Big Data Hadoop—Real Time Project— Social Media

Jayateertha M Tatti

joy.tat@gmail.com

25-Oct-2016

Analyze data set from Stack Exchange

As part of a recruiting exercise of the biggest social media company, they asked candidates to analyze data set from Stack Exchange. We will be using similar data set to arrive at certain key insights..

Project goals:

1) Top 10 most commonly used tags in this data set
2) Average time to answer questions
3) Number of questions which got answered within 1 hour
4) Tags of questions which got answered within 1 hour

Data set in xls format from the URL :

http://www.ics.uci.edu/~duboisc/stackoverflow/answers.csv

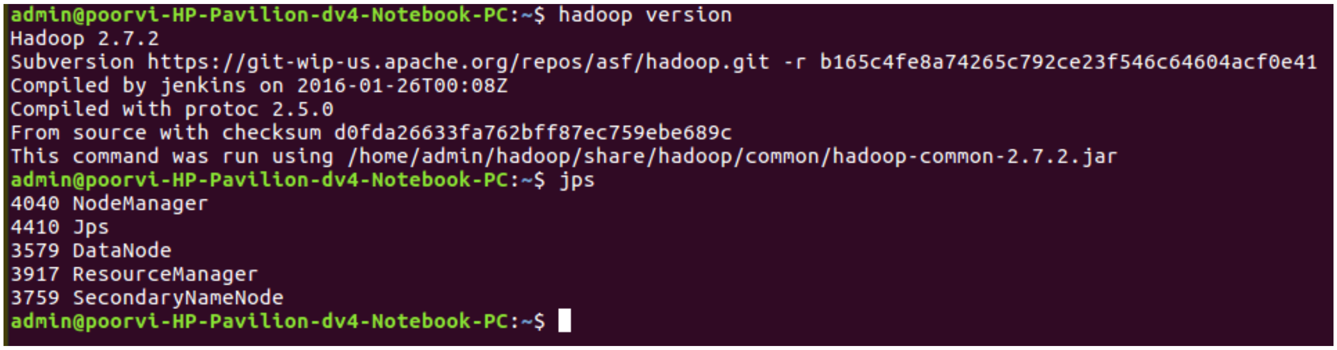
The data set contains the following attributes:

- qid: Unique question id
- i: User id of questioner
- qs: Score of the question
- qt: Time of the question (in epoch time)
- tags: a comma-separated list of the tags associated with the question.
- used on each question.
- qvc: Number of views of this question (at the time of the datadump)
- qac: Number of answers for this question (at the time of the datadump)
- aid: Unique answer id
- j: User id of answerer
- as: Score of the answer
- at: Time of the answer (in epoch time)

Note : I have renamed column with meaningful name.

Hadoop Architecture

- Vmware
- Linux Ubuntu 16.04 Lts
- Hadoop 2.7.2



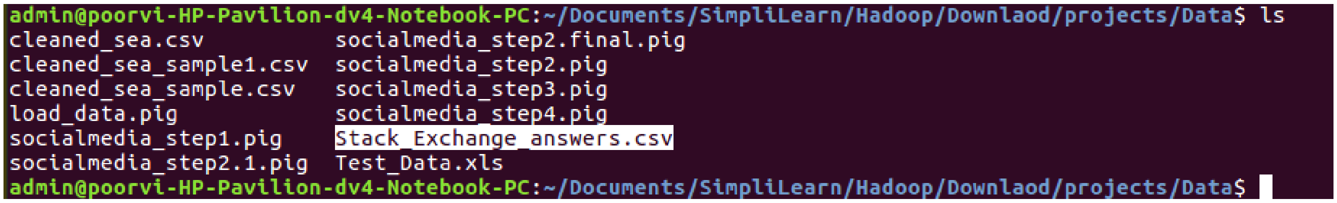
```
admin@poorvi-HP-Pavilion-dv4-Notebook-PC:~$ hadoop version
Hadoop 2.7.2
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r b165c4fe8a74265c792ce23f546c64604acf0e41
Compiled by jenkins on 2016-01-26T00:08Z
Compiled with protoc 2.5.0
From source with checksum d0fda26633fa762bff87ec759ebe689c
This command was run using /home/admin/hadoop/share/hadoop/common/hadoop-common-2.7.2.jar
admin@poorvi-HP-Pavilion-dv4-Notebook-PC:~$ jps
4040 NodeManager
4410 Jps
3579 DataNode
3917 ResourceManager
3759 SecondaryNameNode
admin@poorvi-HP-Pavilion-dv4-Notebook-PC:~$ 
```

Approach

- Referred lesson 3 to load the dataset in HDFS
- Referred lesson 6 Pig to arrive at the below results detailed in following pages

Download dataset

Refer image below: Stack_Exchange_answer.csv has 263541 rows of data.



```
admin@poorvi-HP-Pavilion-dv4-Notebook-PC:~/Documents/SimpliLearn/Hadoop/Downlaod/projects/Data$ ls
cleaned_sea.csv          socialmedia_step2.final.pig
cleaned_sea_sample1.csv  socialmedia_step2.pig
cleaned_sea_sample.csv   socialmedia_step3.pig
load_data.pig            socialmedia_step4.pig
socialmedia_step1.pig    Stack_Exchange_answers.csv
socialmedia_step2.1.pig  Test_Data.xls
admin@poorvi-HP-Pavilion-dv4-Notebook-PC:~/Documents/SimpliLearn/Hadoop/Downlaod/projects/Data$ 
```

