Big Data Project
Covid-19 Tweets Sentimental Analysis

Antarjita Mandal
Section:D
SRN:PES2201800409

Nayana CS Reddy
Section:E
SRN:PES2201800510

Sana Rahman
Section:D
SRN:PES2201800665

INTRODUCTION

In the view of the pandemic many have tweeted regarding the Covid 19 pandemic. It becomes essential to analyse the impact of the onset of the covid-19 pandemic. One best way is to assign a polarity to the tweet and determine which tweet is positive, negative and neutral.

ABOUT OUR DATASET

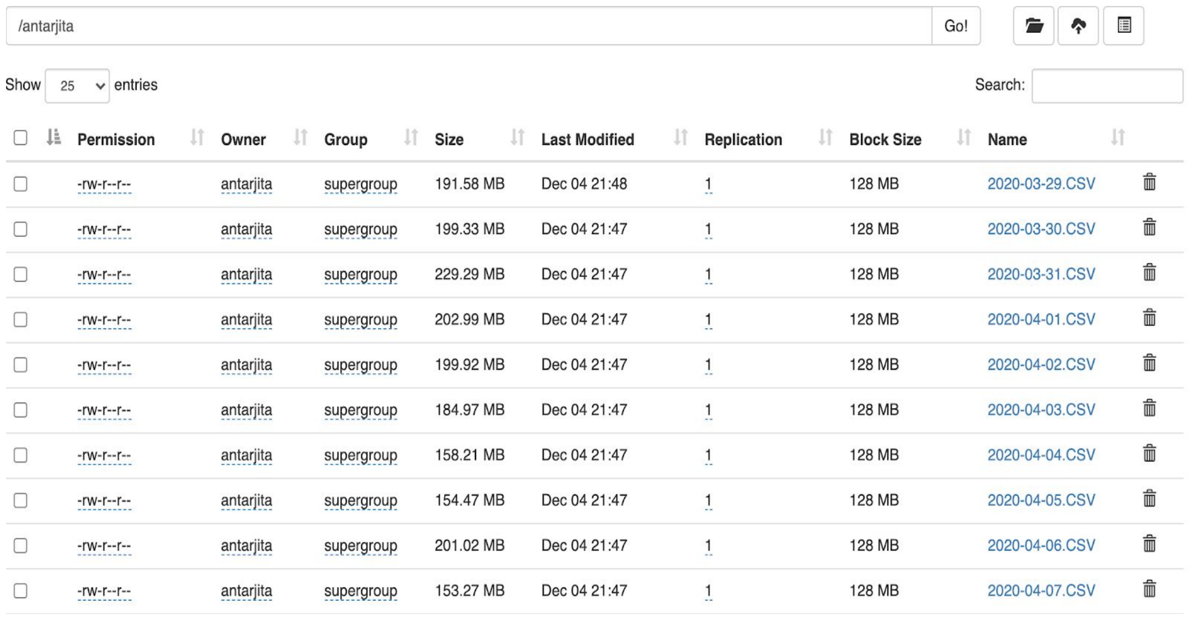
Our data set is Coronavirus (covid19) Tweets - early April from kaggle. It consists of 18 files each containing 22 columns. It contains various columns like friends_count, followers_count, country_code etc and we can check the validity of the account through columns like verified etc.

TOOLS USED:-

We have used Hdfs as the underlying storage. Hive is used for basic query retrieval and Pyspark is used to analyse the data. The below procedure is carried out in a pseudo distributed mode locally on our system (Mac/Ubuntu).

APPROACH :-

All the files are first stored in hdfs which is connected to our hive metastore and pyspark through yarn.

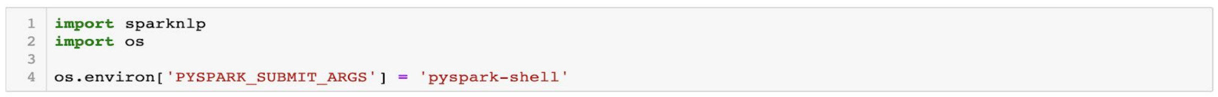


/antarjita Go!

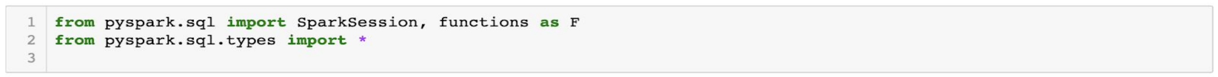
Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	antarjita	supergroup	191.58 MB	Dec 04 21:48	1	128 MB	2020-03-29.CSV
-rw-r--r--	antarjita	supergroup	199.33 MB	Dec 04 21:47	1	128 MB	2020-03-30.CSV
-rw-r--r--	antarjita	supergroup	229.29 MB	Dec 04 21:47	1	128 MB	2020-03-31.CSV
-rw-r--r--	antarjita	supergroup	202.99 MB	Dec 04 21:47	1	128 MB	2020-04-01.CSV
-rw-r--r--	antarjita	supergroup	199.92 MB	Dec 04 21:47	1	128 MB	2020-04-02.CSV
-rw-r--r--	antarjita	supergroup	184.97 MB	Dec 04 21:47	1	128 MB	2020-04-03.CSV
-rw-r--r--	antarjita	supergroup	158.21 MB	Dec 04 21:47	1	128 MB	2020-04-04.CSV
-rw-r--r--	antarjita	supergroup	154.47 MB	Dec 04 21:47	1	128 MB	2020-04-05.CSV
-rw-r--r--	antarjita	supergroup	201.02 MB	Dec 04 21:47	1	128 MB	2020-04-06.CSV
-rw-r--r--	antarjita	supergroup	153.27 MB	Dec 04 21:47	1	128 MB	2020-04-07.CSV

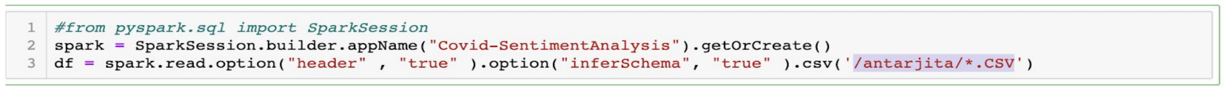
1. First we cleaned our data in pyspark and retained the columns necessary for further analysis.
   - Initial Set-up in pyspark shell:-



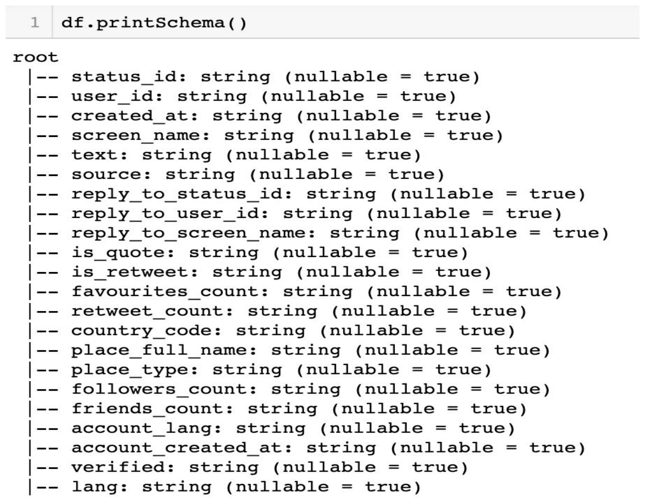
```
import sparknlp
import os

os.environ['PYSPARK_SUBMIT_ARGS'] = 'pyspark-shell'
```



```
from pyspark.sql import SparkSession, functions as F
from pyspark.sql.types import *

```

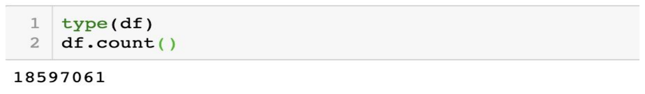


```
#from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("Covid-SentimentAnalysis").getOrCreate()
df = spark.read.option("header" , "true" ).option("inferSchema", "true" ).csv('/antarjita/*.CSV')
```



```
df.printSchema()
```

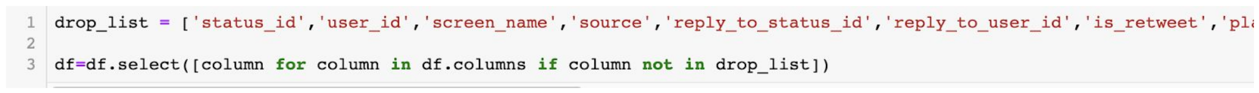
```
root
 |-- status_id: string (nullable = true)
 |-- user_id: string (nullable = true)
 |-- created_at: string (nullable = true)
 |-- screen_name: string (nullable = true)
 |-- text: string (nullable = true)
 |-- source: string (nullable = true)
 |-- reply_to_status_id: string (nullable = true)
 |-- reply_to_user_id: string (nullable = true)
 |-- reply_to_screen_name: string (nullable = true)
 |-- is_quote: string (nullable = true)
 |-- is_retweet: string (nullable = true)
 |-- favourites_count: string (nullable = true)
 |-- retweet_count: string (nullable = true)
 |-- country_code: string (nullable = true)
 |-- place_full_name: string (nullable = true)
 |-- place_type: string (nullable = true)
 |-- followers_count: string (nullable = true)
 |-- friends_count: string (nullable = true)
 |-- account_lang: string (nullable = true)
 |-- account_created_at: string (nullable = true)
 |-- verified: string (nullable = true)
 |-- lang: string (nullable = true)
```



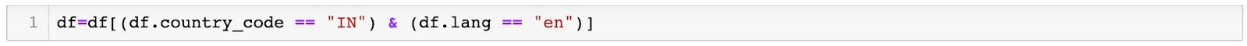
```
type(df)
df.count()
```

```
18597061
```

