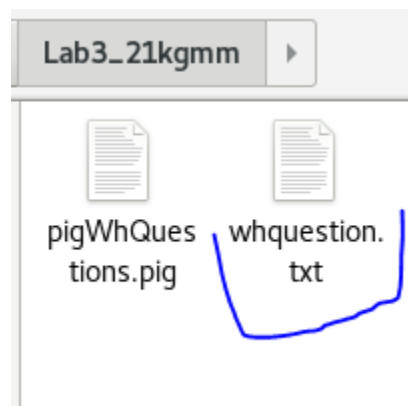


Lab_3

Name: Karim Gamal Mahmoud Mohamed

ID: 21KGMM

1-- Prepare a text file with WH questions by taking example sentences from the following link, name the file as 'whquestion.txt' and save it in your local drive: [WH Question Words | Vocabulary | EnglishClub.](#)



2-- Copy the **whquestion.txt** file into a hdfs location

```
[root@quickstart-bigdata ~]# hdfs dfs -mkdir -p /user/osboxes/Lab3_21kgmm
[root@quickstart-bigdata ~]# hdfs dfs -ls /user/osboxes
Found 1 items
```

```
[root@quickstart-bigdata ~]# hdfs dfs -copyFromLocal /home/osboxes/Lab3_21kgmm/whquestion.txt /user/osboxes/Lab3_21kgmm
```

```
[root@quickstart-bigdata ~]# hdfs dfs -cat /user/osboxes/Lab3_21kgmm/whquestion.txt
What is your name?
What ? I can't hear you.
You did what?
What did you do that for?
When did he leave?
Where do they live?
Which colour do you want?
Who opened the door?
Whom did you see?
Whose are these keys?
Whose turn is it?
Why do you say that?
Why don't I help you?
How does this work?
How was your exam?
see examples below
How far is Pattaya from Bangkok?
How long will it take?
How many cars are there?
How much money do you have?
How old are you?
How come I can't see her?
[root@quickstart-bigdata ~]# █
```

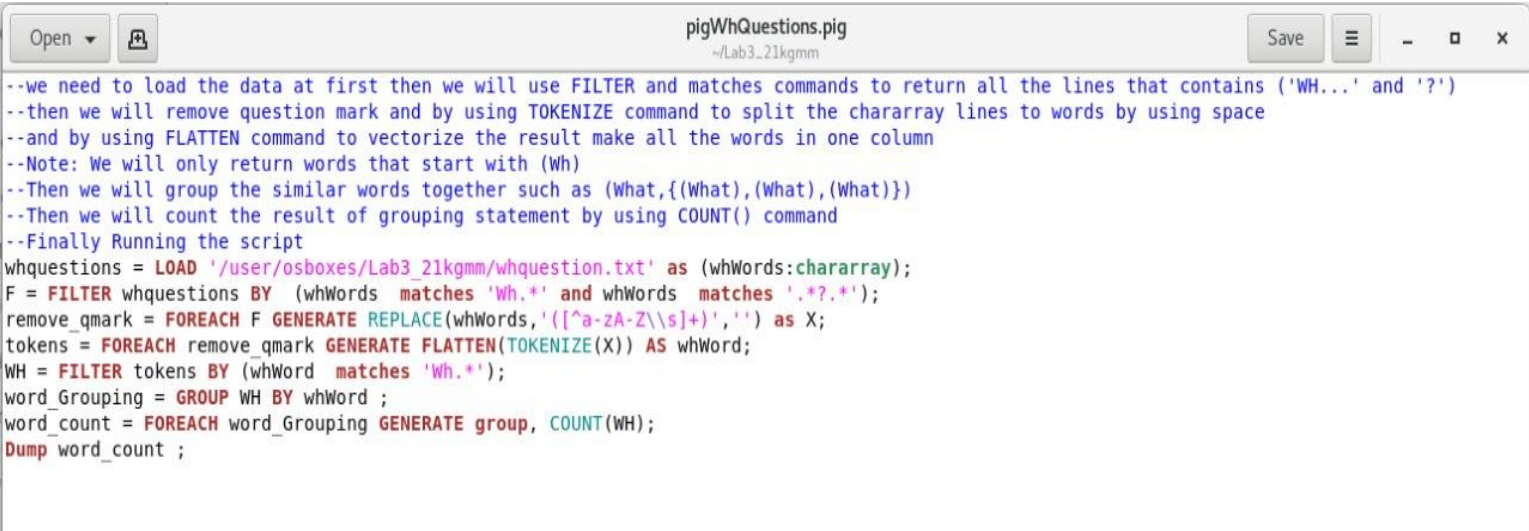
3-- Write commands in PIG to count only the frequency of the WH question words (What, Which, Who, Whom, Why...) from the question file you put in hdfs location.

```
grunt> whquestions = LOAD '/user/osboxes/Lab3_21kgmm/whquestion.txt' as (whWords:chararray);
grunt> F = FILTER whquestions BY (whWords matches 'wh.*' and whWords matches '.*?.*');
grunt> remove_qmark = FOREACH F GENERATE REPLACE (whWords, '([a-zA-Z\s]+)', '') as X;
grunt> tokens = FOREACH remove_qmark GENERATE FLATTEN( TOKENIZE (X)) AS whWord;
```

```
2022-06-25 11:31:44,485 [main] INFO org.apache.pig.impl.util.SpillableMemoryManager -
Selected heap (PS Old Gen) of size 699400192 to monitor. collectionUsageThreshold = 489
580128, usageThreshold = 489580128
```

```
grunt> WH = FILTER tokens BY (whWord matches 'Wh.*');
grunt> word_Grouping = GROUP WH BY whWord ;
grunt> word_count = FOREACH word_Grouping GENERATE group, COUNT(WH);
grunt> Dump word_count ;
```

4-- Accumulate all the commands in a script file called pigwhquestion.pig



```
--we need to load the data at first then we will use FILTER and matches commands to return all the lines that contains ('WH...' and '?')
--then we will remove question mark and by using TOKENIZE command to split the chararray lines to words by using space
--and by using FLATTEN command to vectorize the result make all the words in one column
--Note: We will only return words that start with (Wh)
--Then we will group the similar words together such as (What,{(What),(What),(What)})
--Then we will count the result of grouping statement by using COUNT() command
--Finally Running the script
whquestions = LOAD '/user/osboxes/Lab3_21kgmm/whquestion.txt' as (whWords:chararray);
F = FILTER whquestions BY (whWords matches 'Wh.*' and whWords matches '.*?.*');
remove_qmark = FOREACH F GENERATE REPLACE(whWords,'([a-zA-Z\s]+)','') as X;
tokens = FOREACH remove_qmark GENERATE FLATTEN(TOKENIZE(X)) AS whWord;
WH = FILTER tokens BY (whWord matches 'Wh.*');
word_Grouping = GROUP WH BY whWord ;
word_count = FOREACH word_Grouping GENERATE group, COUNT(WH);
Dump word_count ;
```

5-- Run the script file to generate the final outcome at once

```
[root@quickstart-bigdata ~]# pig /home/osboxes/Lab3_21kgmm/pigWhQuestions.pig
WARNING: Use "yarn jar" to launch YARN applications.
log4j:WARN No appenders could be found for logger (org.apache.hadoop.util.Shell)
.
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more in
fo.
```

```
(Who,1)
(Why,2)
(What,3)
(When,1)
(Whom,1)
(Where,1)
(Which,1)
(Whose,2)
2022-06-25 12:51:55,038 [main] INFO org.apache.pig.Main - Pig script completed in 49 seconds and 687 milliseconds (49687 ms)
```



PigOutput - Notepad

File Edit Format View Help

```
(Who,1)
(Why,2)
(What,3)
(When,1)
(Whom,1)
(Where,1)
(Which,1)
(Whose,2)
.
```

7-- Use the same input data in **whquestion.txt** from within Hive and give the appropriate SQL commands to solve the same word frequency count problem above. Refer to any existing material you found on the web.

Source:

```
hive> CREATE DATABASE whquestiondb;
OK
Time taken: 24.996 seconds
hive> show databases;
OK
default
whquestiondb
Time taken: 4.333 seconds, Fetched: 2 row(s)
hive> use whquestiondb;
OK
Time taken: 0.289 seconds
hive> create external table if not exists whquestion
    > (Wh string)
    > row format delimited
    > fields terminated by ' '
    > location '/user/osboxes/Lab3_21kgmm';
OK
Time taken: 1.653 seconds
hive> Describe whquestion;
OK
wh                                string
Time taken: 0.274 seconds, Fetched: 1 row(s)
hive> load data inpath '/user/osboxes/Lab3_21kgmm/whquestion.txt' into table whquestion
;
Loading data to table whquestiondb.whquestion
```

```
hive> select Wh, count(Wh)
    > from whquestion
    > where Wh REGEXP "^Wh*" group by (Wh) ;
Query ID = osboxes_20220625132430_541bb944-c076-4fe7-b66
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input split size
In order to change the average load for a reducer (in bytes),
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
22/06/25 13:24:34 INFO client.RMProxy: Connecting to ResourceManager/192.168.159.128:8032
```



```

Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-06-25 13:25:22,757 Stage-1 map = 0%, reduce = 0%
2022-06-25 13:25:50,529 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 13.98 sec
2022-06-25 13:26:06,473 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 19.9 sec
MapReduce Total cumulative CPU time: 19 seconds 900 msec
Ended Job = job_1656129011670_0005
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 19.9 sec HDFS Read: 8736 HDFS Write: 240 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 19 seconds 900 msec
OK
What      3
When      1
Where     1
Which     1
Who       1
Whom      1
Whose     2
Why       2
Time taken: 99.154 seconds, Fetched: 8 row(s)
hive> █

```

```

hive> [osboxes@quickstart-bigdata ~]$ hive -f /home/osboxes/Lab3_21kgmm/hiveSql.sql
WARNING: Use "yarn jar" to launch YARN applications.
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding for [org.apache.hadoop.hive.shims.HiveShims.class]

```

```

CREATE DATABASE whquestiondb;
show databases;
use whquestiondb;
create external table if not exists whquestion
(Wh string)
row format delimited
fields terminated by ' '
location '/user/osboxes/Lab3_21kgmm';
Describe whquestion;
load data inpath '/user/osboxes/Lab3_21kgmm/whquestion.txt' into table whquestion;
select Wh, count(Wh)
from whquestion
where Wh REGEXP "^Wh*" group by (Wh) ;

```


Findings:

- Pig is used to perform all kinds of data manipulation operations in Hadoop. And it is abstract over MapReduce. And Pig does not support schema to store data. And not support partitioning.
- Hive is built on the top of Hadoop and is used to process structured data in Hadoop. So we should create database to store our data as input, but in pig it works on any type of data. And Hive supports schema for data insertion in tables. And supports partitioning.

Note: Hive is easier than Pig in commands, but it loads data slowly.

Sources :

- 1- [hive Tutorial => Word Count Example in Hive \(riptutorial.com\)](http://riptutorial.com)
- 2- [hive Tutorial - SELECT Statement \(sodocumentation.net\)](http://sodocumentation.net)
- 3- [Difference between Pig and Hive - GeeksforGeeks](http://GeeksforGeeks)
- 4- [Apache Pig - Running Scripts \(tutorialspoint.com\)](http://tutorialspoint.com)