COLLECTING TWEETS USING TWITTER STREAMING API'S

Sub: Principles of Big Data Management

Phase 2 of Project

Sirisha Rella	(16268829)
Vineeth Dudipalli	(16274444)
Rajeshwari Cholleti	(16272108)

Instructor

Dr. PRAVEEN RAO, Ph.D.

Abstract:

Phase 2 of this project deals with the following requirements:

1. Writing interesting analytical queries on twitter data that we collected.

2. Developing interesting visualizations.

Title:

Technology in different Domains.

Technologies & Tools Used:

1. Spark

2. Python

3. Tableau

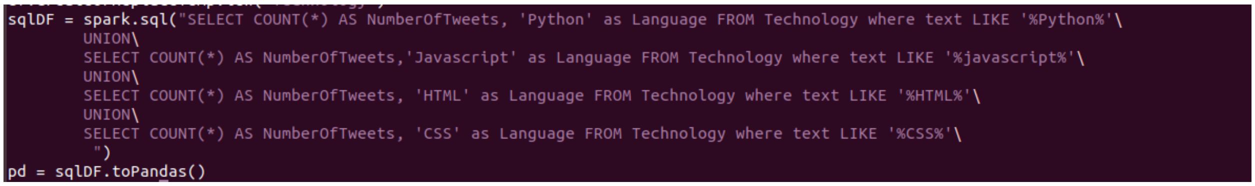
4. Spyder

Queries and Analysis:

Query-1:

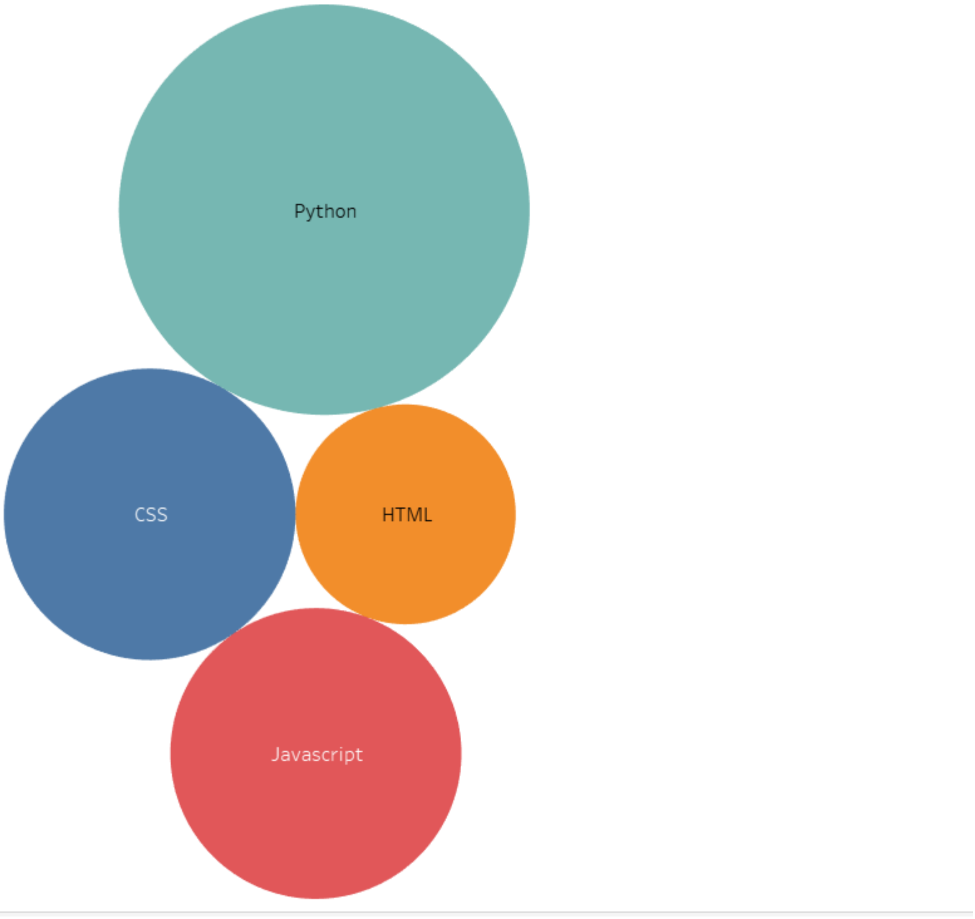
In this query we found the number of users tweeting about Python, JavaScript, HTML and CSS.

Code:



```
sqlDF = spark.sql("SELECT COUNT(*) AS NumberOfTweets, 'Python' as Language FROM Technology where text LIKE '%Python%'\
        UNION\
        SELECT COUNT(*) AS NumberOfTweets,'Javascript' as Language FROM Technology where text LIKE '%javascript%'\
        UNION\
        SELECT COUNT(*) AS NumberOfTweets, 'HTML' as Language FROM Technology where text LIKE '%HTML%'\
        UNION\
        SELECT COUNT(*) AS NumberOfTweets, 'CSS' as Language FROM Technology where text LIKE '%CSS%'\
         ")
pd = sqlDF.toPandas()
```

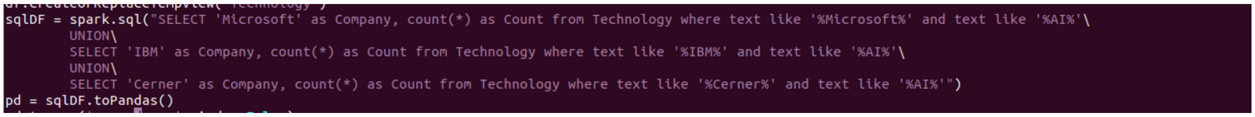
Visualization:



Query-2:

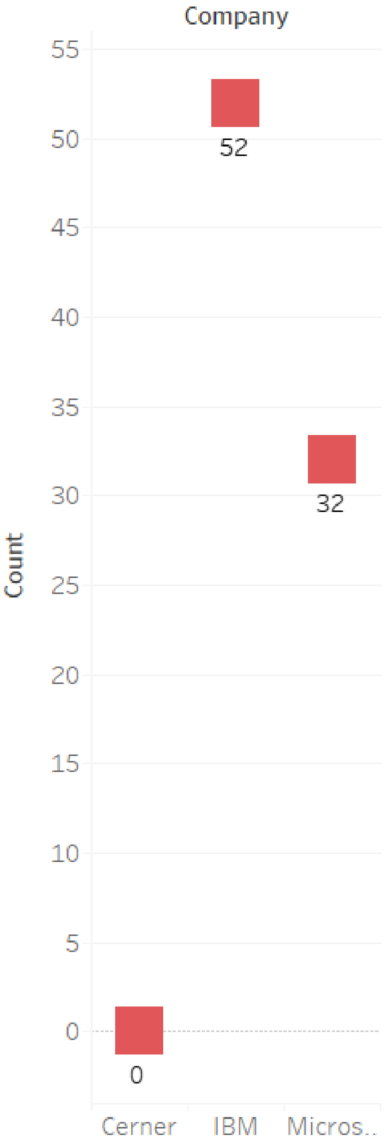
In this query we found how Microsoft, IBM and Cerner associates/employees are actively tweeting about AI.

Code:



```
sqlDF = spark.sql("SELECT 'Microsoft' as Company, count(*) as Count from Technology where text like '%Microsoft%' and text like '%AI%'\
        UNION\
        SELECT 'IBM' as Company, count(*) as Count from Technology where text like '%IBM%' and text like '%AI%'\
        UNION\
        SELECT 'Cerner' as Company, count(*) as Count from Technology where text like '%Cerner%' and text like '%AI%'")
pd = sqlDF.toPandas()
```

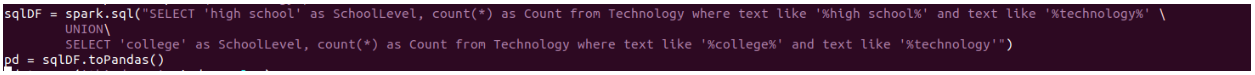
Visualization:



Query-3:

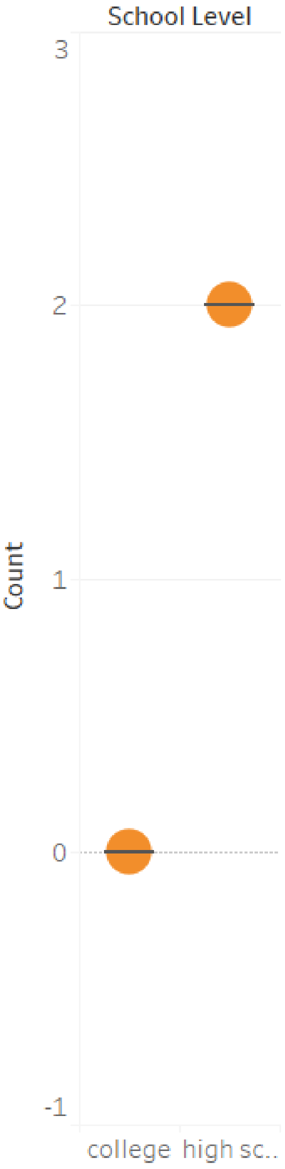
In this query we found count of active high school/college students' achievements/participation in technology.

Code:



```
sqlDF = spark.sql("SELECT 'high school' as SchoolLevel, count(*) as Count from Technology where text like '%high school%' and text like '%technology%' \
        UNION\
        SELECT 'college' as SchoolLevel, count(*) as Count from Technology where text like '%college%' and text like '%technology'")
pd = sqlDF.toPandas()
```

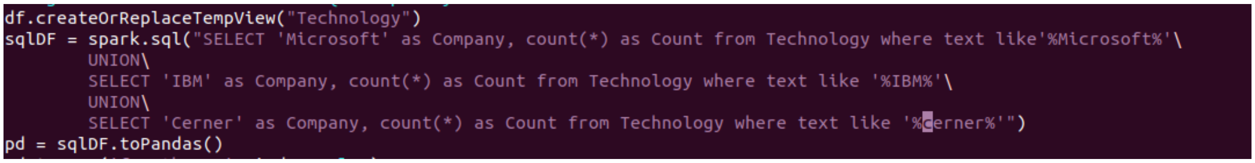
Visualization:



Query-4:

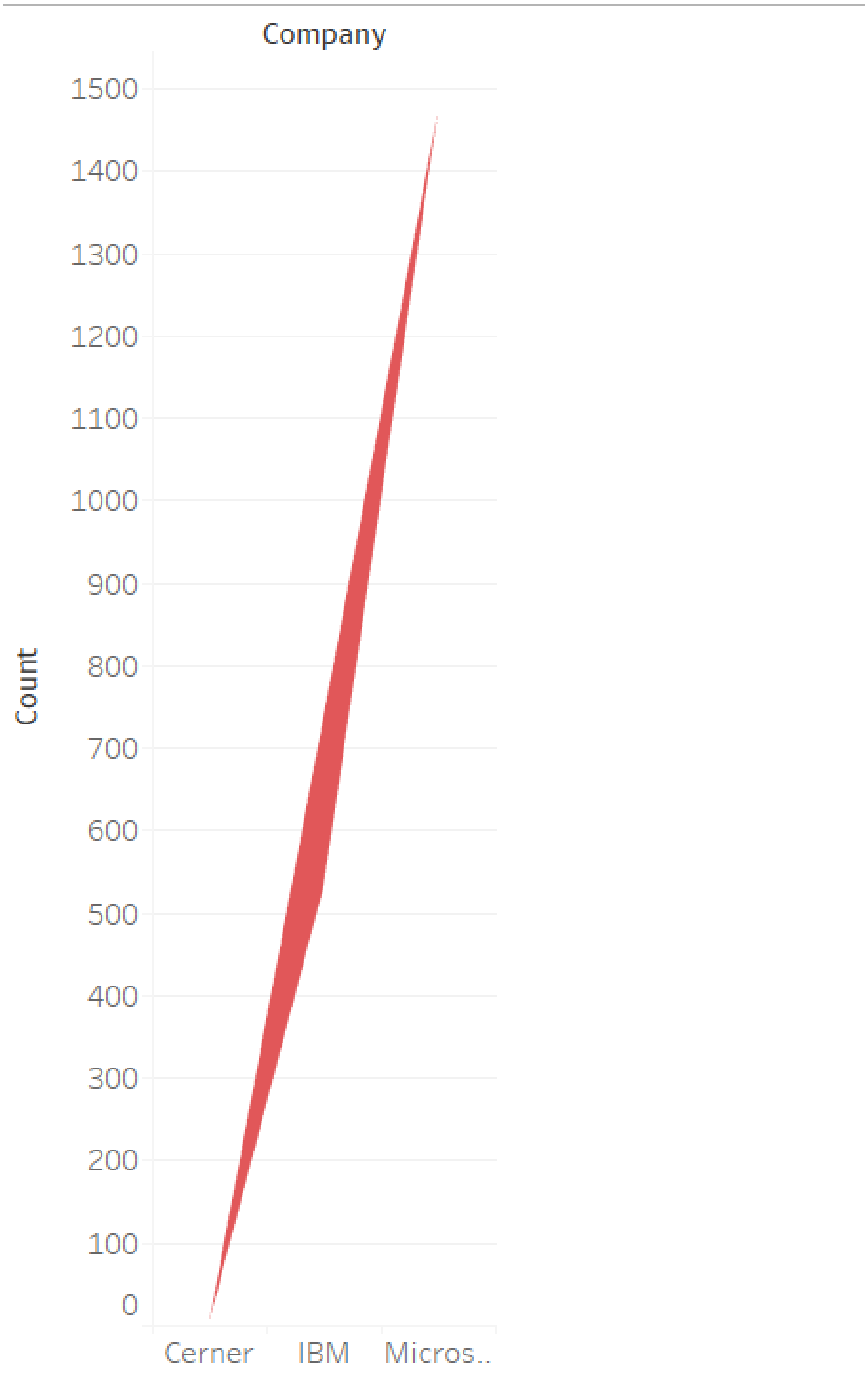
In this query we found the Count of people who are actively tweeting about Microsoft, Cerner and IBM

Code:



```
df.createOrReplaceTempView("Technology")
sqlDF = spark.sql("SELECT 'Microsoft' as Company, count(*) as Count from Technology where text like'%Microsoft%'\
        UNION\
        SELECT 'IBM' as Company, count(*) as Count from Technology where text like '%IBM%'\
        UNION\
        SELECT 'Cerner' as Company, count(*) as Count from Technology where text like '%cerner%'")
pd = sqlDF.toPandas()
```

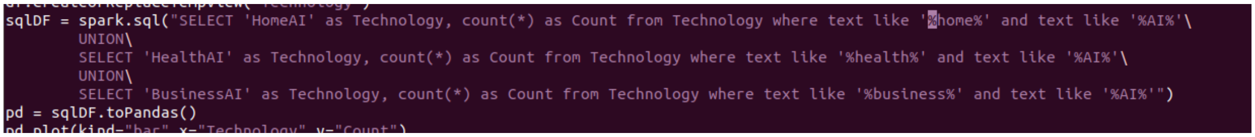
Visualization:


Company
Count
1500
1400
1300
1200
1100
1000
900
800
700
600
500
400
300
200
100
0
Cerner
IBM
Micros..

