# IST769 Homework 5 Submission

## Basic Information

Your Name: Srihari Busam   
Your SUID: sbusam  
Your Email: sbusam@syr.edu  
Date Due: 11/04/2021  
Homework #: 5

## QUESTIONS:

1. Upload all the documents in **datasets/text** into a folder called **text** in HDFS. What HDFS command must you run to verify the files are there after they are uploaded? Your answer should consist of the command you typed to complete the task.
2. In this part you will upload the **clickstream** dataset to HDFS. Specifically, create a **clickstream** folder in HDFS, then create a **logs** and **iplookup** folder inside the clickstream folder. Upload all of the **\*.log** files from the **datasets/clickstream** local folder into **clickstream/logs** in HDFS. Upload the **ip\_lookup.csv** file from the same folder into **clickstream/iplookup** on HDFS. Verify the files are there. Your answer should consist of the commands you typed to complete the task.
3. Use the MapReduce examples:  
   export MREX=/usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar   
   to perform a wordcount on the 2016 State of the Union address, saving the output to the HDFS folder **sotu2016**. Write down the commands to complete the task. How many times does the word **are** appear in the 2016 State of the Union address? Describe a process which could be done to make the wordcount more useful?
4. Type the following command to import the **fudgemart\_v3** database into the local **mysql** instance on the Hadoop client:  
   mysql -u root -p < ~/datasets/fudgemart/mysql.sql   
   The password is **cloudera**. Write down the commands you used to complete these tasks:  
   Use the **sqoop** utility to verify there are tables in the database by listing them from the **fudgemart\_v3** database. Next write a sqoop command to import Fudgemart products in the ‘Clothing’ department into a HDFS folder **/user/cloudera/fudgemart-clothing**
5. Let’s import HDFS data into MySQL. Record each command you type as your solution:
   1. Load **datasets/tweets/tweets.psv** into the HDFS folder **tweets**
   2. Login to MySQL: mysql -u root -p The password is **cloudera**. Create a database **twitter**
   3. Create a table called **tweets** inside the database **twitter** the table should have columns for id, timestamp, date time, username, and tweet\_text.
   4. Export the data from HDFS into the MySQL table.  
      **TIPS:** If your SQOOP job fails it is likely due to the table constraints such as selecting a data type too small for the imported data. Try to insert a row in the table using a sample from the HDFS data. This will help you to ensure your chosen data types will work.

## ANSWERS:

### **ANSWER 1**

|  |
| --- |
| CODE |
| hdfs dfs -mkdir text  hdfs dfs -put datasets/text/\*.txt text/  hdfs dfs -ls text |

Command output:

Graphical user interface, text

Description automatically generated

### **ANSWER2:**

|  |
| --- |
| CODE |
| hdfs dfs -mkdir clickstream  hdfs dfs -mkdir clickstream/logs  hdfs dfs -mkdir clickstream/iplookup  hdfs dfs -ls clickstream/  hdfs dfs -put datasets/clickstream/\*.log clickstream/logs/  hdfs dfs -put datasets/clickstream/ip\_lookup.csv clickstream/iplookup/  hdfs dfs -ls clickstream/logs  hdfs dfs -ls clickstream/iplookup |

### **ANSWER3:**

It seems the are appeared 27 times in the state of the union address text. "are,” and “are.” Also should be considerd part of the wordcount which sould lead to a total of 29 times the word appearing in the text.

**How to use wordcounts?**

In my view if we can visualize the counts as a word cloud, it could help to visualize the theme of the text without even reading the document. This way if we want to classify a set of documents we can leverage word counts.

|  |
| --- |
| CODE |
| export MREX=/usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar  yarn jar $MREX wordcount /user/cloudera/text/2016-state-of-the-union.txt /user/cloudera/sotu2016  hdfs dfs -ls sotu2016  hdfs dfs -cat sotu2016/part-r-00000  # search for words matching are in the Hadoop output file.  hdfs dfs -cat sotu2016/part-r-00000 | grep are |

Screenshot which shows Hadoop map-reduce output searched for **“are”** words:

Text

Description automatically generated

### **ANSWER 4:**

|  |
| --- |
| CODE |
| mysql -u root -p < ~/datasets/fudgemart/mysql.sql  #check that the table exists and data exists  sqoop eval --connect "jdbc:mysql://cloudera/fudgemart\_v3" --username=root --password=cloudera --query "select \* from fudgemart\_products where product\_department=\"Clothing\""  # Move data to Hadoop folder **/user/cloudera/fudgemart-clothing**  sqoop import --connect "jdbc:mysql://cloudera/fudgemart\_v3" --username=root --password=cloudera --query "SELECT \* FROM fudgemart\_products WHERE product\_department=\"Clothing\" AND \$CONDITIONS" --target-dir /user/cloudera/fudgemart-clothing --as-textfile --split-by product\_department |

Screenshot to show data in the table using sqoop eval command:

Screenshot to show the table data through sqoop:

Text

Description automatically generated

Screenshot to show the data imported to Hadoop folder:

Text

Description automatically generated

### **ANSWER 5:**

|  |
| --- |
| CODE |
| # create hdfs folder and move the tweets from datasets  hdfs dfs -mkdir tweets  hdfs dfs -put datasets/tweets/tweets.psv tweets/  # Create the database  mysql -u root -p  CREATE DATABASE twitter;  USE twitter;  CREATE TABLE tweets(id BIGINT, timestamp DOUBLE(20,8), date\_time VARCHAR(100), username VARCHAR(100), tweet\_text VARCHAR(1000));  \q  # export data into table  sqoop export --connect "jdbc:mysql://cloudera/twitter" --username=root --password=cloudera --table tweets --export-dir /user/cloudera/tweets --input-fields-terminated-by "|"  # check data on mysql side  sqoop eval --connect "jdbc:mysql://cloudera/twitter" --username=root --password=cloudera --query "SELECT \* FROM tweets" |

Screenshot to show tweets imported to Hadoop:

Text

Description automatically generated

Screenshot of exporting process:

Text

Description automatically generated

Screenshot of table on sql table data:

Text

Description automatically generated