**Phase2 – Principles of Big Data Project Report**

**Team Members**

1. Pavan Kumar Chongala
2. Gowtham varma vegesna

3. Goutham Gandreddi

**Big Data** is a field that treats ways to analyze, systematically extract information from, or otherwise deal with data sets that are too large or complex to be dealt with by traditional data-processing application software.

**Softwares Used**

* Apache Spark
* Scala (Execution of Queries)
* Zepplin for Visualizations
* High Charts for Visualizations
* HTML, CSS, Javascript for front end UI
* Twitter4J Library for Tweets Extraction
* virtual machine
* pycharm python3

**Hardwares Used**

* Windows Machine
* Intel i5 Processor
* 8GB RAM

**Implementation**

1. Written a Java Program using Twitter 4J Library to extract Streaming Twitter JSON data of Tweets (800 MB) for the trending topic on an Indian Premier League cricket match.

2. The extracted JSON tweets are persisted into the Apache Spark in the form of tables/views.

3. Created a web application with buttons and on the click of each button, a query written in Scala language will be sent to spark server and the outputs files are stored in the form of CSV/JSON files.

4. These CSV/JSON output files are used to visualize the data using Zepplin, Highcharts & Amcharts to get the Bar, Pie, Line, Donut Graphs etc..

**Concept – Indian Premier League (IPL2k19)**

The Indian Premier League (IPL) is a professional Twenty20 cricket league in India contested during March or April and May of every year by eight teams representing eight different cities in India.The league was founded by the Board of Control for Cricket in India (BCCI) in 2008.

The IPL is the most-attended cricket league in the world and in 2014 ranked sixth by average attendance among all sports leagues.[5] In 2010, the IPL became the first sporting event in the world to be broadcast live on YouTube. The brand value of IPL in 2018 was US$6.3 billion.

**Data Analytics on IPL Cricket Match using Twitter Data**

**Query 1: IPL Trending Tweets from Top 10 Geo Locations**

val geoloc = spark.sql("SELECT user.name,user.screen\_name,user.id,geo.coordinates[0] As latt,geo.coordinates[1] As long from table where geo is not null")

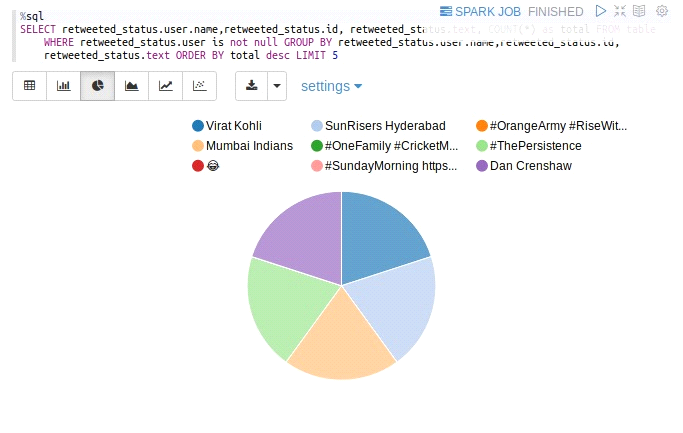
This is used to get the top 10 geo locations and show it on the Google Maps from where the trending has happened.



**Query 2: IPL tweets**

scala> val retwname = sqlContext.sql("SELECT retweeted\_status.user.name,retweeted\_status.id, retweeted\_status.text, COUNT(\*) as total FROM table WHERE retweeted\_status.user is not null GROUP BY retweeted\_status.user.name,retweeted\_status.id, retweeted\_status.text ORDER BY total desc LIMIT 5")

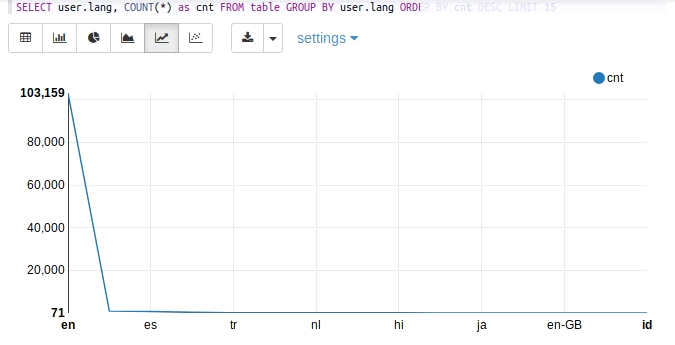
This query is used to see the top retweets and their values in the form of circle shape.



**Query3: IPL Trending Tweets from Different Languages**

scala> var language = spark.sql("SELECT user.lang, COUNT(\*) as cnt FROM table GROUP BY user.lang ORDER BY cnt DESC LIMIT 25");

This is used to know how many tweets related to cricket have been trended from different types of languages.

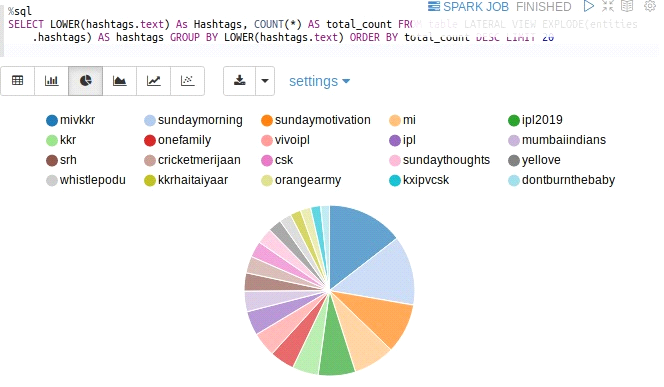


en = english is the mostly used language on twitter for ipl

**Query4: Top Trended Cricket Match hashtags**

scala> val hashtags = spark.sql("SELECT LOWER(hashtags.text) As Hashtags, COUNT(\*) AS total\_count FROM table LATERAL VIEW EXPLODE(entities.hashtags) AS hashtags GROUP BY LOWER(hashtags.text) ORDER BY total\_count DESC LIMIT 20")

This query is used to know how many hashtags related to cricket has been trended through worldwide.



**Query5: the most power team tweets**

%sql

select cause,count(\*)

from

(

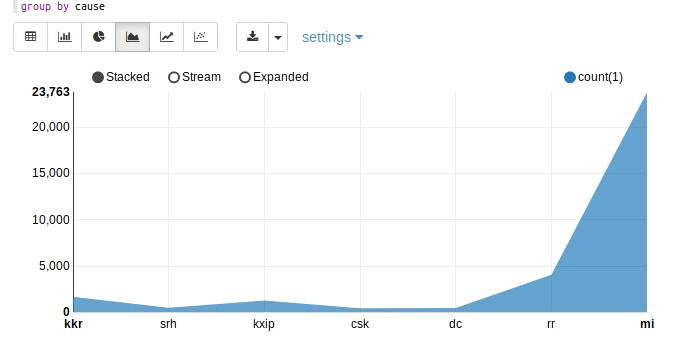
select regexp\_extract(text,'srh|csk|kxp|mi|dc|rr|kkr|rcb fuels',0) as cause from table

)

where nullif(cause,"") is not null

group by cause

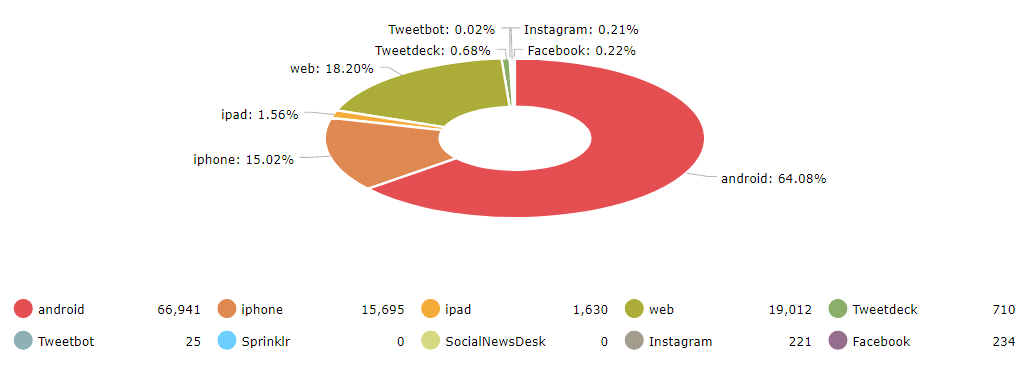
This query is used to know how many tweets have been generated for a timestamp. We have showed the top trends for each time division.



**Query6: Analysis on Twitter Sources**

scala> val sources = spark.sql("SELECT sum(case when lower(source) like '%android%' then 1 else 0 end)android, sum(case when lower(source) like '%iphone%' then 1 else 0 end)iphone, sum(case when lower(source) like '%ipad%' then 1 else 0 end)ipad, sum(case when lower(source) like '%web%' then 1 else 0 end)web, sum(case when lower(source) like '%tweetdeck%' then 1 else 0 end)Tweetdeck, sum(case when lower(source) like '%tweetbot%' then 1 else 0 end)Tweetbot, sum(case when lower(source) like '%www.sprinklr.com%' then 1 else 0 end)Sprinklr, sum(case when lower(source) like '%www.socialnewsdesk.com%' then 1 else 0 end)SocialNewsDesk, sum(case when lower(source) like '%instagram.com%' then 1 else 0 end)Instagram, sum(case when lower(source) like '%facebook.com%' then 1 else 0 end)Facebook from table")

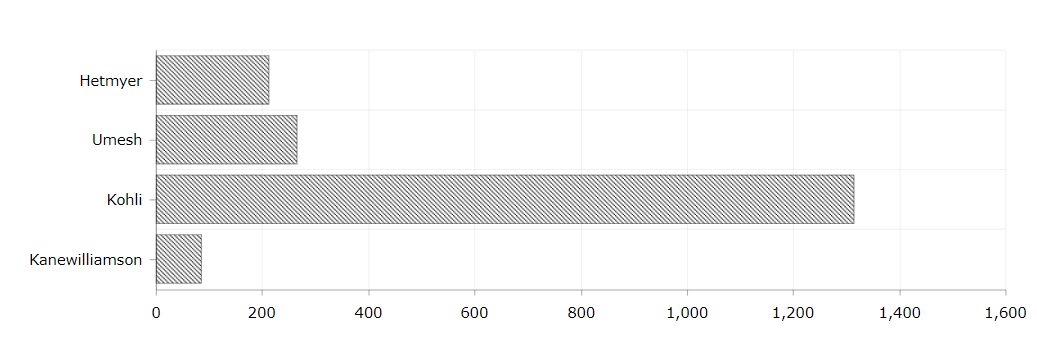
This query is used to know the number of tweets that have been generated from different platforms like iphone, android, web, etc…



**Query7: Analysis on Cricket Players**

scala> var players = spark.sql("SELECT SUM(case when text like '%Hetmyer%' then 1 else 0 end)Hetmyer,SUM(case when text like '%Umesh%' then 1 else 0 end)Umesh,SUM(case when text like '%Kohli%' then 1 else 0 end)Kohli,SUM(case when text like '%IPL2019%' then 1 else 0 end)IPL2019,SUM(case when text like '%Kane%' then 1 else 0 end)KaneWilliamson from table")

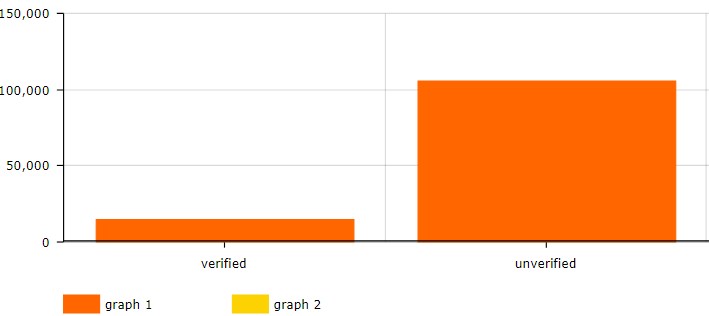
This query is used to know the top cricket players that has been trended through out the match.



**Query8: IPL Tweets from verified Users**

scala> var status = spark.sql("SELECT SUM(case when user.verified=true then 1 else 0 end)Verified,SUM(case when user.verified=false then 1 else 0 end)Unverified from table")

This query is used to know the tweets trending from verified/unverified users



**Query9:**

scala> val temp= spark.sql("SELECT substring(user.created\_at,5,3) as month, count(user.id) from table group by month");

This query is used to know how many users have tweeted the tweets the most



**Query 10:Tweets from Number of countries**

SELECT case when place.country='India' then 'in'

when place.country='United States' then 'us'

when place.country='United Kingdom' then 'gb'

when place.country='Brasil' then 'br'

when place.country='South Africa' then 'za'

when place.country='Australia' then 'au'

when place.country='Italia' then 'it'

when place.country='France' then 'fr'

when place.country='Nigeria' then 'ne'

when place.country='Canada' then 'ca'

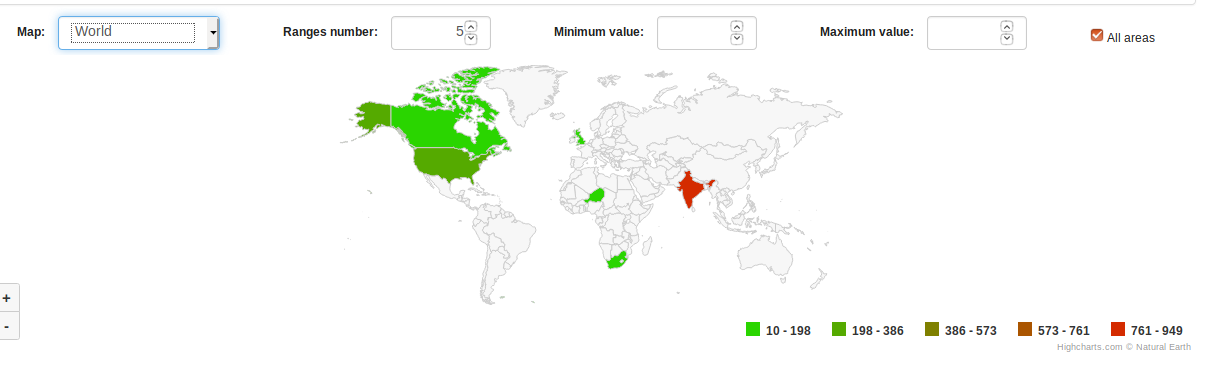
when place.country='Russia' then 'ru'

when place.country='China' then 'cn'

when place.country='Greenland ' then 'gl'

else place.country end as country

,count(\*) AS count FROM table where place.country is not null GROUP BY place.country ORDER BY count DESC limit 10



Githhub link: <https://github.com/gowtham304/PB>