Principles of Big Data Management

CSEE5540

Phase 2 Report

Team :

Chandrasekhar Pentakota (16267781)

Kartheek Katta (16273764)

Jaya Prakash Ravella (16271007)

Links :

https://github.com/chanduhub/TwitterProject_CS5540

Goal:

* To store tweets in Spark SQL and run queries on the data to analyse some interesting things.

* Make use of visualization tools to understand data better.

Environment Setup:

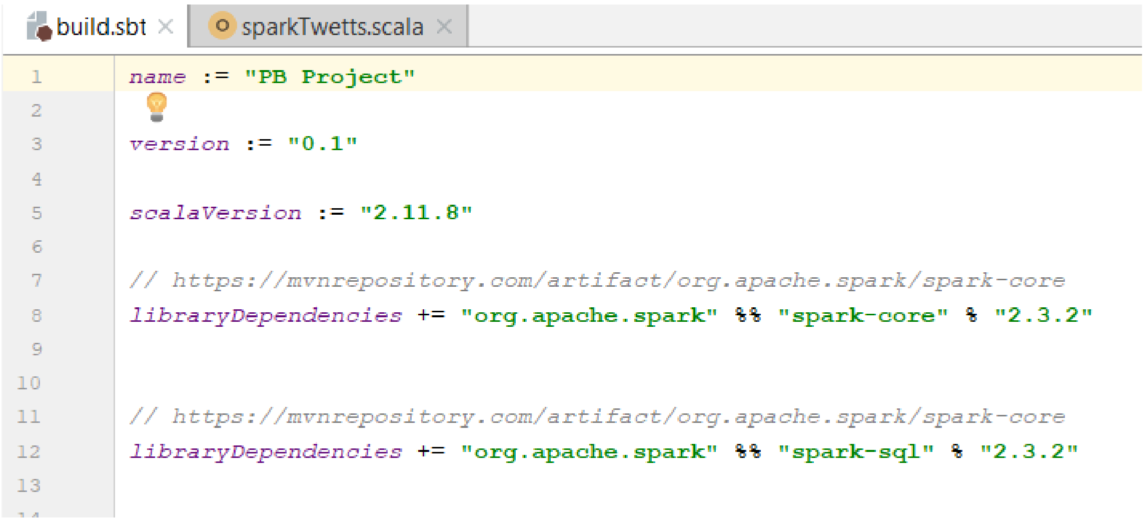
We have installed spark in windows. The following steps are used

1.Download spark,unzip and set SPARK_HOME.

2.Download winutils jar and set it as HADOOP_HOME in environmental variables.

3.Install IntelliJ, the build.sbt file in intelliJ downloads all the required libraries for the project.

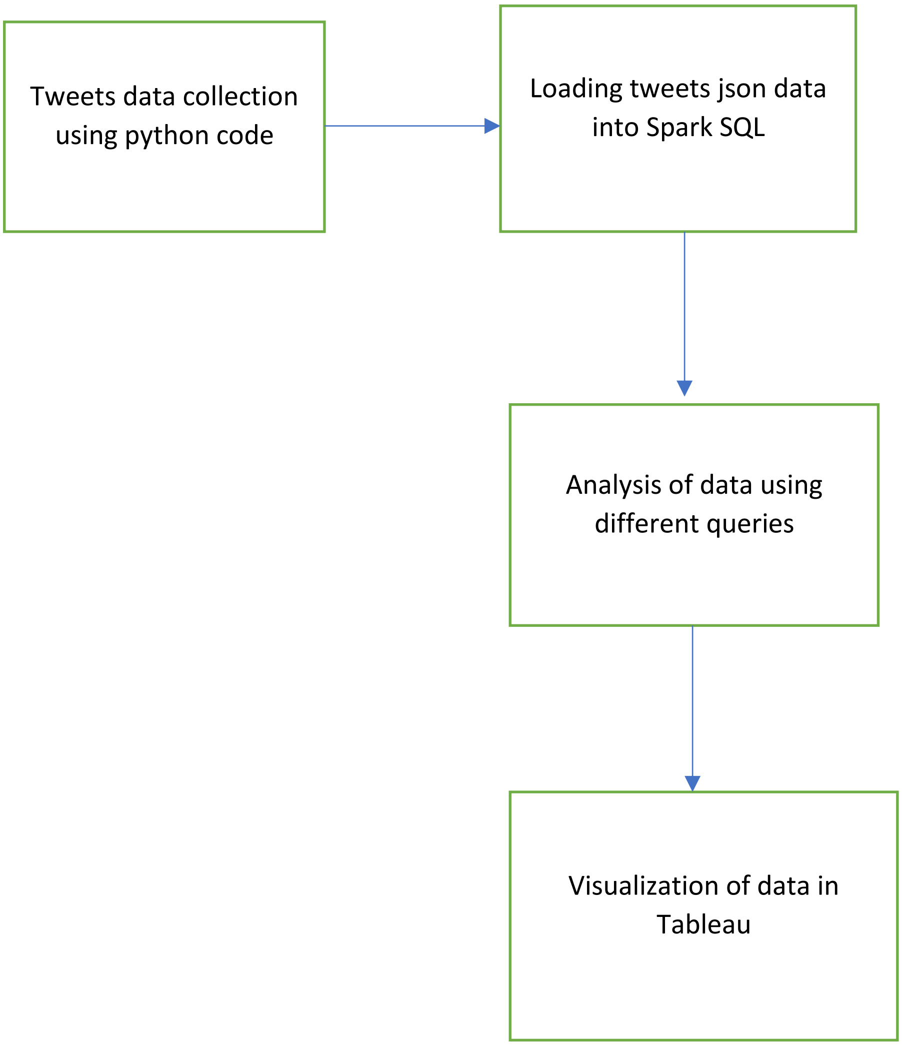
4.Install Tableau Public software for visualization.



```
name := "PB Project"

version := "0.1"

scalaVersion := "2.11.8"

// https://mvnrepository.com/artifact/org.apache.spark/spark-core
libraryDependencies += "org.apache.spark" %% "spark-core" % "2.3.2"


// https://mvnrepository.com/artifact/org.apache.spark/spark-core
libraryDependencies += "org.apache.spark" %% "spark-sql" % "2.3.2"

```

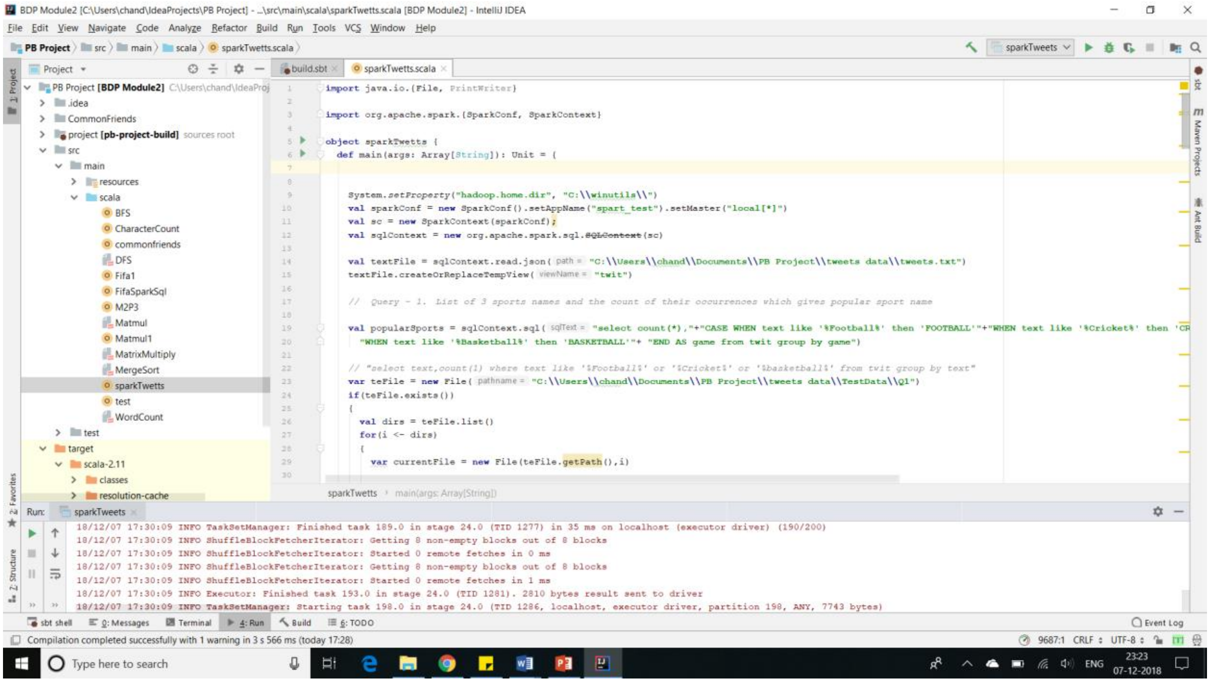
Design:

In phase1, we have collected twitter data by using the code which is of size 650 MB. The total tweets collected are 100K+. In this phase, we have done analysis on this data using spark SQL.


