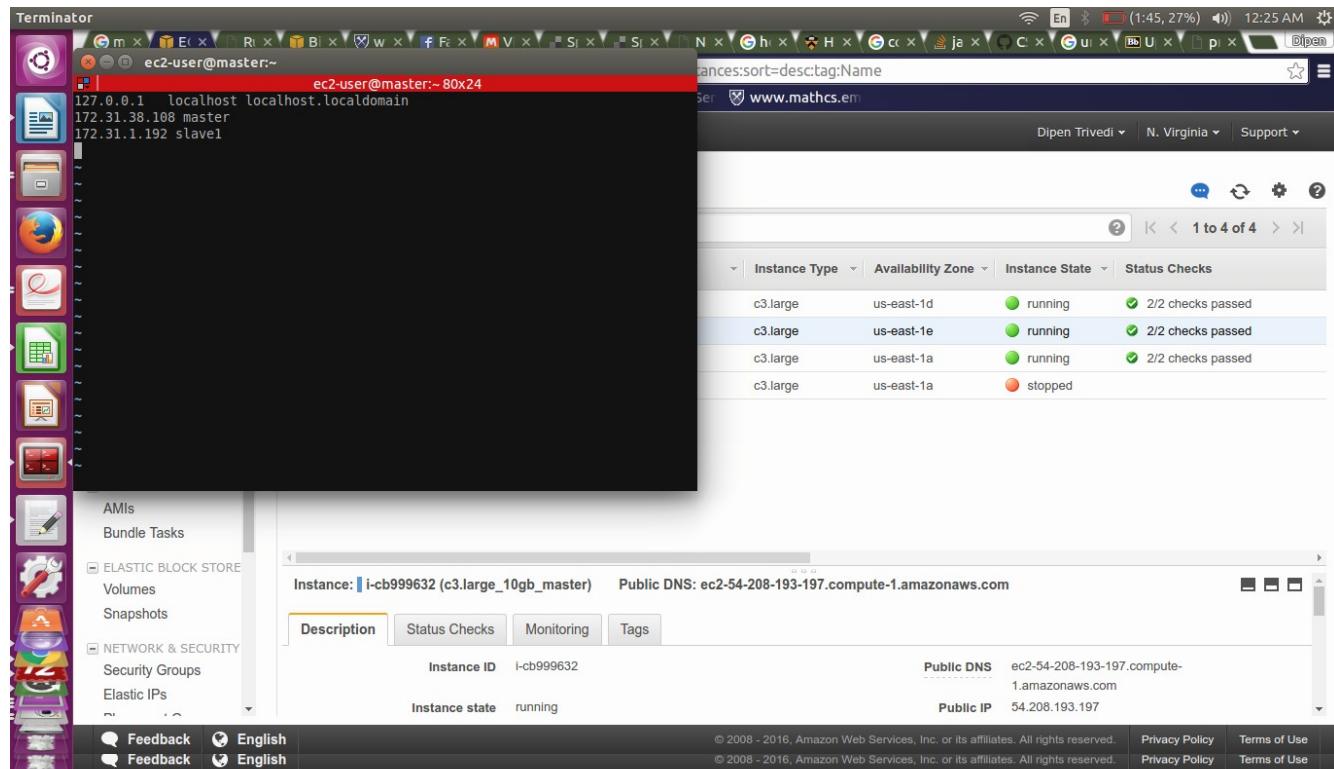
➤ **10GB data sort(1 master, 1 slave):**

To run 1 master and 1 slave, change following files after launching the above all commands to setup the hadoop:

Run the previously configure AMI:

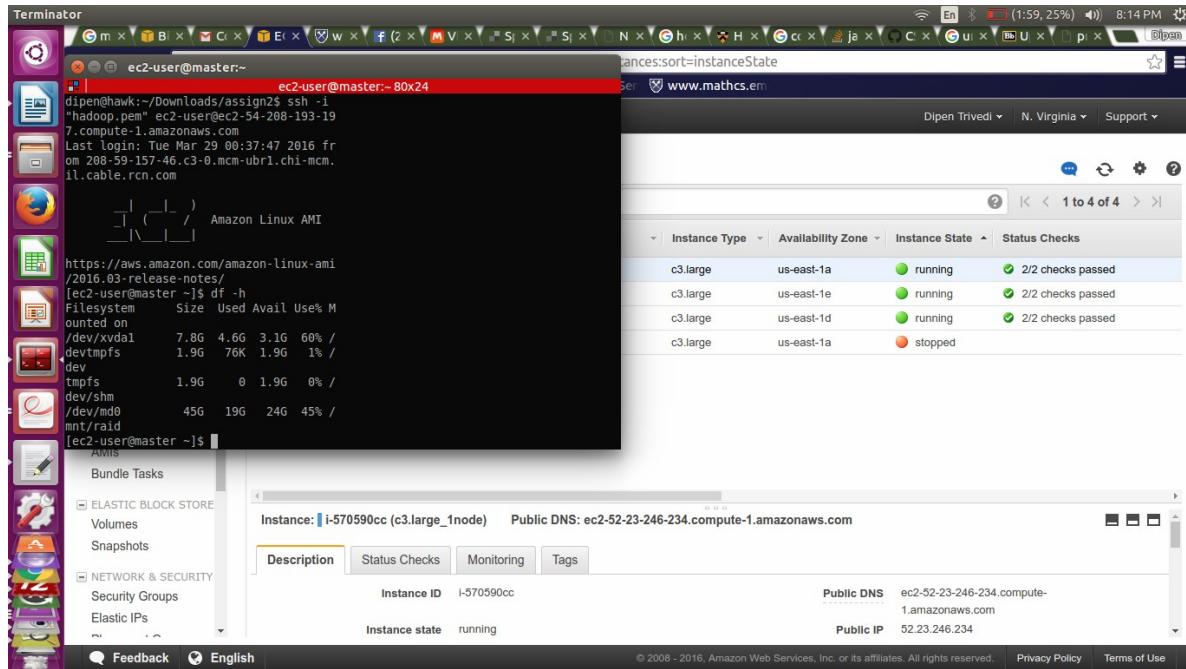
change the host and slave in /etc/hosts:

- ◆ `sudo vi /etc/hosts`



Now, run all daemons as suggested in hadoop configuration.

◆ Mounted drive's size: after sort :



◆ Compile Java File TeraSort.java:

```
hadoop com.sun.tools.javac.Main TeraSort.java
```

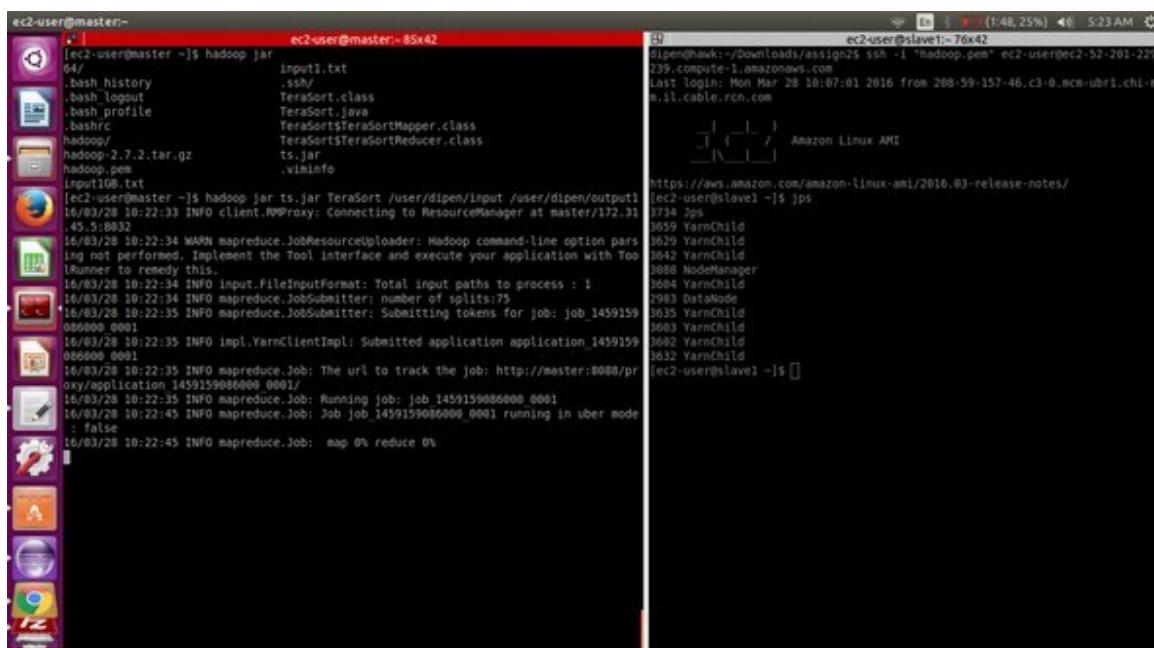
◆ Create Jar file of TeraSort.java:

```
jar cf ts.jar TeraSort*.class
```

◆ Run Jar File ts.jar :

```
jar ts.jar TeraSort /user/dipen/input /user/dipen/output
```

◆ Sorting : Map 0% & Reduce 0% :



## ◆ sorting : Map 40% & Reduce 11% :

```
ec2-user@master:~ ec2-user@master:~ 85x42
16/03/28 10:23:33 INFO mapreduce.Job: map 8% reduce 0%
16/03/28 10:23:37 INFO mapreduce.Job: map 9% reduce 0%
16/03/28 10:23:46 INFO mapreduce.Job: map 10% reduce 0%
16/03/28 10:23:51 INFO mapreduce.Job: map 11% reduce 0%
16/03/28 10:23:55 INFO mapreduce.Job: map 12% reduce 0%
16/03/28 10:23:56 INFO mapreduce.Job: map 13% reduce 0%
16/03/28 10:23:59 INFO mapreduce.Job: map 14% reduce 0%
16/03/28 10:24:04 INFO mapreduce.Job: map 15% reduce 0%
16/03/28 10:24:11 INFO mapreduce.Job: map 16% reduce 0%
16/03/28 10:24:14 INFO mapreduce.Job: map 17% reduce 0%
16/03/28 10:24:19 INFO mapreduce.Job: map 18% reduce 0%
16/03/28 10:24:21 INFO mapreduce.Job: map 19% reduce 0%
16/03/28 10:24:31 INFO mapreduce.Job: map 20% reduce 0%
16/03/28 10:24:37 INFO mapreduce.Job: map 20% reduce 1%
16/03/28 10:24:38 INFO mapreduce.Job: map 21% reduce 1%
16/03/28 10:24:40 INFO mapreduce.Job: map 21% reduce 2%
16/03/28 10:24:43 INFO mapreduce.Job: map 21% reduce 3%
16/03/28 10:24:44 INFO mapreduce.Job: map 22% reduce 3%
16/03/28 10:24:46 INFO mapreduce.Job: map 22% reduce 4%
16/03/28 10:24:56 INFO mapreduce.Job: map 23% reduce 4%
16/03/28 10:24:59 INFO mapreduce.Job: map 24% reduce 5%
16/03/28 10:25:00 INFO mapreduce.Job: map 25% reduce 5%
16/03/28 10:25:02 INFO mapreduce.Job: map 26% reduce 5%
16/03/28 10:25:05 INFO mapreduce.Job: map 27% reduce 5%
16/03/28 10:25:06 INFO mapreduce.Job: map 28% reduce 5%
16/03/28 10:25:09 INFO mapreduce.Job: map 29% reduce 5%
16/03/28 10:25:17 INFO mapreduce.Job: map 30% reduce 5%
16/03/28 10:25:21 INFO mapreduce.Job: map 30% reduce 6%
16/03/28 10:25:24 INFO mapreduce.Job: map 31% reduce 6%
16/03/28 10:25:27 INFO mapreduce.Job: map 32% reduce 8%
16/03/28 10:25:31 INFO mapreduce.Job: map 33% reduce 8%
16/03/28 10:25:43 INFO mapreduce.Job: map 34% reduce 8%
16/03/28 10:25:46 INFO mapreduce.Job: map 35% reduce 8%
16/03/28 10:25:48 INFO mapreduce.Job: map 36% reduce 8%
16/03/28 10:25:51 INFO mapreduce.Job: map 37% reduce 8%
16/03/28 10:25:53 INFO mapreduce.Job: map 38% reduce 8%
16/03/28 10:25:55 INFO mapreduce.Job: map 39% reduce 8%
16/03/28 10:25:58 INFO mapreduce.Job: map 39% reduce 9%
16/03/28 10:26:01 INFO mapreduce.Job: map 39% reduce 10%
16/03/28 10:26:07 INFO mapreduce.Job: map 39% reduce 11%
16/03/28 10:26:09 INFO mapreduce.Job: map 40% reduce 11%

ec2-user@slave1:~ 76x42
En (2:01, 25%) 5:26 AM
dipen@hawk:~/Downloads/assign2$ ssh -i "hadoop.pem" ec2-user@ec2-52-201-229-239.compute-1.amazonaws.com
Last login: Mon Mar 28 10:07:01 2016 from 208-59-157-46.c3-0.mcm-ubr1.chi-mc.mil.cable.rcn.com
[ec2-user@slave1 ~]$ ls
[ec2-user@slave1 ~]$ jps
3734 Jps
3659 YarnChild
3629 YarnChild
3642 YarnChild
3088 NodeManager
3604 YarnChild
2983 DataNode
3635 YarnChild
3603 YarnChild
3602 YarnChild
3632 YarnChild
[ec2-user@slave1 ~]$ 
```

## ◆ sorting : Map 100% & Reduce 100% :

```
ec2-user@master:~ ec2-user@master:~ 85x42
Rack-local map tasks=31
Total time spent by all maps in occupied slots (ms)=6202082
Total time spent by all reduces in occupied slots (ms)=726970
Total time spent by all map tasks (ms)=6202082
Total time spent by all reduce tasks (ms)=726970
Total vcore-milliseconds taken by all map tasks=6202082
Total vcore-milliseconds taken by all reduce tasks=726970
Total megabyte-milliseconds taken by all map tasks=6350931968
Total megabyte-milliseconds taken by all reduce tasks=744417280

Map-Reduce Framework
Map input records=100000000
Map output records=100000000
Map output bytes=9900000000
Map output materialized bytes=10100000450
Input split bytes=8550
Combine input records=100000000
Combine output records=100000000
Reduce input groups=100000000
Reduce shuffle bytes=10100000450
Reduce input records=100000000
Reduce output records=100000000
Spilled Records=396636763
Shuffled Maps=75
Failed Shuffles=0
Merged Map outputs=75
GC time elapsed (ms)=29032
CPU time spent (ms)=1602430
Physical memory (bytes) snapshot=19410812928
Virtual memory (bytes) snapshot=74933293056
Total committed heap usage (bytes)=14615576576

Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0

File Input Format Counters
Bytes Read=10000303104
File Output Format Counters
Bytes Written=99000000000
[ec2-user@master ~]$ 
```

```
ec2-user@slave1:~ 76x42
En (1:37, 23%) 5:36 AM
dipen@hawk:~/Downloads/assign2$ ssh -i "hadoop.pem" ec2-user@ec2-52-201-229-239.compute-1.amazonaws.com
Last login: Mon Mar 28 10:07:01 2016 from 208-59-157-46.c3-0.mcm-ubr1.chi-mc.mil.cable.rcn.com
[ec2-user@slave1 ~]$ ls
[ec2-user@slave1 ~]$ jps
3734 Jps
3659 YarnChild
3629 YarnChild
3642 YarnChild
3088 NodeManager
3604 YarnChild
2983 DataNode
3635 YarnChild
3603 YarnChild
3602 YarnChild
3632 YarnChild
[ec2-user@slave1 ~]$ 
```

**◆ hadoop web interface overview after 100% map-reduce :**

**Summary**

Security is off.  
Safemode is off.

