University of Missouri- Kansas City

MASTERS OF SCIENCE IN COMPUTER SCIENCE

(COURSE NAME: Principles of Big Data)

Project Report

To extract, store and visualize the data using Spark and High charts API

Submitted by:-

Prudhvi Raj Mudunuri 16208160

Velagapudi Bhargav Krishna 16207553

Vipin Reddy Sattineni 16208781

Sudhakar Reddy16209800

Implementation of Queries:

Query1:

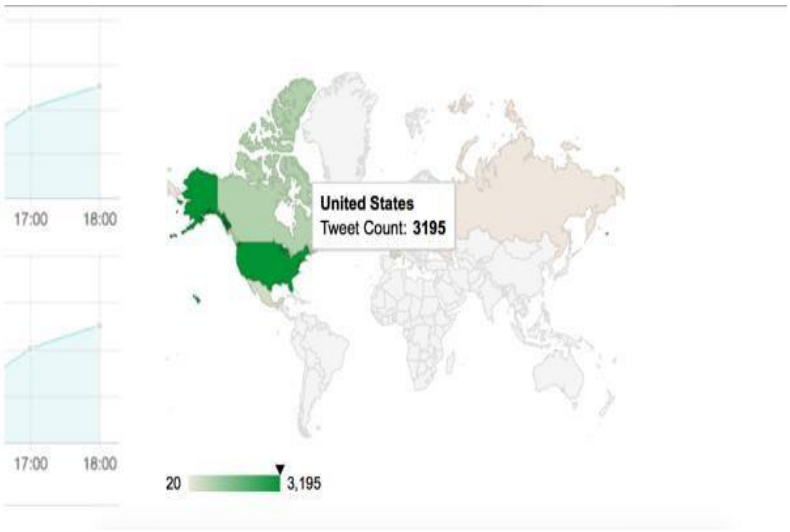
Q) Get country name and count of tweets from the country

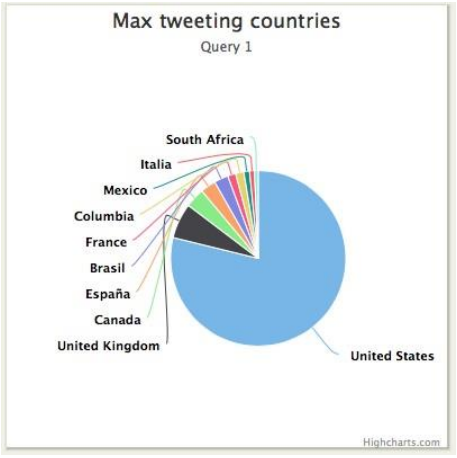
```
val q1 = sqlContext.sql("SELECT place.country,COUNT(*) AS country_count from tweets WHERE place.country is not null GROUP by place.country order by country_count desc limit 10")
```

Save the output:

```
q1.coalesce(1).save("/home/prudhvi/Downloads/Outputs/q1/","com.databricks.spark.csv")
```

Graph:-





Query2:

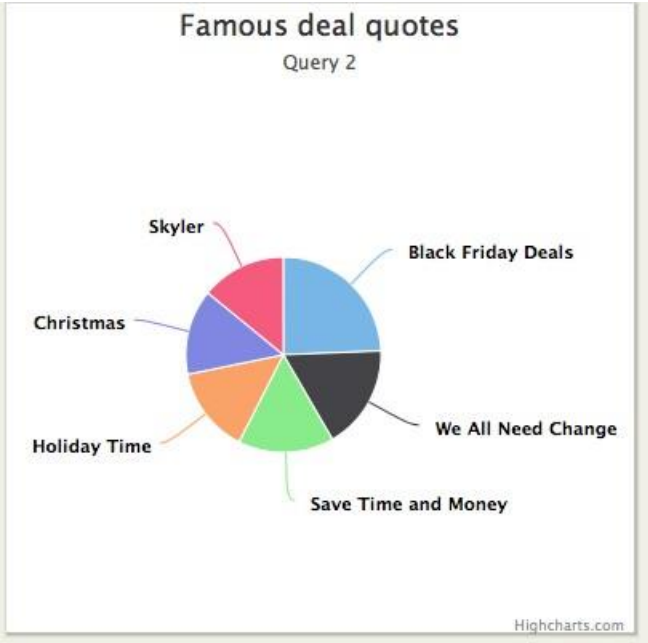
Q) Get the most tweeted text from the tweets collected

```
SELECT aWord, COUNT(*) AS WordOccuranceCount FROM (SELECT
SUBSTRING_INDEX(SUBSTRING_INDEX(concat(user.description, ' '), ' ', aCnt), ' ', -1) AS aWord
FROM tweets CROSS JOIN ( SELECT a.i+b.i*10+c.i*100 + 1 AS aCnt FROM integers a, integers b,
integers c) Sub1 WHERE (LENGTH(text) + 1 - LENGTH(REPLACE(text, ' ', ''))) >= aCnt) Sub2
WHERE Sub2.aWord != '' GROUP BY aWord ORDER BY WordOccuranceCount DESC LIMIT 6
```