Tweets data collection using python code
Loading tweets json data into Spark SQL
Analysis of data using different queries
Visualization of data in Tableau

Implementation:

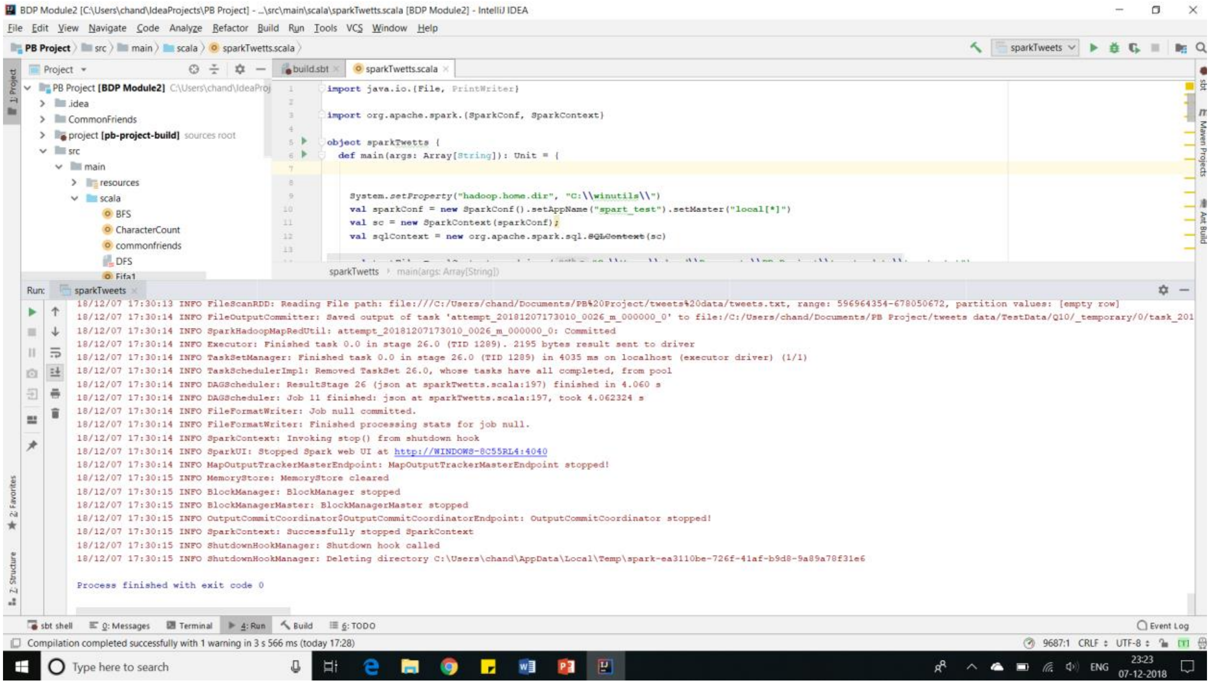
The main code is written in scala spark and the queries are written in Spark SQL. Output data generated from this code is fed into the Tableau to get insights from that data.



```
import java.io.{File, PrintWriter}

import org.apache.spark.{SparkConf, SparkContext}

object sparkTwetts {
  def main(args: Array[String]): Unit = {


    System.setProperty("hadoop.home.dir", "C:\\winutils\\")
    val sparkConf = new SparkConf().setAppName("spart_test").setMaster("local[*]")
    val sc = new SparkContext(sparkConf);
    val sqlContext = new org.apache.spark.sql.~~SQLContext~~(sc)

    val textFile = sqlContext.read.json("C:\\Users\\chand\\Documents\\PB Project\\tweets data\\tweets.txt")
    textFile.createOrReplaceTempView("twit")

    //  Query - 1. List of 3 sports names and the count of their occurrences which gives popular sport name

    val popularSports = sqlContext.sql("select count(*),"+"CASE WHEN text like '%Football%' then 'FOOTBALL'"+"WHEN text like '%Cricket%' then 'CR
      "WHEN text like '%Basketball%' then 'BASKETBALL'"+ "END AS game from twit group by game")

    // "select text,count(1) where text like '%Football%' or '%Cricket%' or '%basketball%' from twit group by text"
    var teFile = new File("C:\\Users\\chand\\Documents\\PB Project\\tweets data\\TestData\\Q1")
    if(teFile.exists())
    {
      val dirs = teFile.list()
      for(i <- dirs)
      {
        var currentFile = new File(teFile.getPath(),i)
```

```
18/12/07 17:30:09 INFO TaskSetManager: Finished task 189.0 in stage 24.0 (TID 1277) in 35 ms on localhost (executor driver) (190/200)
18/12/07 17:30:09 INFO ShuffleBlockFetcherIterator: Getting 8 non-empty blocks out of 8 blocks
18/12/07 17:30:09 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
18/12/07 17:30:09 INFO ShuffleBlockFetcherIterator: Getting 8 non-empty blocks out of 8 blocks
18/12/07 17:30:09 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 1 ms
18/12/07 17:30:09 INFO Executor: Finished task 193.0 in stage 24.0 (TID 1281). 2810 bytes result sent to driver
18/12/07 17:30:09 INFO TaskSetManager: Starting task 198.0 in stage 24.0 (TID 1286, localhost, executor driver, partition 198, ANY, 7743 bytes)
```

Compilation completed successfully with 1 warning in 3 s 566 ms (today 17:28)



```
import java.io.{File, PrintWriter}

import org.apache.spark.{SparkConf, SparkContext}

object sparkTwetts {
  def main(args: Array[String]): Unit = {


    System.setProperty("hadoop.home.dir", "C:\\winutils\\")
    val sparkConf = new SparkConf().setAppName("spart_test").setMaster("local[*]")
    val sc = new SparkContext(sparkConf);
    val sqlContext = new org.apache.spark.sql.~~SQLContext~~(sc)
```

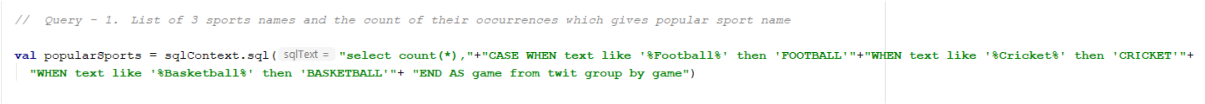
```
18/12/07 17:30:13 INFO FileScanRDD: Reading File path: file:///C:/Users/chand/Documents/PB%20Project/tweets%20data/tweets.txt, range: 596964354-678050672, partition values: [empty row]
18/12/07 17:30:14 INFO FileOutputCommitter: Saved output of task 'attempt_20181207173010_0026_m_000000_0' to file:/C:/Users/chand/Documents/PB Project/tweets data/TestData/Q10/_temporary/0/task_201
18/12/07 17:30:14 INFO SparkHadoopMapRedUtil: attempt_20181207173010_0026_m_000000_0: Committed
18/12/07 17:30:14 INFO Executor: Finished task 0.0 in stage 26.0 (TID 1289). 2195 bytes result sent to driver
18/12/07 17:30:14 INFO TaskSetManager: Finished task 0.0 in stage 26.0 (TID 1289) in 4035 ms on localhost (executor driver) (1/1)
18/12/07 17:30:14 INFO TaskSchedulerImpl: Removed TaskSet 26.0, whose tasks have all completed, from pool
18/12/07 17:30:14 INFO DAGScheduler: ResultStage 26 (json at sparkTwetts.scala:197) finished in 4.060 s
18/12/07 17:30:14 INFO DAGScheduler: Job 11 finished: json at sparkTwetts.scala:197, took 4.062324 s
18/12/07 17:30:14 INFO FileFormatWriter: Job null committed.
18/12/07 17:30:14 INFO FileFormatWriter: Finished processing stats for job null.
18/12/07 17:30:14 INFO SparkContext: Invoking stop() from shutdown hook
18/12/07 17:30:14 INFO SparkUI: Stopped Spark web UI at http://WINDOWS-8C55RL4:4040
18/12/07 17:30:14 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/12/07 17:30:15 INFO MemoryStore: MemoryStore cleared
18/12/07 17:30:15 INFO BlockManager: BlockManager stopped
18/12/07 17:30:15 INFO BlockManagerMaster: BlockManagerMaster stopped
18/12/07 17:30:15 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
18/12/07 17:30:15 INFO SparkContext: Successfully stopped SparkContext
18/12/07 17:30:15 INFO ShutdownHookManager: Shutdown hook called
18/12/07 17:30:15 INFO ShutdownHookManager: Deleting directory C:\Users\chand\AppData\Local\Temp\spark-ea3110be-726f-41af-b9d8-9a89a78f31e6

Process finished with exit code 0
```

