POORNIMA

Experiment - 6

write a program of word count using Map-Reduce

Creating the path files/directories
hdfs dfs -mkdir -p /user/process
create a file wordcount.txt and copy this file in this directory.
hdfs dfs -put /netjs/MapReduce/wordcount.txt /user/process

Map Reduce java code

```
import java.io.IOException
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import orge.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount{

public static class MyMapper extends Mapper<LongWritable,Text,
  Text,IntWritable>{
```

POORNIMA

```
private Text word = new Text();
public void map (LongWritable key, Text value, Context context)
  throws IOException, InterruptedException {

String[] stringArr = value.toString().split("\\s+");
for (String str: stringArr){
    word.set(str);
    context.write
  }
}
}

public static class MyReducer extends Reducer<Text, IntWritable
   Text, IntWritable> {
  private IntWritable result = new IntWritable();
  public void reduce (Text key, Iterable<IntWritable> values,
    Context context)
    throws IOException, InterruptedException {
  int sum = 0;
  for (IntWritable val : values){
      sum += val.get();
  }
  result.set(sum);
  context.write(key, result);
}
}
```

```
public static void main (String [] args ) throws Exception {
    Configuration conf = new Configuration ();
    Job job = Job . get Instance ( conf , "WC" );
    job . set Jar By Class ( word Count . class );
    job . set Mapper Class ( My Mapper . class );
    job . set Reducer class ( My Reducer . class );
    job . set output Key Class ( Text . class );
    job . set Output Value Class ( Int writable . class );
    File Input Format . add Input Path ( job , new Path ( args [0] ));
    File Output Format . set Output Path ( job , new Path ( args [1] ));
    System . exit ( job . wait For Completion ( true ) ? 0 : 1 );
  }
}
```

Required jars for hadoop MapReduce code:-

These are found ~~and~~ inside

share/hadoop directory

hadoop - common - 2.9.0. jar
hadoop - hdfs - 2.9.0 jar
hadoop - hdfs - client - 2.9.0. jar
hadoop - mapreduce - client - core - 2.9.0. jar
hadoop - mapreduce - common . 2.9.0. jar
hadoop - mapreduce - client - jobclient. 2.9.0.jar
hadoop - mapreduce - client - hs - 2.9.0. jar
hadoop - mapreduce - client - app - 2 . 9.0. jar
Commons - io - 2.4. jar

Creating jar of your wordcount mapreduce code
once the code is compiled, create a jar file, in eclipse IDE.
select Export - Java - jar file
To run the map reduce code -
go to directory
bin/hadoop jar/net js/MapReduce/wordcount.jar org
net.js.WordCount /user/process/user/out

Jar Path file as net.js/MapReduce/wordcount.jar org.net.js.WordCount
org.net.js.WordCount is fully qualified path to your jar file to java program class
/user/proces - path to ~~output~~ in directory
/user/out - path to output directory

POORNIMA

Experiment

Implementation of Matrix multiplication using map reduce.

Before writing the code let's first create Matrices and put them in HDFS.

Create two files M1 and M2 and put the matrix values (seperate colums with spaces and rows with a line break)

Put the above files to HDFS at location
/user/ clouders / Matrices /

```
hdfs dfs -mkdir /user/ cloudera /matrice
hdfs dfs -put /path/to /M1 / user/ cloudera/ matrices/
hdfs dfs -put /path/to /M2 / user/ cloudera/ matrices/
```

Creating two program Mapper.py and reducer.py
defining the dimension of Matrix

```
#! usr / lib/ python
import sys
m-r = 2
m-c = 3
n-r = 3
n-c = 2
i = 0
```

Read each line ie a row from stdin and Split then to seperate elements. Map int to each element as we

POORNIMA

read element as string from stdin.

```
for line in sys.stdin: el = map(int, line.split())
```

The mapper will first read the first matrix and then the second differentiate them we can keep a count i of the line number we are reading and first m-r line will bring the first matrix.

```
if (i<m-r):
    for j in range(len(el)):
        for k in range(n-c):
            print "{0}\t{2}\t{3}".format(i, k, j, el[j])
else:
    for j in range(len(el)):
        for k in range(m-r):
            print "{0}\t{1}\t{2}\t{3}".format(k, j, i-m-r, el[j])
i = i+1
```

printing the key values

{0} {1} {2} are the part of key and {3} is the value

{0} {1} {2} actually represent the position of element from A & B to A*B

{0} is the row position of the element.

{1} is the column position of the element

{2} is the position of the element in addition.

Page No.....

