**Principles of Big Data Management**

**Project Phase - 2**

****

**Team Members:**

**Rajeswari Devi Namana-18135299**

**Sai Sravya Reddy Bokka-16240386**

**Ankita Wankhede-16233344**

**Table of Contents**

**1. Introduction……………………………………………………………………………… 2  
2. Working Flow…………………………………………………………………………….. 3  
3. System Setup and specifications….……………………………………………………... 4  
4. Queries…………………………………………………………………………………… 5  
4.1 Query1…………... …………………………………………………………………….. 5  
4.2 Query 2 ………………………………………………………………………………… 6  
4.3 Query 3 …………………………………………………………………………………. 7  
4.4 Query 4 …………………………………………………………………………………. 8  
4.5 Query 5………………………………………………………………………………….10**

**4.6 Query 6………………………………………………………………………………….11**

**4.7 Query 7………………………………………………………………………………….12**

**4.8 Query 8………………………………………………………………………………….14**

**4.9 Query 9………………………………………………………………………………….15**

**4.10 Query 10……………………………………………………………………………….16  
5. References………………………………………………………………………………..18**

**Project – Twitter streaming API**

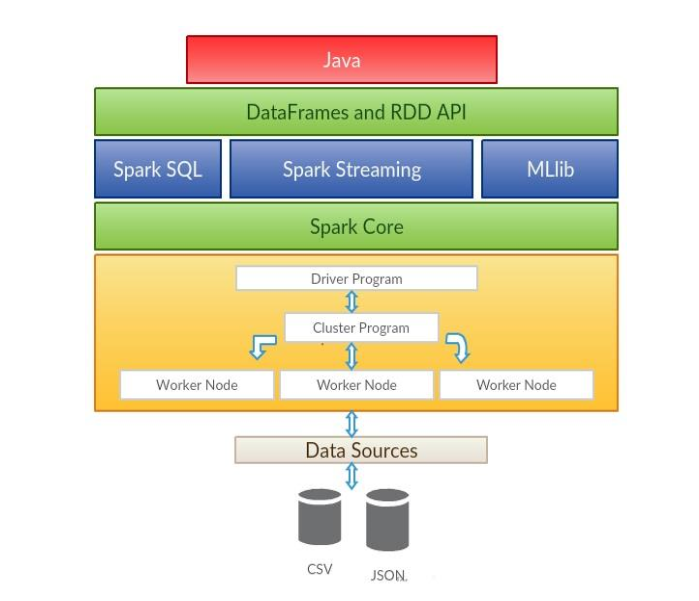
**Introduction:**

A Twitter streaming API return Tweets provide that data encoded using JavaScript Object Notation (JSON). A JSON is based on “key-value” pairs, with named attributes and associated values. The attributes, and their state, are used to describe objects. Many objects such as ‘Tweets and Users’ are used as JSON at Twitter,where objects all encapsulate the core attributes that describe that object. Each Tweet has many objects such as an author, a message, a unique ID, a timestamp of when posted, and sometimes metadata also which is shared by the user. Each User has a Twitter name, an unique ID, number of followers, and in most cases an account biography. Every Tweet also generates 'entity' objects, which are arrays of common Tweet contents such as hashtags, mentions, media, and links. A Tweet can have over 150 attributes associated with it. A sample tweet where “mom” keyword is used for collecting tweets looks like:

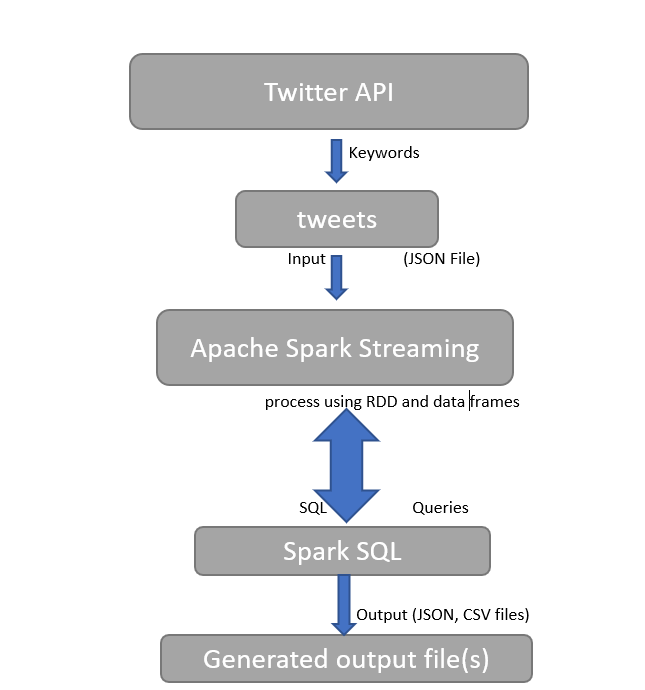


This report presents a design and implementation on how to use store the tweets in Spark SQL using Apache Spark. Using analytic queries to explore and understand the data. For each query, visualizations (e.g., pie chart, ring, bar graphs) are used to represent the output in a format. To implement this project we used ubuntu, a virtual machine and pycharm to develop the visualizations for data representation.