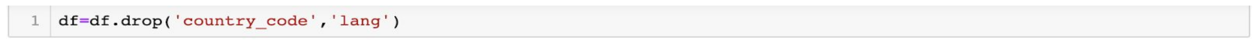
   - Dropping unnecessary columns :-



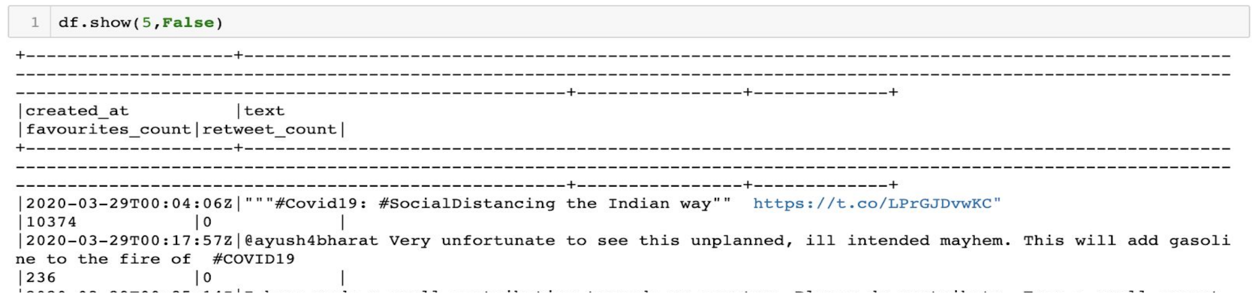
```
drop_list = ['status_id','user_id','screen_name','source','reply_to_status_id','reply_to_user_id','is_retweet','pl

df=df.select([column for column in df.columns if column not in drop_list])
```



```
df=df[(df.country_code == "IN") & (df.lang == "en")]
```



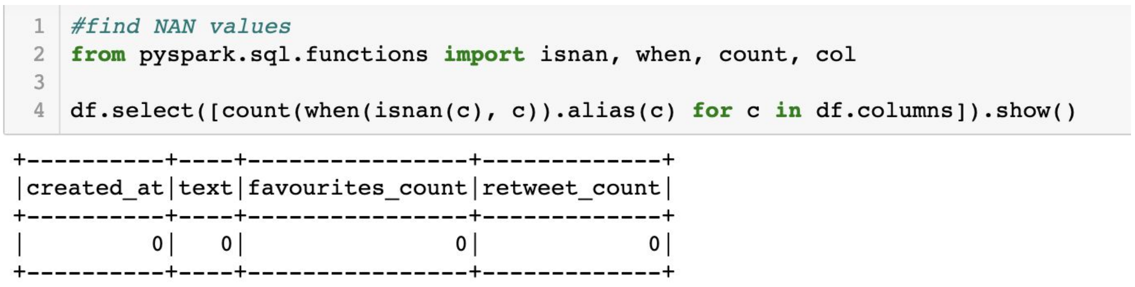
```
df=df.drop('country_code','lang')
```



```
df.show(5,False)
```

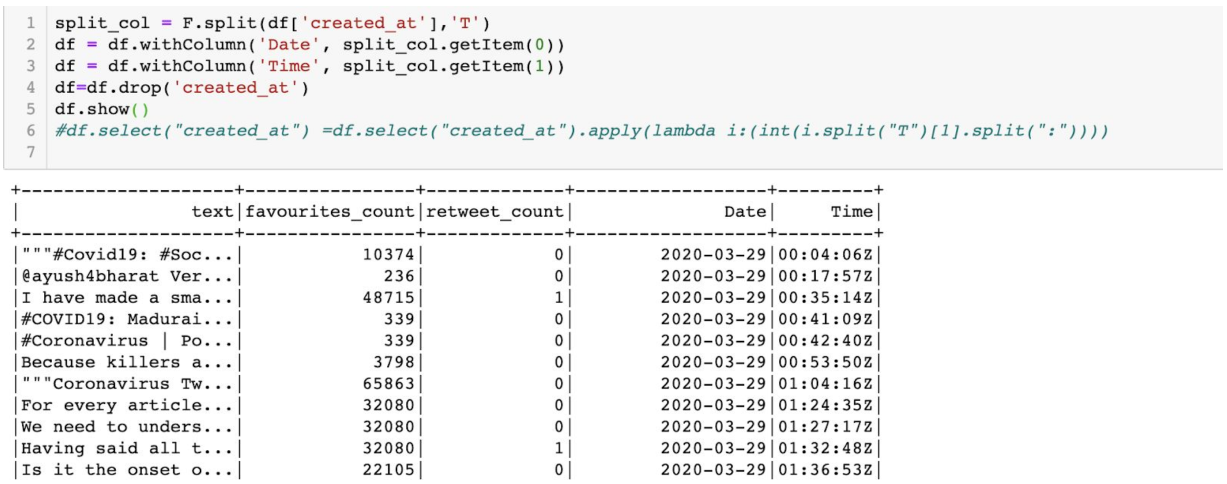
```
+--------------------+------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------+----------------+-------------+
|created_at          |text
|favourites_count|retweet_count|
+--------------------+------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------+----------------+-------------+
|2020-03-29T00:04:06Z|"""#Covid19: #SocialDistancing the Indian way""  https://t.co/LPrGJDvwKC"
|10374           |0            |
|2020-03-29T00:17:57Z|@ayush4bharat Very unfortunate to see this unplanned, ill intended mayhem. This will add gasoli
ne to the fire of  #COVID19
|236             |0            |
```

- Removing Nan values and splitting created_at into valid time:-



```
#find NAN values
from pyspark.sql.functions import isnan, when, count, col

df.select([count(when(isnan(c), c)).alias(c) for c in df.columns]).show()
```

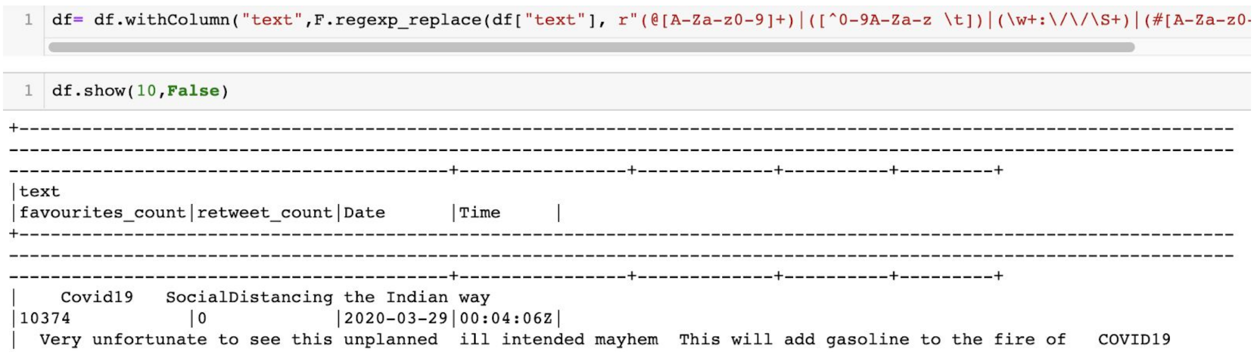
```
+----------+----+----------------+-------------+
|created_at|text|favourites_count|retweet_count|
+----------+----+----------------+-------------+
|         0|   0|               0|            0|
+----------+----+----------------+-------------+
```



```
split_col = F.split(df['created_at'],'T')
df = df.withColumn('Date', split_col.getItem(0))
df = df.withColumn('Time', split_col.getItem(1))
df=df.drop('created_at')
df.show()
#df.select("created_at") =df.select("created_at").apply(lambda i:(int(i.split("T")[1].split(":"))))

```

```
+--------------------+----------------+-------------+------------------+---------+
|                text|favourites_count|retweet_count|              Date|     Time|
+--------------------+----------------+-------------+------------------+---------+
|"""#Covid19: #Soc...|           10374|            0|        2020-03-29|00:04:06Z|
|@ayush4bharat Ver...|             236|            0|        2020-03-29|00:17:57Z|
|I have made a sma...|           48715|            1|        2020-03-29|00:35:14Z|
|#COVID19: Madurai...|             339|            0|        2020-03-29|00:41:09Z|
|#Coronavirus | Po...|             339|            0|        2020-03-29|00:42:40Z|
|Because killers a...|            3798|            0|        2020-03-29|00:53:50Z|
|"""Coronavirus Tw...|           65863|            0|        2020-03-29|01:04:16Z|
|For every article...|           32080|            0|        2020-03-29|01:24:35Z|
|We need to unders...|           32080|            0|        2020-03-29|01:27:17Z|
|Having said all t...|           32080|            1|        2020-03-29|01:32:48Z|
|Is it the onset o...|           22105|            0|        2020-03-29|01:36:53Z|
```

- Removing special characters from the tweets:-



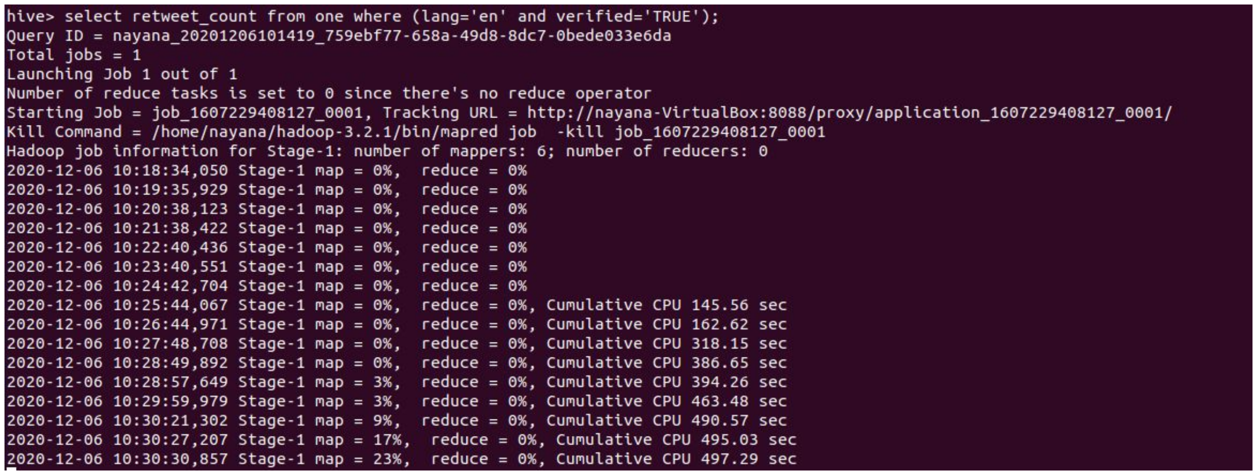
```
df= df.withColumn("text",F.regexp_replace(df["text"], r"(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)|(#[A-Za-z0-
```

```
df.show(10,False)
```

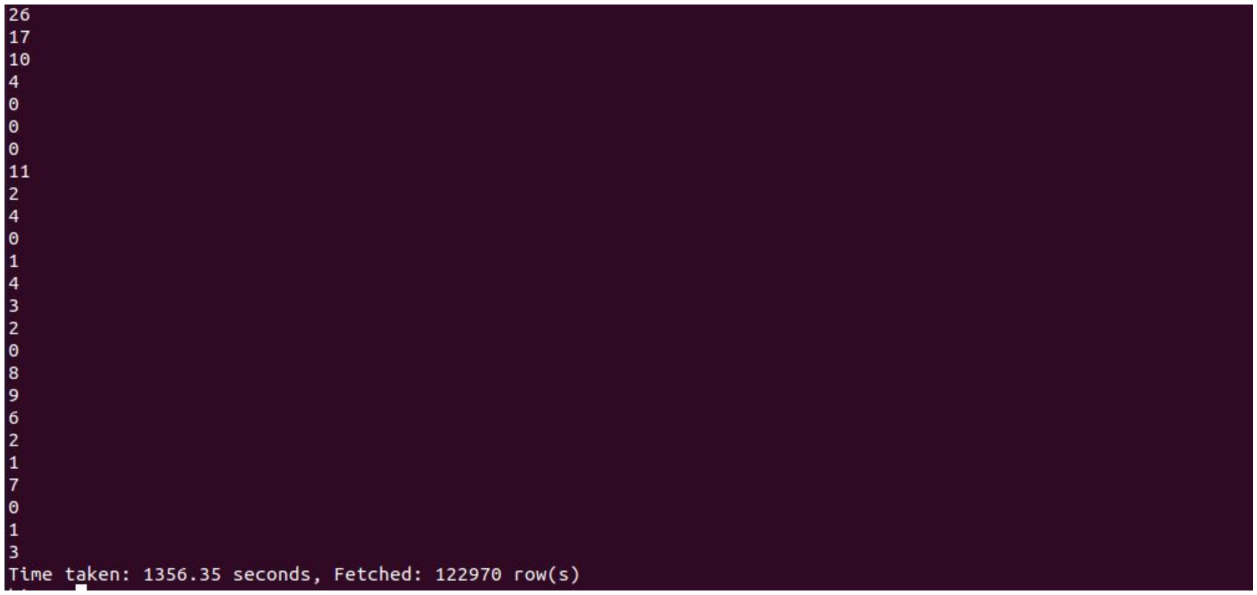
```
+------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------
------------------------------------------+----------------+-------------+----------+---------+
|text
|favourites_count|retweet_count|Date      |Time     |
+------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------
------------------------------------------+----------------+-------------+----------+---------+
|    Covid19   SocialDistancing the Indian way
|10374           |0            |2020-03-29|00:04:06Z|
|  Very unfortunate to see this unplanned  ill intended mayhem  This will add gasoline to the fire of   COVID19
```

2. Initial insights from the data is obtained through basic querying in hive and pyspark.

select retweer_count from one where (lang='en' and verified='TRUE');
This query shows the number of retweets that all the verified accounts that tweeted in english got.
This query was used to check how popular the tweets about coronavirus by famous people were.



```
hive> select retweet_count from one where (lang='en' and verified='TRUE');
Query ID = nayana_20201206101419_759ebf77-658a-49d8-8dc7-0bede033e6da
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1607229408127_0001, Tracking URL = http://nayana-VirtualBox:8088/proxy/application_1607229408127_0001/
Kill Command = /home/nayana/hadoop-3.2.1/bin/mapred job  -kill job_1607229408127_0001
Hadoop job information for Stage-1: number of mappers: 6; number of reducers: 0
2020-12-06 10:18:34,050 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:19:35,929 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:20:38,123 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:21:38,422 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:22:40,436 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:23:40,551 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:24:42,704 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:25:44,067 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 145.56 sec
2020-12-06 10:26:44,971 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 162.62 sec
2020-12-06 10:27:48,708 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 318.15 sec
2020-12-06 10:28:49,892 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 386.65 sec
2020-12-06 10:28:57,649 Stage-1 map = 3%,  reduce = 0%, Cumulative CPU 394.26 sec
2020-12-06 10:29:59,979 Stage-1 map = 3%,  reduce = 0%, Cumulative CPU 463.48 sec
2020-12-06 10:30:21,302 Stage-1 map = 9%,  reduce = 0%, Cumulative CPU 490.57 sec
2020-12-06 10:30:27,207 Stage-1 map = 17%,  reduce = 0%, Cumulative CPU 495.03 sec
2020-12-06 10:30:30,857 Stage-1 map = 23%,  reduce = 0%, Cumulative CPU 497.29 sec
```

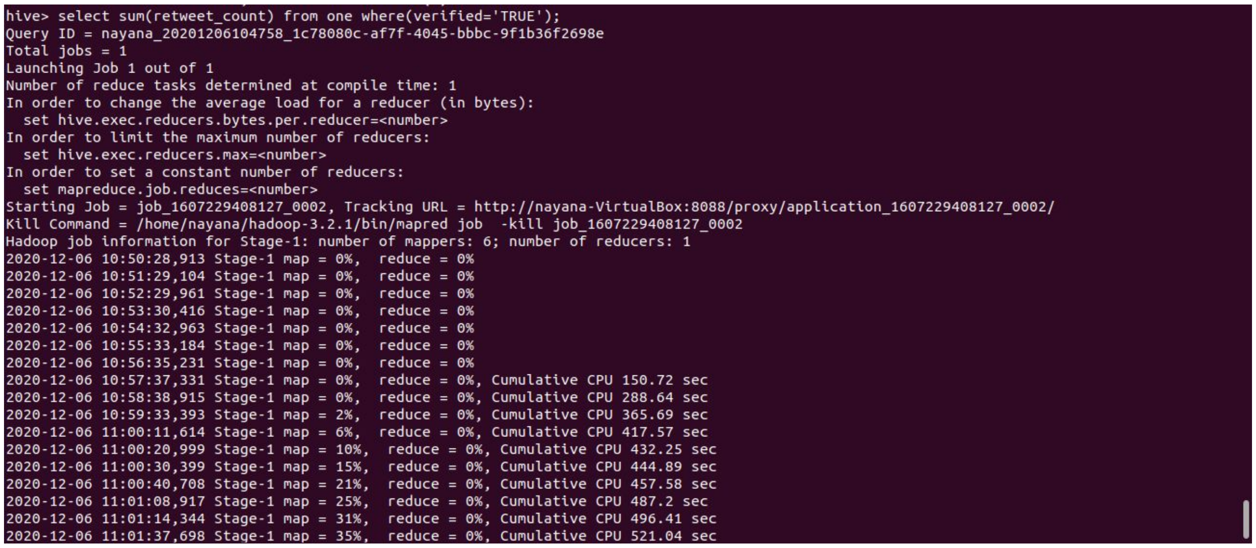


```
26
17
10
4
0
0
0
11
2
4
0
1
4
3
2
0
8
9
6
2
1
7
0
1
3
Time taken: 1356.35 seconds, Fetched: 122970 row(s)
```

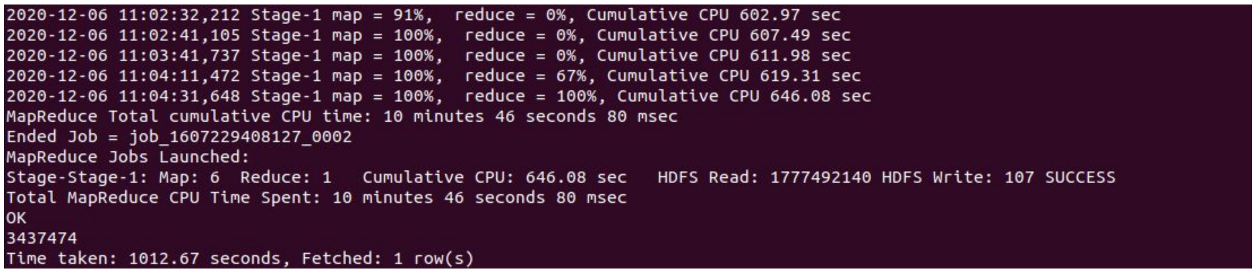
select sum(retweer_count) from one where(verified='TRUE')

This query finds the sum of the number of retweets of all tweets about coronavirus
By verified accounts.

This shows the importance of the topic and the engagement of the verified accounts.