We can see that A's element is repeated B's number of column times ie 2 and B's element is repeated A's number of row times ie 2.

i is used to iterate through each row.
j is used to iterate through each column.
k is used to iterate through each duplicate produced.

for each element in matrix A:
element remains in same row, therefore {0} = i
element is duplicated and distributed to each column, therefore column pos in A*B = Duplication order of element ie {1} = k
and {2} = j
for each element in matrix B:-
Each element remain in the same column, so {1} = j
{0} = k
and {2} = i - m-r

Reducer code

```
#!/usr/lib/python
import sys
m-r = 2
m-c = 3
n-r = 3
n-c = 2, matrix = [ ]
for row in range(m-r):
    r = [ ]
    for col in range(m-c):
```

POORNIMA

```
        s=0
        for el in range(m-c):
            mul=1
            for num in range(2):
                line = sys.stdin.readline()
                n= map(int, line.split('\t'))[-1]
                mul*n
            s+=mul
        r.append(s)
        matrix.append(r)
print('\n'.join([str(x) for x in matrix]))
```

Running the Map-Reduce job on hadoop

```
$ chmod +x -/Desktop/mr/matrix-mul/Mapper.py $ chmod +x
$ chmod +x -/Desktop/mr/matrix-mul/Reducer.py
$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar
> -input /user/cloudera/matrices/
 -output /user/cloudera/mat-output
 mapper -/Desktop/mr/matrix-mul/Mapper.py
 reducer -/Desktop/mr/matrix-mul/Reducer.py
```

get output

```
hdfs dfs -cat /user/cloudera/mat-output/*
```

POORNIMA

Experiment -

Write a program to of map reduce for weather data set.
The aim of program to find the max temp. recorded for each year of NCDC data.
Map reduce is based on set of key value pair. So first we have to decide on the types for the key/value pairs for the input.
Map Phase :- The input for map phase is set of weather data files so the types of input key value pairs are long writable and text and the types of output key value pairs are text and int writable.

Reduce Phase :- Reduce phase takes all the values associated with a particular key. That is all the temp. values belong to a particular year is fed to a same reducer. Then each reducer finds the highest recorded temp.

files that we will create three java classes :-
- HighestMapper.java
- HighestReducer.java
- highestdriver.java.

HighestMapper.java

```
import java.io.IOException;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
```

POORNIMA

```
public class HighestMapper extends MapReduceBase implements
Mapper < LongWritable, Text, Text, IntWritable >
{
 public static final int MISSING =9999;
 public void map ( LongWritable key, Text value, OutputCollector
 < Text, IntWritable > output, Reporter reporter) throws IOException
 {
  String line = value.toString();
  String year = line.substring(15,19);
  int temperature;
  if (line.charAt (87) =='+')
    temperature = Integer.parseInt(line.substring(88,92));
  else
   temperature = Integer.parseInt(line.substring(87,92));
   String quality = line.substring(92,93);
   if (temperature != MISSING && quality.matches("[01459]"))
   output.collect(new Text(year), new IntWritable(temperature));
  }
 }
```

HighestReducer.java

```
import java.io.IOException;
import java.util.Iterator;
import java.org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
```

```
public class HighestReducer extends MapReduceBase implements
Reducer < Text, IntWritable, Text, IntWritable >
{
public void reduce (Text key, Iterator < IntWritable > values,
 OutputCollector < Text, IntWritable > output, Reporter reporter)
throws IoException
{
 int max_temp = 0;
 while (values.hasNext())
 { int current = values.next().get();
  if (max_temp < current)
   max_temp = current;
  }
  output.collect (key, new IntWritable (max_temp/10));
 }
}
```

HighestDriver.java

```
import org.apache.hadoop.fs.path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;

public class HighestDriver extends Configured implements Tool {
```

```
public int run (String[] args) throws Exception
{
  JobConf conf = new JobConf (getConf(), HighestDriver.class);
  Conf.setJobName ("Highest Driver");
  conf.setOutputKeyClass (Text.class);
  Conf.setOutputValueClass (IntWritable.class);
  Conf.setMapperClass (HighestMapper.class);
  Conf.setReducerClass (HighestReducer.class);
  Path inp = new Path (args[1]);
  Path out = new Path (args[2]);
  FileInputFormat.addInputPath (conf, inp);
  FileOutputFormat.setOutputPath (conf, out);
  JobClient.runJob (conf);
  return 0;
  }
  public static void main (String[] args) throws Exception
  {
   int res = ToolRunner.run (new Configuration (), new
        HighestDriver (), args ());
   System.exit (res);
  }
}
```

Build jar file → Add jar file using
/usr/local/hadoop/share/hadoop/common
/usr/local/hadoop/share/hadoop/mapreduce

POORNIMA

Copy input files from local system to hdfs

hadoop fs -copy fromLocal source-path (local System) Destination-path(HDFS)

hd usr @ MKP: /usr/local/hadoop/sbin $ hadoop fs -copy from local /home/mkp/Desktop/1901/HighestTemp/input

Copy input files from local system to HDFS

Hadoop fs -copy Fromlocal source-path (local system) Destination-Path

Executing Jar file

hadoop jar jar-file-path main-class-name input_file directory-path output_directory-path

POORNIMA

Experiment

Install and run Pig then write Pig Latin script to sort group, join, project and filter your data.