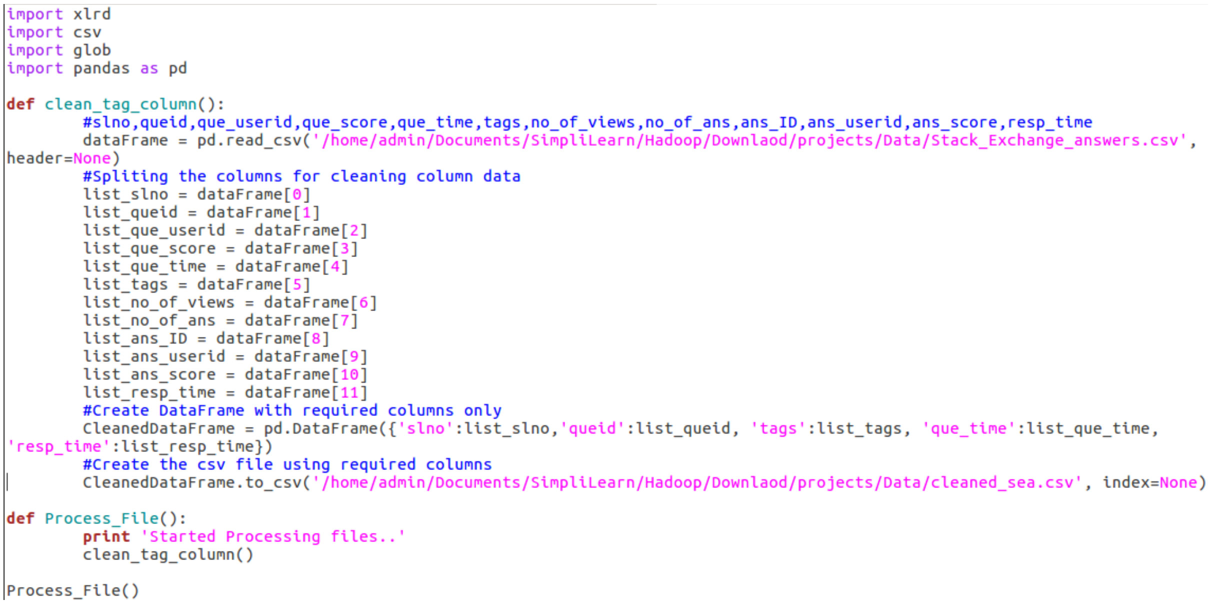
Additional step performed:

I call this as data cleaning step.
The original file Stack_Exchange_answers.csv. The size is 24.5 MB. The number of records are 263541. This file contains unwanted column data too. Hence decided to use only required columns for project analysis. Wrote python program to retain the following columns
slno, queid, tags, que_time, resp_time. These are renamed for easy reference of column names. The output generated is cleaned_sea.csv. This file size is 16.1MB. The number of records are 263541.

- Upload the cleaned_sea.csv file to cloudlab in social folder
- Execute scripts on this file.

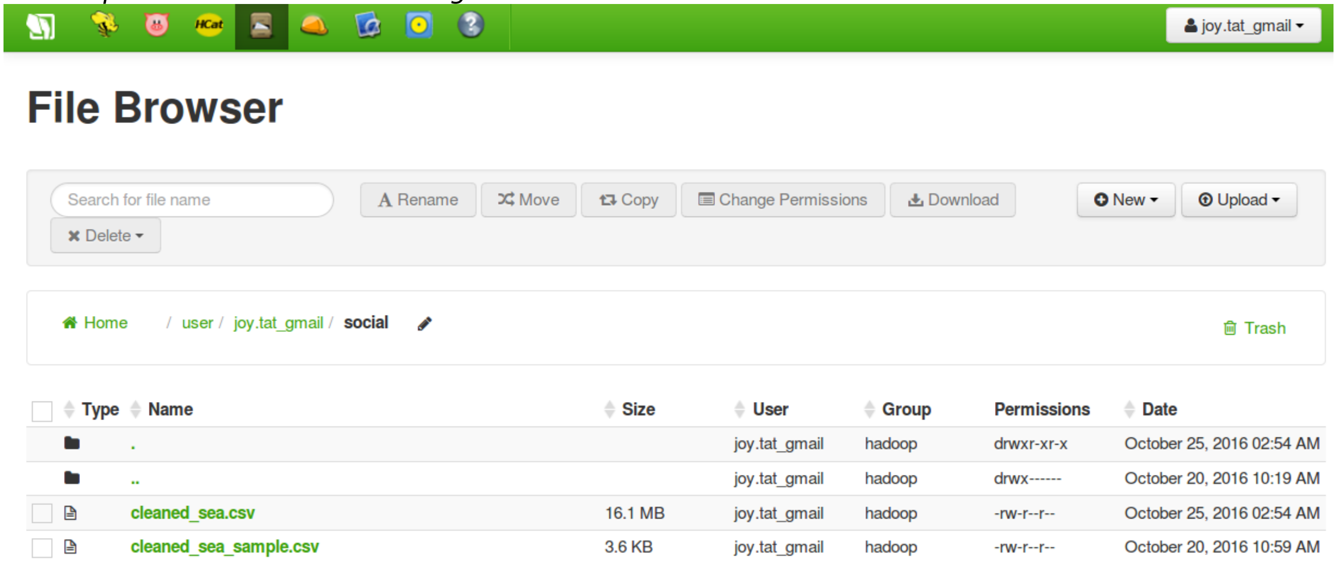
These above steps helped me to speed up the processing. And also provided me an opportunity to enhance my python skills.

