**Big Data Programming**

**Final Report**

**Title:**

Twitter Web series Data Visualization with Spark ETL

**Team Members:**

* Avinash Ganguri
* Akhil Teja Kanugolu
* Geetanjali Makineni
* Bhashitha Siddareddy

Github link: <https://github.com/avinashganguri/Analysis-on-web-series>

**Introduction:**

Twitter Data Analysis with Spark ETL and visualizing is our title. Here we are taking the Twitter’s Web series data. The main objective of our project is doing the ETL process using the Spark’s Batch Processing and then Spark Integration using Web UI. The main source of our data is twitter and then collecting the data with Spark Batch Process. We can perform our transactions on the set of RDD’s and later we load our data in our Hive which is similarly equal to the ETL basic process. This is because we are living in a world where data handling and data using plays one most important role in making the decisions for most of the industries

**Background:**

1. We first collect the tweets from the Twitter api.

2. Later, we import the data that is collected into the hive from the HDFS.

3. Next we will export this data from there into RDBMS by usage of sqoop.

4. We later do the sentimental analysis on the tweets that are collected.

5. Now, we are using spark sql for writing required queries later by visualizing (examples like bar graphs and pie charts and some other type of graphs) the obtained results by using panda or Tableau.

**Goals and Objectives**

**Motivation:**

The motivation behind doing this analysis is we are living in a world where data handling and data using plays one most important role in making the decisions for most of the industries like Banking, Financial, Telecom and Health and IT sector serving ones. The main factors would be for getting the insights would be like managing its sheer volumes of data and its insights. Using the Apache Spark is one of the best amazing kind frameworks which will be handling big data and its real time performance of these analysis.

**Objectives:**

The main objective of our project is doing the ETL process using the Spark’s Batch Processing and then Spark Integration using Web UI. The main source of our data is twitter and then collecting the data with Spark Batch Process. We can perform our transactions on the set of RDD’s and later we load our data in our Hive which is similarly equal to the ETL basic process.

**Features:**

The main feature of the project is to collect the Real timed tweets from the twitter steaming API, also by performing the ETL which means we preprocess the data and extract the necessary data and then we load this data in our HIVE. Later, we feed the data into our HDFS and then we implement SQL and Hive queries. Sqoop is used for transferring data between SQL and HDFS.

**Significance:**

Analyzing of sentimental analysis is done using one tool which is already existing named as ML tool which is TextBlob for prediction of sentiment on the tweets and Later we are using spark for writing the queries by visualizing with Panda.

**Model Design:**

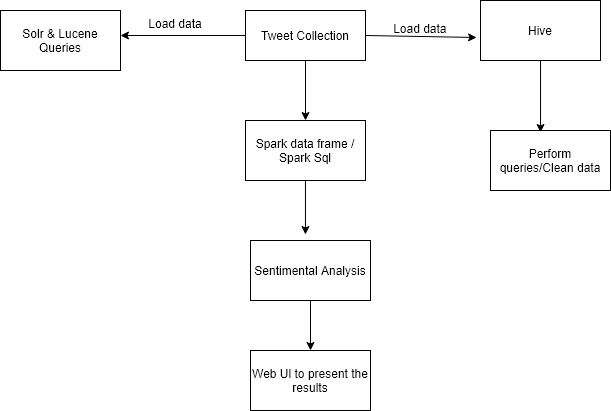


Fig: Architecture / Workflow Diagram

We have now first created one twitter developer account where next we got the tokens and credentials from API from where we took the twitter streaming and tweepy API in the python and stored tweets into the db. Extracted around 2.5 GB of dataset. Saved the data file in json format. Preprocess the data and create the database in spark framework with Scala programming. Will be creating a data frame and use the Sql Context to execute the SQL queries. In our next phases we are implementing it on Solrs and Lucene on our data and we also creating spark data frames on the file and creating different actions on data frames. We finally do the sentiment analysis on our data.

**Dataset:**

We collect the data from Twitter with API with a developer account credentials. Here we took the keywords as Netflix, TV shows, etc., These tweets are in JSON file format. We have collected nearly like 2.5GB size. It has an information that shows like web series, tv series, documentary, Netflix shows, tv shows. We are collecting here using an API which is in batch format which is nearly downloading like around 5kb for a second where we filter the keywords like Netflix shows, TV shows etc.,

|  |  |
| --- | --- |
| **Feature** | **Description** |
| id | Unique id of user |
| [created at](http://jsonviewer.stack.hu/) | Tweet created time stamp in UTC |
| text | UTF-8 text Tweet data |
| source | Type of device used to post the tweet |
| Name | Name of the user |
| Place | Geo location of user at time of tweet posted |
| Screen\_name | Profile or Screen name of the user |
| Lang | Language opted by user |
| user location | Location of the user |
| user\_followers\_count | Count of followers of user |
| user\_friends\_count | Count of friends of user |
| user\_favourites\_count | Count of people favorited the tweet |
| reply\_count | Count of replies to the tweet |
| retweet\_count | Count of retweets of the tweet |

Table: Description of Attributes of Data

Figure: Design of Features/ Attributes

**Analysis:**

We have now first created one twitter developer account where next we got the tokens and credentials from API from where we took the twitter streaming and tweepy API in the python and stored tweets into the db. Extracted around 2.5 GB of dataset. Saved the data file in json format. Preprocess the data and create the database in spark framework with scala programming. Will be creating a data frame and use the Sql Context to execute the SQL queries.

**Implementation:**

**Tweets collected:**

* First, we got a developer access from the Twitter. After that we have created one app to use those API’s.
* The code for which twitter streaming and tweepy for collected tweets is below:

Text

Description automatically generated

Text

Description automatically generated

**Spark using scala:**

Starting the Spark-shell,

**Text

Description automatically generated**

**Query 1:**

Number of Tweets that are from different countries on web series.

Text

Description automatically generated

**Query 2:**

We are here fetching the count of tweets based on the co-ordinates of web series.

We got the information based on the longitude and latitude.

Text

Description automatically generated

**Query 3:**

To fetch the different languages in which the tweets are made about web series.

We got the count of number of tweets about the web series in separate languages.

Text

Description automatically generated

**Query 4:**

Fetching the users account names with a greater number of tweets on web series.

Text

Description automatically generated

**Query 5:**

Fetching the users that have more followers who are tweeting based on web series.