Save the output:

```
q2.coalesce(1).save("/home/prudhvi/Downloads/Outputs/q2/","com.databricks.spark.csv")
```

Graph:



Query3:

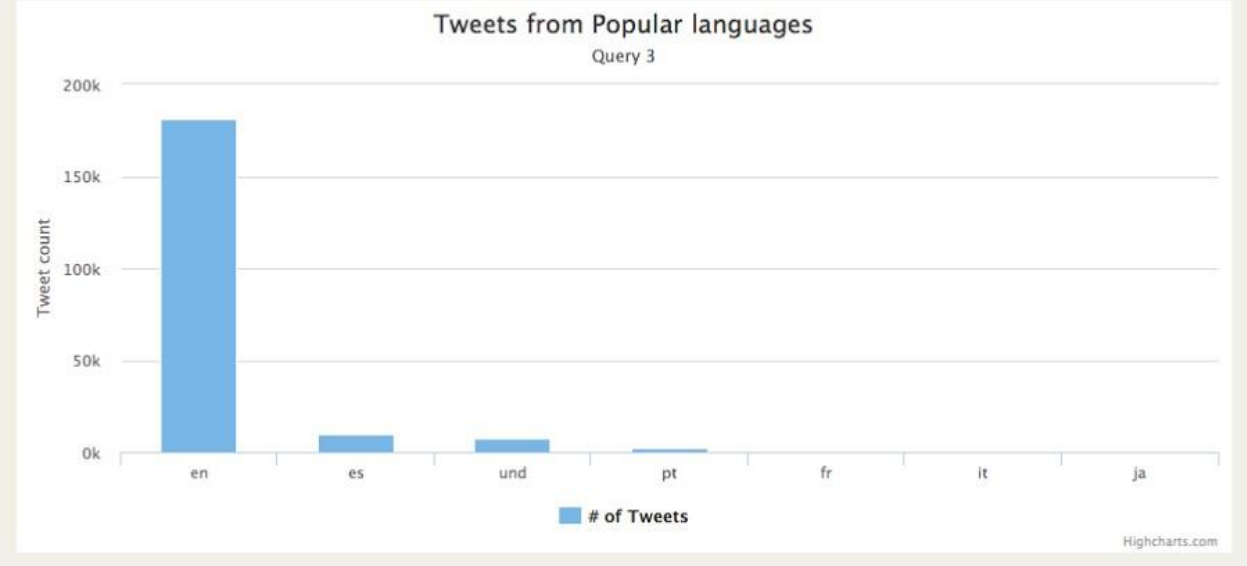
Q) Get the most popular languages used to tweet in twitter

```
val q3  = sqlContext.sql("SELECT DISTINCT lang,COUNT(lang) AS tweet_count FROM tweets GROUP BY lang LIMIT 7")
```

Save output file:

```
q3.coalesce(1).save("/home/prudhvi/Downloads/Outputs/q3/","com.databricks.spark.csv")
```

Graph:-



Query4:

Q) Get the number of people who are talking about walmart

```
val q4  = sqlContext.sql("SELECT COUNT(text) AS Walmart_visitors from tweets Where text regexp('[*mart]')")
```

Save output file:

```
q4.coalesce(1).save("/home/prudhvi/Downloads/Outputs/q4/","com.databricks.spark.csv")
```

Graph:-

Query 5:

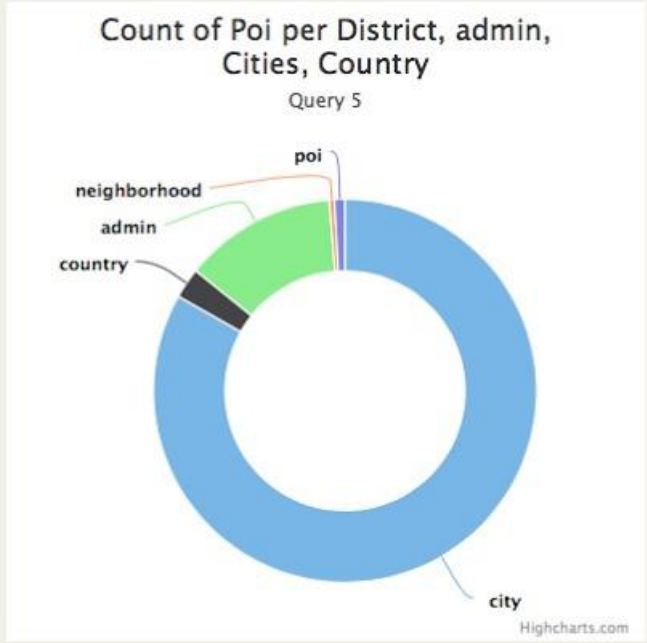
Q) Get the number of cities, countries which are recorded in tweet data

```
val q6  = sqlContext.sql("SELECT DISTINCT place.place_type,COUNT(place.place_type) AS tweet_count FROM tweets GROUP BY place.place_type")
```

Save the output file:

```
q6.coalesce(1).save("/home/prudhvi/Downloads/Outputs/q6/","com.databricks.spark.csv")
```

Graph:-

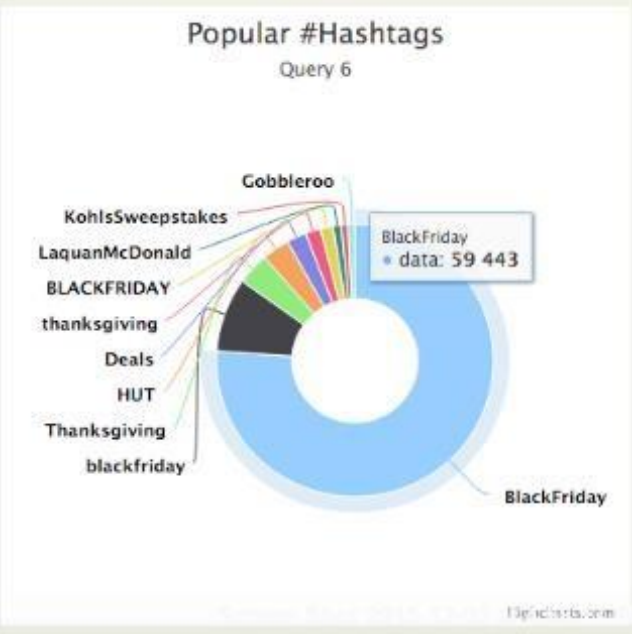


Query 6:

Q) Get the most tweeted hashtag

```
val q7  = sqlContext.sql("SELECT entities.hashtags[0].text, count(entities.hashtags[0].text) as famous_tags FROM tweets group by entities.hashtags[0].text order by famous_tags desc limit 10");
```

Graph:-

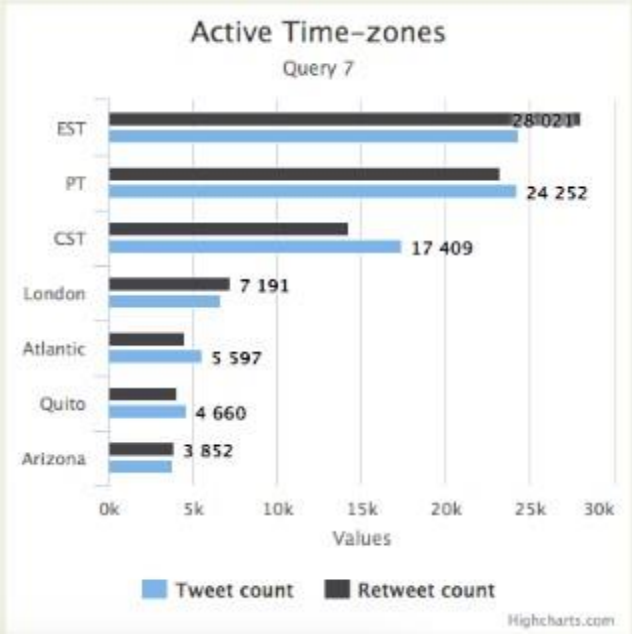


Query 7:

Q) Get the Time zone, Tweet count and retweet count from the data

```
val x1 = sqlContext.sql("select user.time_zone as time_zone, count(*) as Tweet_count from tweets where user.time_zone is not null group by user.time_zone order by Tweet_count desc")

x.registerTempTable("x1")

val x2 = sqlContext.sql("select retweeted_status.user.time_zone as time_zone, count(*) as Retweet_count from tweets where retweeted_status.user.time_zone is not null group by retweeted_status.user.time_zone order by Retweet_count desc")

x2.registerTempTable("x2")

val Query5 = sqlContext.sql("select x1.time_zone, x1.Tweet_count, x2.Retweet_count from x1 inner join x2 on x1.time_zone = x2.time_zone order by x1.Tweet_count desc")
```

Graph:



Query 8:

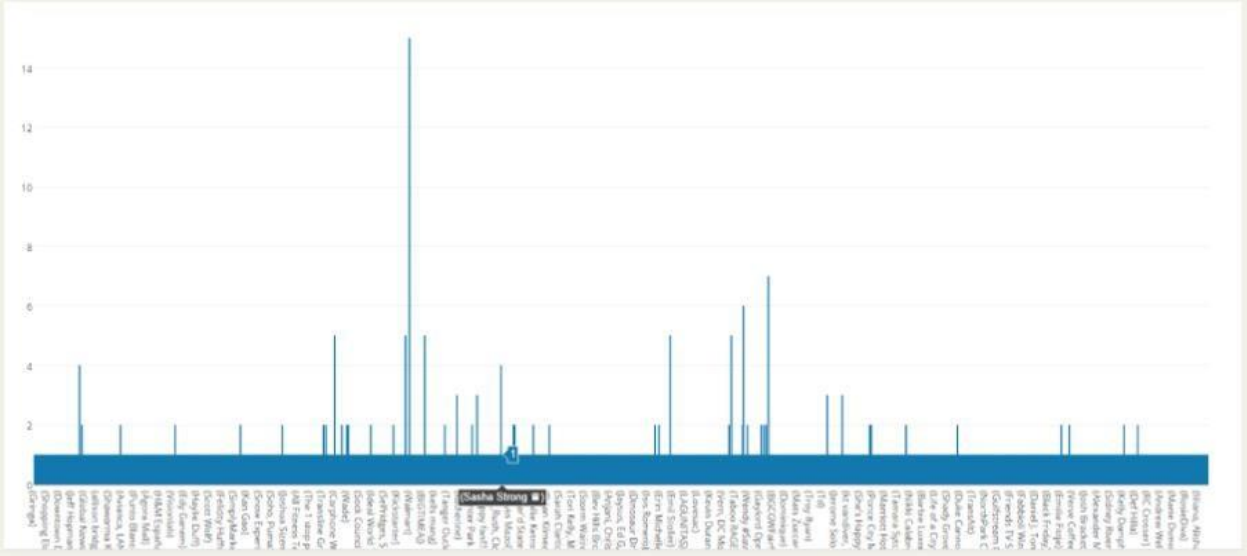
Q) Get the usernames and their country location who are mentioning twitter user in their tweets about the deals or encouraging or promoting them to tweet

val q5 = sqlContext.sql("SELECT entities.user_mentions.name[0],place.country from tweets WHERE place.country IS NOT NULL ORDER by place.country")

save the output file:

q8.coalesce(1).save("/home/prudhvi/Downloads/Outputs/q8/","com.databricks.spark.csv")

Graph:-



Conclusion:

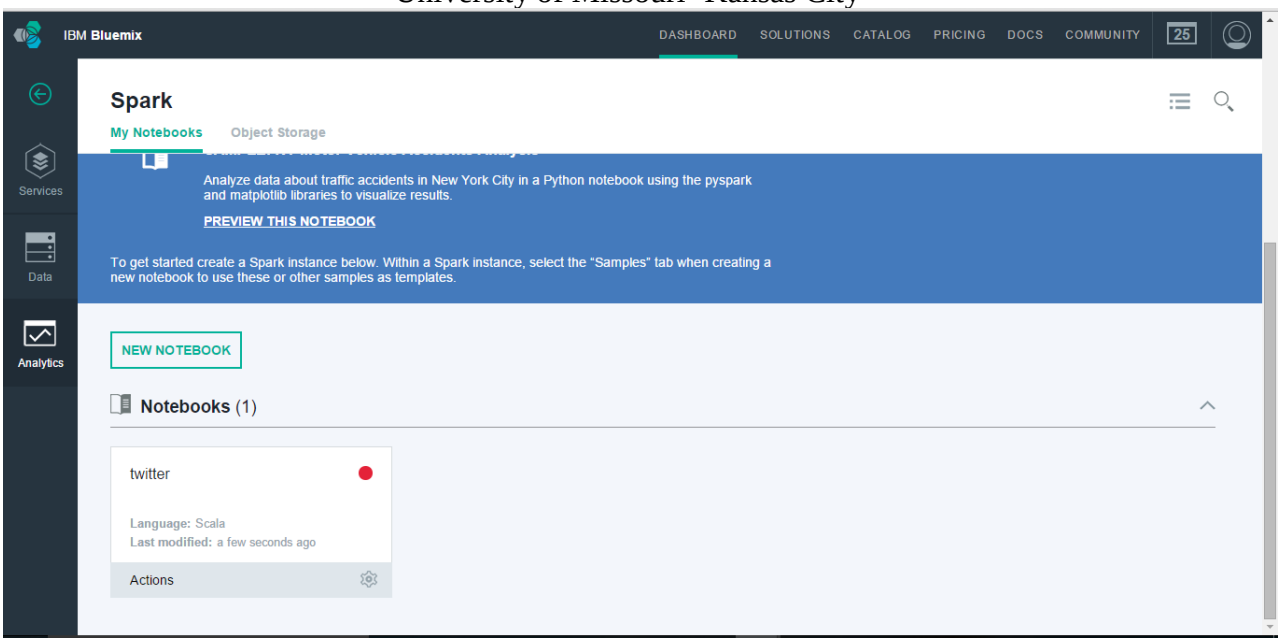
This web UI can be made more interactive by providing the facility for the user to specify his own custom queries and plot the results in graphs.

Further this can be used for sentiment analysis by improvising our query8.

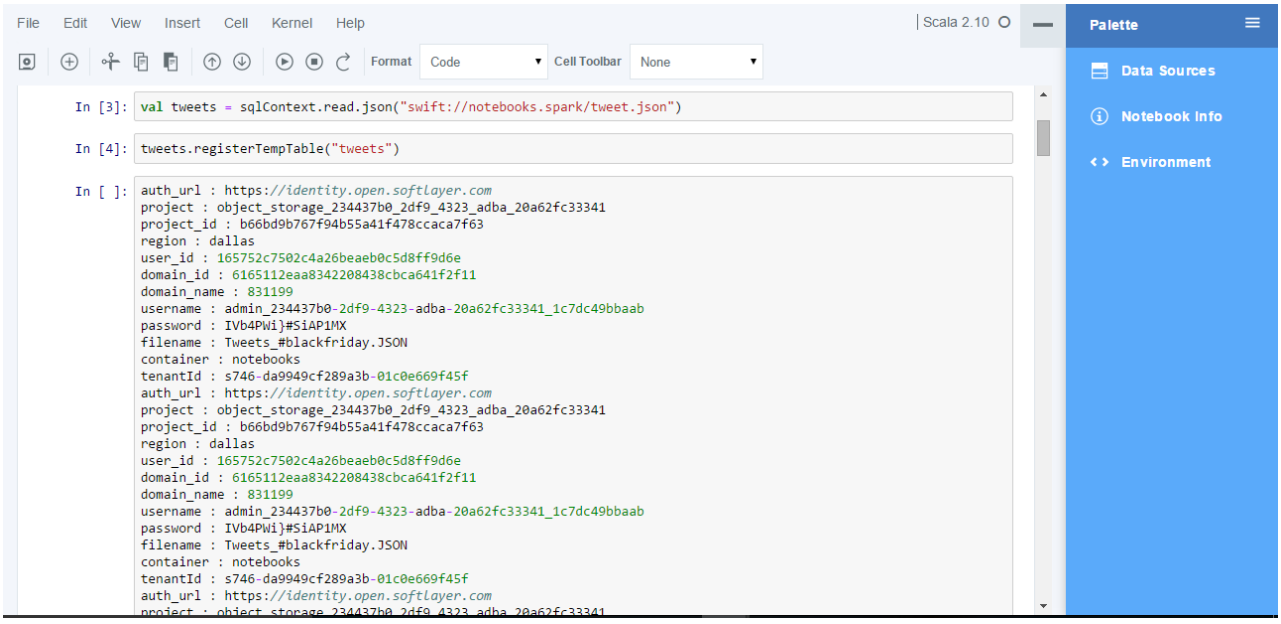
Our UI can be integrated with a third party application for more complicated analysis on the data.

