ITMD 521 – Week-05-I/O **Name: Jyoti Shukla**

CWID: A20378620

Modified, compiled and executed the MapReduce jobs on a local system and remote cluster

Steps:

1. Concerted all the .txt files into a file called all-sotu.txt by using command in /vagrant_data :

Cat 1796-sotu.txt 1993-sotu.txt 1997-sotu.txt 2001-sotu.txt 2005-sotu.txt 2009-sotu.txt 2013-sotu.txt >> all-sotu.txt

2. Copy the WordCount.java and WordCount2.java in the local system /Ubuntu-xenial-large-521/data/
3. Create a file named patterns.txt which contains all the prepositions of the English Language
4. Start the Vagrant by using command: vagrant up
5. We assume that Hadoop-2.5.2 is installed in the vagrant and all the path is defined.\
6. Start all the services by using command :
    a. start-dfs.sh
    b. start-yarn.sh
    c. mr-jobhistory-daemon.sh start historyserver
7. And check that all 7 services are started or not by using command: jps

Part-1:

1. Copy the pattern.txt and all-sotu.txt files in the Hadoop file system by using commands:
    a. hadoop fs -mkdir -p /user/$USER/input- created a folder for input
    b. hadoop fs -mkdir -p /user/$USER/output- created a folder for output
    c. hadoop fs -put /vagrant_data/pattern.txt /user/$USER/input – Copying the pattern.txt file in input folder
    d. hadoop fs -put /vagrant_data/all-sotu.txt /user/$USER/input – Copying the al-sotu.txt files in input folder.
2. Compile the java files i.e. WordCoutn.java and WordCoutn2.java by using command:

    hadoop com.sun.tools.javac.Main *.java and then create a jar file of this java classes named wc.jar by using command: jar cf wc.jar *.class

```
vagrant@itmd521:/vagrant_data$ ls
1796-sotu.txt  2001-sotu.txt  2013-sotu.txt     checkpoint_signature  lock.machine-action-6c4ab815c138300f815678f50701a86d.lock  WordCount2.java
1993-sotu.txt  2005-sotu.txt  all-sotu.txt      jp-leases             machine-index                                             WordCount.java
1997-sotu.txt  2009-sotu.txt  checkpoint_cache  lock.dotlock.lock     pattern.txt
vagrant@itmd521:/vagrant_data$ hadoop com.sun.tools.javac.Main *.java
vagrant@itmd521:/vagrant_data$ ls
1796-sotu.txt  2013-sotu.txt        lock.machine-action-6c4ab815c138300f815678f50701a86d.lock  WordCount2$TokenizerMapper.class
1993-sotu.txt  all-sotu.txt         machine-index                                              WordCount2$TokenizerMapper$CountersEnum.class
1997-sotu.txt  checkpoint_cache     pattern.txt                                                WordCount.class
2001-sotu.txt  checkpoint_signature WordCount2.class                                           WordCount$IntSumReducer.class
2005-sotu.txt  jp-leases            WordCount2$IntSumReducer.class                             WordCount.java
2009-sotu.txt  lock.dotlock.lock    WordCount2.java                                            WordCount$TokenizerMapper.class
vagrant@itmd521:/vagrant_data$ jar cf wc.jar *.class
vagrant@itmd521:/vagrant_data$
```

3. Run the application and store the result in output folder which is created in previous step by using command:

    hadoop jar wc.jar WordCount /user/$USER/input /user/$USER/output/part01

and copy the part1 result set into local system data folder by using command:

hadoop fs -put /user/$USER/output/part1 /vagrant_data/

4. Sorting the top ten words from the resultset by using command:

   sort -n -k1.4 /vagrant_data/part1/part-r-00000 | tail -15

Administrator: Windows PowerShell

```
vagrant@itmd521:/vagrant_data$ sort -n -k1.4 /vagrant_data/part1/part-r-00000  | tail -15
on      234
be      244
is      393
for     445
$14,500 1
$1,500  1
$2,500  1
we      560
$1,600  3
in      640
our     657
of      1142
and     1217
to      1433
the     1867
vagrant@itmd521:/vagrant_data$
```

Hence, we are done with part 1

Part-02:

1. Run the application and store the result in the output folder by using command:
   hadoop jar wc.jar WordCount2 /user/$USER/input /user/$USER/output/part02

   and copy the part02 result set into local system data folder by using command:

   hadoop fs -put /user/$USER/output/part02 /vagrant_data/

2. Sorting the top ten words from the resultset by using command:

   sort -n -k1.4 /vagrant_data/part2/part-r-00000 | tail -15

```
the     1867
vagrant@itmd521:/vagrant_data$ sort -n -k1.4 /vagrant_data/part2/part-r-00000  | tail -15
on      234
be      244
is      393
for     445
$14,500 1
$1,500  1
$2,500  1
we      560
$1,600  3
in      640
our     657
of      1142
and     1217
to      1433
the     1867
vagrant@itmd521:/vagrant_data$
```

Hence, we are done with part 2

ITMD 521 – Week-05-I/O **Name: Jyoti Shukla**
CWID: A20378620

Part-03:

1. Modifying the WordCount.java to look for only words that occur only four times as shown below:

```
        }
if(sum>=4){
        result.set(sum);
        context.write(key, result);
}
    }
```

2. Remove the old jar files and all .class files and compile the java classes again : hadoop com.sun.tools.javac.Main *.java and then created a jar file of this java classes named wc.jar by using command: jar cf wc.jar *.class
3. Run the application and store the result in the output folder by using command: hadoop jar wc.jar WordCount /user/$USER/input /user/$USER/output/part03

   and copy the part03 result set into local system data folder by using command:

   hadoop fs -put /user/$USER/output/part03 /vagrant_data/

4. Sorting the top ten words from the resultset by using command:

   sort -n -k1.4 /vagrant_data/part3/part-r-00000 | tail -15

```
the     1867
vagrant@itmd521:/vagrant_data$ sort -n -k1.4 /vagrant_data/part3/part-r-00000  | tail -15
not     195
by      197
are     207
And     215
on      234
be      244
is      393
for     445
we      560
in      640
our     657
of      1142
and     1217
to      1433
the     1867
vagrant@itmd521:/vagrant_data$
```

Hence, we are done with part 3

Part-04:

1. By using application WordCount2, we will skip all the prepositions from the all-sotu.txt file by using command:

hadoop jar wc.jar WordCount2 -Dwordcount.case.sensitive=false /user/$USER/input/all-sotu-txt /user/$USER/ouput/part04/ -skip /user/$USER/input/pattern.txt

2. Copy the part04 result set into local system data folder by using command:

hadoop fs -put /user/$USER/output/part04 /vagrant_data/

3. Sorting the top ten words from the resultset by using command:

sort -n -k1.4 /vagrant_data/part4/part-r-00000 | tail -15

```
to      1433
the     1867
vagrant@itmd521:/vagrant_data$ sort -n -k1.4 /vagrant_data/part4/part-r-00000  | tail -15
--      174
not     199
are     212
it      235
be      245
is      393
$14,500 1
$1,500  1
$2,500  1
$1,600  3
th      603
our     709
we      750
and     1432
the     1965
vagrant@itmd521:/vagrant_data$
```

Hence, we are done with part 4.