**Master Password: cloudproject1**

**Part 2: Docker**

**To build the docker image based on the docker file attached in myimage folder**

|  |
| --- |
|  |
|  | docker build -t hadoop . |
|  |  |
|  | **To start the docker image as a container** |
|  | docker run --name hadoop \ |
|  | -p 9870:9870 \ |
|  | -p 9868:9868 \ |
|  | -p 9864:9864 \ |
|  | -p 50060:50060 \ |
|  | -p 50030:50030 \ |
|  | -p 19888:19888 \ |
|  | -p 10033:10033 \ |
|  | -p 8032:8032 \ |
|  | -p 8030:8030 \ |
|  | -p 8088:8088 \ |
|  | -p 8033:8033 \ |
|  | -p 8042:8042 \ |
|  | -p 8188:8188 \ |
|  | -p 8047:8047 \ |
|  | -p 8788:8788 \ |
|  | -ti hadoop |
|  |  |
|  |  |
|  | **To run a mapreduce program and view the output (steps included in demo.sh)** |
|  | sh ${HADOOP\_HOME}/demo.sh |
|  |  |

**Part 3: N-Gram**

**These are the steps we followed:-**

1. Write the java code in an IDE(Eclipse) of choice. Compile it and export it as a jar file to a path on the local PC. Use WinSCP to copy this jar file to a location on the VM.
2. Move the jar file from linux file system to root of hdfs with the below command:

**bin/hdfs dfs -put input/ngramInputFile.txt /**

3) Run the n-gram program using this command:

**bin/hadoop jar input/ngramFreqCounter.jar com.cc.mapreduce.ngramFreqCounter 2 /ngramInputFile.txt /ngramOutput**

Here, 2 is the value for n. /ngramInputFile.txt is the path of the input file. /ngramOutput is the path where the output file has to be created.

4) Check if the MapReduce job is completed in putty or on the web application(<http://159.65.43.106:19888>).

5) View the output file created after the MapReduce job using the command:

**bin/hdfs dfs -cat /ngramOutput/part-r-00000**. The output file name(**part-r-00000)** is subject to change. It’s correct name should be verified before running the above command. We have attached the test case(test input file) as part of our submission.

The source code file is also submitted.

**Part 4: Anonymous log analysis**

The 4 JAVA files in part-4 Anonymous log analysis are the source codes of the solutions to the four questions. The steps below can be replicated

**These are the steps we followed:-**

1) Write the java code in an IDE(Eclipse) of choice. Compile it and export it as a jar file to a path on the local PC. Use WinSCP to copy this jar file to a location on the VM. This step is the same for all the 4 cases.

2) Move the jar file from linux file system to root of hdfs with the below command:

**bin/hdfs dfs -put input/access\_log / .** This step is the same for all the 4 cases.

3-5)

Please note that the output file name(**part-r-00000)** is subject to change. It’s correct name should be verified before running the command to view the output file.

And, check if the MapReduce job is completed in putty or on the web application(<http://159.65.43.106:19888>) before proceeding to view the output file.

* Run the 1st question’s code using this command:

**sudo bin/hadoop jar input/urlCounter.jar com.cc.mapreduce2.urlCounter /access\_log /urlOutput**

And, view the output file created after the MapReduce job using the command:

**bin/hdfs dfs -cat /urlOutput/part-r-00000**

* Run the 2nd question’s code using this command:

**sudo bin/hadoop jar input/ipCounter.jar com.cc.mapreduce3.ipCounter /access\_log /ipOutput**

And, view the output file created after the MapReduce job using the command:

**bin/hdfs dfs -cat /ipOutput/part-r-00000**

* Run the 3rd question’s code using this command:

**sudo bin/hadoop jar input/maxURLCounter.jar com.cc.mapreduce4.maxURLCounter /access\_log /maxURLOutput**

And, view the output file created after the MapReduce job using the command:

**bin/hdfs dfs -cat /maxURLOutput/part-r-00000**

* Run the 4rth question’s code using this command:

**sudo bin/hadoop jar input/maxIPCounter.jar com.cc.mapreduce5.maxIPCounter /access\_log /maxIPOutput**

And, view the output file created after the MapReduce job using the command:

**bin/hdfs dfs -cat /maxIPOutput/part-r-00000**

The source code files of the 4 questions is submitted.