# Lab Exercise 11: Hadoop, HDFS, MapReduce and Pig

|  |  |
| --- | --- |
| **Purpose:** | This lab introduces the *Hadoop and MapReduce environment that* you will be working on for the next lab. After completing the tasks in this lab you should able to:   * Get help on the various Hadoop commands * Observe a MapReduce job in action * Query various Hadoop servers regarding status * Understand and execute “Pig” statements |
|  | |
| **Tasks:** | Tasks you’ll be completing in this lab include:   * Run Hadoop and Hadoop fs and collect help information * Run a shell script to perform a word count activity * Run a MapReduce job to produce similar output * Investigate the UI for MapReduce/HDFS components to track system behavior * Run “Pig” statements to execute the same tasks done with MapReduce |
|  | |
| **References:** | References used in this lab are located in your ***Student Resource Guide.*** See the Guide for:   * Hadoop Commands * HDFS Commands |

### Workflow Overview

### LAB Instructions

| **Step** | **Action** |
| --- | --- |
| 1 | **Check your Hadoop Environment:**   1. Make sure you are connected to the right directory. Execute the command:   **cd ~/LAB11**   1. Execute the command:   **printenv | grep ‘^H’**   1. Are there any Hadoop variables defined? Which ones? 2. Execute the command:   **hadoop fs –ls**   1. What do you see? 2. Execute the command:   **ls**  This should be different |
| 3 | **Get Help:**  1. Execute the command:  **hadoop –help 2>&1 | tee Hadoop.hlp**   1. Execute the command:   **hadoop fs –help | tee HDFS.hlp**   1. Now list the files by executing the commands:   **clear && more Hadoop.hlp**  **clear && more HDFS.hlp**  These files are also contained in your Student Resource Guide. These commands can be run from your command shell whenever you need them.   1. Now we create an alias so we don’t have to type “hadoop fs” every time   **alias hdfs=”hadoop fs”** |
| 4 | **Load data into HDFS:**  In this step, we will be loading an input data file into HDFS that we will be using in later activities.  1. Execute the following commands.  **hdfs –copyFromLocal greeneggsandham.txt input/greeneggsandham.txt**  **hdfs –ls input**  The file *greeneggsandham.txt* is now in HDFS. |
| 5 | Execute MapReduce:  In this step, you will run a MapReduce job and observe its output. This job is identical to the example job discussed as part of the lecture.  1. First, execute the command  **time ./wf.sh greeneggsandham.txt**  2. How long did it take to produce its output? \_\_\_\_\_  3. List the file “MRwordcount.sh” by executing  **more MRwordcount.sh**  This should be identical to the file used as an example in the lecture.  4. Execute the following command:  **time ./MRwordcount.sh**  5. What do you see? \_\_\_\_\_  6. How long did it take to execute this script? \_\_\_\_\_  7. Is this command slower than the Unix script file? Why do you think that is? \_\_\_\_\_  8. Looking at the output  The output of the MapReduce job is stored in a subdirectory of the output directory in HDFS. This directory is named “d” followed by a string of numbers (and is listed in the output of the MapReduce command). You can see the content of the directory by executing the following command:  **hdfs –cat output/d\*/part-\***   1. Answer the following Questions:    1. Compare the results from wf.sh and MRwordcount.sh. Are they consistent?    2. Which word repeated the maximum number of times in wf.sh and in MRwordcount.sh?    3. Is Dr. Seuss successful in using just 50 words? |