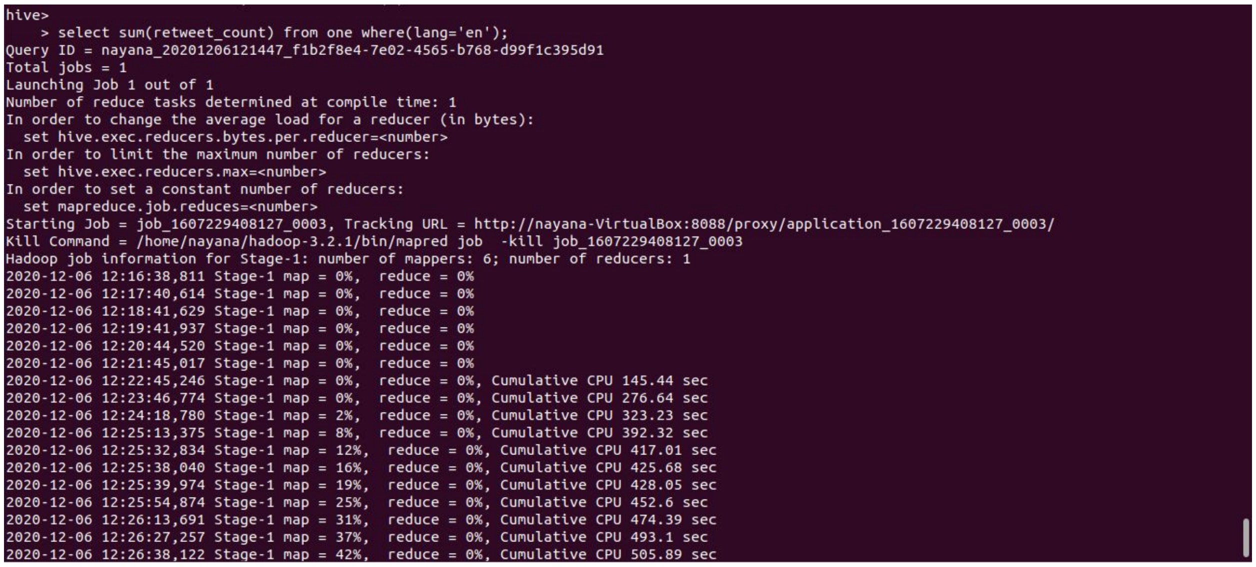
```
hive> select sum(retweet_count) from one where(verified='TRUE');
Query ID = nayana_20201206104758_1c78080c-af7f-4045-bbbc-9f1b36f2698e
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<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>
Starting Job = job_1607229408127_0002, Tracking URL = http://nayana-VirtualBox:8088/proxy/application_1607229408127_0002/
Kill Command = /home/nayana/hadoop-3.2.1/bin/mapred job  -kill job_1607229408127_0002
Hadoop job information for Stage-1: number of mappers: 6; number of reducers: 1
2020-12-06 10:50:28,913 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:51:29,104 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:52:29,961 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:53:30,416 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:54:32,963 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:55:33,184 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:56:35,231 Stage-1 map = 0%,  reduce = 0%
2020-12-06 10:57:37,331 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 150.72 sec
2020-12-06 10:58:38,915 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 288.64 sec
2020-12-06 10:59:33,393 Stage-1 map = 2%,  reduce = 0%, Cumulative CPU 365.69 sec
2020-12-06 11:00:11,614 Stage-1 map = 6%,  reduce = 0%, Cumulative CPU 417.57 sec
2020-12-06 11:00:20,999 Stage-1 map = 10%,  reduce = 0%, Cumulative CPU 432.25 sec
2020-12-06 11:00:30,399 Stage-1 map = 15%,  reduce = 0%, Cumulative CPU 444.89 sec
2020-12-06 11:00:40,708 Stage-1 map = 21%,  reduce = 0%, Cumulative CPU 457.58 sec
2020-12-06 11:01:08,917 Stage-1 map = 25%,  reduce = 0%, Cumulative CPU 487.2 sec
2020-12-06 11:01:14,344 Stage-1 map = 31%,  reduce = 0%, Cumulative CPU 496.41 sec
2020-12-06 11:01:37,698 Stage-1 map = 35%,  reduce = 0%, Cumulative CPU 521.04 sec
```



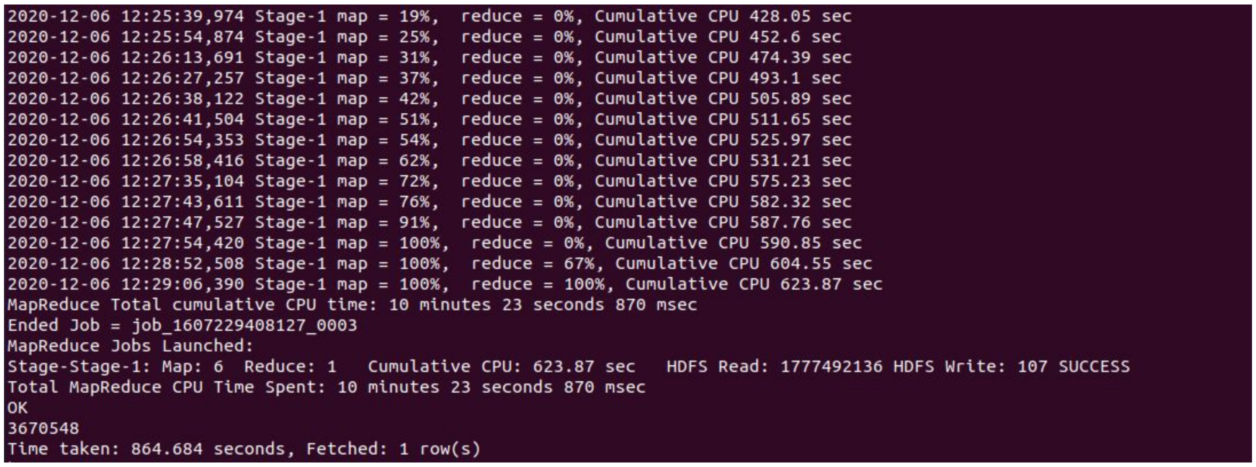
```
2020-12-06 11:02:32,212 Stage-1 map = 91%,  reduce = 0%, Cumulative CPU 602.97 sec
2020-12-06 11:02:41,105 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 607.49 sec
2020-12-06 11:03:41,737 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 611.98 sec
2020-12-06 11:04:11,472 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 619.31 sec
2020-12-06 11:04:31,648 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 646.08 sec
MapReduce Total cumulative CPU time: 10 minutes 46 seconds 80 msec
Ended Job = job_1607229408127_0002
MapReduce Jobs Launched:
Stage-Stage-1: Map: 6  Reduce: 1   Cumulative CPU: 646.08 sec   HDFS Read: 1777492140 HDFS Write: 107 SUCCESS
Total MapReduce CPU Time Spent: 10 minutes 46 seconds 80 msec
OK
3437474
Time taken: 1012.67 seconds, Fetched: 1 row(s)
```

select sum(retweet_count) from one where(lang='en')

This query gives the number of retweet counts of all tweets on the coronavirus topic tweeted in English.



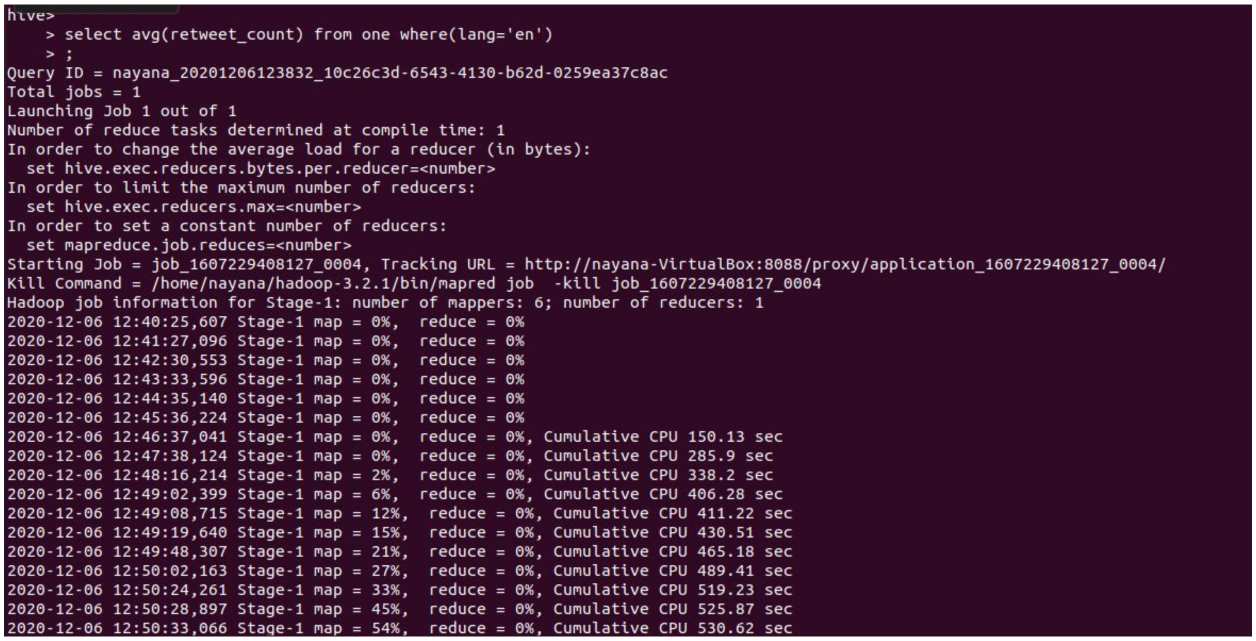
```
hive>
    > select sum(retweet_count) from one where(lang='en');
Query ID = nayana_20201206121447_f1b2f8e4-7e02-4565-b768-d99f1c395d91
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<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>
Starting Job = job_1607229408127_0003, Tracking URL = http://nayana-VirtualBox:8088/proxy/application_1607229408127_0003/
Kill Command = /home/nayana/hadoop-3.2.1/bin/mapred job  -kill job_1607229408127_0003
Hadoop job information for Stage-1: number of mappers: 6; number of reducers: 1
2020-12-06 12:16:38,811 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:17:40,614 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:18:41,629 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:19:41,937 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:20:44,520 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:21:45,017 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:22:45,246 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 145.44 sec
2020-12-06 12:23:46,774 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 276.64 sec
2020-12-06 12:24:18,780 Stage-1 map = 2%,  reduce = 0%, Cumulative CPU 323.23 sec
2020-12-06 12:25:13,375 Stage-1 map = 8%,  reduce = 0%, Cumulative CPU 392.32 sec
2020-12-06 12:25:32,834 Stage-1 map = 12%,  reduce = 0%, Cumulative CPU 417.01 sec
2020-12-06 12:25:38,040 Stage-1 map = 16%,  reduce = 0%, Cumulative CPU 425.68 sec
2020-12-06 12:25:39,974 Stage-1 map = 19%,  reduce = 0%, Cumulative CPU 428.05 sec
2020-12-06 12:25:54,874 Stage-1 map = 25%,  reduce = 0%, Cumulative CPU 452.6 sec
2020-12-06 12:26:13,691 Stage-1 map = 31%,  reduce = 0%, Cumulative CPU 474.39 sec
2020-12-06 12:26:27,257 Stage-1 map = 37%,  reduce = 0%, Cumulative CPU 493.1 sec
2020-12-06 12:26:38,122 Stage-1 map = 42%,  reduce = 0%, Cumulative CPU 505.89 sec
```



