HOME ASSIGNMENT 1

Roll No: 208W1A1299

Name: MOHAMMAD RIZWANULLAH

PROJECT NAME : Twitter data analysis.

The output of Mapreduce will be in below format

The output of the data preprocessing will be to identify the day in the weekend and hour in the day where celebrites are tweeting in twitter

Question answered:

1. What hour of the day does @PrezOno's tweet the most on average, using every day we have twitter data? Directory - https://github.uc.edu/loganasr/HadoopMapReduce-Twitter/tree/master/TweetsByHour
2. What day of the week does @PrezOno tweet the most on average? Use the same example as in #1 but for days of the week. Directory - https://github.uc.edu/loganasr/HadoopMapReduce-Twitter/tree/master/TweetsByDay
3. How does @PrezOno's tweet length compare to the average of all others? What is his average length? All others? Directory - https://github.uc.edu/loganasr/HadoopMapReduce-Twitter/tree/master/TweetLength

Instructions:

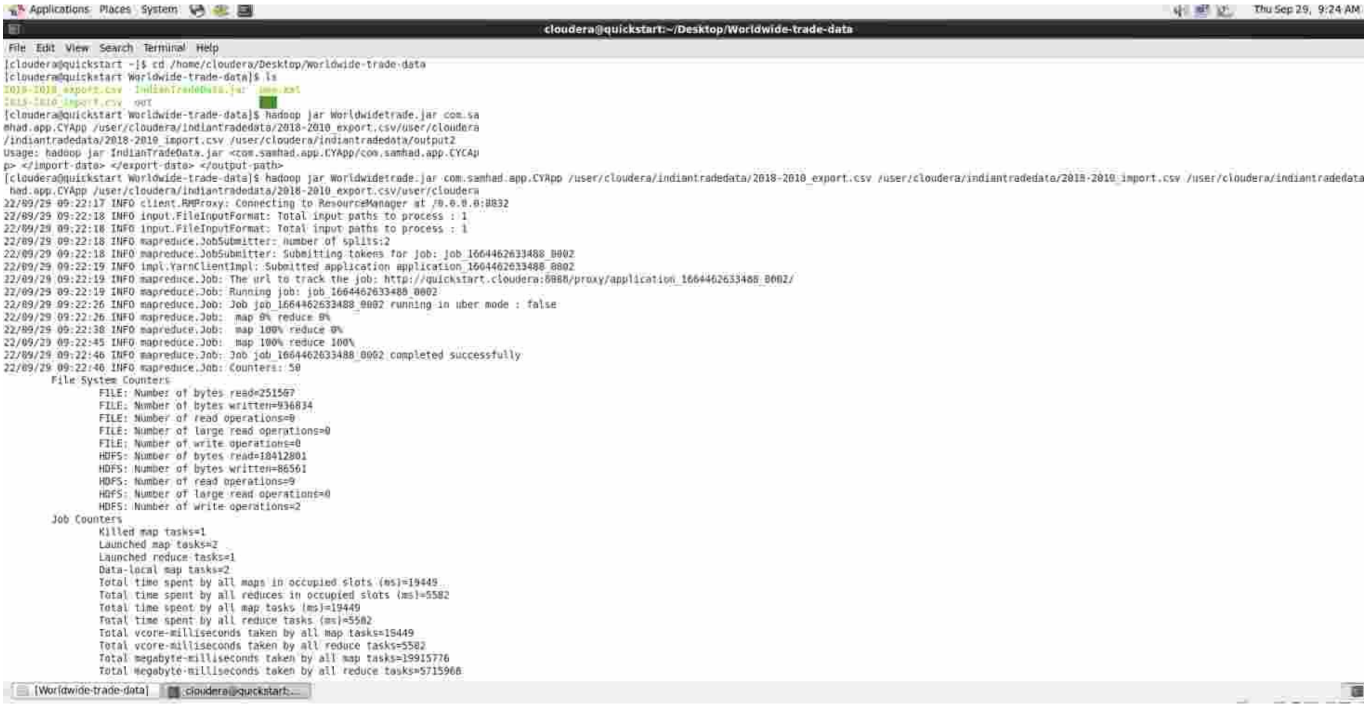
A sample data file has been included in /data directory to support quick validations through the Hadoop streaming mode. However, the file does not contain tweets from @PrezOno and hence, it would be necessary update the user_name for filtering the tweets.

Sample command: cat /data/sample-data | ./mapTweetsByHour.py | sort | ./reduceTweetsByHour.py

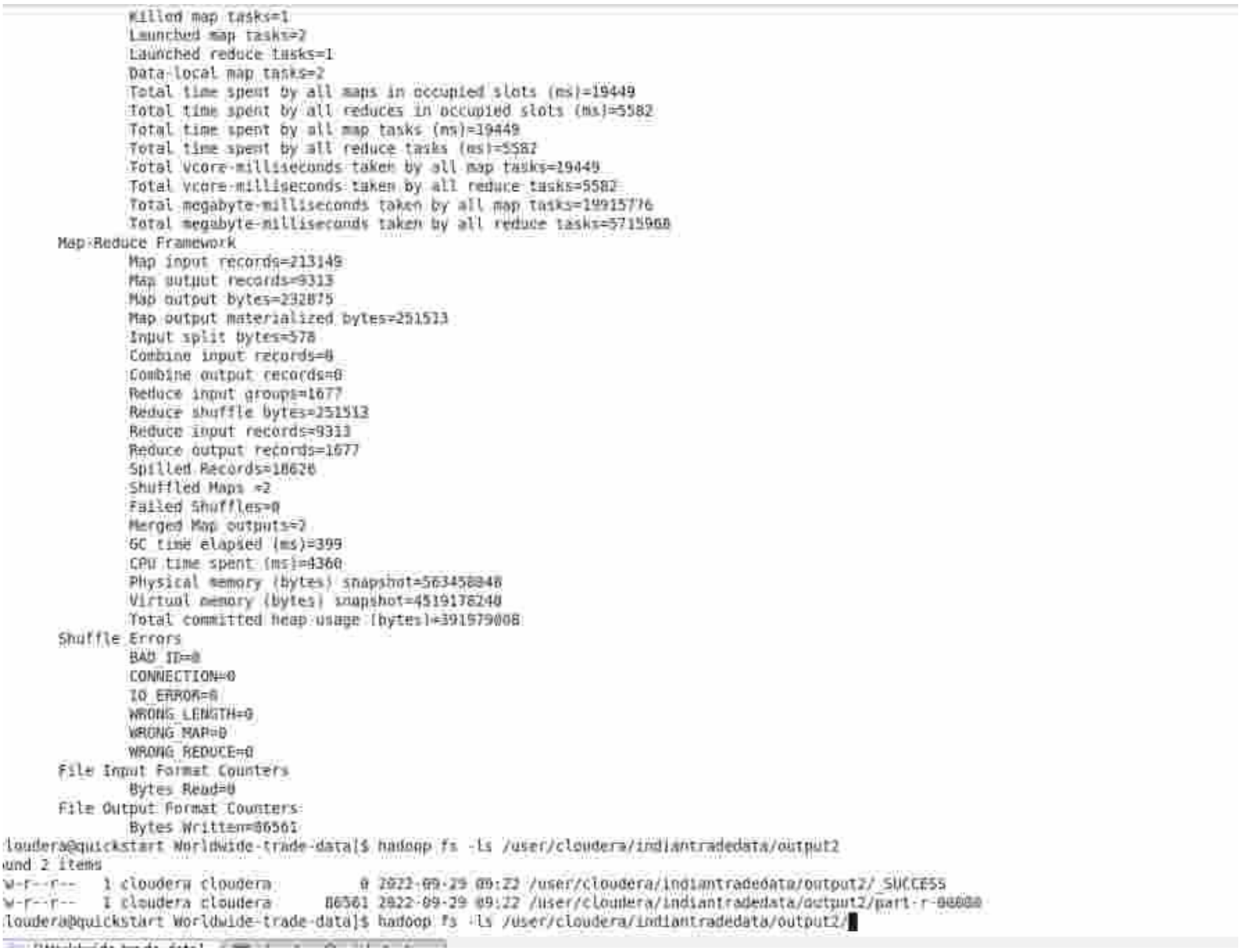
To run the map reduce programs in the hadoop cluster, utilize the following command.