23 files and directories, 81 blocks = 104 total filesystem object(s).

Heap Memory used 138.13 MB of 174.5 MB Heap Memory. Max Heap Memory is 889 MB.

Non Heap Memory used 33.14 MB of 34.44 MB Committed Non Heap Memory. Max Non Heap Memory is 214 MB.

Configured Capacity:	59.43 GB
DFS Used:	9.39 GB (15.79%)
Non DFS Used:	8.84 GB
DFS Remaining:	41.21 GB (69.33%)
Block Pool Used:	9.39 GB (15.79%)
DataNodes usages% (Min/Median/Max/stdDev):	15.79% / 15.79% / 15.79% / 0.00%
Live Nodes	1 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	2
Number of Blocks Pending Deletion	0
Block Deletion Start Time	3/28/2016, 4:57:53 AM

**◆ validate the data using Valsort :**

```
ec2-user@master:~/64 [ec2-user@master 64]$ ./valsort /mnt/raid/part-r-00000
Records: 100000000
Checksum: 2faef2a1dfa8909
Duplicate keys: 0
SUCCESS - all records are in order
[ec2-user@master 64]$
```

➤ 100GB data sort(1 master, 16 slave) :

To run 1 master and 1 slave, change following files after launching the above all commands to setup the hadoop:

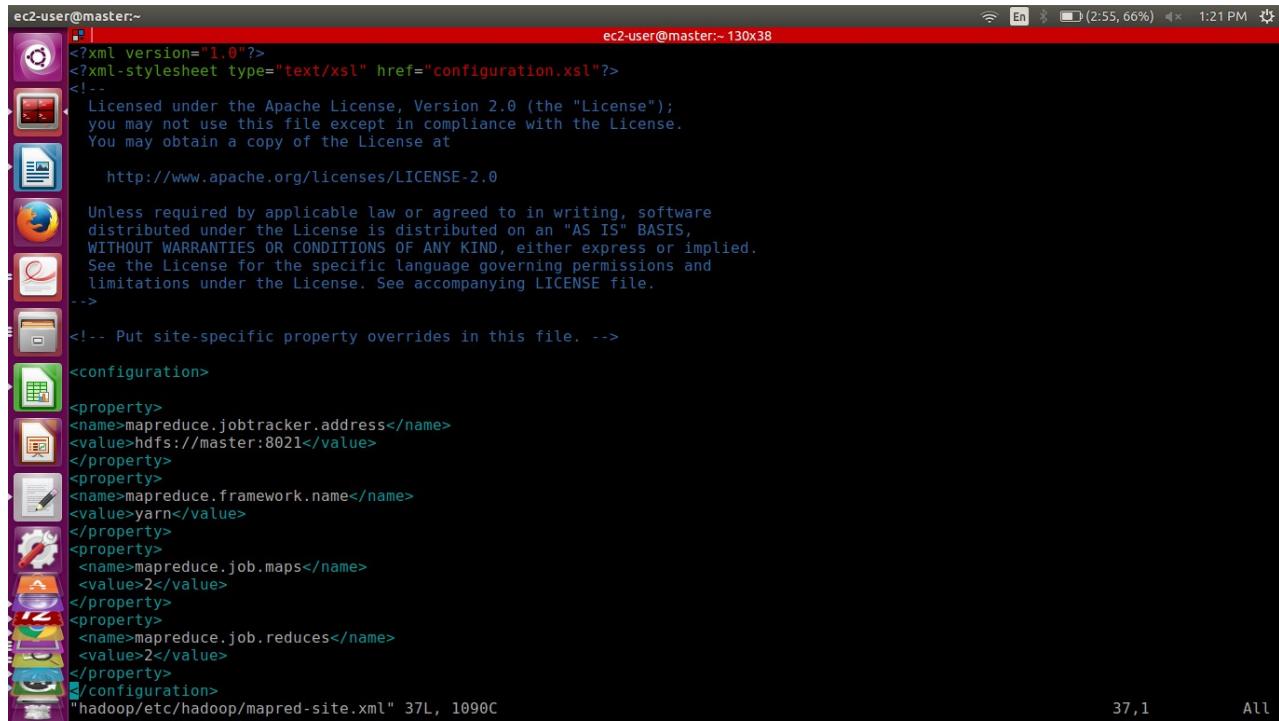
- Run the previously configure AMI.
  - Now, following are the changes that we have to made to configure the Multiple slave in the Master Cluster:

### 1. change in core-site.xml :

Add the hadoop.tmp.dir property.

## 2. change in mapred-site.xml

Add mapreduce.job.maps and mapreduce.job.reduces properties.



```

ec2-user@master:~ ec2-user@master:~ 130x38
<?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.
-->

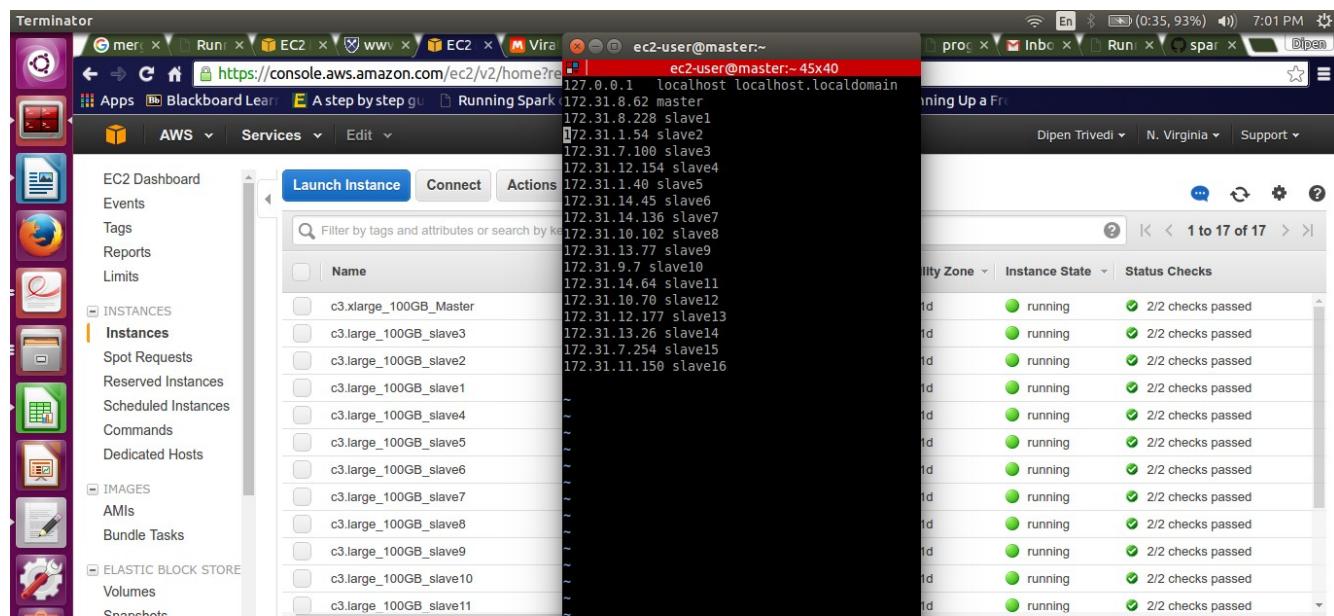
<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
<name>mapreduce.jobtracker.address</name>
<value>hdfs://master:8021</value>
</property>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
<property>
<name>mapreduce.job.maps</name>
<value>2</value>
</property>
<property>
<name>mapreduce.job.reduces</name>
<value>2</value>
</property>
</configuration>
".hadoop/etc/hadoop/mapred-site.xml" 37L, 1090C

```

## 3. change the /etc/hosts in master:

◆ sudo vi /etc/hosts



#### 4. change the `~/.ssh/config` in master:

- ◆ `sudo vi ~/.ssh/config`

```

Host master
HostName ec2-54-210-242-230.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/hadoop.pem

Host slave1
HostName ec2-54-208-69-138.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/hadoop.pem

Host slave2
HostName ec2-52-91-171-180.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/hadoop.pem

Host slave3
HostName ec2-54-173-249-152.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/hadoop.pem

Host slave4
HostName ec2-52-91-37-167.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/hadoop.pem

Host slave5
HostName ec2-52-90-173-44.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/hadoop.pem

Host slave6
HostName ec2-52-23-235-191.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/hadoop.pem

Host slave7
HostName ec2-52-205-255-176.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/hadoop.pem

```

#### 5. change the `/hadoop/etc/hadoop/slaves` file in master:

- ◆ `sudo vi /hadoop/etc/hadoop/slaves`

```

slave2
slave3
slave4
slave5
slave6
slave7
slave8
slave9
slave10
slave11
slave12
slave13
slave14
slave15
slave16

```

- Mount the unmounted drives in the master and slaves. For that run the bash file created for mounting drives, formating name nodes and changing the /etc/hosts and /hadoop/etc/slaves.
- ◆ *bash Mount1.sh*

```

[ec2-user@slave2:~]$ bash Mount1.sh
Agent pid 11673
Identity added: hadoop.pem (hadoop.pem)
Loaded plugins: priorities, update-motd, upgrade-helper
Package mdadm-3.2.6-7.32.amzn1.x86_64 already installed and latest version
Nothing to do
umount: /dev/xvdb: not mounted
mdadm: super1.x cannot open /dev/xvdc: Device or resource busy
mdadm: ddf: Cannot use /dev/xvdc: Device or resource busy
mdadm: Cannot use /dev/xvdc: It is busy
mdadm: cannot open /dev/xvdc: Device or resource busy
mke2fs 1.42.12 (29-Aug-2014)
/dev/md0 contains a ext4 file system labelled 'MY RAID'
        last mounted on /mnt/raid on Tue Mar 29 22:38:21 2016
Proceed anyway? (y,n) y
/dev/md0 is mounted; will not make a filesystem here!
mount: /dev/md0 is already mounted or /mnt/raid busy
        /dev/md0 is already mounted on /mnt/raid
File Number : 2
[ec2-user@slave2 ~]$ 

```

- Now, put the 100GB input file into the hdfs file system.
- Create Output folder in hdfs file system.
- Following screen shot shows the file transfer between master and slaves hdfs file system.

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/xvda1	7.8G	4.7G	3.1G	61%	/
devtmpfs	15G	76K	15G	1%	/dev
tmpfs	15G	0	15G	0%	/dev/shm
/dev/md0	301G	94G	192G	33%	/mnt/raid

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/xvda1	7.8G	3.7G	4.1G	48%	/
devtmpfs	1.9G	80K	1.9G	1%	/dev
tmpfs	1.9G	0	1.9G	0%	/dev/shm
/dev/md0	60G	4.8G	52G	9%	/mnt/raid

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/xvda1	7.8G	3.7G	4.1G	48%	/
devtmpfs	1.9G	80K	1.9G	1%	/dev
tmpfs	1.9G	0	1.9G	0%	/dev/shm
/dev/md0	60G	5.6G	51G	10%	/mnt/raid

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/xvda1	7.8G	3.7G	4.1G	48%	/
devtmpfs	1.9G	84K	1.9G	1%	/dev
tmpfs	1.9G	0	1.9G	0%	/dev/shm
/dev/md0	30G	4.9G	24G	18%	/mnt/raid

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/xvda1	7.8G	3.7G	4.1G	48%	/
devtmpfs	1.9G	84K	1.9G	1%	/dev
tmpfs	1.9G	0	1.9G	0%	/dev/shm
/dev/md0	30G	4.9G	24G	18%	/mnt/raid

- start all the daemons:
  - ◆ *hadoop namenode -format*
  - ◆ *./hadoop/sbin/start-all.sh*

```
ec2-user@master:~$ jps
Starting namenodes on [master]
master: starting namenode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-namenode-master.out
slave5: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave5.out
slave2: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave2.out
slave7: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave7.out
slave8: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave8.out
slave4: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave4.out
slave3: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave3.out
slave1: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave1.out
slave14: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave14.out
slave6: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave6.out
slave12: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave12.out
slave11: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave11.out
slave16: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave16.out
slave9: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave9.out
slave15: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave15.out
slave3: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave3.out
slave10: starting datanode, logging to /home/ec2-user/hadoop/logs
/hadoop-ec2-user-datanode-slave10.out

[ec2-user@slave5 ~]$ jps
6676 Jps
[ec2-user@slave5 ~]$ 
```

- ◆ *./hadoop/sbin/yarn-daemon.sh start nodemanager*

```
ec2-user@master:~$ jps
starting yarn daemons
starting resourcemanager, logging to /home/ec2-user/hadoop/logs/
yarn-ec2-user-resourcemanager-master.out
slave2: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave2.out
slave10: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave10.out
slave9: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave9.out
slave7: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave7.out
slave8: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave8.out
slave3: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave3.out
slave5: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave5.out
slave15: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave15.out
slave1: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave1.out
slave13: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave13.out
slave4: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave4.out
slave6: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave6.out
slave14: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave14.out
slave16: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave16.out
slave11: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave11.out
slave12: starting nodemanager, logging to /home/ec2-user/hadoop/
logs/yarn-ec2-user-nodemanager-slave12.out

[ec2-user@slave5 ~]$ jps
6676 Jps
[ec2-user@slave5 ~]$ 
```

◆ *jps*

The screenshot shows a Terminator window with eight terminal sessions. The sessions are arranged in two rows of four. The top row contains sessions for slave16, slave2, slave8, slave12, and slave14. The bottom row contains sessions for slave1, slave3, slave9, slave13, and slave16. Each session displays the output of the 'jps' command, which lists Java processes. The processes shown are DataNodes and NodeManagers, with IDs ranging from 5000 to 6000. The terminal window has a dark background with light-colored text. The right side of the window shows system status icons.