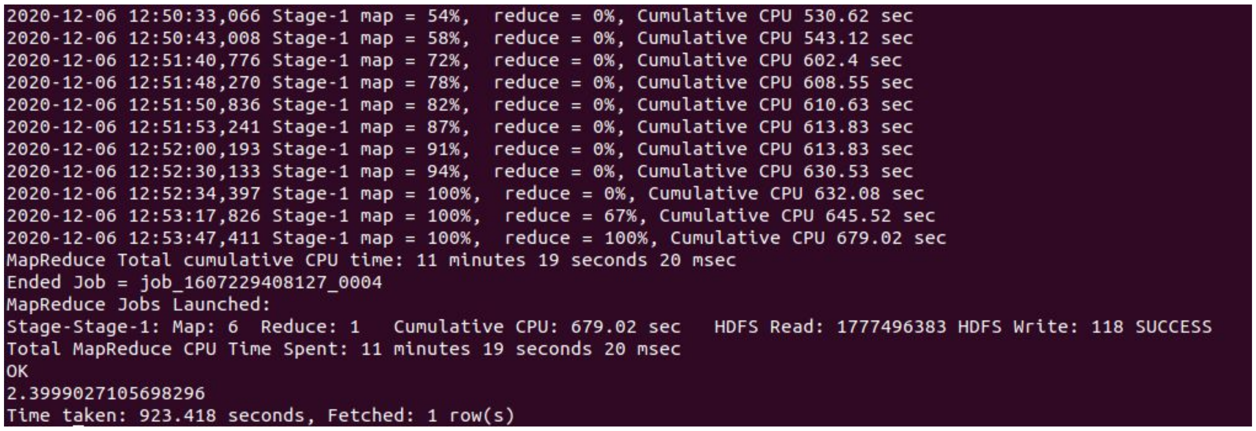
```
2020-12-06 12:25:39,974 Stage-1 map = 19%,  reduce = 0%, Cumulative CPU 428.05 sec
2020-12-06 12:25:54,874 Stage-1 map = 25%,  reduce = 0%, Cumulative CPU 452.6 sec
2020-12-06 12:26:13,691 Stage-1 map = 31%,  reduce = 0%, Cumulative CPU 474.39 sec
2020-12-06 12:26:27,257 Stage-1 map = 37%,  reduce = 0%, Cumulative CPU 493.1 sec
2020-12-06 12:26:38,122 Stage-1 map = 42%,  reduce = 0%, Cumulative CPU 505.89 sec
2020-12-06 12:26:41,504 Stage-1 map = 51%,  reduce = 0%, Cumulative CPU 511.65 sec
2020-12-06 12:26:54,353 Stage-1 map = 54%,  reduce = 0%, Cumulative CPU 525.97 sec
2020-12-06 12:26:58,416 Stage-1 map = 62%,  reduce = 0%, Cumulative CPU 531.21 sec
2020-12-06 12:27:35,104 Stage-1 map = 72%,  reduce = 0%, Cumulative CPU 575.23 sec
2020-12-06 12:27:43,611 Stage-1 map = 76%,  reduce = 0%, Cumulative CPU 582.32 sec
2020-12-06 12:27:47,527 Stage-1 map = 91%,  reduce = 0%, Cumulative CPU 587.76 sec
2020-12-06 12:27:54,420 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 590.85 sec
2020-12-06 12:28:52,508 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 604.55 sec
2020-12-06 12:29:06,390 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 623.87 sec
MapReduce Total cumulative CPU time: 10 minutes 23 seconds 870 msec
Ended Job = job_1607229408127_0003
MapReduce Jobs Launched:
Stage-Stage-1: Map: 6  Reduce: 1   Cumulative CPU: 623.87 sec   HDFS Read: 1777492136 HDFS Write: 107 SUCCESS
Total MapReduce CPU Time Spent: 10 minutes 23 seconds 870 msec
OK
3670548
Time taken: 864.684 seconds, Fetched: 1 row(s)
```

select avg(retweet_count) from one where(lang='en');

This query gives the average number of retweets per tweet written in english.



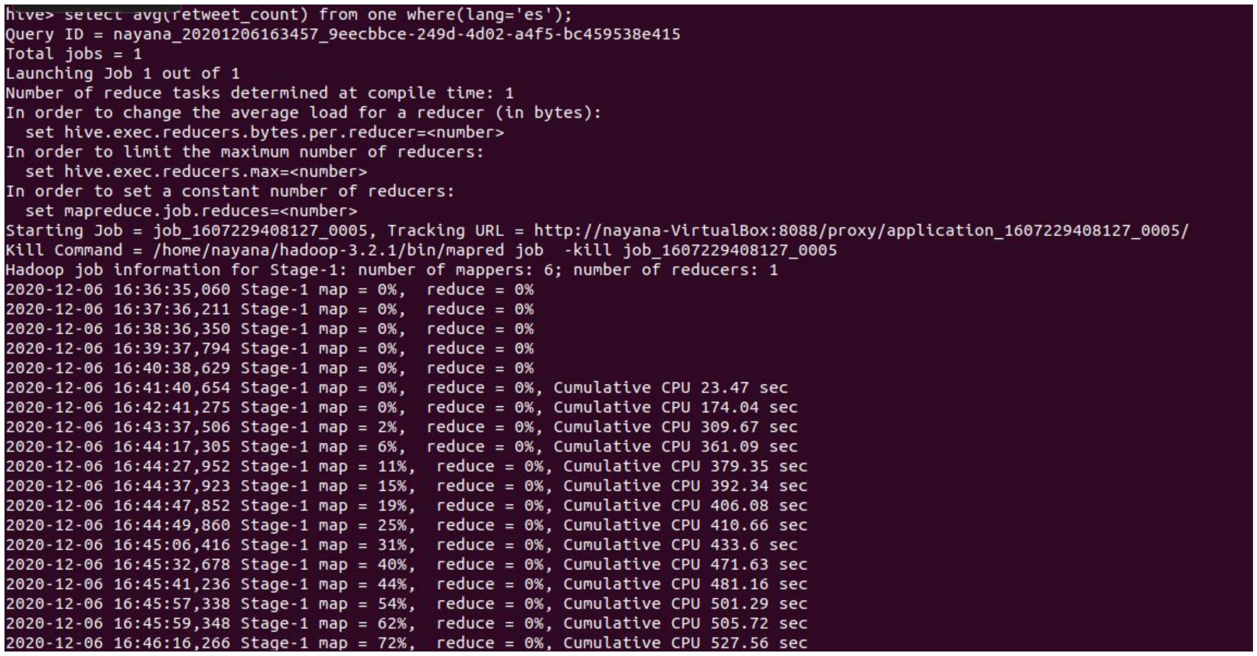
```
hive>
    > select avg(retweet_count) from one where(lang='en')
    > ;
Query ID = nayana_20201206123832_10c26c3d-6543-4130-b62d-0259ea37c8ac
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  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>
Starting Job = job_1607229408127_0004, Tracking URL = http://nayana-VirtualBox:8088/proxy/application_1607229408127_0004/
Kill Command = /home/nayana/hadoop-3.2.1/bin/mapred job  -kill job_1607229408127_0004
Hadoop job information for Stage-1: number of mappers: 6; number of reducers: 1
2020-12-06 12:40:25,607 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:41:27,096 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:42:30,553 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:43:33,596 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:44:35,140 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:45:36,224 Stage-1 map = 0%,  reduce = 0%
2020-12-06 12:46:37,041 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 150.13 sec
2020-12-06 12:47:38,124 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 285.9 sec
2020-12-06 12:48:16,214 Stage-1 map = 2%,  reduce = 0%, Cumulative CPU 338.2 sec
2020-12-06 12:49:02,399 Stage-1 map = 6%,  reduce = 0%, Cumulative CPU 406.28 sec
2020-12-06 12:49:08,715 Stage-1 map = 12%,  reduce = 0%, Cumulative CPU 411.22 sec
2020-12-06 12:49:19,640 Stage-1 map = 15%,  reduce = 0%, Cumulative CPU 430.51 sec
2020-12-06 12:49:48,307 Stage-1 map = 21%,  reduce = 0%, Cumulative CPU 465.18 sec
2020-12-06 12:50:02,163 Stage-1 map = 27%,  reduce = 0%, Cumulative CPU 489.41 sec
2020-12-06 12:50:24,261 Stage-1 map = 33%,  reduce = 0%, Cumulative CPU 519.23 sec
2020-12-06 12:50:28,897 Stage-1 map = 45%,  reduce = 0%, Cumulative CPU 525.87 sec
2020-12-06 12:50:33,066 Stage-1 map = 54%,  reduce = 0%, Cumulative CPU 530.62 sec
```