hadoop jar /root/hadoop-2.7.1/share/hadoop/tools/lib/hadoop-streaming-2.7.1.jar -input /data/twitter -output myoutput -file *.py -mapper mapTweetsByHour.py -reducer reduceTweetsByhour.py

Execution Screenshots:


Applications Places System
Thu Sep 29, 9:24 AM
cloudera@quickstart:~/Desktop/Worldwide-trade-data
File Edit View Search Terminal Help
[Worldwide-trade-data]
cloudera@quickstart:...

```
[cloudera@quickstart ~]$ cd /home/cloudera/Desktop/Worldwide-trade-data
[cloudera@quickstart Worldwide-trade-data]$ ls
2018-2010_export.csv  IndianTradeData.jar  pom.xml
2018-2010_import.csv  out
[cloudera@quickstart Worldwide-trade-data]$ hadoop jar Worldwidetrade.jar com.sa
mhad.app.CYApp /user/cloudera/indiantradedata/2018-2010_export.csv/user/cloudera
/indiantradedata/2018-2010_import.csv /user/cloudera/indiantradedata/output2
Usage: hadoop jar IndianTradeData.jar <com.samhad.app.CYApp/com.samhad.app.CYCAp
p> </import-data> </export-data> </output-path>
[cloudera@quickstart Worldwide-trade-data]$ hadoop jar Worldwidetrade.jar com.samhad.app.CYApp /user/cloudera/indiantradedata/2018-2010_export.csv /user/cloudera/indiantradedata/2018-2010_import.csv /user/cloudera/indiantradedata
 had.app.CYApp /user/cloudera/indiantradedata/2018-2010_export.csv/user/cloudera
22/09/29 09:22:17 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
22/09/29 09:22:18 INFO input.FileInputFormat: Total input paths to process : 1
22/09/29 09:22:18 INFO input.FileInputFormat: Total input paths to process : 1
22/09/29 09:22:18 INFO mapreduce.JobSubmitter: number of splits:2
22/09/29 09:22:18 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1664462633488_0002
22/09/29 09:22:19 INFO impl.YarnClientImpl: Submitted application application_1664462633488_0002
22/09/29 09:22:19 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1664462633488_0002/
22/09/29 09:22:19 INFO mapreduce.Job: Running job: job_1664462633488_0002
22/09/29 09:22:26 INFO mapreduce.Job: Job job_1664462633488_0002 running in uber mode : false
22/09/29 09:22:26 INFO mapreduce.Job:  map 0% reduce 0%
22/09/29 09:22:38 INFO mapreduce.Job:  map 100% reduce 0%
22/09/29 09:22:45 INFO mapreduce.Job:  map 100% reduce 100%
22/09/29 09:22:46 INFO mapreduce.Job: Job job_1664462633488_0002 completed successfully
22/09/29 09:22:46 INFO mapreduce.Job: Counters: 50
        File System Counters
                FILE: Number of bytes read=251507
                FILE: Number of bytes written=936834
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=18412801
                HDFS: Number of bytes written=86561
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Killed map tasks=1
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=19449
                Total time spent by all reduces in occupied slots (ms)=5582
                Total time spent by all map tasks (ms)=19449
                Total time spent by all reduce tasks (ms)=5582
                Total vcore-milliseconds taken by all map tasks=19449
                Total vcore-milliseconds taken by all reduce tasks=5582
                Total megabyte-milliseconds taken by all map tasks=19915776
                Total megabyte-milliseconds taken by all reduce tasks=5715968
```



```
                Killed map tasks=1
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=19449
                Total time spent by all reduces in occupied slots (ms)=5582
                Total time spent by all map tasks (ms)=19449
                Total time spent by all reduce tasks (ms)=5582
                Total vcore-milliseconds taken by all map tasks=19449
                Total vcore-milliseconds taken by all reduce tasks=5582
                Total megabyte-milliseconds taken by all map tasks=19915776
                Total megabyte-milliseconds taken by all reduce tasks=5715968
        Map-Reduce Framework
                Map input records=213149
                Map output records=9313
                Map output bytes=232875
                Map output materialized bytes=251513
                Input split bytes=578
                Combine input records=0
                Combine output records=0
                Reduce input groups=1677
                Reduce shuffle bytes=251513
                Reduce input records=9313
                Reduce output records=1677
                Spilled Records=18626
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=399
                CPU time spent (ms)=4360
                Physical memory (bytes) snapshot=563458048
                Virtual memory (bytes) snapshot=4519178240
                Total committed heap usage (bytes)=391579008
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=86561
loudera@quickstart Worldwide-trade-data]$ hadoop fs -ls /user/cloudera/indiantradedata/output2
und 2 items
w-r--r--   1 cloudera cloudera          0 2022-09-29 09:22 /user/cloudera/indiantradedata/output2/_SUCCESS
w-r--r--   1 cloudera cloudera      86561 2022-09-29 09:22 /user/cloudera/indiantradedata/output2/part-r-00000
loudera@quickstart Worldwide-trade-data]$ hadoop fs -ls /user/cloudera/indiantradedata/output2/
```

HOME ASSIGNMENT 2

Roll No: 208W1A1299

Name: MOHAMMAD RIZWANULLAH

PROJECT NAME:Twitter data analysis.

HadoopMapReduce-Twitter

Implementing MapReduce algorithms in Hadoop using the Twitter dataset(schema - https://github.com/episod/twitter-api-fields-as-crowdsourced/wiki)

Question answered:

1. What hour of the day does @PrezOno's tweet the most on average, using every day we have twitter data? Directory - https://github.uc.edu/loganasr/HadoopMapReduce-Twitter/tree/master/TweetsByHour
2. What day of the week does @PrezOno tweet the most on average? Use the same example as in #1 but for days of the week. Directory - https://github.uc.edu/loganasr/HadoopMapReduce-Twitter/tree/master/TweetsByDay
3. How does @PrezOno's tweet length compare to the average of all others? What is his average length? All others? Directory - https://github.uc.edu/loganasr/HadoopMapReduce-Twitter/tree/master/TweetLength

Instructions:

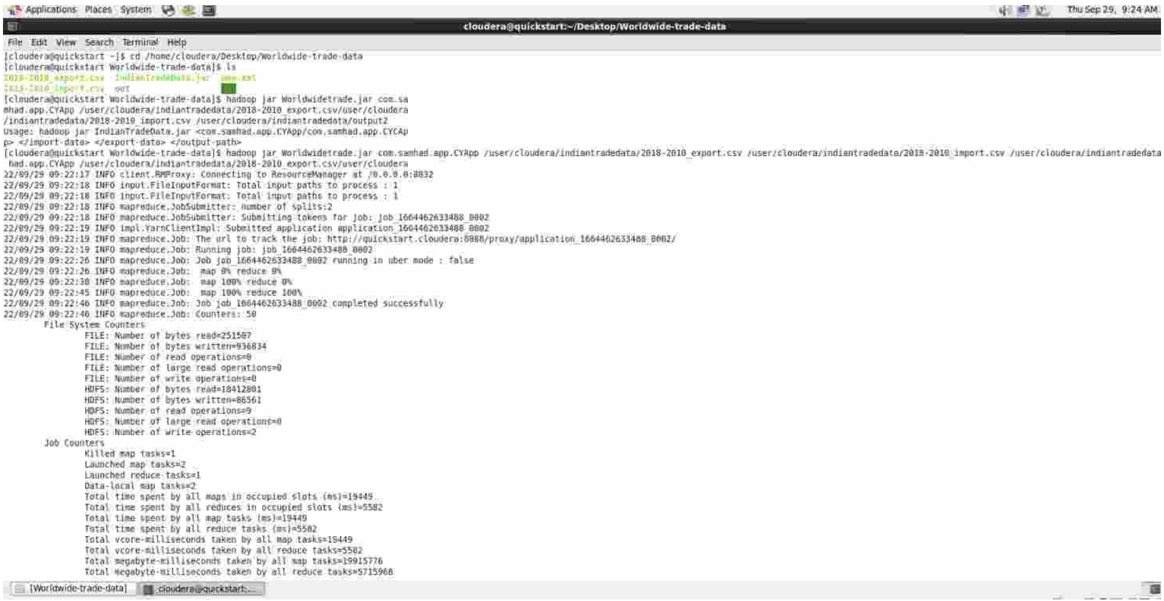
A sample data file has been included in /data directory to support quick validations through the Hadoop streaming mode. However, the file does not contain tweets from @PrezOno and hence, it would be necessary update the user_name for filtering the tweets.

Sample command: cat /data/sample-data | ./mapTweetsByHour.py | sort | ./reduceTweetsByHour.py

To run the map reduce programs in the hadoop cluster, utilize the following command.

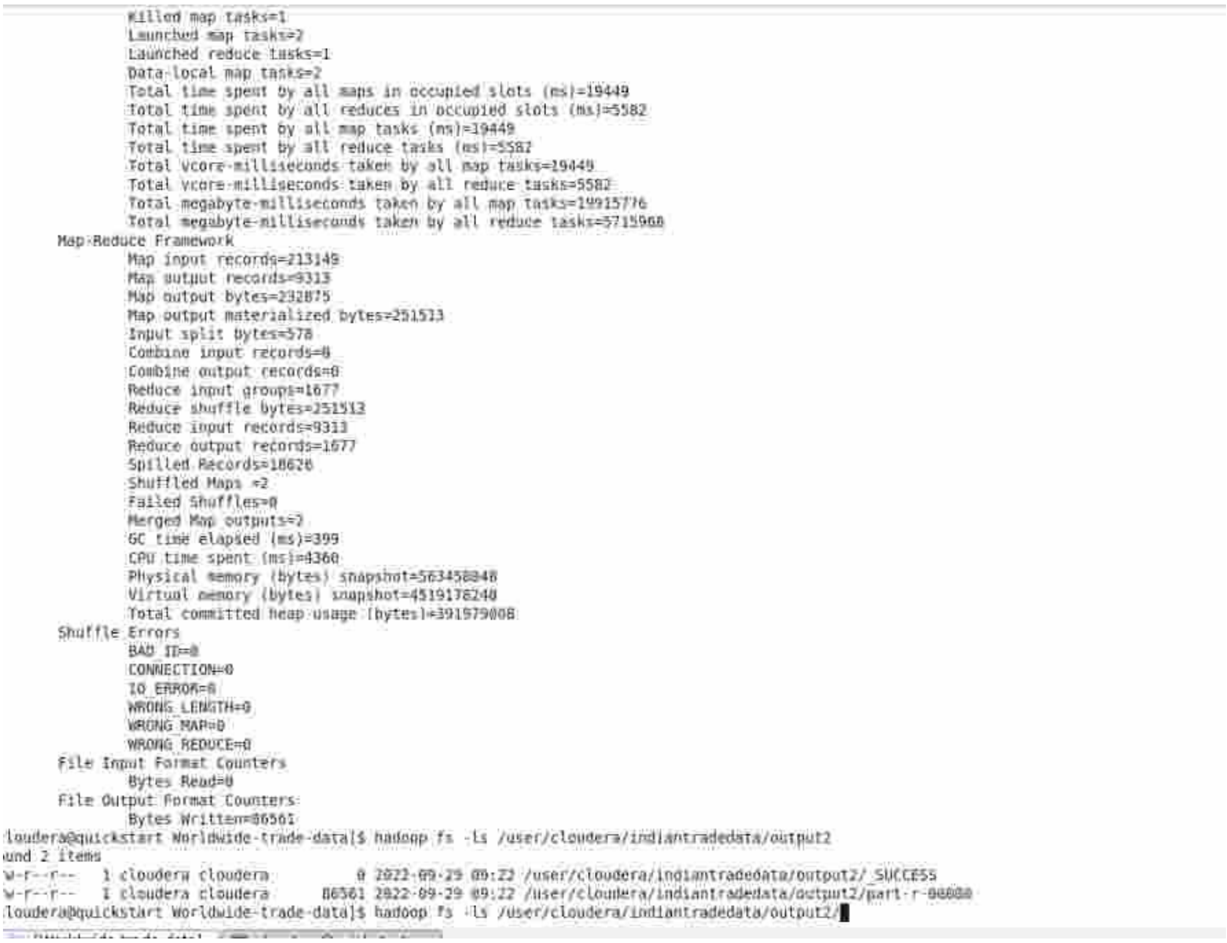
hadoop jar /root/hadoop-2.7.1/share/hadoop/tools/lib/hadoop-streaming-2.7.1.jar -input /data/twitter -output myoutput -file *.py -mapper mapTweetsByHour.py -reducer reduceTweetsByhour.py

Execution Screenshots:



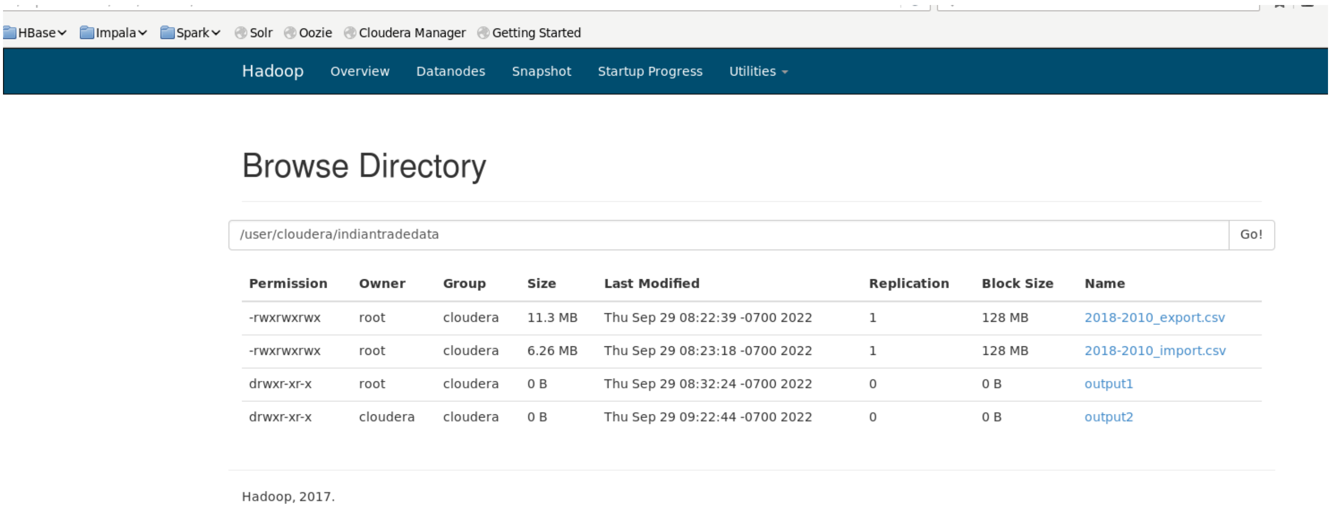
Applications Places System
Thu Sep 29, 9:24 AM
cloudera@quickstart:~/Desktop/Worldwide-trade-data
File Edit View Search Terminal Help
[Worldwide-trade-data]
cloudera@quickstart:...

```
[cloudera@quickstart ~]$ cd /home/cloudera/Desktop/Worldwide-trade-data
[cloudera@quickstart Worldwide-trade-data]$ ls
2018-2010_export.csv  IndianTradeData.jar  ime.txt
2018-2010_import.csv  out
[cloudera@quickstart Worldwide-trade-data]$ hadoop jar Worldwidetrade.jar com.sa
mhad.app.CYApp /user/cloudera/indiantradedata/2018-2010_export.csv/user/cloudera
/indiantradedata/2018-2010_import.csv /user/cloudera/indiantradedata/output2
Usage: hadoop jar IndianTradeData.jar <com.samhad.app.CYApp/com.samhad.app.CYCAp
p> </import-data> </export-data> </output-path>
[cloudera@quickstart Worldwide-trade-data]$ hadoop jar Worldwidetrade.jar com.samhad.app.CYApp /user/cloudera/indiantradedata/2018-2010_export.csv /user/cloudera/indiantradedata/2018-2010_import.csv /user/cloudera/indiantradedata
 had.app.CYApp /user/cloudera/indiantradedata/2018-2010_export.csv/user/cloudera
22/09/29 09:22:17 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
22/09/29 09:22:18 INFO input.FileInputFormat: Total input paths to process : 1
22/09/29 09:22:18 INFO input.FileInputFormat: Total input paths to process : 1
22/09/29 09:22:18 INFO mapreduce.JobSubmitter: number of splits:2
22/09/29 09:22:18 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1664462633488_0002
22/09/29 09:22:19 INFO impl.YarnClientImpl: Submitted application application_1664462633488_0002
22/09/29 09:22:19 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1664462633488_0002/
22/09/29 09:22:19 INFO mapreduce.Job: Running job: job_1664462633488_0002
22/09/29 09:22:26 INFO mapreduce.Job: Job job_1664462633488_0002 running in uber mode : false
22/09/29 09:22:26 INFO mapreduce.Job:  map 0% reduce 0%
22/09/29 09:22:38 INFO mapreduce.Job:  map 100% reduce 0%
22/09/29 09:22:45 INFO mapreduce.Job:  map 100% reduce 100%
22/09/29 09:22:46 INFO mapreduce.Job: Job job_1664462633488_0002 completed successfully
22/09/29 09:22:46 INFO mapreduce.Job: Counters: 50
        File System Counters
                FILE: Number of bytes read=251507
                FILE: Number of bytes written=936834
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=18412801
                HDFS: Number of bytes written=86561
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Killed map tasks=1
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=19449
                Total time spent by all reduces in occupied slots (ms)=5582
                Total time spent by all map tasks (ms)=19449
                Total time spent by all reduce tasks (ms)=5582
                Total vcore-milliseconds taken by all map tasks=19449
                Total vcore-milliseconds taken by all reduce tasks=5582
                Total megabyte-milliseconds taken by all map tasks=19915776
                Total megabyte-milliseconds taken by all reduce tasks=5715968
```



```
                Killed map tasks=1
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=19449
                Total time spent by all reduces in occupied slots (ms)=5582
                Total time spent by all map tasks (ms)=19449
                Total time spent by all reduce tasks (ms)=5582
                Total vcore-milliseconds taken by all map tasks=19449
                Total vcore-milliseconds taken by all reduce tasks=5582
                Total megabyte-milliseconds taken by all map tasks=19915776
                Total megabyte-milliseconds taken by all reduce tasks=5715968
        Map-Reduce Framework
                Map input records=213149
                Map output records=9313
                Map output bytes=232875
                Map output materialized bytes=251513
                Input split bytes=578
                Combine input records=0
                Combine output records=0
                Reduce input groups=1677
                Reduce shuffle bytes=251513
                Reduce input records=9313
                Reduce output records=1677
                Spilled Records=18626
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=399
                CPU time spent (ms)=4360
                Physical memory (bytes) snapshot=563458048
                Virtual memory (bytes) snapshot=4519178240
                Total committed heap usage (bytes)=391979008
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=86561
loudera@quickstart Worldwide-trade-data]$ hadoop fs -ls /user/cloudera/indiantradedata/output2
und 2 items
w-r--r--   1 cloudera cloudera          0 2022-09-29 09:22 /user/cloudera/indiantradedata/output2/_SUCCESS
w-r--r--   1 cloudera cloudera      86561 2022-09-29 09:22 /user/cloudera/indiantradedata/output2/part-r-00000
loudera@quickstart Worldwide-trade-data]$ hadoop fs -ls /user/cloudera/indiantradedata/output2/
```

Checking for output file.