Deployment in BLUEMIX:

Running spark as a service:-



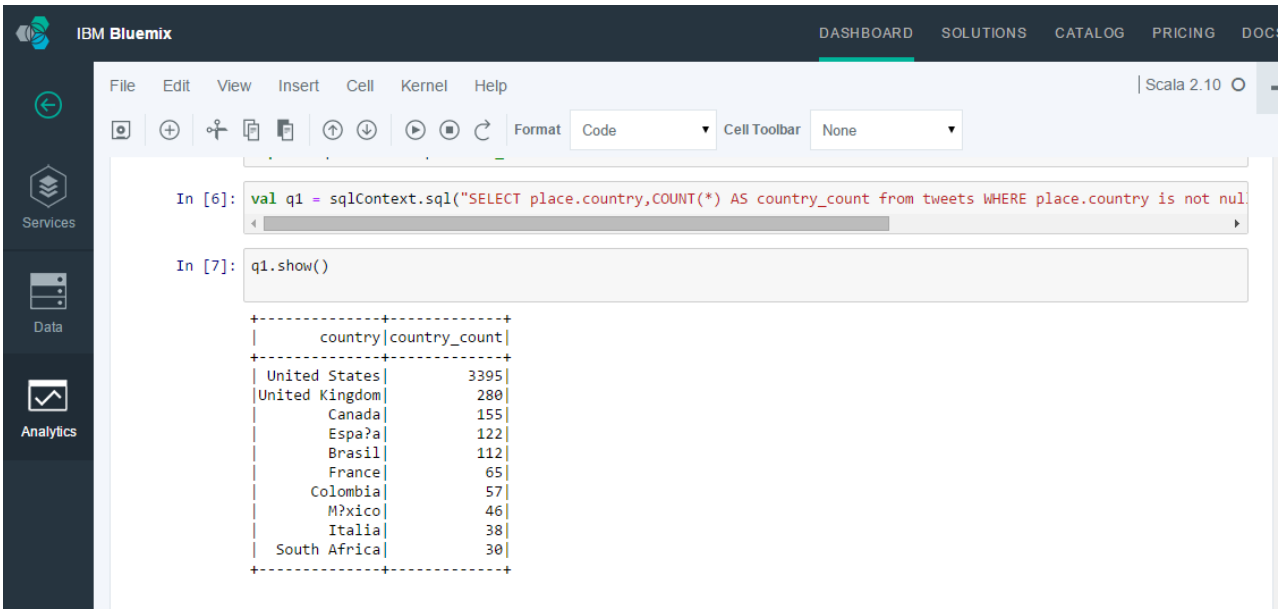
Adding context in BLUEMIX



Automatic schema detection :

```
In [5]: tweets.printSchema()

root
 |-- contributors: string (nullable = true)
 |-- coordinates: struct (nullable = true)
 |    |-- coordinates: array (nullable = true)
 |    |    |-- element: double (containsNull = true)
 |    |-- type: string (nullable = true)
 |-- created_at: string (nullable = true)
 |-- entities: struct (nullable = true)
 |    |-- hashtags: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- indices: array (nullable = true)
 |    |    |    |    |-- element: long (containsNull = true)
 |    |    |    |-- text: string (nullable = true)
 |    |-- media: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- display_url: string (nullable = true)
 |    |    |    |-- expanded_url: string (nullable = true)
 |    |    |    |-- id: long (nullable = true)
 |    |    |    |-- id_str: string (nullable = true)

In [2]: val sqlContext = new org.apache.spark.sql.SQLContext(sc)
```

Executing queries:



Bluemix [URL:-]

http://twitterthanksgivin.mybluemix.net/

Screen shots: