List of Lab Exercises

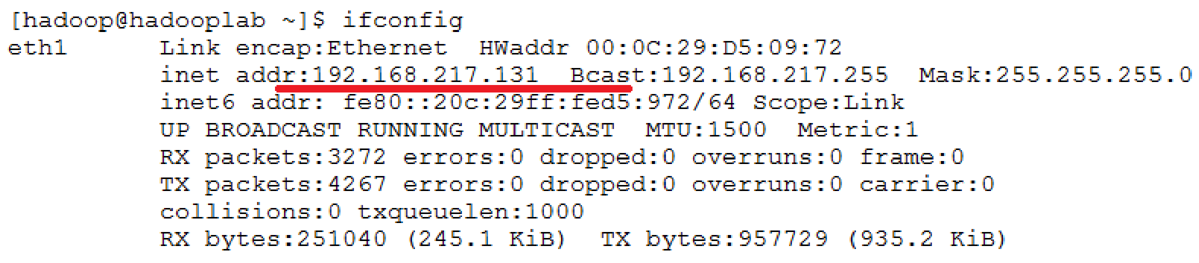
1. Lab Details

VM is running CentOS 6.2

User Name: hadoop (Password will be provided by instructor)

a. Get your VM's IP address

Run ifconfig and make note of you ip address



```
[hadoop@hadooplab ~]$ ifconfig
eth1      Link encap:Ethernet  HWaddr 00:0C:29:D5:09:72
          inet addr:192.168.217.131  Bcast:192.168.217.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fed5:972/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3272 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4267 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:251040 (245.1 KiB)  TX bytes:957729 (935.2 KiB)
```

b. Configure your VM's hosts file

sudo vi /etc/hosts

Change the following ip address to the one obtained in the previous step

192.168.217.131   hadooplab.bigdataleap.com   hadooplab

Save and exit the file. And verify if the settings are working file by running the following command.

ping   hadooplab.bigdataleap.com

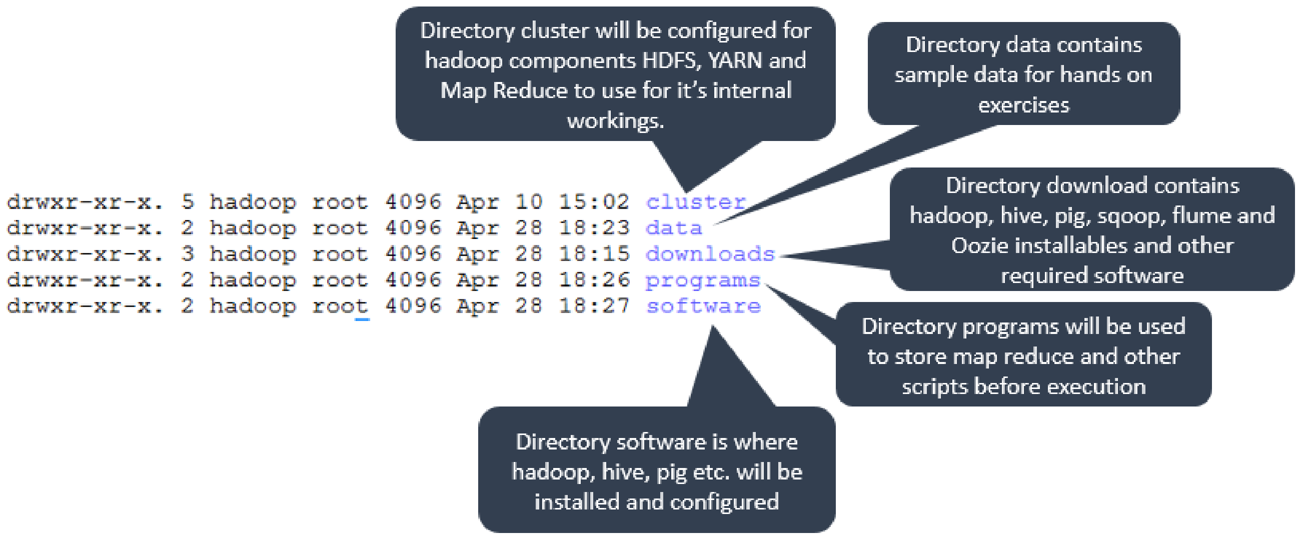
If you are getting reply from the VM, then it is configured properly.

c. Know the directories available in the VM for hands on exercises

Go to lab directory is available in /home/hadoop and list the directories available inside it.

cd /home/hadoop/lab

/home/hadoop/lab contains the following directories and will be used for the following purposes.



```
drwxr-xr-x. 5 hadoop root 4096 Apr 10 15:02 cluster
drwxr-xr-x. 2 hadoop root 4096 Apr 28 18:23 data
drwxr-xr-x. 3 hadoop root 4096 Apr 28 18:15 downloads
drwxr-xr-x. 2 hadoop root 4096 Apr 28 18:26 programs
drwxr-xr-x. 2 hadoop root 4096 Apr 28 18:27 software
```

Directory cluster will be configured for hadoop components HDFS, YARN and Map Reduce to use for it's internal workings.
Directory data contains sample data for hands on exercises
Directory download contains hadoop, hive, pig, sqoop, flume and Oozie installables and other required software
Directory programs will be used to store map reduce and other scripts before execution
Directory software is where hadoop, hive, pig etc. will be installed and configured

2. Setting up Hadoop 2.0 pseudo-distributed cluster

All directory paths are under home directory ***/home/hadoop***

d. Untar Hadoop jar file

- Go to lab/software

  cd /home/hadoop/lab/software
- Untar Hadoop files into software folder

  tar -xvf /home/hadoop/lab/downloads/hadoop-2.3.0.tar.gz
- *Browse through the directories and check which subdirectory contains what files*

e. Set up .bash_profile

- Open .bash_profile file under home directory

  cd /home/hadoop

  vi .bash_profile

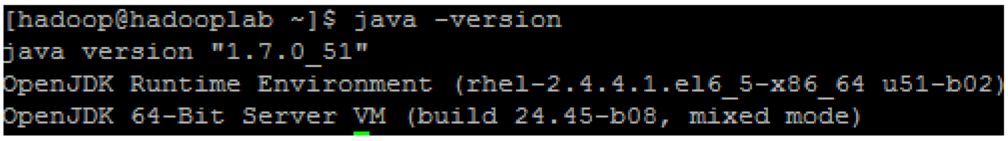
  Enter the following settings (This is already configured in the .bash_profile)

```
export JAVA_HOME=/usr/lib/jvm/jre-1.7.0-openjdk.x86_64
export HADOOP_INSTALL=/home/hadoop/lab/software/hadoop-2.3.0
export HADOOP_COMMON_HOME=/home/hadoop/lab/software/hadoop-2.3.0
export HADOOP_HDFS_HOME=/home/hadoop/lab/software/hadoop-2.3.0
export HADOOP_MAPRED_HOME=/home/hadoop/lab/software/hadoop-2.3.0
export HADOOP_YARN_HOME=/home/hadoop/lab/software/hadoop-2.3.0
export HADOOP_COMMON_LIB_NATIVE_DIR=/home/hadoop/lab/software/hadoop-2.3.0/lib/native
export HADOOP_OPTS=" -Djava.library.path=/home/hadoop/lab/software/hadoop-2.3.0/lib"
export JAVA_LIBRARY_PATH=$JAVA_LIBRARY_PATH:/home/hadoop/lab/software/hadoop-2.3.0/lib/native
export HADOOP_CONF_DIR=/home/hadoop/lab/software/hadoop-2.3.0/etc/hadoop
export YARN_CONF_DIR=$HADOOP_CONF_DIR
export PATH=$PATH:$HADOOP_INSTALL/bin
```