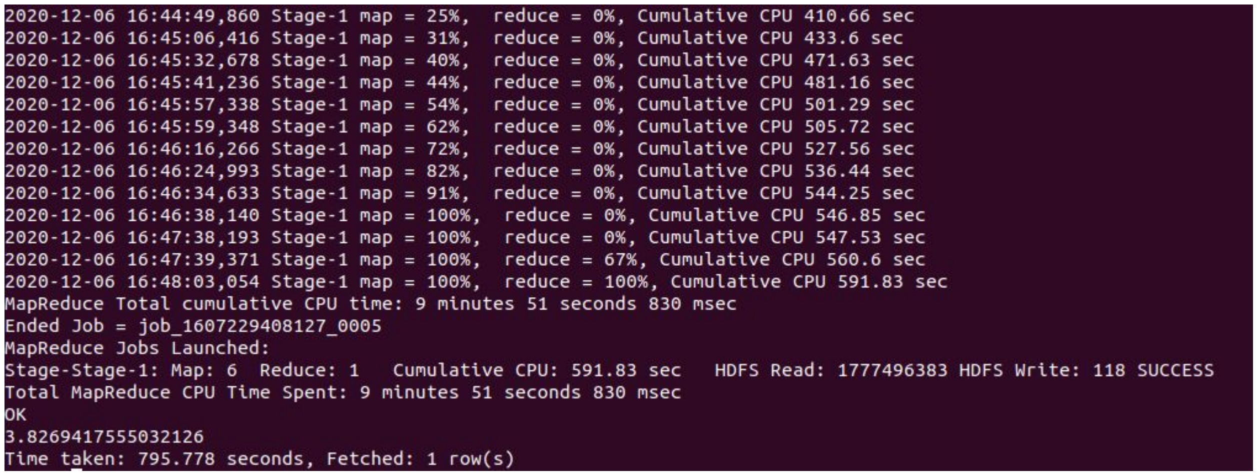
```
2020-12-06 12:50:33,066 Stage-1 map = 54%,  reduce = 0%, Cumulative CPU 530.62 sec
2020-12-06 12:50:43,008 Stage-1 map = 58%,  reduce = 0%, Cumulative CPU 543.12 sec
2020-12-06 12:51:40,776 Stage-1 map = 72%,  reduce = 0%, Cumulative CPU 602.4 sec
2020-12-06 12:51:48,270 Stage-1 map = 78%,  reduce = 0%, Cumulative CPU 608.55 sec
2020-12-06 12:51:50,836 Stage-1 map = 82%,  reduce = 0%, Cumulative CPU 610.63 sec
2020-12-06 12:51:53,241 Stage-1 map = 87%,  reduce = 0%, Cumulative CPU 613.83 sec
2020-12-06 12:52:00,193 Stage-1 map = 91%,  reduce = 0%, Cumulative CPU 613.83 sec
2020-12-06 12:52:30,133 Stage-1 map = 94%,  reduce = 0%, Cumulative CPU 630.53 sec
2020-12-06 12:52:34,397 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 632.08 sec
2020-12-06 12:53:17,826 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 645.52 sec
2020-12-06 12:53:47,411 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 679.02 sec
MapReduce Total cumulative CPU time: 11 minutes 19 seconds 20 msec
Ended Job = job_1607229408127_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 6  Reduce: 1   Cumulative CPU: 679.02 sec   HDFS Read: 1777496383 HDFS Write: 118 SUCCESS
Total MapReduce CPU Time Spent: 11 minutes 19 seconds 20 msec
OK
2.3999027105698296
Time taken: 923.418 seconds, Fetched: 1 row(s)
```

select avg(retweet_count) from one where(lang='es')

This gives the average number of retweets of tweets written in español



```
hive> select avg(retweet_count) from one where(lang='es');
Query ID = nayana_20201206163457_9eecbbce-249d-4d02-a4f5-bc459538e415
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
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>
Starting Job = job_1607229408127_0005, Tracking URL = http://nayana-VirtualBox:8088/proxy/application_1607229408127_0005/
Kill Command = /home/nayana/hadoop-3.2.1/bin/mapred job  -kill job_1607229408127_0005
Hadoop job information for Stage-1: number of mappers: 6; number of reducers: 1
2020-12-06 16:36:35,060 Stage-1 map = 0%,  reduce = 0%
2020-12-06 16:37:36,211 Stage-1 map = 0%,  reduce = 0%
2020-12-06 16:38:36,350 Stage-1 map = 0%,  reduce = 0%
2020-12-06 16:39:37,794 Stage-1 map = 0%,  reduce = 0%
2020-12-06 16:40:38,629 Stage-1 map = 0%,  reduce = 0%
2020-12-06 16:41:40,654 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 23.47 sec
2020-12-06 16:42:41,275 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 174.04 sec
2020-12-06 16:43:37,506 Stage-1 map = 2%,  reduce = 0%, Cumulative CPU 309.67 sec
2020-12-06 16:44:17,305 Stage-1 map = 6%,  reduce = 0%, Cumulative CPU 361.09 sec
2020-12-06 16:44:27,952 Stage-1 map = 11%,  reduce = 0%, Cumulative CPU 379.35 sec
2020-12-06 16:44:37,923 Stage-1 map = 15%,  reduce = 0%, Cumulative CPU 392.34 sec
2020-12-06 16:44:47,852 Stage-1 map = 19%,  reduce = 0%, Cumulative CPU 406.08 sec
2020-12-06 16:44:49,860 Stage-1 map = 25%,  reduce = 0%, Cumulative CPU 410.66 sec
2020-12-06 16:45:06,416 Stage-1 map = 31%,  reduce = 0%, Cumulative CPU 433.6 sec
2020-12-06 16:45:32,678 Stage-1 map = 40%,  reduce = 0%, Cumulative CPU 471.63 sec
2020-12-06 16:45:41,236 Stage-1 map = 44%,  reduce = 0%, Cumulative CPU 481.16 sec
2020-12-06 16:45:57,338 Stage-1 map = 54%,  reduce = 0%, Cumulative CPU 501.29 sec
2020-12-06 16:45:59,348 Stage-1 map = 62%,  reduce = 0%, Cumulative CPU 505.72 sec
2020-12-06 16:46:16,266 Stage-1 map = 72%,  reduce = 0%, Cumulative CPU 527.56 sec
```



```
2020-12-06 16:44:49,860 Stage-1 map = 25%,  reduce = 0%, Cumulative CPU 410.66 sec
2020-12-06 16:45:06,416 Stage-1 map = 31%,  reduce = 0%, Cumulative CPU 433.6 sec
2020-12-06 16:45:32,678 Stage-1 map = 40%,  reduce = 0%, Cumulative CPU 471.63 sec
2020-12-06 16:45:41,236 Stage-1 map = 44%,  reduce = 0%, Cumulative CPU 481.16 sec
2020-12-06 16:45:57,338 Stage-1 map = 54%,  reduce = 0%, Cumulative CPU 501.29 sec
2020-12-06 16:45:59,348 Stage-1 map = 62%,  reduce = 0%, Cumulative CPU 505.72 sec
2020-12-06 16:46:16,266 Stage-1 map = 72%,  reduce = 0%, Cumulative CPU 527.56 sec
2020-12-06 16:46:24,993 Stage-1 map = 82%,  reduce = 0%, Cumulative CPU 536.44 sec
2020-12-06 16:46:34,633 Stage-1 map = 91%,  reduce = 0%, Cumulative CPU 544.25 sec
2020-12-06 16:46:38,140 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 546.85 sec
2020-12-06 16:47:38,193 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 547.53 sec
2020-12-06 16:47:39,371 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 560.6 sec
2020-12-06 16:48:03,054 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 591.83 sec
MapReduce Total cumulative CPU time: 9 minutes 51 seconds 830 msec
Ended Job = job_1607229408127_0005
MapReduce Jobs Launched:
Stage-Stage-1: Map: 6  Reduce: 1   Cumulative CPU: 591.83 sec   HDFS Read: 1777496383 HDFS Write: 118 SUCCESS
Total MapReduce CPU Time Spent: 9 minutes 51 seconds 830 msec
OK
3.8269417555032126
Time taken: 795.778 seconds, Fetched: 1 row(s)
```

Surprisingly, according to the last two queries, the average number of retweets in español (3.8269) is greater than the average number of retweets in english (2.399). This could probably be because English is more popular than español. Most people use English to tweet (i.e, even people who don't have many followers, and hence lesser retweets). This might have brought down the average retweet_count.

Insights from the dataset

1) Top five retweets replied back

```
retweet=df['retweet_count','text'].sort(col('retweet_count').desc()).rdd.flatMap(lambda x: x).collect()

#retweet.show(5,False)
for i in range(5):
    retweet[2*i+1]=retweet[2*i+1].lower()
    print(i,']', retweet[2*i+1],'\n')
```

```
0 ] dear   ji  matching steps with your relentless efforts to save our people from  coronavirus we have made this pra
yer to appreciate the spirit of  coronawarriors  amp  make the cry of  humanity reach out to the almighty   sambhaall
ena

1 ] ten  10  new ventilators installed at rims on 07 04 2020   ready to overcome  covid 19   thanks to dr shanta sing
h  director rims and other officials for your tireless efforts

2 ] big   rajasthan govt is going all out to take over 84 private hospitals  a circular has been sent to all hospital
s to give over the control within two hours of when the order will be released  they have been informed today  covid
19india  coronavirus

3 ] 12 staffers of jaslok hospital including nurses  technicians  helpers and a cook turn positive for covid 19  swab
s of 300 staffers and patients taken  hospital declared a containment zone  covid 19

4 ] i love my india      stayhomestaysafe
```