Session	Content
ec2-user@slave16:~	[ec2-user@master ~]\$ jps [ec2-user@master ~]\$ clear
ec2-user@slave2:~	[ec2-user@slave2 ~]\$ jps 20651 NameNode 21391 NodeManager 21618 Jps 21005 ResourceManager [ec2-user@master ~]\$
ec2-user@slave8:~	Run "sudo yum update" to apply updates. [ec2-user@slave8 ~]\$ jps 6244 Jps 6096 NodeManager 5980 DataNode [ec2-user@slave8 ~]\$
ec2-user@slave12:~	3 package(s) needed for security, out of 3 available Run "sudo yum update" to apply all updates. [ec2-user@slave12 ~]\$ clear
ec2-user@slave14:~	016.03-release-notes/ 3 package(s) needed for security, out of 3 available Run "sudo yum update" to apply all updates. [ec2-user@slave14 ~]\$ clear
ec2-user@slave1:~	[ec2-user@slave1 ~]\$ jps 6976 DataNode 7256 Jps 7092 NodeManager [ec2-user@slave1 ~]\$
ec2-user@slave3:~	[ec2-user@slave3 ~]\$ jps 6440 Jps 6291 NodeManager 6175 DataNode [ec2-user@slave3 ~]\$
ec2-user@slave9:~	[ec2-user@slave9 ~]\$ jps 5618 DataNode 5734 NodeManager 5885 Jps [ec2-user@slave9 ~]\$
ec2-user@slave13:~	[ec2-user@slave13 ~]\$ jps 5500 NodeManager 5384 DataNode 5651 Jps [ec2-user@slave13 ~]\$
ec2-user@slave16:~	[ec2-user@slave16 ~]\$ jps 5833 Jps 5565 DataNode 5681 NodeManager [ec2-user@slave16 ~]\$

- Run the ts.jar file and to take the log details run the jar command in the nohub. Following screen shot shows the starting of Map-Reduce and the command to run jar.
  - ◆ *nohup hadoop jar ts.jar TeraSort /user/dipen/input /user/dipen/output &*
  - ◆ *nano nohup.out*

## Map 0% - Reduce 0%

```
16/03/30 08:31:52 INFO client.RMProxy: Connecting to ResourceManager at master/172.31.8.62:8032
16/03/30 08:31:52 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
16/03/30 08:31:53 INFO input.FileInputFormat: Total input paths to process : 1
16/03/30 08:31:53 INFO mapreduce.JobSubmitter: number of splits:745
16/03/30 08:31:54 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1459325552741_0001
16/03/30 08:31:54 INFO impl.YarnClientImpl: Submitted application application_1459325552741_0001
16/03/30 08:31:54 INFO mapreduce.Job: The url to track the job: http://master:8088/proxy/application_1459325552741_0001/
16/03/30 08:31:54 INFO mapreduce.Job: Running job: job_1459325552741_0001
16/03/30 08:32:02 INFO mapreduce.Job: Job job_1459325552741_0001 running in uber mode : false
16/03/30 08:32:02 INFO mapreduce.Job: map 0% reduce 0%
16/03/30 08:32:55 INFO mapreduce.Job: map 8% reduce 0%
16/03/30 08:33:10 INFO mapreduce.Job: map 9% reduce 0%
16/03/30 08:33:12 INFO mapreduce.Job: map 10% reduce 0%
16/03/30 08:33:13 INFO mapreduce.Job: map 11% reduce 0%
16/03/30 08:33:25 INFO mapreduce.Job: map 12% reduce 0%
16/03/30 08:33:29 INFO mapreduce.Job: map 13% reduce 0%
16/03/30 08:33:33 INFO mapreduce.Job: map 14% reduce 0%
16/03/30 08:33:36 INFO mapreduce.Job: map 15% reduce 0%
16/03/30 08:33:38 INFO mapreduce.Job: map 16% reduce 0%
16/03/30 08:33:44 INFO mapreduce.Job: map 17% reduce 0%
16/03/30 08:33:50 INFO mapreduce.Job: map 17% reduce 1%
```

## Map 100% - Reduce 18%

```
16/03/30 08:47:02 INFO mapreduce.Job: map 95% reduce 1%
16/03/30 08:47:21 INFO mapreduce.Job: map 96% reduce 2%
16/03/30 08:47:36 INFO mapreduce.Job: map 97% reduce 2%
16/03/30 08:47:46 INFO mapreduce.Job: map 97% reduce 3%
16/03/30 08:47:53 INFO mapreduce.Job: map 98% reduce 3%
16/03/30 08:48:08 INFO mapreduce.Job: map 99% reduce 3%
16/03/30 08:48:15 INFO mapreduce.Job: map 99% reduce 4%
16/03/30 08:48:27 INFO mapreduce.Job: map 100% reduce 4%
16/03/30 08:48:43 INFO mapreduce.Job: map 100% reduce 5%
16/03/30 08:49:04 INFO mapreduce.Job: map 100% reduce 6%
16/03/30 08:49:22 INFO mapreduce.Job: map 100% reduce 7%
16/03/30 08:49:46 INFO mapreduce.Job: map 100% reduce 8%
16/03/30 08:50:10 INFO mapreduce.Job: map 100% reduce 9%
16/03/30 08:50:36 INFO mapreduce.Job: map 100% reduce 10%
16/03/30 08:51:03 INFO mapreduce.Job: map 100% reduce 11%
16/03/30 08:51:33 INFO mapreduce.Job: map 100% reduce 12%
16/03/30 08:52:03 INFO mapreduce.Job: map 100% reduce 13%
16/03/30 08:52:33 INFO mapreduce.Job: map 100% reduce 14%
16/03/30 08:53:00 INFO mapreduce.Job: map 100% reduce 15%
16/03/30 08:53:30 INFO mapreduce.Job: map 100% reduce 16%
16/03/30 08:54:00 INFO mapreduce.Job: map 100% reduce 17%
16/03/30 08:54:18 INFO mapreduce.Job: map 100% reduce 18%
```

## Map 100% - Reduce 100%

The image shows two terminal windows side-by-side. The left window displays a long log of Hadoop mapreduce jobs from March 16, 2016, at 05:08:23. The logs show numerous map and reduce tasks reaching 100% completion. The right window shows the command `df -h` being run, displaying disk usage statistics for various partitions like /dev/xvda1, /dev/tmpfs, and /dev/md0.

```

ec2-user@master:~$ 
ec2-user@master:~$ 16/03/27 05:08:23 INFO mapreduce.Job: map 100% reduce 63%
16/03/27 05:09:20 INFO mapreduce.Job: map 100% reduce 64%
16/03/27 05:10:14 INFO mapreduce.Job: map 100% reduce 65%
16/03/27 05:11:12 INFO mapreduce.Job: map 100% reduce 66%
16/03/27 05:12:06 INFO mapreduce.Job: map 100% reduce 67%
16/03/27 05:17:14 INFO mapreduce.Job: map 100% reduce 68%
16/03/27 05:17:50 INFO mapreduce.Job: map 100% reduce 69%
16/03/27 05:18:26 INFO mapreduce.Job: map 100% reduce 70%
16/03/27 05:18:59 INFO mapreduce.Job: map 100% reduce 71%
16/03/27 05:19:32 INFO mapreduce.Job: map 100% reduce 72%
16/03/27 05:20:11 INFO mapreduce.Job: map 100% reduce 73%
16/03/27 05:20:47 INFO mapreduce.Job: map 100% reduce 74%
16/03/27 05:21:23 INFO mapreduce.Job: map 100% reduce 75%
16/03/27 05:21:56 INFO mapreduce.Job: map 100% reduce 76%
16/03/27 05:22:29 INFO mapreduce.Job: map 100% reduce 77%
16/03/27 05:23:05 INFO mapreduce.Job: map 100% reduce 78%
16/03/27 05:23:42 INFO mapreduce.Job: map 100% reduce 79%
16/03/27 05:24:18 INFO mapreduce.Job: map 100% reduce 80%
16/03/27 05:24:51 INFO mapreduce.Job: map 100% reduce 81%
16/03/27 05:25:27 INFO mapreduce.Job: map 100% reduce 82%
16/03/27 05:26:04 INFO mapreduce.Job: map 100% reduce 83%
16/03/27 05:26:43 INFO mapreduce.Job: map 100% reduce 84%
16/03/27 05:27:25 INFO mapreduce.Job: map 100% reduce 85%
16/03/27 05:28:10 INFO mapreduce.Job: map 100% reduce 86%
16/03/27 05:28:55 INFO mapreduce.Job: map 100% reduce 87%
16/03/27 05:29:37 INFO mapreduce.Job: map 100% reduce 88%
16/03/27 05:30:19 INFO mapreduce.Job: map 100% reduce 89%
16/03/27 05:31:05 INFO mapreduce.Job: map 100% reduce 90%
16/03/27 05:31:47 INFO mapreduce.Job: map 100% reduce 91%
16/03/27 05:32:32 INFO mapreduce.Job: map 100% reduce 92%
16/03/27 05:33:14 INFO mapreduce.Job: map 100% reduce 93%
16/03/27 05:33:59 INFO mapreduce.Job: map 100% reduce 94%
16/03/27 05:34:44 INFO mapreduce.Job: map 100% reduce 95%
16/03/27 05:35:30 INFO mapreduce.Job: map 100% reduce 96%
16/03/27 05:36:15 INFO mapreduce.Job: map 100% reduce 97%
16/03/27 05:37:00 INFO mapreduce.Job: map 100% reduce 98%
16/03/27 05:37:42 INFO mapreduce.Job: map 100% reduce 99%
16/03/27 05:38:27 INFO mapreduce.Job: map 100% reduce 100%

```

### ■ hadoop web-interface overview :

- ◆ open web browser and write :

*ec2-54-210-242-230.compute-1.amazonaws.com:50070*

The image shows a Google Chrome browser window displaying the Hadoop NameNode web interface. The URL is `ec2-54-210-242-230.compute-1.amazonaws.com:50070/dfshealth.html#tab-overview`. The page provides an overview of the cluster, including basic information like Started, Version, and Cluster ID, and a detailed Summary section with metrics such as Configured Capacity, DFS Used, and Block Pool ID.

Started:	Wed Mar 30 08:12:21 UTC 2016
Version:	2.7.2_r1654c4fe8a74205c792ee23f548cd4604acf0e41
Compiled:	2016-01-26T00:08Z by jenkins from (detached from b165c4f)
Cluster ID:	CID-843331b1-0ddf-4b89-b3dd-d7623bd97a7
Block Pool ID:	BP-2107450688-172.31.8.62-145932379568

Summary	
Security is off.	
SafeMode is off.	
26 files and directories, 754 blocks = 780 total filesystem object(s).	
Heap Memory used 110.27 MB of 448.5 MB Heap Memory. Max Heap Memory is 889 MB.	
Non Heap Memory used 33.22 MB of 33.38 MB Committed Non Heap Memory. Max Non Heap Memory is 214 MB.	
Configured Capacity:	802.91 GB
DFS Used:	93.88 GB (11.69%)
Non DFS Used:	370.2 GB
DFS Remaining:	338.83 GB (42.2%)
Block Pool Used:	93.88 GB (11.69%)
DataNodes usages% (Min/Median/Max/stdDev):	9.11% / 11.45% / 26.96% / 4.63%
Live Nodes	15 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	0
Number of Blocks Pending Deletion	0
Block Deletion Start Time	3/30/2016, 3:12:21 AM

## ■ hadoop web-interface Live Nodes :

Datanode Information

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
slave8:50010 (172.31.10.102:50010)	2	In Service	59.43 GB	5.55 GB	48.58 GB	5.3 GB	48	5.55 GB (9.34%)	0	2.7.2
slave7:50010 (172.31.14.136:50010)	2	In Service	59.43 GB	7.31 GB	12.94 GB	39.18 GB	61	7.31 GB (12.31%)	0	2.7.2
slave12:50010 (172.31.10.70:50010)	2	In Service	59.43 GB	5.42 GB	52.23 GB	1.78 GB	44	5.42 GB (9.11%)	0	2.7.2
slave1:50010 (172.31.8.228:50010)	2	In Service	29.91 GB	4.66 GB	24.83 GB	427.36 MB	39	4.66 GB (15.59%)	0	2.7.2
slave16:50010 (172.31.11.150:50010)	2	In Service	59.43 GB	5.54 GB	11.55 GB	42.34 GB	45	5.54 GB (9.33%)	0	2.7.2
slave14:50010 (172.31.13.26:50010)	2	In Service	59.43 GB	5.8 GB	13.75 GB	39.89 GB	47	5.8 GB (9.75%)	0	2.7.2
slave3:50010 (172.31.7.100:50010)	2	In Service	29.91 GB	5.67 GB	16.86 GB	7.38 GB	47	5.67 GB (18.96%)	0	2.7.2
slave15:50010 (172.31.7.254:50010)	2	In Service	59.43 GB	6.8 GB	13.19 GB	39.44 GB	55	6.8 GB (11.45%)	0	2.7.2
slave9:50010 (172.31.13.77:50010)	2	In Service	59.43 GB	7.06 GB	13.33 GB	39.05 GB	58	7.06 GB (11.87%)	0	2.7.2
slave5:50010 (172.31.1.40:50010)	2	In Service	59.43 GB	6.05 GB	12.49 GB	40.89 GB	50	6.05 GB (10.17%)	0	2.7.2
slave2:50010 (172.31.1.54:50010)	2	In Service	29.91 GB	8.06 GB	20.9 GB	969.24 MB	66	8.06 GB (26.96%)	0	2.7.2
slave11:50010 (172.31.14.64:50010)	2	In Service	59.43 GB	5.67 GB	36.96 GB	16.8 GB	46	5.67 GB (9.54%)	0	2.7.2
slave10:50010 (172.31.9.7:50010)	2	In Service	59.43 GB	7.43 GB	51.9 GB	106.51 MB	61	7.43 GB (12.51%)	0	2.7.2
slave6:50010 (172.31.14.45:50010)	2	In Service	59.43 GB	5.93 GB	38.21 GB	15.29 GB	48	5.93 GB (9.97%)	0	2.7.2
slave13:50010 (172.31.12.177:50010)	2	In Service	59.43 GB	6.93 GB	12.31 GB	40.2 GB	56	6.93 GB (11.66%)	0	2.7.2

## ■ Valsort of the first 10 data of the output:

```
[ec2-user@master 64]$ ./valsrt /mnt/raid/op1.txt
Records: 500012708
Checksum: ee6c27245251d89
Duplicate keys: 0
SUCCESS - all records are in order
[ec2-user@master 64]$
```

```
[ec2-user@master 64]$
```

■ Valsort of the last 10 data of the output :

```
[ec2-user@master 64]$ ./valsrt /mnt/raid/op2.txt
Records: 499987291
Checksum: ee683173c0538cf
Duplicate keys: 0
SUCCESS - all records are in order
[ec2-user@master 64]$ ]
```

■ Stop all daemons :

- ◆ *./hadoop/etc/hadoop/stop-all.sh*

```
ec2-user@master:- 64x36
0.0.0.0: IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
0.0.0.0: Someone could be eavesdropping on you right now (man-in-the-middle attack)!
0.0.0.0: It is also possible that a host key has just been changed.
0.0.0.0: The fingerprint for the ECDSA key sent by the remote host is
0.0.0.0: 90:38:53:a7:d4:f3:4b:2e:db:20:e1:6c:78:37:bc:9e.
0.0.0.0: Please contact your system administrator.
0.0.0.0: Add correct host key in /home/ec2-user/.ssh/known_hosts
to get rid of this message.
0.0.0.0: Offending ECDSA key in /home/ec2-user/.ssh/known_hosts:
3
0.0.0.0: ECDSA host key for 0.0.0.0 has changed and you have requested strict checking.
0.0.0.0: Host key verification failed.
stopping yarn daemons
stopping resourcemanager
slave2: stopping nodemanager
slave7: stopping nodemanager
slave13: stopping nodemanager
slave6: stopping nodemanager
slave1: stopping nodemanager
slave3: stopping nodemanager
slave5: stopping nodemanager
slave16: stopping nodemanager
slave11: stopping nodemanager
slave15: stopping nodemanager
slave9: stopping nodemanager
slave4: no nodemanager to stop
slave10: stopping nodemanager
slave8: stopping nodemanager
slave14: stopping nodemanager
slave12: stopping nodemanager
no proxyserver to stop
[ec2-user@master ~]$ ]
```

```
ec2-user@slave5:- 80x41
[ec2-user@slave5 ~]$ jps
6511 Jps
6222 DataNode
6338 NodeManager
[ec2-user@slave5 ~]$ jps
6594 Jps
[ec2-user@slave5 ~]$ ]
```

**■ Performance Evaluation Table (10 Gb | 100 Gb):**

- Execution time and Throughput :

Data Size (GB)	Execution Time	Throughput
10 GB	920	10.87
100 GB	4266	23.44

**■ Problems Faced During the configuration and Running the program :**

- Add hadoop.tmp.dir property in the core-site.xml in the master node. If it is not added, then when the program starts mapping and reducing it will stored the output data in the RAM rather than the mounted drive (/mnt/raid). So, after running the few percent of the reducing the program stops running and throws the error no more memory space left.
- Add mapreduce.job.maps and mapreduce.job.reduces properties in the mapered-sitx.xml in the master node. If it is not added the data works on the single cpu rather than the two cpu, which will slow down the reducing task and after some percentage of reducing task stop.

