PHASE – 1

BY

Pravalhika Kampally (Class ID: 12)

Joshmitha Thammareddy (Class ID: 25)

Rupesh Sai Ram Doddala(Class ID: 7)

GOALS

Motivation:

This project is motivated to develop a system to store, analyze and visualize Twitters tweets.

Significance:

It provides a wealth of information that helps to create meaningful tweets that resonates with target audience.
Compare followers with different personas, demographics, interests and consumer behaviours to see brand measures up etc.
Watch individual Tweet performance, cumulative overview to compare monthly activity etc.

Objectives:

- Work on the tweets related to Search Engine and to figure out how to store them in Spark SQL.
- Write interesting analytical queries to explore and understand the data collected.
- Develop interesting visualizations of written queries.

Features:

Planning to add Sentiment Analysis on the data and analyze whether tweet is positive or negative.

Reference:

https://www.digitalvidya.com/blog/twitter-sentiment-analysis-introduction-and-techniques/

https://www.earthdatascience.org/courses/earth-analytics/get-data-using-apis/use-twitter-api-r/

PHASE 1

OBJECTIVE:

The main purpose of this project is to develop a system to store, analyze and visualize Twitter's tweets. The tasks to be performed in this phase are as follows :

- To work on the tweets related to the recently released mobile phones & their accessories and to figure out how to store them in Spark SQL.
- To write interesting analytical queries to explore and understand the data collected.
- To develop interesting visualizations of the above written queries.

DATASET : Twitter data set(Phones/E-Accessories)

IMPLEMENTATION:

- Initially collected the tweets in JSON format for which a Python program is written, the output of the program contains the tweets with all the details like the IDs, URLs, Hashtags, Created at, Text etc.

- The twitter data is collected on the concept based on to analyse & visualize the data regarding various phone/e-accessories.
- The extracted JSON tweets are persisted into the Apache Spark in the form of tables.
- Query written in Scala language will be sent to spark server and the outputs files are stored in the form of CSV/JSON files.
- These CSV/JSON output files are used to visualize the data using Bar Graphs, Pie Charts through Tableau.
- Key-words used in the tweets extraction are as follows :
  iphone, iphonex, iPhoneXs, iPhoneXR, iPhoneXr, iPhone, #iPhone, AirPods, mobile, watch, technology, Accessories, Mac, iOS, update, music, latest etc.

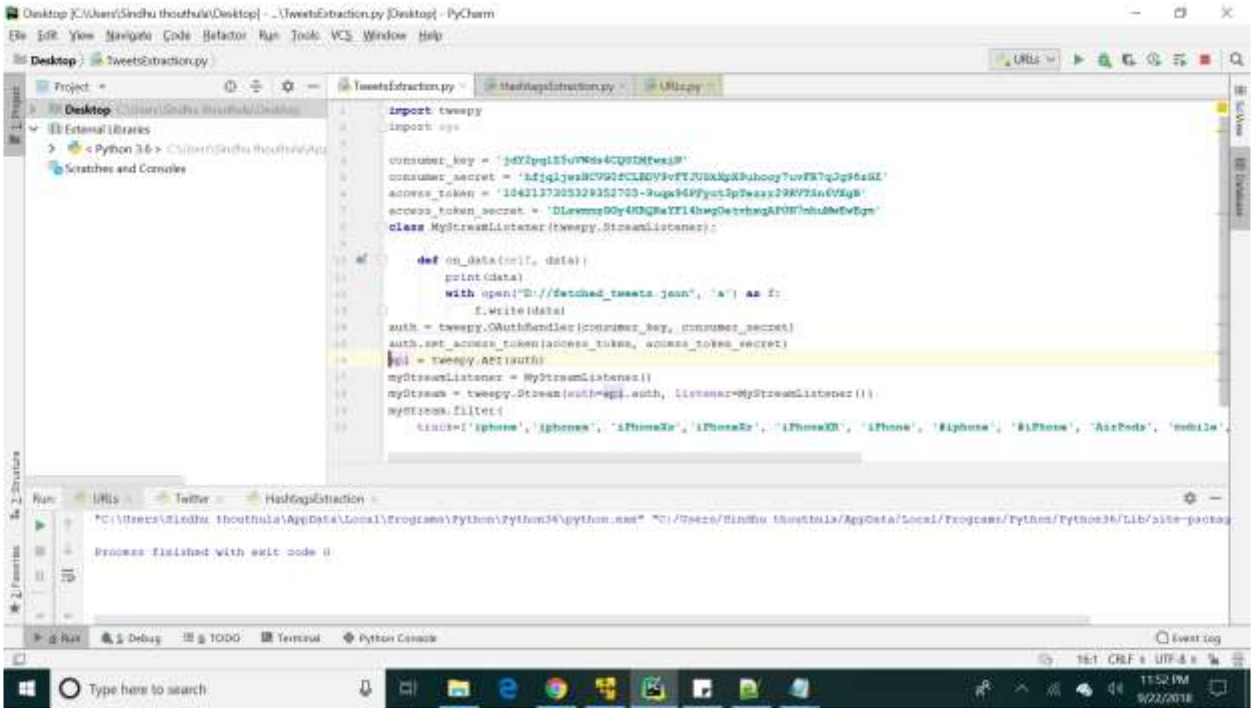
1) TWEETS COLLECTION :

- Initially a Twitter Developer Account is created using this url.
  https://developer.twitter.com
- The credentials to access the Twitter API are generated in the form of API ACCESS_TOKEN, ACCESS_SECRET, CONSUMER_KEY, CONSUMER_SECRET and are as follows :

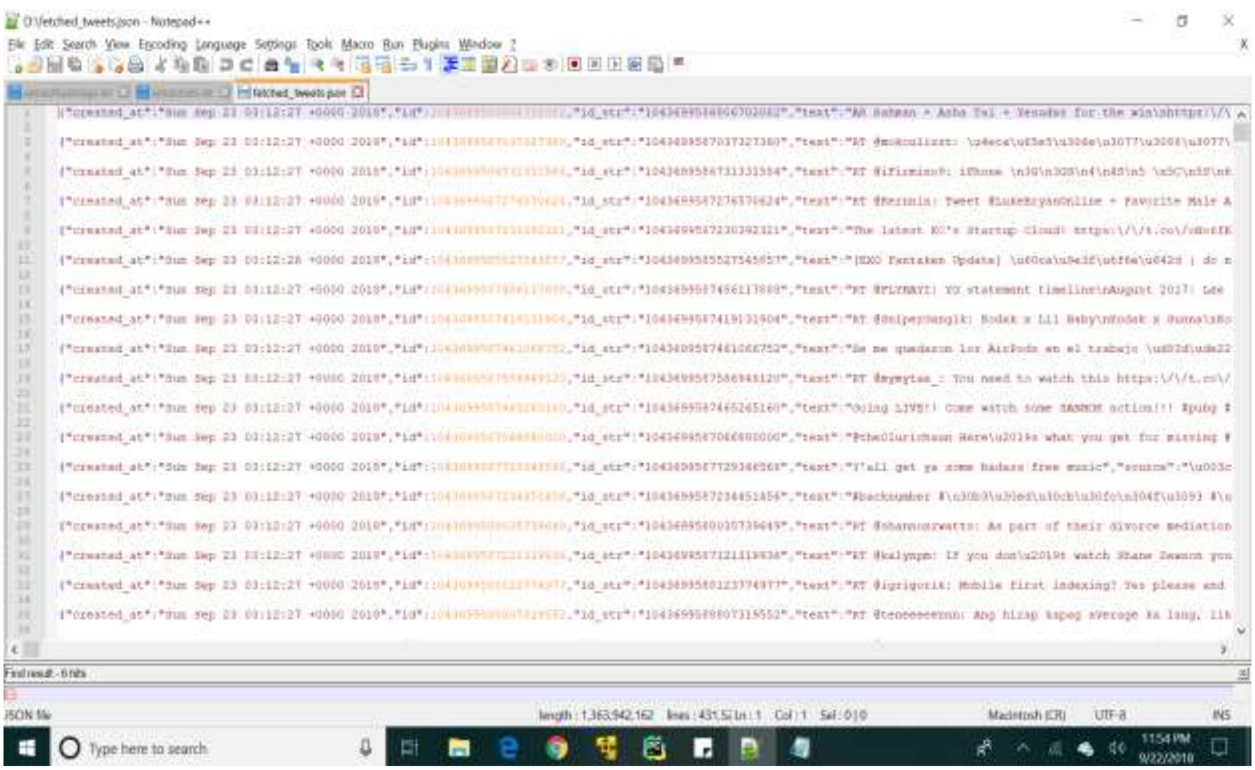
Tokens and Keys Generated :

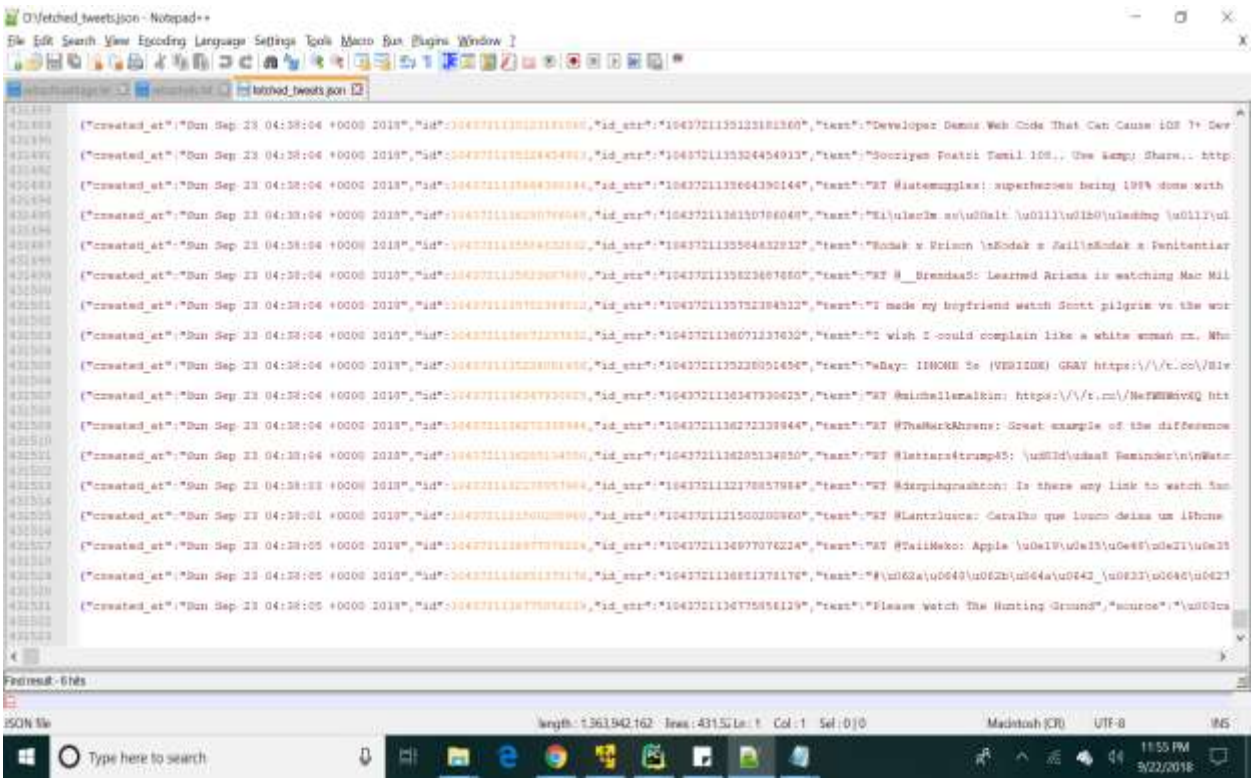
```
Consumer API keys:-

1) jdY2pqlZ5uVWds4CQGIMfwxiN (API key)

2) hfjqljwzHCVG0fCLBDV9vFTJUSkXpX9uhcoy7uvFR7qJq96zGZ (API secret key)

Access token & access token secret:-

1) 1042137305329352705-9uqx96PFyct3pTezrr29RVTSn6VXgB (Access token)

2) DLswmmzGGy4KRQReYF14hwgOetvbxqAP0N7mhuMwEwEgm (Access token secret)
Read and write (Access level)
```

Python Code for Tweets Collection :



Collected Tweets Output :



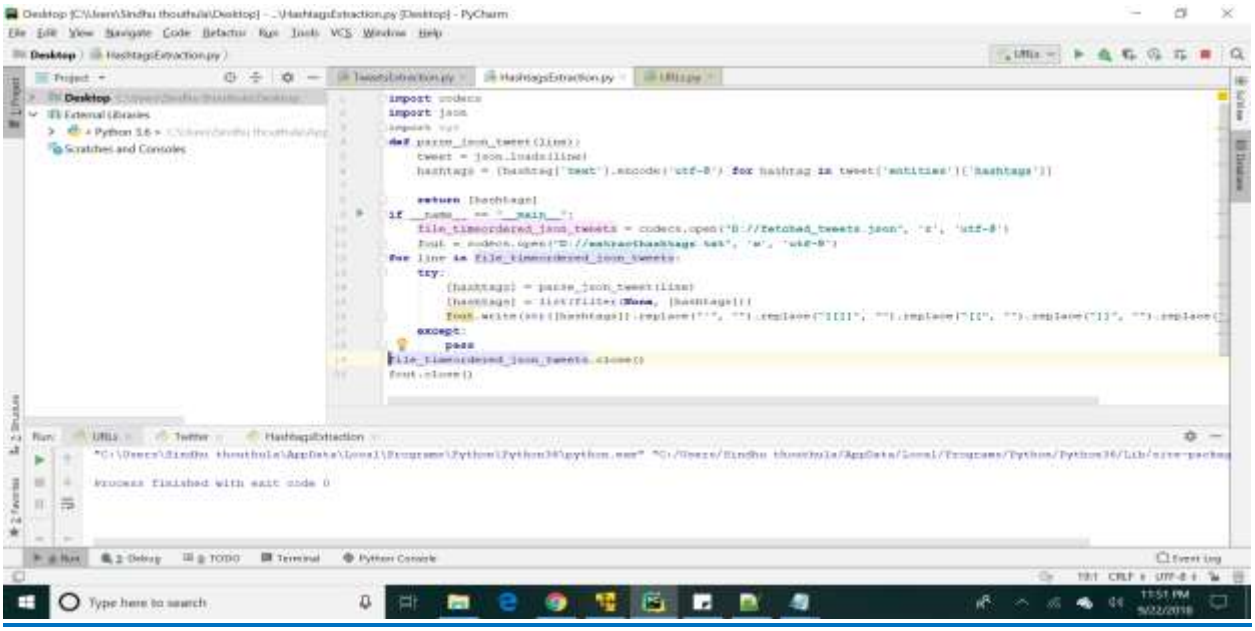


2) HASHTAGS AND URLS EXTRACTION :

