POORNIMA

Experiment - 1

Downloading and Installing the Cloudera VM

Instructions (Windows)

Hardware Requirements: (A) 8GB RAM (B) 20GB disk free

Instructions:

1) Install Virtual Box: Go to ~~internet~~ internet and download and install virtual-box for your computer.

2) Download the Cloudera vm.

3) Unzip the Cloudera VM.

4) Start Virtual Box.

5) Begin Importing: Import the VM by going to file→ Import Appliances.

6) Click the folder Icon.

7) Select the Cloudera_Quickstart_VM_5.4.2.0-virtualbox

8) Click next to proceed.

9) Click Import.

10) The virtual machine Image will be imported.

output :

[D, A, E, B, C]

[A]

POORNIMA

1) launch cloudera vm.

2) Cloudera vm booting.
3) The Cloudera vm desktop.

Experiment -2

Implementation of following data structure
a) linked list

```
import java.util.*;
public class Test {
    public static void main(String args[])
    {
        // Creating object of the
        // class linked list
    LinkedList<String> ll = new LinkedList<String>();
// Adding elements to the linked list
    ll.add("A");
    ll.add("B");
    ll.addLast("C");
    ll.addFirst("D");
    ll.add(2,"C");
System.out.println(ll);
    ll.remove("B");
    ll.remove(3);
    ll.removeFirst();
```

output:

pop operation

4
3
2
1
0

Element on stack top : 4

Element is found at position : 3

Element not found

```
11. Remove Last ();
System.out.println (11);
}
```

(ii) Stack

```
import java.io.*;
import java.util.*;
class Test
{
// Pursuing element on the top of the stack
static void stack-push (Stack < Integer > stack)
{
    for (int i=0; i<5; i++) {
        stack push(i);
    }
}
// poping element from the top of the stack
static void stack_pop (Stack <Integer> stack){
System.out.println (" pop operation :");
for ( int i= 0; i<5; i++) {
    Integer y= (Integer) stack.pop();
    System.out.println (y);
}
}
// Displaying element on the top of Stack
static void Stack-peek ( Stack < Integer > Stack){
    Integer element = ( Integer ) stack.peek();
    System.out.println ("Element on stack top :" +
```

```
element ");
    }
    // Searching element in the ~~stack~~ stack.
    static void stack_search (stack < Integer > static,
    int element)
    {
      Integer pos = (Integer) stack.search (element);
        if ( pos == -1)
              System.out.println (" Element not found");
      else
          System.out.println (" Element is found at
position :" + pos);
    }
public static void main (String args [ ]) {
    ~~static~~ stack <Integer> stack = new stack <Integer> ();
      stack_push (stack);
      stack_pop( stack);
        stack_push (stack);
        stack_peek( stack);
          stack_search( stack, 2);
          stack_search (stack, 6);
    }
}
```

output :

Elements of queue (0, 1, 2, 3, 4)

removed element = 0

(1, 2, 3, 4)

head of queue - 1

Size of queue - 4

POORNIMA

Queue:

```
import java.util.LinkedList;
import java.util.Queue;
public class QueueExample {
    public static void main(String[] args)
{
Queue<Integer> q = new LinkedList<Integer>();
    // Adds element {0, 1, 2, 3, 4} to
        // the Queue
for (int i=0; i<5; i++)
        q.add(i);
// displaying contents of the Queue
System.out.println("Elements of queue "+q);

// to remove the head of Queue
    int removable = q.remove();
    System.out.println("removed element" +
        remove date);
System.out.println(q);
// to view the head of Queue
        int head = q.peek();
        System.out.println(q);
        int size = q.size();
    System.out.println("Size of queue "+ size)
    }
}
```

output :

[Implementing , set , Example)

output

a, 100

b, 200

c, 300

d, 400

POORNIMA

(14) Set

```
import java.util.*;
public class setExample {
    public static void main (String []args) {
        // Creating linkedlist set using set
    Set <String> data = new LinkedHashSet < String >();

    data.add ( " Implementing ");
     data. add( "set");
     data. add( "Example");

    System.out.println(data);
     }
}
```

(15) Map

```
import java.util.*;
    public static void main (String args[]) {
        Map < String > Integer > hm = new HashMap <
        String , Integer>();
    hm.put ("a", new Integer (100));
    hm.put ("b", new Integer (200));
     hm.put ("c", new Integer (300));
      hm.put ("d", new Integer (400));

    // Traversing through the map
for ( Map.Entry < String , Integer > me; hm.entrySet ()){
```

Page No.....

