Twitter Big Data analysis on Drones

By

Shweta Parihar, Priyanka Bala, Komali Ghanta

INTRODUCTION

In this project we have collected tweets data which we found impressively much more than just text and a user name. People usually do not realize how much data they generate from posting, retweeting etc. Each action on the twitter are stores in a very organized manner in JSON format. This project we collected tweets related to all types of drones, loaded the downloaded tweets in HDFS and analyzed the data using hive and tried to visualize the data using Highcharts bar charts.

Steps Implemented

1. **Download Twitter Data.**

   a. Created a Twitter application on the Twitter Application Management Portal and generated the authorization keys. These were used by the Java application to connect to Twitter.
   b. A java application was created that used Twitter Streaming API to download the RAW JSON data associated with the tweets that were selected from the Twitter's stream based on the keywords provided in the application.
   c. Successfully downloaded 890,000 Tweets approximately 2.9 GB of RAW JSON data.

2. **Building Hive Tables**

   1. Utilized jq tools to analyze the RAW JSON Structures and created Strings for creating smaller JSON files that contained only specific attributes needed. For example one of the JSON that we created only contained the user id and the https url of the profile picture. We used these string with the combination of Linux command sed and cat to create smaller JSON files. Below is the example of the same.

jq String and linux command for User_profile_picture table

{user_id: .user.id,profile_image_url_https: .user.profile_image_url_https}

cat DronesV2RAW.json | ./jq -c -M '{tweet_id: .id,user_id: .user.id}' | sed 's/\\"/\\\\"/g'> /home/biadmin/twitter/User_profile_picture.json

2. Created table using JSON Serde: Utilized JSON Serde with the combination of Hive Create table and Load commands to create of the four table. Serde was used as it helps load the data of the JSON directly in to the tables. Below are the command executed

```
Create table JQTweets_user_r (tweet_id bigint,user_id bigint)
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.JsonSerde'
STORED AS TEXTFILE;

LOAD DATA LOCAL INPATH '/home/biadmin/twitter/tweet_user_r.json' OVERWRITE INTO
TABLE  JQTweets_user_r;

CREATE TABLE JQTweets (tweet_id bigint,text string,possibly_sensitive boolean,lang
string,retweeted boolean,retweet_count int,favorited boolean,favorite_count
int,created_at string,geo string,id_str string,timestamp_ms bigint)
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.JsonSerde'
STORED AS TEXTFILE;

LOAD DATA LOCAL INPATH '/home/biadmin/twitter/tweet.json' OVERWRITE INTO TABLE
JQTweets;

CREATE External TABLE JQUser (user_id bigint,location string,lang string,favourites_count
int,verified boolean,contributors_enabled boolean,name boolean,created_at
string,followers_count int,geo_enabled boolean,utc_offset string,time_zone
string,friends_count int,screen_name string,is_translator boolean)
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.JsonSerde'
STORED AS TEXTFILE;

LOAD DATA LOCAL INPATH '/home/biadmin/twitter/user.json' OVERWRITE INTO TABLE
JQUser;

Create table User_profile_picture (user_id bigint,profile_image_url_https string)
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.JsonSerde'
STORED AS TEXTFILE;

LOAD DATA LOCAL INPATH '/home/biadmin/twitter/User_profile_picture.json'
OVERWRITE INTO TABLE  User_profile_picture;
```

3. Analyzed Drone tweets using hive queries
   Created five hive queries to analyze twitter drone data using keywords like join, group by, case, count etc... We have created these queries keeping in mind that the output will provide interesting observations about drones.

1. Query 1: Real time Drone application analysis

   Upon researching various news articles and web content it was decided to consider the following real world drone application

   - Defense
   - Toys
   - Agriculture
   - Photography
   - Product Delivery
   - Medical
   - Internet