| 6 | **Check on MapReduce and HDFS Progress:**  In this step, we will look at some of the status information about MapReduce and Hadoop. First we look at the administrative interfaces for the JobTracker and the TaskTracker components of the MapReduce framework, and then NameNode User Interface (UI) for HDFS.  0.1. Create a shortcut on your desktop for each UI that you will be investigating. For each of the following steps, right-click on your Windows desktop and select New>>Shortcut.  Enter the following URL: http://<IP-ADDRESS-OF-SERVER>”:”<PORT\_NUMBER>/ For the JobTracker, PORT\_NUMBER will be 50030. IP-ADDRESS-OF-SERVER is the IP address for your server that you received at the beginning of the course. Click “Next.”  Name the shortcut “JobTracker” and click “Finish”  0.2. Do the same for the NameNode (port number 50070) and for the TaskTracker (port # 50060).  1. Click on the desktop icon labeled “Jobtracker”. This will bring up the UI for the JobTracker MapReduce node.  2. Examine the output. Do you see anything similar to what you saw in the output of the script you just ran? Anything different?  3. Click on the desktop icon labeled “TaskTracker” This will bring up the UI for the MapReduce TaskTracker node.  4. Examine the output. Do you see anything similar to what you saw in the output of the script you just ran? Anything different?  Now we look at the UI for the NameNode node in HDFS.  5. Click on the desktop icon labeled “NameNode” This invokes the UI for the NameNode node in an HDFS implementation. Examine the output.  If you are strictly working as an analyst, you may never need to look at the administrative interface to these components of the Hadoop framework. On the other hand, if things aren’t working out as you might have expected, you can use these interfaces to take a deeper look “under the hood” at the mechanics of Hadoop. |
| 7 | **Execute Pig statements and verify results:**  We will now work on “Pig”. As mentioned in the Student Resource Guide “Pig” provides a SQL like query interface to extract the required output from data in HDFS (or stored in a local standard file system). Execute the following steps:   1. At the $ prompt run the command pig   **$ pig** |
|  | 1. We will be executing the “pig” statements in the “grunt” mode, and you will observe the “grunt” prompt on your screen. 2. At the “>grunt” prompt execute the following statements:   **myinput = load 'hdfs://localhost:9000/user/gpadmin/input/greeneggsandham.txt' USING TextLoader() as (myword:chararray);**  **words = FOREACH myinput GENERATE FLATTEN(TOKENIZE(LOWER(\*)));**  **grouped = GROUP words BY $0;**  **counts = FOREACH grouped GENERATE group, COUNT(words);**  **counts = ORDER counts BY $1;**  **dump counts;**   * The first statement loads the “greeneggsandham.txt” we loaded earlier on to the hdfs. * The second statement tokenizes the entire text and creates a table “words”. We have also ensured all the characters are converted to “Lower” case. * The third statement “groups” the table “words”. * In the fourth statement creates a table “counts” which generates a key value pair of the {unique word, number of occurrences} * Fifth statement orders the table “counts” in the ascending order of the count(words). * The final “dump” statement will show the execution of the backend MapReduce jobs generated by the “pig” statements and display the results on the screen.   **Note that you can always use the “dump” statement to inspect the output on the “stdout” after any of the previous statements one through four.**   1. Review the display on the screen and answer the following Questions:    1. How many jobs were submitted at the backend and what are the job ids?    2. How many records were read by the input?    3. What is the term frequency of the following words? 2. Sam 3. Anywhere 4. Not    1. Which words have the maximum and minimum term frequencies? |
|  | 1. Exit “grunt” mode with following command at the “>grunt” prompt:   **quit**   1. Now let us execute the same statements in a “batch” mod e executing the pig script we have already have in the file “pig.txt” in the LAB11 directory. Make sure you are connected to the right directory. Execute the command:   **cd ~/LAB11**   1. You can run Pig in batch mode using [Pig scripts](http://pig.apache.org/docs/r0.9.1/start.html#pig-scripts) and the "pig" command (in local or hadoop mode). We use Pig scripts to place Pig Latin statements and Pig commands in a single file. While not required, it is good practice to identify the file using the \*.pig extension. The script file we have in our directory is “test.pig”. To execute this in hadoop mode use the following commands and observe the results:   **$ pig test.pig**  **or**  **$ pig -x mapreduce test.pig** |