Create program named : clean_sea_file.py
Refer the below image



```
import xlrd
import csv
import glob
import pandas as pd

def clean_tag_column():
        #slno,queid,que_userid,que_score,que_time,tags,no_of_views,no_of_ans,ans_ID,ans_userid,ans_score,resp_time
        dataFrame = pd.read_csv('/home/admin/Documents/SimpliLearn/Hadoop/Downlaod/projects/Data/Stack_Exchange_answers.csv',
header=None)
        #Spliting the columns for cleaning column data
        list_slno = dataFrame[0]
        list_queid = dataFrame[1]
        list_que_userid = dataFrame[2]
        list_que_score = dataFrame[3]
        list_que_time = dataFrame[4]
        list_tags = dataFrame[5]
        list_no_of_views = dataFrame[6]
        list_no_of_ans = dataFrame[7]
        list_ans_ID = dataFrame[8]
        list_ans_userid = dataFrame[9]
        list_ans_score = dataFrame[10]
        list_resp_time = dataFrame[11]
        #Create DataFrame with required columns only
        CleanedDataFrame = pd.DataFrame({'slno':list_slno,'queid':list_queid, 'tags':list_tags, 'que_time':list_que_time,
'resp_time':list_resp_time})
        #Create the csv file using required columns
        CleanedDataFrame.to_csv('/home/admin/Documents/SimpliLearn/Hadoop/Downlaod/projects/Data/cleaned_sea.csv', index=None)

def Process_File():
        print 'Started Processing files..'
        clean_tag_column()

Process_File()
```

Refer Uploaded CloudLab file image



Type	Name	Size	User	Group	Permissions	Date
	.		joy.tat_gmail	hadoop	drwxr-xr-x	October 25, 2016 02:54 AM
	..		joy.tat_gmail	hadoop	drwx------	October 20, 2016 10:19 AM
	cleaned_sea.csv	16.1 MB	joy.tat_gmail	hadoop	-rw-r--r--	October 25, 2016 02:54 AM
	cleaned_sea_sample.csv	3.6 KB	joy.tat_gmail	hadoop	-rw-r--r--	October 20, 2016 10:59 AM

Analysis 1:

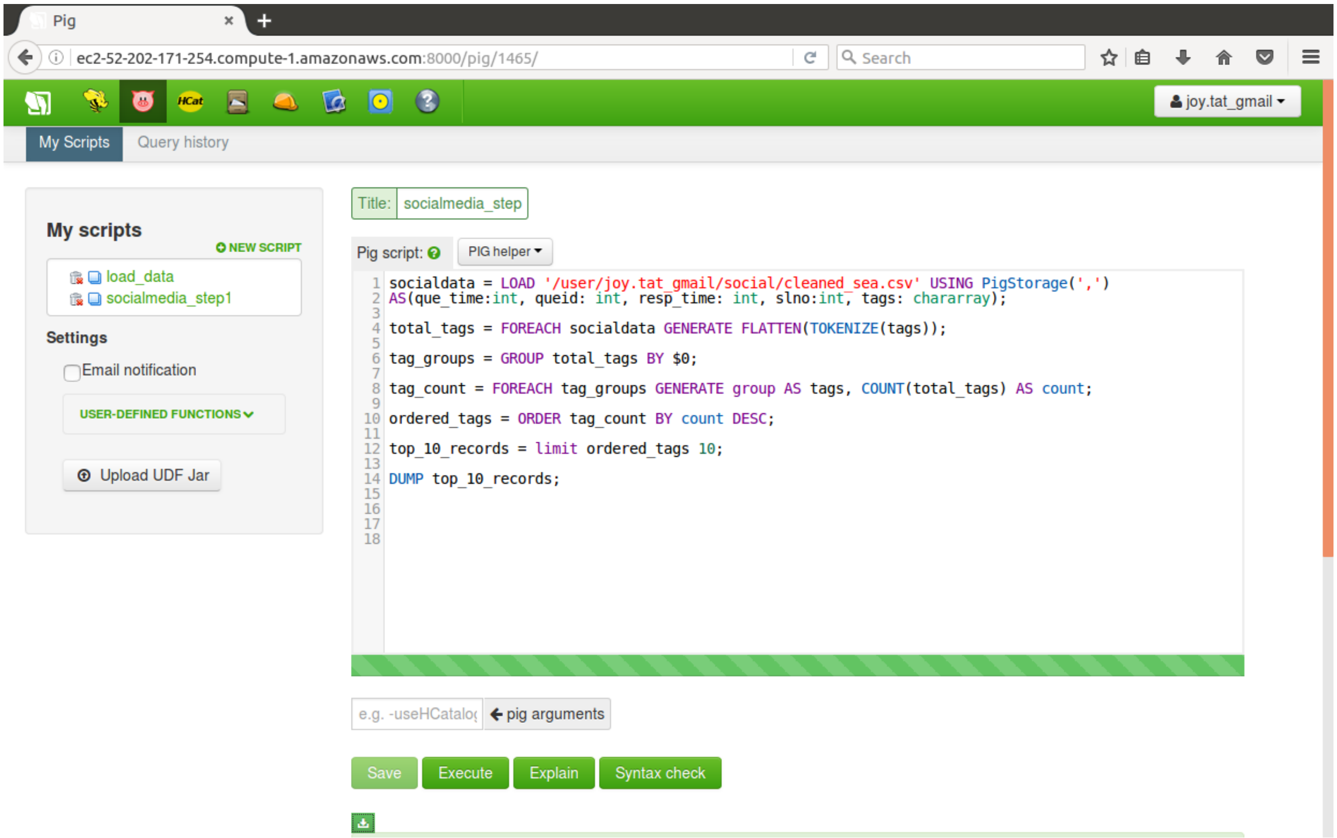
- Analyze the entire data set and arrive Top 10 most commonly used tags in this data set
- Input the data in HDFS
- Use Pig to arrive at the conclusions

Query is :

```
socialdata = LOAD '/user/joy.tat_gmail/social/cleaned_sea.csv' USING PigStorage(',')
AS(que_time:int, queid: int, resp_time: int, slno:int, tags: chararray);

total_tags = FOREACH socialdata GENERATE FLATTEN(TOKENIZE(tags));

tag_groups = GROUP total_tags BY $0;

tag_count = FOREACH tag_groups GENERATE group AS tags, COUNT(total_tags) AS count;

ordered_tags = ORDER tag_count BY count DESC;
```

```
top_10_records = limit ordered_tags 10;

DUMP top_10_records;
```