   We considered keywords for each application and wrote the query to check how many tweets per applications was found in dataset.

```
select
count(case when text like "%police%" or text like "%protesters%" or text like "%drone strike%" or text like "%drone strikes%" or text like "%drone strike%" or text like "%drone strikes%" then 1 end),
count(case when text like "%rc drone%" or text like "%Parrot%" or text like "%AR Drone%" or text like "%skyviper%" or text like "%sky_viper%" or text like "%sky viper%" or text like "%syma%" or text like "%SYMA%" or text like "%DJI%"or text like "%DJI Phantom%" then 1 end),
count(case when text like "%agri%" or text like "%agribotix%" or text like "%air dog%" or text like "%airdog%" or text like "%sensefly%" or text like "%spreading wings%" or text like "%spreadingwings%" then 1 end),
count(case when text like "%DjiPhantom 2 Vision+%" or text like "%Phantom 2 vision+%"or text like "%video%shoot%FPS%"or
text like "%shoot%video%FPS%"or text like "%shoot%video%" then 1 end),
count(case when text like "%Amazon%" or text like "%parcelcopter%"or text like "%pizza delivery%"or text like "%francesco%" then 1 end),
count(case when text like "%Blood Delivery%" or text like "%Mayo Clinic%"or text like "%difibrillator%"or text like "%Ambulance copter%"or text like "%ambulancecopter%" then 1 end),
count(case when text like "%Facebook%" or text like "%internet.org%"or text like "%facebook%"or text like "%facebook drone%"
or text like "%Aquila%" then 1 end)
from JQTweets ;
```

2. Query 2: Among the top featured Drone companies (Manufactures/retailers) which one is the most famous in twitter world.

   Upon researching various news articles and web content it was decided to consider the following manufactures making drones:

- DJI
- Facebook
- Amazon
- Parrot
- Blade
- Swann
- Troy
- Hubsan
- Lego
- Skyrocket
- Aee
- Yuneec
- Xfreem
- Airdog
- Agribotix

We considered keywords for each manufacturer and wrote the query to check how many tweets per manufacturer was found in dataset.

```
select
 count(case when text like "%DJI%" or text like "%dji%" or text like "%Dji%" then 1 end),
 count(case when text like "%Facebook drone%" or text like "% Facebookdrone %" or text like "% facebook drone %" or text like "% facebookdrones%" or text like "% facebookdrone %" then 1 end),
 count(case when text like "%Amazon drones%" or text like "%Amazondrones%" or text like "%amazon drones%" or text like "%amazondrones%" then 1 end),
 count(case when text like "% Parrot%" or text like "%parrot%" then 1 end),
 count(case when text like "% Blade%" or text like "%blade%" then 1 end),
 count(case when text like "% Swann%" or text like "%swann%" then 1 end),
 count(case when text like "% Troy%" or text like "%troy%" then 1 end),
 count(case when text like "% Hubson%" or text like "%hubson%" then 1 end),
 count(case when text like "% Lego%" or text like "%lego%" then 1 end),
 count(case when text like "% SkyRocket%" or text like "%skyroket%" then 1 end),
 count(case when text like "% Yuneec%" or text like "%yuneec%" then 1 end),
 count(case when text like "% Aee%" or text like "%aee%" then 1 end),
 count(case when text like "% Xtreem%" or text like "%xtreem%" then 1 end),
 count(case when text like "% Agribotix %" or text like "% agribotix %" then 1 end),
 count(case when text like "% Airdog %" or text like "% airdog %" then 1 end)
from JQTweets ;
```

3. Query 3: Analyzed 10 most used languages to tweet about drones.
   This query was written to get top 10 languages that was used to tweet about drones.

```
select lang,count(lang) as lang_no from JQTweets group by lang order by lang_no desc limit 10
```

4. Query 4: Analyzed users who have tweeted most about drones.
   This query helps us calculate the user who has the highest number of tweets about drones.