- Save and exit .bash_profile
- run following command

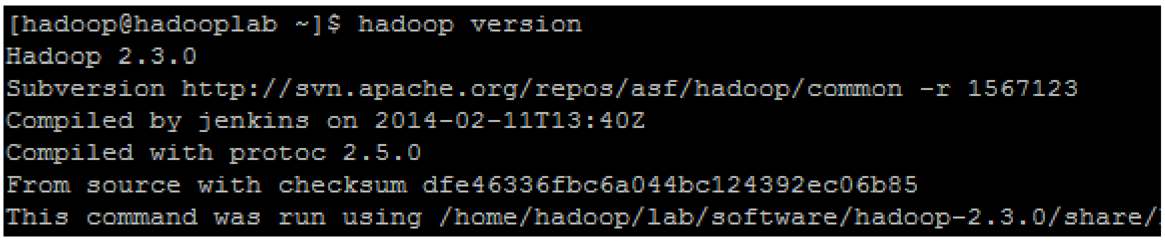
  . .bash_profile
- Verify whether variable are defined or not by typing ***export*** at command prompt
- Check the following versions

  java –version



```
[hadoop@hadooplab ~]$ java -version
java version "1.7.0_51"
OpenJDK Runtime Environment (rhel-2.4.4.1.el6_5-x86_64 u51-b02)
OpenJDK 64-Bit Server VM (build 24.45-b08, mixed mode)
```

hadoop version



```
[hadoop@hadooplab ~]$ hadoop version
Hadoop 2.3.0
Subversion http://svn.apache.org/repos/asf/hadoop/common -r 1567123
Compiled by jenkins on 2014-02-11T13:40Z
Compiled with protoc 2.5.0
From source with checksum dfe46336fbc6a044bc124392ec06b85
This command was run using /home/hadoop/lab/software/hadoop-2.3.0/share/
```

f. Configuring pseudo-distributed mode

Go to conf directory of hadoop installation folder

cd /home/hadoop/lab/software/hadoop-2.3.0/etc/hadoop

The following files are available in the reference folder of the lab distribution files on your windows or mac machine.

- Modify core-site.xml

```
<configuration>
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://hadooplab.bigdataleap.com:8020/</value>
</property>
</configuration>
```

- Modify hdfs-site.xml

```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.blocksize</name>
<value>67108864</value>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:///home/hadoop/lab/cluster/hdfs/nn</value>
</property>
<property>
<name>fs.checkpoint.dir</name>
<value>file:///home/hadoop/lab/cluster/hdfs/snn</value>
</property>
<property>
<name>dfs.namenode.checkpoint.period</name>
<value>3600</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>file:///home/hadoop/lab/cluster/hdfs/dn</value>
</property>
<property>
<name>dfs.namenode.secondary.http-address</name>
<value>hadooplab.bigdataleap.com:50090</value>
</property>
</configuration>
```

- Modify yarn-site.xml

```
<configuration>
  <property>
    <name>yarn.resourcemanager.address</name>
    <value>hadooplab.bigdataleap.com:8032</value>
  </property>
  <property>
    <name>yarn.resourcemanager.webapp.address</name>
    <value>hadooplab.bigdataleap.com:8088</value>
  </property>
  <property>
    <name>yarn.nodemanager.local-dirs</name>
    <value>/home/hadoop/lab/cluster/yarn/local</value>
  </property>
  <property>
    <name>yarn.nodemanager.remote-app-log-dir</name>
    <value>/home/hadoop/lab/cluster/yarn/remote</value>
  </property>
  <property>
    <name>yarn.nodemanager.log-dirs</name>
    <value>/home/hadoop/lab/cluster/yarn/logs</value>
  </property>
  <property>
    <name>yarn.nodemanager.resource.memory-mb</name>
    <value>3072</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
  <property>
    <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
  <property>
    <name>yarn.scheduler.maximum-allocation-mb</name>
    <value>3072</value>
  </property>
  <property>
    <name>yarn.scheduler.minimum-allocation-mb</name>
    <value>300</value>
  </property>
  <property>
   <name>yarn.nodemanager.vmem-check-enabled</name>
   <value>false</value>
  </property>
  <property>
   <name>yarn.log.server.url</name>
   <value>http://hadooplab.bigdataleap.com:19888/jobhistory/logs</value>
  </property>
  <property>
    <name>yarn.nodemanager.vmem-pmem-ratio</name>
    <value> 4 </value>
  </property>
</configuration>
```

- Modify mapred-site.xml

```
<configuration>
<property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
</property>
<property>
    <name>mapreduce.cluster.local.dir</name>
    <value>/home/hadoop/lab/cluster/mr/local</value>
</property>
<property>
    <name>mapreduce.map.memory.mb</name>
    <value>300</value>
</property>
<property>
    <name>mapreduce.reduce.memory.mb</name>
    <value>300</value>
</property>
<property>
    <name>mapreduce.map.java.opts</name>
    <value>-Xmx300m</value>
</property>
<property>
    <name>mapreduce.reduce.java.opts</name>
    <value>-Xmx300m</value>
</property>
<property>
    <name>mapreduce.jobhistory.webapp.address</name>
    <value>hadooplab.bigdataleap.com:19888</value>
</property>
<property>
<property>
    <name>mapreduce.map.log.level</name>
    <value>INFO</value>
</property>
<property>
    <name>mapreduce.reduce.log.level</name>
    <value>INFO</value>
</property>
<property>
    <name>yarn.app.mapreduce.am.resource.mb</name>
    <value>300</value>
</property>
<property>
        <name>mapreduce.cluster.administrators</name>
        <value>hadoop</value>
</property>
<property>
        <name>mapreduce.reduce.log.level</name>
        <value>INFO</value>
</property>
<property>
        <name>mapreduce.map.log.level</name>
        <value>INFO</value>
</property>
</configuration>
```

g. Copy the 64 bit libraries

- Copy the 64 bit native libraries
  Go to the following directory
  cd /home/hadoop/lab/downloads/lib64bit/

cp libhadoop.so.1.0.0 $HADOOP_INSTALL/lib/native/

cp libhdfs.so.0.0.0 $HADOOP_INSTALL/lib/native/

h. Configure JAVA_HOME

- Go to ***/home/hadoop/lab/software/hadoop-2.3.0/etc/hadoop*** directory

  export JAVA_HOME=/usr/lib/jvm/jre-1.7.0-openjdk.x86_64

  Enter the above line at the beginning of all the following files:
  - hadoop-env.sh
  - mapred-env.sh
  - yarn-env.sh