POORNIMA

```
System.out.println(me.getKey() + ":");
System.out.print(me.getValue());
    }
  }
}
```

POORNIMA

Experiment - 4

Installing Hadoop: Perform setting up and Installing Hadoop in its three operating modes:
Standalone, Pseudo distribute, Fully distributed

Step by step procedure for Hadoop installation on UBUNTU

Open terminal in UBUNTU and execute the following commands one by one.

1) Sudo apt update.
2) Sudo apt install openjdk -8-jdk -y
3) java -version ; javac -version
4) Sudo apt install openssh-server openssh-client -y
5) ssh-keygen -t rsa -P "" -f ~/.ssh/id_rsa
6) cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
7) chmod 640 ~/.ssh/authorized_keys
8) ssh localhost
9) wget https://downloads.apache.org/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz
10) tar xzf hadoop-3.2.1.tar.gz
11) Sudo nano .bashrc

hadoop Related options.

export HADOOP_HOME = /home/hadoop-3.2.1
export HADOOP_INSTALL = $HADOOP_HOME
export HADOOP_MAPED_HOME = $HADOOP_HOME
export HADOOP_COMMON_HOME = $HADOOP_HOME

export HADOOP_HDFS_HOME = $HADOOP_HOME
export YARN_HOME = $HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR = $HADOOP_HOME/lib/native
export PATH = $PATH : $HADOOP_HOME/sbin : $HADOOP_HOME/bin
export HADOOP_OPTS = "-Djava.library.path = $HADOOP_HOME/lib/native"

12) Source ~/.bashrc

13) Sudo nano $HADOOP_HOME/etc/hadoop/hadoop-env.sh

export JAVA_HOME = /usr/lib/jvm/java-8-openjdk-amd64
which javac
readlink -f /usr/bin/javac

14) Sudo nano $HADOOP_HOME/etc/hadoop/core-site.xml

<configuration>
<property>
<name> hadoop.tmp.dir </name>
<value> /home/tmpdata </value>
</property>
<property>
<name> fs.default.name </name>
<value> hdfs://127.0.0.1:9000 </value>
</property>
</configuration>

15) Sudo nano $HADOOP_HOME/etc/hadoop/hdfs-site.xml

<configuration>
<property>
<name> dfs.data.dir </name>
<value> /home/dfsdata/namenode </value>
</property>
<property>
<name> dfs.data.dir </name>
<value> /home/dfsdata/datanode </value>
</property>
<name> dfs.replication </name>
<value> 1 </value>
</property>
</configuration>

16) Sudo nano $HADOOP_HOME/etc/hadoop/mapred-site*.xml

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

17) Sudo nano $HADOOP_HOME/etc/hadoop/yarn-site.xml

<configuration>
<property>
<name> yarn.nodemanager.aux-services </name>

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

</property>
<property>
<name> yarn.resourcemanager.hostname </name>
<value> 127.0.0.1 </value>
</property>
<property>
<name> yarn.nm.enable </name>
<value> 0 </value>
</property>
<property>
<name> yarn.nodemanager.env-whitelist </name>
<value> JAVA_HOME, HADOOP_COMMON_HOME, HADOOP_HDFS_HOME, HADOOP_CONF_DIR, CLASSPATH_PERPEND_DISTCACHE, HADOOP_YARN_HOME, HADOOP_MAPRED_HOME.
</value>
</property>
</configuration>

18) hdfs namenode -format
Navigate to the hadoop-3.2.1/sbin directory
19) ./start_dfs.sh
20) ./start_yarn.sh

21) jps
22) https://localhost:9870
23) http://localhost:8088
24) http://localhost:8088

Experiment - 4

How to Execute Word Count Program in Mapreduce using cloudera Distribution Hadoop (CDH)

steps to write Mapreduce code for Word Count.

1) First open Eclipse → then select file → New → Java Project → Name it Word Count → then finish.

2) Create Three Java classes into the project. Name then WC Driver (having the main function), WC Mapper, WC Reducer.

3) You have to include two Reference libraries for that:- Right click on Project → then select Build Path → click on Configure Build Path.

4) In the above figure, you can see the Add External JARs option on the Right Hand Side. Click on it and add the below mention files. you can find these files in /usr/lib/
1) /usr/lib/hadoop-0.20-mapreduce/hadoop-core-2.00-

mr/ - cdh5-1-3-0-jar

2) /usr/lib/hadoop/hadoop/hadoop-common-2-0-0-cdh5-13-0.jar

Mapper Code : you have to copy paste this program into WCMapper Java class file.