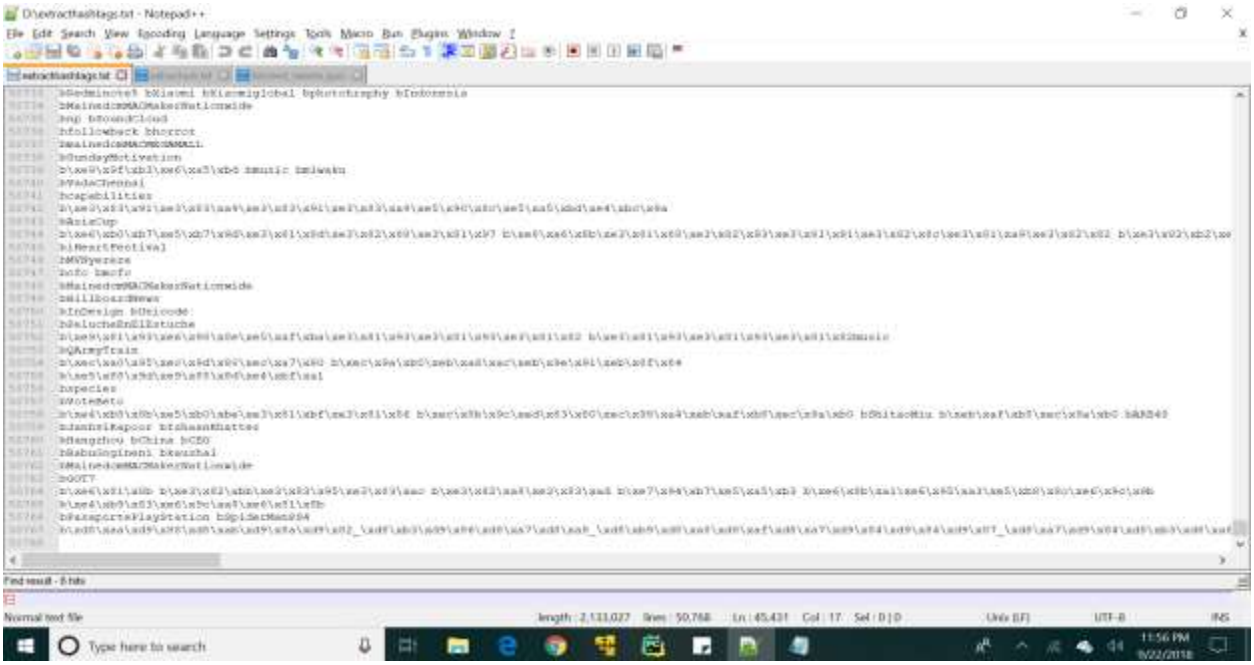
To extract the hashtags and URLs from the collected Tweets, we have again run two Python programs through which we generated the files containing the hashtags and URls alone.

RELATED SCREENSHOTS :

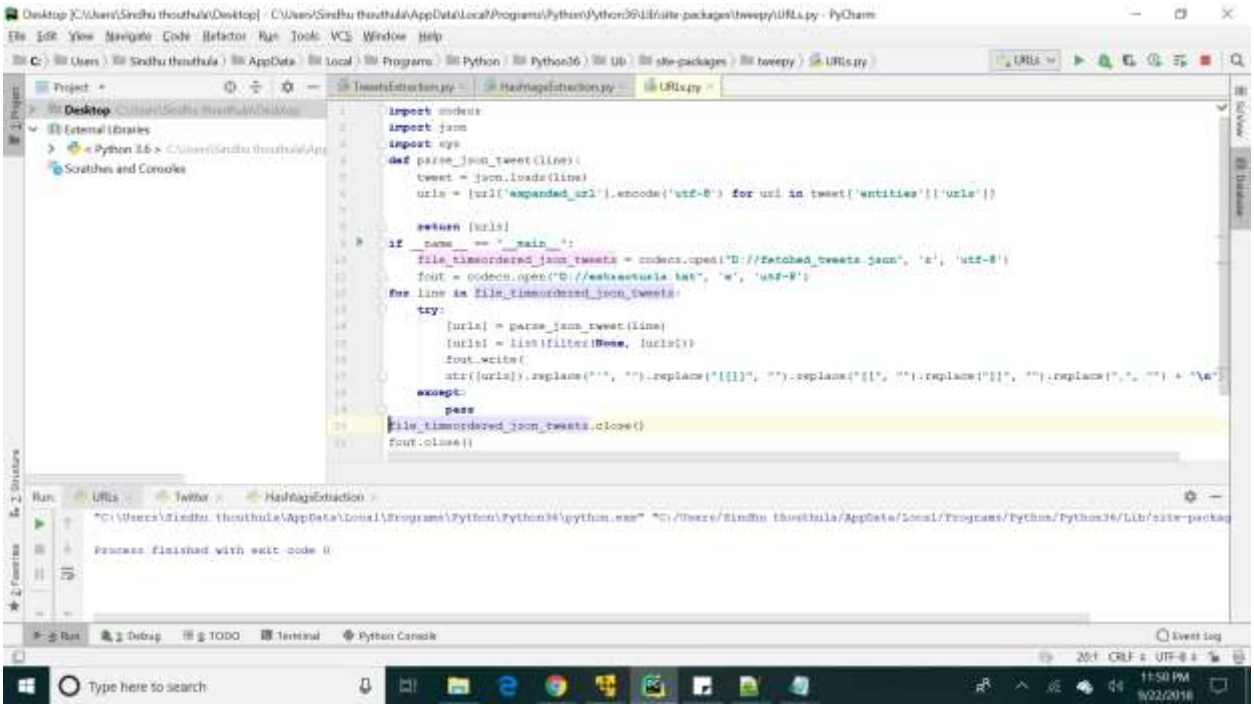
Hashtags Code :



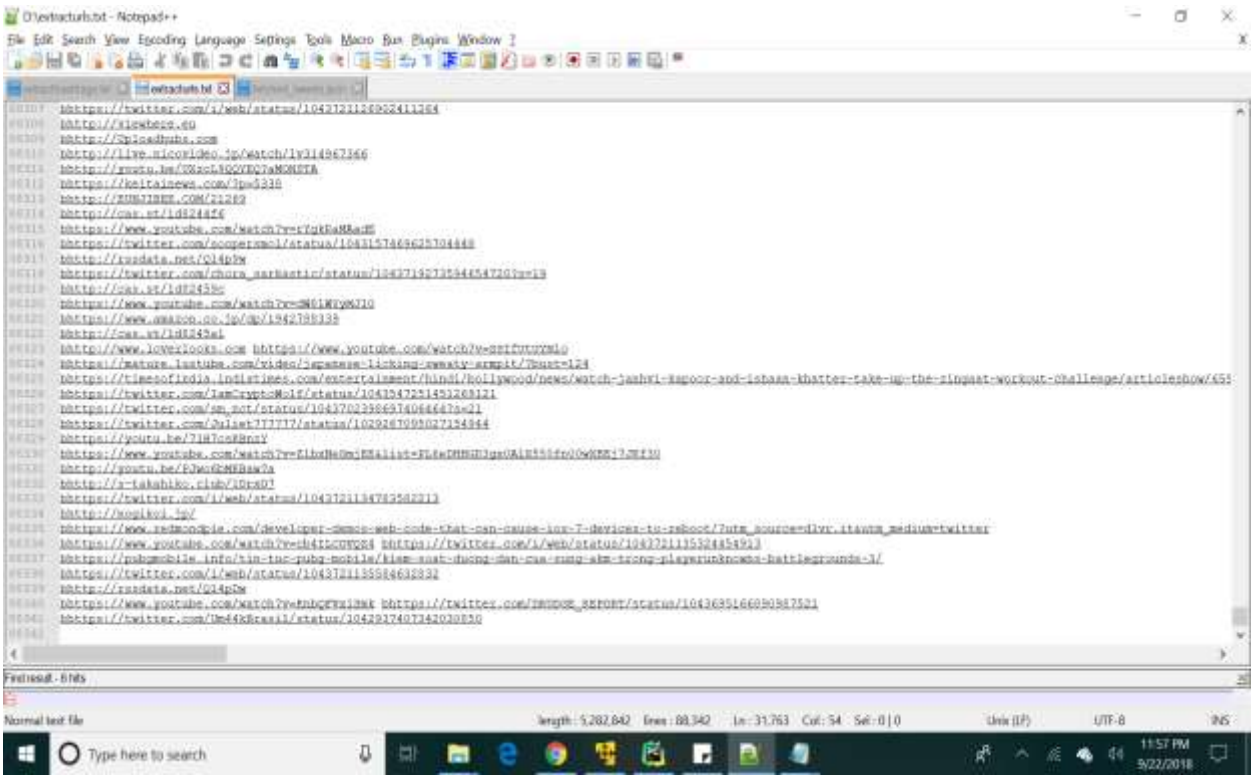
Hashtags Output :



URLs Code :

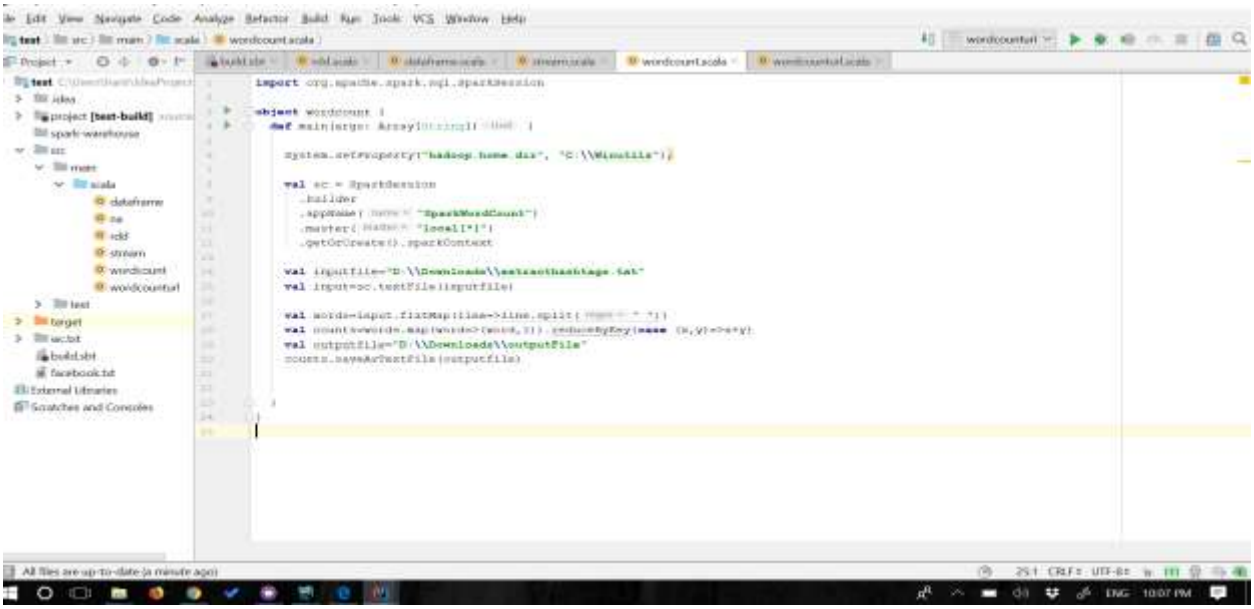


URLs Output :

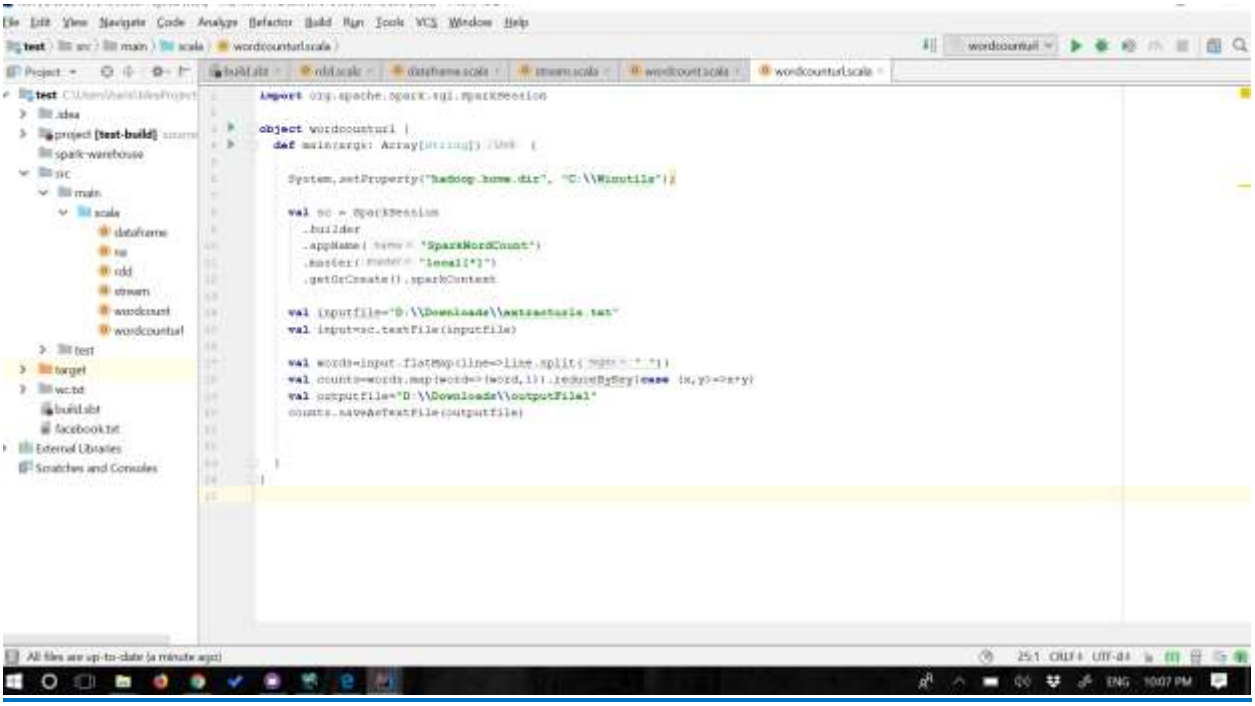


3) RUNNING THE WORDCOUNT IN APACHE HADOOP AND APACHE SPARK :