**Architecture:**

****

**Working flow:**

****

**System setup and specifications:**

**Operating System: Linux(Ubuntu 14.0.4) , Redhat**

**IDE: Webstorm**

**Browser: Google Chrome**

**Language: Python**

**Database: Apache Spark SQL**

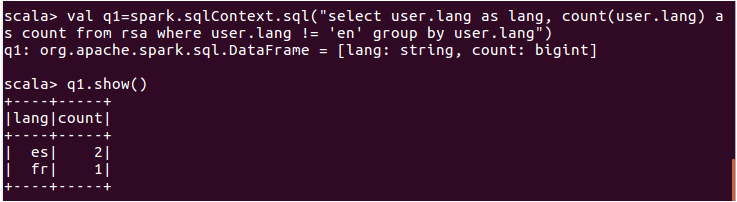
**Library: Apache Spark Core, Spark Ml lib, Spark SQL, Twitter Streaming API.**

**Queries:**

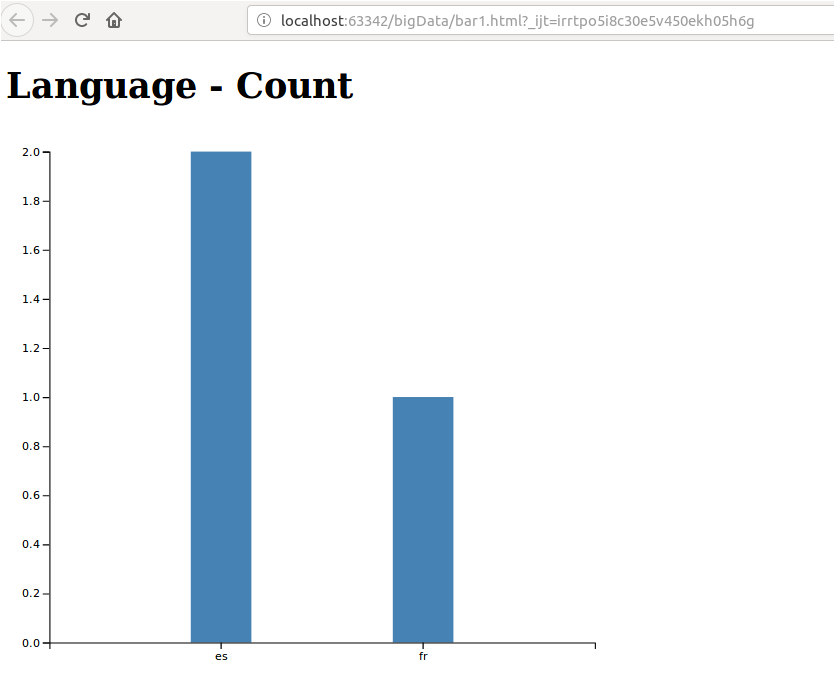
**Query 1: A dataframe to return the number of users who do not use English (Language Count)**

This query displays the count of the users who do not use English language. The count of the users is calculated by using the count() function which is predefined and using a where clause.

And also to simplify we use group by clause. To visualize the output bar graph is used.

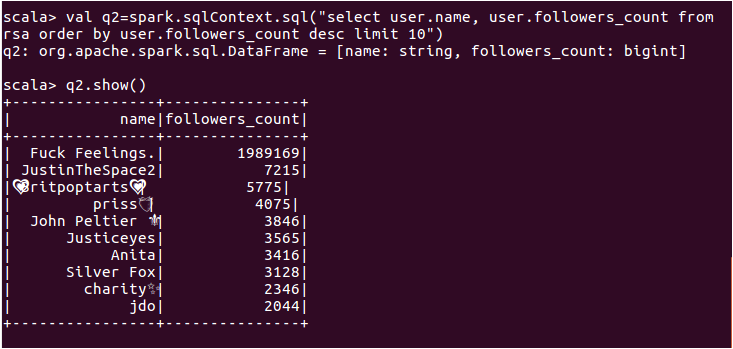


**Visualization:**

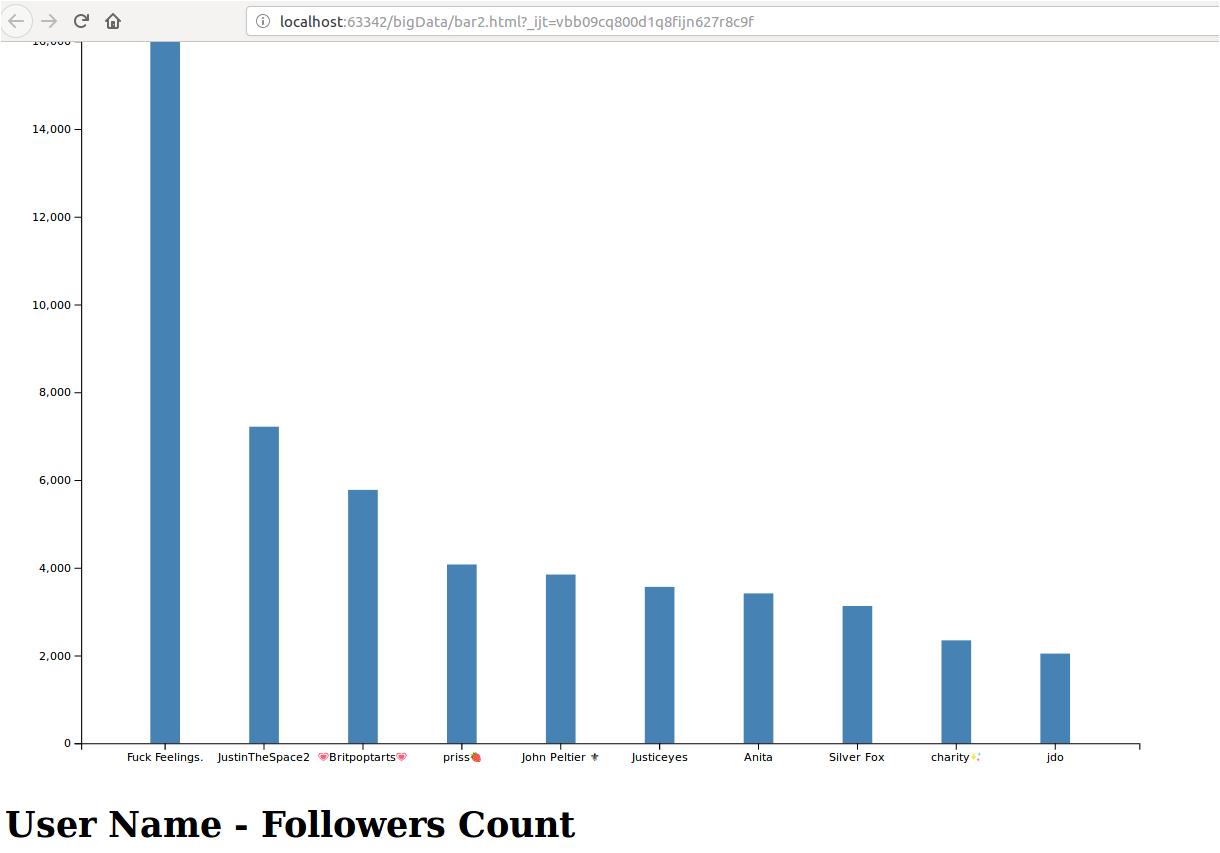


**Query 2: A dataframe to get the top 10 users who have the highest number of followers.**

This query displays the top 10 user names who have the highest number of followers. Also the followers count is displayed. To get the top 10 we use “desc limit 10” clause. To visualize the data bar graph is used.

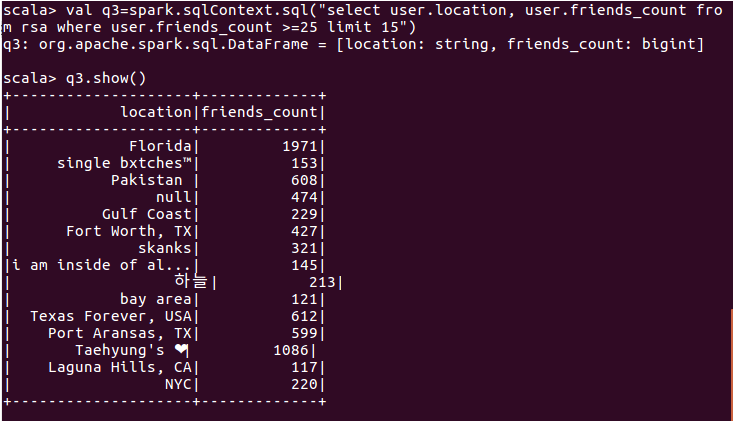
****

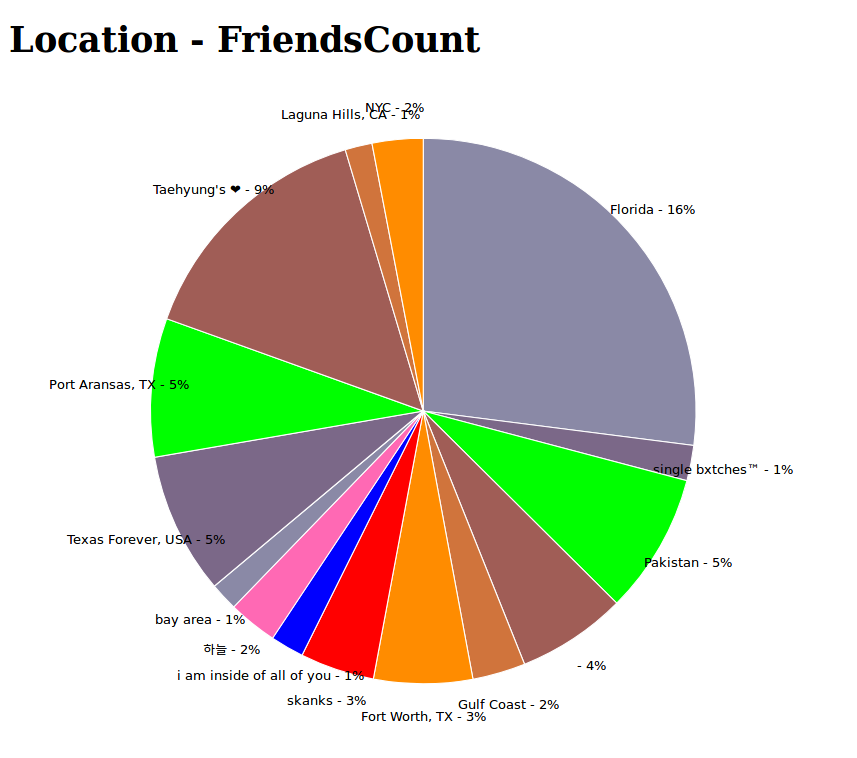
**Visualization:**

****

**Query 3: Dataframe to group the number of user’s in a location based on friends count.**

In the query we group the first 15 users based on location and friend’s count(>=25). To get the users with friends count greater than or equal to 25 we use the where clause ‘user.friends\_Count >=25’. And to limit the users to first 15 ‘limit 15’ is used.

To visualize the output data we used pie chart with the location and their respective percentages.