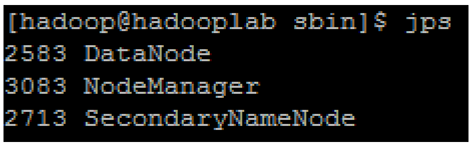
i. Format the namenode

- Enter the following command at prompt
  hdfs namenode –format
- Go to /home/hadoop/lab/cluster/hdfs/nn/current directory and verify whether all files have been created.
  - fsimage (file system image) and it's md5 file (fingerprint)
  - VERSION (contains unique cluster, layout version and other details…)

```
[hadoop@hadooplab hadoop]$ cd /home/hadoop/lab/cluster/hdfs/nn/current/
[hadoop@hadooplab current]$ ls -l
total 16
-rw-r--r--. 1 hadoop root 218 Apr 29 13:58 fsimage_0000000000000000000
-rw-r--r--. 1 hadoop root  62 Apr 29 13:58 fsimage_0000000000000000000.md5
-rw-r--r--. 1 hadoop root   2 Apr 29 13:58 seen_txid
-rw-r--r--. 1 hadoop root 207 Apr 29 13:58 VERSION
```

j. Start HDFS and YARN services

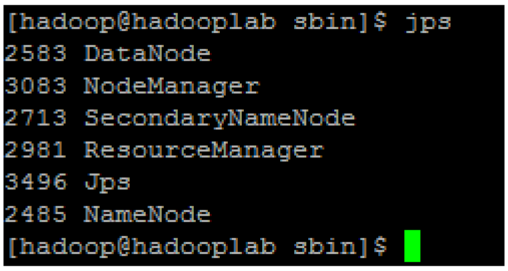
- Go to ***/home/hadoop/lab/software/hadoop-2.3.0/sbin*** directory and type the following command
  ./start-dfs.sh
  ## Note: verify if all the following three processes have started by typing ***jps*** command



```
[hadoop@hadooplab sbin]$ jps
2583 DataNode
3083 NodeManager
2713 SecondaryNameNode
```

- And then type the following command
  ./start-yarn.sh

- Run jps and verify if all the following processes are running



```
[hadoop@hadooplab sbin]$ jps
2583 DataNode
3083 NodeManager
2713 SecondaryNameNode
2981 ResourceManager
3496 Jps
2485 NameNode
[hadoop@hadooplab sbin]$
```

- If all five processes are running, then hadoop is up and running
- Run the history server, which will provide information about completed jobs
  Go to ***/home/hadoop/lab/software/hadoop-2.3.0/sbin*** directory and type the following command

  ./mr-jobhistory-daemon.sh start historyserver

  And run jps to confirm if the history server is started or not.

```
[hadoop@hadooplab sbin]$ ./mr-jobhistory-daemon.sh start historyserver
starting historyserver, logging to /home/hadoop/lab/software/hadoop-2.3.0
bigdataleap.com.out
[hadoop@hadooplab sbin]$ jps
3165 DataNode
3286 SecondaryNameNode
5546 Jps
5513 JobHistoryServer
3076 NameNode
3560 ResourceManager
3655 NodeManager
```

3. HDFS Lab

a. Verify what all files are available in the hdfs file system

hadoop fs –ls /

b. Copy files into HDFS

- Create the following HDFS directories

```
hadoop fs    –mkdir    /lab
hadoop fs    –mkdir    /lab/mr
hadoop fs    –mkdir    /lab/hive
hadoop fs    –mkdir    /lab/pig
hadoop fs    –mkdir    /lab/sqoop
```

- Check directories in HDFS

  hadoop   fs -ls /lab

- ***Copy file****s from linux directory* ***/home/hadoop/lab/data*** to HDFS directory /lab/mr

```
hadoop  fs  -copyFromLocal    /home/hadoop/lab/data/txns     /lab/mr
hadoop  fs  -copyFromLocal    /home/hadoop/lab/data/custs    /lab/mr
hadoop  fs   -copyFromLocal   /home/hadoop/lab/data/words    /lab/mr/

hadoop  fs  –ls  /lab/mr/
```

- Go to the following directory on the linux machine
  cd /home/hadoop/lab/cluster/hdfs/dn/current/BP-*/current/finalized
  and verify the blocks have been created.

c. HDFS Filesystem statistics

hdfs dfsadmin –report

Gives you detailed report of the hdfs system including

- total capacity allocated, used, available
- no of files, block

d. Checking health of files in HDFS

Gives you detailed report of hdfs files (All files or a specific files)

hdfs fsck /

hdfs fsck /lab/mr/txns -files -blocks -locations

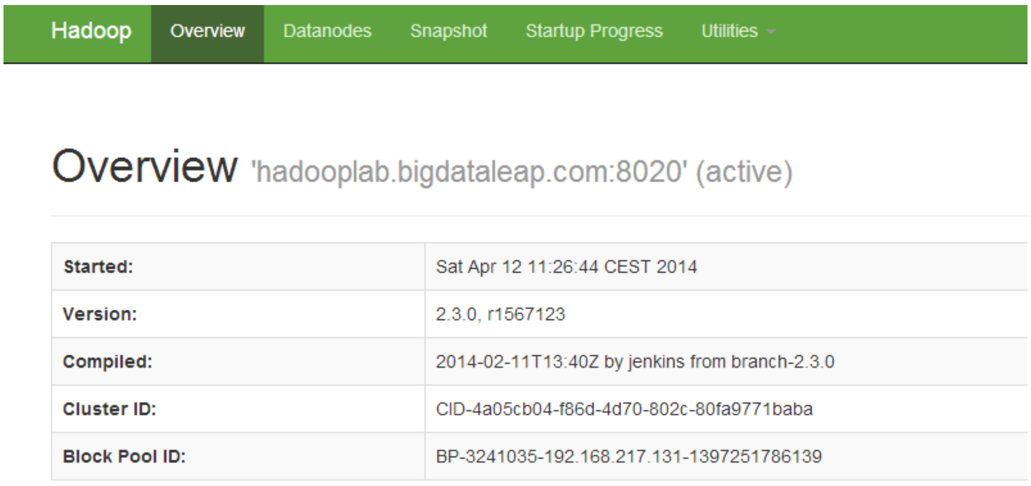
Gives you detailed report of the file that is specified