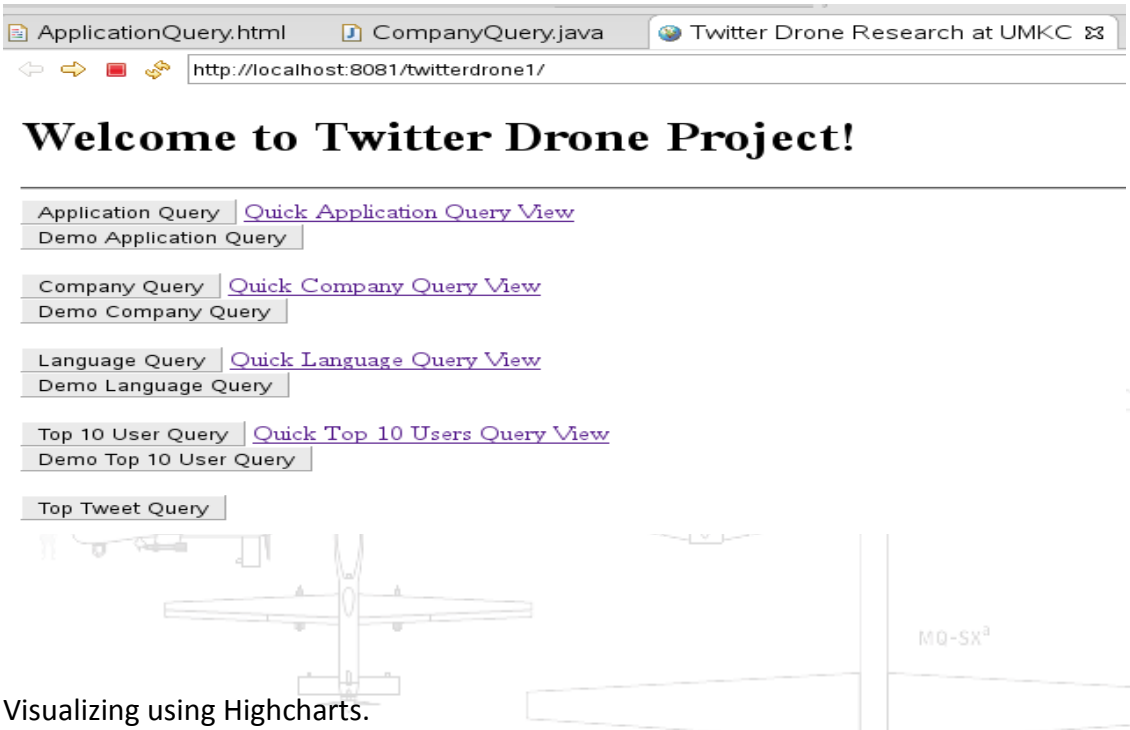
```
select u.screen_name,count(distinct r.tweet_id) as tweet_count from JQUser u join JQTweets_user_r r on u.user_id=r.user_id group by u.screen_name order by tweet_count desc limit 10;
```

5. Query 5: Most Popular tweet
   This query helps us to calculate the most popular tweet using the favorite and retweet count.

```
Select .text,u.screen_name,i.profile_image_url_https,(t.retweet_count+t.favorite_count) as sum from JQTweets_demo t join JQTweets_user_r_demo r on t.tweet_id=r.tweet_id join JQUser_demo u on r.user_id=u.user_id join user_profile_picture_demo I on u.user_id=i.user_id order by sum desc limit 1;
```

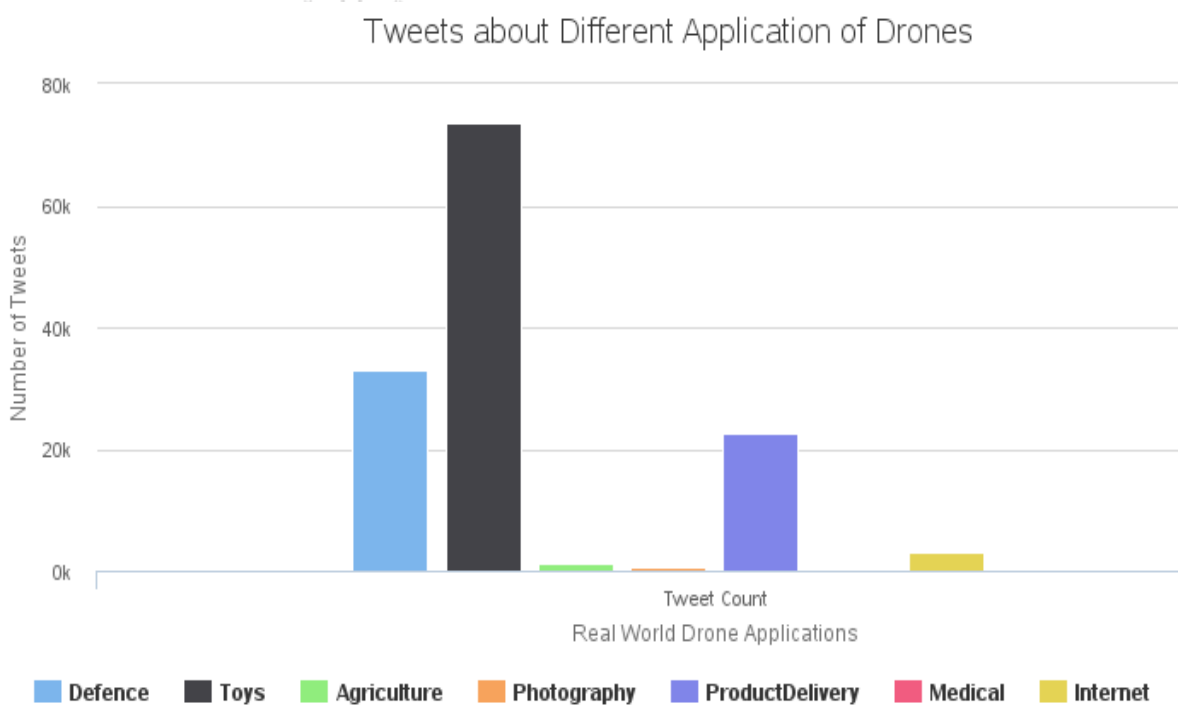
4. Implementing the above queries in au.screen_ web application to get output.