Compilation completed successfully with 1 warning in 3 s 566 ms (today 17:28)

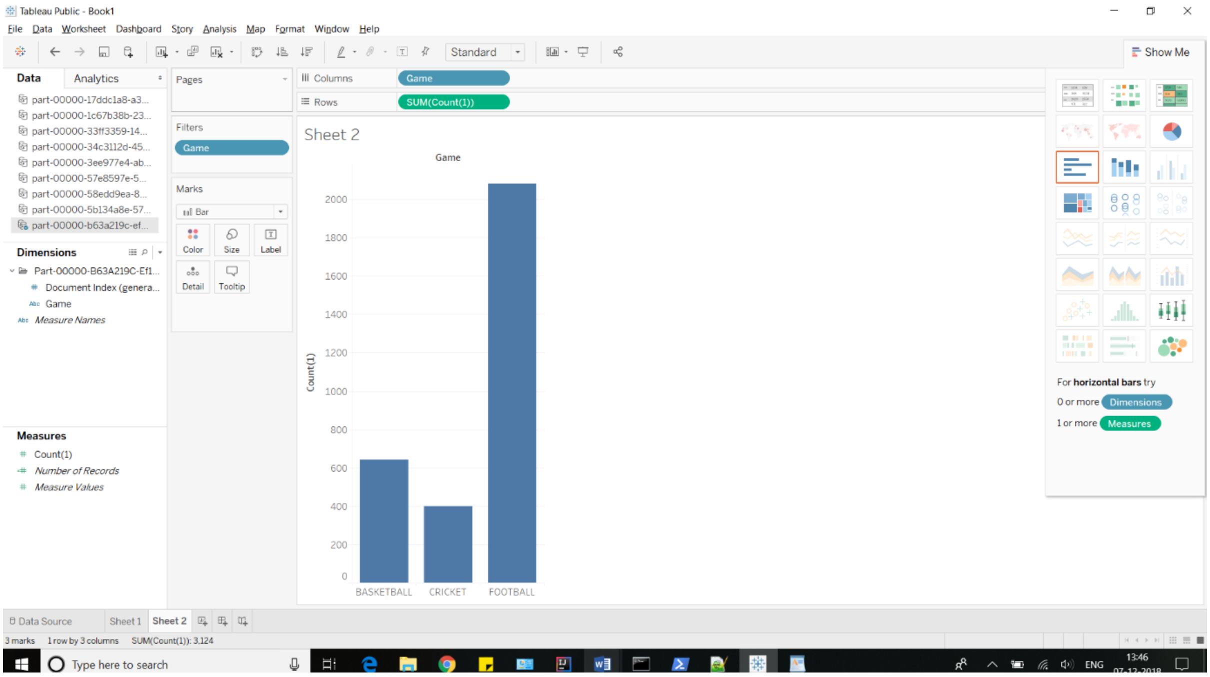
Process finished with the success code.

Queries and output :

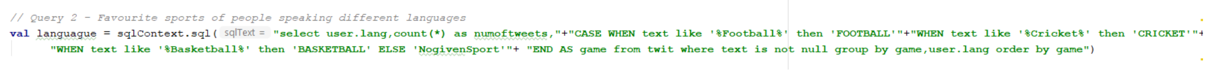
Query1 : List of 3 sports names and the count of their occurrences which gives popular sport name among them



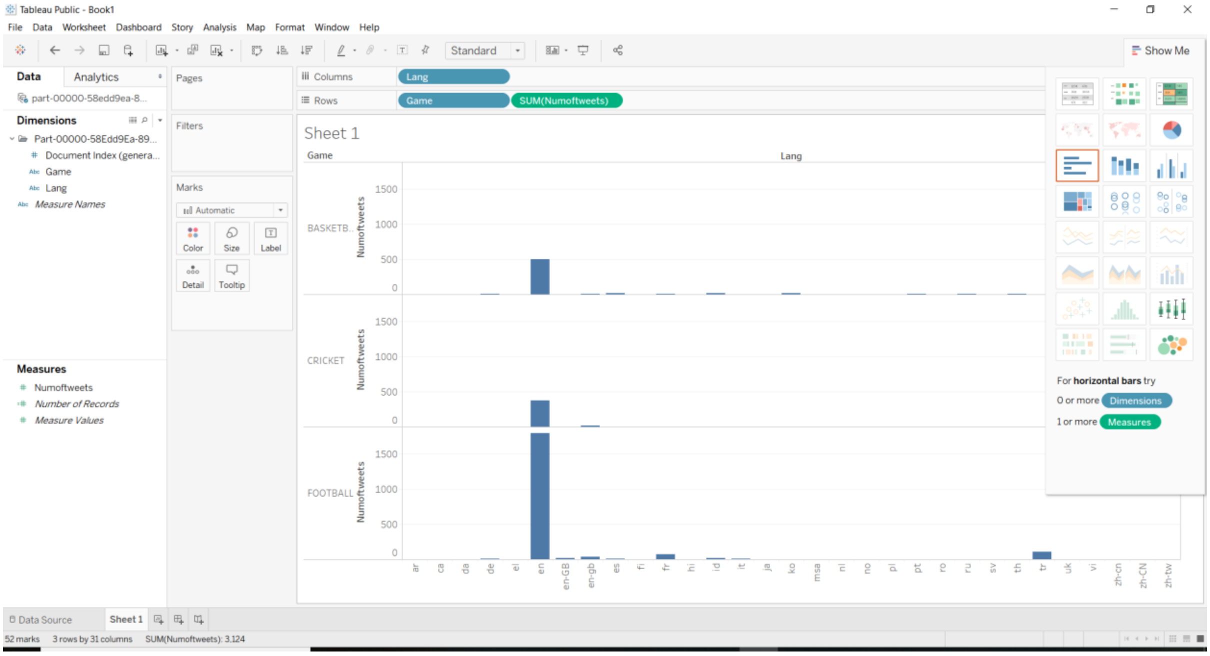
```
//  Query - 1.  List of 3 sports names and the count of their occurrences which gives popular sport name

val popularSports = sqlContext.sql( sqlText = "select count(*),"+"CASE WHEN text like '%Football%' then 'FOOTBALL'"+"WHEN text like '%Cricket%' then 'CRICKET'"+
  "WHEN text like '%Basketball%' then 'BASKETBALL'"+ "END AS game from twit group by game")
```



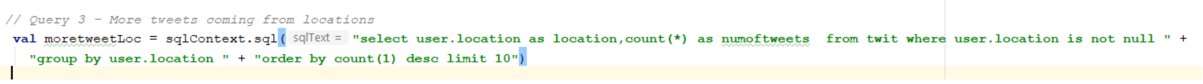
Query 2: Favourite sports of people speaking different languages