- Top 5 FavouriteTweets

2)top five most favourited tweets

```
favourite_tweets = df['favourites_count','text'].sort(col('favourites_count').desc()).rdd.flatMap(lambda x: x).coll
for i in range(5):
    favourite_tweets[2*i+1]=favourite_tweets[2*i+1].lower()
    print(i,']', favourite_tweets[2*i+1],'\n')
```

```
0 ] stop the stigma   covid 19   a south mumbai coronavirus survivor speaks to me about the harassment he faced  i pr
ay this doesn t repeat  let s show some empathy  love and compassion  he battled and survived this and is back to nor
mal

1 ] a south mumbai resident  a  coronavirus survivor speaks to me  harassed  named  shamed  affluent  educated neighb
ourhood misbehaves  are we  covidiots to stigmatise this  society needs to show empathy  get real  accept and coopera
te  interview will be on air soon

2 ] news reports coming in    taken into icu  condition worsens  coronavirus

3 ]   ji is suffering from multiple ailments including hypertension  chronic kidney disease and hypertrophy of prosta
te  in the wake of the apex court s directive on  covid 19 an ailing and ageing   chief must be given parole   releas
elaluprasadyadav

4 ] a tiger tests positive for  covid19
```

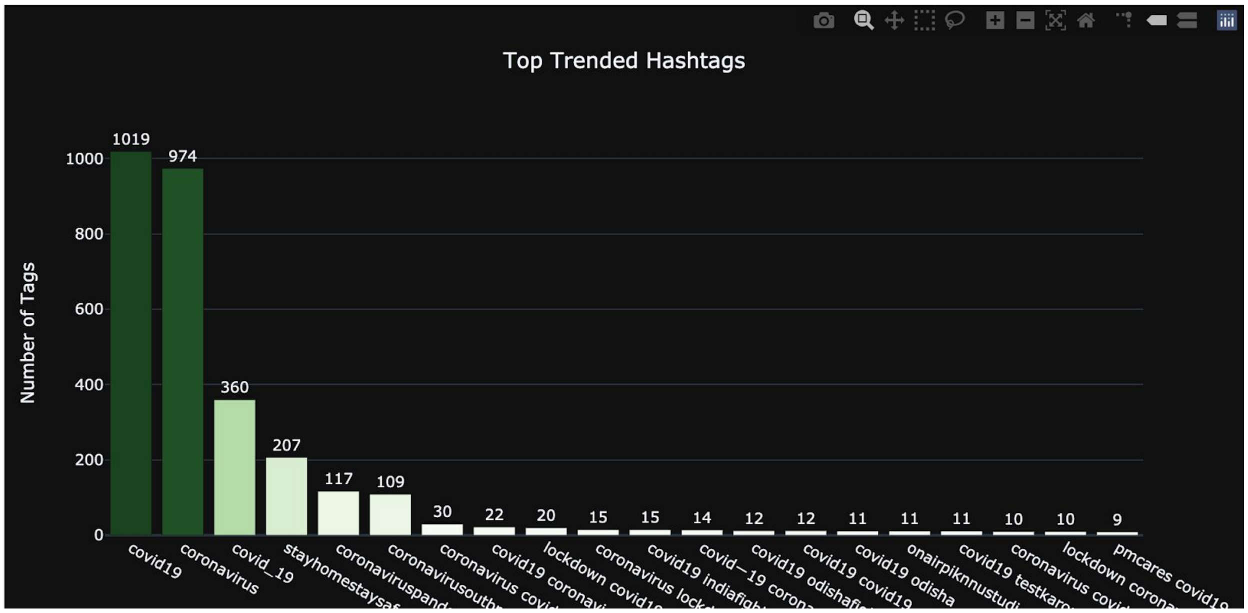
- Viral Hashtags

```python
import re
def find_hash(text):
    line=re.findall(r'(?<=#)\w+',text)
    return " ".join(line)
data['hash']=data['text'].apply(lambda x:find_hash(x))
```

```python
hastags=list(data[(data['hash'].notnull())&(data['hash']!="")]['hash'])
hastags = [each_string.lower() for each_string in hastags]
hash_df=dict(Counter(hastags))
top_hash_df=pd.DataFrame(list(hash_df.items()),columns = ['word','count']).sort_values('count',ascending=False)[:2
top_hash_df.head(10)
```

	word	count
1	covid19	1019
5	coronavirus	974
7	covid_19	360
1479	stayhomestaysafe	207
29	coronaviruspandemic	117
2	coronavirusoutbreak	109
47	coronavirus covid19	30
260	covid19 coronavirus	22
57	lockdown covid19	20
251	coronavirus lockdown	15

```python
import plotly.graph_objects as go
fig = go.Figure(go.Bar(
    x=top_hash_df['word'],y=top_hash_df['count'],
    marker={'color': top_hash_df['count'],
    'colorscale': 'greens'},
    text=top_hash_df['count'],
    textposition = "outside",
))
fig.update_layout(title_text='Top Trended Hashtags',xaxis_title="Hashtags ",
                  yaxis_title="Number of Tags ",template="plotly_dark",height=500,title_x=0.5)
fig.show()
```



3. The sentimental analysis of the data is done in Pyspark.

- Libraries imported for NLP

```python
from textblob import TextBlob
from pyspark.sql.functions import udf
from pyspark.sql.types import DoubleType
import seaborn as sns
```

- Checking the value of positivity, negativity or neutrality of the sentiments

```python
def apply_blob(sentence):
    temp = TextBlob(sentence).sentiment[0]
    if temp == 0.0:
        return 0.0 # Neutral
    elif temp >= 0.0:
        return 1.0 # Positive
    else:
        return 2.0 # Negative
```

```python
sentiment = udf(apply_blob, DoubleType())
```

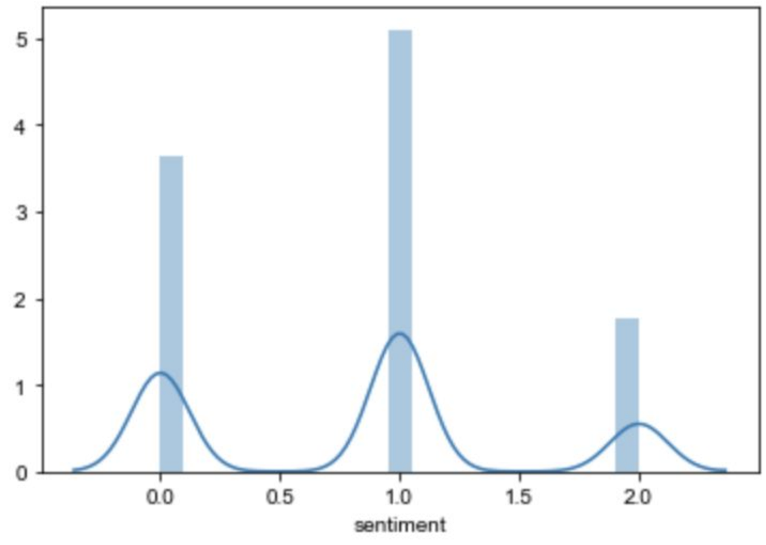
```python
result=df.withColumn("sentiment", sentiment(df['text']))
result.show()
```

```
+--------------------+----------------+-------------+-------------------+---------+---------+
|                text|favourites_count|retweet_count|               Date|     Time|sentiment|
+--------------------+----------------+-------------+-------------------+---------+---------+
|    Covid19    Soc...|           10374|            0|         2020-03-29|00:04:06Z|      0.0|
|  Very unfortunat...|             236|            0|         2020-03-29|00:17:57Z|      2.0|
|I have made a sma...|           48715|            1|         2020-03-29|00:35:14Z|      2.0|
| COVID19   Madurai...|             339|            0|         2020-03-29|00:41:09Z|      0.0|
| Coronavirus    Po...|             339|            0|         2020-03-29|00:42:40Z|      0.0|
|Because killers a...|            3798|            0|         2020-03-29|00:53:50Z|      1.0|
|   Coronavirus Tw...|           65863|            0|         2020-03-29|01:04:16Z|      0.0|
|For every article...|           32080|            0|         2020-03-29|01:24:35Z|      2.0|
|We need to unders...|           32080|            0|         2020-03-29|01:27:17Z|      1.0|
|Having said all t...|           32080|            1|         2020-03-29|01:32:48Z|      1.0|
|Is it the onset o...|           22105|            0|         2020-03-29|01:36:53Z|      0.0|
|Never in history ...|            4189|            1|         2020-03-29|01:40:05Z|      1.0|
```

Where 0.0=neutral
1.0=positive
2.0=negative

- Tweets Sentiment distribution

```
df_res_pandas = result.toPandas()
sns.distplot(df_res_pandas['sentiment'])
sns.set(rc={'figure.figsize':(11.7,8.27)})
```



- To determine the polarity of the tweets:-

```
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import matplotlib.pyplot as plt
stopwords = set(STOPWORDS)
```

```
stopwords
```

```
'to',
'too',
'under',
'until',
'up',
'very',
'was',
"wasn't",
'we',
"we'd",
"we'll",
"we're",
"we've",
'were',
"weren't",
'what',
"what's",
'when',
"when's",
'where',
```

```python
df_res_pandas['sentiment'] = ' '
df_res_pandas['polarity'] = None
for i,tweets in enumerate(df_res_pandas.text) :
    blob = TextBlob(tweets)
    df_res_pandas['polarity'][i] = blob.sentiment.polarity
    if blob.sentiment.polarity > 0 :
        df_res_pandas['sentiment'][i] = 'positive'
    elif blob.sentiment.polarity < 0 :
        df_res_pandas['sentiment'][i] = 'negative'
    else :
        df_res_pandas['sentiment'][i] = 'neutral'
df_res_pandas.head(10)
```

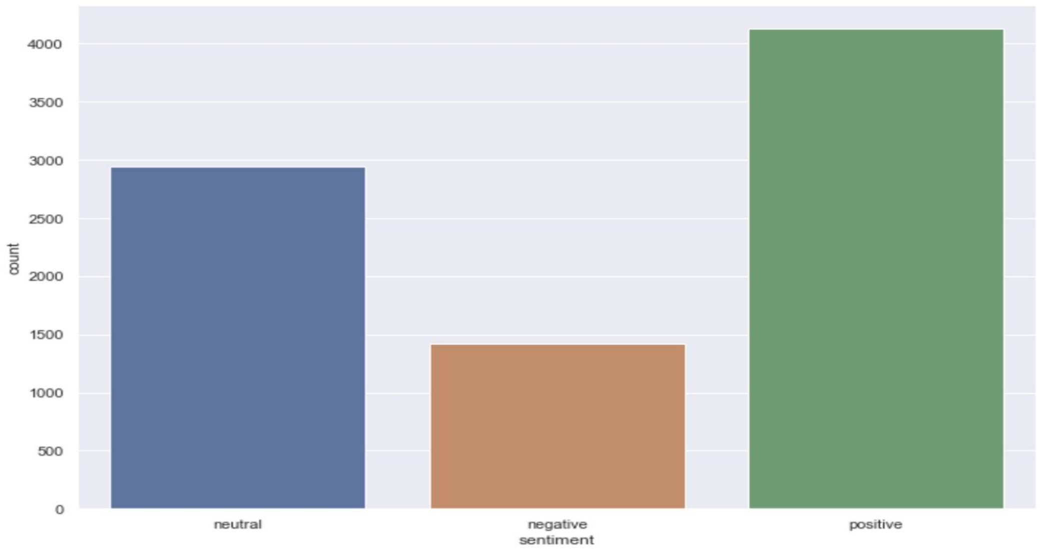
	text	favourites_count	retweet_count	Date	Time	sentiment	polarity
0	Covid19 SocialDistancing the Indian way ...	10374	0	2020-03-29	00:04:06Z	neutral	0
1	Very unfortunate to see this unplanned ill ...	236	0	2020-03-29	00:17:57Z	negative	-0.575
2	I have made a small contribution towards my co...	48715	1	2020-03-29	00:35:14Z	negative	-0.25
3	COVID19 Madurai Kavalan app uses GPS trackin...	339	0	2020-03-29	00:41:09Z	neutral	0
4	Coronavirus Police deploy Drones to monito...	339	0	2020-03-29	00:42:40Z	neutral	0
5	Because killers are having a team of activists...	3798	0	2020-03-29	00:53:50Z	positive	0.09375
6	Coronavirus Tweets from Indian authorities ...	65863	0	2020-03-29	01:04:16Z	neutral	0
7	For every article and every opinion attacking ...	32080	0	2020-03-29	01:24:35Z	negative	-0.25
8	We need to understand most of India has a join...	32080	0	2020-03-29	01:27:17Z	positive	0.142424
9	Having said all this migrant workers shoukd ha...	32080	1	2020-03-29	01:32:48Z	positive	0.275

This piece of code determines how much positive, negative or neutral the tweets are. 0.09375 polarity means it is slightly positive , more towards the neutral side.

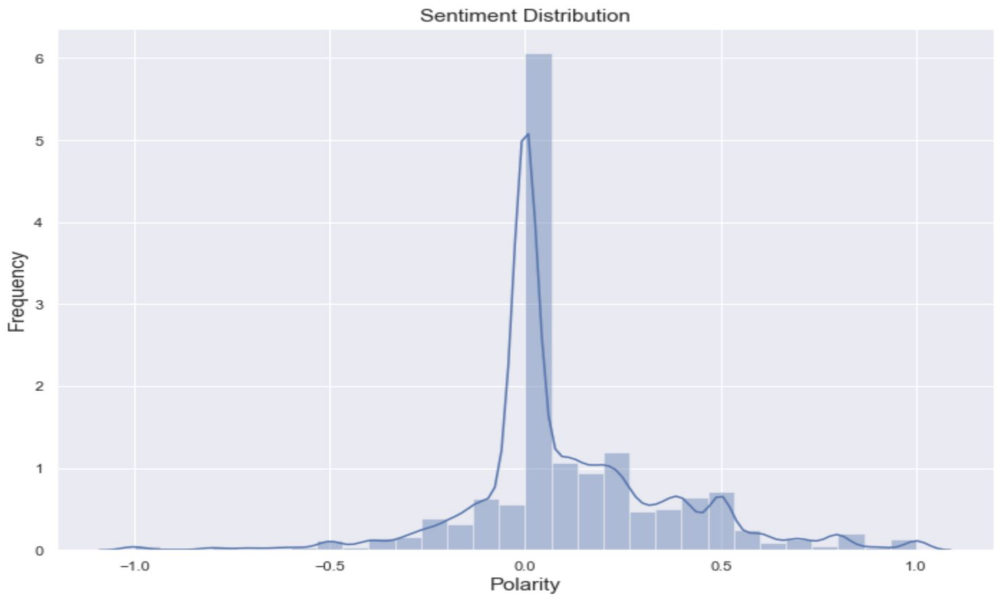
- Number of positive, negative and neutral tweets

```
print(df_res_pandas.sentiment.value_counts())
sns.countplot(x='sentiment', data = df_res_pandas);
```

```
positive    4129
neutral     2945
negative    1424
Name: sentiment, dtype: int64
```

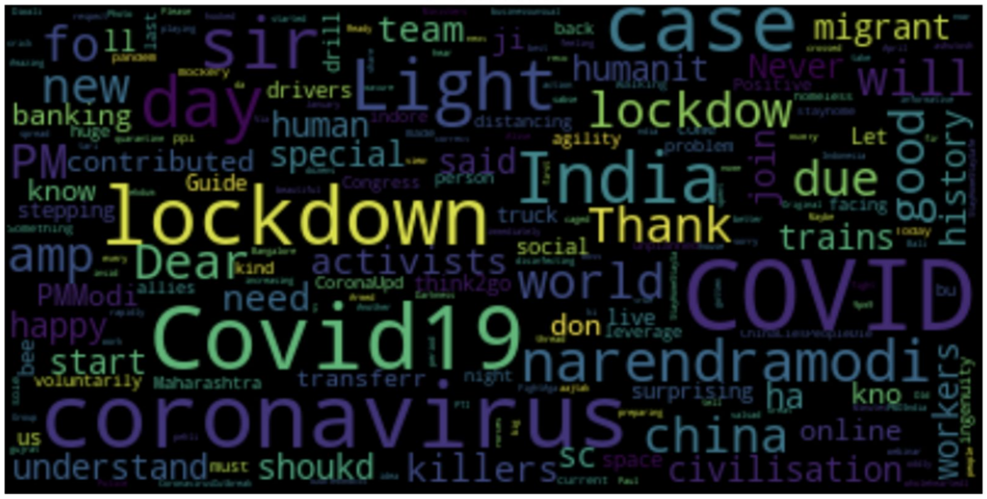


```
sns.distplot(df_res_pandas['polarity'], bins=30)
plt.title('Sentiment Distribution',size = 15)
plt.xlabel('Polarity',size = 15)
plt.ylabel('Frequency',size = 15)
plt.show();
```

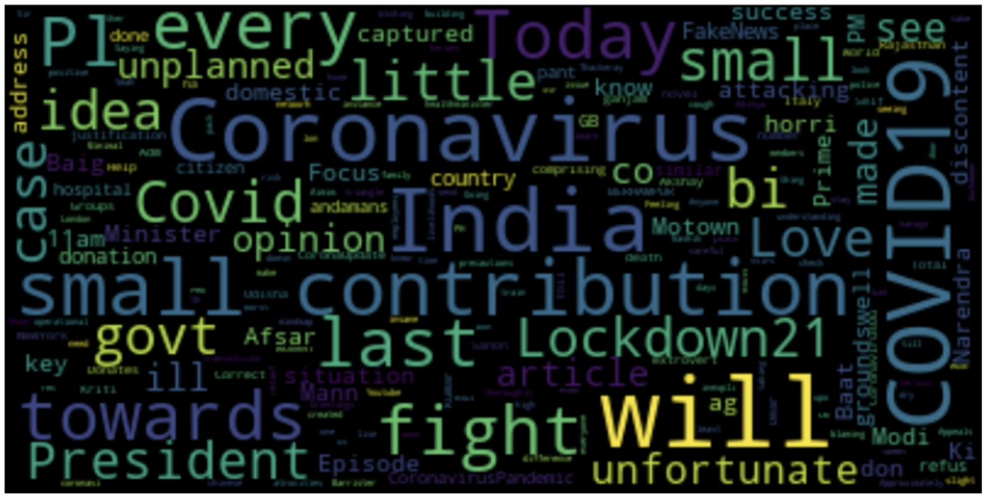


- Word Clouds:-

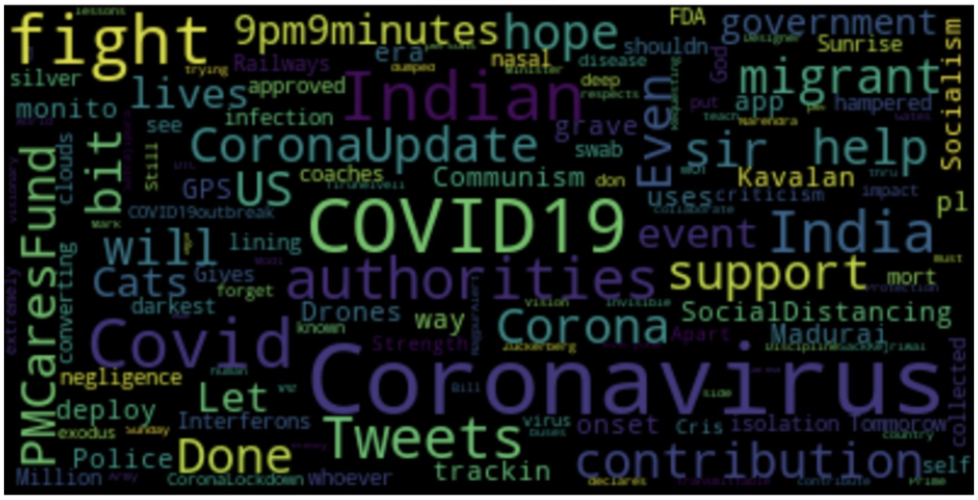
POSITIVE



NEGATIVE



NEUTRAL



Conclusion

Most of the tweets in the early april are toward and positive or neutral side during the early stages of the pandemic.