- Total number of blocks and their size
- Under replicated or missing blocks, if any

e. HDFS Web UI

- Open your browser & enter the following url

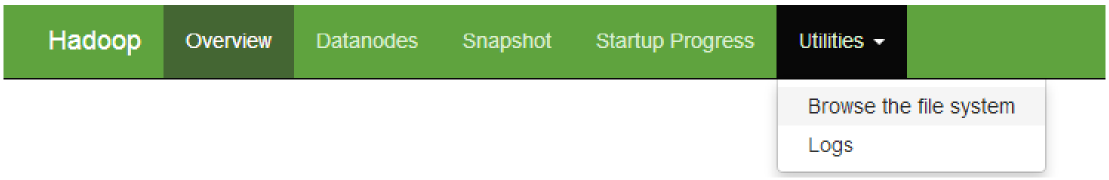
- http://hadooplab.bigdataleap.com:50070/



Overview 'hadooplab.bigdataleap.com:8020' (active)

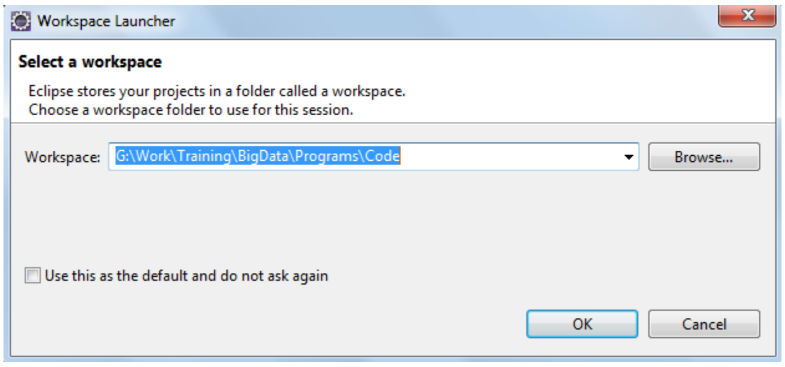
Started:	Sat Apr 12 11:26:44 CEST 2014
Version:	2.3.0, r1567123
Compiled:	2014-02-11T13:40Z by jenkins from branch-2.3.0
Cluster ID:	CID-4a05cb04-f86d-4d70-802c-80fa9771baba
Block Pool ID:	BP-3241035-192.168.217.131-1397251786139

- File system explorer and log explorer is available under utilities menu



4. Map Reduce - Word Count

- Start Eclipse on your windows or mac machine (Version 3.4 above)
- Select location where you want to create your project



Select a workspace

Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: G:\Work\Training\BigData\Programs\Code

☐ Use this as the default and do not ask again

- If you are starting eclipse for the first time, on the home page of eclipse click on **"Workbench"**
- Create a new Java Project called **MR*Lab***
  Hint: ***File -> New -> Others ->Java Project***
- *Go to project explorer and expand the project.*
- *Under project* ***MRLab-> src***
  create a package ***com.bigdataleap.samples.wordcount***
- Add the Hadoop jar files to the project
  Hint: Right Click on **MR*Lab -> Properties -> Java Build Path->Libraries->Add External Jars.*** Add the following jars

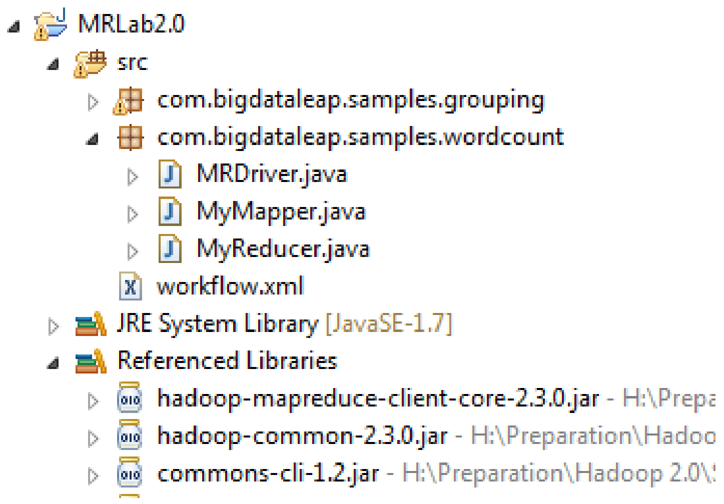
hadoop-2.3.0\share\hadoop\mapreduce\hadoop-mapreduce-client-core-2.3.0.jar
hadoop-2.3.0\share\hadoop\common\hadoop-common-2.3.0.jar
hadoop-2.3.0\share\hadoop\common\lib\commons-cli-1.2.jar

- Copy the following java sources to *project **MRLab-> src ->com.bigdataleap.com***

  All Code is available in **reference folder of your windows or mac machine**.

  MRDriver.java
  MyMapper.java
  MyReducer.java

  Hint: Can drag the files and drop it under the package in the project

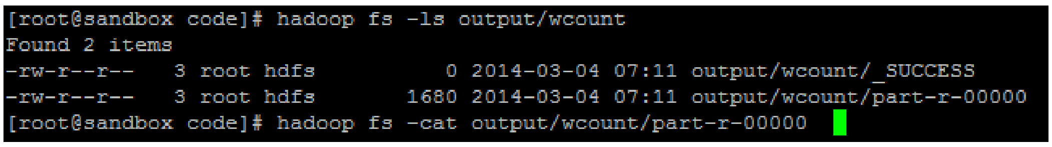


```
MRLab2.0
  src
    com.bigdataleap.samples.grouping
    com.bigdataleap.samples.wordcount
      MRDriver.java
      MyMapper.java
      MyReducer.java
    workflow.xml
  JRE System Library [JavaSE-1.7]
  Referenced Libraries
    hadoop-mapreduce-client-core-2.3.0.jar - H:\Prepa
    hadoop-common-2.3.0.jar - H:\Preparation\Hadoo
    commons-cli-1.2.jar - H:\Preparation\Hadoop 2.0\
```

- Verify if compilation error is shown. Eclipse automatically compiles and shows if there are any compilation error exists. If not error shown, it can be assumed that files are compiled correctly.
- Create the jar file
  Right Click on **MR*Lab* -> *export->java-> jar file*** (Click on browse and choose your desktop location and enter ***wordcount.jar*** for your jar file name)
- Transfer the jar file to VM under ***/home/hadoop/lab/programs***
  Hint: Use WinSCP software to ftp the jar file to linux VM
- cd /home/hadoop/lab/programs
- Run the job

  yarn jar wordcount.jar com.bigdataleap.samples.wordcount.MRDriver /lab/mr/words /lab/mr/wcount/