Text

Description automatically generated

**Query 6:**

Fetching the number of tweets based on different web series.

Text

Description automatically generated

**Query 7:**

Fetching the non-verified users who are having more number of followers.

Text

Description automatically generated

**Query 8:**

Users with a greater number of retweets for their tweets on web series.

Text

Description automatically generated

**Query 9:**

Number of Tweets that are on basis of different locations in the United States on web series.

Text

Description automatically generated

**Query 10:**

Fetching the top tweeted text and checking if someone re tweeted it on web series.

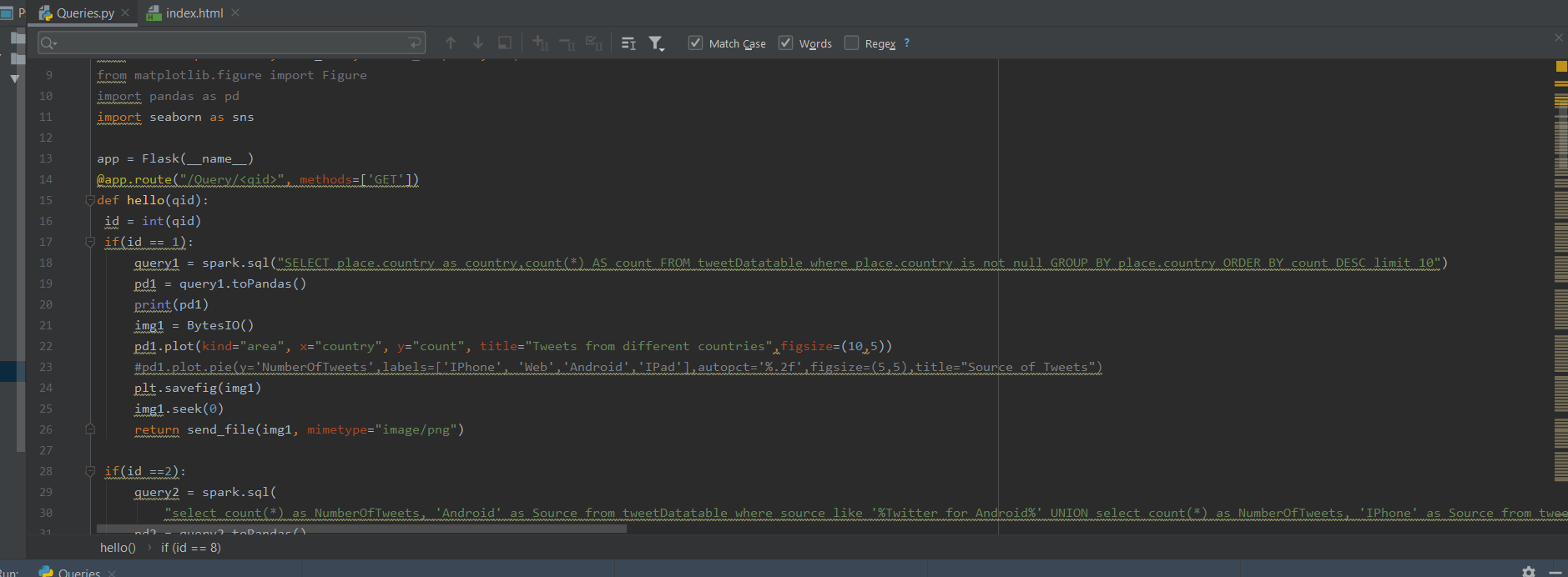
Text

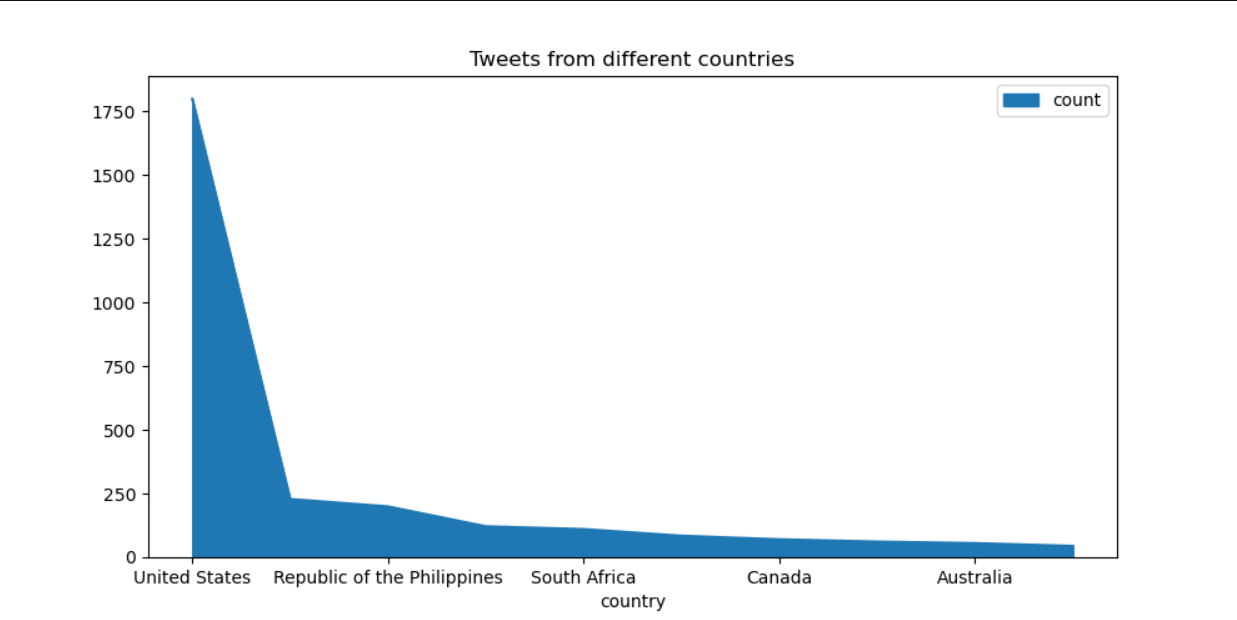
Description automatically generated

**Spark using python: (Considered python programming as the execution time is much faster compared to scala dataframes – Visualization is done based on the python programming with MatplotLib)**

**Query 1:**

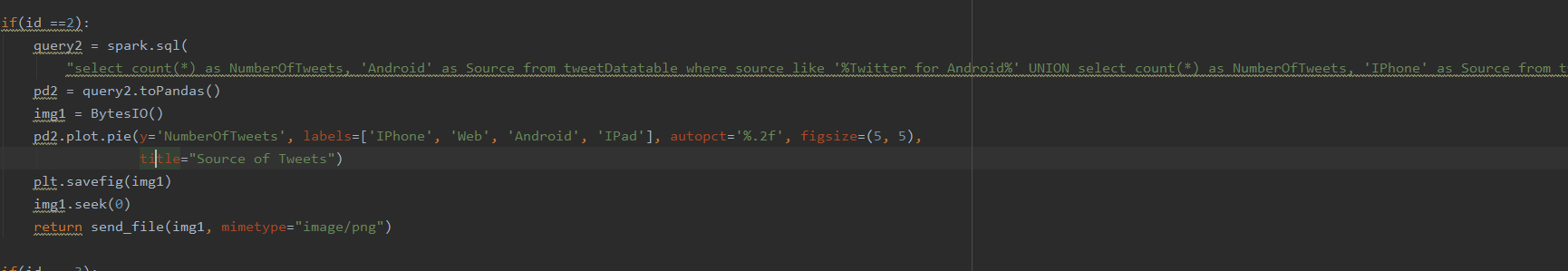
Counted the number of tweets based on the place and arranged in descending order.

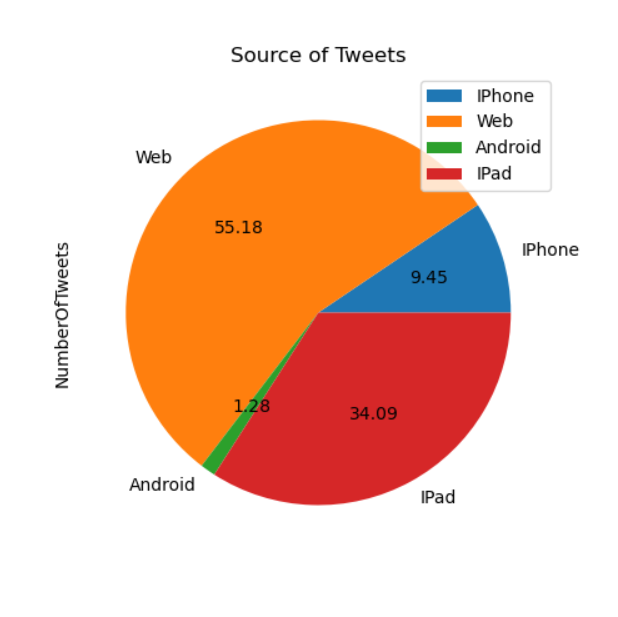




**Query2:**

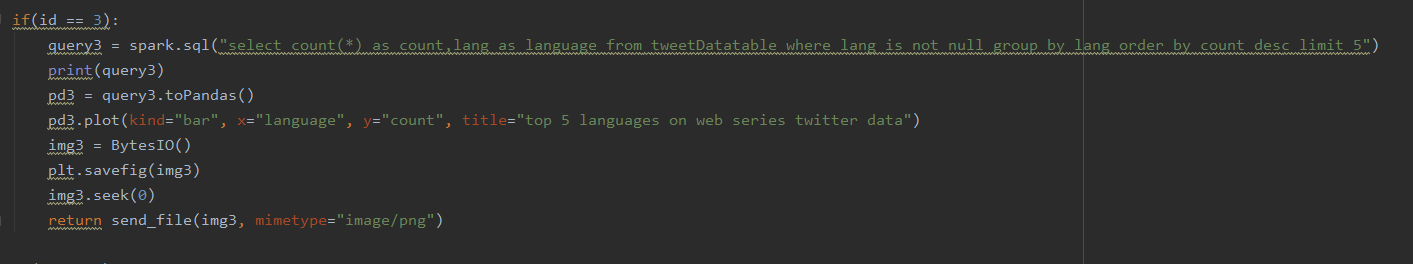
Counted the no.of tweets based on the sources of data by performing the Union operation.

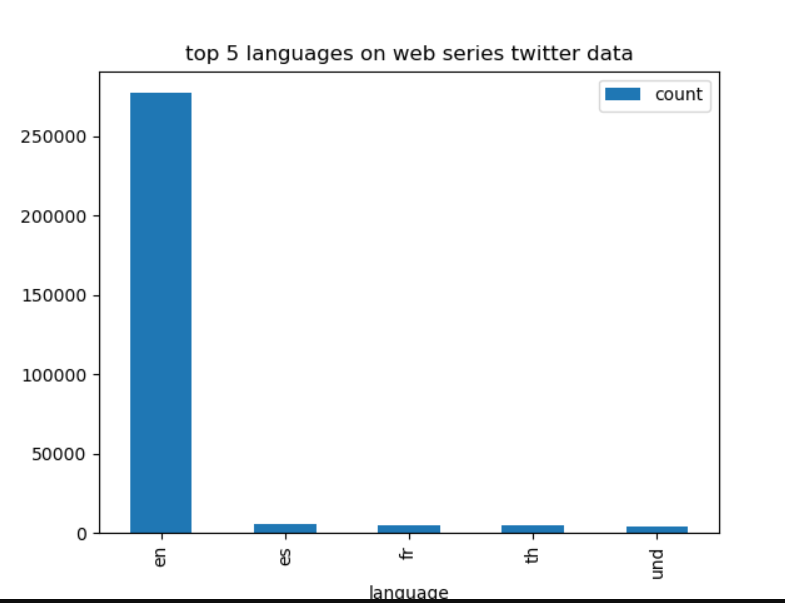




**Query3:**

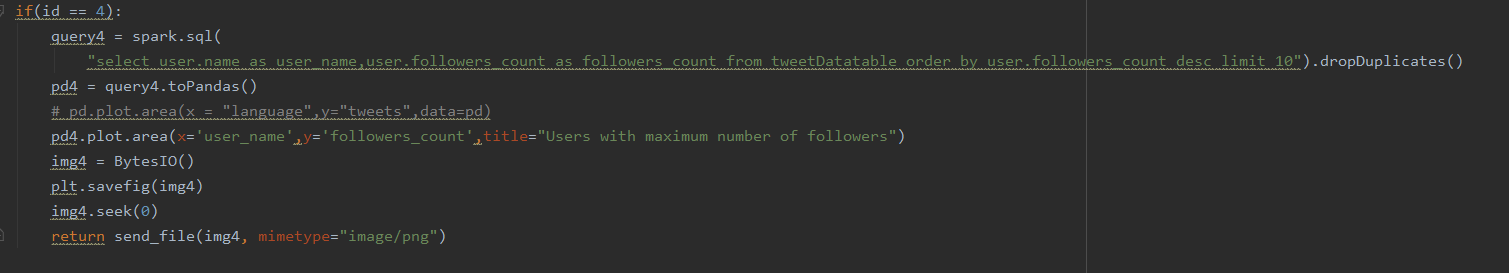
Counted the tweets by grouping into languages and order in descending order with pulling top 5.

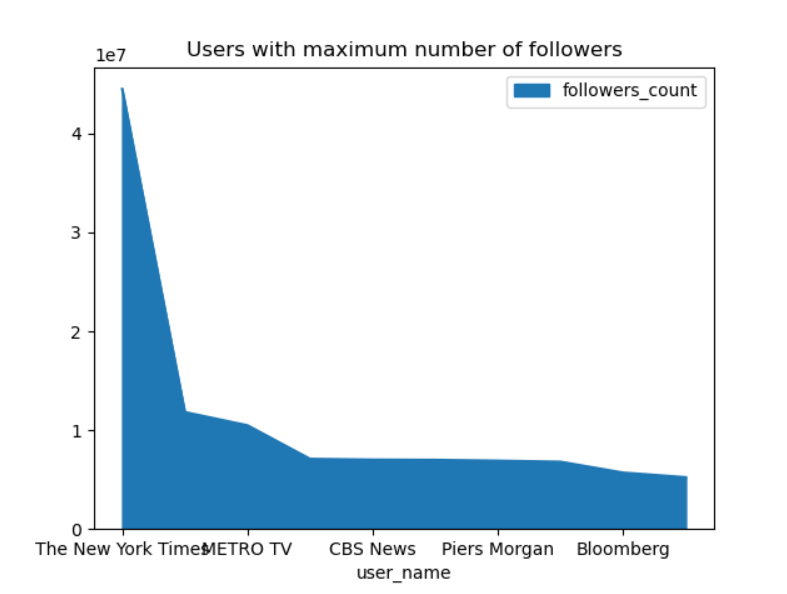




**Query4:**

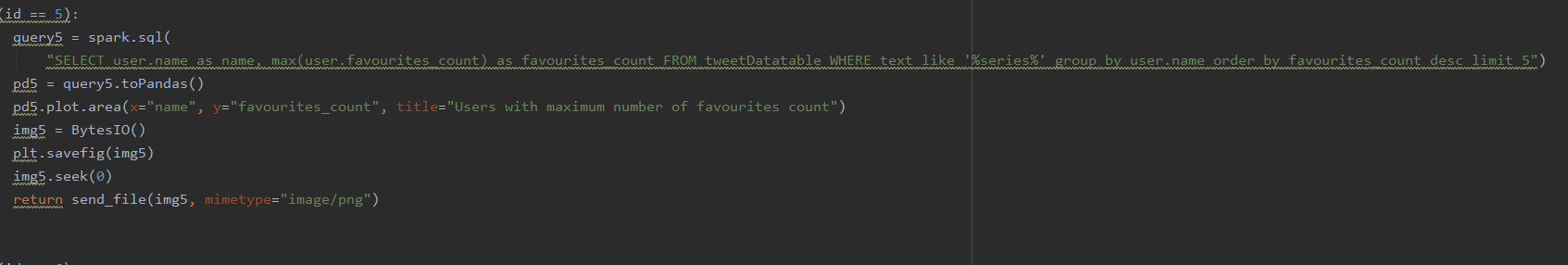
Pulled the top 10 user\_name by followers\_count by dropping the duplicates





**Query5:**

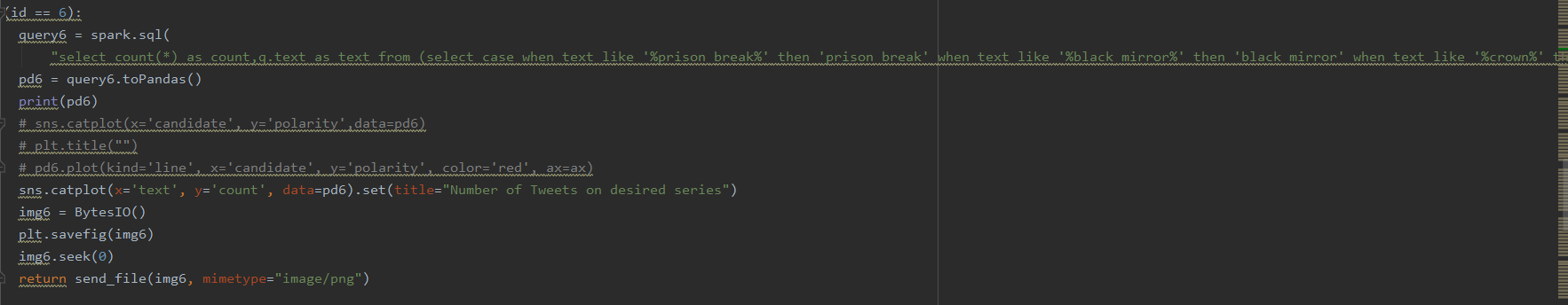
Based on top5 maximum favourites\_count of tweets by user\_name

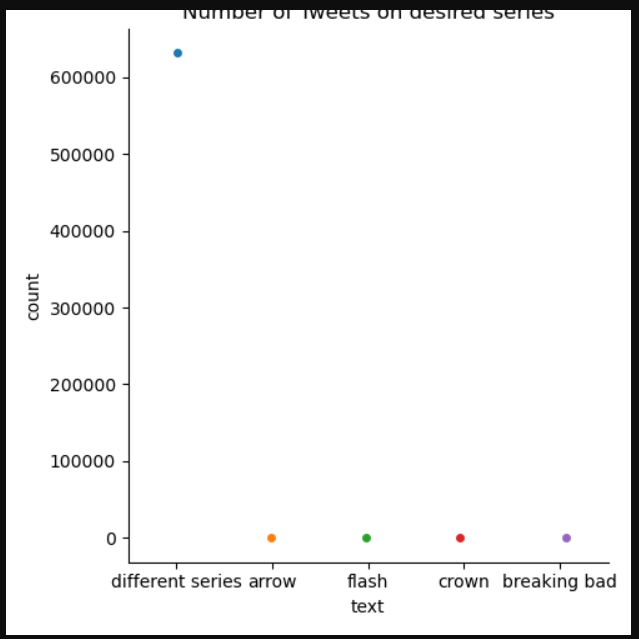




**Query6:**

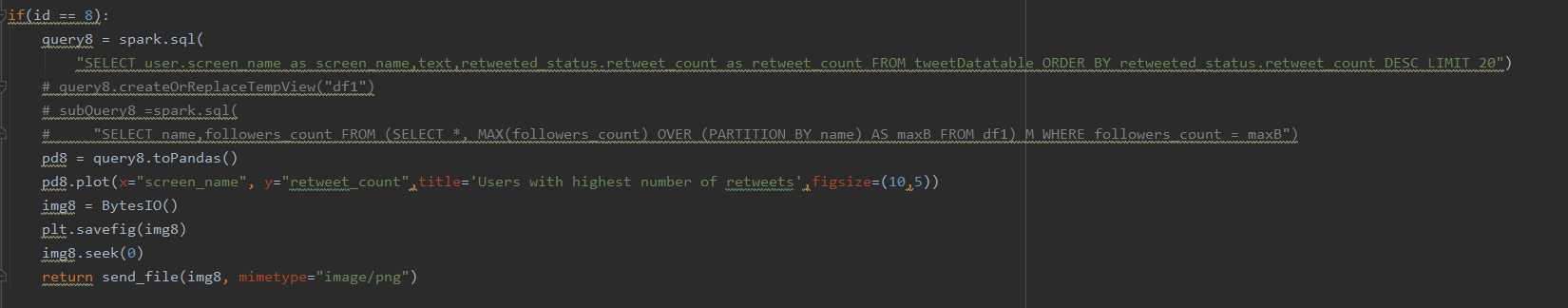
Pulling the count of tweets based on the desired series from the text attribute.

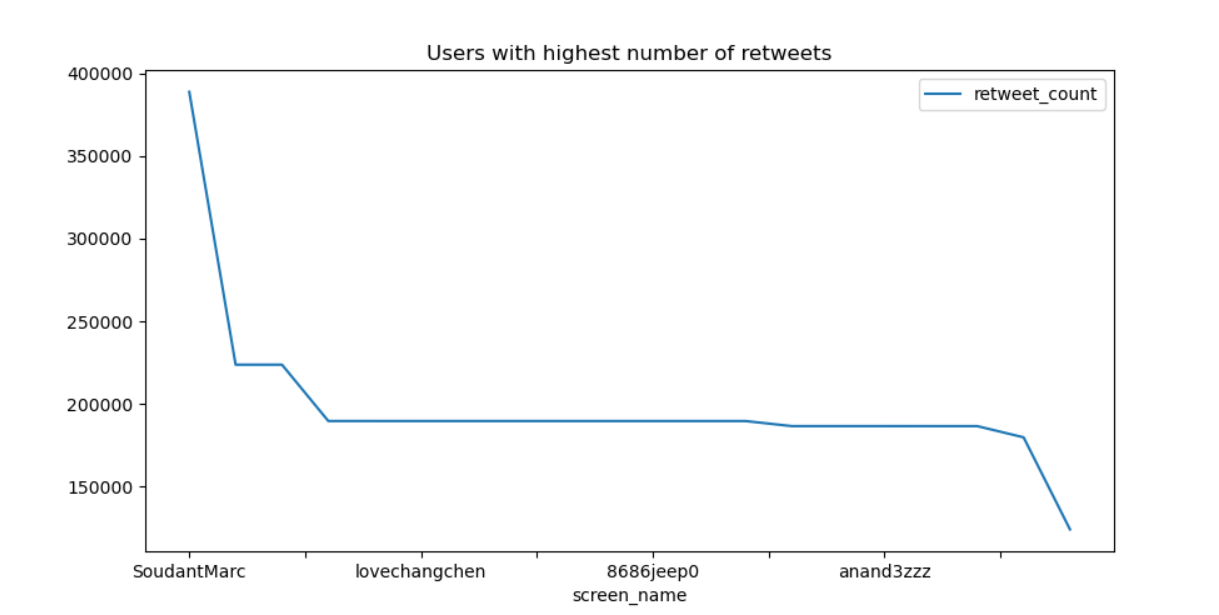




**Query7:**

Ordered the count of tweets of top 20 for the screen name.

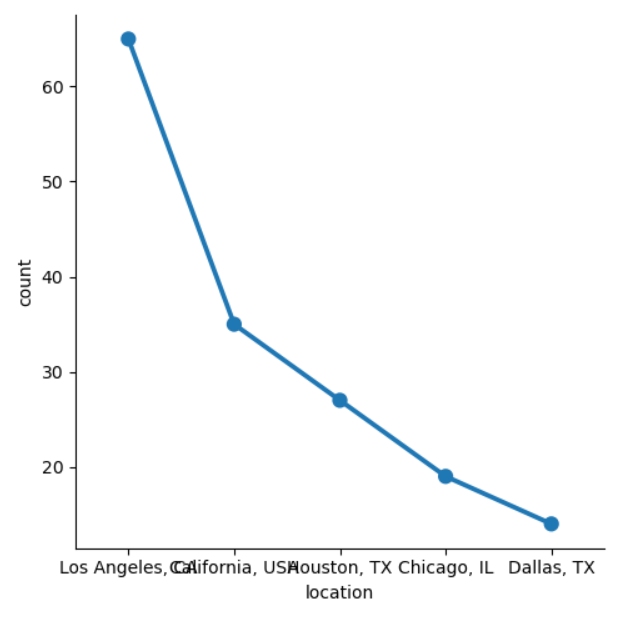




**Query8:**

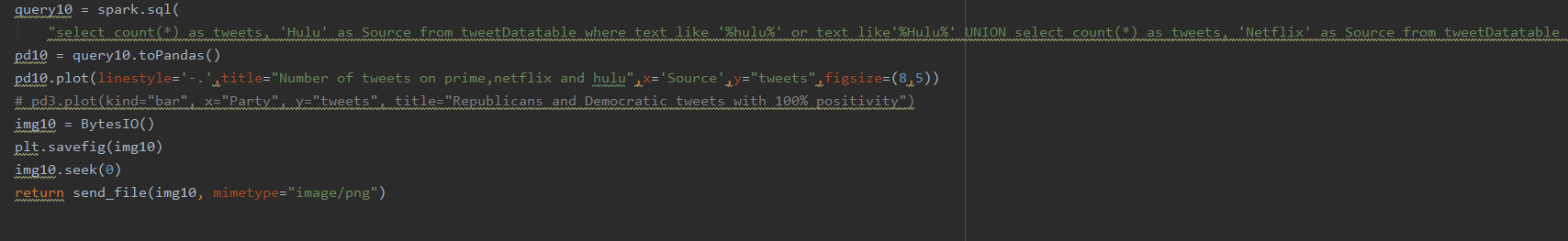
Collected the top5 places in United states.

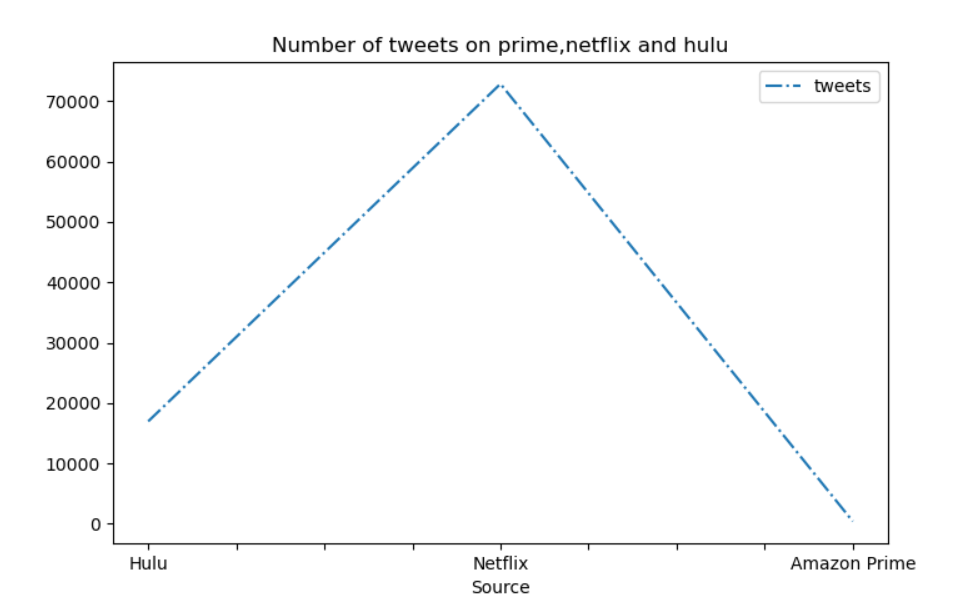




**Query9:**

Count of tweets for Netflix, Hulu, prime





**Solr:**

Started solr with confiruation 6.6 and JDK 8.

Created the core name with samplecore

**Text

Description automatically generated**

Converted the JSON file to CSV using python which makes loading data to solr easy.

Text

Description automatically generated

Later edited the schema.xml with the attributes of csv file.

**Text

Description automatically generated**

Overview after loading the csv data to solr.

**Graphical user interface, text, application, email

Description automatically generated**

**Query1:**

Pulled the 10 records of data in json form.

**Graphical user interface, text

Description automatically generated**

**Query2:**

Pulled the data with user\_location has USA (Regex)

**Graphical user interface, text, application, email

Description automatically generated**

**Query3:**

Pulled the data has Netflix with in text (Regex)

**Graphical user interface, text

Description automatically generated**

**Query4:**

Using fuzzy search of user\_favourites greater than 7000 and text having hulu.

**Graphical user interface, text, application, email

Description automatically generated**

**Query5:**

Collected the response with having text:”prime” and prime:”love”

**Graphical user interface, text, application

Description automatically generated**

**Query 6:**

Proximity query: Pulled the data having user\_name with prince with proximity 3.

**Graphical user interface, text, application

Description automatically generated**

**Query 7:**

Based on created\_at attribute to pull the data before 2019.

**Graphical user interface, text

Description automatically generated**

**Hive**

**Query 1:**

Viewing the number of tweets based on popular streaming service Netflix.

Text

Description automatically generated

**Query 2:**

Viewing the number of tweets based on popular streaming service Hulu.

Text

Description automatically generated

**Query 3:**

Details of tweets regarding web series that have a greater favourites count.

Text

Description automatically generated

**Query 4:**

Details of users and their tweets about web series that has more number of followers count.

Text

Description automatically generated

**GraphFrames**:

Graphical user interface, text, application

Description automatically generated

Displaying the edges of the graph created,

Graphical user interface, application, Word

Description automatically generated

Displaying the vertices of the graph created,

Graphical user interface, text, application

Description automatically generated

The inDegree of the graph is,

Table

Description automatically generated

The outDegree of the graph is,

A picture containing graphical user interface

Description automatically generated

Triangle count for the graph,

Table

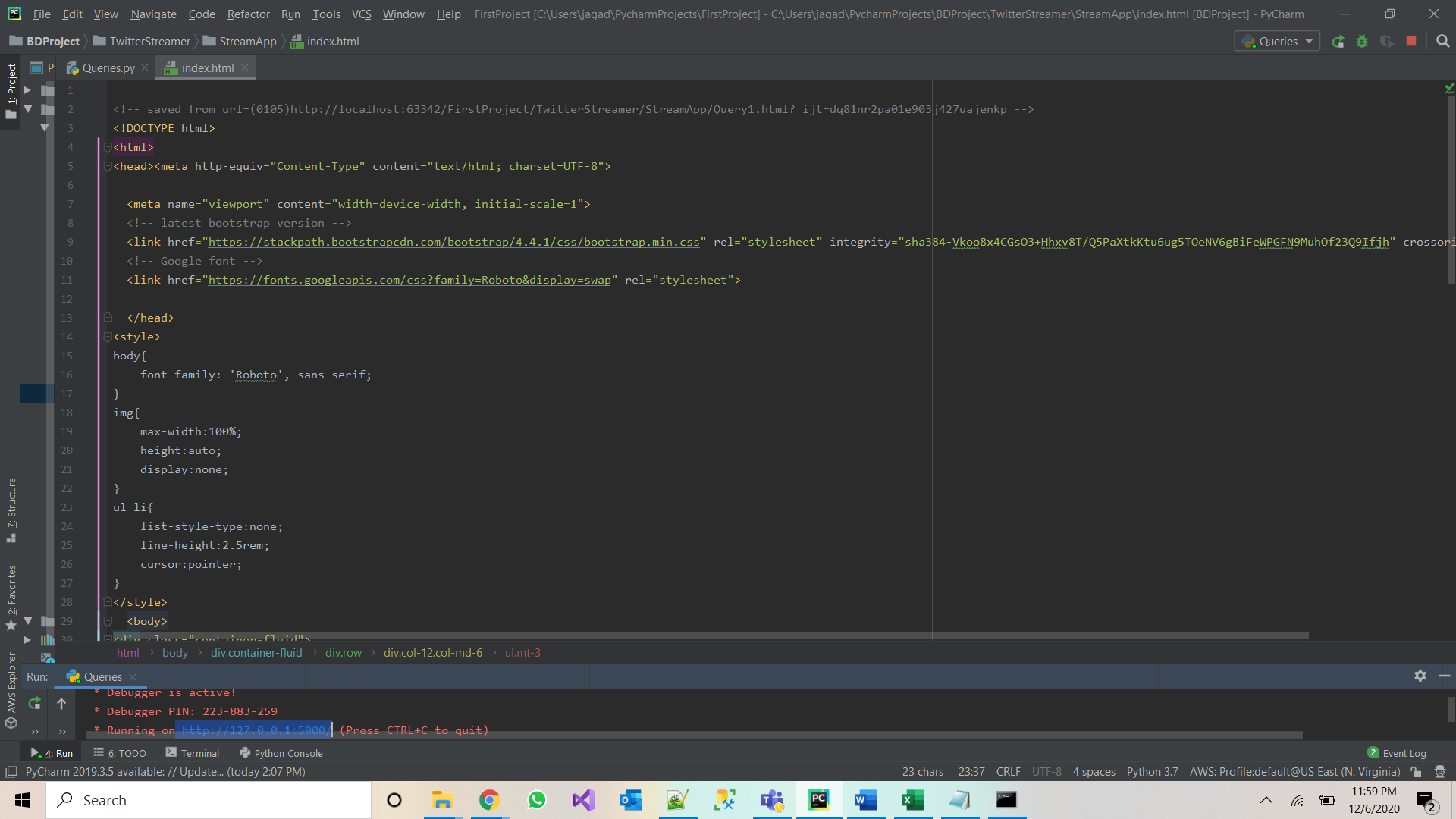
Description automatically generated

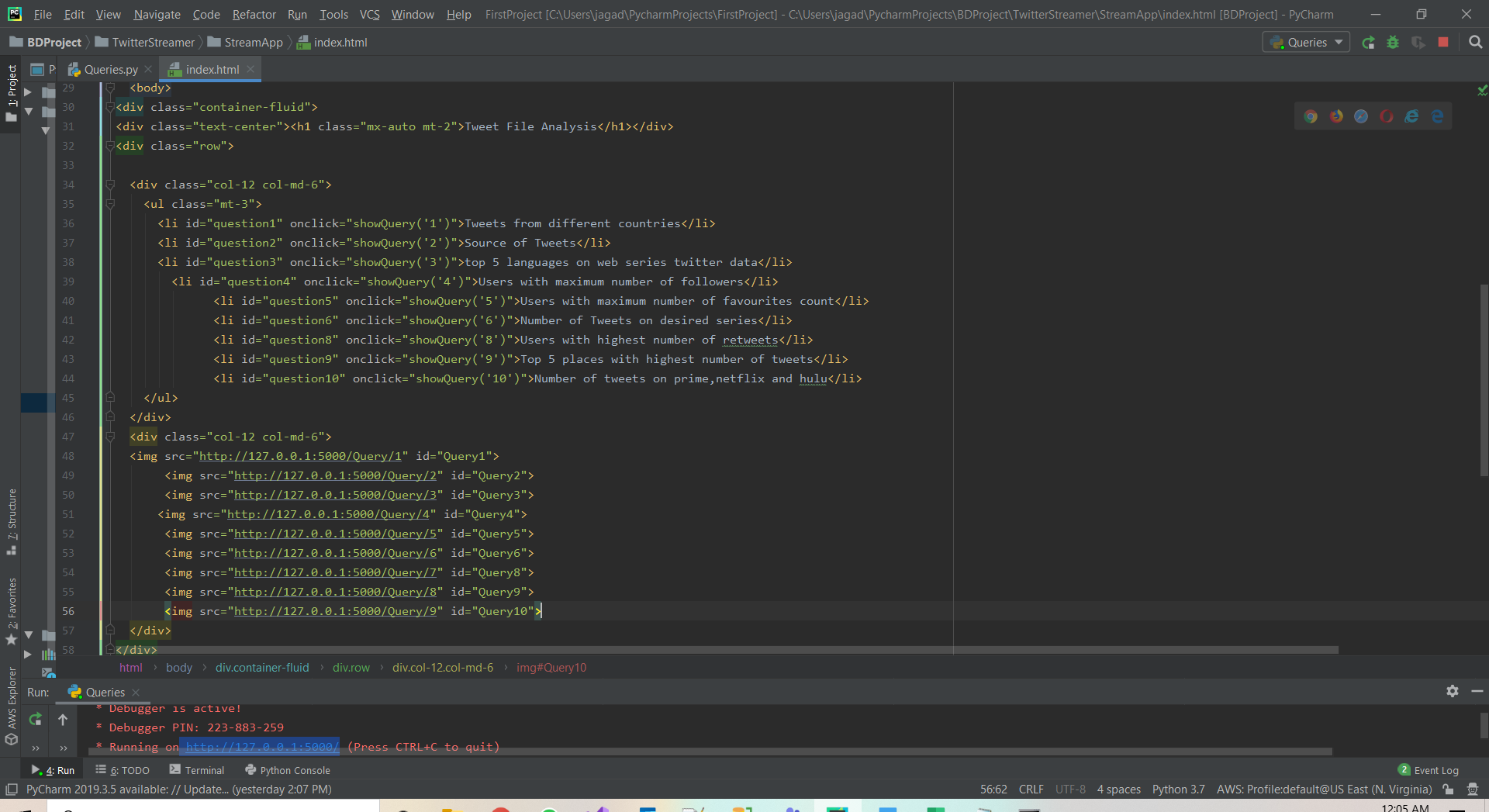
The vertices and edges are saved into different folders in the local system using the above command.

**Docker:**

Created the index.html where we linked all the queries we performed in spark.

Later we hosted on docker.





After creating the queries using spark, we deployed spark visualization into docker.

**Text

Description automatically generated**

Push the latest image on to docker.Text

Description automatically generated

Using ps commands checking the containers. After that run the image on the 5000 port. And created the container.

A picture containing text, monitor, screenshot, screen

Description automatically generated

Bdpproject docker created with tag name latest.

Graphical user interface, text, application, email

Description automatically generated

Hosted the index template on the docker.

**Graphical user interface, application, Word

Description automatically generated**

**Conclusion:**

Done the ETL process using the Spark’s Batch Processing and then Spark Integration using Web UI. Performed our transactions on the set of RDD’s and later we load our data in Hive which is similarly equal to the ETL basic process. This is because we are living in a world where data handling and data using plays one most important role in making the decisions for most of the industries. And at the present scenario where one has leisure time to binge watch series, where this web extraction and data analysis can be helpful.

**Future work:**

* Can use better Machine Learning Algorithm’s such as clustering, sentimental analysis
* Try to stream the live data instead of collecting and parsing previous data from Twitter
* Analyze more data sources such as Facebook, Reddit and not only depending on Twitter
* Migrating to cloud fully without containerizing.

**Project Management**

Done the ETL operations on the dataset and used spark streaming with scala by creating Dataframes and RDD’s with both scala and python programming (For initial increment - scala, for final report – python) it is observed that execution is much better in python compared to scala. Collected the dataset from twitter using tweepy and saved in json format. The tweet dataset is then converted into csv with data preprocessing to use and load it in solr. Applied algorithmic searches such as fuzzy and proximity in solr. And also loaded the data into hive table for query counts using different attributes/features. Used graphframes to find vertices, edges and indegree, outdegree, triangle count.

Finally created an index.html page for hosting it into the Docker container.

**Team Responsibilities for Implementation,**

* Twitter data streaming (Avinash, Geetanjali)
* Data processing (Akhil, Bhashitha)
* Spark streaming using scala (Bhashitha, Avinash)
* Spark streaming using python (Akhil, Geetanjali)
* Convertion of json data to csv (Geetanjali, Bhashitha)
* Solr execution (Avinash, Akhil)
* Hive implementation (Akhil, Geetanjali)
* Graphframes(Bhashitha, Avinash)
* Index web page (Geetanjali,Avinash)
* Dockerization (Akhil,Bhashitha)

**Contributions: (workload was equally divided as a pairs for each task)**

* Avinash Ganguri (25%)
* Akhil Teja Kanugolu (25%)
* Geetanjali Makineni (25%)
* Bhashitha Siddareddy (25%)

**Issues:**

* Jdk version compatability between 15 and 8 for solr
* As json data is large we faced difficult in loading the data to hive and solr
* Converting json to csv is quite hectic because of the text filed
* Faced difficulties with container in Docker.

**STORY TELLING:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **WHO** | **WHAT** | **WHEN** | **WHERE** | **WHY** |
| **CHAPTER 1** | Helps the advertiser’s for pushing the advertisement and series makers for making quick analysis. | Pulling the data helps them in predicting the TRP which in result helps the series makers and advertisers. | As a result of COVID-19 most of the people settled back in their homes lacking entertainment. | All over the world. | Most of the people believe in IMDB rating rather than genre concept. |
| **CHAPTER 2** | The dataset is about the web series data where we sampled the tv shows, web series, Netflix series. The information disclosed about the shows watching around the world. Helps the advertiser’s for pushing the advertisement and series makers for making quick analysis. | The events like which show is being watched more an where it is being watched and which kind of people are watching are seen. It helps advertisers in pulling the data helps them in predicting the TRP which in result helps the series makers and advertisers. | The events take place on how people react to the series they tweet. As a result of COVID-19 most of the people settled back in their homes lacking entertainment. | All over the world. As everyone in the world haves an interest on web series and this data is being used everywhere. Thus, we are working on this data taking this into consideration. | Most of the people believe in IMDB rating rather than genre concept. That is the reason behind doing this web series data collection. |
| **CHAPTER 3**  **(Scientist and AI)** | People who watch web series which make them feel comfortable. | Collected the data using python as json file which we visualized by various platform, Finally, hosted in docker. | Data of 2019 in all over the world collected with various hash tags related to web series of Netflix, prime, Hulu. | After getting the data. Later, Visualized the data using spark, solr, hive. Related to PB and BDP. | Helps to understanding the data by visualizing the data stream pulled from twitter. |
| **CHAPTER 4**  **(Users)** | The set of people who are having free time and interested to watch the series can access the info. | Helps to Visualize the Data or impact of series on the current society based on various features/attributes. | Based on the Data available on the hosted index, user can get the contrast between the OTT platforms and the web series and their pulse among. | Deployed to docker and can host on EC2 instance. | Make’s aware of types of OTT platforms impact on the society. |
| **CHAPTER 5**  **(Society)** | Users who are mostly spends their time on watching online series. Types of OTT platforms are sampled. No data is sampled. | This data which makes aware of the current situation about the OTT platforms, which mostly doesn’t have social or cultural impacts but only have idea about the OTT platforms. As the data grabbed from the twitter streaming will have security reasons. | If the tweets pulled based on the hashtags which web series related to region, caste, cultural can create positive or negative impact on the social or cultural but mostly it creates the negative impact which may breach the security. | Impact will be on the users who mostly rely on web series. If the Data collected concentrated on the particular set of webseries can cause the evaluative bias. | At current pandemic situation it will be consequential to people who are misled by the various paid platforms which give the review of the series. |

**References:**

1. [developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json](https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json)
2. [www.info.com/**Twitter Data Visualization**](https://www.info.com/serp?q=twitter%20data%20visualization&segment=info.0376&s1aid=144007235&s1cid=357227086&s1agid=1264438724330452&s1kid=kwd-79027647541819:loc-190&utm_source=adcenter&msclkid=fa5513571bee1e925c7b4c24b6d1fd75)
3. [spark.apache.org/docs/latest/](https://spark.apache.org/docs/latest/streaming-programming-guide.html)
4. <https://www.youtube.com/watch?v=lWzcZZEQYMk>
5. https://docs.docker.com/docker-hub/publish/customer\_faq/