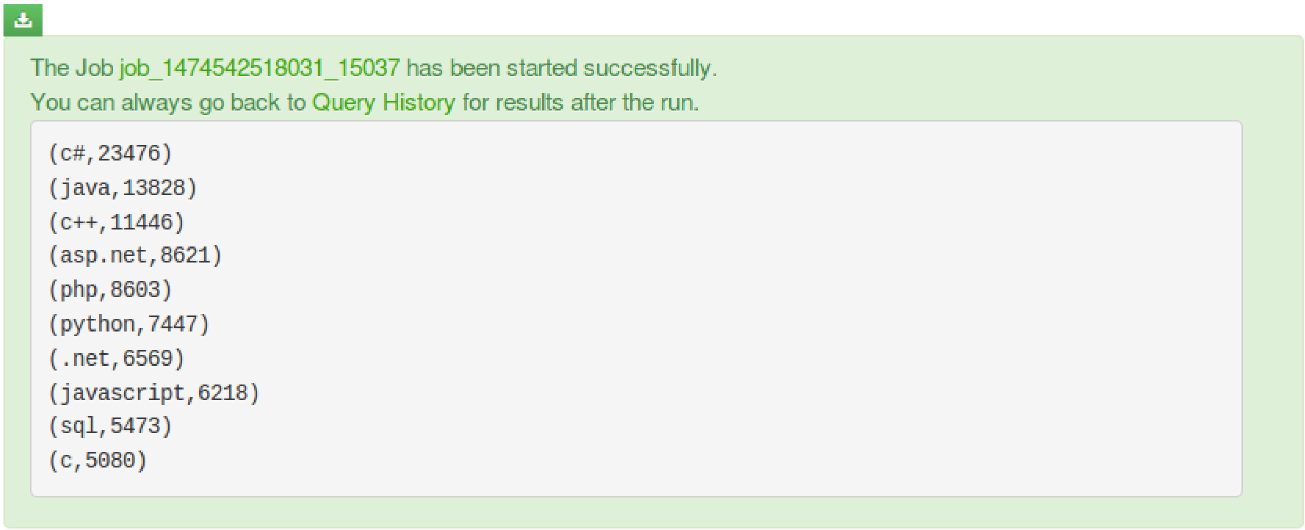
Social Media Step1 Query



Title: socialmedia_step

```
socialdata = LOAD '/user/joy.tat_gmail/social/cleaned_sea.csv' USING PigStorage(',')
AS(que_time:int, queid: int, resp_time: int, slno:int, tags: chararray);

total_tags = FOREACH socialdata GENERATE FLATTEN(TOKENIZE(tags));

tag_groups = GROUP total_tags BY $0;

tag_count = FOREACH tag_groups GENERATE group AS tags, COUNT(total_tags) AS count;

ordered_tags = ORDER tag_count BY count DESC;

top_10_records = limit ordered_tags 10;

DUMP top_10_records;
```

Results



The Job job_1474542518031_15037 has been started successfully.
You can always go back to Query History for results after the run.

```
(c#,23476)
(java,13828)
(c++,11446)
(asp.net,8621)
(php,8603)
(python,7447)
(.net,6569)
(javascript,6218)
(sql,5473)
(c,5080)
```

Analysis 2:

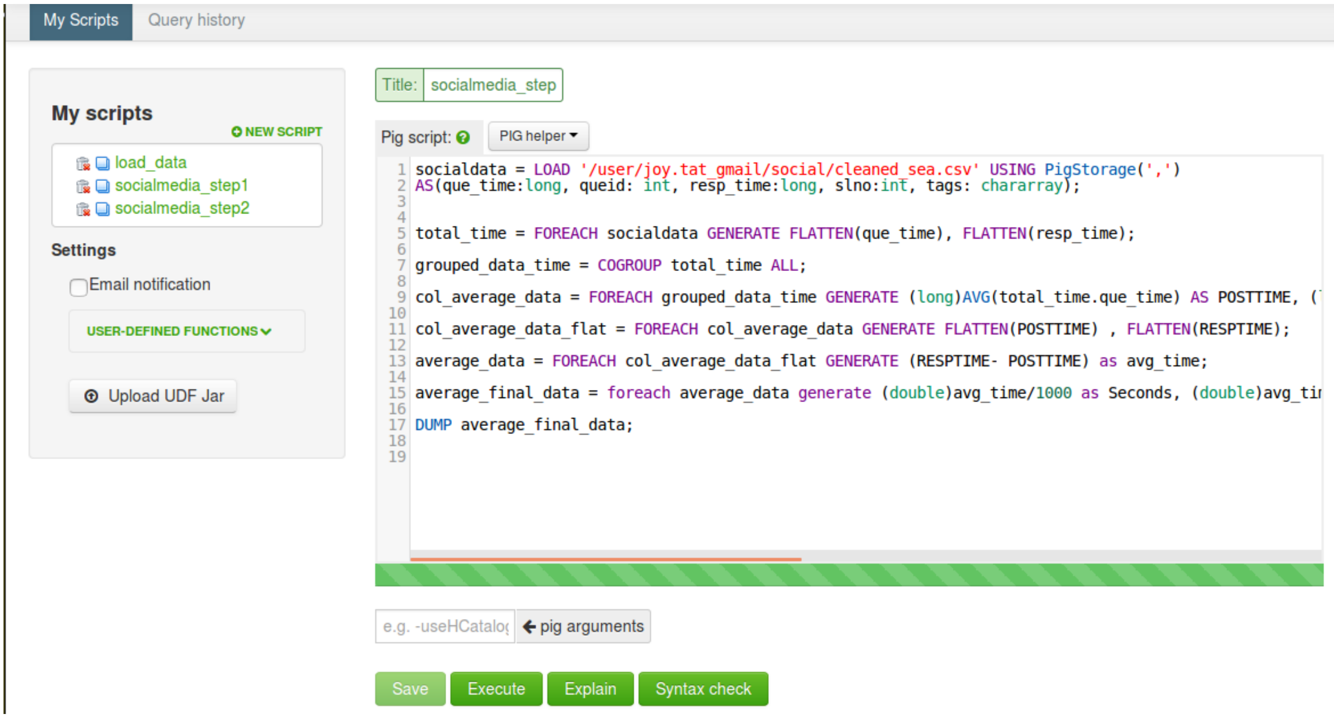
- Analyze the entire data set and arrive average time to answer questions

- Input the data in HDFS
- Use Pig to arrive at the conclusions

Query is:

```
socialdata = LOAD '/user/joy.tat_gmail/social/cleaned_sea.csv' USING PigStorage(',')
AS(que_time:long, queid: int, resp_time:long, slno:int, tags: chararray);


total_time = FOREACH socialdata GENERATE FLATTEN(que_time), FLATTEN(resp_time);

grouped_data_time = COGROUP total_time ALL;

col_average_data = FOREACH grouped_data_time GENERATE (long)AVG(total_time.que_time) AS POSTTIME,
(long)AVG(total_time.resp_time) AS RESPTIME;

col_average_data_flat = FOREACH col_average_data GENERATE FLATTEN(POSTTIME) , FLATTEN(RESPTIME);

average_data = FOREACH col_average_data_flat GENERATE (RESPTIME- POSTTIME) as avg_time;

average_final_data = foreach average_data generate (double)avg_time/1000 as Seconds,
(double)avg_time/(1000*60) as Minutes, (double)avg_time/(1000*60*60) as Hours, (double)avg_time/
(1000*60*60*24) as Days;

DUMP average_final_data;
```

Social Media Step 2 Query



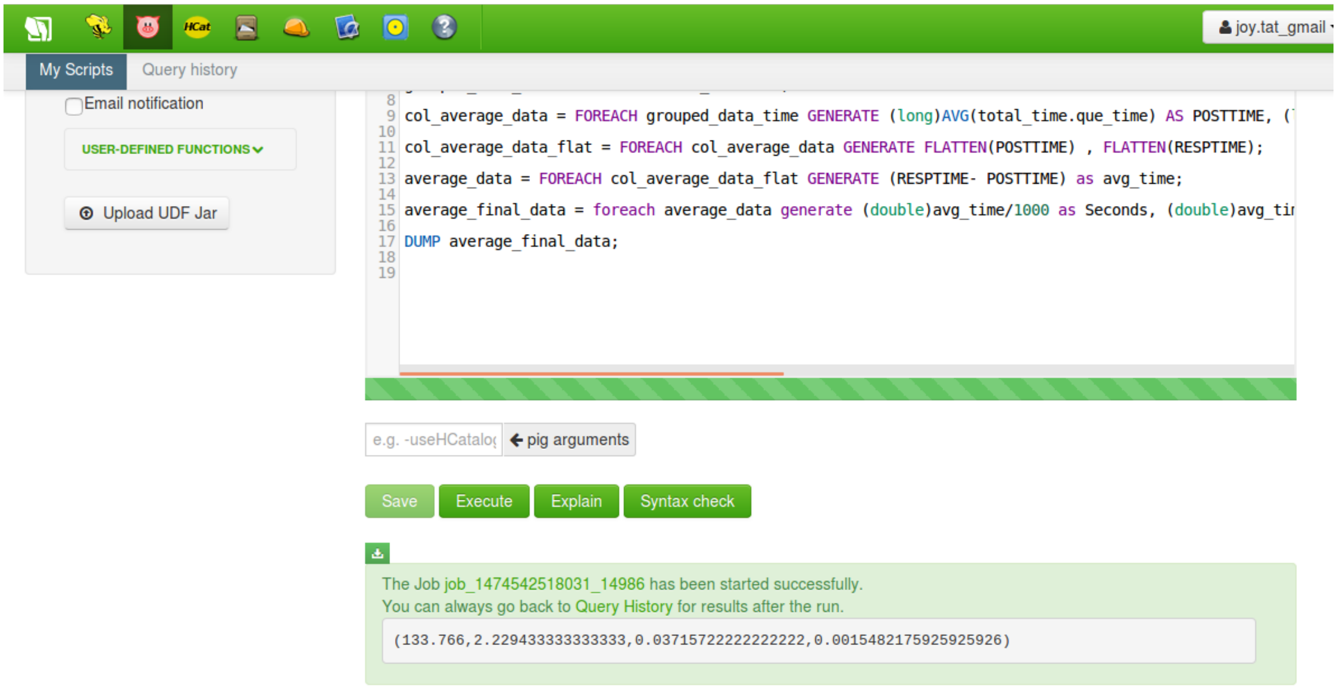
Title: socialmedia_step

```
socialdata = LOAD '/user/joy.tat_gmail/social/cleaned_sea.csv' USING PigStorage(',')
AS(que_time:long, queid: int, resp_time:long, slno:int, tags: chararray);


total_time = FOREACH socialdata GENERATE FLATTEN(que_time), FLATTEN(resp_time);

grouped_data_time = COGROUP total_time ALL;

col_average_data = FOREACH grouped_data_time GENERATE (long)AVG(total_time.que_time) AS POSTTIME, (

col_average_data_flat = FOREACH col_average_data GENERATE FLATTEN(POSTTIME) , FLATTEN(RESPTIME);

average_data = FOREACH col_average_data_flat GENERATE (RESPTIME- POSTTIME) as avg_time;

average_final_data = foreach average_data generate (double)avg_time/1000 as Seconds, (double)avg_tir

DUMP average_final_data;
```

Results:

Average response in seconds : 133.766

Average response in minutes: 2.229



```
col_average_data = FOREACH grouped_data_time GENERATE (long)AVG(total_time.que_time) AS POSTTIME, (

col_average_data_flat = FOREACH col_average_data GENERATE FLATTEN(POSTTIME) , FLATTEN(RESPTIME);

average_data = FOREACH col_average_data_flat GENERATE (RESPTIME- POSTTIME) as avg_time;

average_final_data = foreach average_data generate (double)avg_time/1000 as Seconds, (double)avg_ti

DUMP average_final_data;
```

The Job job_1474542518031_14986 has been started successfully.
You can always go back to Query History for results after the run.

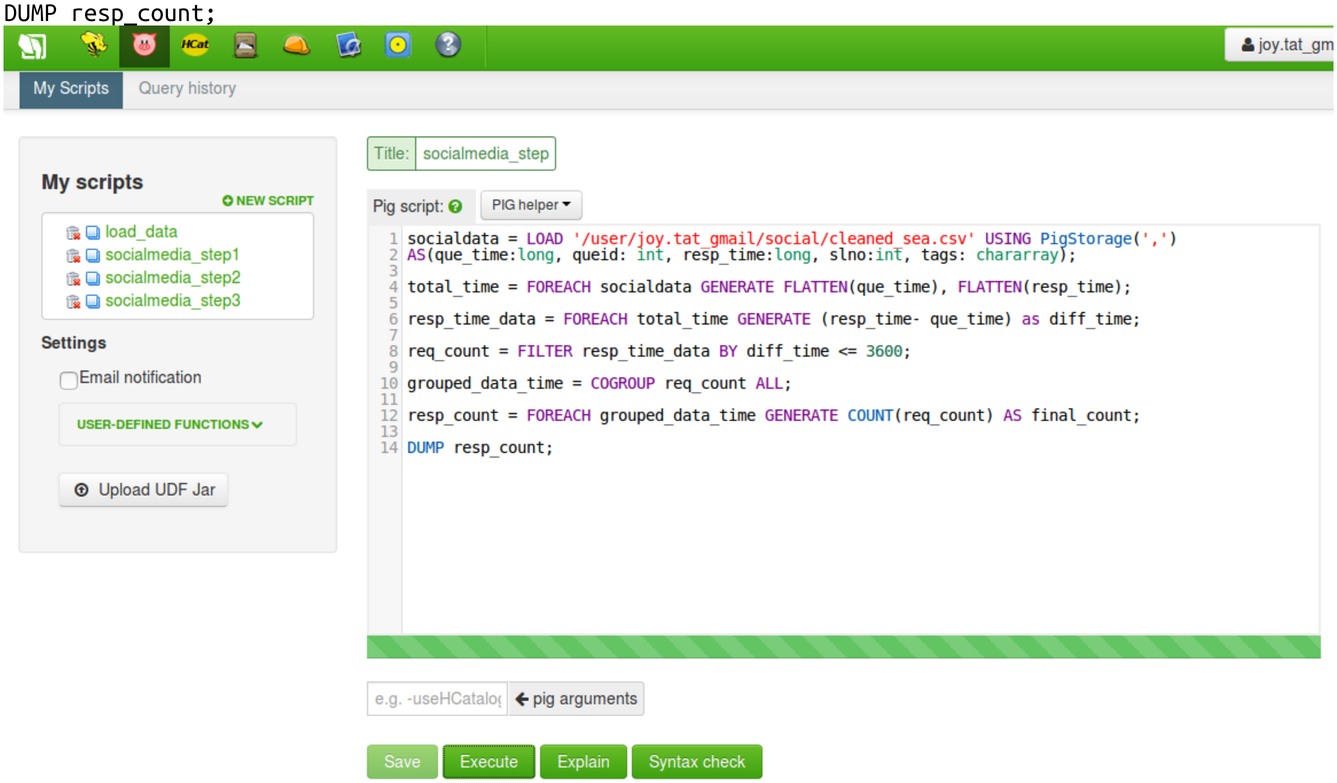
```
(133.766,2.229433333333333,0.03715722222222222,0.0015482175925925926)
```

Analysis 3:

- Analyze the entire data set and arrive number of questions which got answered within 1 hour
- Input the data in HDFS
- Use Pig to arrive at the conclusions

Query is :

```
socialdata = LOAD '/user/joy.tat_gmail/social/cleaned_sea.csv' USING PigStorage(',')
AS(que_time:long, queid: int, resp_time:long, slno:int, tags: chararray);

total_time = FOREACH socialdata GENERATE FLATTEN(que_time), FLATTEN(resp_time);

resp_time_data = FOREACH total_time GENERATE (resp_time- que_time) as diff_time;

req_count = FILTER resp_time_data BY diff_time <= 3600;

grouped_data_time = COGROUP req_count ALL;

resp_count = FOREACH grouped_data_time GENERATE COUNT(req_count) AS final_count;

DUMP resp_count;
```



Title: socialmedia_step

My scripts

- load_data
- socialmedia_step1
- socialmedia_step2
- socialmedia_step3

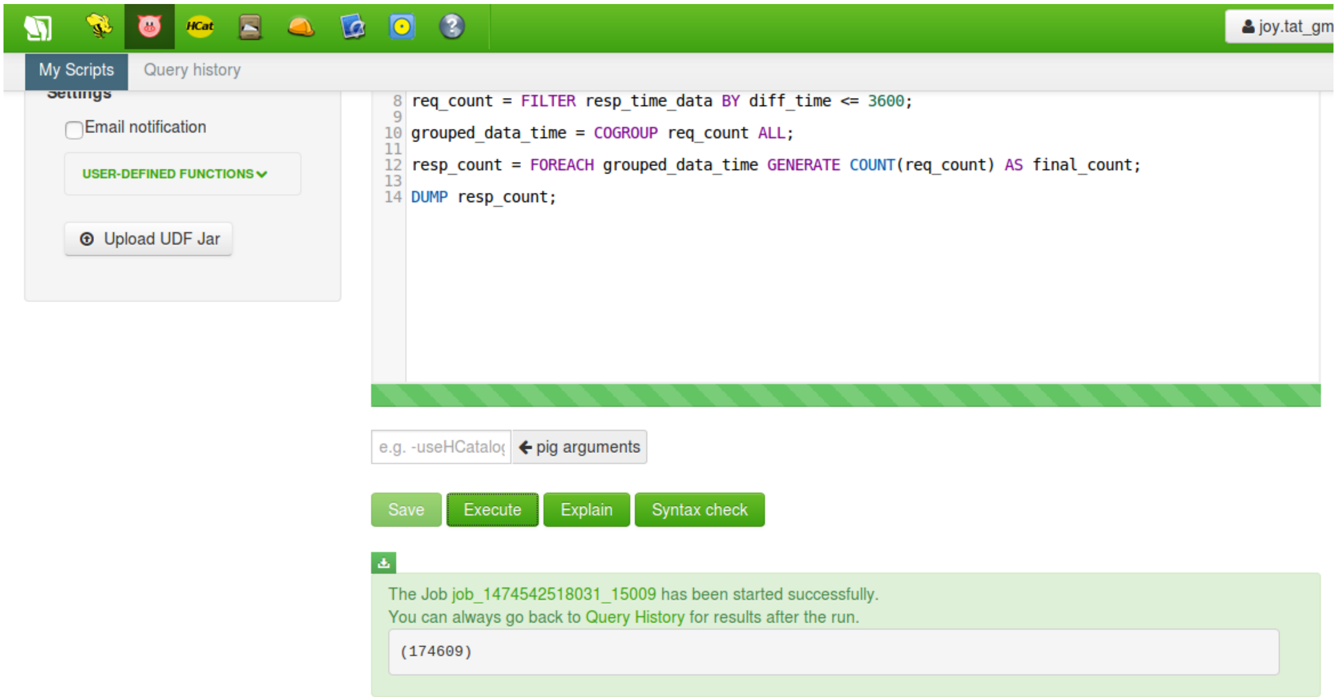
Settings

□ Email notification

```
socialdata = LOAD '/user/joy.tat_gmail/social/cleaned_sea.csv' USING PigStorage(',')
AS(que_time:long, queid: int, resp_time:long, slno:int, tags: chararray);

total_time = FOREACH socialdata GENERATE FLATTEN(que_time), FLATTEN(resp_time);

resp_time_data = FOREACH total_time GENERATE (resp_time- que_time) as diff_time;

req_count = FILTER resp_time_data BY diff_time <= 3600;

grouped_data_time = COGROUP req_count ALL;

resp_count = FOREACH grouped_data_time GENERATE COUNT(req_count) AS final_count;

DUMP resp_count;
```

Results:

Totally 174699 queries were answered in 1 hour.



```
req_count = FILTER resp_time_data BY diff_time <= 3600;

grouped_data_time = COGROUP req_count ALL;

resp_count = FOREACH grouped_data_time GENERATE COUNT(req_count) AS final_count;

DUMP resp_count;
```

The Job job_1474542518031_15009 has been started successfully.
You can always go back to Query History for results after the run.

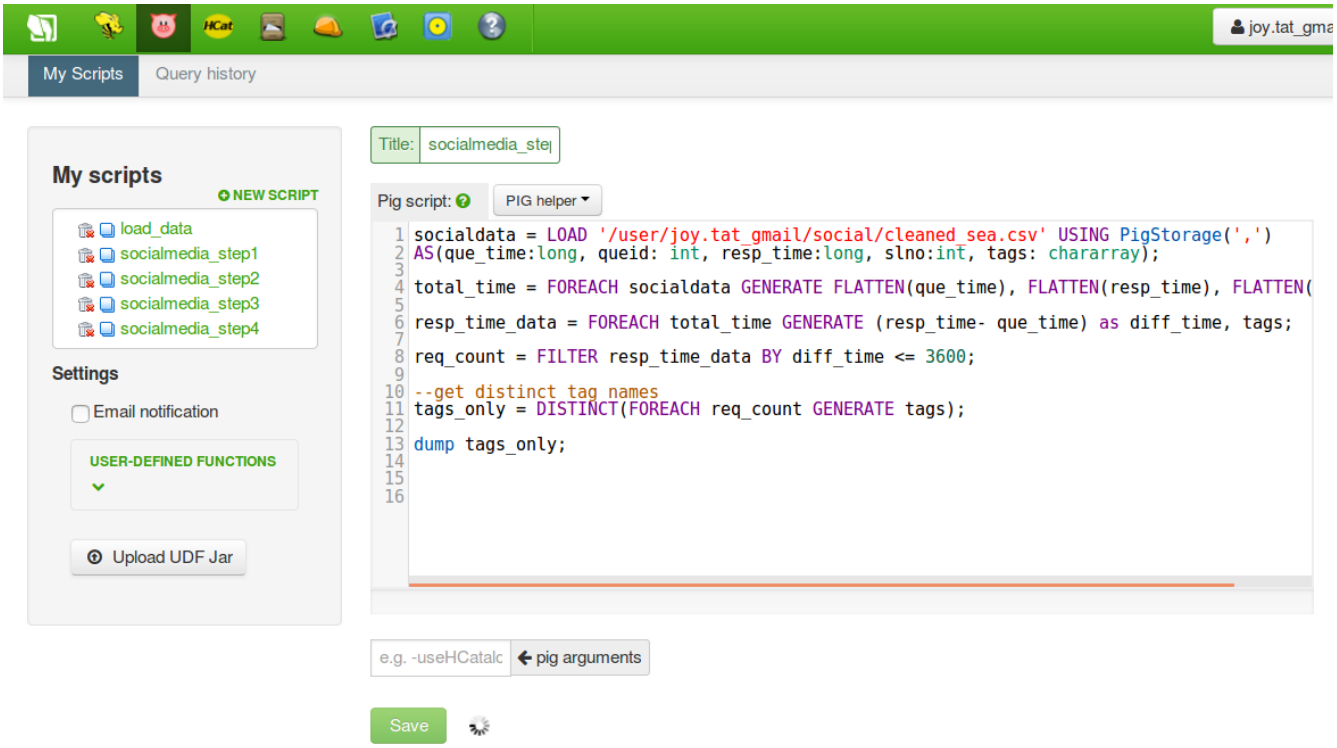
```
(174609)
```

Analysis 4:

- Analyze the entire data set and arrive tags of questions which got answered within 1 hour
- Input the data in HDFS
- Use Pig to arrive at the conclusions

Query is :

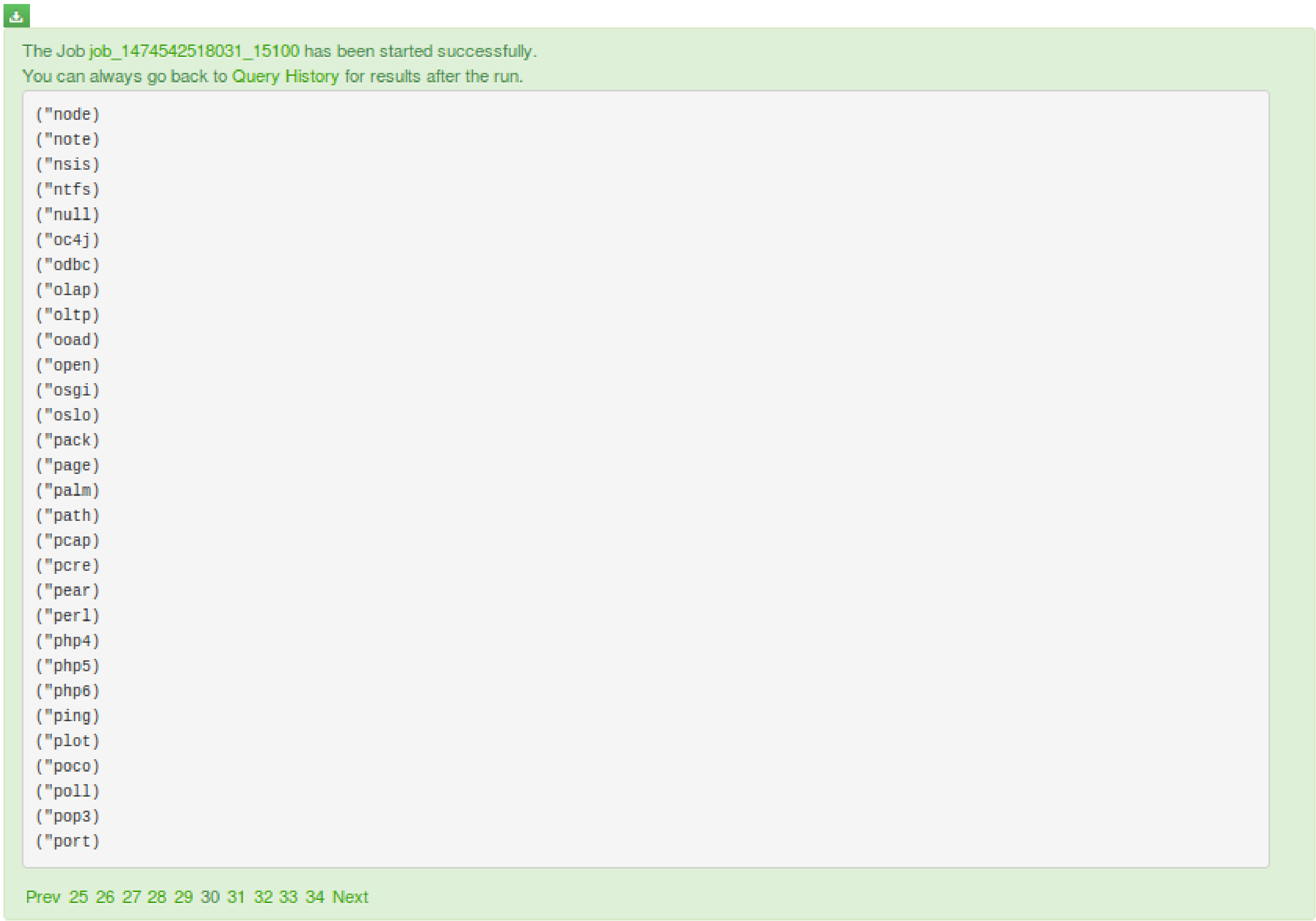
```
socialdata = LOAD '/user/joy.tat_gmail/social/cleaned_sea.csv' USING PigStorage(',')
AS(que_time:long, queid: int, resp_time:long, slno:int, tags: chararray);

total_time = FOREACH socialdata GENERATE FLATTEN(que_time), FLATTEN(resp_time), FLATTEN(tags);

resp_time_data = FOREACH total_time GENERATE (resp_time- que_time) as diff_time, tags;

req_count = FILTER resp_time_data BY diff_time <= 3600;

--get distinct tag names
tags_only = DISTINCT(FOREACH req_count GENERATE tags);

dump tags_only;
```



Title: socialmedia_ste

```
socialdata = LOAD '/user/joy.tat_gmail/social/cleaned_sea.csv' USING PigStorage(',')
AS(que_time:long, queid: int, resp_time:long, slno:int, tags: chararray);

total_time = FOREACH socialdata GENERATE FLATTEN(que_time), FLATTEN(resp_time), FLATTEN(

resp_time_data = FOREACH total_time GENERATE (resp_time- que_time) as diff_time, tags;

req_count = FILTER resp_time_data BY diff_time <= 3600;

--get distinct tag names
tags_only = DISTINCT(FOREACH req_count GENERATE tags);

dump tags_only;
```

Results:



The Job job_1474542518031_15100 has been started successfully.
You can always go back to Query History for results after the run.

```
("node)
("note)
("nsis)
("ntfs)
("null)
("oc4j)
("odbc)
("olap)
("oltp)
("ooad)
("open)
("osgi)
("oslo)
("pack)
("page)
("palm)
("path)
("pcap)
("pcre)
("pear)
("perl)
("php4)
("php5)
("php6)
("ping)
("plot)
("poco)
("poll)
("pop3)
("port)
```