- Check the output directory
  hadoop fs –ls /lab/mr/wcount

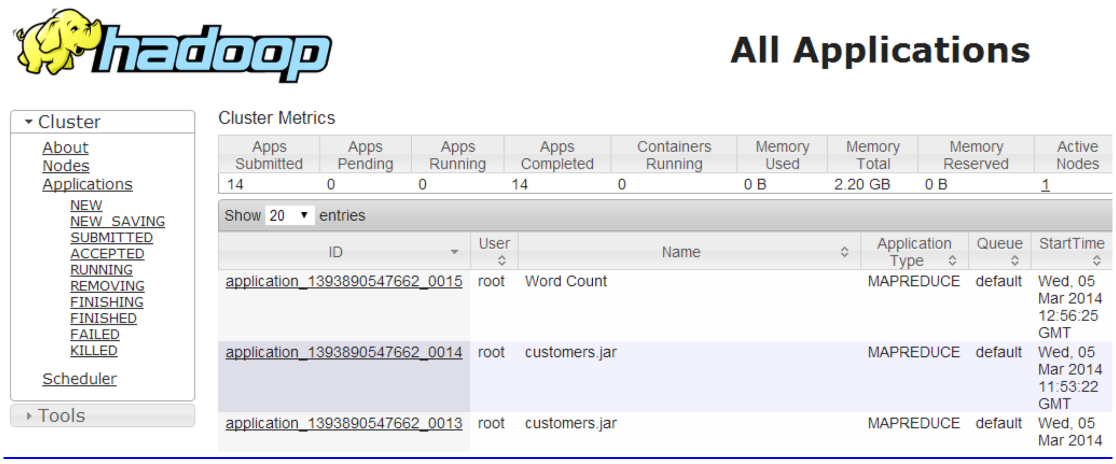


```
[root@sandbox code]# hadoop fs -ls output/wcount
Found 2 items
-rw-r--r--   3 root hdfs          0 2014-03-04 07:11 output/wcount/_SUCCESS
-rw-r--r--   3 root hdfs       1680 2014-03-04 07:11 output/wcount/part-r-00000
[root@sandbox code]# hadoop fs -cat output/wcount/part-r-00000
```

➢ Print the output file
hadoop fs -cat /lab/mr/wcount/part-r-00000

a. Job Tracker Web UI

- Open your browser & enter the following url
http://hadooplab.bigdataleap.com:8088/



All Applications

- Cluster
  - About
  - Nodes
  - Applications
    - NEW
    - NEW_SAVING
    - SUBMITTED
    - ACCEPTED
    - RUNNING
    - REMOVING
    - FINISHING
    - FINISHED
    - FAILED
    - KILLED
  - Scheduler
- Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Active Nodes
14	0	0	14	0	0 B	2.20 GB	0 B	1

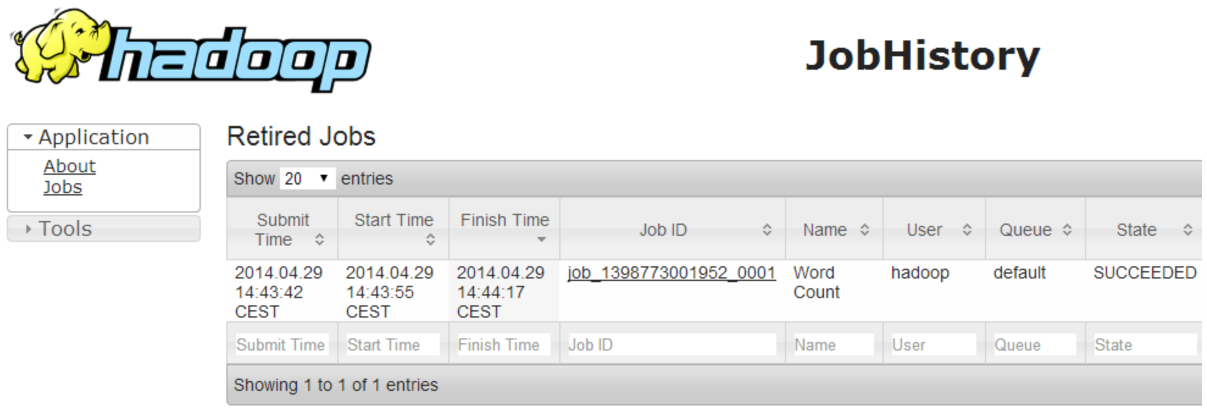
Show 20 entries

ID	User	Name	Application Type	Queue	StartTime
application_1393890547662_0015	root	Word Count	MAPREDUCE	default	Wed, 05 Mar 2014 12:56:25 GMT
application_1393890547662_0014	root	customers.jar	MAPREDUCE	default	Wed, 05 Mar 2014 11:53:22 GMT
application_1393890547662_0013	root	customers.jar	MAPREDUCE	default	Wed, 05 Mar 2014

- Job information will be available while in execution state. Once the job is completed, the job information is moved to history server.

a. Verify history server UI for completed jobs information

http://hadooplab.bigdataleap.com:19888/



JobHistory

- Application
  - About
  - Jobs
- Tools

Retired Jobs

Show 20 entries

Submit Time	Start Time	Finish Time	Job ID	Name	User	Queue	State
2014.04.29 14:43:42 CEST	2014.04.29 14:43:55 CEST	2014.04.29 14:44:17 CEST	job_1398773001952_0001	Word Count	hadoop	default	SUCCEEDED

Showing 1 to 1 of 1 entries

b. Write Log Statements for Debugging

Write some sysout statements in the map or reduce functions. These can be found in the stdout logs and can be accessed from the job history UI.

5. Map Reduce Lab – Working on Retail Data

A Sports retail company, Live Life, wants to find out the following indicators to plan its next year product strategy, from its current year's sales.

Input File: txns

Txnid, date, custid, amount, category, product, city, state, credit or cash

```
00000000,12-22-2011,4000183,033.76,Outdoor Play Equipment,Lawn Water Slides,Kansas City,Kansas,credit
00000001,05-19-2011,4000379,192.20,Gymnastics,Vaulting Horses,Lexington,Kentucky,credit
00000002,02-07-2011,4000905,064.51,Gymnastics,Gymnastics Rings,Salt Lake City,Utah,credit
00000003,09-20-2011,4000914,197.92,Outdoor Recreation,Tetherball,Denver  ,Colorado,credit
00000004,11-08-2011,4000270,132.20,Exercise & Fitness,Exercise Bands,Brownsville,Texas,credit
00000005,06-23-2011,4000312,168.74,Team Sports,Basketball,Miami,Florida,credit
00000006,10-30-2011,4000468,127.91,Water Sports,Boating,Gilbert,Arizona,credit
00000007,06-15-2011,4000230,060.58,Team Sports,Cheerleading,New York,New York,credit
00000008,06-02-2011,4000926,168.79,Outdoor Play Equipment,Playhouses,Midland,Texas,credit
00000009,07-25-2011,4000585,102.64,Exercise & Fitness,Weightlifting Gloves,Jacksonville ,Florida,credit
00000010,01-12-2011,4000877,105.89,Outdoor Play Equipment,Lawn Water Slides,New York,New York,credit
```

c. Write map reduce programs to find out Solve the following two problems

- Find out how much revenue was generated by products in each state?
- Find out Top selling products (By revenue generation) for each state?
- The retail company wants to for product recommendation when customers buy certain products from its retail stores or ecommerce websites. For that it wants to understand customers buying patterns like which products are bought together by same customers.