Hashtags Extraction Code – Spark :



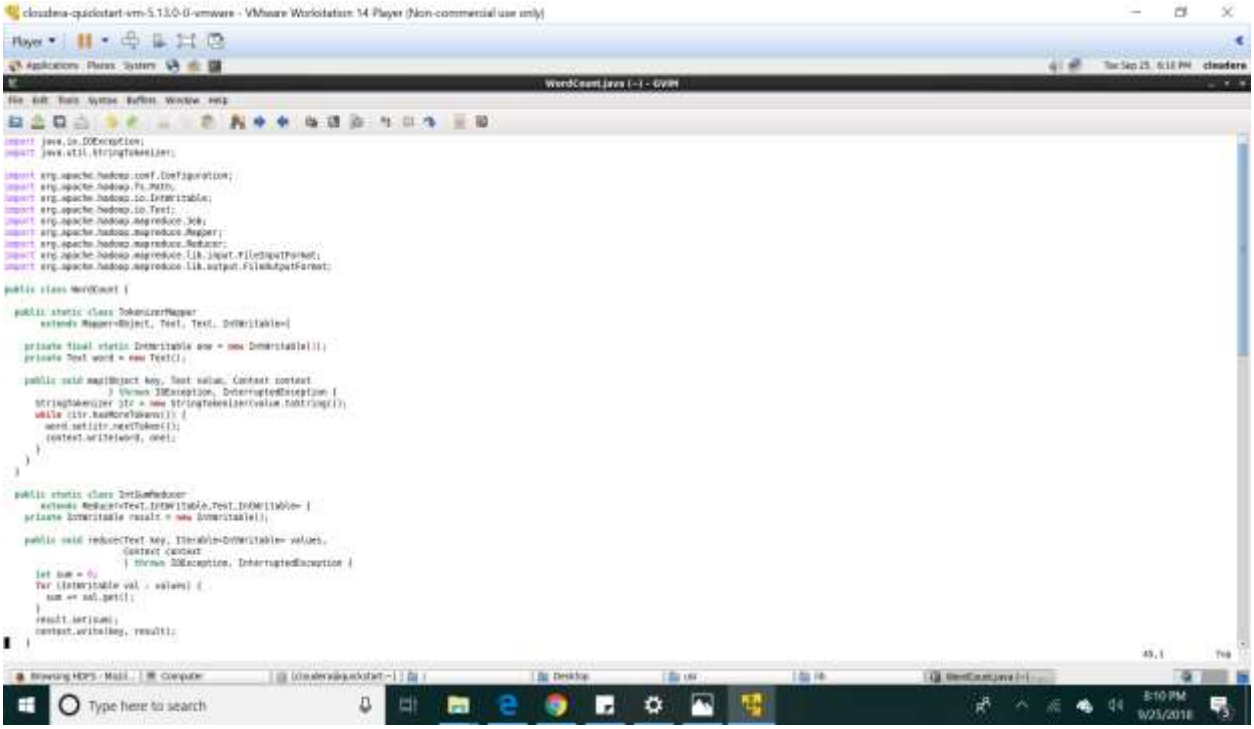
URLs Extraction Code – Spark :

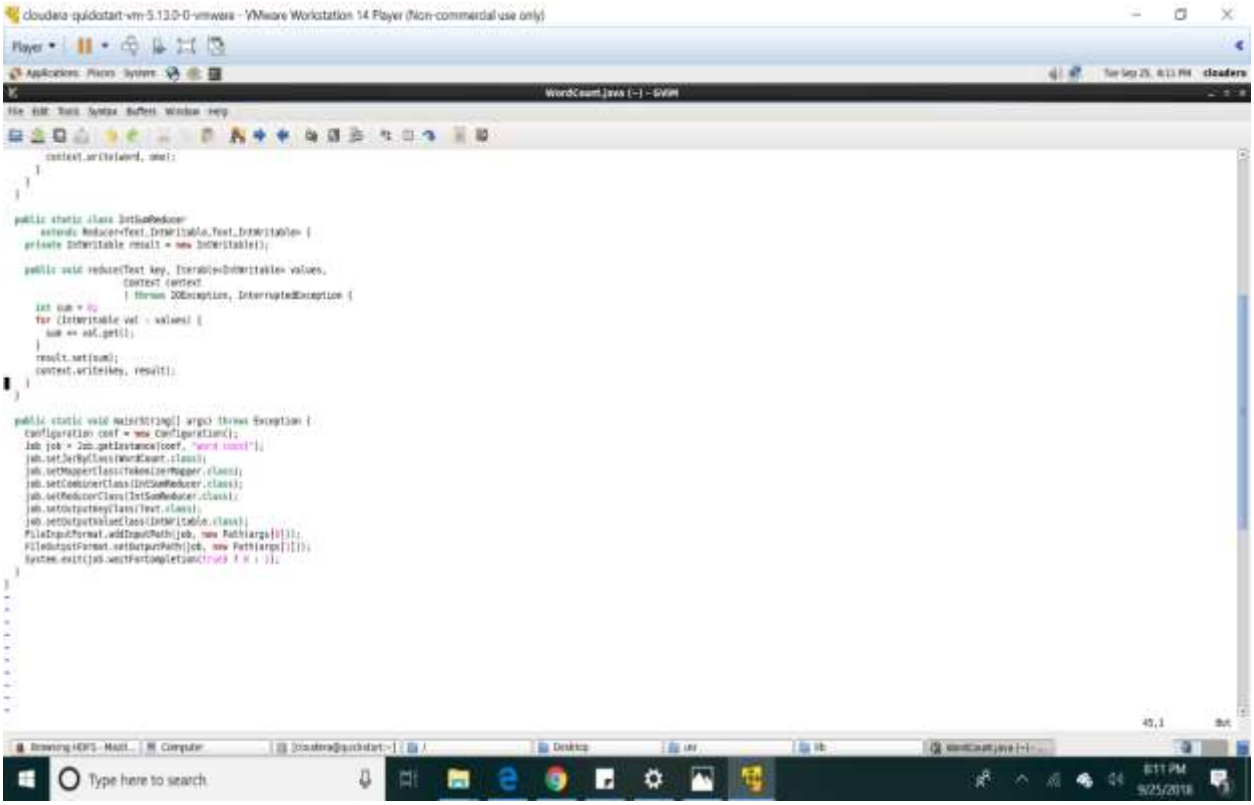


WordCount in Hadoop :

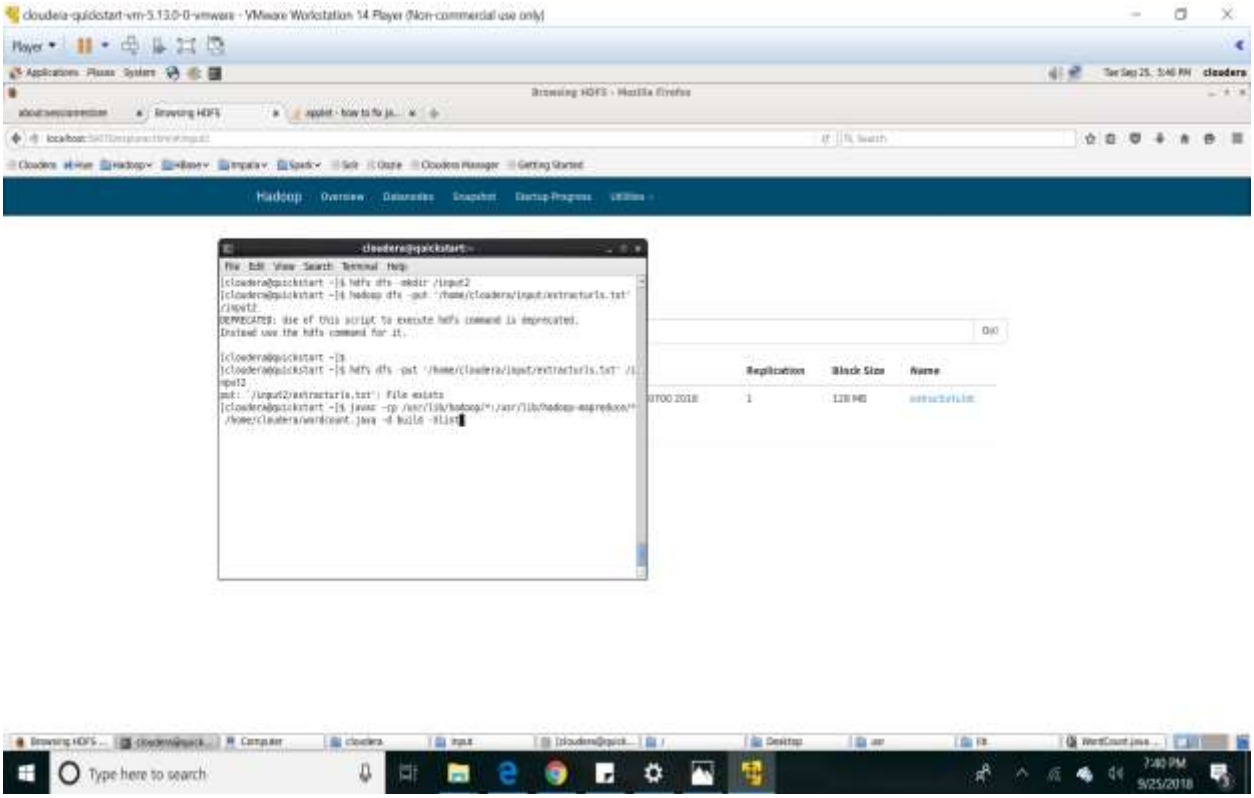
```
$ cat > /home/cloudera/extracturls.txt
$ hdfs dfs -mkdir /input2
$ hdfs dfs -put '/home/cloudera/input/extracturls.txt' /input2
$ hdfs dfs -cat /input2/extracturls.txt
$ hadoop jar wordcount.jar WordCount /input2 /wordcountoutput2
$ hdfs dfs -get /wordcountoutput2 /home/cloudera/output2
$ hdfs dfs -cat/output2/part-r-00000
```

WordCount Java Program – Hadoop :

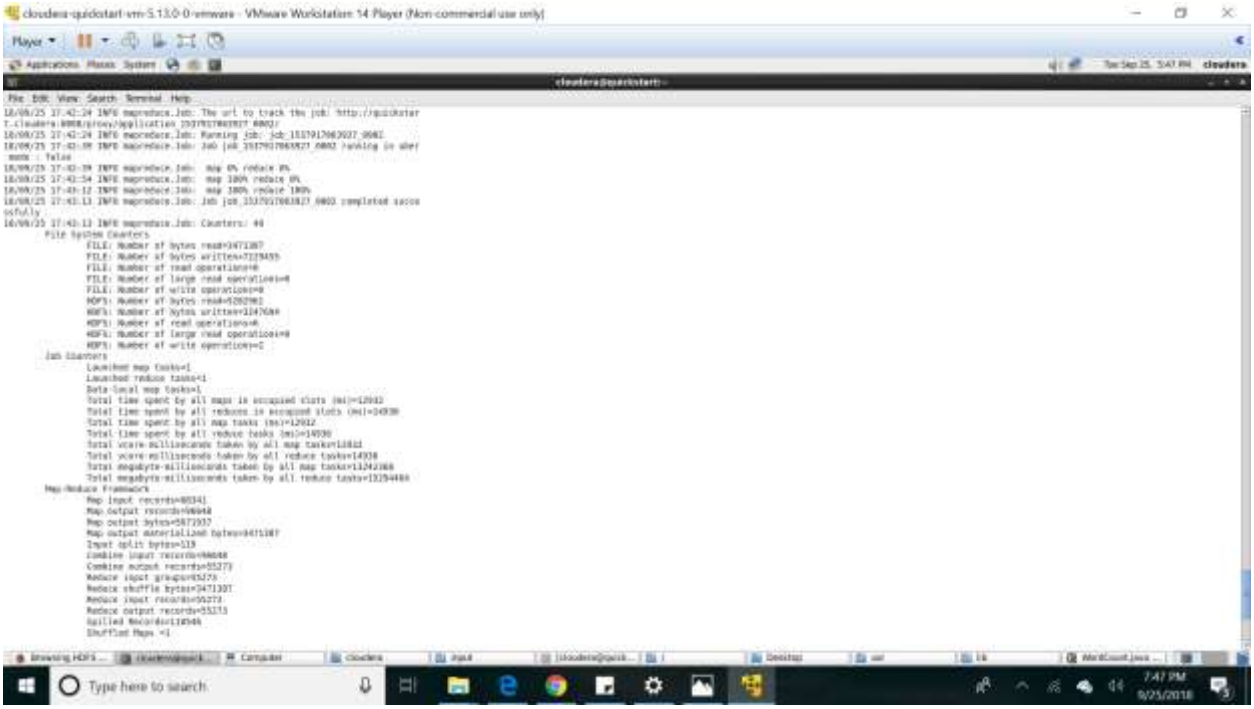




HDFS Commands to run the WordCount in Hadoop :



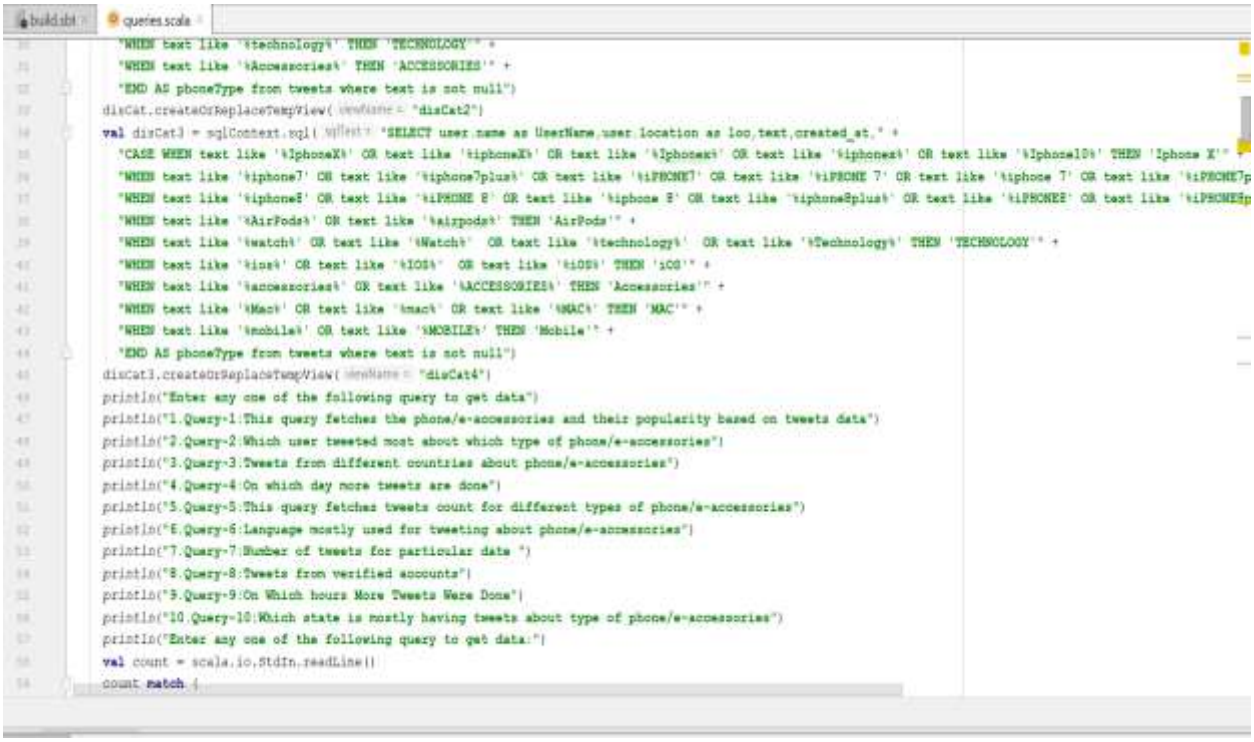
MapReduce Successful Output : map 100% reduce 100%

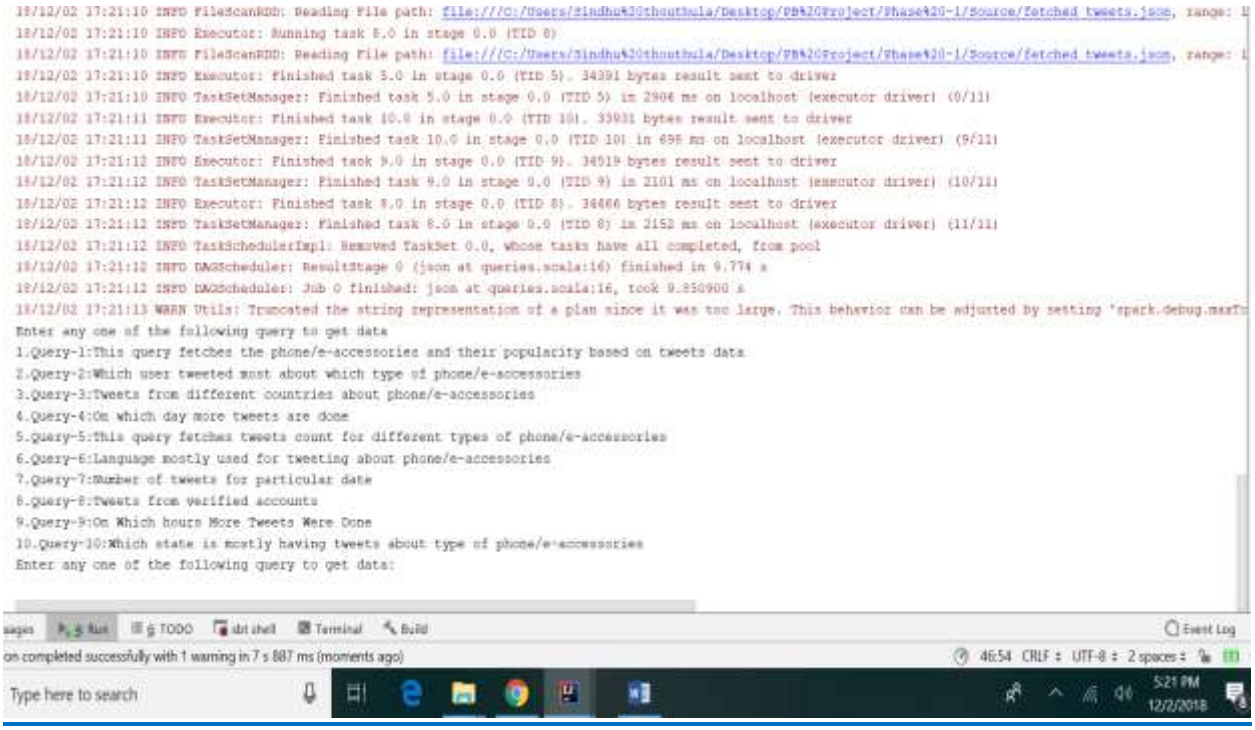


TWITTER ANALYSIS & VISUALIZATION OF PHONES/E-ACCESSORIES

ANALYTICAL QUERIES:

The following are the 10 queries on which we performed the visualizations.





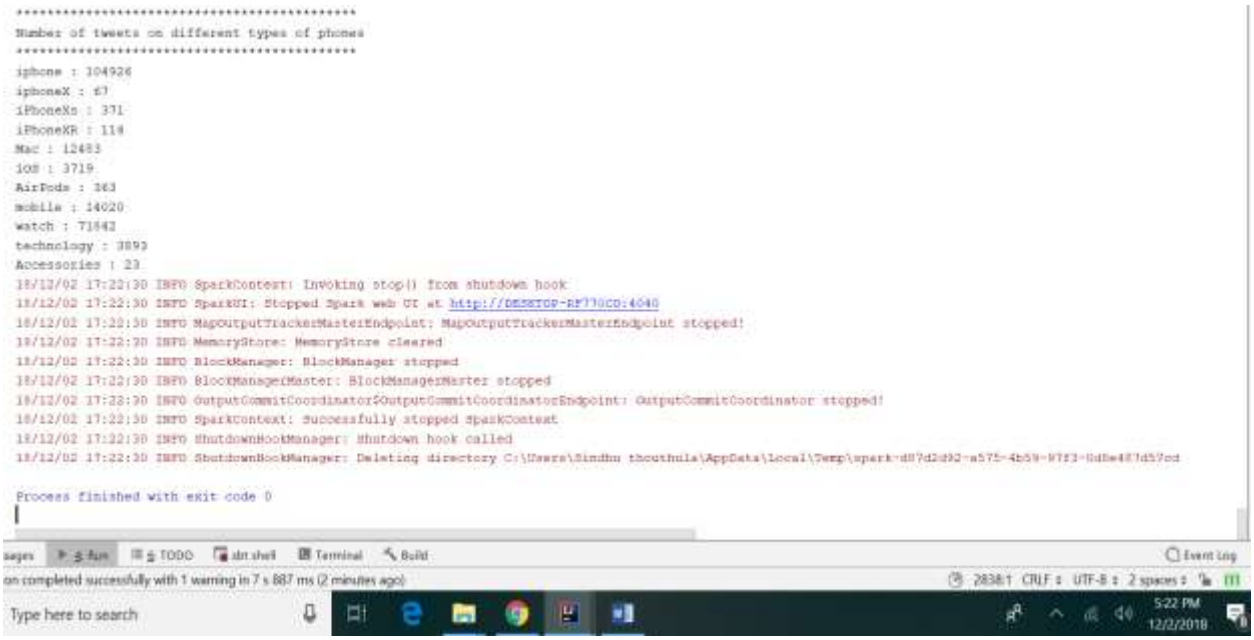
1.Query to fetch the tweets of phones/e-accessories based on the popularity.

This query is written to analyze the tweets that are made by the users on the particular key-words which would reflect the popularity of the phones/e-accessories. It would result the count i.e., how many times the phone/e-accessories would appear in the tweets made.

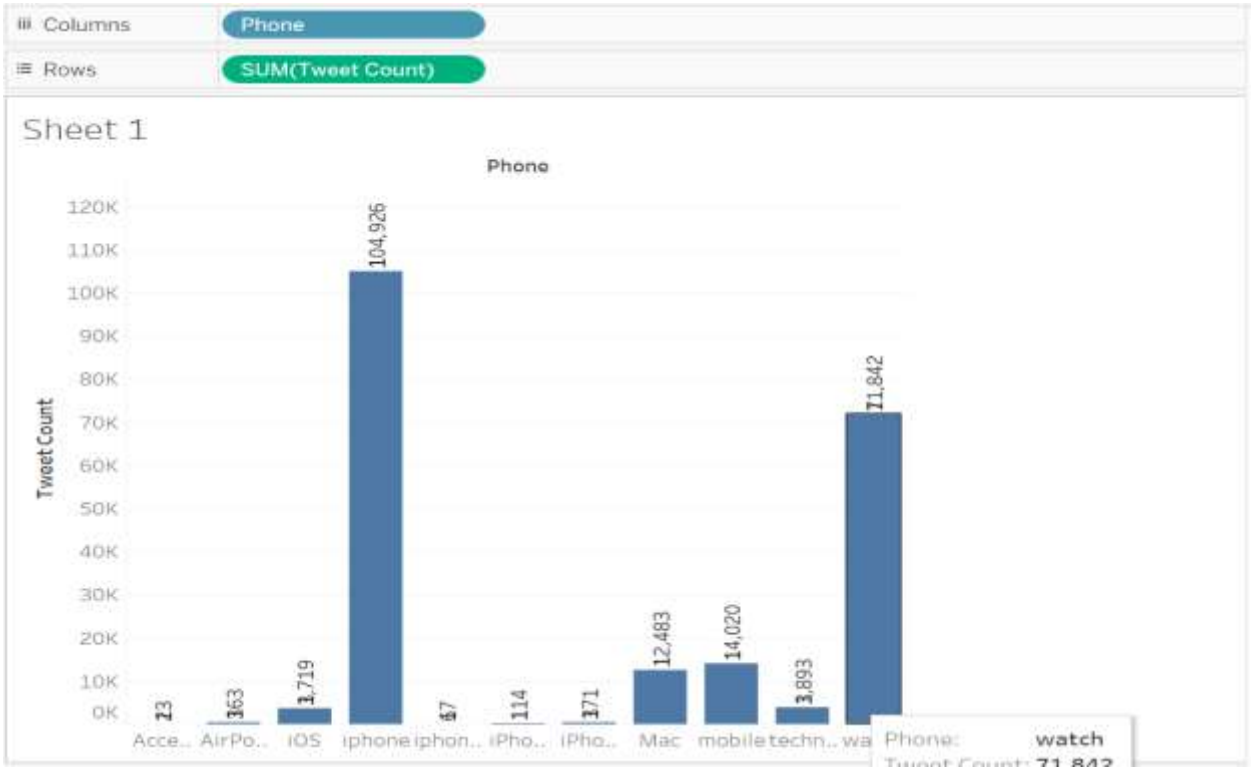
Query-Code :

```
/*----------------------Query 1: This query fetches the phones and its popularity based on tweets data-------------------------*/
val textFile = sc.textFile( path = "C:\\Users\\Sindhu thouthula\\Desktop\\PB Project\\Phase -1\\Source\\fetched_tweets.json")
val iphone = (textFile.filter(line => line.contains("iphone")).count())
val iphoneX = (textFile.filter(line => line.contains("iphoneX")).count())
val iPhoneXs = (textFile.filter(line => line.contains("iPhoneXs")).count())
val iPhoneXR = (textFile.filter(line => line.contains("iPhoneXR")).count())
val Mac = (textFile.filter(line => line.contains("Mac")).count())
val iOS = (textFile.filter(line => line.contains("iOS")).count())
val AirPods = (textFile.filter(line => line.contains("AirPods")).count())
val mobile = (textFile.filter(line => line.contains("mobile")).count())
val watch = (textFile.filter(line => line.contains("watch")).count())
val technology = (textFile.filter(line => line.contains("technology")).count())
val Accessories = (textFile.filter(line => line.contains("Accessories")).count())
println("*********************************************")
println("Number of tweets on different types of phones")
println("*********************************************")
println("iphone : %s".format(iphone))
println("iphoneX : %s".format(iphoneX))
println("iPhoneXs : %s".format(iPhoneXs))
println("iPhoneXR : %s".format(iPhoneXR))
println("Mac : %s".format(Mac))
println("iOS : %s".format(iOS))
println("AirPods : %s".format(AirPods))
println("mobile : %s".format(mobile))
println("watch : %s".format(watch))
println("technology : %s".format(technology))
println("Accessories : %s".format(Accessories))
```

Executed Query Output :



Executed Query Output Visualization :



2.Query for finding which user tweeted more about the type of phone/e-accessories.

This query is written to find the user that most tweeted about the phone/e-accessories so that it would result the count of how many times a user tweeted at most for each kind particularly.

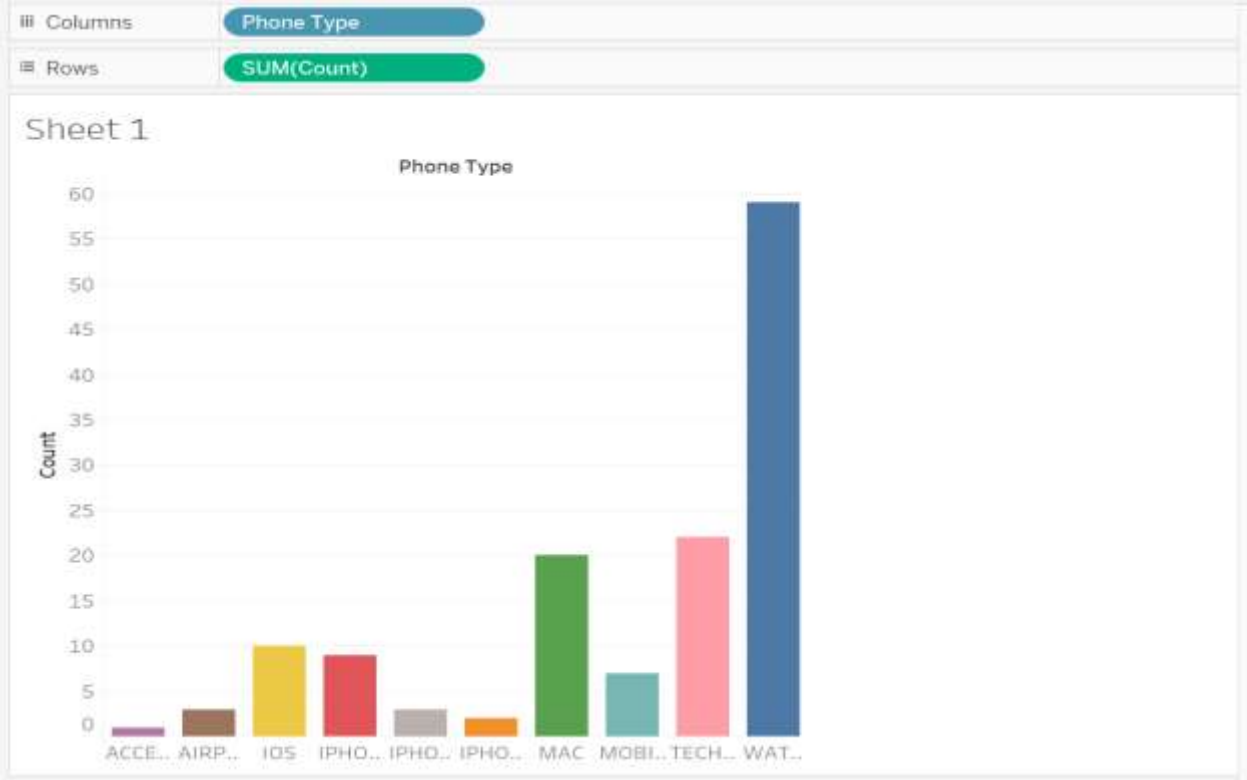
Query-Code :

```
val r1 = sqlContext.sql( sqlText = "SELECT UserName,'IPHONE' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='IPHONE' " +
  "group by UserName order by count desc limit 1")
val r2 = sqlContext.sql( sqlText = "SELECT UserName,'IPHONEX' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='IPHONEX' " +
  "group by UserName order by count desc limit 1 ")
val r3 = sqlContext.sql( sqlText = "SELECT UserName,'IPHONEXS' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='IPHONEXS' " +
  "group by UserName order by count desc limit 1 ")
val r4 = sqlContext.sql( sqlText = "SELECT UserName,'IPHONEXR' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='IPHONEXR' " +
  "group by UserName order by count desc limit 1 ")
val r5 = sqlContext.sql( sqlText = "SELECT UserName,'MAC' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='MAC' " +
  "group by UserName order by count desc limit 1 ")
val r6 = sqlContext.sql( sqlText = "SELECT UserName,'IOS' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='IOS' " +
  "group by UserName order by count desc limit 1 ")
val r7 = sqlContext.sql( sqlText = "SELECT UserName,'AIRPODS' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='AIRPODS' " +
  "group by UserName order by count desc limit 1 ")
val r8 = sqlContext.sql( sqlText = "SELECT UserName,'MOBILE' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='MOBILE' " +
  "group by UserName order by count desc limit 1 ")
val r9 = sqlContext.sql( sqlText = "SELECT UserName,'WATCH' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='WATCH' " +
  "group by UserName order by count desc limit 1")
val r10 = sqlContext.sql( sqlText = "SELECT UserName,'TECHNOLOGY' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='TECHNOLOGY' " +
  "group by UserName order by count desc limit 1")
val r11 = sqlContext.sql( sqlText = "SELECT UserName,'ACCESSORIES' as phoneType,count(*) as count FROM disCat2 WHERE phoneType='ACCESSORIES' " +
  "group by UserName order by count desc limit 1 ")

val rdd1 = r1.union(r2).union(r3).union(r4).union(r5).union(r6).union(r7).union(r8).union(r9) union (r10).union(r11)

println("****************************************")
println("Which user tweeted more on which type of phone")
println("****************************************")
rdd1.show()
```

Executed Query Output :

```
18/12/02 17:25:21 INFO DAGScheduler: Job 3 finished: show at queries.scala:119, took 0.240341 s
+--------------------+-----------+-----+
|            UserName|  phoneType|count|
+--------------------+-----------+-----+
|     Ast_By_Lamarcus|     IPHONE|    9|
|                 U 6|   IPHONEX8|    2|
|        Shane Salkey|   IPHONEXR|    3|
|    jenna montemayor|        MAC|   20|
|Muhammad Farhan B...|        IOS|   10|
|           Ryan Ngala|    AIRPODS|    3|
| Bharti Airtel India|     MOBILE|    7|
|       94.9 THE BULL|      WATCH|   59|
|              rwsana| TECHNOLOGY|   22|
|     Official_TFShop|ACCESSORIES|    1|
+--------------------+-----------+-----+

18/12/02 17:25:21 INFO SparkContext: Invoking stop() from shutdown hook
18/12/02 17:25:21 INFO SparkUI: Stopped Spark web UI at http://DESKTOP-SF770CO:4040
18/12/02 17:25:21 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/12/02 17:25:23 INFO MemoryStore: MemoryStore cleared
18/12/02 17:25:23 INFO BlockManager: BlockManager stopped
18/12/02 17:25:23 INFO BlockManagerMaster: BlockManagerMaster stopped
18/12/02 17:25:23 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
18/12/02 17:25:23 INFO SparkContext: Successfully stopped SparkContext
18/12/02 17:25:23 INFO ShutdownHookManager: Shutdown hook called
18/12/02 17:25:23 INFO ShutdownHookManager: Deleting directory C:\Users\Sindhu thouthula\AppData\Local\Temp\spark-f11bdba4-dc07-4da7-a022-6d8f3f5baf5f

Process finished with exit code 0
```

Executed Query Output Visualization :



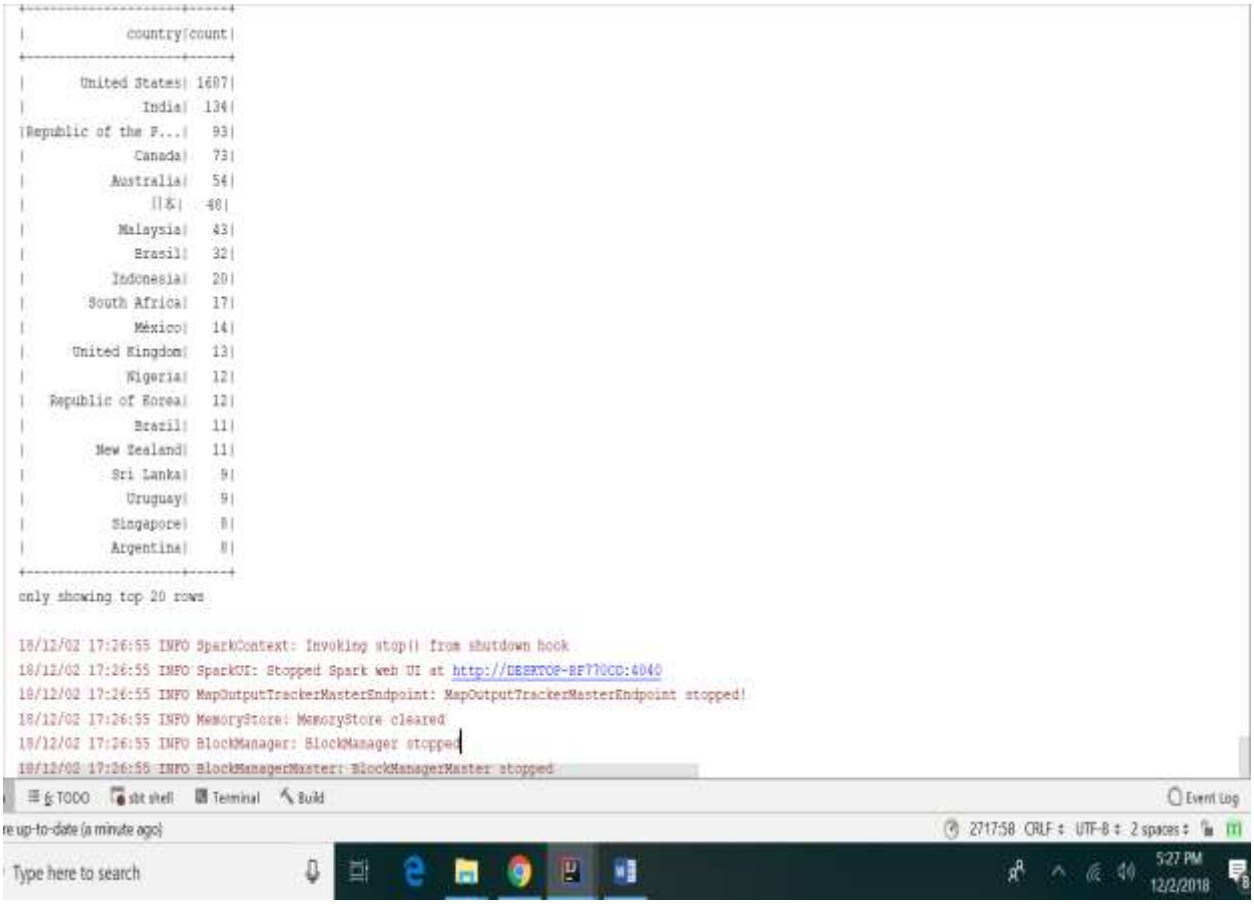
3.Query for fetching tweets from different countries.

This query is written to find the tweets based on the locations such that it would count how many tweets are posted about the phones/e-accessories from different countries.

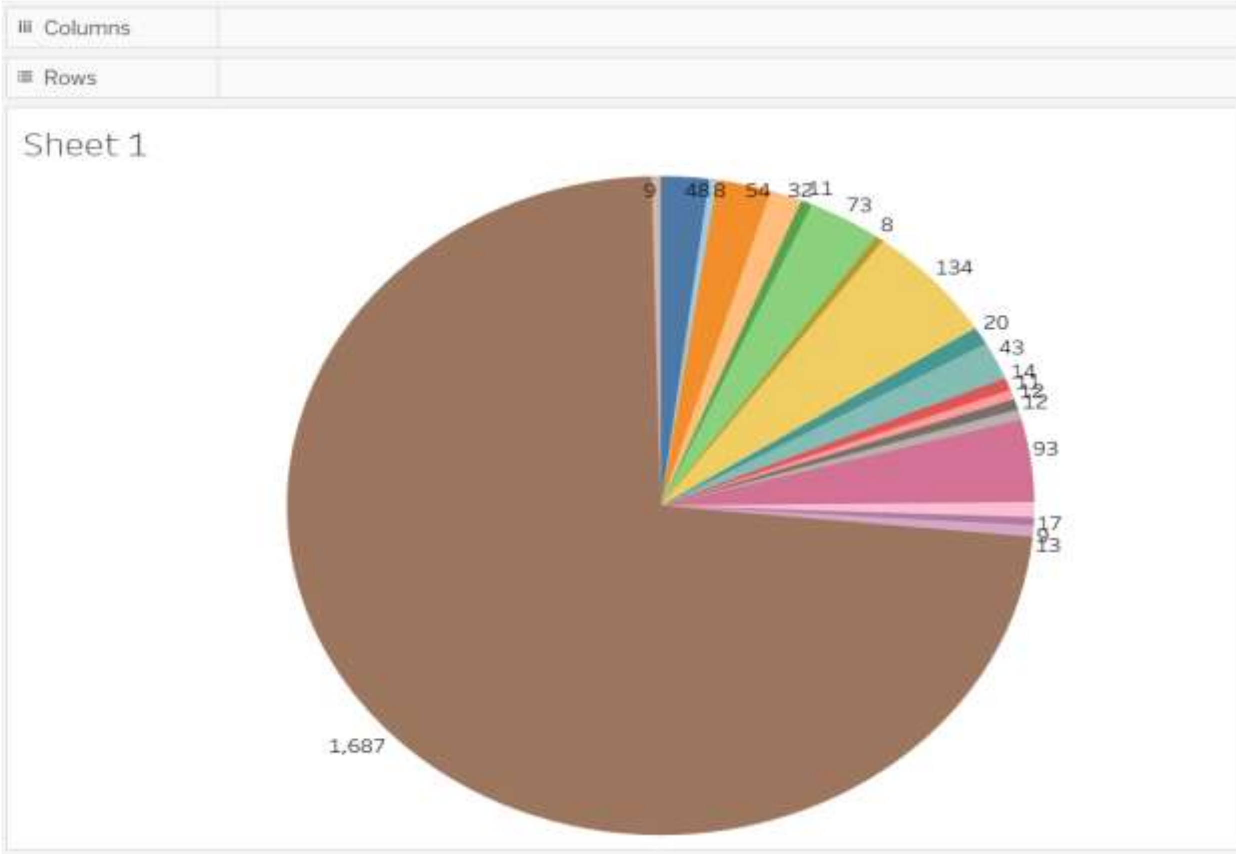
Query-Code :

```
/*----------------------------------Query 3: Tweets from different countries about phones ----------------------------------*/
case "3" =>
 val countrytweetscount = sqlContext.sql( sqlText = "SELECT distinct place.country, count(*) as count FROM tweets where place.country is not null " + "GROUP BY
 countrytweetscount.createOrReplaceTempView( viewName = "countrytweetscount")
 println("****************************************")
 println("Tweets from different countries")
 println("****************************************")
 countrytweetscount.show()
```

Executed Query Output :



Executed Query Output Visualization :



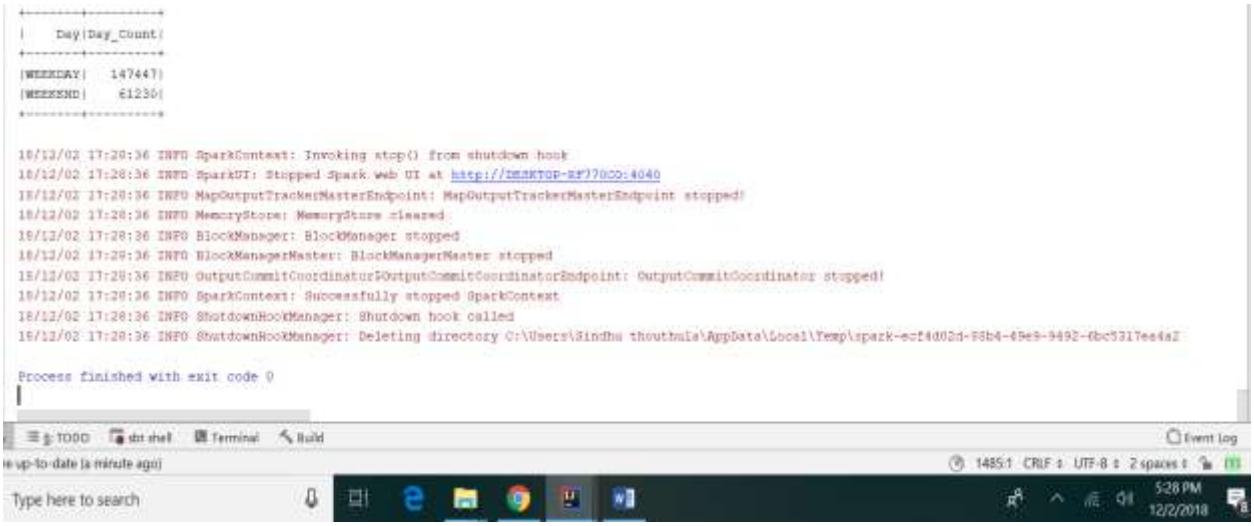
4.Query to fetch tweets to check on which day most of the tweets were made.

This query is written to find out the day on which more tweets were done so that it would count about giving us a figure.

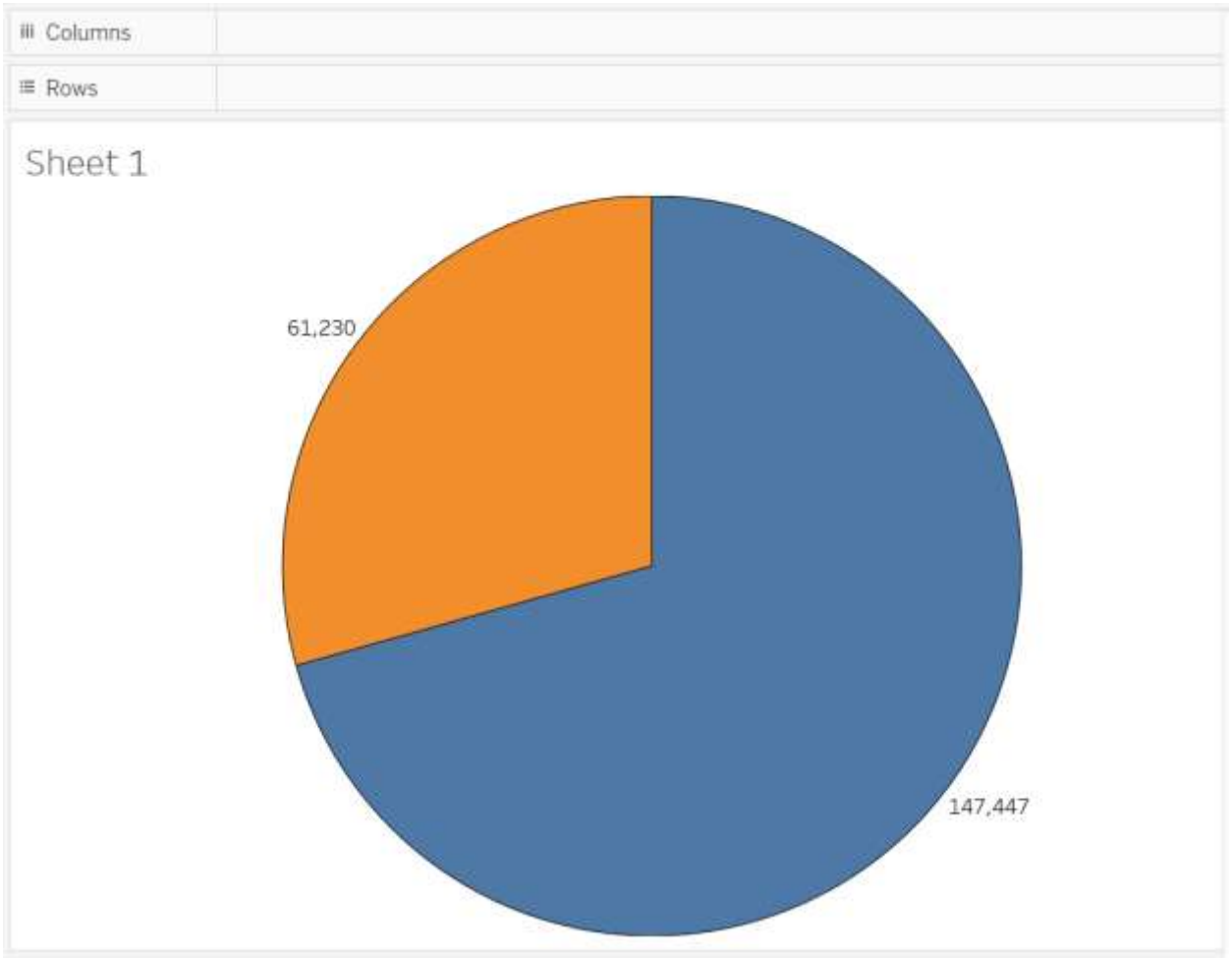
Query-Code :

```
/*------------------------------Query 4 : On which Day More Tweets are posted------------------------------*/
case "4" =>
  val day_data = sqlContext.sql( sqlText = "SELECT substring(user.created_at,1,3) as day from tweets where text is not null")

  day_data.createOrReplaceTempView( viewName = "day_data")

  val days_final = sqlContext.sql(
    """ SELECT Case
      |when day LIKE '%Mon%' then 'WEEKDAY'
      |when day LIKE '%Tue%' then 'WEEKDAY'
      |when day LIKE '%Wed%' then 'WEEKDAY'
      |when day LIKE '%Thu%' then 'WEEKDAY'
      |when day LIKE '%Fri%' then 'WEEKDAY'
      |when day LIKE '%Sat%' then 'WEEKEND'
      |when day LIKE '%Sun%' then 'WEEKEND'
      | else
      | null
      | end as day1 from day_data where day is not null""".stripMargin)

  days_final.createOrReplaceTempView( viewName = "days_final")

  val res = sqlContext.sql( sqlText = "SELECT day1 as Day,Count(*) as Day_Count from days_final where day1 is not null group by day1 order by count(*) desc")

  println("**********************************")
  println("On Which Day More Tweets Were Done")
  println("**********************************")
  res.show()
```

Executed Query Output :



Executed Query Output Visualization :



5.Query to fetch the tweets for the various series of the phone/e-accessories.

This query is written to extract the tweets that made on the different series of the phone/e-accessories so that it would count for it.

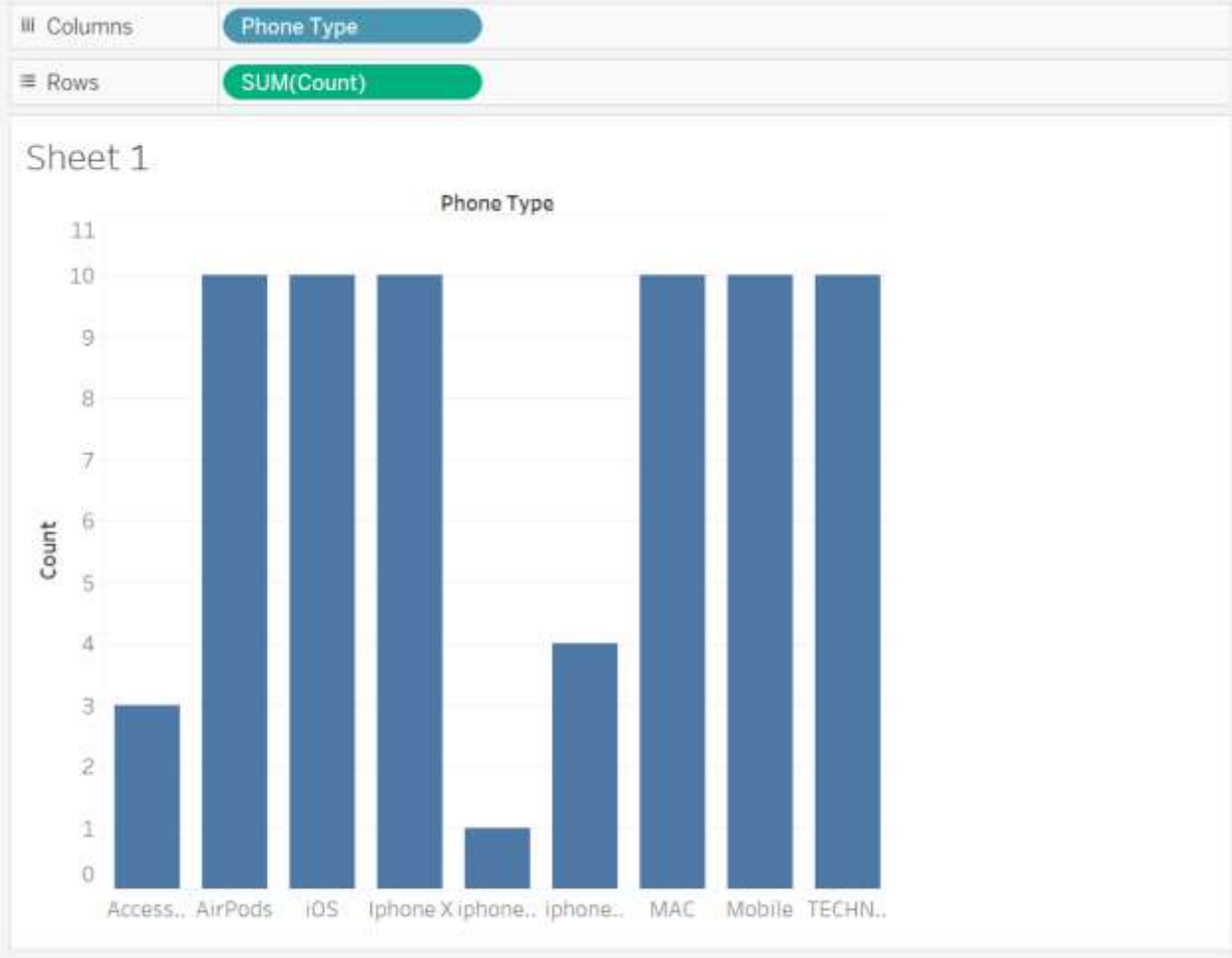
Query-Code :

```
/*-----------------------------------Query 5: Tweets count for different types of phone models ------------------------------------*/
case "5" =>
  val r1 = sqlContext.sql( sqlText = "SELECT loc,'Iphone X' as phoneType,count(*) as count FROM disCat4 WHERE phoneType='Iphone X' " +
    "group by loc order by count desc limit 10")
  val r2 = sqlContext.sql( sqlText = "SELECT loc,'iphone7 Series' as phoneType,count(*) as count FROM disCat4 WHERE phoneType='iphone7 Series' " +
    "group by loc order by count desc limit 10")
  val r3 = sqlContext.sql( sqlText = "SELECT loc,'iphone8 Series' as phoneType,count(*) as count FROM disCat4 WHERE phoneType='iphone8 Series' " +
    "group by loc order by count desc limit 10")
  val r4 = sqlContext.sql( sqlText = "SELECT loc,'AirPods' as phoneType,count(*) as count FROM disCat4 WHERE phoneType='AirPods' " +
    "group by loc order by count desc limit 10")
  val r5 = sqlContext.sql( sqlText = "SELECT loc,'TECHNOLOGY' as phoneType,count(*) as count FROM disCat4 WHERE phoneType='TECHNOLOGY' " +
    "group by loc order by count desc limit 10")
  val r6 = sqlContext.sql( sqlText = "SELECT loc,'iOS' as phoneType,count(*) as count FROM disCat4 WHERE phoneType='iOS' " +
    "group by loc order by count desc limit 10")
  val r7 = sqlContext.sql( sqlText = "SELECT loc,'Accessories' as phoneType,count(*) as count FROM disCat4 WHERE phoneType='Accessories' " +
    "group by loc order by count desc limit 10")
  val r8 = sqlContext.sql( sqlText = "SELECT loc,'MAC' as phoneType,count(*) as count FROM disCat4 WHERE phoneType='MAC' " +
    "group by loc order by count desc limit 10")
  val r9 = sqlContext.sql( sqlText = "SELECT loc,'Mobile' as phoneType,count(*) as count FROM disCat4 WHERE phoneType='Mobile' " +
    "group by loc order by count desc limit 10")
  val rdd1 = r1.union(r2).union(r3).union(r4).union(r5).union(r6).union(r7).union(r8).union(r9)
  rdd1.createOrReplaceTempView( viewName = "rdd1")
  val res = sqlContext.sql( sqlText = 'SELECT phoneType, Count(*) as Count from rdd1 where phoneType is not null group by phoneType")
  println("****************************************")
  println("Model Type")
  println("****************************************")
  res.show()
```

Executed Query Output :

```
+--------------+-----+
|     phoneType|Count|
+--------------+-----+
|           iOS|   10|
|      Iphone X|   10|
|       AirPods|   10|
|   Accessories|    3|
|           MAC|   10|
|    TECHNOLOGY|   10|
|        Mobile|   10|
|iphone8 Series|    4|
|iphone7 Series|    1|
+--------------+-----+

18/12/02 17:31:14 INFO SparkContext: Invoking stop() from shutdown hook
18/12/02 17:31:14 INFO SparkUI: Stopped Spark web UI at http://DESKTOP-RF77OCO:4040
18/12/02 17:31:14 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/12/02 17:31:16 INFO MemoryStore: MemoryStore cleared
18/12/02 17:31:16 INFO BlockManager: BlockManager stopped
18/12/02 17:31:16 INFO BlockManagerMaster: BlockManagerMaster stopped
18/12/02 17:31:16 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
18/12/02 17:31:16 INFO SparkContext: Successfully stopped SparkContext
18/12/02 17:31:16 INFO ShutdownHookManager: Shutdown hook called
18/12/02 17:31:16 INFO ShutdownHookManager: Deleting directory C:\Users\Sindhu thouthula\AppData\Local\Temp\spark-91623296-27d9-4328-a6f7-e0901875befe

Process finished with exit code 0
```

Executed Query Output Visualization :



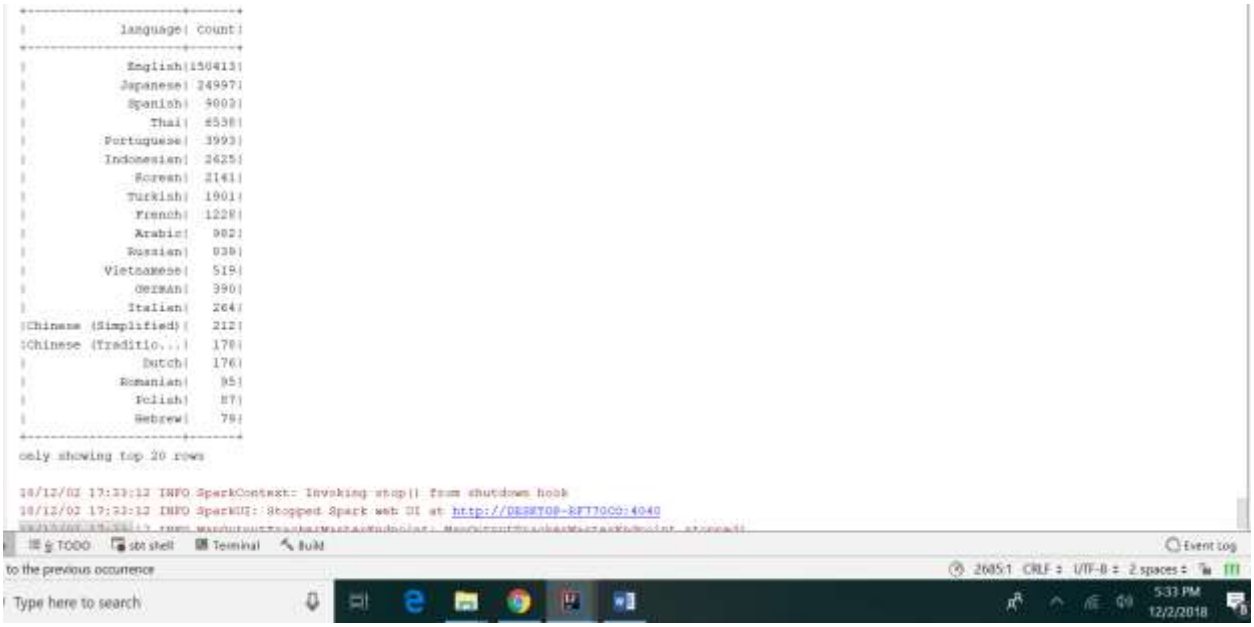
6.Query to fetch the languages mostly used for tweeting about the phone/e-accessories.

This query is written to analyze the language mostly used by the users so that it would count how many times the users tweeted about the phone/e-accessories in a particular language.

Query-Code :

```
/*-----------------------------------Query 6 Popular languages used for tweeting tweets about phones ----------------------------------*/
case "6" =>
  val langWstCount = sqlContext.sql( sqlText = "SELECT distinct id," +
    "CASE when user.lang LIKE '%en%' then 'English'" +
    "when user.lang LIKE '%ja%' then 'Japanese'" +
    "when user.lang LIKE '%es%' then 'Spanish'" +
    "when user.lang LIKE '%fr%' then 'French'" +
    "when user.lang LIKE '%it%' then 'Italian'" +
    "when user.lang LIKE '%ru%' then 'Russian'" +
    "when user.lang LIKE '%ar%' then 'Arabic'" +
    "when user.lang LIKE '%bn%' then 'Bengali'" +
    "when user.lang LIKE '%cs%' then 'Czech'" +
    "when user.lang LIKE '%da%' then 'Danish'" +
```

Executed Query Output :



language	count
English	150413
Japanese	24997
Spanish	9003
Thai	6538
Portuguese	3993
Indonesian	2625
Korean	2141
Turkish	1901
French	1228
Arabic	902
Russian	839
Vietnamese	519
German	390
Italian	264
Chinese (Simplified)	212
Chinese (Traditio...)	178
Dutch	176
Romanian	95
Polish	87
Hebrew	78

only showing top 20 rows

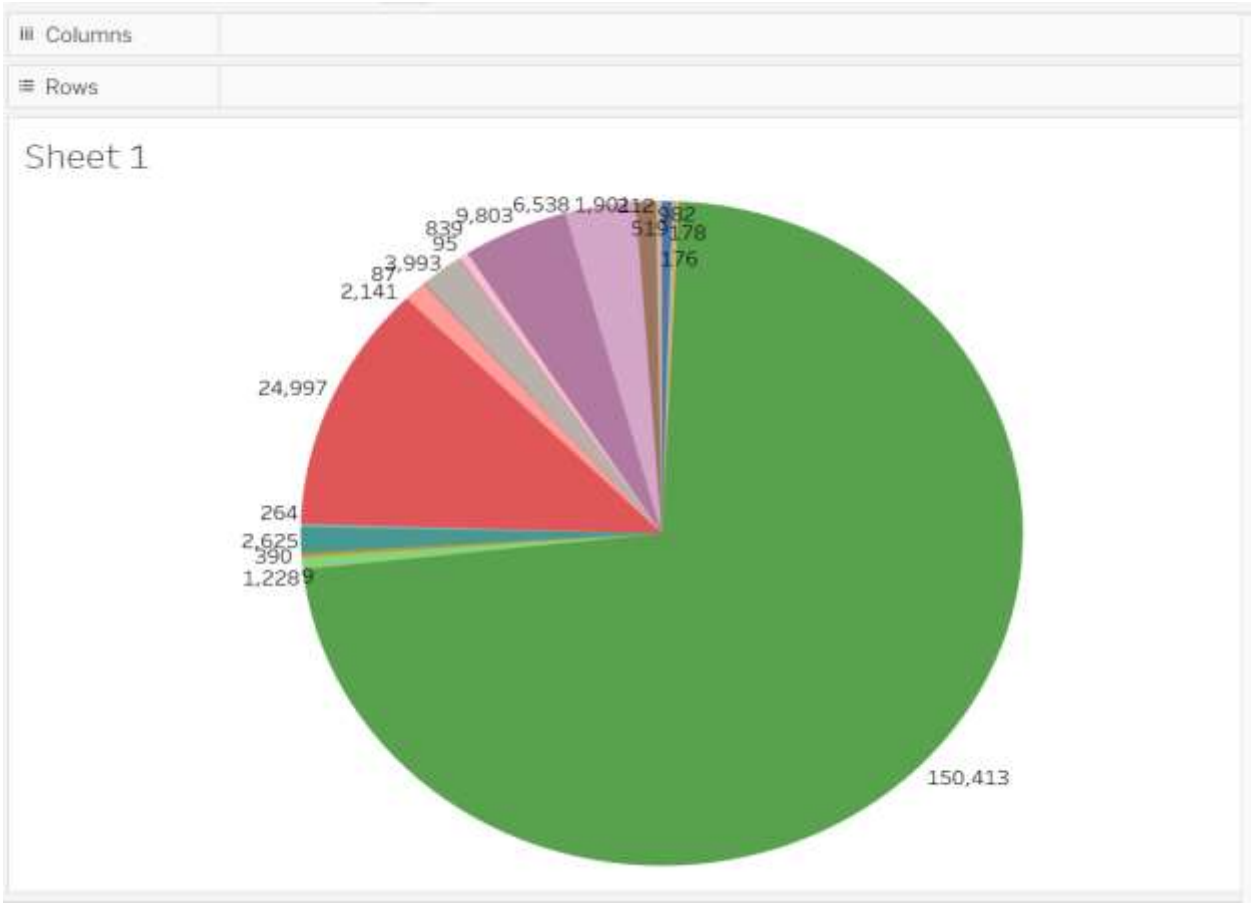
18/12/02 17:33:12 INFO SparkContext: Invoking stop() from shutdown hook
18/12/02 17:33:12 INFO SparkUI: Stopped Spark web UI at http://DESKTOP-RFT7OCO:4040

TODO | sbt shell | Terminal | Build | Event Log

to the previous occurrence | 2685:1 | CRLF | UTF-8 | 2 spaces

Type here to search | 5:33 PM 12/2/2018

Executed Query Output Visualization :



7.Query for fetching the count of the tweets made on a particular day.

This query is written to analyze, depending upon the collected data for each day how many tweets are made.

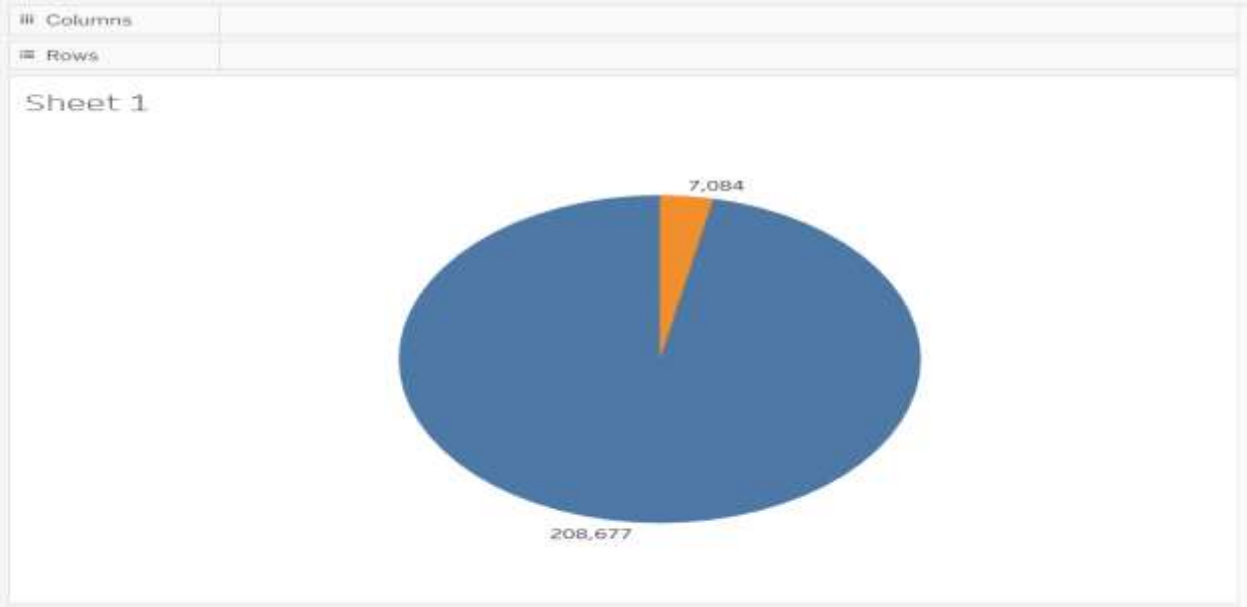
Query-Code :

```
/*-------------------------------Query 7 number of tweets for particular date ------------------------------*/
case "7" =>
  val tweetcount = sqlContext.sql( sqlText = "SELECT SUBSTR(created_at, 0, 10) tweet_date, COUNT(1) tweet_count FROM  tweets GROUP  BY SUBSTR(created_at, 0,
  tweetcount.createOrReplaceTempView( viewName = "tweetcount")
  println("****************************************")
  println("Tweet Count")
  println("****************************************")
  tweetcount.show()
```

Executed Query Output :

```
+----------+-----------+
|tweet_date|tweet_count|
+----------+-----------+
|Sun Sep 23|     208677|
|      null|       7084|
+----------+-----------+

18/12/02 17:34:31 INFO SparkContext: Invoking stop() from shutdown hook
18/12/02 17:34:31 INFO SparkUI: Stopped Spark web UI at http://3MGBTOP-KF770CU:4040
18/12/02 17:34:31 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/12/02 17:34:32 INFO MemoryStore: MemoryStore cleared
18/12/02 17:34:32 INFO BlockManager: BlockManager stopped
18/12/02 17:34:32 INFO BlockManagerMaster: BlockManagerMaster stopped
18/12/02 17:34:32 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
18/12/02 17:34:32 INFO SparkContext: Successfully stopped SparkContext
18/12/02 17:34:32 INFO ShutdownHookManager: Shutdown hook called
18/12/02 17:34:32 INFO ShutdownHookManager: Deleting directory C:\Users\Sindhu thouthula\AppData\Local\Temp\spark-1455ee88-72e5-4944-8706-4e7a136bf1c

Process finished with exit code 0
```

Executed Query Output Visualization :

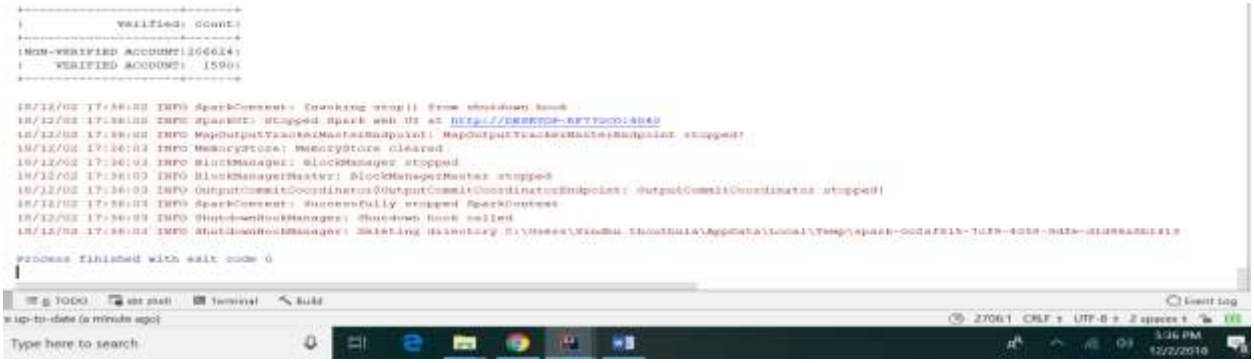


8.Query for fetching the tweets made from verified accounts.