Downloading apache PIG from
https://download.apache.org/Pig/
Pig releases → Please make sure you're downloading from the archieves.

Extract the pig directory files as they are in zip format into
"E:/hadoop-env"

Setting up Environmental Variables

Add new environmental variable to/as
name - PIG_HOME
value E:/hadoop-env/pig-0.17.0

Add a new path variable as
%PIG_HOME%/bin

Starting Apache Pig
Note that Hadoop must be running in background.
Open cmd as administrator and run
pig -version

we recieveee an exception, to fix this error need to change pig.cmd file located in the "pig-0.17.0\bin" directory by changing the Hadoop-Bin-PATH value from "%Hadoop-Home%\bin" to "%Hadoop_home%\libexec".

Now let's try to run the 'pig-version'

1. Writing a Pig script

create and open an Apache Pig script file in an editer ie:- gedit

Command sudo gedit /home/user/output.pig

writing some PIG commands inside output Pig

A = load "/user/information.txt"
using pigstorage(' ') as (FName: chararray, LName: chararray, Mobile No: chararray, City: chararray, Profession: chararray);

B = ForEach A generate FName, Mobile, Profession;
Dump B;

Execute the apache PIG script

Command pig /home/user/output.pig

POORNIMA

Experiment -

Install and run Hive then use Hive to create, alter and drop database, tables, views, functions and indexes.

Downloading microsoft SDK → downloading microsoft SDK v.7.1 to get the tools, compilers, headers and libraries that are necessary to run hadoop.

Cygwin : Download and install Unix command line tool Cygwin to run the Unix commands on windows as per your windows.

Maven Download and install maven 3.1.1. The installation of apache Maven is simple as extracting the files and adding bin directory to path environmental variable.

Step 1 :- Start windows 7 pc and open a new command prompt

run → mvn -v to check installation.

2. Download protocol buffers 2.5.0 and extract to a folder in C drive
3. Type the below commands prompts to check environmental variables.
4. Download source and build a windows package : Download hadoop Sources tarball hadoop-2.6.4-src.tar.gz and extract to a folder having short path.

use C:\hdp to avoid runtime problem due to maximum path length limitation in windows.

Exract hadoop-2.6.0.tar.gz to a folder. Add Environment variable Hadoop-home = "C:\hdp\hadoop-2.6.4-src" and edit

Page No.....

POORNIMA

19. Open in browser

HIVE Install

1. Open cmd and write
   Java -version

2. Download any binary for hive from any source available on
3. Paste the HIVE in C drive, eg C:\hdp\apache-hive-2.1.0-bin
4. Set Environmental variables named %HIVE_HOME%\bin
5. Install and start MySQL, if you have not.
6. download mysql-connector-java-5.0.5.jar
7. Use My SQL command line and execute
   $ Mysql -u root -p
   Enter password
   mysql > create Database metastore_db;

8. Create a user [hive user] in Mysql database using root user. Let's take hive user as 'userhive' and 'hive password' as hivepwd.
   mysql > CREATE USER 'hiveuser'@'%' IDENTIFIED By 'hive password"
   mysql > GRANT all on *.* to 'hiveuser'@ localhost identified by hive password.
   mysql > Flush privileges;

9. Go to %hive_Home %\conf folder and copy "hive-default.xml.template" file. Now ~~ren~~ rename the copied file as "hive-site.xml"

POORNIMA

The template file has formatting needed for "hive-site.xml"

10. Edit hive-site.xml in %.HIVE_HOME%/conf directory and add the configuration.

Database commands in HIVE

Create (Database) [If not Exists] db_name
[Comment database_comment]
[location hdfs_path]
[With DBProperties (property_name = property_value, ---)];

Example

hive> create database if not exists first DB comment
'this is first demo" location
location '/user/hive/warehouse/newdb'
with DBProperties ('create by' = 'abhay', 'create for' = 'dezyre');

Drop database

Drop (Database) [IF Exist] database_name [Restrict/cascade]
hive > drop database if exist first DB cascade;

Alteration

Alter (Database) database_name SET DBProperties
(property_name = property_value, ---);

Alter (Database) database_name SET OWNER [USER/ROLE] user_to
role;

Page No.....

Describing table

Describe [Extended / Formatted] [db_name.] table_name
[.col_name (.field_name]

hive > describe extended college.college_students;