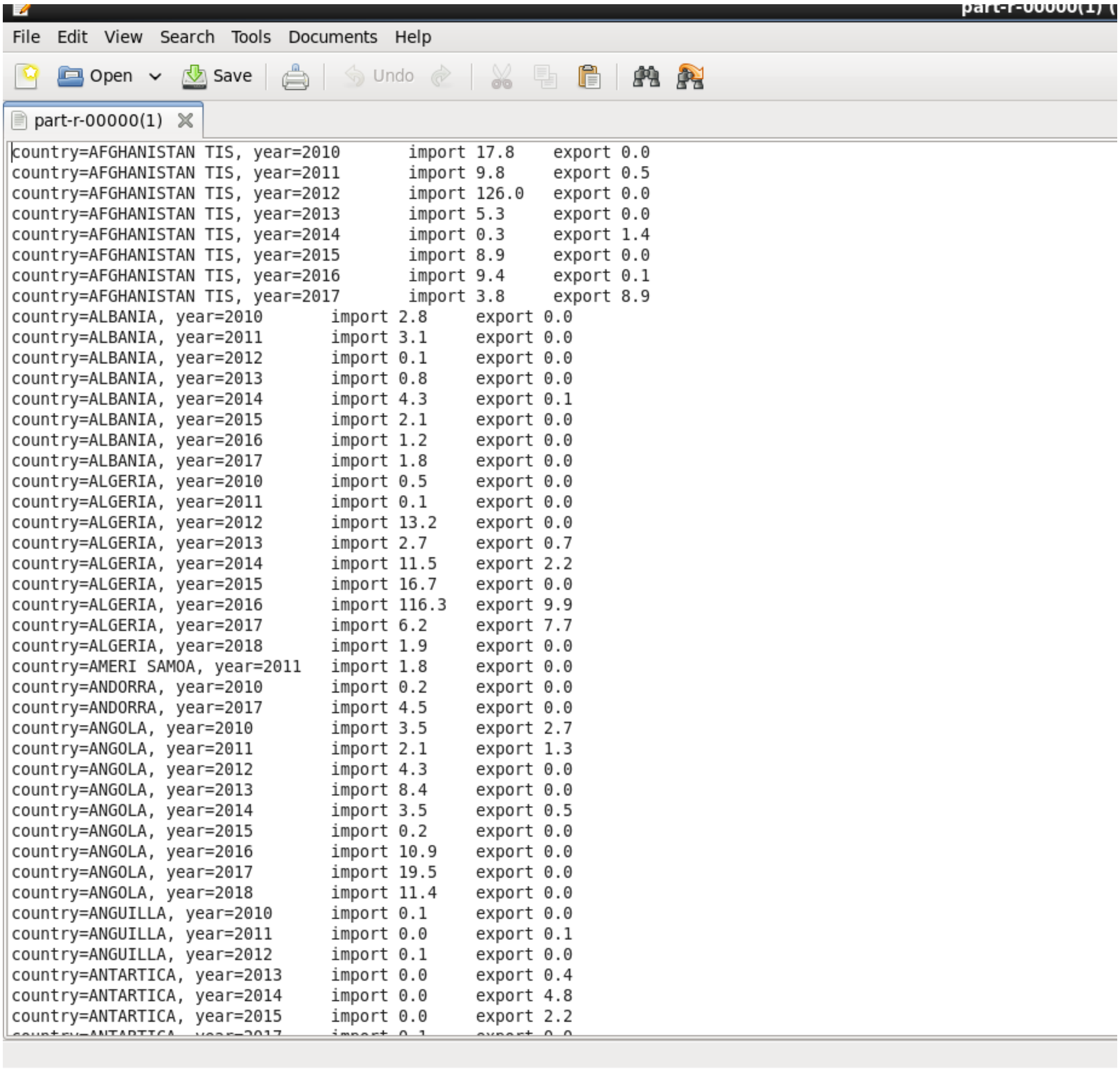
Browse Directory

/user/cloudera/indiantradedata

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rwxrwxrwx	root	cloudera	11.3 MB	Thu Sep 29 08:22:39 -0700 2022	1	128 MB	2018-2010_export.csv
-rwxrwxrwx	root	cloudera	6.26 MB	Thu Sep 29 08:23:18 -0700 2022	1	128 MB	2018-2010_import.csv
drwxr-xr-x	root	cloudera	0 B	Thu Sep 29 08:32:24 -0700 2022	0	0 B	output1
drwxr-xr-x	cloudera	cloudera	0 B	Thu Sep 29 09:22:44 -0700 2022	0	0 B	output2

Hadoop, 2017.

Output:



```
country=AFGHANISTAN TIS, year=2010	import 17.8	export 0.0
country=AFGHANISTAN TIS, year=2011	import 9.8	export 0.5
country=AFGHANISTAN TIS, year=2012	import 126.0	export 0.0
country=AFGHANISTAN TIS, year=2013	import 5.3	export 0.0
country=AFGHANISTAN TIS, year=2014	import 0.3	export 1.4
country=AFGHANISTAN TIS, year=2015	import 8.9	export 0.0
country=AFGHANISTAN TIS, year=2016	import 9.4	export 0.1
country=AFGHANISTAN TIS, year=2017	import 3.8	export 8.9
country=ALBANIA, year=2010	import 2.8	export 0.0
country=ALBANIA, year=2011	import 3.1	export 0.0
country=ALBANIA, year=2012	import 0.1	export 0.0
country=ALBANIA, year=2013	import 0.8	export 0.0
country=ALBANIA, year=2014	import 4.3	export 0.1
country=ALBANIA, year=2015	import 2.1	export 0.0
country=ALBANIA, year=2016	import 1.2	export 0.0
country=ALBANIA, year=2017	import 1.8	export 0.0
country=ALGERIA, year=2010	import 0.5	export 0.0
country=ALGERIA, year=2011	import 0.1	export 0.0
country=ALGERIA, year=2012	import 13.2	export 0.0
country=ALGERIA, year=2013	import 2.7	export 0.7
country=ALGERIA, year=2014	import 11.5	export 2.2
country=ALGERIA, year=2015	import 16.7	export 0.0
country=ALGERIA, year=2016	import 116.3	export 9.9
country=ALGERIA, year=2017	import 6.2	export 7.7
country=ALGERIA, year=2018	import 1.9	export 0.0
country=AMERI SAMOA, year=2011	import 1.8	export 0.0
country=ANDORRA, year=2010	import 0.2	export 0.0
country=ANDORRA, year=2017	import 4.5	export 0.0
country=ANGOLA, year=2010	import 3.5	export 2.7
country=ANGOLA, year=2011	import 2.1	export 1.3
country=ANGOLA, year=2012	import 4.3	export 0.0
country=ANGOLA, year=2013	import 8.4	export 0.0
country=ANGOLA, year=2014	import 3.5	export 0.5
country=ANGOLA, year=2015	import 0.2	export 0.0
country=ANGOLA, year=2016	import 10.9	export 0.0
country=ANGOLA, year=2017	import 19.5	export 0.0
country=ANGOLA, year=2018	import 11.4	export 0.0
country=ANGUILLA, year=2010	import 0.1	export 0.0
country=ANGUILLA, year=2011	import 0.0	export 0.1
country=ANGUILLA, year=2012	import 0.1	export 0.0
country=ANTARTICA, year=2013	import 0.0	export 0.4
country=ANTARTICA, year=2014	import 0.0	export 4.8
country=ANTARTICA, year=2015	import 0.0	export 2.2
country=ANTARTICA, year=2017	import 0.1	export 0.0
```

Output: We are able to get the most accurate times where celebrities tweet in which hour and in which day of the week

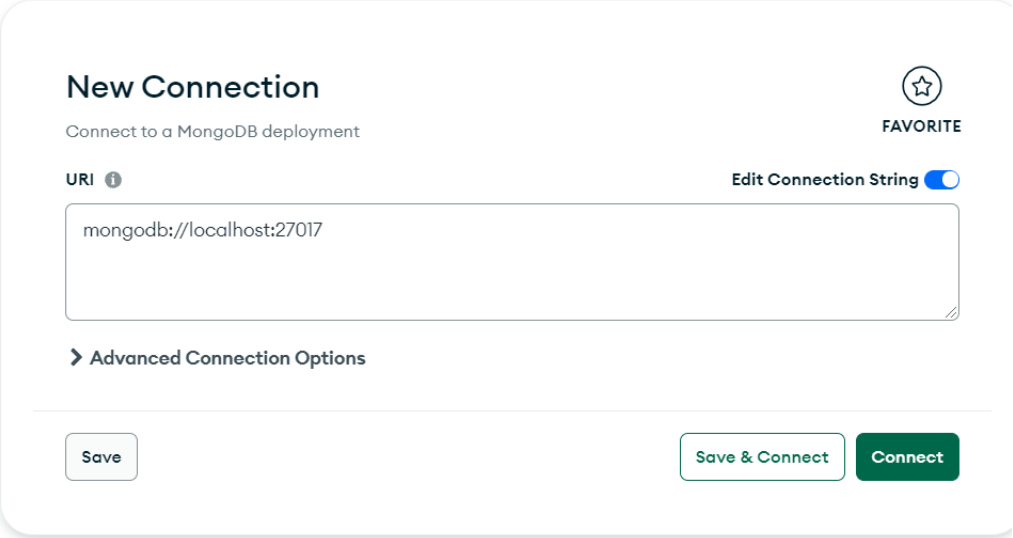
HOME ASSIGNMENT 3

Roll No: 208W1A1299

Name: MOHAMMAD RIZWANULLAH

Problem Statement : Implementation of mongodb using Twitter dataset.