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.Reporter;

public class WCMapper extends MapReduceBase implements
    Mapper<LongWritable, Text, Text, IntWritable> {
public void map(LongWritable key, Text value, OutputCollector
    <Text, IntWritable> output, Reporter rep)
    throws IOException
{
    String line = value.toString();
    for (String word : line.split(" "))
    {
        if(word.length()>0)
    {
        output.collect(new Text(word), new IntWritable(1));
```

```
    }
  }
}
```

Reducer Code: You have to copy paste this program into the Reducer Java class file.

Code:

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

public class WReducer extends MapReduceBase implements
Reducer < Text, IntWritable, Text, IntWritable > {
    public void reduce < Text key, Iterator < IntWritable > value,
    OutputCollector < Text, IntWritable > output,
    Reporter rep) throws IOException
    {
        int count = 0;
        while (value.hasNext())
        {
            IntWritable i = value.next();
            count += i.get();
        }
```

output.collect(key, New IntWritable(count));
}
}

Driver code: You have to ~~wpg~~ copy paste this program into the WC Driver Java class file.

Code

```
import java.io.IOException;
import org.apachae.hadoop.conf.configure;
import org.apachae.hadoop.fs.path;
import org.apachae.hadoop.io.IntWritable;
import org.apachae.hadoop.io.Text;
import org.apachae.hadoop.mapred.Jobconf;
import org.apache.hadoop.util.ToolRunner;

Public class WCDriver extends Configured implements
    Tools {
    public int run (String args[]) throws Exception {
    if (args.length<2) {
    System.out.println("Please give valid input");
    return -1;
    }
    Job Conf Conf = new Job Conf (WC Driver.class);
File Input format.Set Input Paths (Conf, new path (args[0]));
File Output format.Set output Path Conf.new path (args[1]);
Conf.Set Mapper class (WC Mapper.class);
Conf.Set Reducer class (WC Reducer.class);
```

```
Conf.setMapOutputKeyClass(Text.class);
Conf.setMapOutputValueClass(IntWritable.class);
Conf.setOutputKeyClass(Text.class);
Conf.setOutputValueClass(IntWritable.class);
    JobClient.runJob(Conf);
    return 0;
}

public static void main(String args[]) throws Exception
{
    int exitCode = ToolRunner.run(new WCDriver(), args);
    System.out.println(exitCode);
    }
}
```

6) Now you have to make a jar file. Right Click on Project → click on Export → Select export destination as Jar files → Name the jar file (WordCount.jar) → Click on next → at last click on finish. Now copy this file into the Workspace directing of cloudera.

7) Open the terminal on CDH & change the directory to workspace. You can do this by using "cd workspace" command. Now, create a text file (WC file.txt) and move it to HDFS. For that open terminal and write this code (remember you should be in the same directory as jar file you have created just now.

POORNIMA

8) Now, run this command to copy the file input file into the HDFS
hadoop fs - put WCfile.txt WCFile.txt

9) Now to run the jar file by writing the code as below
$ hadoop jar wordCount.jar WCdriver WCfile.txt WCoutput

10) After Executing the code, you can see the result in WCoutput file or by writing following command on terminal:
hadoop fs - cat WCOutput/part-00000

POORNIMA

Experiment No. 05

Aim: Implement the following file management tasks in hadoop.

- Adding files and directories.
- Retriving files
- Deleting files.

Steps to Run

Step 1)

Checks for and directory in hadoop

$ hadoop fs -ls

found 5 items

drw xx — xx — — — n base /base

drw xx — — x - 8 /sols

drw xx wx r wx - hdfs /usr

drw xx - x r - x - hdfs /usr

Step 2)

Create directory "Experiment 4" inside "user" directory then check - '4'.

hadoop fs -lc /usr/ gdfs/ user/ experiment - 4

Steps3)

Create file in system in home directory /home/ Cloudaa and write in file.

$ cat > file /.txt

Lalighy ajeet

Step4)

Copy this file in hadoop in /usr/ experiment - 4 directory

POORNIMA

$hadoop fs -ls put/ home / Cloudera /file.txt /usr/ experiment -s

Step 5) Check "file.text" file in hadoop in /usr/ experiment 4 found 1 items
-rw -r-1 --1 Cloudera supergroup
/usr/ experiment 5/ file.txt.

Step 6) Show Content of the file "file.txt" $ hadoop fs - cat/ usr/ experiment -s/ file.txt

Step 7) Remove /delete /file.txt from hdfs $hadoop fs - rm /usr/ experiment -s/ file.txt
22/04/07 21:59:14 WSO fs: Trash Policy Default name node trash Configuration.

deletion interval = 0 interval = 0 min

Deleted /usr/ experiment s/ file.txt

$ hadoop fs - ot/ usr/ experiment s/ file.txt

Directory this exist
$ hadoop fs - ls /usr/
found 7 item

- a -admin supergroup /usr/ admin

POORNIMA

-a	cloudera	cloudera	/usr/cloudera
-	- Cloudera	Supergroup	/user/ experiments
-	- mapred	hadoop	/user/ history
-	- hive	hive	/user/ hive
-	- oozie	hive	/user/ ~~spark~~ oozie
-	- spark	spark	/user/ spark