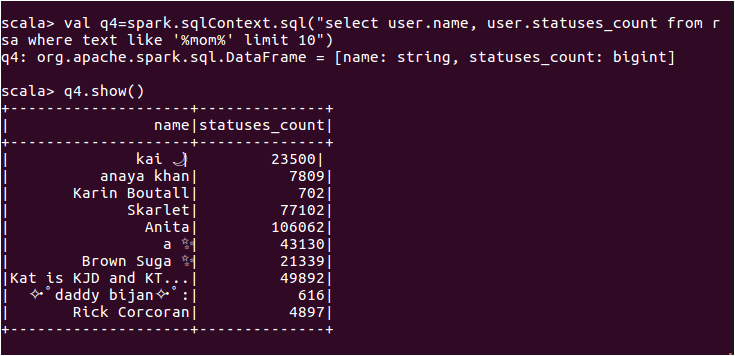


**Query 4: Dataframe to get user statuses count with a particular text**

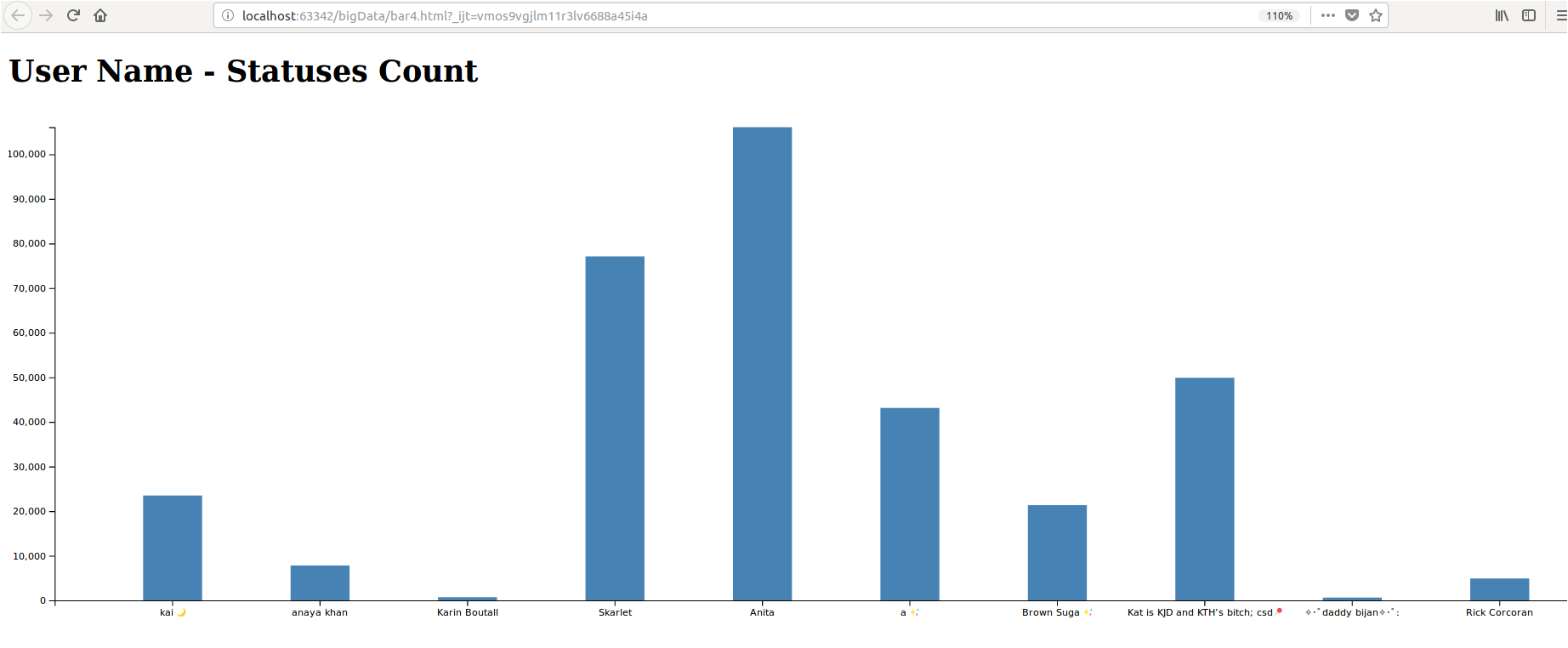
In the query we displayed the first 10 users whose tweeter status consists of the word ‘mom’.

We displayed the user name, and statuses count. To check for the word we used a where clauses with “text like ‘%mom%’ “ , and to the limited number of users we used “ limit 10”.

To visualize the output data bar graph is used.