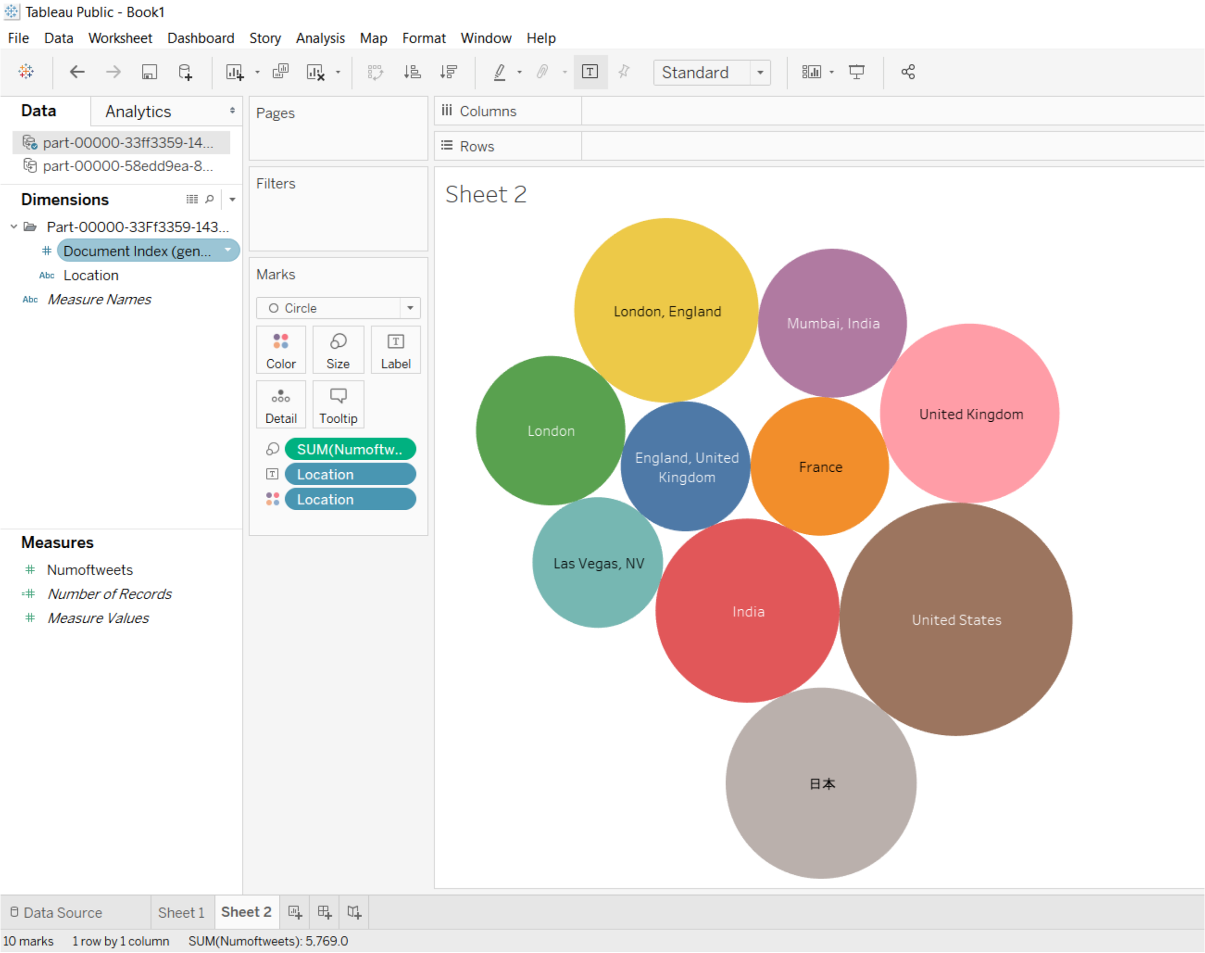
```
// Query 2 - Favourite sports of people speaking different languages
val languague = sqlContext.sql( sqlText = "select user.lang,count(*) as numoftweets,"+"CASE WHEN text like '%Football%' then 'FOOTBALL'"+"WHEN text like '%Cricket%' then 'CRICKET'"+
      "WHEN text like '%Basketball%' then 'BASKETBALL' ELSE 'NogivenSport'"+ "END AS game from twit where text is not null group by game,user.lang order by game")
```



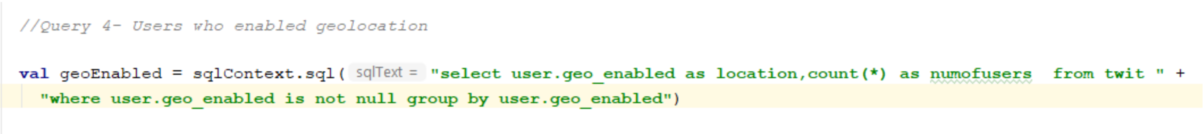
Query 3: More tweets coming from locations



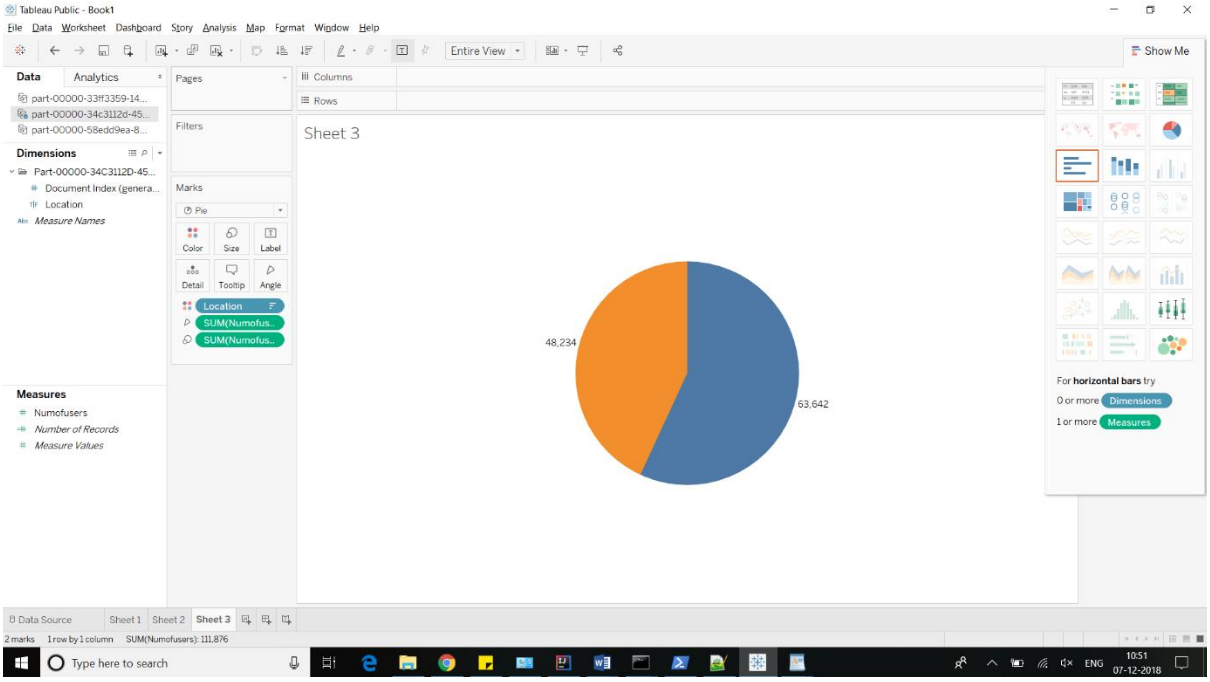
```
// Query 3 - More tweets coming from locations
 val moretweetLoc = sqlContext.sql( sqlText = "select user.location as location,count(*) as numoftweets  from twit where user.location is not null " +
   "group by user.location " + "order by count(1) desc limit 10")
```



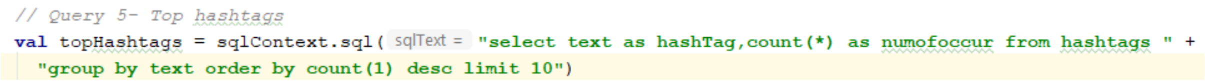
Query 4: Users who enabled geolocation



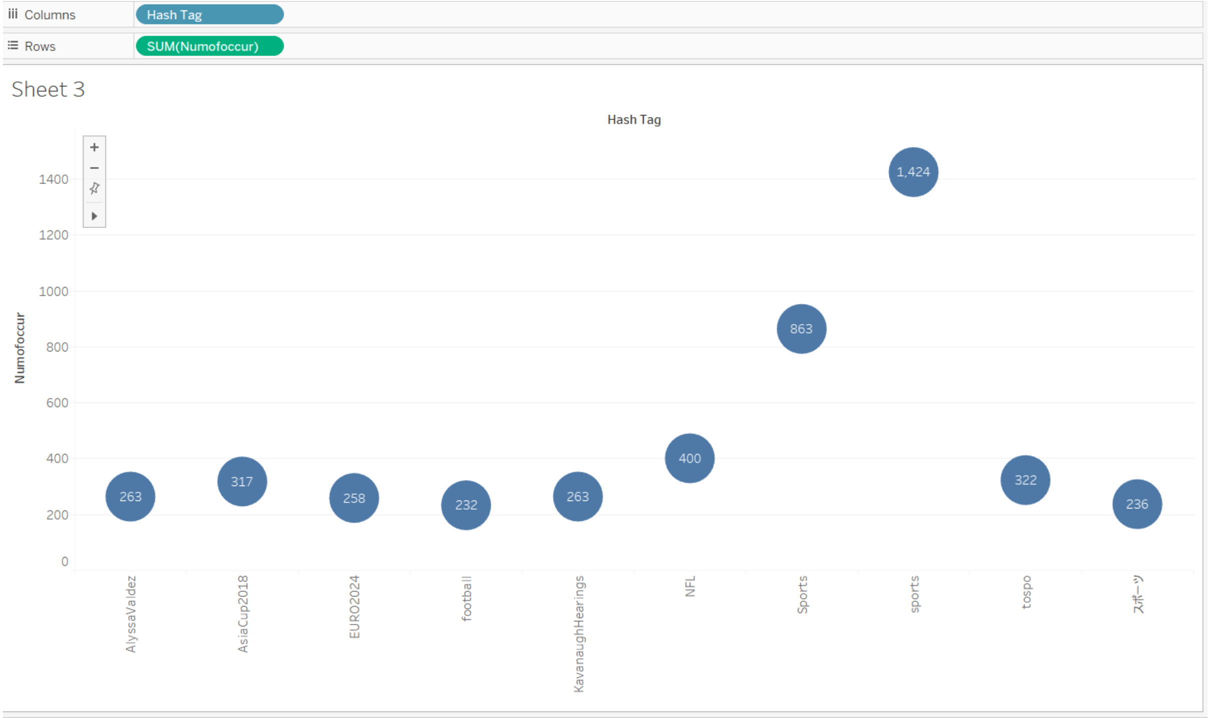
```
//Query 4- Users who enabled geolocation

val geoEnabled = sqlContext.sql( sqlText = "select user.geo_enabled as location,count(*) as numofusers  from twit " +
  "where user.geo_enabled is not null group by user.geo_enabled")
```



Query5: Top Hashtags in sports tweets



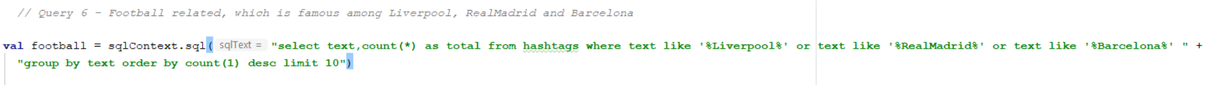
```
// Query 5- Top hashtags
val topHashtags = sqlContext.sql( sqlText = "select text as hashTag,count(*) as numofoccur from hashtags " +
  "group by text order by count(1) desc limit 10")
```



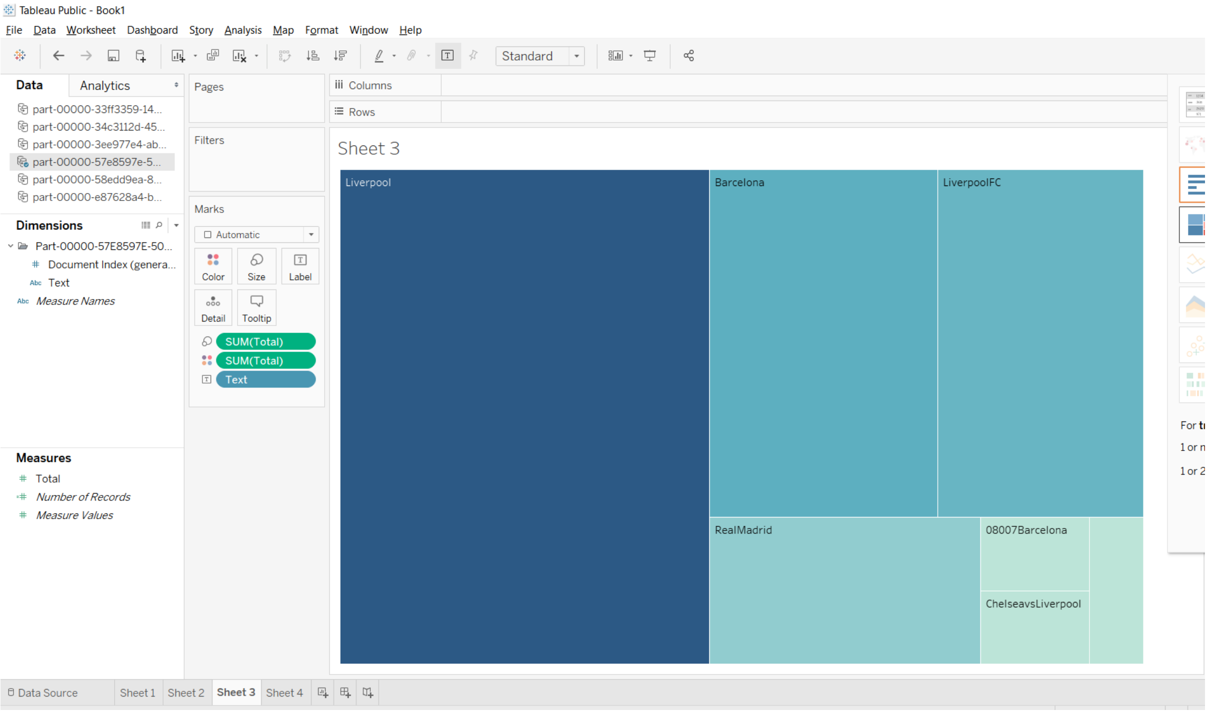
Columns: Hash Tag

Rows: SUM(Numofoccur)

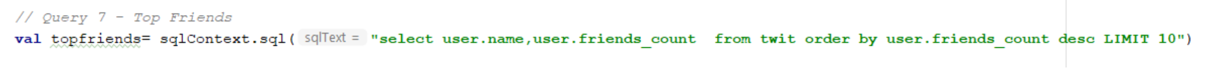
Query 6: Football related, which is famous among Liverpool, RealMadrid and Barcelona