6. Unit Testing Map Reduce Programs

a. Unit Testing

b. Create another package ***com.bigdataleap.samples.grouping*** under MRLab/src

c. Add the Hadoop jar files to the project

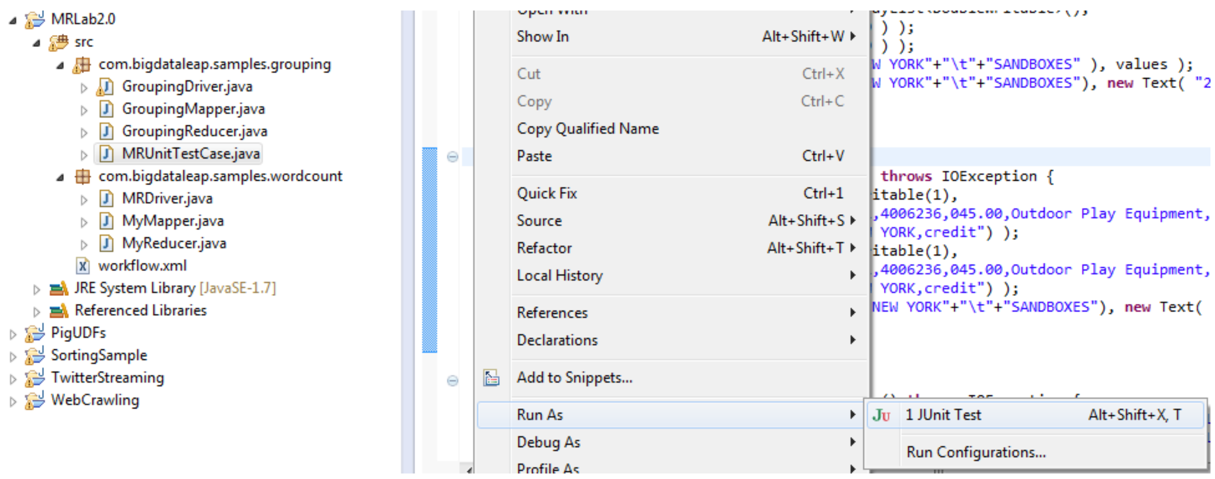
Hint: Right Click on **MR*Lab* -*> Properties -> Java Build Path->Libraries->Add External Jars.*** Add the following jars

apache-mrunit-1.0.0-hadoop2\lib\ commons-logging-1.1.1
apache-mrunit-1.0.0-hadoop2\lib\ junit-4.10
apache-mrunit-1.0.0-hadoop2\lib\ mrunit-1.0.0-hadoop2
And all the jars from
hadoop-2.3.0\share\hadoop\common\lib

d. Copy the following java sources to *project* ***MRLab-> src ->com.bigdataleap.com***

Add all the files to the project available on windows **\References\Code\com\bigdataleap\samples\grouping** *directory.*

e. Run the junit test cases and verify the results
Select the MRUnitTestCase soruce file and righ click on it and then run junit test



7. Creating Counters to get insight into data or job

Find out total number of credit and cash transactions.

```
enum RETAIL_TXN_RECORDS {
    TOTAL_TXNS,
    TOTAL_CREDIT_CARD_TXNS,
    TOTAL_CASH_TXNS
}

@Override
public void map(LongWritable key, Text value,
        Context context)
        throws IOException, InterruptedException  {

      String txnString = value.toString();
      String[] txnData = txnString.split( "," );
      double amount = Double.parseDouble( txnData[3] );

      context.getCounter( RETAIL_TXN_RECORDS.TOTAL_TXNS ).increment( 1 );

      if( txnData[8].equalsIgnoreCase( "credit" ) )
          context.getCounter( RETAIL_TXN_RECORDS.TOTAL_CREDIT_CARD_TXNS ).increment( 1 );
      if( txnData[8].equalsIgnoreCase( "cash" ) )
          context.getCounter( RETAIL_TXN_RECORDS.TOTAL_CASH_TXNS ).increment( 1 );


      // writing customer number and amount spent by each of them
      context.write( new Text( txnData[2].trim().toUpperCase() ), new DoubleWritable( amount ) );
}
```

8. Sqoop Configuration

f. Untar sqoop-1.4.4.bin.gz file into install directory

Go to the software install directory AND the sqoop tar file

cd /home/hadoop/lab/software

tar -xvf /home/hadoop/lab/downloads/sqoop-1.4.4.bin.tar.gz

g. Configure bash profile (This is already configured)

Go to /home/hadoop directory

Open the file by entering - *vi .bash_profile*

Append the following two lines in the file (This is already configured). Just verify it.

export SQOOP_HOME=/home/hadoop/lab/software/sqoop-1.4.4.bin__hadoop-2.0.4-alpha

export PATH=$PATH:$SQOOP_HOME/bin

Run the .bash_profile again

. <space>.bash_profile

h. Verify the installation

- Run ***sqoop help*** on linux prompt to verify if sqoop is running or not. If not, rectify the problems before proceeding to the next step.

i. Copy the mysql jdbc jar file into the $SQOOP_HOME/lib folder

- We will import and export data from mysql, so copy mysql jdbc jar file into sqoop's lib folder
  cp /home/hadoop/lab/downloads/mysql-connector-java-5.1.30-bin.jar $SQOOP_HOME/lib/

9. Sqoop Lab: Export and Import of data from RDBMS

Step 1 - Logging into mysql prompt

```
mysql -u root -p
```

Step 2 - Create and select a database

```
create database lab;

use lab;
```

Step 3 - Create table

```
CREATE TABLE customers (
CustID VARCHAR(7) NOT NULL,
FirstName VARCHAR(20) NOT NULL,
LastName VARCHAR(20) NOT NULL,
Age INT NOT NULL,
Profession VARCHAR(50) NOT NULL,
PRIMARY KEY ( CustID ) );
```

Exit the mysql prompt. And the type the following commands on linux prompt.

Step 7 – export data (From HDFS to MySql)

```
sqoop export  --connect jdbc:mysql://localhost/lab  --table customers  --username root
--password hadoop123 --export-dir  /lab/mr/custs
```

Step 8 – Verify Exported data

Login to mysql prompt and verify

```
use lab;
select * from customers;
```

Verify if you have exported 9999 customer records to mysql or not.