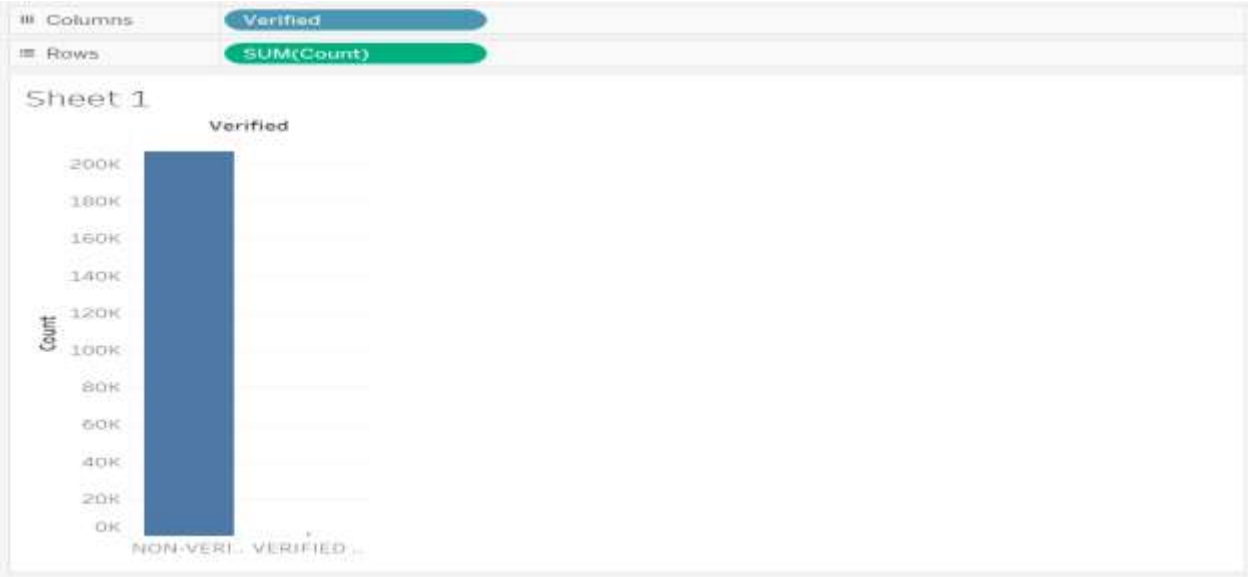
This query is written to make an analysis on the number of verified users. This query counts for the tweets made from the verified accounts only and gives us the value.

Query-Code :

Executed Query Output :



Executed Query Output Visualization :



9.Query for fetching the tweets based on the hours on which most of them were made.

This query is written to analyze on which hours tweets are made like mornings, afternoon, evenings etc.

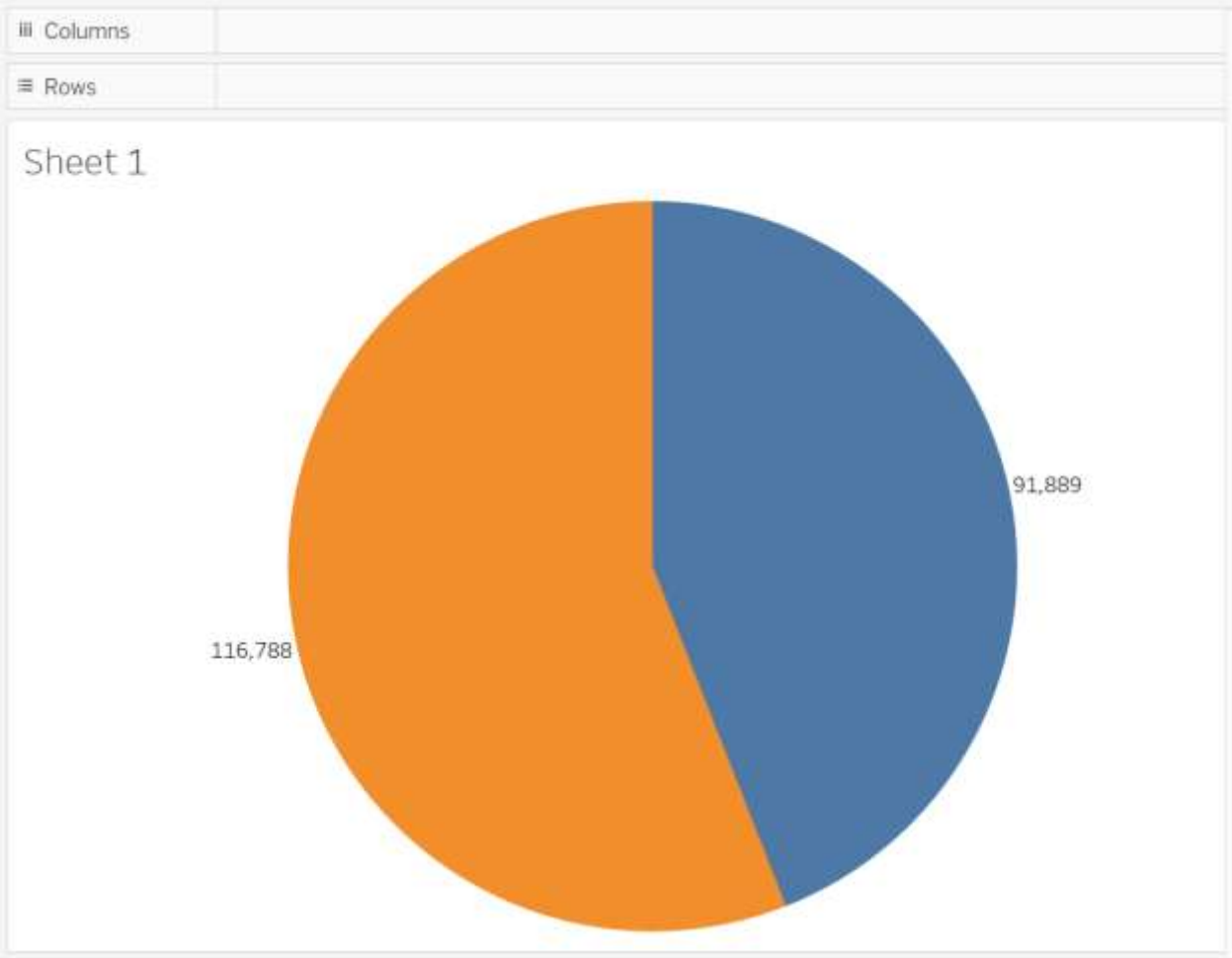
Query-Code :

```
/*----------------------------------Query 9 On Which hours More Tweets Were Done ----------------------------------*/
case "9" =>
  val timehour = sqlContext.sql( sqlText = "SELECT SUBSTRING(created_at,12,2) as hour from tweets where text is not null")

  timehour.createOrReplaceTempView( viewName = "timehour")

  val timeAnalysis = sqlContext.sql(
    """ SELECT Case
      |when hour>=0 and hour <4 then 'midnight'
      |when hour>=4 and hour <7 then 'early Morning'
      |when hour>=7 and hour <12 then 'Morning'
      |when hour>=12 and hour <15 then 'afternoon'
      |when hour>=15 and hour <18 then 'evening'
      |when hour>=18 and hour <=23 then 'night'
      |end as time from timehour""".stripMargin)

  timeAnalysis.createOrReplaceTempView( viewName = "timeAnalysis")

  val res = sqlContext.sql( sqlText = "SELECT time as hour,Count(*) as tweets_count from timeAnalysis where time is not null group by time order by count(*)

  println("**********************************")
  println("On Which hours More Tweets Were Done")
  println("**********************************")
```

Executed Query Output :

```
+-------------+------------+
|         hour|tweets_count|
+-------------+------------+
|     midnight|      116788|
|early Morning|       91689|
+-------------+------------+

18/12/02 17:37:10 INFO BlockManagerInfo: Removed broadcast_4_piece0 on DESKTOP-RF77OCO:55163 in memory (size: 15.5 KB, free: 892.2 MB)
18/12/02 17:37:10 INFO SparkContext: Invoking stop() from shutdown hook
18/12/02 17:37:10 INFO SparkUI: Stopped Spark web UI at http://DESKTOP-RF77OCO:4040
18/12/02 17:37:10 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/12/02 17:37:10 INFO MemoryStore: MemoryStore cleared
18/12/02 17:37:10 INFO BlockManager: BlockManager stopped
18/12/02 17:37:10 INFO BlockManagerMaster: BlockManagerMaster stopped
18/12/02 17:37:10 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
18/12/02 17:37:10 INFO SparkContext: Successfully stopped SparkContext
18/12/02 17:37:10 INFO ShutdownHookManager: Shutdown hook called
18/12/02 17:37:10 INFO ShutdownHookManager: Deleting directory C:\Users\Sindhu thouthula\AppData\Local\Temp\spark-212bdae2-d417-4685-81e9-67daa806a02e

Process finished with exit code 0
```

Executed Query Output Visualization :

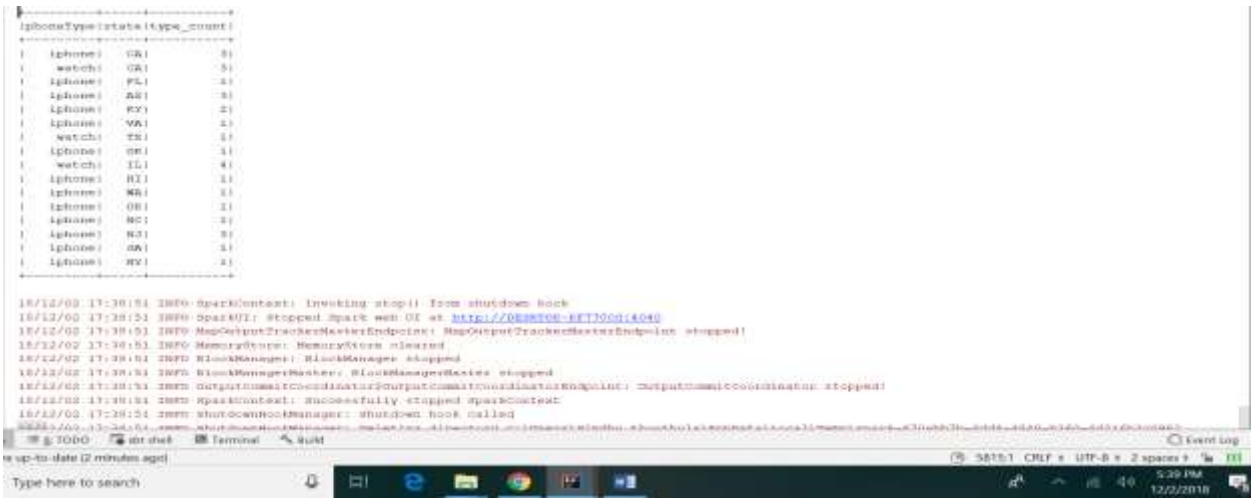


10.Query for fetching tweets based on which state has more tweets about particular type of phone/e-accessories.

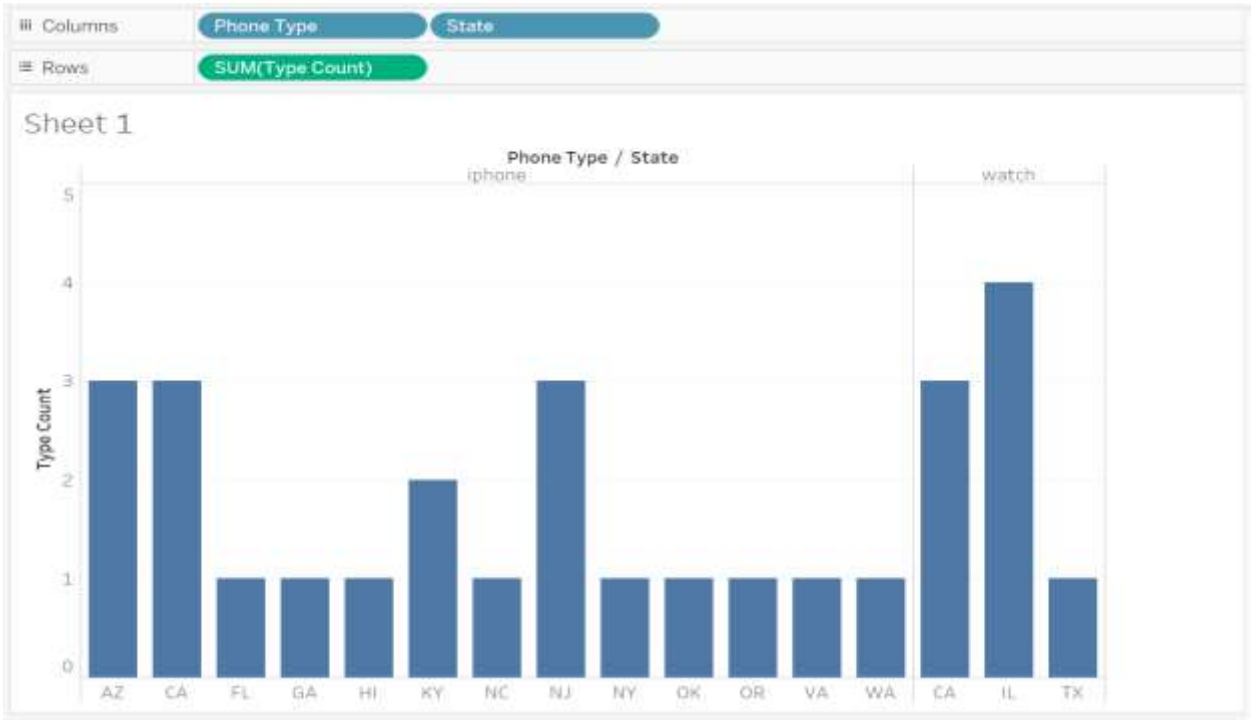
This query is written to check which state users made most number of tweets about a type of phone/e-accessories.

Query-Code :

Executed Query Output :



Executed Query Output Visualization :



CODE TESTING:

We can test the code by opening the Project folder using IntelliJ IDEA and can run the code and see that the queries are being executed, later we can see the visualizations can be opened in Tableau and we can click on each of them to see the visualizations of ten queries.

Individual task: 3 queries each with respective visualization

Collection of data and Word Count (together)

Work to be Completed: Sentiment Analysis.