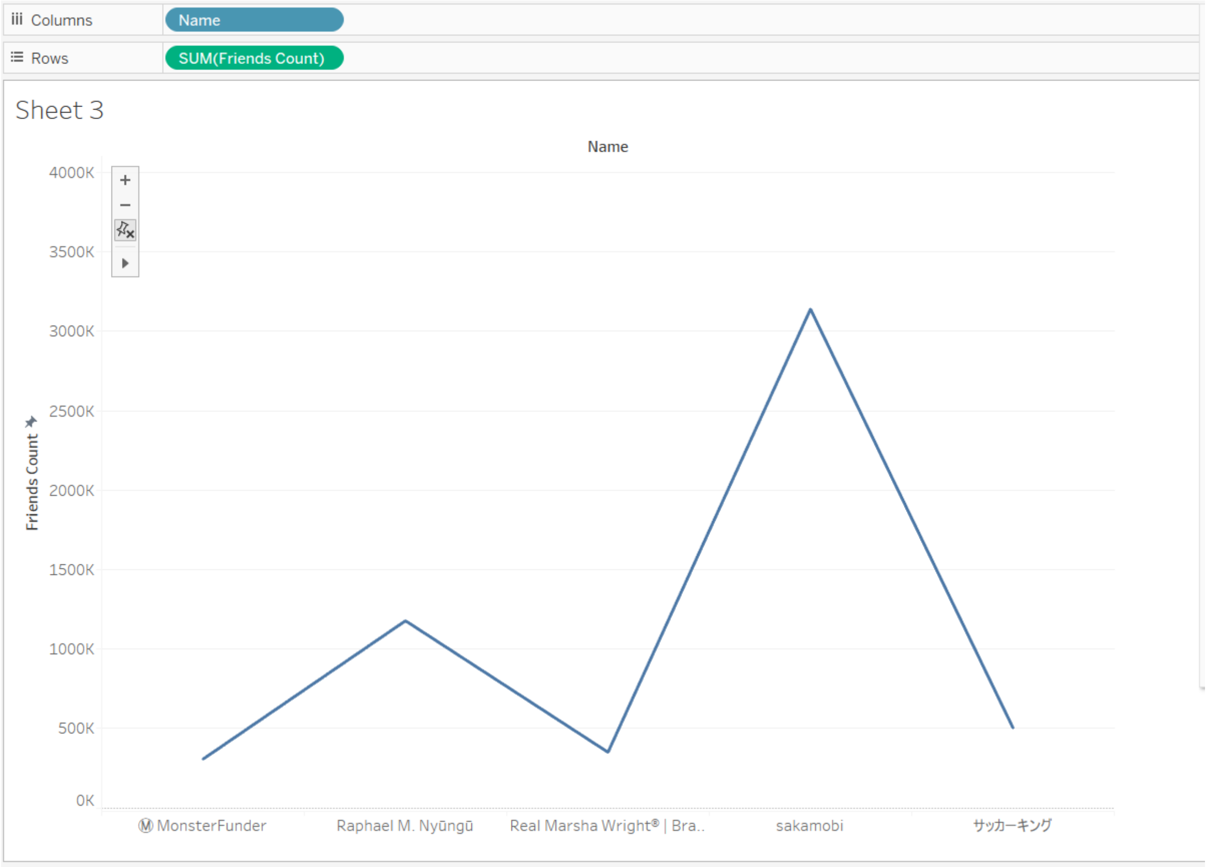
```
  // Query 6 - Football related, which is famous among Liverpool, RealMadrid and Barcelona

val football = sqlContext.sql( sqlText = "select text,count(*) as total from hashtags where text like '%Liverpool%' or text like '%RealMadrid%' or text like '%Barcelona%' " +
  "group by text order by count(1) desc limit 10")
```



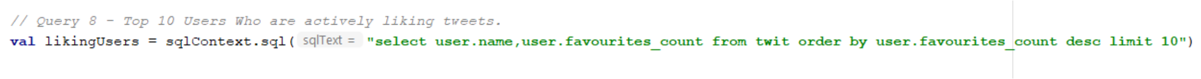
Query7: Users who got more friends who are tweeting about sports



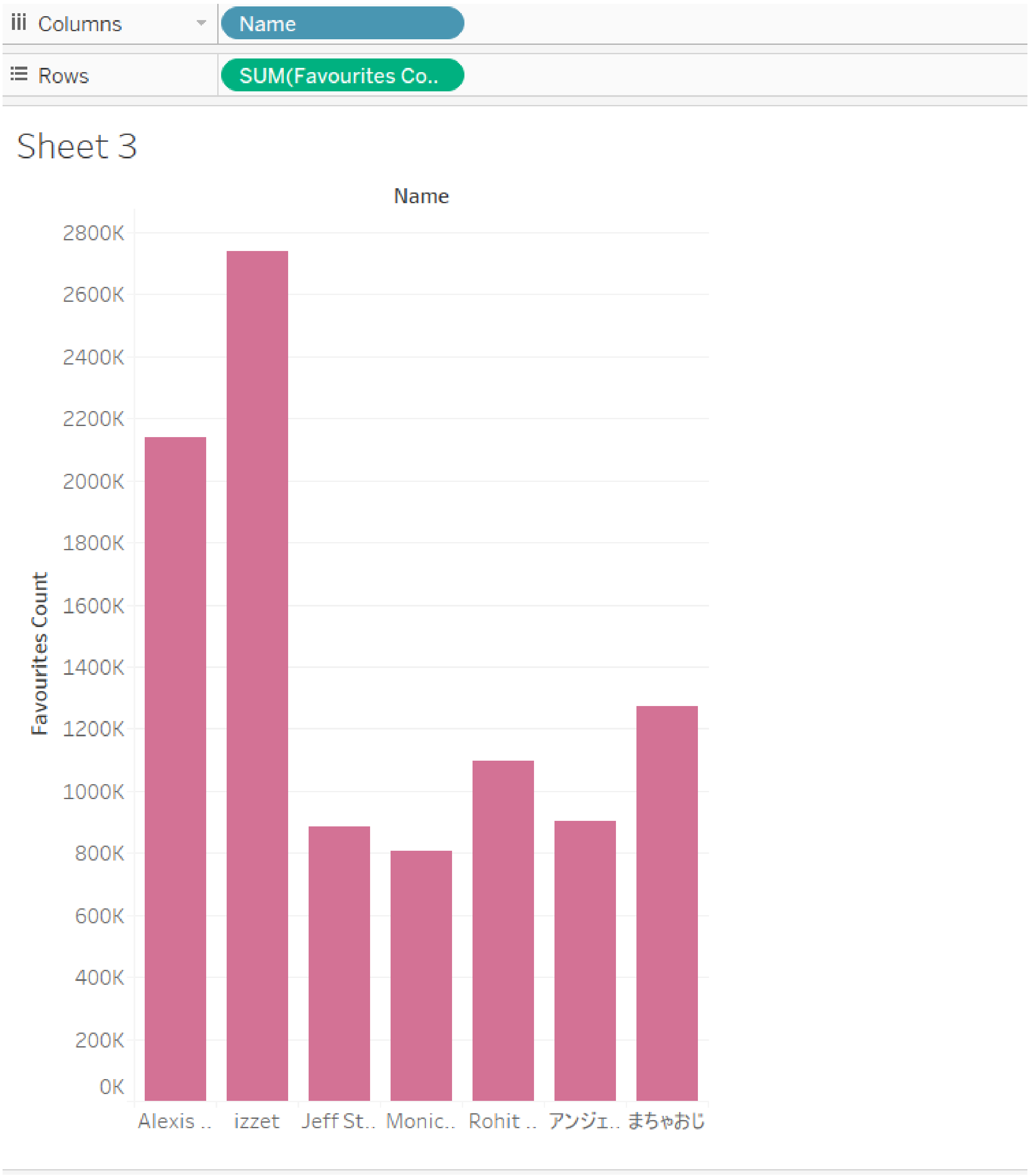
```
// Query 7 - Top Friends
val topfriends= sqlContext.sql( sqlText = "select user.name,user.friends_count  from twit order by user.friends_count desc LIMIT 10")
```



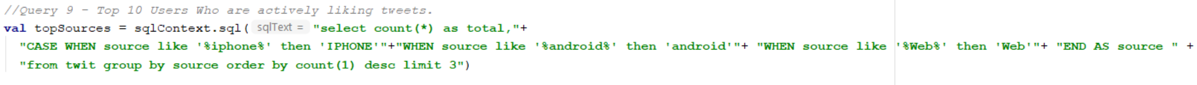
Query 8: Top 10 Users who are actively liking tweets



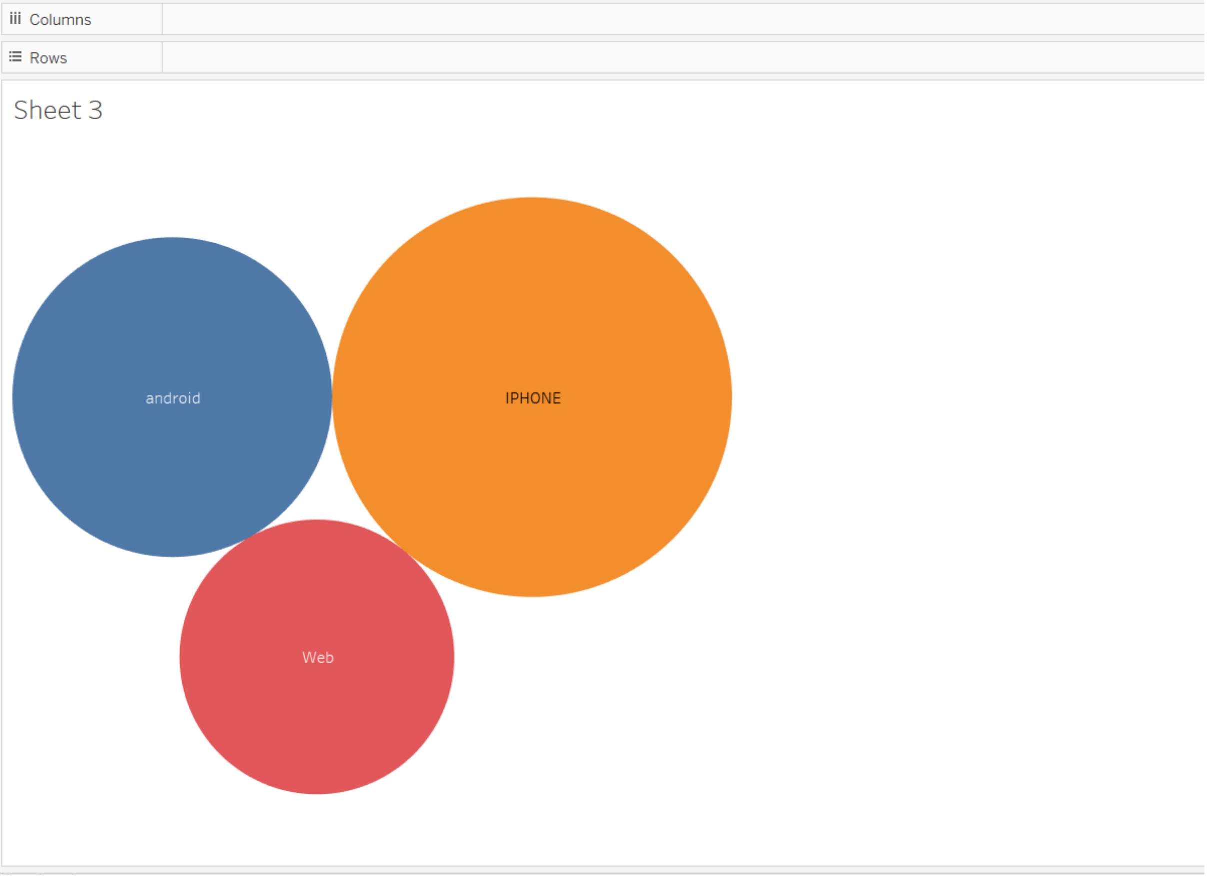
```
// Query 8 - Top 10 Users Who are actively liking tweets.
val likingUsers = sqlContext.sql( sqlText = "select user.name,user.favourites_count from twit order by user.favourites_count desc limit 10")
```



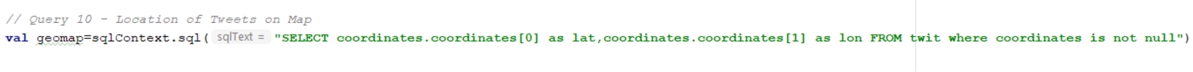
Query 9: Top 10 Users Who are actively liking tweets.



```
//Query 9 - Top 10 Users Who are actively liking tweets.
val topSources = sqlContext.sql( sqlText = "select count(*) as total,"+
  "CASE WHEN source like '%iphone%' then 'IPHONE'"+"WHEN source like '%android%' then 'android'"+ "WHEN source like '%Web%' then 'Web'"+ "END AS source " +
  "from twit group by source order by count(1) desc limit 3")
```



Query10: Location of sports from where tweets are coming



```
// Query 10 - Location of Tweets on Map
val geomap=sqlContext.sql( sqlText = "SELECT coordinates.coordinates[0] as lat,coordinates.coordinates[1] as lon FROM twit where coordinates is not null")
```

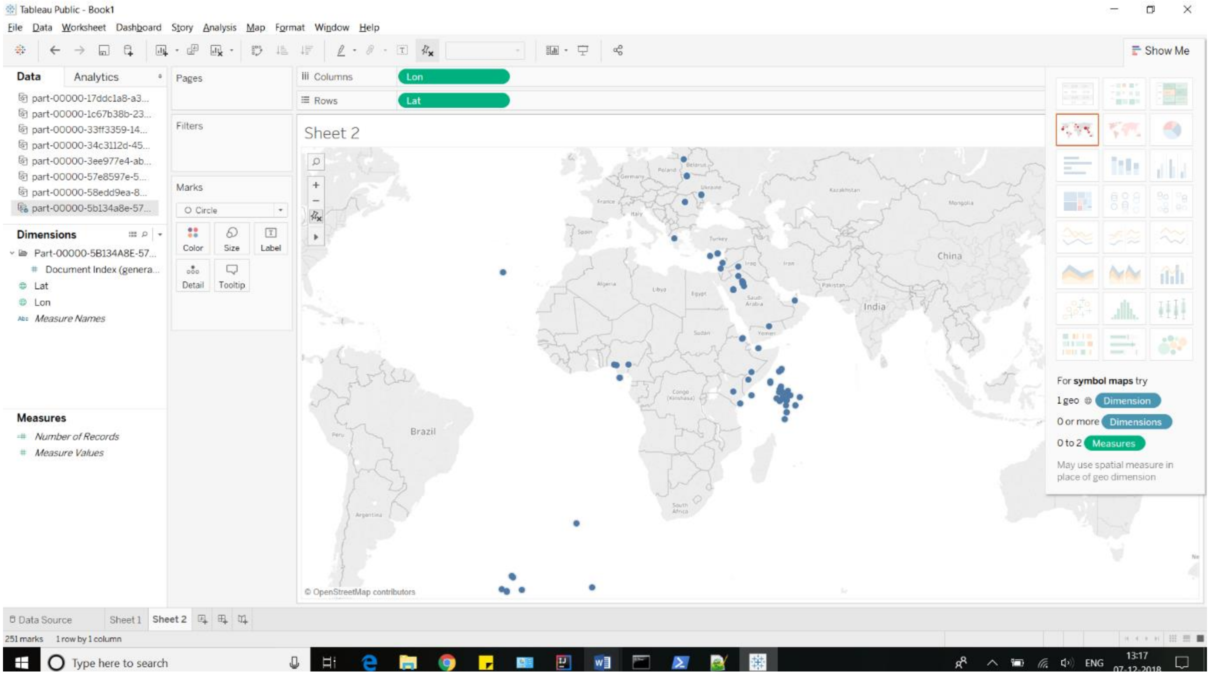
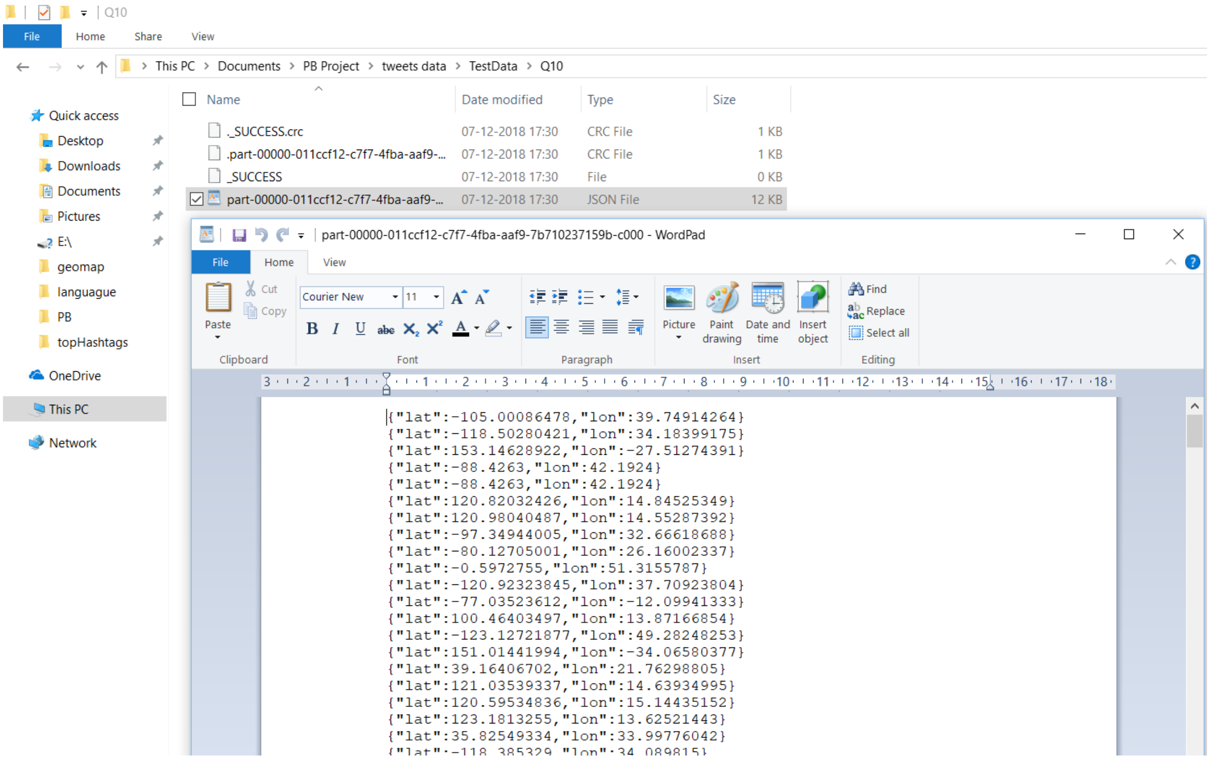