## ■ Configuration files Description :

Below are the description of the files and it explains what changes have to be done in these file to go from 1 node to 16 nodes cluster.

### 1)conf/master :

This is the master file of the HDFS system, which contains the private ip of the master. It differentiate master of the HDFS system with the slaves of the system. In slave nodes this file is not present.

### 2)conf/slaves :

This is the slave file of the HDFS system, which contains the private ip of the slave. Master node's conf/slaves file contains all the private ip of the slave nodoes. While slave node's conf/slaves file contains only the private ip of itself.

To go from 1 node to 16 nodes, only change required is to add all the new slave nodes' private ip, which were not there in the 1 node cluster.

### 3)conf/core-site.xml:

This file contains the configuration settings for the HDFS file system such as I/O settings like data storage, memory input data paths which are common to HDFS and Map-Reduce.

To go from 1 node to 16 nodes, only change required is to add the temporary data storage path, because in 16 nodes the data size to process is very large. So, the RAM of the master is not able to process such amount of large data.

### 4)conf/hdfs-site.xml :

This file contains the configuration settings for HDFS daemons, namenode, secondary name node, the data node and the data replication details.

The only change required to go from 1 node to 16 nodes is to change the path value of the data node and name node.

**5)conf/mapred-site.xml :**

This file contains the configuration settings for the Map-Reduce daemons and job tracker address.

To go from 1 node to 16 nodes, the changes have to be made is in the mapreduce.job.maps and mapreduce.job.reduces with the value of no of cores in the system( here,value 2).

**■ Questions:****1) What is a Master node? What is a Slaves node?****Ans:**

Master node is the heart of the HDFS system. The main task of the Master node is to schedule the task among the slaves and divide the input data into slaves in equal amount. It also keep track of all the slaves and their work. Master is unique for the entire HDFS system.

Slave node or Data Node is the worker node running under the master node, whose work is to complete the task assigned by the Master node. It also stored the data chunk given by the master. There can be multiple slave nodes in the HDFS system.

**2) Why do we need to set unique available ports to those configuration files on a shared environment? What errors or side-effects will show if we use same port number for each user?****Ans:**

Each service is joint with the uniquely identified port in the HDFS. So, we can not start 2 different services using the same port. Two services can not work simultaneously on the single port. This kind of conflict cause error by stopping both services in the system. So, it is better to use services with their unique ports.

**3) How can we change the number of mappers and reducers from the configuration file?****Ans:**

In HDFS, there is a file called mapred-site.xml in the hadoop/etc/hadoop folder. To change the number of mappers and reducers we have change the value of the property **mapreduce.job.maps** and

**mapreduce.job.reduces** in the mapred-site.xml.

```
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.jobtracker.address</name>
<value>hdfs://master:8021</value>
</property>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
<property>
<name>mapreduce.job.maps</name>
<value>2</value>
</property>
<property>
<name>mapreduce.job.reduces</name>
<value>2</value>
</property>
</configuration>
```

**3) How can we change the number of mappers and reducers from the configuration file?**

**Ans:**

In HDFS, there is a file called mapred-site.xml in the hadoop/etc/hadoop folder. To change the number of mappers and reducers we have change the value of the property **mapreduce.job.maps** and **mapreduce.job.reduces** in the mapred-site.xml.

## **7. Spark Sort:**

### **● Spark Configuration:**

After starting the instance we have to run following commands to install the spark in the system:

- ◆ *sudo apt-get update*
  
- ◆ *sudo apt-get install openjdk-7-jdk*

Install Spark :

- ◆ *wget http://d3kbcqa49mib13.cloudfront.net/spark-1.0.2.tgz*

Install hadoop :

- ◆ *wget <http://mirrors.ocf.berkeley.edu/apache/spark/spark-1.6.1/spark-1.6.1-bin-hadoop2.6.tgz>*

install scala :

- ◆ *wget <http://www.scala-lang.org/files/archive/scala-2.10.4.tgz>*
- ◆ *sudo mkdir /usr/local/src/scala*
- ◆ *sudo tar xvf scala-2.10.4.tgz -C /usr/local/src/scala/*

Open bashrc file:

- ◆ *vi .bashrc*
- add following lines :

- ◆ *export SCALA\_HOME=/usr/local/src/scala/scala-2.10.4*
- ◆ *export PATH=\$SCALA\_HOME/bin:\$PATH*
- ◆ *. .bashrc*

Launching the Cluster :

Set environment variables for the AWS access key and secret access key:

- ◆ *export AWS\_ACCESS\_KEY\_ID=AKPAICCB6WRTXULWLCL*
- ◆ *export AWS\_SECRET\_ACCESS\_KEY\_ID=KYDz2c0xfd6DvKN/yOyN53*

Give permission to the key:

- ◆ *chmod 600 spark.pem*

Now, to run the 1 master, 1 slave and 1 master 16 slaves following steps should be followed:

### ➤ 10GB data sort(1 master, 1 slave) :

Run the “spark-ec2” file to create 1 master and 1 slave with c3.large instance type, extra data volume of 30GB and spot instance price=0.04:

- ◆ *./spark-1.6.1/ec2/spark-ec2 -k spark -i spark.pem --slaves=1 --instance-type=c3.large --ebs-vol-size=30 --spot-price=0.04 launch spark\_test*

In case of connection refuse run following command:

- ◆ *./spark-1.6.1/ec2/spark-ec2 -k spark -i spark.pem --slaves=1 --instance-type=c3.large --ebs-vol-size=30 --spot-price=0.04 launch spark\_test --resume*

After Spark finish launching cluster, the following screen will be shown:

```
ec2-52-87-185-138.compute-1.amazonaws.com
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-87-185-138.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmetad: [FAILED]
Starting GANGLIA gmetad: [OK]
Stopping httpd: [FAILED]
Starting httpd: [OK]
[timing] ganglia setup: 00h 00m 02s
Connection to ec2-52-23-161-128.compute-1.amazonaws.com closed.
Spark standalone cluster started at http://ec2-52-23-161-128.compute-1.amazonaws.com:8080
Ganglia started at http://ec2-52-23-161-128.compute-1.amazonaws.com:5080/ganglia
Done!
ubuntu@ip-172-31-45-14:~/spark-1.6.1-bin-hadoop2.6/ec2$ ls
deploy.generic lib README spark-ec2 spark_ec2.py spark.pem
ubuntu@ip-172-31-45-14:~/spark-1.6.1-bin-hadoop2.6/ec2$ cd
ubuntu@ip-172-31-45-14:~$ ls
sbt-0.13.5.deb spark-1.6.1-bin-hadoop2.6 spark.pem
scala-2.10.4.tgz spark-1.6.1-bin-hadoop2.6.tgz
ubuntu@ip-172-31-45-14:~$
```

Now, the spark setup is done. We have to do following change to hadoop's hdfs-site.xml. Following are the changes should be done in these files:

- ◆ *ephemeral/conf/hdfs-site.xml*

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

Now, copy all these hdfs changes to slave with the following command:

- ◆ *./copy-dir ~/ephemeral-hdfs/conf/*

Now, go to the hdfs file system:

- ◆ *./ephemeral-hdfs/bin/hadoop/stop\_dfs.sh*

Now, go to the spark file system:

- ◆ *./spark/sbin/stop\_all.sh*

Now, go to the spark file system:

- ◆ *./spark/sbin/start\_all.sh*

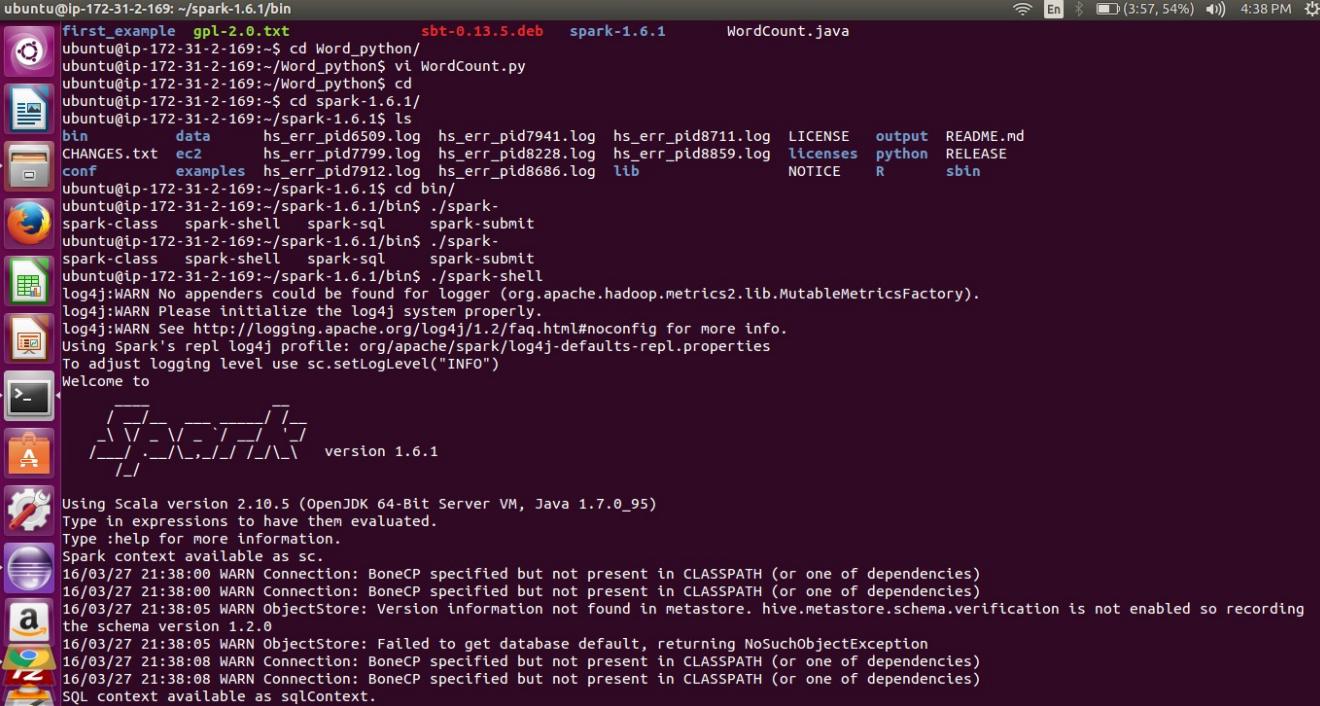
Now, go to the hdfs file system:

- ◆ *./ephemeral-hdfs/bin/hadoop/.start\_dfs.sh*

Now, check of running daemons with following command:

- ◆ *jps*

Now, open spark shell to run the spark program :



```

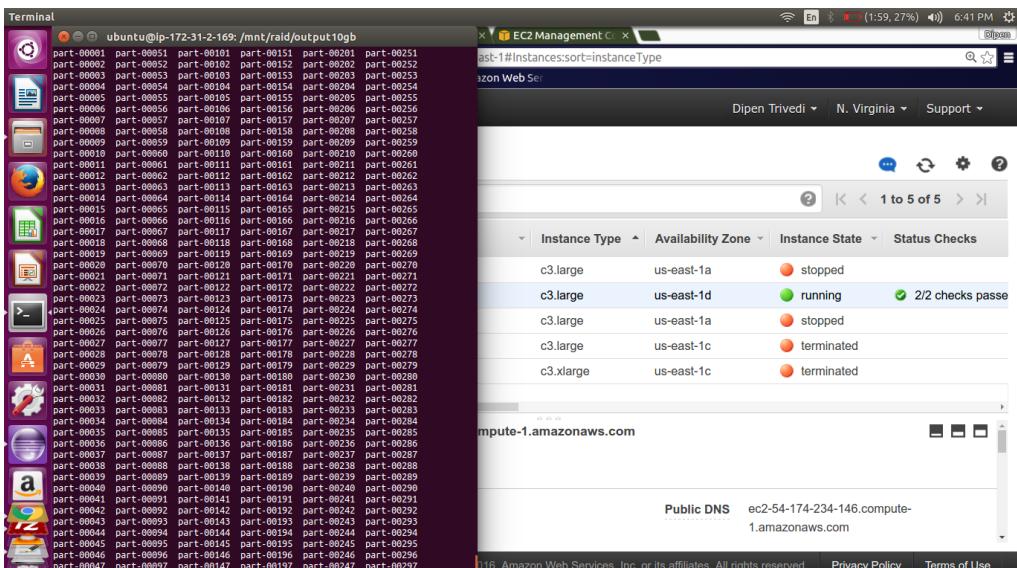
ubuntu@ip-172-31-2-169: ~/spark-1.6.1/bin
first_example gpl-2.0.txt      sbt-0.13.5.deb  spark-1.6.1      WordCount.java
ubuntu@ip-172-31-2-169:~$ cd Word_python/
ubuntu@ip-172-31-2-169:~/Word_python$ vi WordCount.py
ubuntu@ip-172-31-2-169:~/Word_python$ cd
ubuntu@ip-172-31-2-169:~$ cd spark-1.6.1/
ubuntu@ip-172-31-2-169:~/spark-1.6.1$ ls
bin          data      hs_err_pid6509.log  hs_err_pid7941.log  hs_err_pid8711.log  LICENSE  output  README.md
CHANGES.txt  ec2       hs_err_pid7799.log  hs_err_pid8228.log  hs_err_pid8859.log  licenses  python  RELEASE
conf        examples  hs_err_pid7912.log  hs_err_pid8686.log  lib           NOTICE   R      sbin
ubuntu@ip-172-31-2-169:~/spark-1.6.1$ cd bin/
ubuntu@ip-172-31-2-169:~/spark-1.6.1/bin$ ./spark-
spark-class  spark-shell  spark-sql  spark-submit
ubuntu@ip-172-31-2-169:~/spark-1.6.1/bin$ ./spark-
spark-class  spark-shell  spark-sql  spark-submit
ubuntu@ip-172-31-2-169:~/spark-1.6.1/bin$ ./spark-shell
log4j:WARN No appenders could be found for logger (org.apache.hadoop.metrics2.lib.MutableMetricsFactory).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Using Spark's repl log4j profile: org/apache/spark/log4j-defaults-repl.properties
To adjust logging level use sc.setLogLevel("INFO")
Welcome to
    _____
   /     \
  /       \
 /  _   _ \
 \  \ / / /
  \_ \_ \
    / /
   version 1.6.1
Using Scala version 2.10.5 (OpenJDK 64-Bit Server VM, Java 1.7.0_95)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
16/03/27 21:38:00 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
16/03/27 21:38:05 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
16/03/27 21:38:05 WARN ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not enabled so recording the schema version 1.2.0
16/03/27 21:38:05 WARN ObjectStore: Failed to get database default, returning NoSuchObjectException
16/03/27 21:38:08 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
16/03/27 21:38:08 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
SQL context available as sqlContext.

```

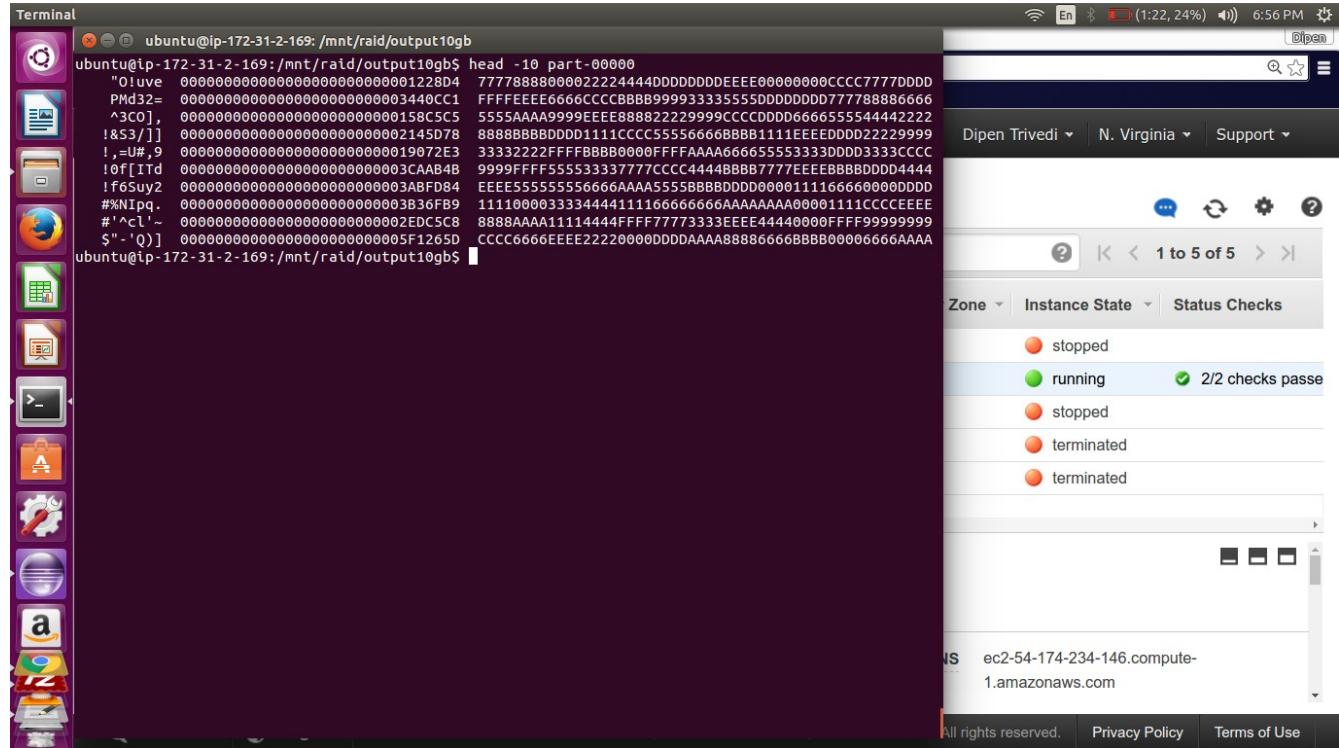
Now, run following spark program in the shell :

- ◆ *TerasortFile = sc.textFile("hdfs://PUBLIC DNS : 9000/user/hadoop/input/input\_100GB.txt")*
- ◆ *TeraSortObj = TerasortFile.flatMap(lambda line:line.split("\n")).map(lambda dicto: (str(dicto[:10]),str(dicto[10:])).sortByKey().map(lambda (a,b) : a+b)*
- ◆ *TeraSortObj.saveAsTextFile("hdfs://Public DNS :9000/user/hadoop/output\_100GB")*

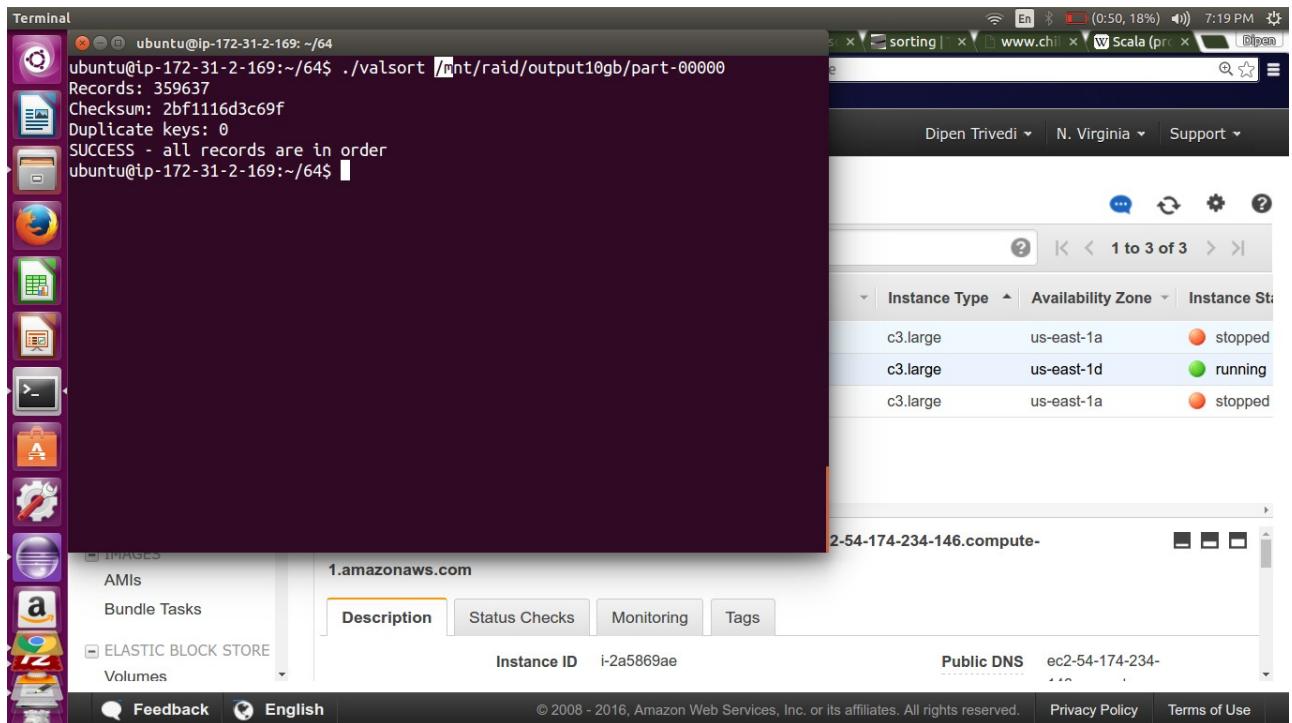
Following screen shot shows the number of files created after the program:



- ◆ First 10 records of Part-00000 file :



#### ◆ Valsort of Part-00000 file :



## ➤ 100GB data sort(1 master, 16 slaves) :

Run the “spark-ec2” file to create 1 master and 16 slave with c3.large instance type, extra data volume of 30GB and spot instance price=0.04:

- ◆ ./spark-1.6.1/ec2/spark-ec2 -k spark -i spark.pem --slaves=16 --instance-type=c3.large --ebs-vol-size=30 --spot-price=0.04 launch spark test

In case of connection refuse run following command:

- ◆ `./spark-1.6.1/ec2/spark-ec2 -k spark -i spark.pem --slaves=16 --instance-type=c3.large --ebs-vol-size=30 --spot-price=0.04 launch spark_test --resume`

After Spark finish launching clusters, the following screen will be shown:

```
ubuntu@ip-172-31-22-196: ~/spark-1.6.1/ec2
[ec2-54-165-130-83.compute-1.amazonaws.com
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-88-250-38.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-88-90-228.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-90-181-179.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-152-70-129.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-88-199-192.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-90-103-51.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-165-1-50.compute-1.amazonaws.com closed.
$ Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-152-121-13.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-174-38-17.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-174-16-43.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-90-64-19.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-90-244-152.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-85-192-107.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-207-223-96.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-90-235-122.compute-1.amazonaws.com closed.
```

Now, the spark setup is done. We have to do following change to hadoop's hdfs-site.xml. Following are the changes should be done in these files:

- ◆ *ephemeral/conf/hdfs-site.xml*

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

Now, copy all these hdfs chages to slave with the following command:

- ◆ *./copy-dir ~/ephemeral-hdfs/conf/*

Now, go to the hdfs file system:

- ◆ *./ephemeral-hdfs/bin/hadoop/stop\_dfs.sh*

Now, go to the spark file system:

- ◆ *./spark/sbin/stop\_all.sh*

Now, go to the spark file system:

- ◆ *./spark/sbin/start\_all.sh*

Now, go to the hdfs file system:

- ◆ *./ephemeral-hdfs/bin/hadoop/.start\_dfs.sh*

```
ubuntu@ip-172-31-22-196:~/spark-1.6.1/ec2          ubuntu@ip-172-31-22-196:~/spark-1.6.1/ec2 144x40

[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-54-165-1-50.compute-1.amazonaws.com,172.31.28.118' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-54-152-121-13.compute-1.amazonaws.com,172.31.24.59' (ECDSA) to the list of known hosts.
Warning: Permanently added 'ec2-54-152-121-13.compute-1.amazonaws.com,172.31.24.59' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-54-174-38-17.compute-1.amazonaws.com,172.31.23.210' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-54-174-16-43.compute-1.amazonaws.com,172.31.24.24' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-52-90-64-19.compute-1.amazonaws.com,172.31.16.203' (ECDSA) to the list of known hosts.
Warning: Permanently added 'ec2-52-90-64-19.compute-1.amazonaws.com,172.31.16.203' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-52-90-244-152.compute-1.amazonaws.com,172.31.26.49' (ECDSA) to the list of known hosts.
Warning: Permanently added 'ec2-52-90-244-152.compute-1.amazonaws.com,172.31.26.49' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-54-85-192-107.compute-1.amazonaws.com,172.31.25.228' (ECDSA) to the list of known hosts.
Warning: Permanently added 'ec2-54-85-192-107.compute-1.amazonaws.com,172.31.25.228' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-52-207-223-96.compute-1.amazonaws.com,172.31.27.233' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-52-90-235-122.compute-1.amazonaws.com,172.31.26.44' (ECDSA) to the list of known hosts.
Warning: Permanently added 'ec2-52-90-235-122.compute-1.amazonaws.com' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
Warning: Permanently added 'ec2-54-165-130-83.compute-1.amazonaws.com,172.31.29.217' (ECDSA) to the list of known hosts.
Warning: Permanently added 'ec2-54-165-130-83.compute-1.amazonaws.com' (ECDSA) to the list of known hosts.
[  0%] id_rsa                                          100% 1679   1.6KB/s  00:00
[timing] rsync /root/spark-ec2: 00h 00m 02s
Running setup-slave on all cluster nodes to mount filesystems, etc...
```

Now, check of running daemons with following command:

◆ *jps*

Now, open spark shell to run the spark program :

```
ubuntu@ip-172-31-2-169:~/spark-1.6.1/bin
first_example gpl-2.0.txt          sbt-0.13.5.deb   spark-1.6.1      WordCount.java
ubuntu@ip-172-31-2-169:~$ cd Word_python/
ubuntu@ip-172-31-2-169:~/Word_python$ vi WordCount.py
ubuntu@ip-172-31-2-169:~/Word_python$ cd
ubuntu@ip-172-31-2-169:~$ cd spark-1.6.1/
ubuntu@ip-172-31-2-169:~/spark-1.6.1$ ls
bin          data      hs_err_pid6509.log  hs_err_pid7941.log  hs_err_pid8711.log  LICENSE  output  README.md
CHANGES.txt  ec2       hs_err_pid7799.log  hs_err_pid8228.log  hs_err_pid8859.log  licenses  python  RELEASE
conf        examples  hs_err_pid7912.log  hs_err_pid8686.log  lib                  NOTICE    R       sbin
ubuntu@ip-172-31-2-169:~/spark-1.6.1$ cd bin/
ubuntu@ip-172-31-2-169:~/spark-1.6.1/bin$ ./spark-
spark-class  spark-shell  spark-sql  spark-submit
ubuntu@ip-172-31-2-169:~/spark-1.6.1/bin$ ./spark-
spark-class  spark-shell  spark-sql  spark-submit
ubuntu@ip-172-31-2-169:~/spark-1.6.1/bin$ ./spark-shell
log4j:WARN No appenders could be found for logger (org.apache.hadoop.metrics2.lib.MutableMetricsFactory).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Using Spark's repl log4j profile: org/apache/spark/log4j-defaults-repl.properties
To adjust logging level use sc.setLogLevel("INFO")
Welcome to
    / \ \ / - \ \ \ - / \ / \ / \
    \ \ / . \ \ \ - / \ / \ / \
    / / \ / \ / \ / \ / \ / \ / \
version 1.6.1

Using Scala version 2.10.5 (OpenJDK 64-Bit Server VM, Java 1.7.0_95)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
16/03/27 21:38:00 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
16/03/27 21:38:00 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
16/03/27 21:38:05 WARN ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not enabled so recording the schema version 1.2.0
16/03/27 21:38:05 WARN ObjectStore: Failed to get database default, returning NoSuchObjectException
16/03/27 21:38:08 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
16/03/27 21:38:08 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
SQL context available as sqlContext.
```

Now, run following spark program in the shell :

- ◆ *TerasortFile = sc.textFile("hdfs://PUBLIC DNS : 9000/user/hadoop/input/input\_100GB.txt")*

```
TeraSortObj = TerasortFile.flatMap(lambda line:line.split("\n"))
.map(lambda dicto:
(str(dicto[:10]),str(dicto[10:])).sortByKey().map(lambda (a,b) : a+b)
```

```
TeraSortObj.saveAsTextFile("hdfs://Public DNS :9000/user/hadoop/output_100GB")
```

- ◆ Following Web interface shows the job task in progress:

The screenshot shows a Google Chrome browser window titled "PySparkShell - Spark Jobs - Google Chrome". The address bar displays the URL "ec2-52-201-249-65.compute-1.amazonaws.com:4040/jobs/". The main content area is titled "Spark Jobs" and shows the following information:

- Total Uptime: 6.1 min
- Scheduling Mode: FIFO
- Active Jobs: 1
- Event Timeline (link)
- Active Jobs (1)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	sortByKey at <stdin>:1	2016/03/28 06:33:53	1.4 min	0/1	128/745

The left sidebar contains icons for various Spark components: Spark Shell, Spark History Server, Spark Streaming, Spark MLlib, Spark GraphX, and Spark SQL.

- ◆ Following Web interface shows the 100% sorting of the job task :

The screenshot shows the PySparkShell application running in Google Chrome. The main content area displays the "Completed Jobs (2)" section. It lists two completed jobs:

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
1	sortByKey at <stdin>:1	2016/03/28 06:41:11	8.0 min	1/1	745/745
0	sortByKey at <stdin>:1	2016/03/28 06:33:53	7.3 min	1/1	745/745

The sidebar on the left contains various icons for different application features. At the bottom of the browser window, there is a download bar showing "gensort-lin....tar.gz" and a "Show all downloads..." link.

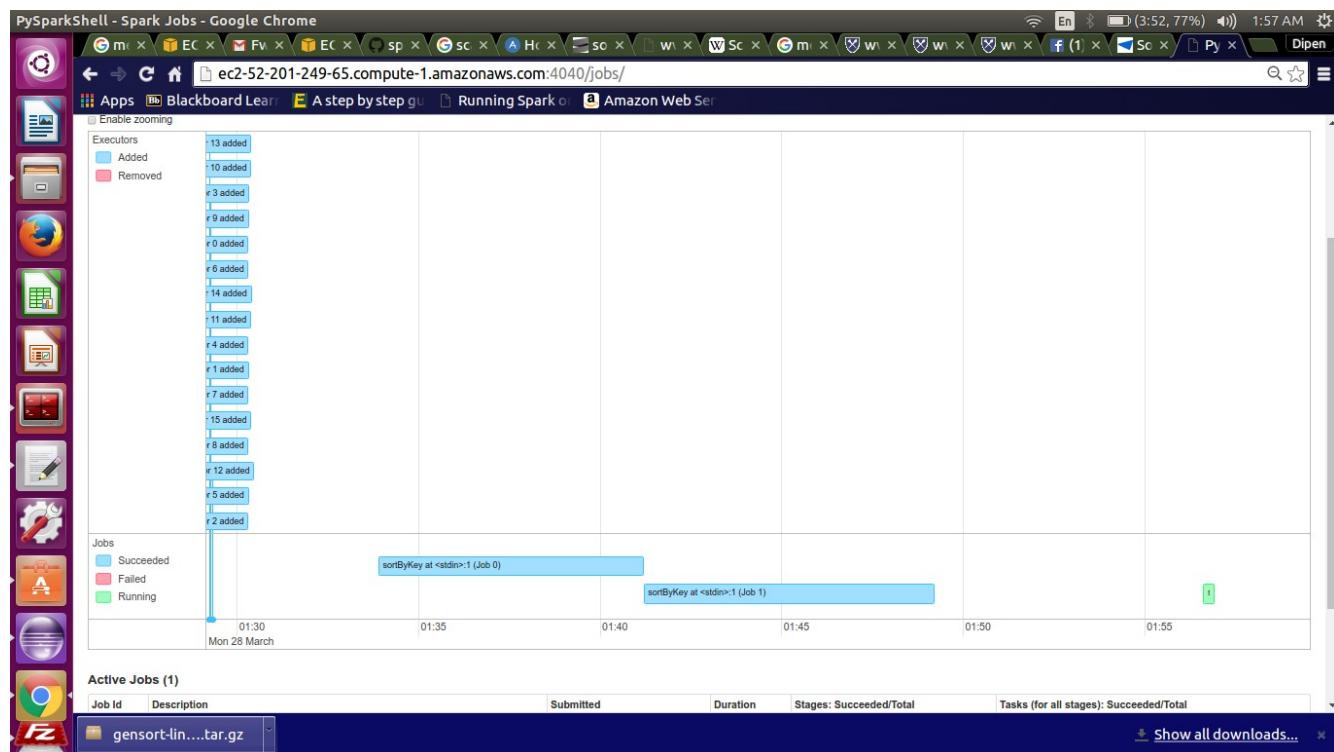
- ◆ Following Web interface shows writing in the file job task in progress:

The screenshot shows the PySparkShell application running in Google Chrome. The main content area displays the "Active Jobs (1)" section. It lists one active job:

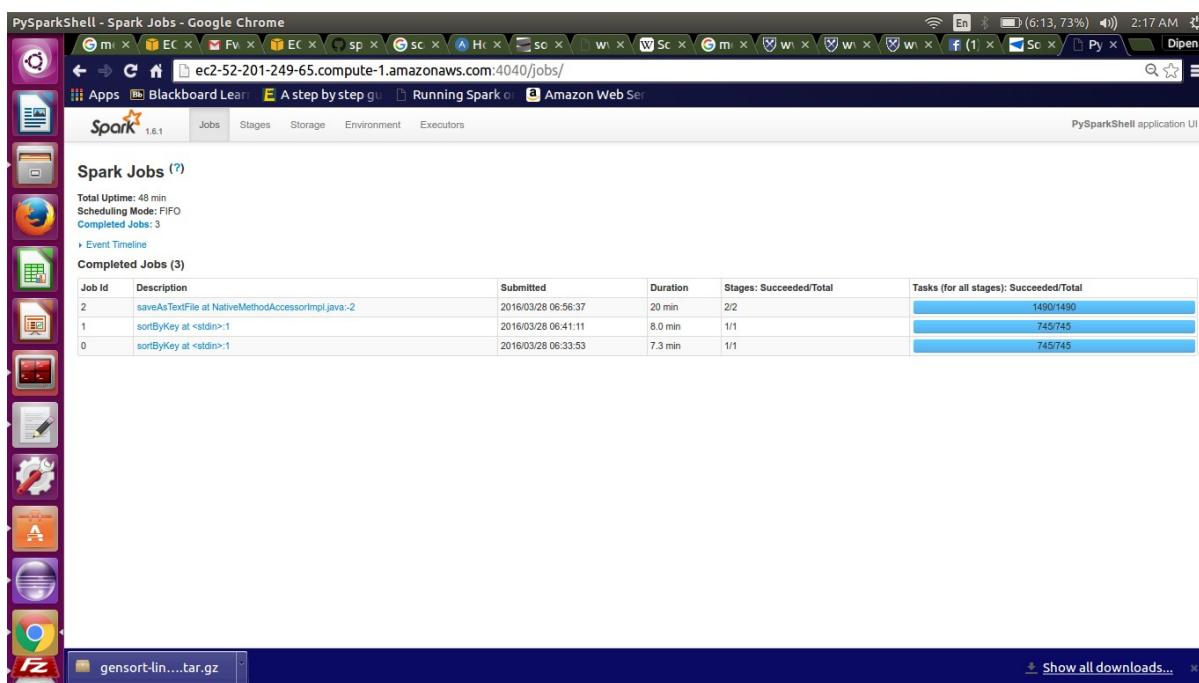
Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
2	saveAsTextFile at NativeMethodAccessorImpl.java:-2	2016/03/28 06:56:37	19 s	0/2	0/1490

Below the active job, the "Completed Jobs (2)" section is shown, which lists the same two completed jobs as in the previous screenshot. The sidebar on the left contains various icons for different application features.

- ◆ Following Web interface shows the job task time line in progress:



- ◆ Following Web interface shows the 100% job done and time taken to finish the job task:



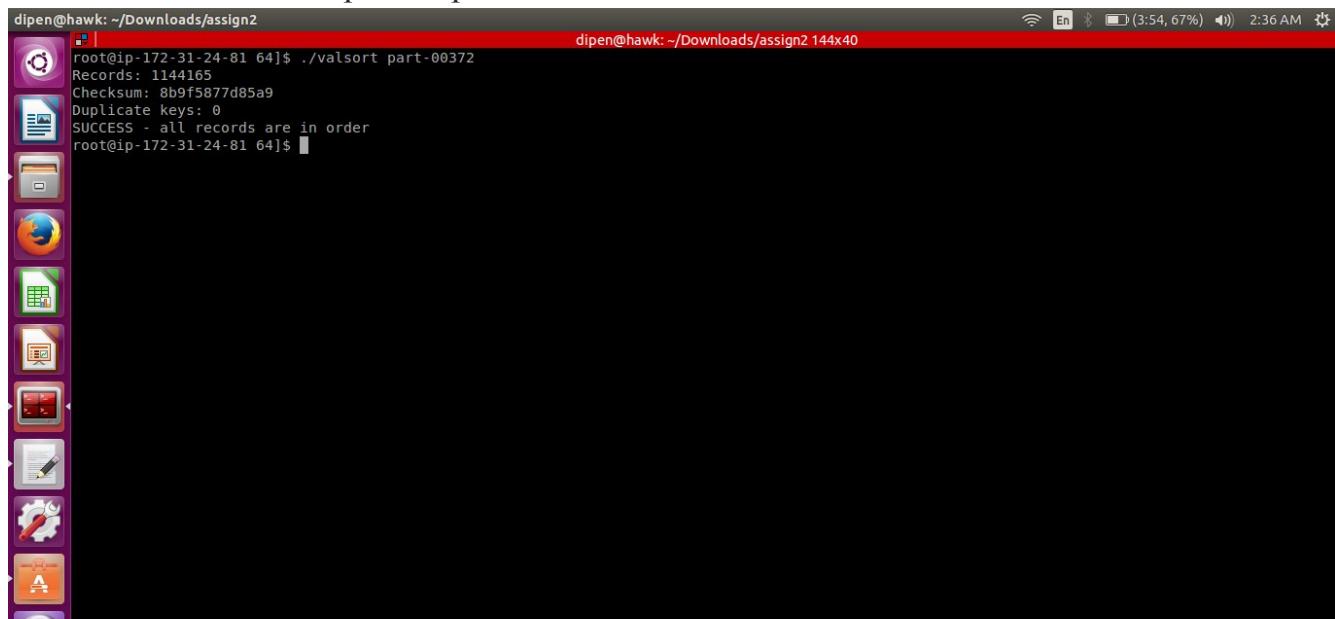
- ◆ Following screen shot shows the total 744 files created after the Job done :

```
dipen@hawk: ~/Downloads/assign2
dipen@hawk: ~/Downloads/assign2 144x40
-rw-r--r-- 1 root supergroup 102222252 2016-03-28 07:15 /user/root/output_100gb.txt/part-00706
-rw-r--r-- 1 root supergroup 101472327 2016-03-28 07:15 /user/root/output_100gb.txt/part-00707
-rw-r--r-- 1 root supergroup 141059259 2016-03-28 07:15 /user/root/output_100gb.txt/part-00708
-rw-r--r-- 1 root supergroup 110276892 2016-03-28 07:15 /user/root/output_100gb.txt/part-00709
-rw-r--r-- 1 root supergroup 94730328 2016-03-28 07:15 /user/root/output_100gb.txt/part-00710
-rw-r--r-- 1 root supergroup 190837449 2016-03-28 07:15 /user/root/output_100gb.txt/part-00711
-rw-r--r-- 1 root supergroup 172395333 2016-03-28 07:15 /user/root/output_100gb.txt/part-00712
-rw-r--r-- 1 root supergroup 124293708 2016-03-28 07:15 /user/root/output_100gb.txt/part-00713
-rw-r--r-- 1 root supergroup 120305097 2016-03-28 07:15 /user/root/output_100gb.txt/part-00714
-rw-r--r-- 1 root supergroup 107904258 2016-03-28 07:15 /user/root/output_100gb.txt/part-00715
-rw-r--r-- 1 root supergroup 79425027 2016-03-28 07:15 /user/root/output_100gb.txt/part-00716
-rw-r--r-- 1 root supergroup 127654461 2016-03-28 07:15 /user/root/output_100gb.txt/part-00717
-rw-r--r-- 1 root supergroup 78449877 2016-03-28 07:15 /user/root/output_100gb.txt/part-00718
-rw-r--r-- 1 root supergroup 128487051 2016-03-28 07:15 /user/root/output_100gb.txt/part-00719
-rw-r--r-- 1 root supergroup 132851565 2016-03-28 07:15 /user/root/output_100gb.txt/part-00720
-rw-r--r-- 1 root supergroup 116209764 2016-03-28 07:15 /user/root/output_100gb.txt/part-00721
-rw-r--r-- 1 root supergroup 135995667 2016-03-28 07:15 /user/root/output_100gb.txt/part-00722
-rw-r--r-- 1 root supergroup 150924114 2016-03-28 07:15 /user/root/output_100gb.txt/part-00723
-rw-r--r-- 1 root supergroup 147515445 2016-03-28 07:15 /user/root/output_100gb.txt/part-00724
-rw-r--r-- 1 root supergroup 143295075 2016-03-28 07:15 /user/root/output_100gb.txt/part-00725
-rw-r--r-- 1 root supergroup 77077440 2016-03-28 07:15 /user/root/output_100gb.txt/part-00726
-rw-r--r-- 1 root supergroup 124562493 2016-03-28 07:15 /user/root/output_100gb.txt/part-00727
-rw-r--r-- 1 root supergroup 163677096 2016-03-28 07:15 /user/root/output_100gb.txt/part-00728
-rw-r--r-- 1 root supergroup 147236958 2016-03-28 07:15 /user/root/output_100gb.txt/part-00729
-rw-r--r-- 1 root supergroup 92219985 2016-03-28 07:15 /user/root/output_100gb.txt/part-00730
-rw-r--r-- 1 root supergroup 143139942 2016-03-28 07:15 /user/root/output_100gb.txt/part-00731
-rw-r--r-- 1 root supergroup 122086563 2016-03-28 07:15 /user/root/output_100gb.txt/part-00732
-rw-r--r-- 1 root supergroup 117310545 2016-03-28 07:15 /user/root/output_100gb.txt/part-00733
-rw-r--r-- 1 root supergroup 181297889 2016-03-28 07:15 /user/root/output_100gb.txt/part-00734
-rw-r--r-- 1 root supergroup 111680316 2016-03-28 07:15 /user/root/output_100gb.txt/part-00735
-rw-r--r-- 1 root supergroup 148061331 2016-03-28 07:15 /user/root/output_100gb.txt/part-00736
-rw-r--r-- 1 root supergroup 166205358 2016-03-28 07:15 /user/root/output_100gb.txt/part-00737
-rw-r--r-- 1 root supergroup 113725557 2016-03-28 07:15 /user/root/output_100gb.txt/part-00738
-rw-r--r-- 1 root supergroup 176472549 2016-03-28 07:15 /user/root/output_100gb.txt/part-00739
-rw-r--r-- 1 root supergroup 117385587 2016-03-28 07:15 /user/root/output_100gb.txt/part-00740
-rw-r--r-- 1 root supergroup 108249669 2016-03-28 07:15 /user/root/output_100gb.txt/part-00741
-rw-r--r-- 1 root supergroup 185911506 2016-03-28 07:15 /user/root/output_100gb.txt/part-00742
-rw-r--r-- 1 root supergroup 113241645 2016-03-28 07:15 /user/root/output_100gb.txt/part-00743
-rw-r--r-- 1 root supergroup 114428952 2016-03-28 07:15 /user/root/output_100gb.txt/part-00744
root@ip-172-31-24-81 ~$
```

- ◆ Valsort of the output file part-00000:

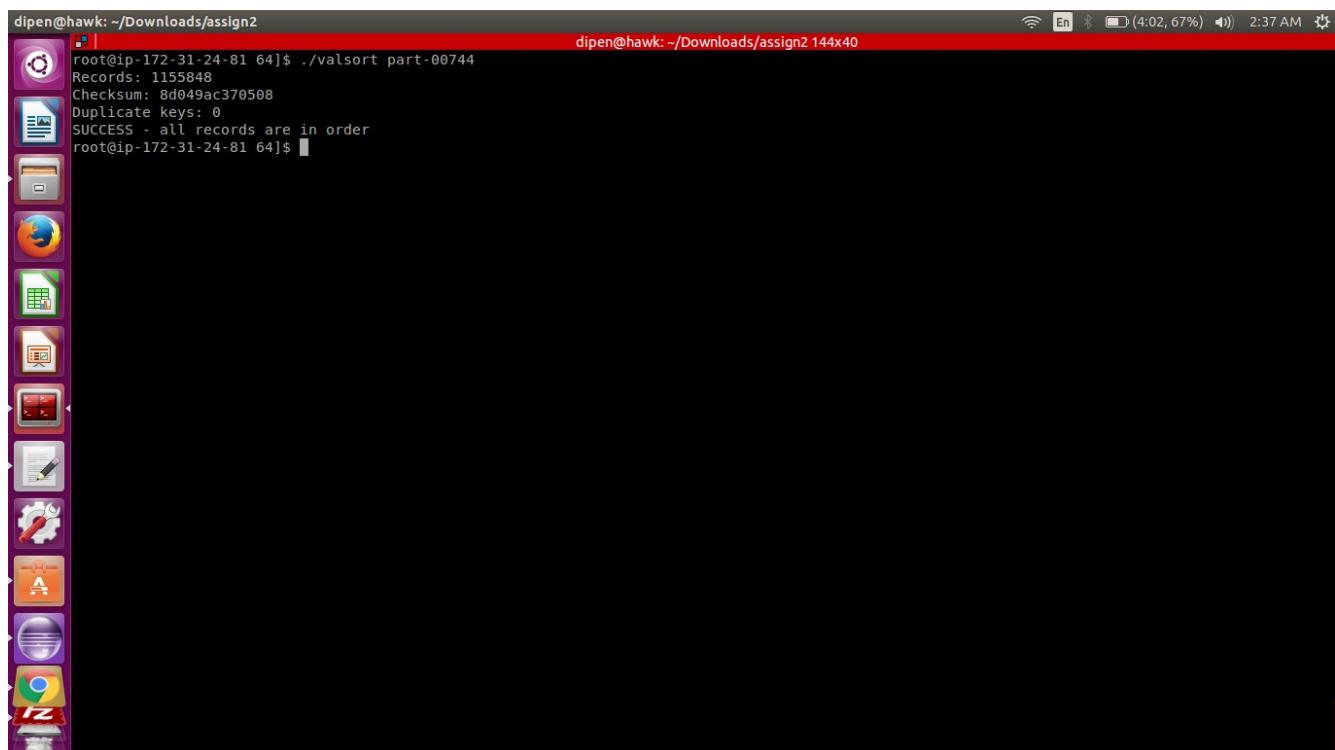
```
dipen@hawk: ~/Downloads/assign2
dipen@hawk: ~/Downloads/assign2 144x40
Root@ip-172-31-24-81 64]$ ./valsor part-00000
Records: 170316
Checksum: d00172f4a600f
Duplicate keys: 0
SUCCESS - all records are in order
root@ip-172-31-24-81 64]$
```

◆ Valsort of the output file part-00372:



```
dipen@hawk: ~/Downloads/assign2
[1] | dipen@hawk: ~/Downloads/assign2 144x40
root@ip-172-31-24-81 64]$ ./valsrt part-00372
Records: 1144165
Checksum: 8b0f5877d85a9
Duplicate keys: 0
SUCCESS - all records are in order
root@ip-172-31-24-81 64]$
```

◆ Valsort of the output file part-00744:



```
dipen@hawk: ~/Downloads/assign2
[1] | dipen@hawk: ~/Downloads/assign2 144x40
root@ip-172-31-24-81 64]$ ./valsrt part-00744
Records: 1155848
Checksum: 8d049ac370508
Duplicate keys: 0
SUCCESS - all records are in order
root@ip-172-31-24-81 64]$
```

**■ Performance Evaluation Table (10GB | 100GB) :**

Data size (GB)	Execution Time (seconds)	Throughput (MB/sec)
10	710	14.08
100	1640	60.59

**■ Problems Faced during the Configuration and Running :**

- In the **/hadoop/bin/hdfs-site.xml** change the property **dfs.replication** with default value of **3** to **1**. Because of this default value 3 replication of the data while reducing was 3 times, which was resulting in the 3 times more time to reduce the data. By changing it to one stops the data replication.

## **8. Shared Memory Sort:**

The shared memory program is created in java to perform the replication tasks of Mapping and Reducing done by the hadoop and spark. The program is performed on the 10GB data. Following is the Methodology/flow of the program:

### **➤ Methodology :**

- ◆ Created input of size 10GB and call `inputFilePartition()` method to split the input into the key and value.
- ◆ Then the function will divide the input into small chunks and store the chunks into the separate temporary files.
- ◆ Created MergeSort() to sort the chunks respect to the keys and store them in the temporary files. This fuction will be called within the Thread.run() method.
- ◆ Multiple threads(1,2,4,8) are created to run the MergeSort(). For example, for two threads, the divided chunks will be sorted in the 2 separate parts and then they will be sorted in parallel.
- ◆ Store the temporary files's addresses into the ArrayList.
- ◆ Created a function `FileWrite()` to write the chunks into the temporary files.
- ◆ Created the function `outputTempFilesMerging()` to merge the sorted chunks.
- ◆ Before merging the chunks, we have to take the first line of the chunks and sort them. Take the sorted key, fetch the value according to the key from the hash map and write that value into the Output file.

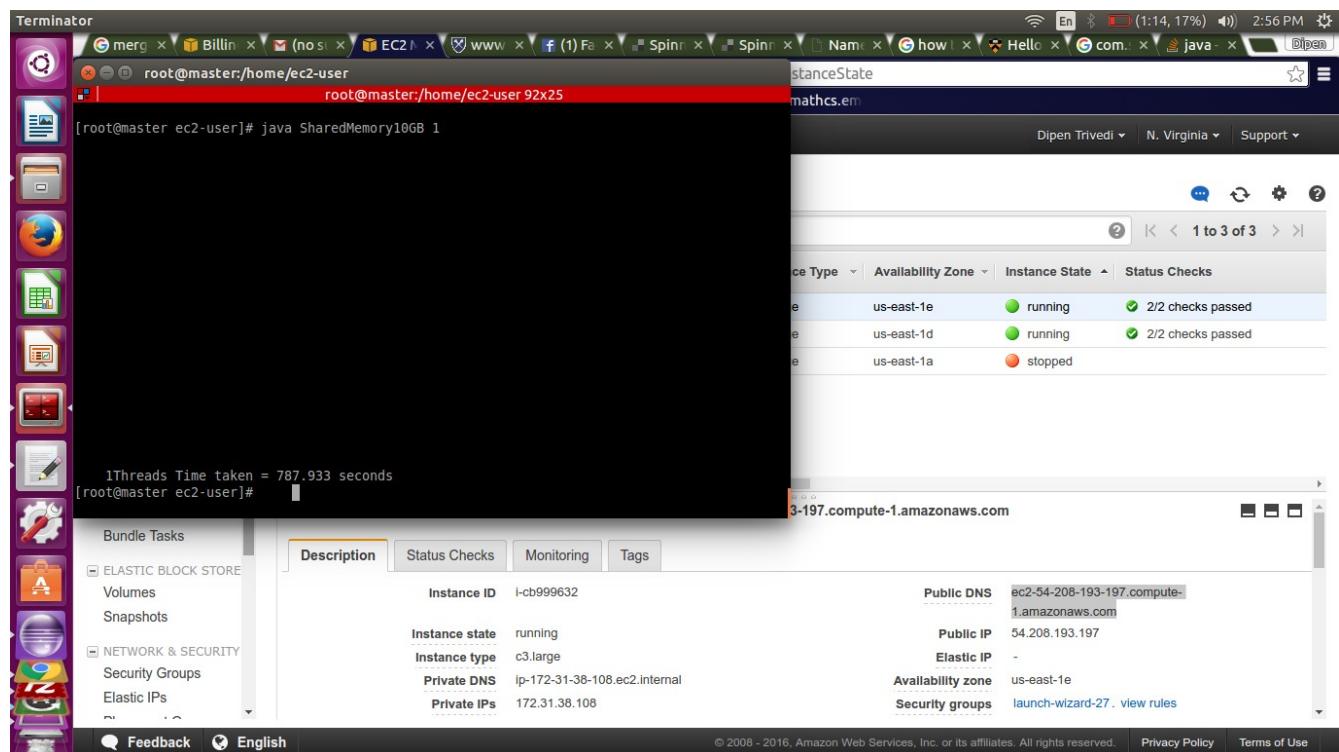
## ➤ Program Flow :

- ◆ Use the configured instance(Page -) as mentioned above.
- ◆ Compile the java file.  

$$\textit{javac SharedMemory.java}$$
- ◆ Run the java file SharedMemory file.  

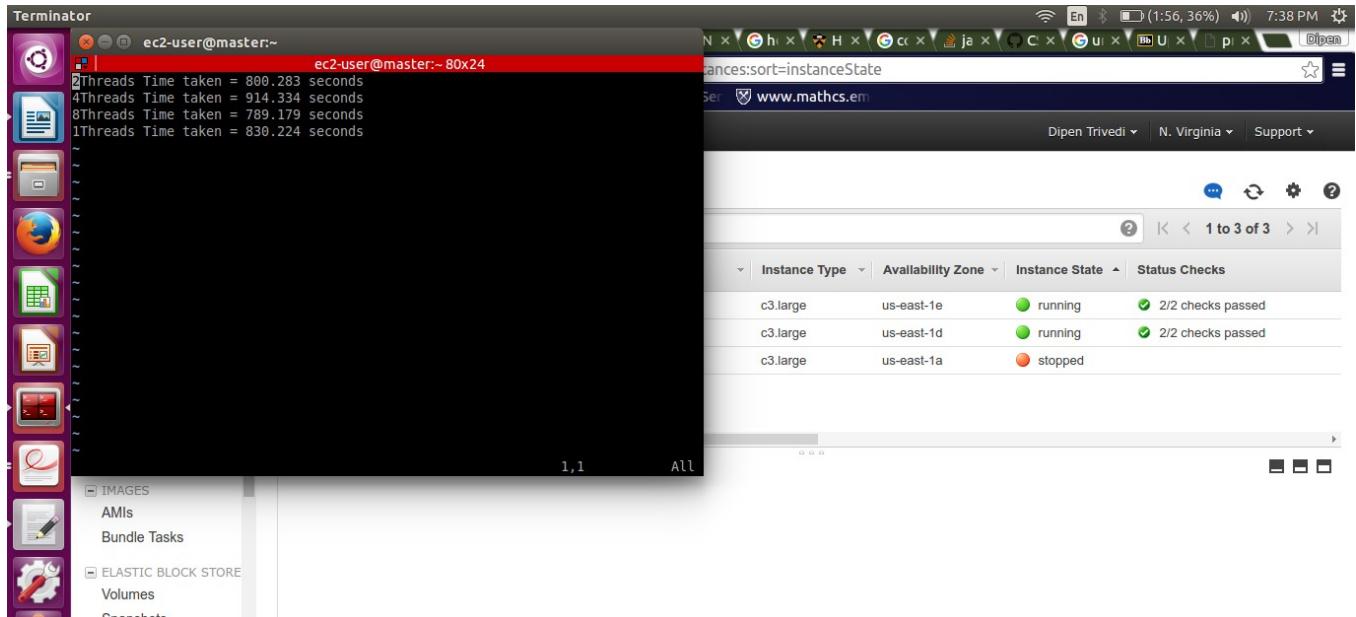
$$\textit{java SharedMemory 1} >> \textit{output.txt}$$

### ◆ Shared Memory 1-thread(10Gb data) :



- ◆ Now, open the output.txt

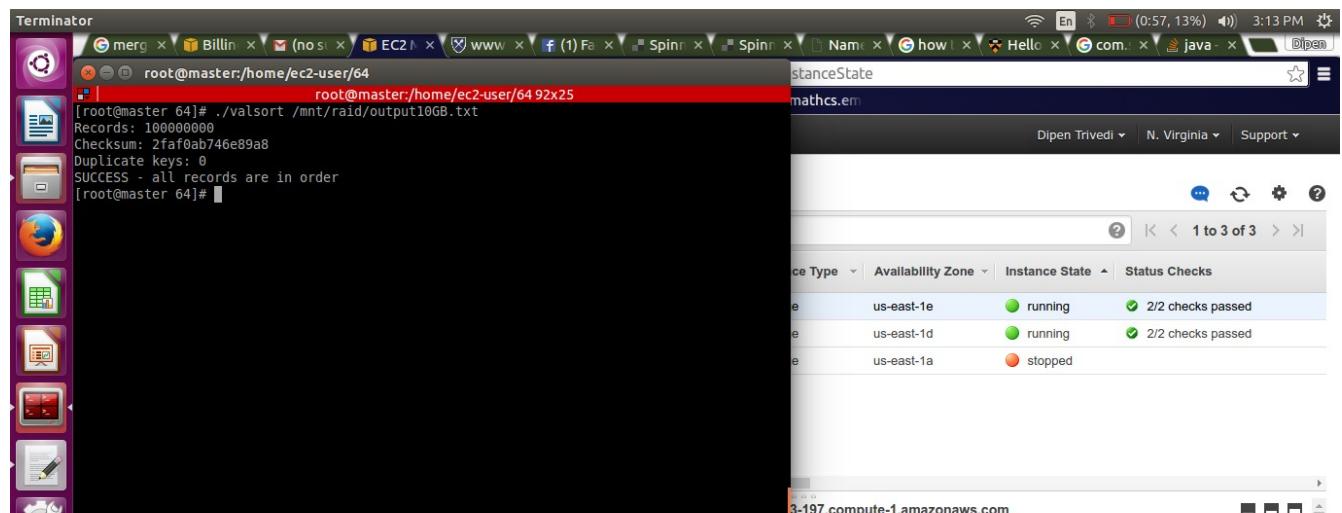
*vi output.txt*



- ◆ Now, run the valsrt on the output file.

*.valsrt /mnt/raid/output10GB.txt*

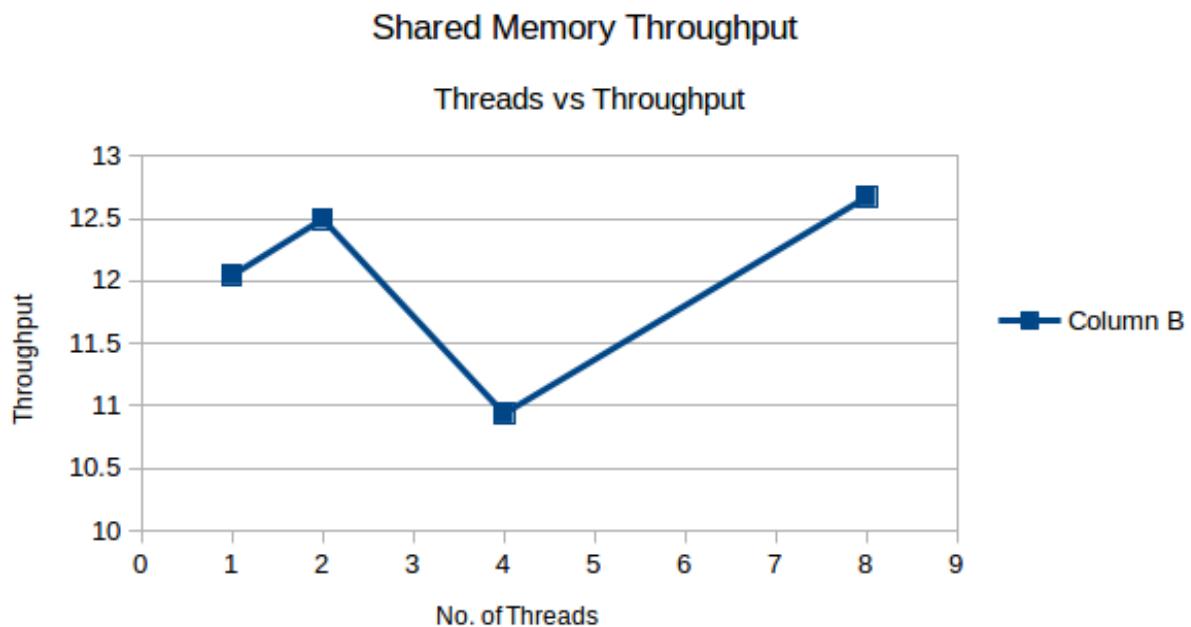
- ◆ Shared Memory Valsort(10Gb data) :



➤ Performance Evaluation with Tables and Graph (1 GB data) :

◆ Shared Memory : Throughput Table

Nubmer of Threads	Throughput
1	12.044
2	12.495
4	10.936
8	12.671

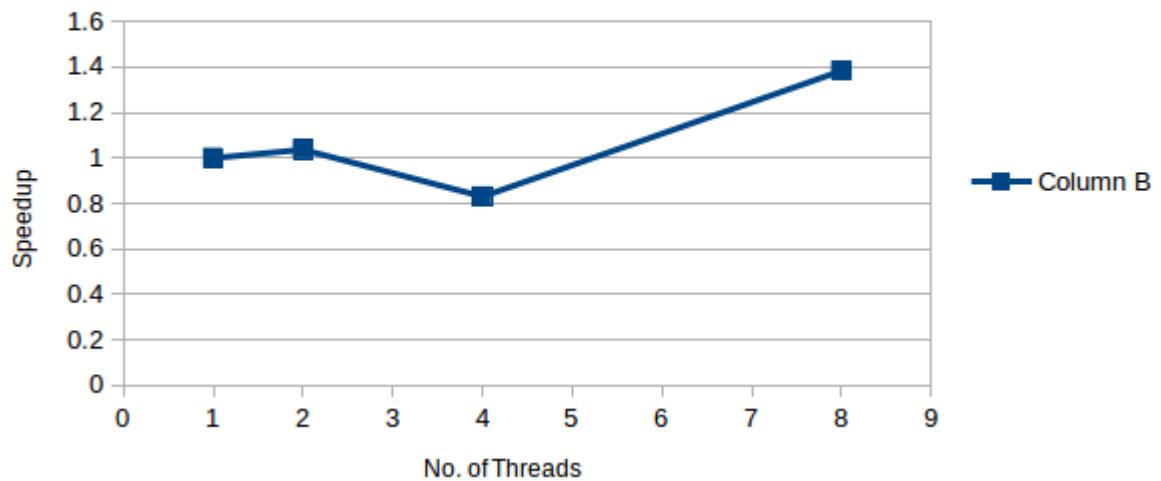


◆ Speedup Table with respect to Execution Time of Thread 1:

Nubmer of Threads	Time	Speedup
1	830.224	1
2	800.283	0.96
4	914.334	1.1
8	789.179	0.95

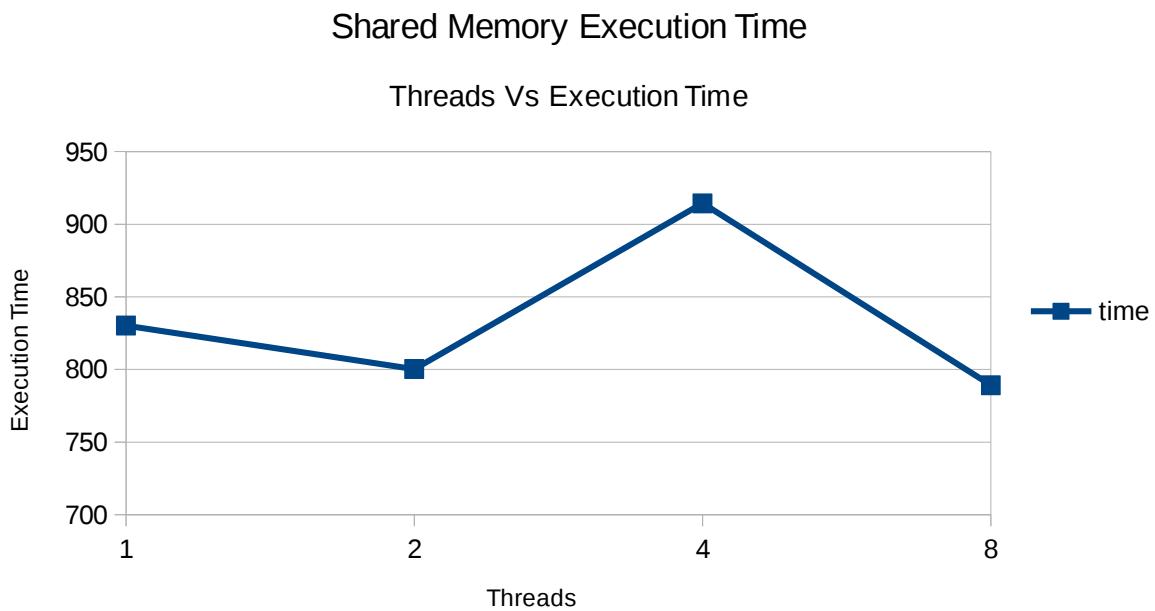
Shared Memory Speedup

Speedup vs Threads



◆ Execution Time Table (10 GB Data Size):

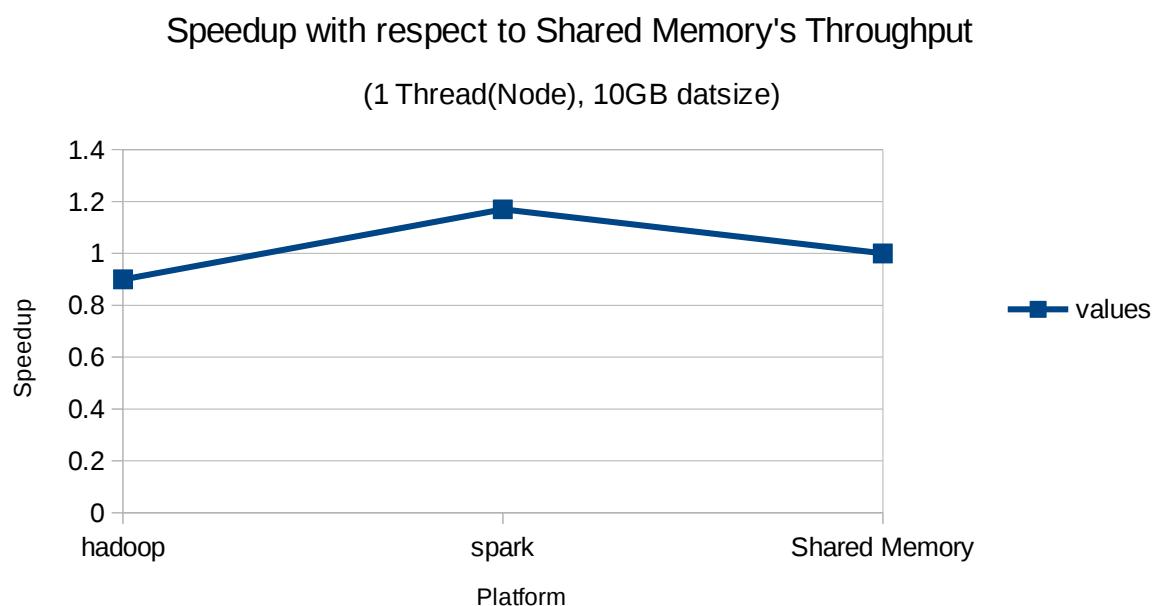
Nubmer of Threads	Execution Time (sec)
1	830.224
2	800.283
4	914.334
8	789.179



## 9. Performance Evaluation (Hadoop-Spark-Shared Memory) :

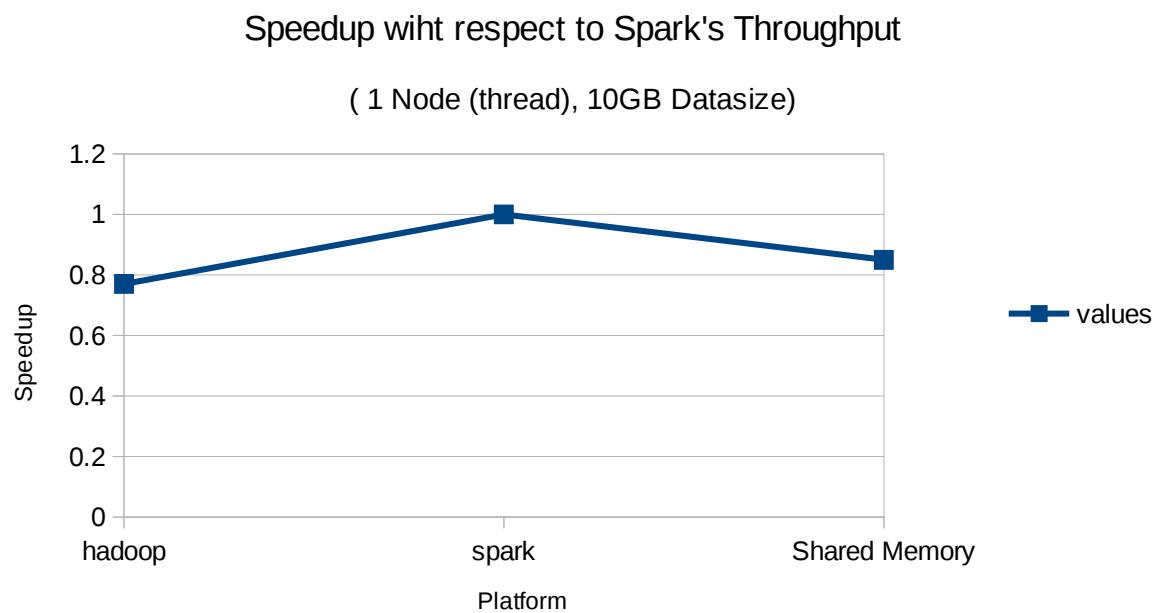
- Speedup with respect to Shared-Memory's Throughput :  
(1 Thread(node), 10GB )

Platform	Execution Time (sec)	Throughput	Speedup
hadoop	920	10.87	0.9
spark	710	14.08	1.17
Shared Memory	830.224	12.04	1



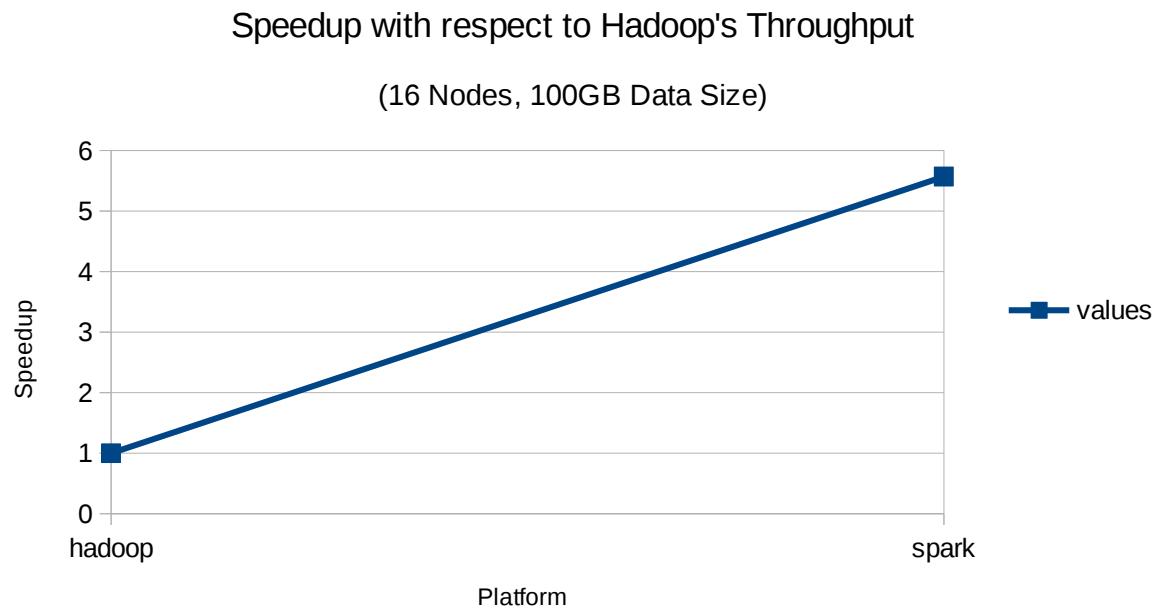
➤ Speedup with respect to Spark's Throughput :  
(1 Thread(node), 10GB )

Platform	Execution Time (sec)	Throughput	Speedup
hadoop	920	10.87	0.77
spark	710	14.08	1
Shared Memory	830.224	12.044	0.85



➤ Speedup with respect to Hadoop's Throughput :  
(16 nodes, 100GB )

Platform	Execution Time (sec)	Throughput	Speedup
hadoop	4266	10.87	1
spark	1640	60.59	5.57



**■ Question :****1) Which seems to be best at 1 node scale?****Ans :**

Spark performs best at 1 node compare to hadoop and shared memory which can be seen from the shared memory graph.

**2) How about 16 nodes?****Ans :**

From the speedup comparison graph of hadoop vs spark for 16 node, it is clear that spark is 5.57 time faster than the hadoop.

**3) Can you predict which would be best at 100 node scale?****Ans :**

From the above comparisons I can say that spark will perform better for the 100 node cluster, because spark uses the intermediate cashed data. Only problem will be the memory storing.

**4) How about 1000 node scales?****Ans :**

I guess it depend on what scale(cost or time) we are considering. If we consider cost than definitely spark is a bad option as we will have storage issue.

But if consider time than spark is much much faster compare to the hadoop. So, it depends on the user's perspective.

## 10. Conclusion :

By the end of this programing assignment and the above graphs, it was concluded that the shared memory for 1 thread take lesser time compare to Hadoop for (1 master, 1 slave) and more time than Spark (1 master, 1 slave). The performance of spark is better than the performance of hadoop in the Amazon AWS EC-2 instances.

We used d2.xlarge, c3.large and c3.4xlarge during this assignment. Which helped us to know more about these instances. We learned about the spot instances and on-demand instances. How and when to use EBS storage. I

In this assignment, we learned how to configure hadoop and spark on the single and multiple node cluster. There is no doubt that without script to perform such configuration is very hard as well as tedious specially for 16 node configuration. I made a script for hadoop which unmount the mounted drive, mounts the unmounted drive, changes the hosts and slaves and formats the whole HDFS namenode. But I was not able to make this script run for all the multiple node with a single run. I had to go to each node and run this bash file.

Yes, It was a lot of work that I had to for the configuration, but at the end I learned how the hadoop and spark works and how to manage the storage in the cloud.

## 11. REFERENCES:

- [1] AMAZON-EC2 VIRTUAL SERVER HOSTING
- [2] [www.Eduonix.com/hadoop-configuration](http://www.Eduonix.com/hadoop-configuration)
- [3] [www.geeksforgeeks.org/mergeSort](http://www.geeksforgeeks.org/mergeSort)
- [4] <http://spark.apache.org/downloads.html>
- [5] <http://spark.apache.org/docs/latest/cluster-overview.html>
- [6] spark/configuration
- [7] <http://hadoop.apache.org/>