Step 9 – Import data (From MySql to HDFS)

```
sqoop import  --connect jdbc:mysql://localhost/lab  --table customers  --username  root
--password hadoop123 --target-dir  /lab/sqoop/customers    --m 1
```

Step 10 – Verify imported data

Check the imported file in hdfs

```
hadoop   fs   -cat   /lab/sqoop/customers/part-m-00000
```

10. HIVE Configuration

j. Untar hive-0.12.0.tar.gz file into install directory

Go to the software install directory AND the sqoop tar file

cd /home/hadoop/lab/software

tar -xvf /home/hadoop/lab/downloads/apache-hive-0.13.0-bin.tar

k. Configure bash profile (This is already configured)

Go to /home/hadoop directory

Open the file by entering - *vi .bash_profile*

Append the following two lines in the file

export HIVE_HOME=/home/hadoop/lab/software/hive-0.12.0

export PATH=$PATH:$HIVE_HOME/bin

Run the .bash_profile again

. <space>.bash_profile

l. Configure Mysql as metastore for Hive

cd $HIVE_HOME/conf

Make a copy of hive-default.xml.template as hive-site.xml

cp hive-default.xml.template hive-site.xml

- ***Configure the followings in the hive-site.xml (replace the values of the properties as shown below). hive-site.xml if already available in the reference directory. Transfer the hive-site.xml using WinSCP.***

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://localhost/hive?createDatabaseIfNotExist=true</value>
  <description>JDBC connect string for a JDBC metastore</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>com.mysql.jdbc.Driver</value>
  <description>Driver class name for a JDBC metastore</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>root</value>
  <description>username to use against metastore database</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>hadoop123</value>
  <description>password to use against metastore database</description>
</property>
```

m. Copy the mysql jdbc jar file into the $HIVE_HOME/lib folder

cp /home/hadoop/lab/downloads/mysql-connector-java-5.1.30-bin.jar $HIVE_HOME/lib/

n. Verify the installation

- Run ***hive*** on linux prompt to verify if sqoop is running or not. If not, rectify the problems before proceeding to the next step.

11. Creating Hive tables and running SQL Queries

- Run hive and verify if enters hive shell
  hive

- **Create database**

  create database retail;

- **Select database**

  use retail;

- **Create table for storing transactional records**

  ```
  create table txnrecords (txnno INT, txndate STRING, custno INT, amount DOUBLE, category
  STRING, product STRING, city STRING, state STRING, spendBy STRING )
  row format delimited
  fields terminated by ','
  stored as textfile;
  ```

- **Load the data into the table**

  ```
  LOAD DATA LOCAL INPATH '/home/hadoop/lab/data/txns'  OVERWRITE INTO TABLE
  txnrecords;
  ```

- **Describing metadata or schema of the table**

  ```
  describe txnrecords;
  ```

- **Counting no of records**

  ```
  select count(*) from txnrecords;
  ```

- **Counting total spending by category of products**

  ```
  select category, sum( amount ) from txnrecords  group by category;
  ```

- **Top 10 customers**

  ```
  select custno, sum( amount ) as total from txnrecords group by custno

  order by total limit 10;
  ```

12. Creating Hive tables with partitions and clusters

- **Select database**

  ```
  use retail;
  ```

- **Create partitioned table**

  ```
  create table txnrecsByCat (txnno INT, txndate STRING, custno INT, amount DOUBLE, product
  STRING, city STRING, state STRING, spendBy STRING )
  partitioned by ( category STRING )
  clustered by (state) INTO 10 buckets
  row format delimited
  fields terminated by ','
  stored as textfile;
  ```

- **Configure Hive to allow partitions**

set hive.exec.dynamic.partition.mode=nonstrict;
set hive.exec.dynamic.partition=true;
set hive.enforce.bucketing=true;

- **Load data into partition table**

  FROM txnrecords txn INSERT OVERWRITE TABLE txnrecsByCat PARTITION (category) SELECT txn.txnno, txn.txndate, txn.custno, txn.amount, txn.product, txn.city, txn.state, txn.spendBy, txn.Category DISTRIBUTE BY Category;

- **Show partitions created**

  show partitions txnrecsbycat;

- **Verify files under HDFS to check how Hive has created multiple directories for multiple partitions**

  hadoop fs -ls /user/hive/warehouse/retail.db/txnrecsbycat

  Then go inside each partition and check how files are created for each buckets.

- **Explain an sql query specific to a partition and check hive can determine which partition to scan**

  explain extended select state, sum(amount) as total from txnrecsbycat where category = "Puzzles" group by state;

```
Path -> Partition:
  hdfs://hadooplab.bigdataleap.com:8020/user/hive/warehouse/retail.db/txnrecsbycat/category=Puzzles
    Partition
      base file name: category=Puzzles
```

13. Using Flume to Stream data into HDFS

a. Untar flume

cd /home/hadoop/lab/software
tar -xvf /home/hadoop/lab/downloads/apache-flume-1.4.0.tar.gz

b. Configure flume to read streaming Log Files and write to HDFS

Go to /home/hadoop/lab/software/apache-flume-1.4.0/conf folder and create a file called flumelab.conf

touch flumelab.conf

Open the file

vi flumelab.conf

and enter the following

```
lab1.sources = source1
lab1.sinks = sink1
lab1.channels = channel1

lab1.sources.source1.type = exec
lab1.sources.source1.command = tail -F /home/hadoop/lab/data/tstream
lab1.sources.source1.channels = channel1

lab1.sinks.sink1.type = hdfs
lab1.sinks.sink1.hdfs.path = hdfs://hadooplab.bigdataleap.com/lab/tstream/
lab1.sinks.sink1.hdfs.filePrefix = tweets
# setting log rolling conditions
lab1.sinks.sink1.hdfs.rollInterval=60
lab1.sinks.sink1.hdfs.rollSize=104857600
lab1.sinks.sink1.hdfs.rollCount=10000
# setting log compression type
lab1.sinks.sink1.hdfs.fileType = CompressedStream
lab1.sinks.sink1.hdfs.codeC = GzipCodec

lab1.channels.channel1.type = memory
lab1.channels.channel1.capacity = 100000
lab1.channels.channel1.transactionCapacity = 50000

lab1.sources.source1.channels = channel1
lab1.sinks.sink1.channel = channel1
```

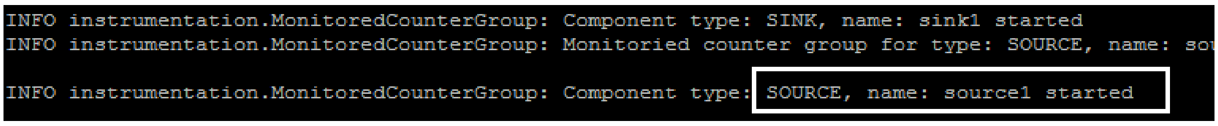
c. Starting the flume agent with source, channel and sink

Go to the bin folder of apache flume and change the permission to execute the ng-flume script and start the channel

cd /home/hadoop/lab/software/apache-flume-1.4.0/bin

chmod 744 flume-ng

./flume-ng agent --conf-file ../conf/flumelab.conf --name lab1 -Dflume.root.logger=INFO,console



```
INFO instrumentation.MonitoredCounterGroup: Component type: SINK, name: sink1 started
INFO instrumentation.MonitoredCounterGroup: Monitoried counter group for type: SOURCE, name: so

INFO instrumentation.MonitoredCounterGroup: Component type: SOURCE, name: source1 started
```

## Once sink and source is started, proceed to the next step…

d. Starting the logging tweets

Open one more putty session connecting the VM and do the below steps in the new terminal

Go to the below directory

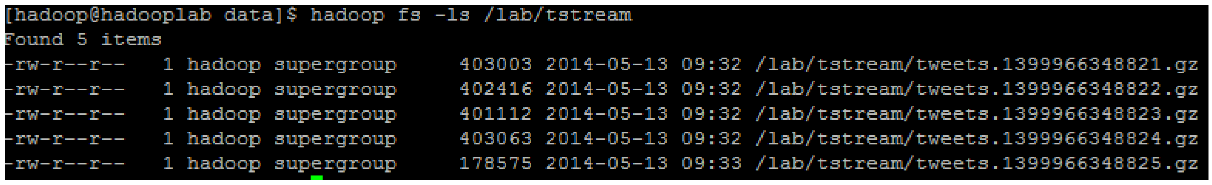
cd /home/hadoop/lab/data

Run the command (this command writes the tweets file into tstream every 5 seconds, simulating the scenarios of continuous tweets stream)

watch -n 10 'cat tweets >> tstream'

e. Starting the logging tweets

Check the hdfs location if the tweets have been streamed or not.



```
[hadoop@hadooplab data]$ hadoop fs -ls /lab/tstream
Found 5 items
-rw-r--r--   1 hadoop supergroup     403003 2014-05-13 09:32 /lab/tstream/tweets.1399966348821.gz
-rw-r--r--   1 hadoop supergroup     402416 2014-05-13 09:32 /lab/tstream/tweets.1399966348822.gz
-rw-r--r--   1 hadoop supergroup     401112 2014-05-13 09:32 /lab/tstream/tweets.1399966348823.gz
-rw-r--r--   1 hadoop supergroup     403063 2014-05-13 09:32 /lab/tstream/tweets.1399966348824.gz
-rw-r--r--   1 hadoop supergroup     178575 2014-05-13 09:33 /lab/tstream/tweets.1399966348825.gz
```

After about one or two minutes stop both the watch process and the flume-ng-agent process

14. Pig Configuration

a. Untar hive-0.12.0.tar.gz file into install directory

Go to the software install directory AND the sqoop tar file

cd /home/hadoop/lab/software

tar -xvf /home/hadoop/lab/downloads/pig-0.12.0.tar.gz

b. Configure bash profile (This is already configured)

Go to /home/hadoop directory

Open the file by entering - *vi .bash_profile*

Append the following two lines in the file

export PIG_INSTALL=/home/hadoop/lab/software/pig-0.12.0 *export PATH=$PATH:$PIG_INSTALL/bin*

Run the .bash_profile again

. <space>.bash_profile

15. Pig Programming (Analysing unstructured data)

a. Analyze tweet trends by using the input files created by flume

#load the tweets file into the following structure
tweets = load '/lab/tstream' using PigStorage('\t') AS (user:chararray, location:chararray, message:chararray);

#tokenize each tweet message
tweetwords = FOREACH tweets GENERATE FLATTEN(TOKENIZE(message)) AS word;

#search for only hash tags in the tweet messages
hashtags = FILTER tweetwords BY UPPER(word) MATCHES '#\\s*(\\w+)';

#groups each hash tag
taggroups = group hashtags by word;

#count the occurrence of each hash tag
tagcount = FOREACH taggroups GENERATE group AS tags, COUNT(hashtags) AS count;

#order by no of occurrence of each hash tag
tagorder = ORDER tagcount BY count DESC;

illustrate tagorder;

store tagorder into '/lab/pig/trends';

b. Verify the output

hadoop fs –cat /lab/pig/trends/part-r-00000 | more

16. Setting up Oozie

a. Untar oozie files and configure the path .bash_profile

cd /home/hadoop/lab/software

tar -xvf /home/hadoop/lab/downloads/oozie-4.0.0.tar.gz

cd /home/hadoop/lab/software/oozie-4.0.0

b. Configure core-site and restart the dfs services

## Go to hadoop’s conf directory and add the following lines to the core-site.xml

```
<property>
 <name>hadoop.proxyuser.hadoop.hosts</name>
 <value>hadooplab.bigdataleap.com</value>
</property>
<property>
 <name>hadoop.proxyuser.hadoop.groups</name>
 <value>root</value>
</property>
```

## Go to hadoop's sbin directory and restart the dfs services

```
cd  $HADOOP_INSTALL/etc/hadoop
./stop-dfs.sh
./start-dfs.sh
```

## And then run jps to verify if the processes have started or not.

Then wait for few minutes before continuing with next step.

c. Create the oozie sharelib directory in hdfs

Go to bin directory of oozie

```
cd $OOZIE_HOME/bin
./oozie-setup.sh  sharelib   create   -fs   hdfs://hadooplab.bigdataleap.com:8020
```

## After completion, verify if the sharelib directory is created in hdfs

```
hadoop fs -ls /user/hadoop/share/lib
```

d. Create oozie database

Login to mysql and create oozie database

```
mysql -u root –p
create database oozie;
```

then exit the mysql prompt.

## Go to bin directory of oozie and run the following commands

```
./ooziedb.sh create -sqlfile oozie.sql -run
```

e. Start oozie process and verify its status

## Start oozie by entering the following command from oozie's bin directry

./oozied.sh start

## And verify status

./oozie admin -oozie http://localhost:11000/oozie -status

it should show system mode is NORMAL.

Oozie is up and running now.

17. Creating a workflow using Oozie

a. Create the workflow and define the properties

## Transfer the following files available in the reference directory of your windows or mac machine to /home/hadoop/lab/programs directory of VM using WinSCP

- Oozie workflow.xml
- job.properties
- top10.pig

b. Copy all necessary files into hdfs

## Copy the above files and mysql jar file into hdfs

hadoop fs -copyFromLocal /home/hadoop/lab/downloads/mysql-connector-java-5.1.30-bin.jar /user/hadoop/share/lib/sqoop/

## Create working directories for storing oozie job files and copy all the required files.

hadoop fs -mkdir /user/hadoop/oozie

hadoop fs -mkdir /user/hadoop/oozie/apps

hadoop fs -mkdir /user/hadoop/oozie/apps/top10

cd /home/hadoop/lab/programs

hadoop fs -copyFromLocal workflow.xml /user/hadoop/oozie/apps/top10/

hadoop fs -copyFromLocal top10.pig /user/hadoop/oozie/apps/top10/

c. Run the job and check it's status

## Submit the job to oozie
oozie job -oozie http://hadooplab.bigdataleap.com:11000/oozie -config /home/hadoop/lab/programs/job.properties –run

## It should start the job and provide the job id. Using the job id status of the job can be obtained.

```
oozie job -oozie http://hadooplab.bigdataleap.com:11000/oozie -info  <job id>

## Job status can also be verified using browser
http://hadooplab.bigdataleap.com:11000/
```

d. Verify if job results are as expected

```
hadoop fs -ls /lab/oozie/custs    (Sqoop action output containing all imported records)

hadoop fs -ls /lab/oozie/top10 (Pig action output containing professions and their counts)

## Check the output of the job

hadoop fs -cat /lab/oozie/top10/part-r-00000
```