Step 8)

Delete directory "Experiment-5"

$ hadoop fs -rmdir /user/ experiment-5

$ hadoop fs -ls /user/

Found 6 item

drwxr-x---	- admin	Supergroup	/user/admin
"	- Cloudera	cloudera	/user/ admin
"	- mapred	hadoop	/user/ history
"	- hive	hive	/user/ hive
"	- oozie	oozie	/user/ oozie
drwxr-xr-x	- spark	spark	/user/ spark

Steps :-

Download the dataset - available on the NCEI website
Save this txt file in /home/ cloudera directory

Step1: Open Eclipse → then select file → New → Java Project → Name it weather Project.

Step2: Then select use an execution environment → choose Java SE-1.7

1) Go on next
2) In Java setting → select weather project - Go in libraries - add external Jars
3) Go in file system → Go in usr - lib - hadoop - then select all jar files - Click on ok
4) Go in add jar files - hadoop - client - select all jar files - ok
5) Finish.

Step3) Now we are going to Create a package file
1) Go in weather Project - src - right click - New - Class - Name - My max min - finish.

2) Write java code for analysis weather data in Mymax Min class - save it.

Step 4: Now export this java file :-
for this go -- go in file -- export -- select jar file
-- next -- select jar file specification -- select export destination -- jar file -- /home / Cloudera / weather Project - finish.

step 5) Copy Weather data.txt file from download folder to / home / Cloudera folder.

step 6) go on terminal and write Command
i) hdfs dfs -ls /
(It will show list of directory & files in HDFS
ii) Now make a directory in ~~hdfs~~ hdfs with name "for Weather".
(iii) Now copy Weather data.txt file to HDFS by using put option.
(iv) Command : hdfs dfs - put / home/ Cloudera weather data.txt / ~~format~~ weather /.

step 7) Now check file copied or not
hdfs dfs - cat / for weather / weather.txt /

step 8) Now Compile.
hadoop jar /home / Cloudera / Weather Project -jar My Max Min / for weather / weather data.txt / weatherout /

step 9) Check weatherout directory (it will contain output file) hdfs dfs - ls / weatherout

step 10) hdfs dfs - cat / Run output file which contains result weather out / part -r - 00000

Weather dataset.docx

23907 20150101 2.423 -98.03 30.62 2.2 -0.6
0.8 0.9 6.2 1.47C 3.7 1.1 2.5 99.9
854 97.2 0.369 0.308 -99.000 -99.000
-99.000 7.0 -8.1 -9999.0 -9999.0

POORNIMA

Program:

MyMaxMin.java

```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.lib.output.FileOutput
import org.apache.hadoop.mapreduce.lib.output.TextOutput;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.conf.Configuration;

Public class MyMaxMin {
    public static class MaxTemperatureMapper extends
        Mapper <LongWritable, Text, Text, Text> {

    public static find int Missing = 9999;
    @ override
        public void map (LongWritable arg0, Text,
            Value, Context, Context)
    throws IO Exception, Interrupt Exception {
        String line = Value.toString();
        if( !(line.length() == 0)) {
        String data = line.substring (6, 14);
        float temp_Max = float.parseFloat (line.substring
            (39, 45).trim();
```

```
float temp_Min = Float.parseFloat(line.substring(47,53).
                 trim());
if( temp_max > 30.0) {
context.write(new Text("The Day is Hot Day :" +date),
Text(String.valueOf(temp_max)));
}

if (temp_min <15) {
Context.write( new Text("The Day is cold Day :" +date),
        new Text(String.valueOf(temp_min)));
}
}
}
}
```

```
public static class MaxTemperatureReducer extends
    Reducer <Text, Text, Text, Text> {

public void reduce (Text key, Iterator <Text> Values,
Context context)
    throws IOException, InterruptException {
String temperature = values.next().toString();
Context.write (key, new Text(temperature));
```

POORNIMA

```
    }
}
public static void main (String [] args) throw
    Exception {
    Configuration conf = new Configuration ();
job job = new job (conf, " weather example ");
job . set Jar By ( class (My Max Min. class);
job . set Map output key class (Text . class);
job . set Map output Value class (Text . class);
job . set mapper class (Max Temperature mapper . class);
job . set Reducer class ( max Temperature mapper . class);
job . set output format class ( Text output format class);

Path output = new Path (args[1]);
File input Format . add Input Path ( job, new path (args[0]);
File input Format . set output Path ( job, new path (args[1]);

Output path . get file system (conf). delete ( output path);
~~path~~
System . exit ( job . wait for Completion (true) ? 0 : 1);
    }
}
```