**HU Extension Assignment 05 E63 Big Data Analytics**

Issued on: February 27, 2016 Due by 11:30PM EST, March 04, 2016

Please work in Hue’s Hive Editor.

**Problem 1.** Create your own tables KINGJAMES with columns for words and frequencies and insert into the table the result of Hadoop MapReduce GREP program which produce word counts on file all-bible. File is provided with this assignment. Tell us all words in table KINGJAMES which start with letter “w” and are 4 or more characters long and appear more than 250. There are not that many of those words so you can count them by hand. However, you want to be more automated so please change your query so that it gives you the number of such words as its output. When comparing a word with a string your use LIKE operator, like

word like ‘a%’ or word like ‘%th%’

Symbol ‘%’ means any number of characters. You measure the length of a string using function length() and you change the case of a word to all lower characters using function lower().

hadoop fs -mkdir all-bible

hadoop fs -copyFromLocal /mnt/hgfs/shared\_dir\_with\_cloudera\_vm/all-bible all-bible/

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar grep all-bible/all-bible hadoop-grep-bible-output '\w+'

create table KINGJAMES (freq INT, word STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' stored as textfile;

LOAD DATA INPATH "/user/cloudera/hadoop-grep-bible-output/part-r-00000" INTO TABLE KINGJAMES;

select \* from KINGJAMES where word like 'w%' and length(word)>=4 and freq>250;

select count(1) from KINGJAMES where word like 'w\_\_\_%' and freq>250;

**Problem 2**. Create your own table SHAKE similar to the one we used in class and populate it with results of MapReduce GREP program applied to the file all-shakespeare which is provided with this assignment. Create your own MERGED table similar to the one we used in class. The table will list all the word and the frequencies with which they appear in either table SHAKE or KINGJAMES. Your table will be “better” than the one we used in class. In class we only inserted into that table words that appear in both texts. Please use **outer joins** to populate the table with words that also appear in one but not the other text. Tell us how many words appear in table SHAKE but not in KINGJAMES and how many appear in KINGJAMES and not in SHAKE. Select 10 words from each group for us. To solve this problem you will have to consult Hive Tutorial at <https://cwiki.apache.org/confluence/display/Hive/Tutorial> or simply Google around the Web.

hadoop fs -mkdir all-shakespeare

hadoop fs -copyFromLocal /mnt/hgfs/shared\_dir\_with\_cloudera\_vm/all-shakespeare all-shakespeare

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar grep all-shakespeare/all-shakespeare hadoop-grep-shakespeare-output '\w+'

create table SHAKE (freq INT, word STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' stored as textfile;

LOAD DATA INPATH "/user/cloudera/hadoop-grep-shakespeare-output/part-r-00000"

INTO TABLE SHAKE;

create table MERGED (freq\_in\_bible INT, freq\_in\_shake INT, word STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' stored as textfile;

INSERT OVERWRITE TABLE MERGED

select k.freq AS freq\_in\_bible, s.freq AS freq\_in\_shake,

COALESCE(s.word, k.word) AS word

from KINGJAMES k FULL OUTER JOIN SHAKE s ON (k.word = s.word);

select count(1) from MERGED where word IS NULL;

select count(1) from MERGED where freq\_in\_bible IS NULL AND freq\_in\_shake IS NOT NULL;

select word from MERGED where freq\_in\_bible IS NULL AND freq\_in\_shake IS NOT NULL LIMIT 10;

select count(1) from MERGED where freq\_in\_bible IS NOT NULL AND freq\_in\_shake IS NULL;

select word from MERGED where freq\_in\_bible IS NOT NULL AND freq\_in\_shake IS NULL LIMIT 10;

**Problem 3**. When you have your three queries for counting common words, words that are present in Bible but not in Shakespeare and the words present in Shakespeare but not in Bible refined and working, collect the execution times of those queries. This is not straightforward, since Hive does not give you a simple tool to time your queries. You can look in query logs (a tab next to the Results tab) and sum execution times of map and reduce jobs. That is close enough. Then change your Hue Query Editor and switch to Impala Editor. Run your queries in that editor. This time you have no way of read the time. You just make a subjective estimate. Compare the execution time of queries with Impala and Hive. Impala is usually much faster. One thing to notice here is that you can use Impala on some of Hive tables. Unfortunately not all. Hive is more versatile than Impala.

Hive editor:

Common words:

select count(1) from MERGED where freq\_in\_bible IS NOT NULL AND freq\_in\_shake IS NOT NULL;

18sec

select count(1) from MERGED where freq\_in\_bible IS NOT NULL AND freq\_in\_shake IS NULL;

18sec

select count(1) from MERGED where freq\_in\_bible IS NULL AND freq\_in\_shake IS NOT NULL;

18sec

Impala editor:

select count(1) from MERGED where freq\_in\_bible IS NOT NULL AND freq\_in\_shake IS NOT NULL;

2 sec

select count(1) from MERGED where freq\_in\_bible IS NOT NULL AND freq\_in\_shake IS NULL;

2 sec

select count(1) from MERGED where freq\_in\_bible IS NULL AND freq\_in\_shake IS NOT NULL;

2 sece

**Problem 4.** Please create Hive table APACHELOG for extraction of the content of Apache server logs:

CREATE TABLE apachelog (

host STRING,

identity STRING,

user STRING,

time STRING,

request STRING,

status STRING,

size STRING,

referer STRING,

agent STRING)

ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.RegexSerDe' WITH SERDEPROPERTIES ( "input.regex" = "([^ ]\*) ([^ ]\*) ([^ ]\*) (-|\\[[^\\]]\*\\]) ([^ \"]\*|\"[^\"]\*\") (-|[0-9]\*) (-|[0-9]\*)(?: ([^ \"]\*|\"[^\"]\*\") ([^ \"]\*|\"[^\"]\*\"))?", "output.format.string" = "%1$s %2$s %3$s %4$s %5$s %6$s %7$s %8$s %9$s" )

STORED AS TEXTFILE;

Please expand the above regular expression to single line before copying the entire statement to Hue Hive editor.

Test success of creation of that table using two single line samples of Apache logs contained in files apache.access.2.log and apache.access.log (note files do not have .txt suffix) contained in the attached file examples\_older.zip. Once you are convinced that you can safely insert those two samples into your table apachelog, insert a bigger log contained in file apache\_log\_1.txt. Tell us how many lines of apache logs you have in table apachelog.

We are also attaching two groups of example data files for Hive: examples\_older.zip and examples.zip. You might find those files useful if you want to keep on learning about the technology. You could get those files by downloading Hive distributions, as described in notes.

describe apachelog;

select \* from apachelog;

LOAD DATA LOCAL INPATH "/mnt/hgfs/shared\_dir\_with\_cloudera\_vm/examples\_older/apache.access.2.log"

INTO TABLE apachelog;

LOAD DATA LOCAL INPATH "/mnt/hgfs/shared\_dir\_with\_cloudera\_vm/examples\_older/apache.access.log"

INTO TABLE apachelog;

select \* from apachelog;

LOAD DATA LOCAL INPATH "/mnt/hgfs/shared\_dir\_with\_cloudera\_vm/access\_log\_1.txt"

INTO TABLE apachelog;

select \* from apachelog LIMIT 10;

select count(1) from apachelog;

39346

Please, describe every step of your work and present all intermediate and final results in a Word document. Please, copy past text version of your command. We cannot retype text that is in JPG images. Please, always submit a copy of original, working scripts and class files you used as separate files. Sometimes we need to run your code and retyping is too costly. Please, submit to the class drop box. For issues and comments visit the class Discussion Board .