Start new connection:



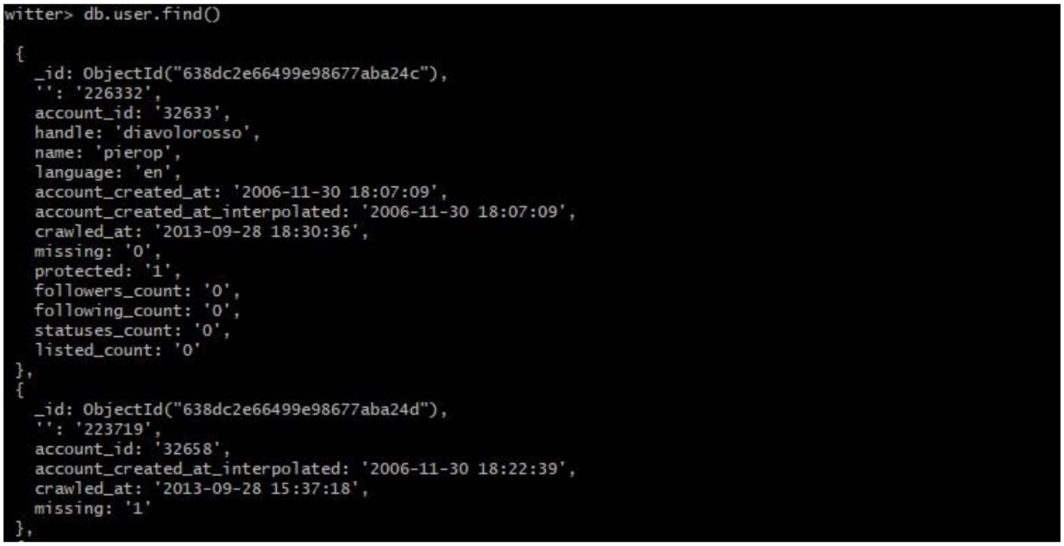
New Connection

Connect to a MongoDB deployment

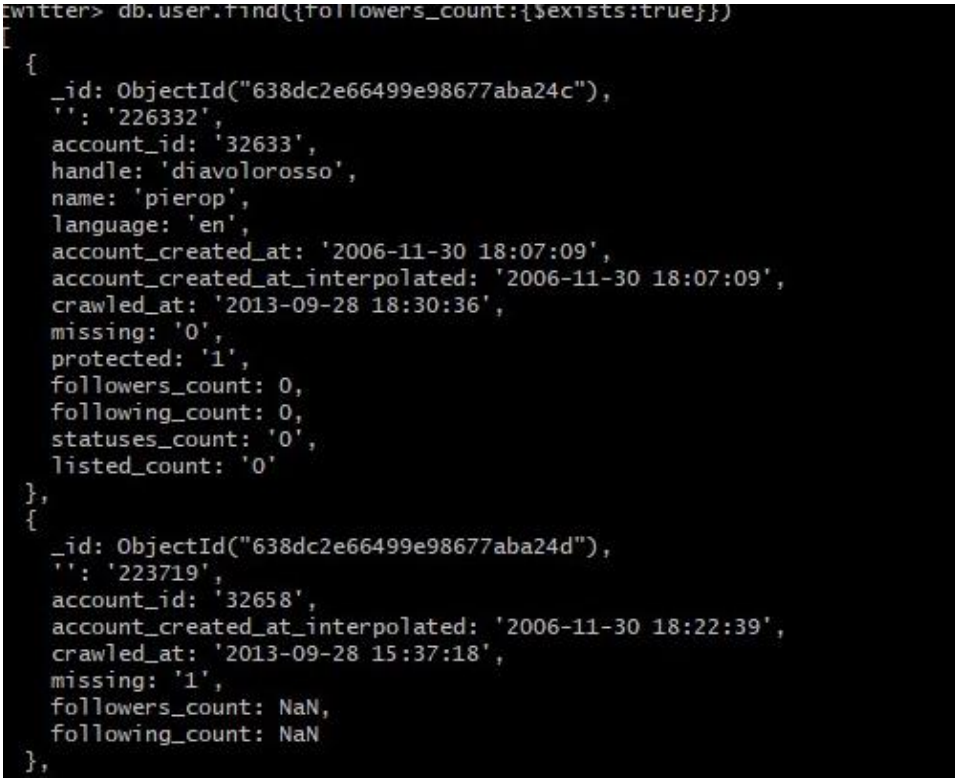
URI: mongodb://localhost:27017

Edit Connection String: on

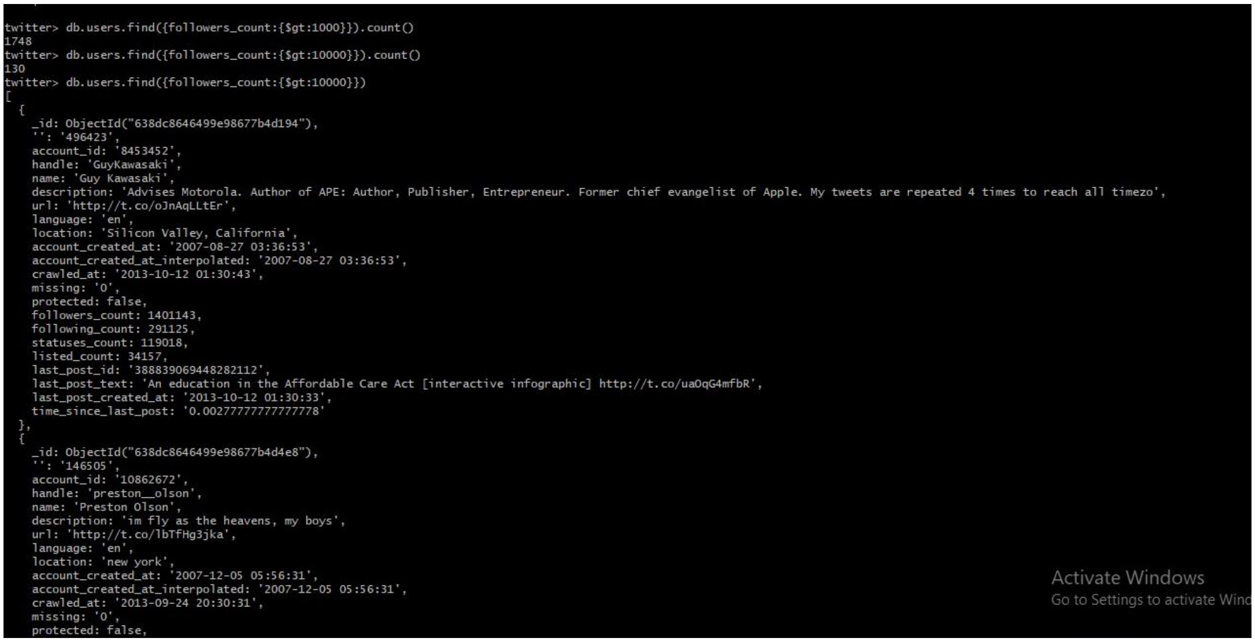
Import dataset into mongodb and convert into json format. And that can be used for analytics.