Tableau Public - Book1
File Data Worksheet Dashboard Story Analysis Map Format Window Help
Data Analytics
part-00000-17ddc1a8-a3...
part-00000-1c67b38b-23...
part-00000-33ff3359-14...
part-00000-34c3112d-45...
part-00000-3ee977e4-ab...
part-00000-57e8597e-5...
part-00000-58edd9ea-8...
part-00000-5b134a8e-57...
Dimensions
Part-00000-5B134A8E-57...
Document Index (genera...
Lat
Lon
Measure Names
Measures
Number of Records
Measure Values
Pages
Filters
Marks
Circle
Color Size Label
Detail Tooltip
Columns Lon
Rows Lat
Sheet 2
China
India
Brazil
© OpenStreetMap contributors
Show Me
For symbol maps try
1 geo Dimension
0 or more Dimensions
0 to 2 Measures
May use spatial measure in place of geo dimension
Data Source Sheet 1 Sheet 2
251 marks 1 row by 1 column
Type here to search
ENG
13:17
07-12-2018

Testing:

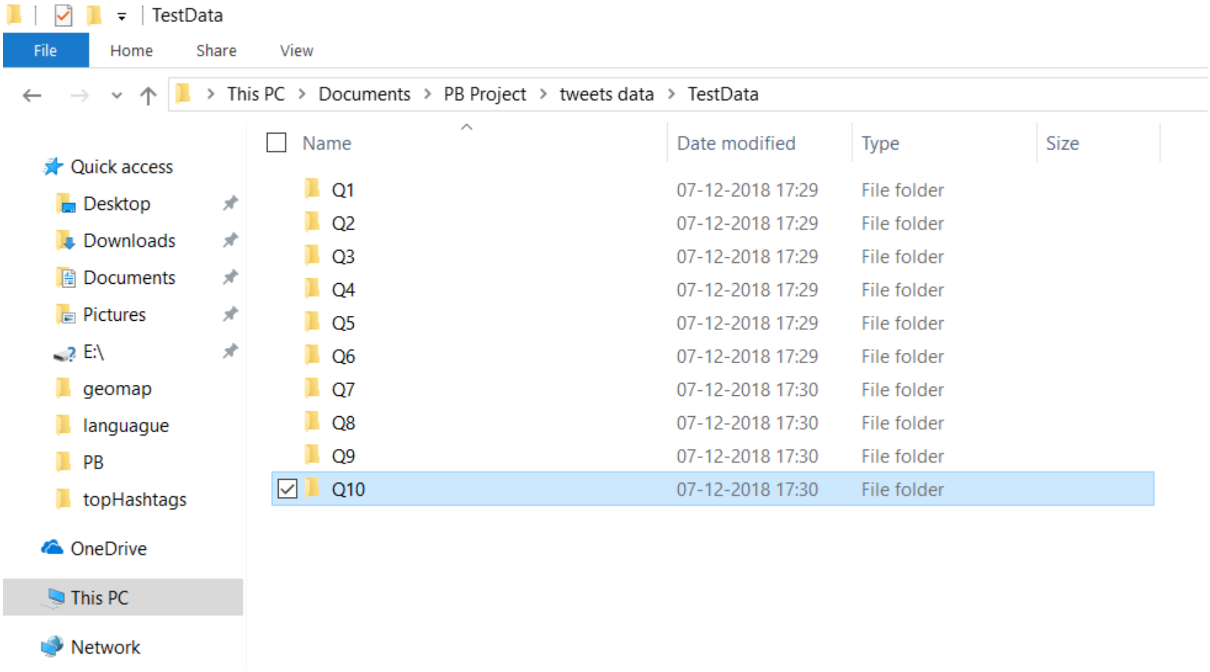
Generated output from the queries are written to the json files to local directory and done manual testing on the output fields whether the output is generated or not.


Q10
File Home Share View
This PC > Documents > PB Project > tweets data > TestData > Q10
Quick access
Desktop
Downloads
Documents
Pictures
E:\
geomap
languague
PB
topHashtags
OneDrive
This PC
Network
part-00000-011ccf12-c7f7-4fba-aaf9-7b710237159b-c000 - WordPad
File Home View
Paste Cut Copy
Courier New 11
Picture Paint drawing Date and time Insert object
Find Replace Select all
Clipboard Font Paragraph Insert Editing

Name	Date modified	Type	Size
._SUCCESS.crc	07-12-2018 17:30	CRC File	1 KB
.part-00000-011ccf12-c7f7-4fba-aaf9-...	07-12-2018 17:30	CRC File	1 KB
_SUCCESS	07-12-2018 17:30	File	0 KB
part-00000-011ccf12-c7f7-4fba-aaf9-...	07-12-2018 17:30	JSON File	12 KB

```
{"lat":-105.00086478,"lon":39.74914264}
{"lat":-118.50280421,"lon":34.18399175}
{"lat":153.14628922,"lon":-27.51274391}
{"lat":-88.4263,"lon":42.1924}
{"lat":-88.4263,"lon":42.1924}
{"lat":120.82032426,"lon":14.84525349}
{"lat":120.98040487,"lon":14.55287392}
{"lat":-97.34944005,"lon":32.66618688}
{"lat":-80.12705001,"lon":26.16002337}
{"lat":-0.5972755,"lon":51.3155787}
{"lat":-120.92323845,"lon":37.70923804}
{"lat":-77.03523612,"lon":-12.09941333}
{"lat":100.46403497,"lon":13.87166854}
{"lat":-123.12721877,"lon":49.28248253}
{"lat":151.01441994,"lon":-34.06580377}
{"lat":39.16406702,"lon":21.76298805}
{"lat":121.03539337,"lon":14.63934995}
{"lat":120.59534836,"lon":15.14435152}
{"lat":123.1813255,"lon":13.62521443}
{"lat":35.82549334,"lon":33.99776042}
{"lat":-118.385329,"lon":34.089815}
```

Output folders that are generated from the queries.



Name	Date modified	Type	Size
Q1	07-12-2018 17:29	File folder	
Q2	07-12-2018 17:29	File folder	
Q3	07-12-2018 17:29	File folder	
Q4	07-12-2018 17:29	File folder	
Q5	07-12-2018 17:29	File folder	
Q6	07-12-2018 17:29	File folder	
Q7	07-12-2018 17:30	File folder	
Q8	07-12-2018 17:30	File folder	
Q9	07-12-2018 17:30	File folder	
☑ Q10	07-12-2018 17:30	File folder	

References:

https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object

Thank You