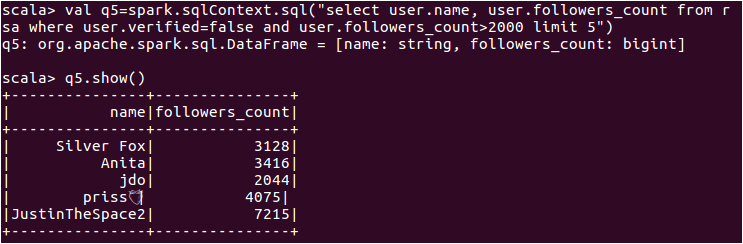


**Visualization**:

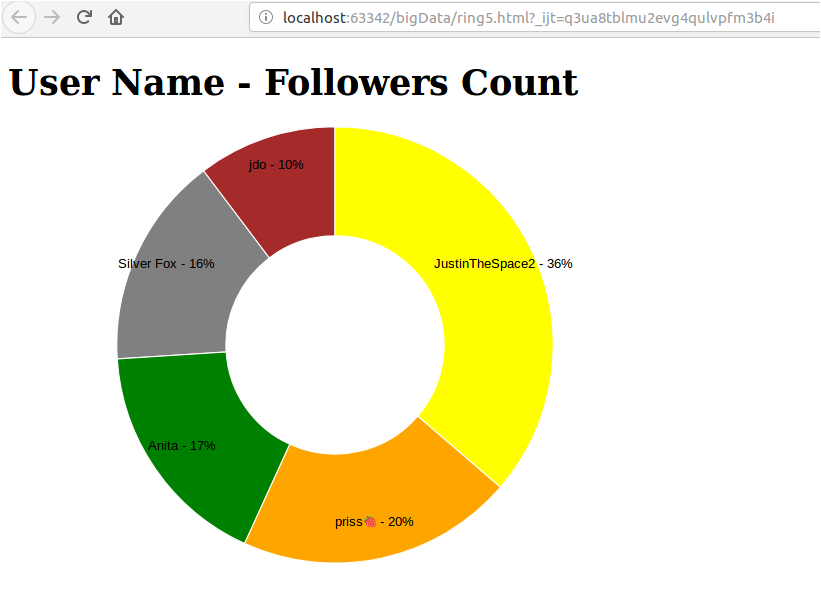


**Query 5: Dataframe to get users who are not celebrities with a very high followers count.**

This query displays the user names of top 5 users who are **not celebrities** that is by flagging the verified attribute to false and fetching users whose followers count is greater than 2000. For checking the users if they are verified or not where clause is used “ user.verified=false” and “user.followers\_count>2000”. To visualize data ring chart is used.

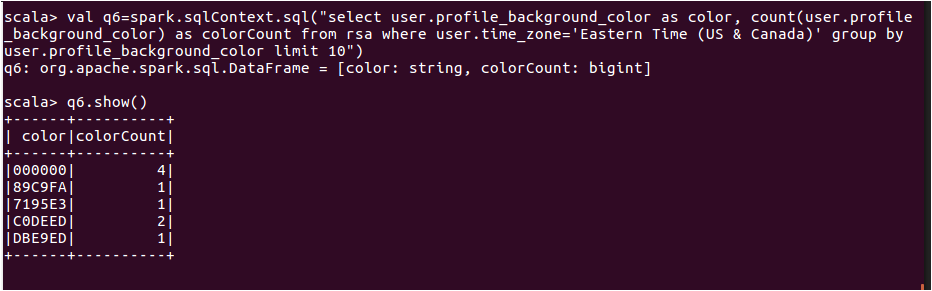


**Visualization:**

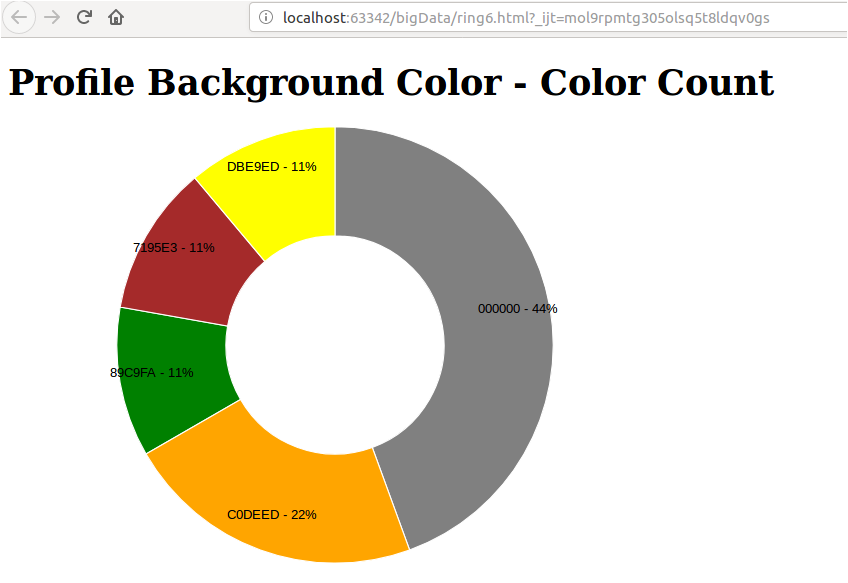


**Query 6: Dataframe to the users background color in particular time\_zone.**

This query displays user’s profile background color and grouping them by color and displaying the results only where the time zone is **“Eastern time (US & Canada)”.** To get the count of the users with profile background color count(user.profile\_background-color) is used. To visualize the output data ring chart is used.

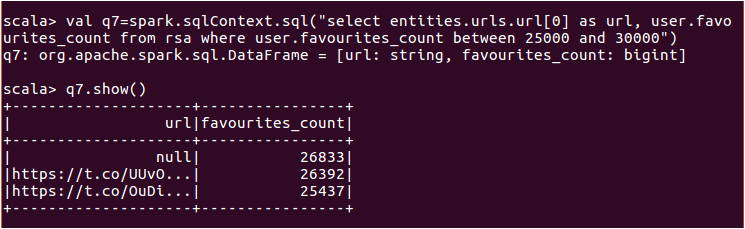


**Visualization:**

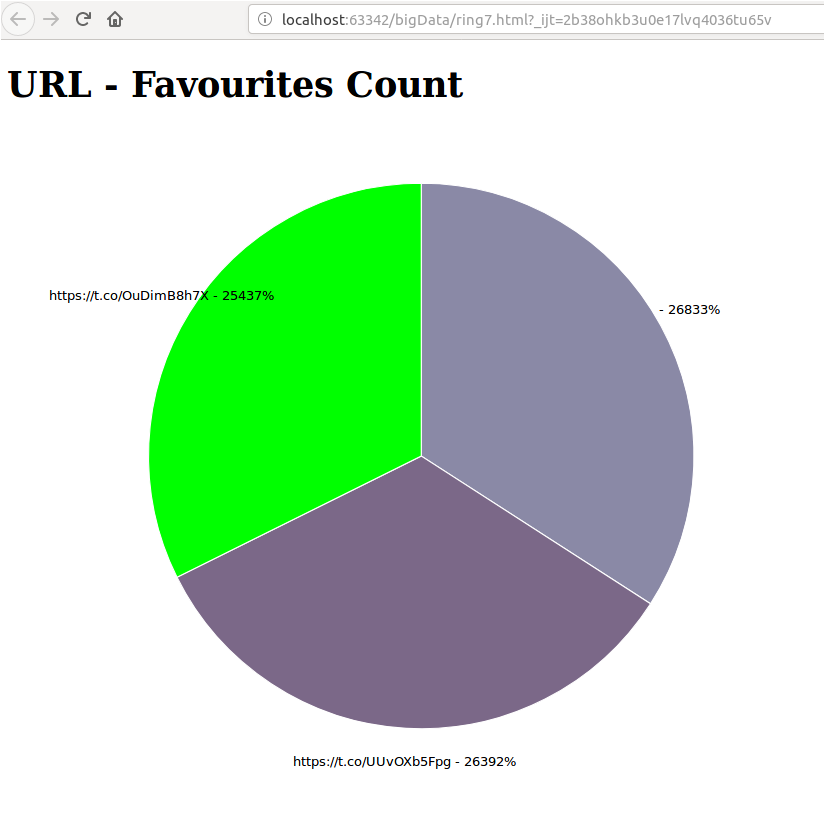


**Query 7: Dataframe to display popular URLs.**

This query displays popular URLS with a favourites count between 25000 to 30000**.** To check the count where clause is used “ user.favourites\_count between 25000 and 30000”. For visualization pie chart is used.



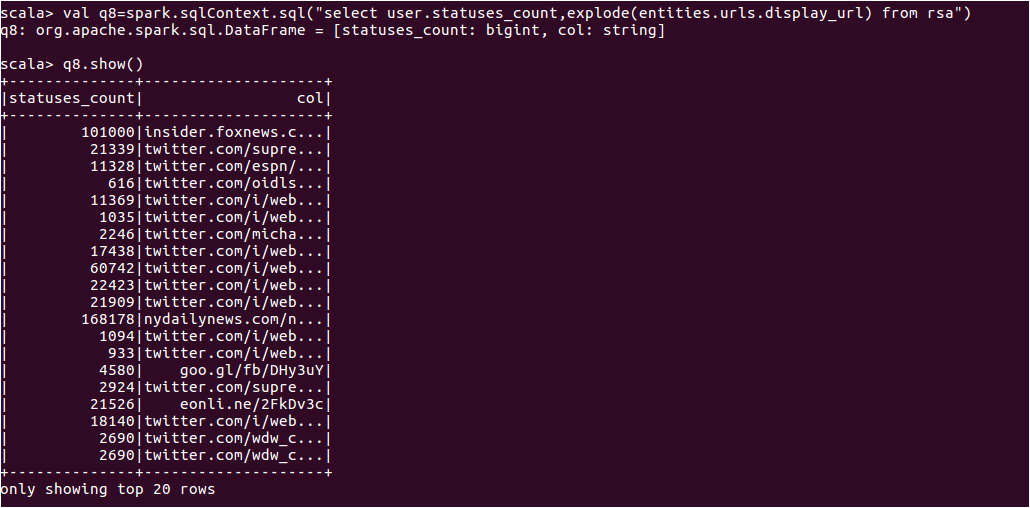
**Visualization:**



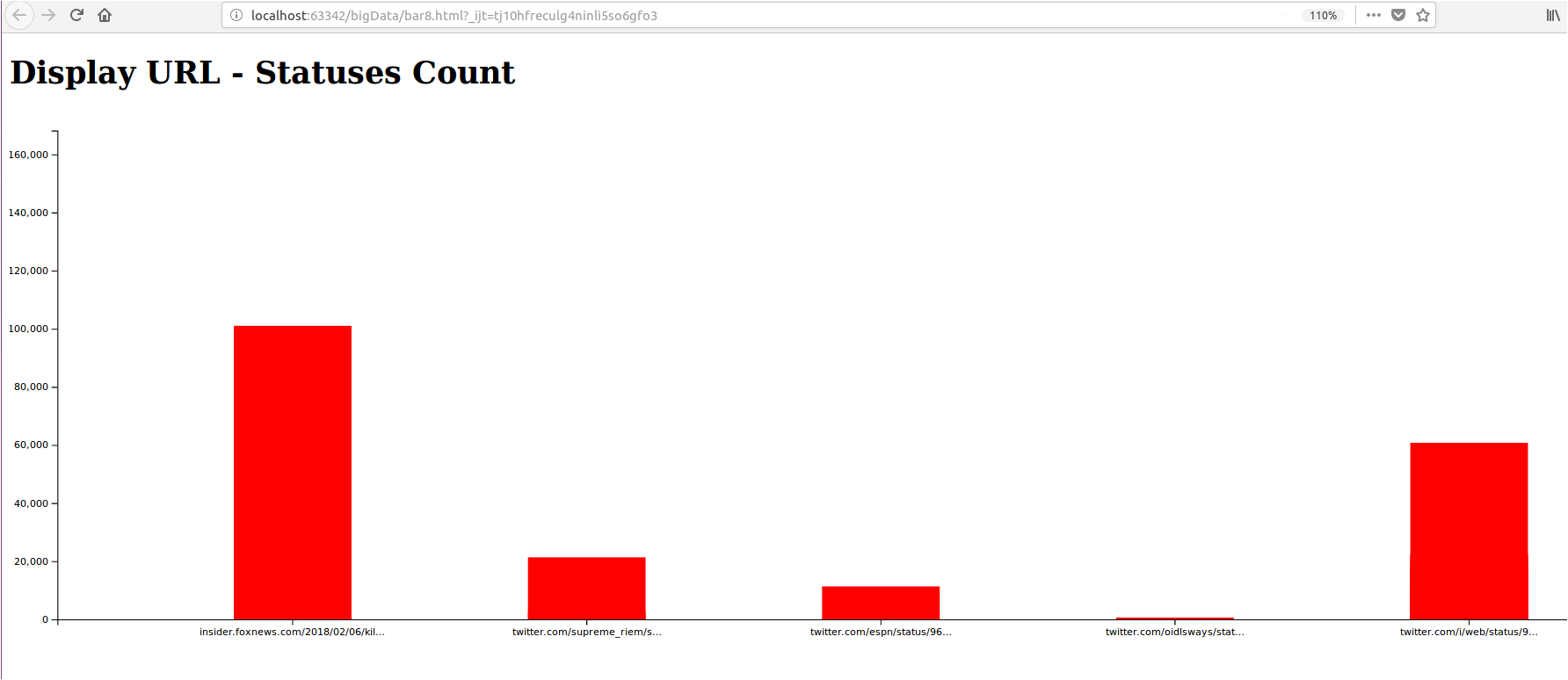
**Query8: Dataframe to display user statuses\_count thereby by displaying urls.**

In this query we are displaying the url’s of the users and also the statuses count. All the statuses count in that url’s

To get entire url displayed we used explode command. The command is explode(entities.urls.display\_url).

****

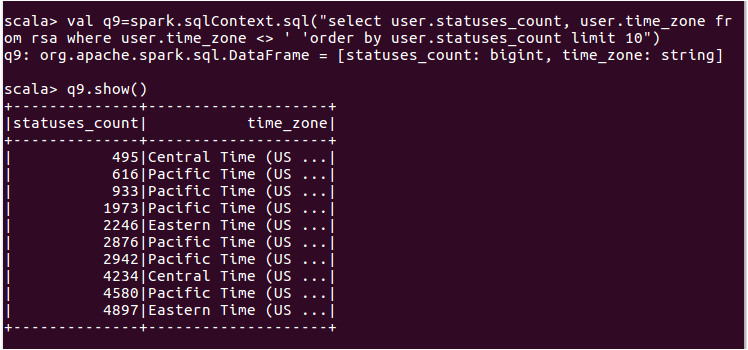
**Visualization:**

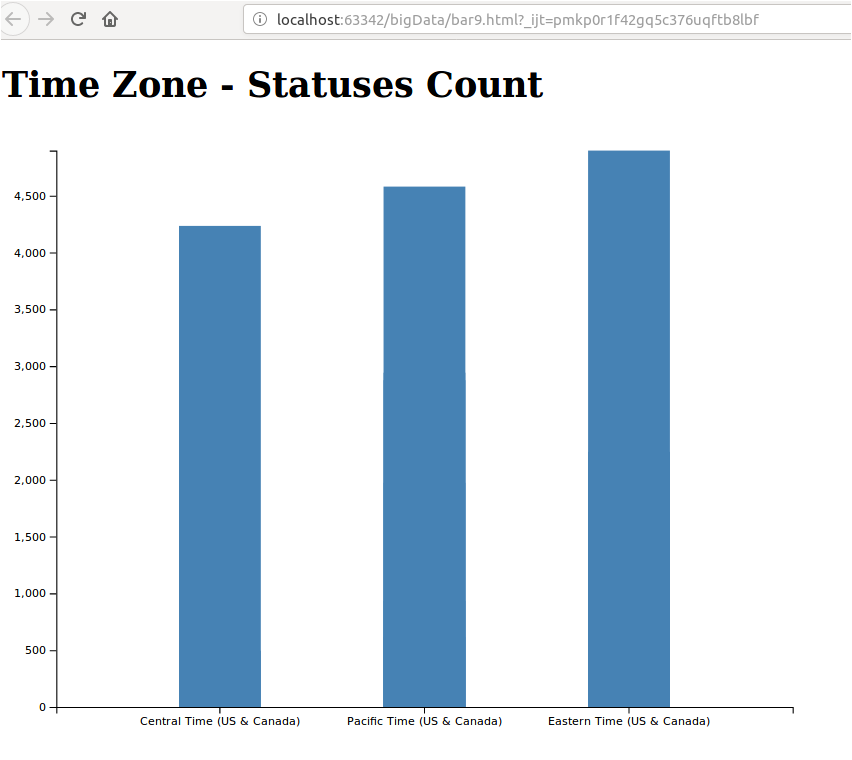


**Query9: Dataframe to generate time zone and status counts**

Displaying user’s status counts in ascending order and their time zones where time zone is not

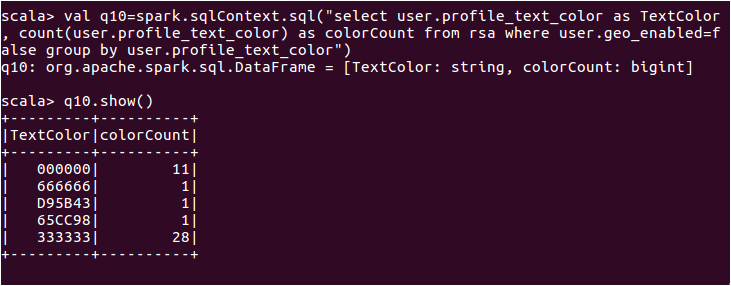
null.