- A web application was created using JSP or Dynamic web pages on eclipse and Highchart API. The purpose of this web application is to give user a UI using which the user can execute the above stated analytical queries without writing them.
- The Home page of the application gives a button for each query and also contain link for quick result review.
- When user clicks any of the query buttons it will call the Java servlet. Java servlet upon receiving the request call the doGet method. This method was implemented to connect with HDFS, execute the corresponding query and getting the results. After Servlet receives the result it creates a CSV file containing the result and redirects to a HTML page that reads the csv file and create an interactive chart for the user to visualize the results.

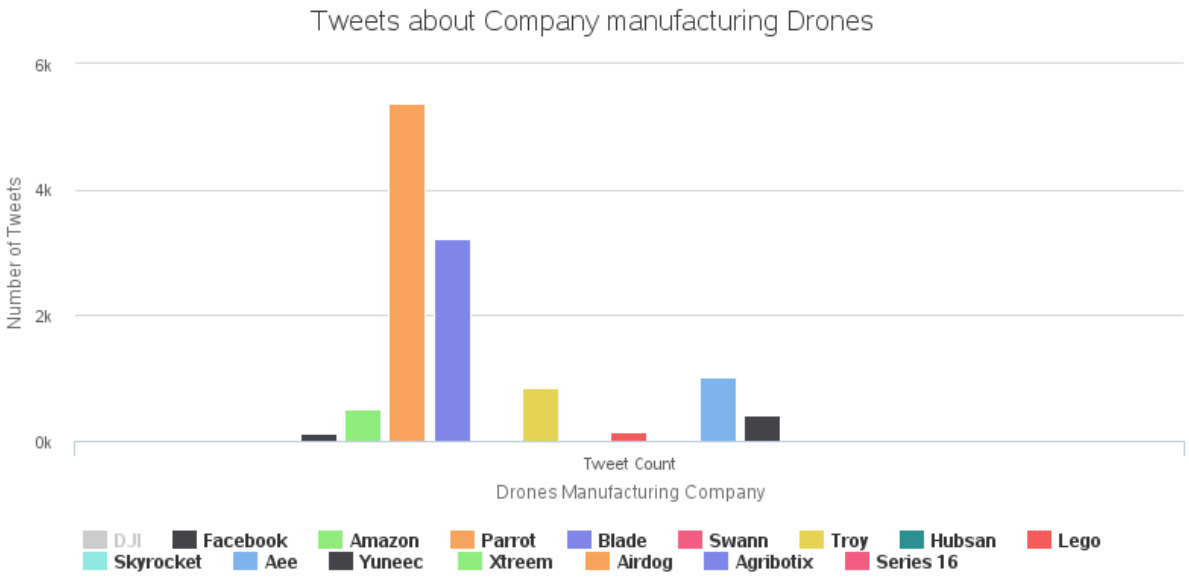


5. Visualizing using Highcharts.

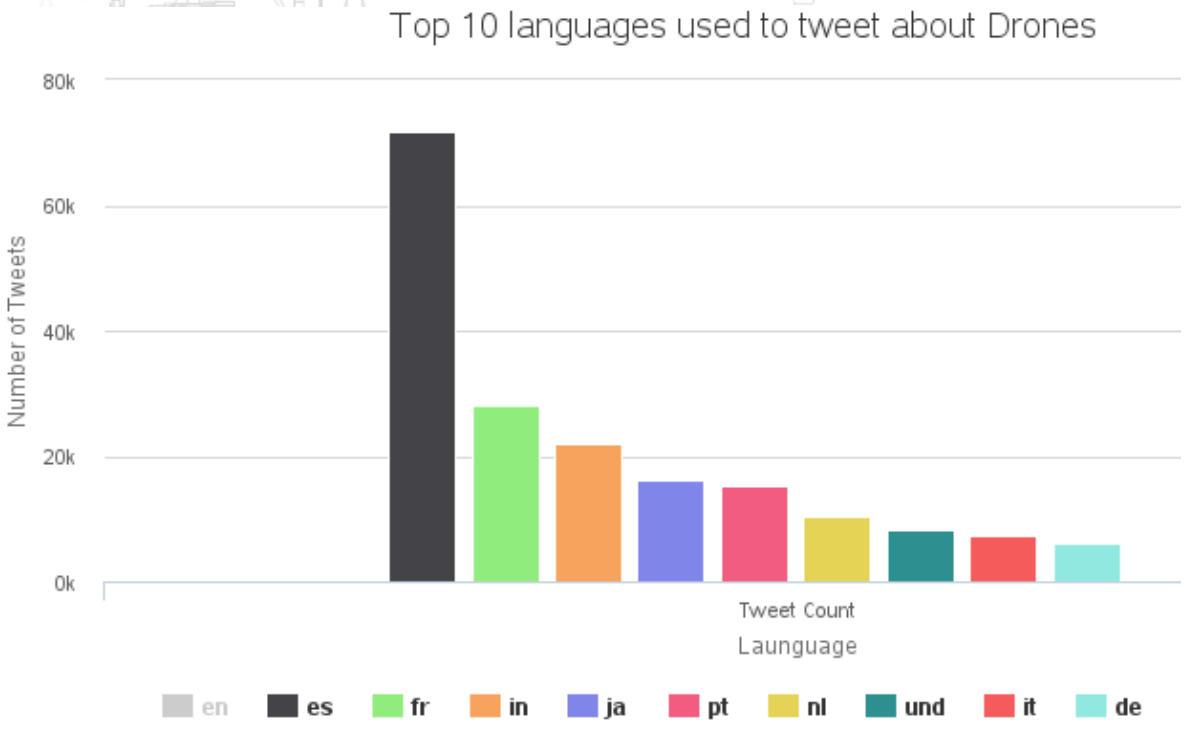
Query 1 Visualization



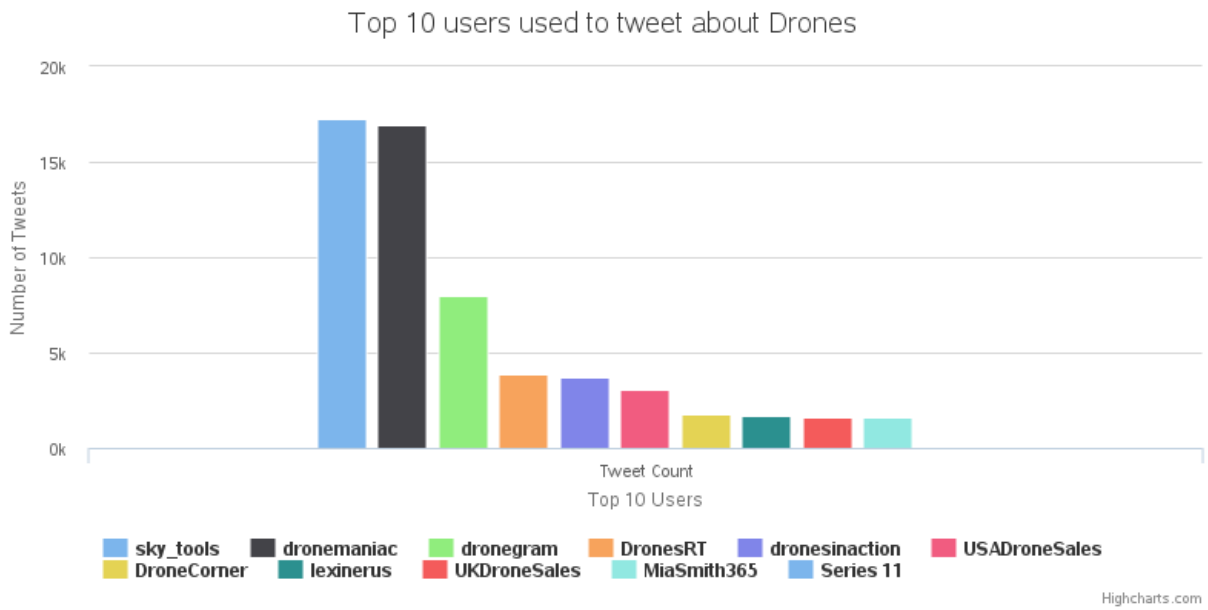
Query 2 Visualization



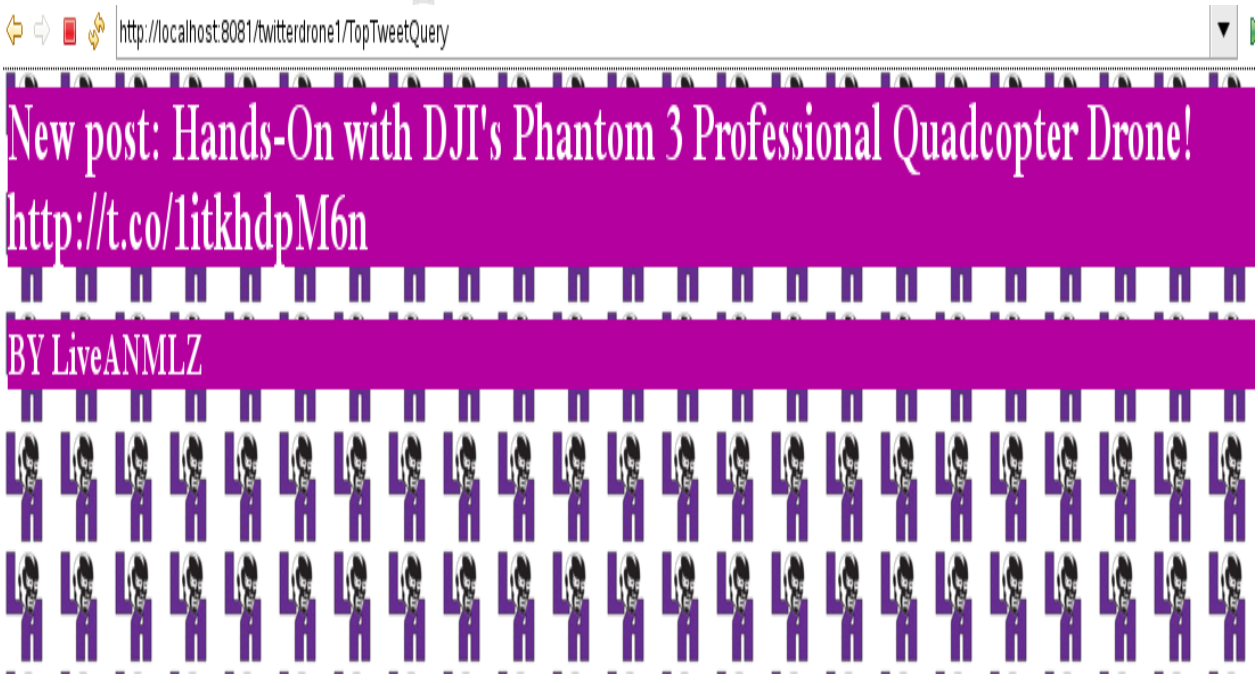
Query 3 Visualization



Query 4 Visualization



Query 4 Visualization



6. Testing

Testing the output of the different steps along this project have been very important.

a. The java application that was created to download the twitter data was tested by testing the JSON output after every few minutes initially with the help of http://jsonlint.com/. JSONLINT is an online JSON validator.

b. The output of the jq command was also tested using JSON lint to make sure that small JSON file were properly built.

c. As the small JSON files that were created using JQ and RAW JSON Twitter file they should have the same number of JSON lines. We made sure that all JSON files has same number of line.

d. After creating the tables we ran the select count (*) from <Table Name> query for each table to see that all of them should have same amount of rows.

e. All analytical queries were ran on HIVE shell and web application to check if they have the same results.

Tools, APIs and Technologies used

1. Twitter Stream API
2. Jq
3. IBM Biginsight
4. JSON lint
5. Eclipse
6. Java
7. Highchart

References:

1. http://twitter4j.org/en/index.html
2. https://jqplay.org/
3. http://jsonlint.com/
4. http://www.highcharts.com/