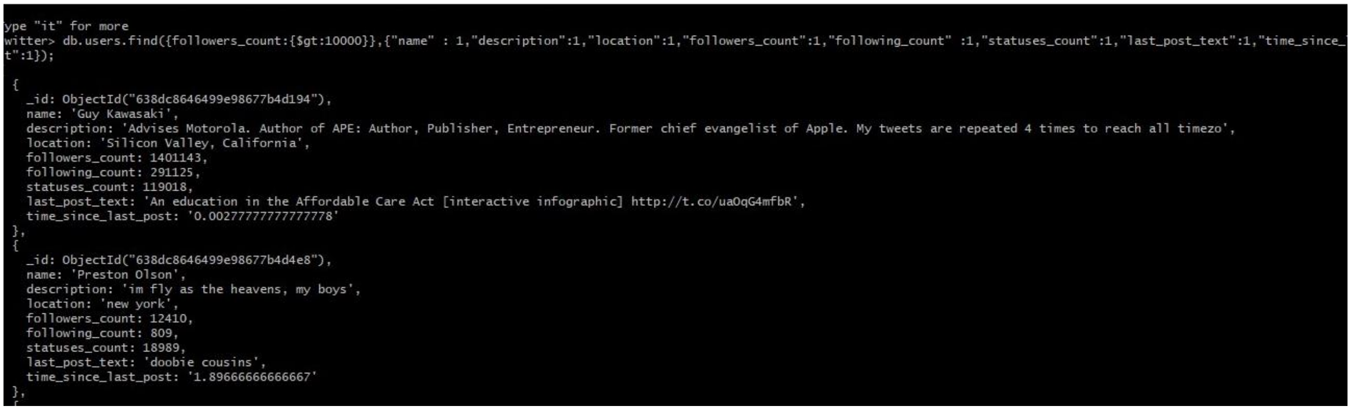
```
witter> db.user.find()

 {
   _id: ObjectId("638dc2e66499e98677aba24c"),
   '': '226332',
   account_id: '32633',
   handle: 'diavolorosso',
   name: 'pierop',
   language: 'en',
   account_created_at: '2006-11-30 18:07:09',
   account_created_at_interpolated: '2006-11-30 18:07:09',
   crawled_at: '2013-09-28 18:30:36',
   missing: '0',
   protected: '1',
   followers_count: '0',
   following_count: '0',
   statuses_count: '0',
   listed_count: '0'
 },
 {
   _id: ObjectId("638dc2e66499e98677aba24d"),
   '': '223719',
   account_id: '32658',
   account_created_at_interpolated: '2006-11-30 18:22:39',
   crawled_at: '2013-09-28 15:37:18',
   missing: '1'
 },
```



```
twitter> db.user.find({followers_count:{$exists:true}})
[
  {
    _id: ObjectId("638dc2e66499e98677aba24c"),
    '': '226332',
    account_id: '32633',
    handle: 'diavolorosso',
    name: 'pierop',
    language: 'en',
    account_created_at: '2006-11-30 18:07:09',
    account_created_at_interpolated: '2006-11-30 18:07:09',
    crawled_at: '2013-09-28 18:30:36',
    missing: '0',
    protected: '1',
    followers_count: 0,
    following_count: 0,
    statuses_count: '0',
    listed_count: '0'
  },
  {
    _id: ObjectId("638dc2e66499e98677aba24d"),
    '': '223719',
    account_id: '32658',
    account_created_at_interpolated: '2006-11-30 18:22:39',
    crawled_at: '2013-09-28 15:37:18',
    missing: '1',
    followers_count: NaN,
    following_count: NaN
  },
```



```
twitter> db.users.find({followers_count:{$gt:1000}}).count()
1748
twitter> db.users.find({followers_count:{$gt:10000}}).count()
130
twitter> db.users.find({followers_count:{$gt:10000}})
[
  {
    _id: ObjectId("638dc8646499e98677b4d194"),
    '': '496423',
    account_id: '8453452',
    handle: 'GuyKawasaki',
    name: 'Guy Kawasaki',
    description: 'Advises Motorola. Author of APE: Author, Publisher, Entrepreneur. Former chief evangelist of Apple. My tweets are repeated 4 times to reach all timezo',
    url: 'http://t.co/oJnAqLLtEr',
    language: 'en',
    location: 'Silicon Valley, California',
    account_created_at: '2007-08-27 03:36:53',
    account_created_at_interpolated: '2007-08-27 03:36:53',
    crawled_at: '2013-10-12 01:30:43',
    missing: '0',
    protected: false,
    followers_count: 1401143,
    following_count: 291125,
    statuses_count: 119018,
    listed_count: 34157,
    last_post_id: '388839069448282112',
    last_post_text: 'An education in the Affordable Care Act [interactive infographic] http://t.co/uaOqG4mfbR',
    last_post_created_at: '2013-10-12 01:30:33',
    time_since_last_post: '0.002777777777777778'
  },
  {
    _id: ObjectId("638dc8646499e98677b4d4e8"),
    '': '146505',
    account_id: '10862672',
    handle: 'preston__olson',
    name: 'Preston Olson',
    description: 'im fly as the heavens, my boys',
    url: 'http://t.co/lbTfHg3jka',
    language: 'en',
    location: 'new york',
    account_created_at: '2007-12-05 05:56:31',
    account_created_at_interpolated: '2007-12-05 05:56:31',
    crawled_at: '2013-09-24 20:30:31',
    missing: '0',
    protected: false,
```



```
ype "it" for more
witter> db.users.find({followers_count:{$gt:10000}},{"name" : 1,"description":1,"location":1,"followers_count":1,"following_count" :1,"statuses_count":1,"last_post_text":1,"time_since_
t":1});

 {
   _id: ObjectId("638dc8646499e98677b4d194"),
   name: 'Guy Kawasaki',
   description: 'Advises Motorola. Author of APE: Author, Publisher, Entrepreneur. Former chief evangelist of Apple. My tweets are repeated 4 times to reach all timezo',
   location: 'Silicon Valley, California',
   followers_count: 1401143,
   following_count: 291125,
   statuses_count: 119018,
   last_post_text: 'An education in the Affordable Care Act [interactive infographic] http://t.co/uaOqG4mfbR',
   time_since_last_post: '0.002777777777777778'
 },
 {
   _id: ObjectId("638dc8646499e98677b4d4e8"),
   name: 'Preston Olson',
   description: 'im fly as the heavens, my boys',
   location: 'new york',
   followers_count: 12410,
   following_count: 809,
   statuses_count: 18989,
   last_post_text: 'doobie cousins',
   time_since_last_post: '1.8966666666666667'
 },
```

HOME ASSIGNMENT 4

Roll No: 208W1A1299

Name: MOHAMMAD RIZWANULLAH

Aim: To do analysis using Apache spark in data bricks.

Execution:

1)Open databricks and create a new cluster with latest configuration version.

2) Now create a Notebook to work with spark.

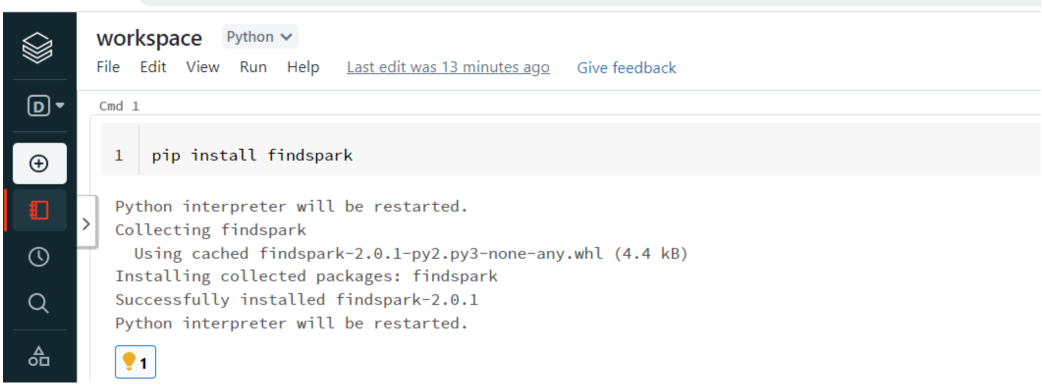
3) Import csv file or any file for using it and do analysis using pyspark.

4) Now write code to import dataset and create some RDD to read input in spark.

5) Run the cell.

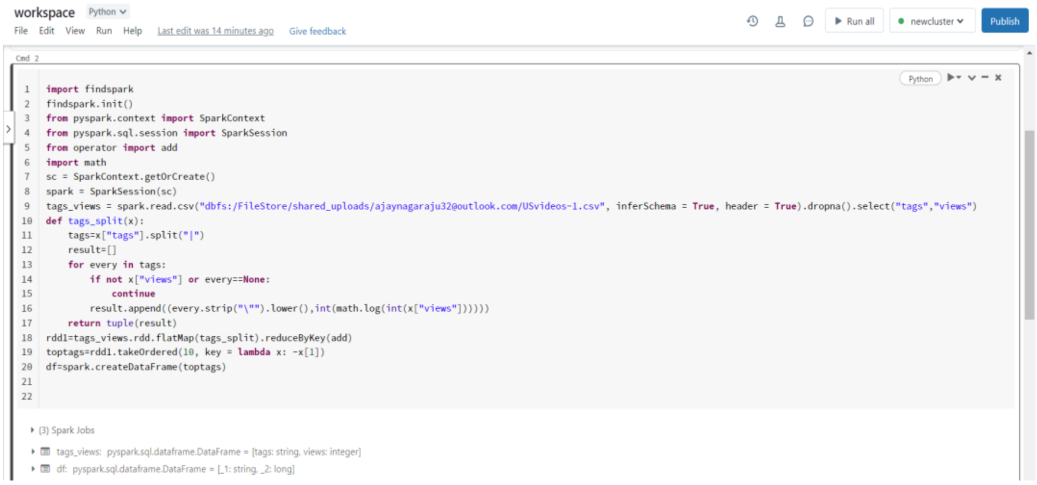
Below are the screenshots of execution:

Twitter followers count and protected data analysis.



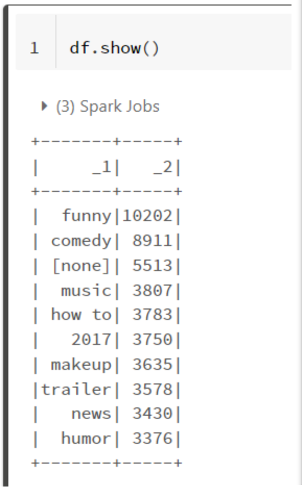
```
pip install findspark
```

```
Python interpreter will be restarted.
Collecting findspark
  Using cached findspark-2.0.1-py2.py3-none-any.whl (4.4 kB)
Installing collected packages: findspark
Successfully installed findspark-2.0.1
Python interpreter will be restarted.
```



```
import findspark
findspark.init()
from pyspark.context import SparkContext
from pyspark.sql.session import SparkSession
from operator import add
import math
sc = SparkContext.getOrCreate()
spark = SparkSession(sc)
tags_views = spark.read.csv("dbfs:/FileStore/shared_uploads/ajaynagaraju32@outlook.com/USvideos-1.csv", inferSchema = True, header = True).dropna().select("tags","views")
def tags_split(x):
    tags=x["tags"].split("|")
    result=[]
    for every in tags:
        if not x["views"] or every==None:
            continue
        result.append((every.strip("\"").lower(),int(math.log(int(x["views"])))))
    return tuple(result)
rdd1=tags_views.rdd.flatMap(tags_split).reduceByKey(add)
toptags=rdd1.takeOrdered(10, key = lambda x: -x[1])
df=spark.createDataFrame(toptags)
```

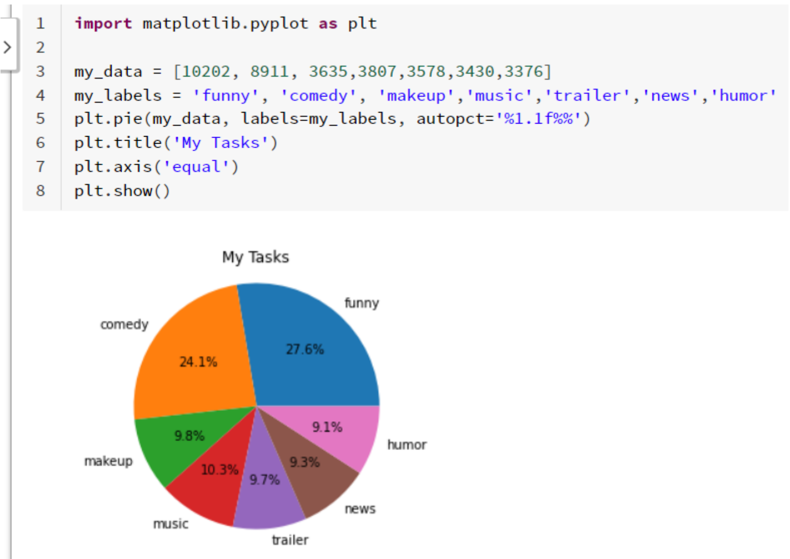
```
▸ (3) Spark Jobs
▸ tags_views:  pyspark.sql.dataframe.DataFrame = [tags: string, views: integer]
▸ df:  pyspark.sql.dataframe.DataFrame = [_1: string, _2: long]
```



```
df.show()
```

(3) Spark Jobs

```
+-------+-----+
|     _1|   _2|
+-------+-----+
|  funny|10202|
| comedy| 8911|
| [none]| 5513|
|  music| 3807|
| how to| 3783|
|   2017| 3750|
| makeup| 3635|
|trailer| 3578|
|   news| 3430|
|  humor| 3376|
+-------+-----+
```



```
import matplotlib.pyplot as plt

my_data = [10202, 8911, 3635,3807,3578,3430,3376]
my_labels = 'funny', 'comedy', 'makeup','music','trailer','news','humor'
plt.pie(my_data, labels=my_labels, autopct='%1.1f%%')
plt.title('My Tasks')
plt.axis('equal')
plt.show()
```