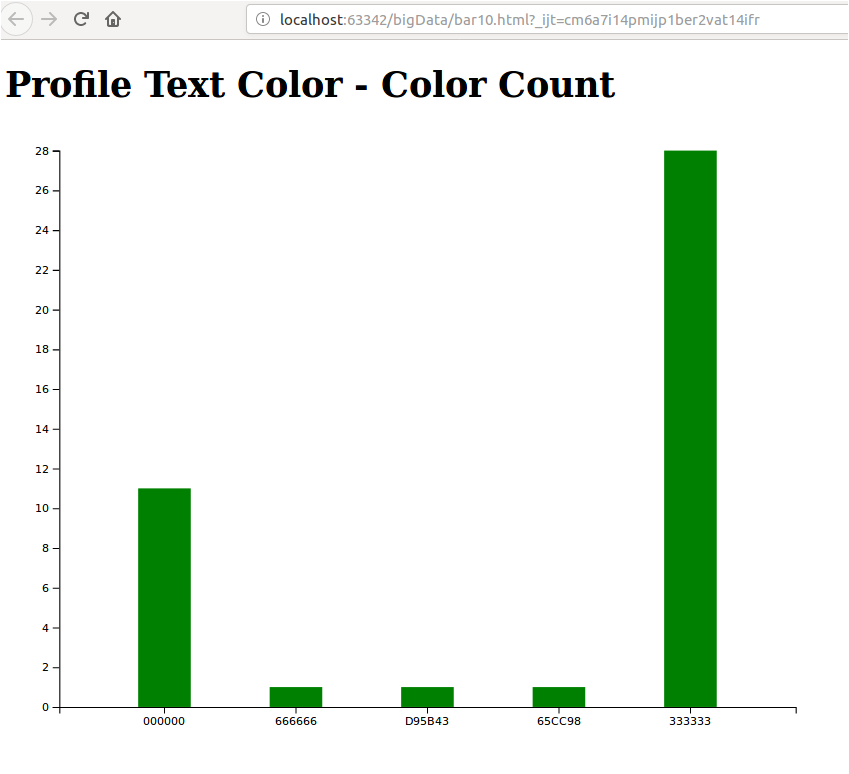
**Visualization:**

**Query10: dataframe to generate profile text color and the respective color count**

This query is used to retrieve the profile text color of a user whoes geo enabled is flagged as false. Here colorcount is the count of the profile\_text\_color and using where clause ‘user.geo\_enabled=false’, we flagged it.



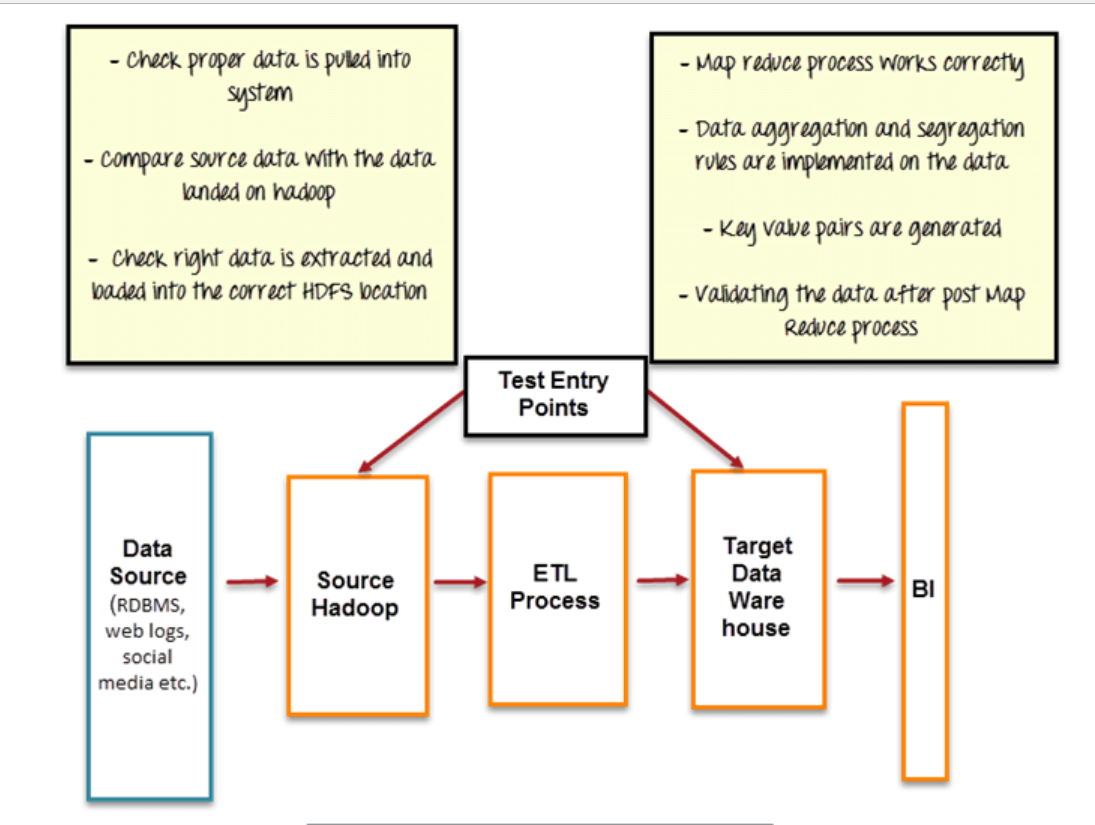
**Visualization:**



**Test Scenario:**

Installed hadoop and Spark in different Operating system such as Mac OS, Linux (RedHat and Ubuntu) and checked how it work on various Operating Systems.

**Testing Steps in verifying Big Data Applications**



### **Test Environment Needs:**

Test Environment needs depend on the type of application you are testing. For Big data testing, test environment should encompass,

* It should have enough space for storage and process large amount of data.
* It should have cluster with distributed nodes and data.
* It should have minimum CPU and memory utilization to keep performance high.

**References**:

<https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json>

<https://spark.apache.org/docs/1.6.1/sql-programming-guide.html>

<https://www.uclassify.com/doc>

<http://stackoverflow.com/questions/35211993/how-to-join-mutiple-columns-inspark-sql-using-java-for-filtering-in-dataframe>

<http://stackoverflow.com/questions/18471043/twitter-stream-api-to-get-userstweets>

<http://getbootstrap.com/css/#grid-offsetting>