Query-5:

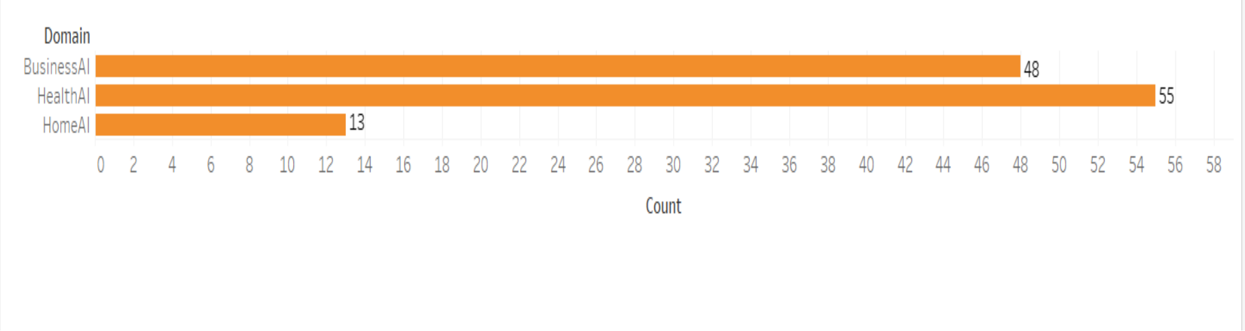
In this query we found tweets about how AI is impacting in health, home and business AI.

Code:



```
df.createOrReplaceTempView("Technology")
sqlDF = spark.sql("SELECT 'HomeAI' as Technology, count(*) as Count from Technology where text like '%home%' and text like '%AI%'\
        UNION\
        SELECT 'HealthAI' as Technology, count(*) as Count from Technology where text like '%health%' and text like '%AI%'\
        UNION\
        SELECT 'BusinessAI' as Technology, count(*) as Count from Technology where text like '%business%' and text like '%AI%'")
pd = sqlDF.toPandas()
pd.plot(kind="bar", x="Technology", y="Count")
```

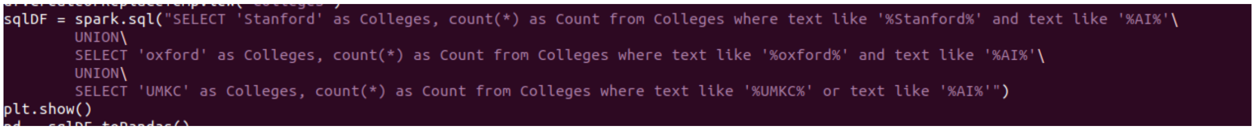
Visualization:



Query-6:

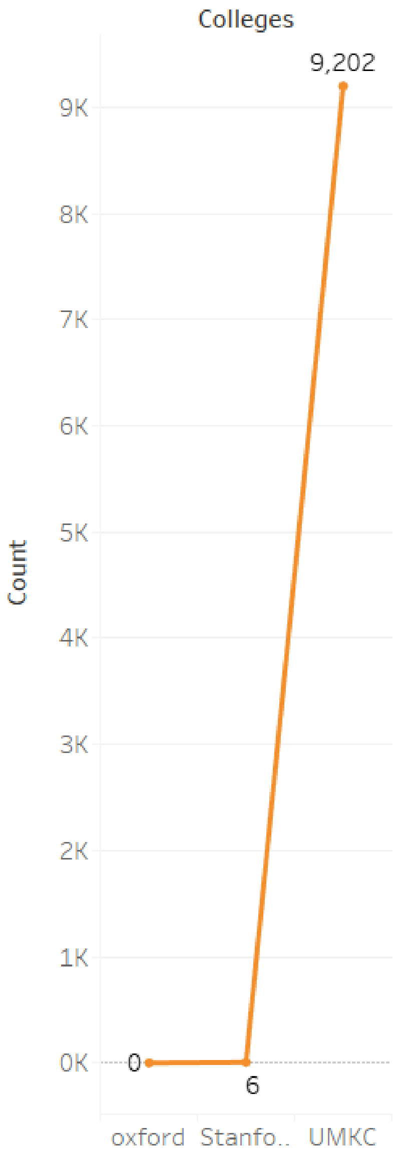
In this query we found which students who are actively participating/tweeting about python programming based on college name.

Code:



```
sqlDF = spark.sql("SELECT 'Stanford' as Colleges, count(*) as Count from Colleges where text like '%Stanford%' and text like '%AI%'\
         UNION\
         SELECT 'oxford' as Colleges, count(*) as Count from Colleges where text like '%oxford%' and text like '%AI%'\
         UNION\
         SELECT 'UMKC' as Colleges, count(*) as Count from Colleges where text like '%UMKC%' or text like '%AI%'")
plt.show()
```

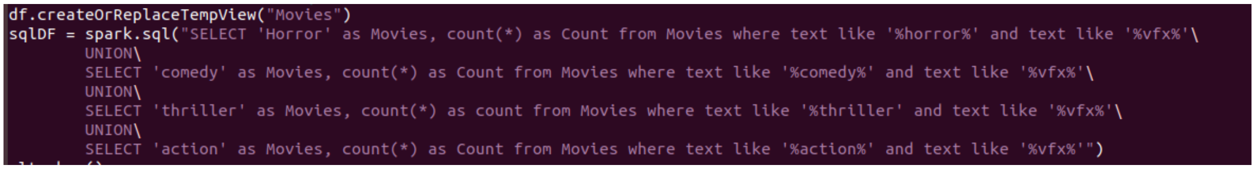
Visualization:



Query-7:

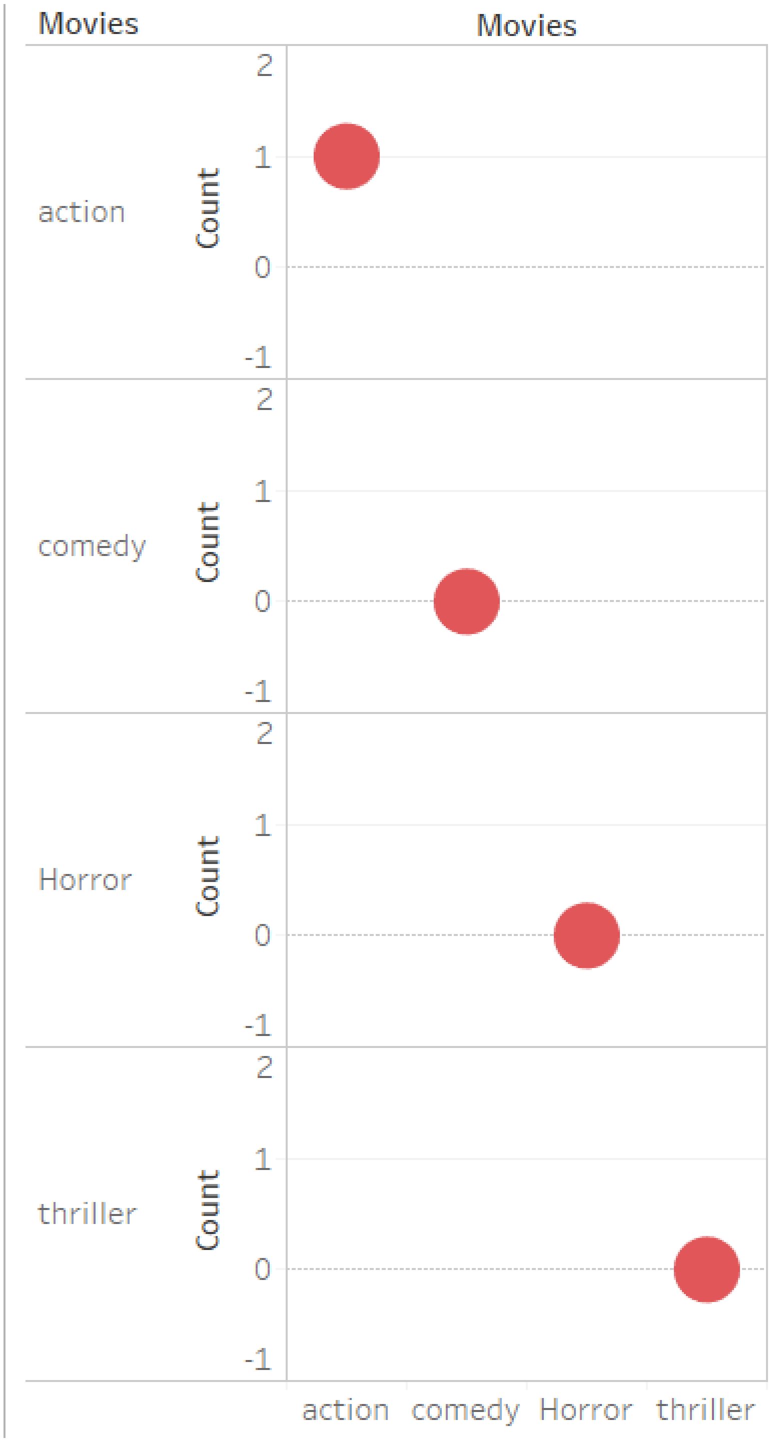
In this query we found tweets that talk about different kind of films that are using vfx effects.

Code:



```
df.createOrReplaceTempView("Movies")
sqlDF = spark.sql("SELECT 'Horror' as Movies, count(*) as Count from Movies where text like '%horror%' and text like '%vfx%'\
        UNION\
        SELECT 'comedy' as Movies, count(*) as Count from Movies where text like '%comedy%' and text like '%vfx%'\
        UNION\
        SELECT 'thriller' as Movies, count(*) as count from Movies where text like '%thriller' and text like '%vfx%'\
        UNION\
        SELECT 'action' as Movies, count(*) as Count from Movies where text like '%action%' and text like '%vfx%'")
```

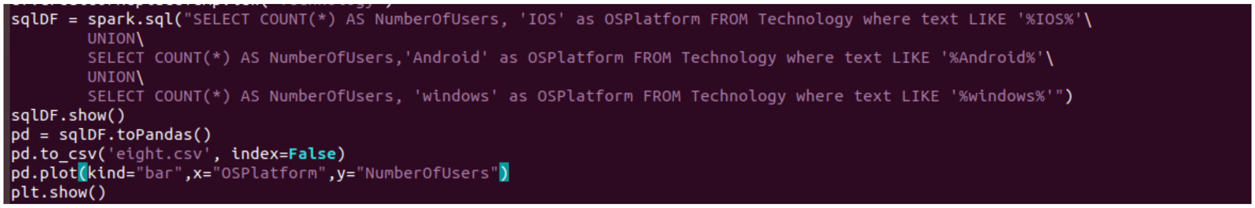
Visualization:


Movies
Movies
action
comedy
Horror
thriller
Count
2
1
0
-1
action
comedy
Horror
thriller

Query-8:

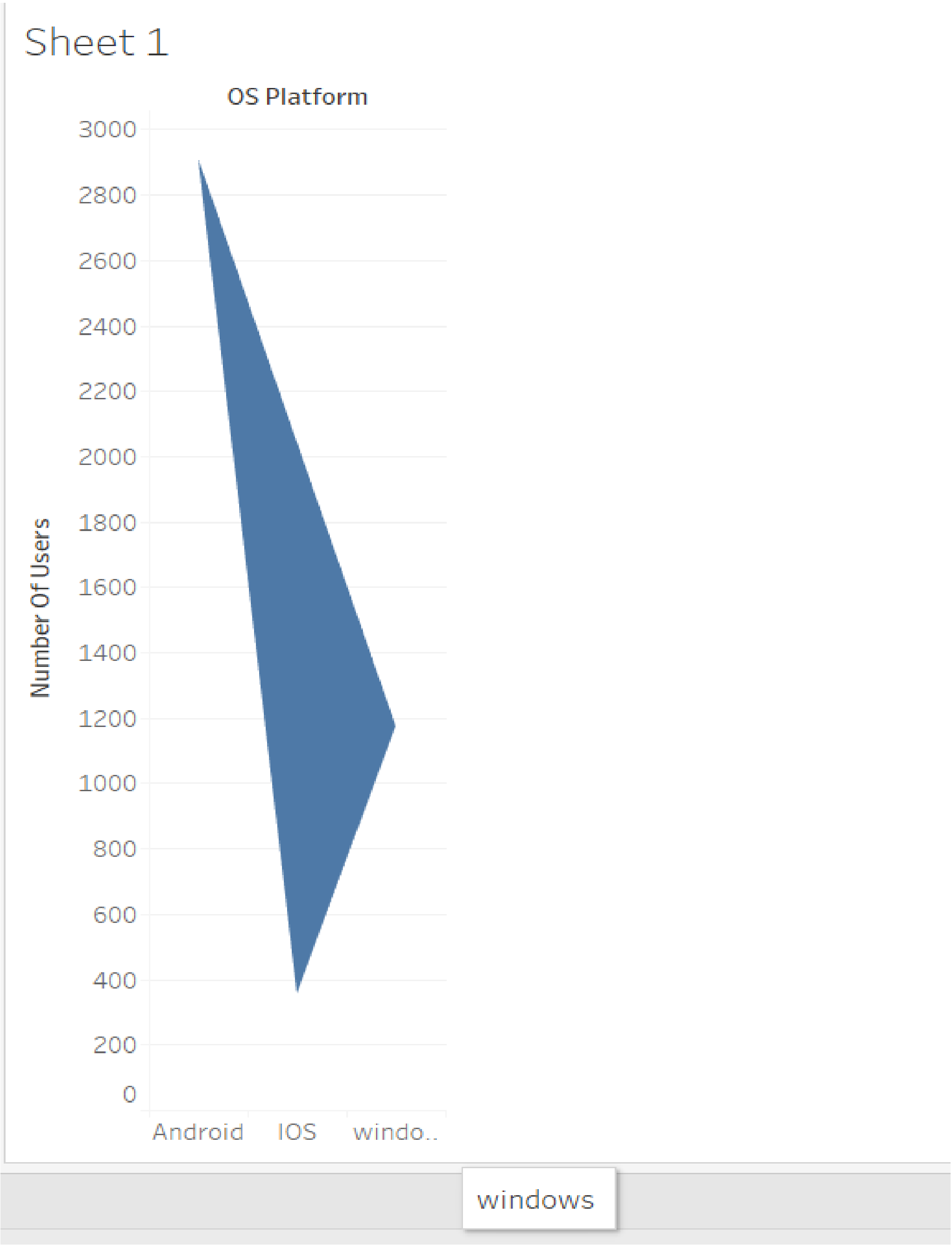
In this query we found number of people tweeting about IOS, Android and Windows operating systems.

Code:



```
sqlDF = spark.sql("SELECT COUNT(*) AS NumberOfUsers, 'IOS' as OSPlatform FROM Technology where text LIKE '%IOS%'\
        UNION\
        SELECT COUNT(*) AS NumberOfUsers,'Android' as OSPlatform FROM Technology where text LIKE '%Android%'\
        UNION\
        SELECT COUNT(*) AS NumberOfUsers, 'windows' as OSPlatform FROM Technology where text LIKE '%windows%'")
sqlDF.show()
pd = sqlDF.toPandas()
pd.to_csv('eight.csv', index=False)
pd.plot(kind="bar",x="OSPlatform",y="NumberOfUsers")
plt.show()
```

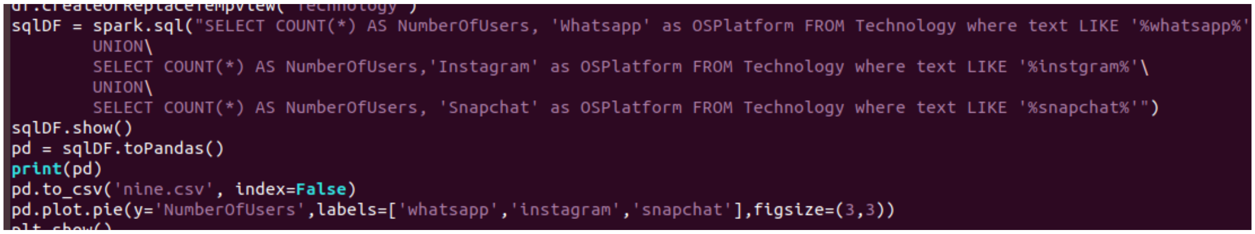
Visualization:


Sheet 1
OS Platform
Number Of Users
3000
2800
2600
2400
2200
2000
1800
1600
1400
1200
1000
800
600
400
200
0
Android
IOS
windo..
windows

Query-9:

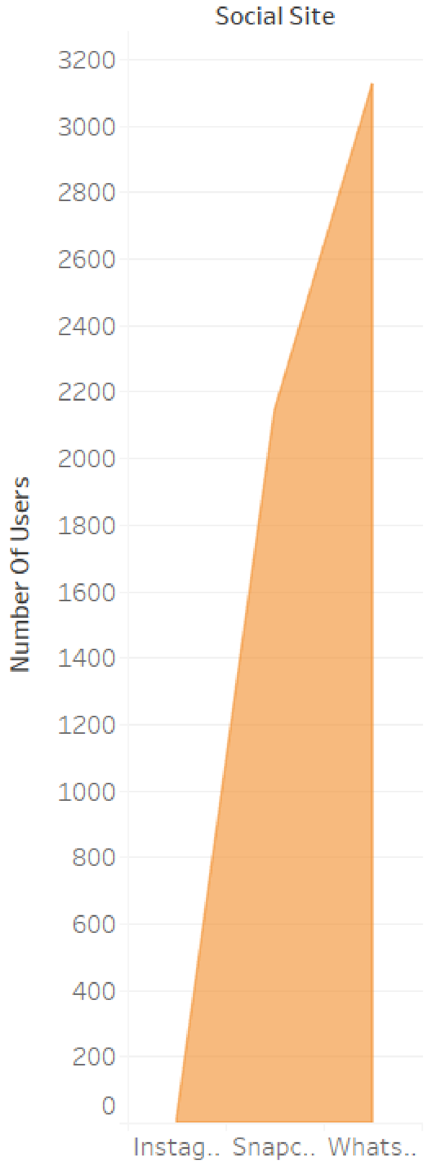
In this query we found number of people tweeting about WhatsApp, Instagram and snapchat

Code:



```
df.createOrReplaceTempView('Technology')
sqlDF = spark.sql("SELECT COUNT(*) AS NumberOfUsers, 'Whatsapp' as OSPlatform FROM Technology where text LIKE '%whatsapp%'
        UNION\
        SELECT COUNT(*) AS NumberOfUsers,'Instagram' as OSPlatform FROM Technology where text LIKE '%instgram%'\
        UNION\
        SELECT COUNT(*) AS NumberOfUsers, 'Snapchat' as OSPlatform FROM Technology where text LIKE '%snapchat%'")
sqlDF.show()
pd = sqlDF.toPandas()
print(pd)
pd.to_csv('nine.csv', index=False)
pd.plot.pie(y='NumberOfUsers',labels=['whatsapp','instagram','snapchat'],figsize=(3,3))
plt.show()
```

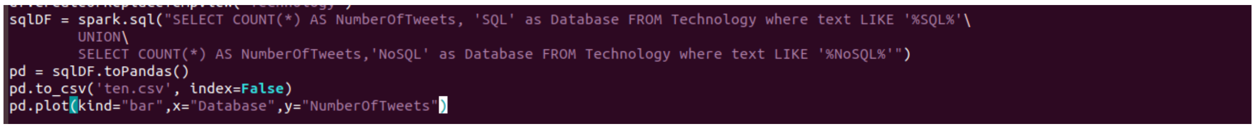
Visualization:


Social Site
Number Of Users
3200
3000
2800
2600
2400
2200
2000
1800
1600
1400
1200
1000
800
600
400
200
0
Instag..
Snapc..
Whats..

Query-10:

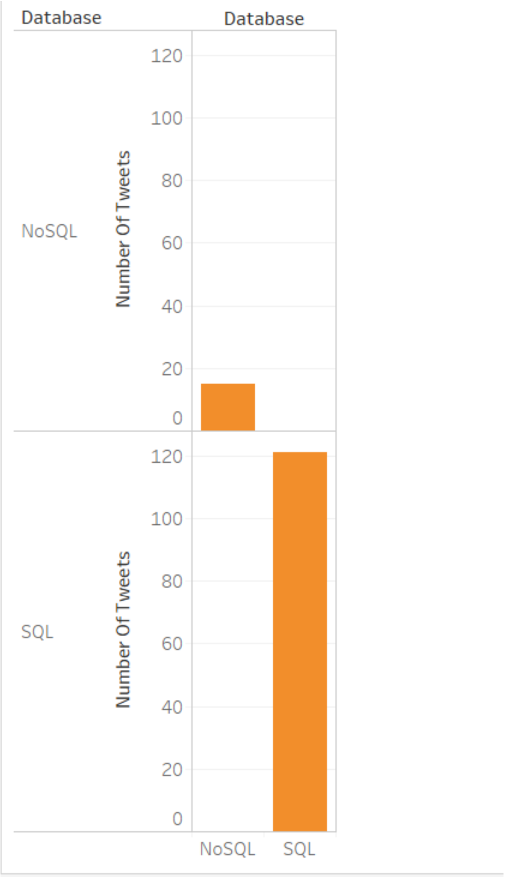
In this query we found number of people tweeting about SQL and NoSQL

Code:



```
sqlDF = spark.sql("SELECT COUNT(*) AS NumberOfTweets, 'SQL' as Database FROM Technology where text LIKE '%SQL%'\
        UNION\
        SELECT COUNT(*) AS NumberOfTweets,'NoSQL' as Database FROM Technology where text LIKE '%NoSQL%'")
pd = sqlDF.toPandas()
pd.to_csv('ten.csv', index=False)
pd.plot(kind="bar",x="Database",y="NumberOfTweets")
```

Visualization:

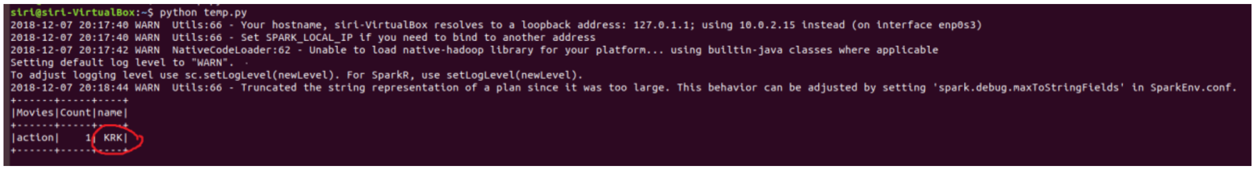


Testing:

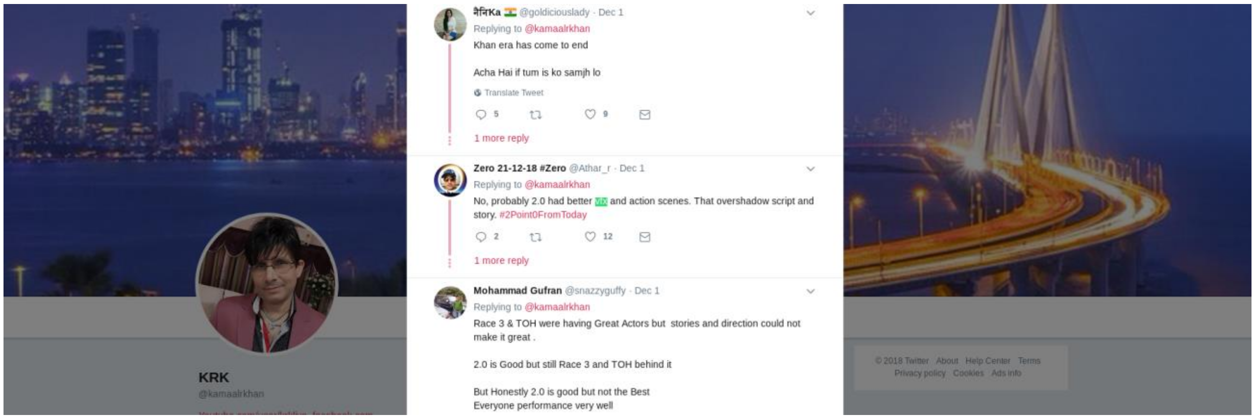
Manual testing:

We tried to test the results manually by taking the keywords used for the query and finding the tweets for the same keywords using twitter search engine in google and tried to match the tweets with the twitter data we retrieved.

For instance, consider the VFX query for which we retrieved the profile user name of the particular tweet and searched it manually in twitter to find the tweet and check it whether he tweeted or not.



```
siri@siri-VirtualBox:~$ python temp.py
2018-12-07 20:17:40 WARN  Utils:66 - Your hostname, siri-VirtualBox resolves to a loopback address: 127.0.1.1; using 10.0.2.15 instead (on interface enp0s3)
2018-12-07 20:17:40 WARN  Utils:66 - Set SPARK_LOCAL_IP if you need to bind to another address
2018-12-07 20:17:42 WARN  NativeCodeLoader:62 - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
2018-12-07 20:18:44 WARN  Utils:66 - Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.debug.maxToStringFields' in SparkEnv.conf.
+------+-----+----+
|Movies|Count|name|
+------+-----+----+
|action|    1| KRK|
+------+-----+----+
```



KRK
@kamaalrkhan

नैनिKa 🇮🇳 @goldiciouslady · Dec 1
Replying to @kamaalrkhan
Khan era has come to end

Acha Hai if tum is ko samjh lo

Translate Tweet
5 9
1 more reply

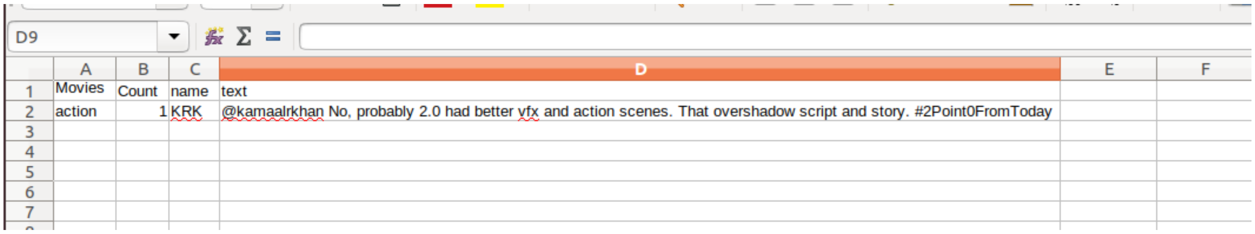
Zero 21-12-18 #Zero @Athar_r · Dec 1
Replying to @kamaalrkhan
No, probably 2.0 had better vfx and action scenes. That overshadow script and story. #2Point0FromToday
2 12
1 more reply

Mohammad Gufran @snazzyguffy · Dec 1
Replying to @kamaalrkhan
Race 3 & TOH were having Great Actors but stories and direction could not make it great .

2.0 is Good but still Race 3 and TOH behind it

But Honestly 2.0 is good but not the Best
Everyone performance very well

© 2018 Twitter About Help Center Terms
Privacy policy Cookies Ads info

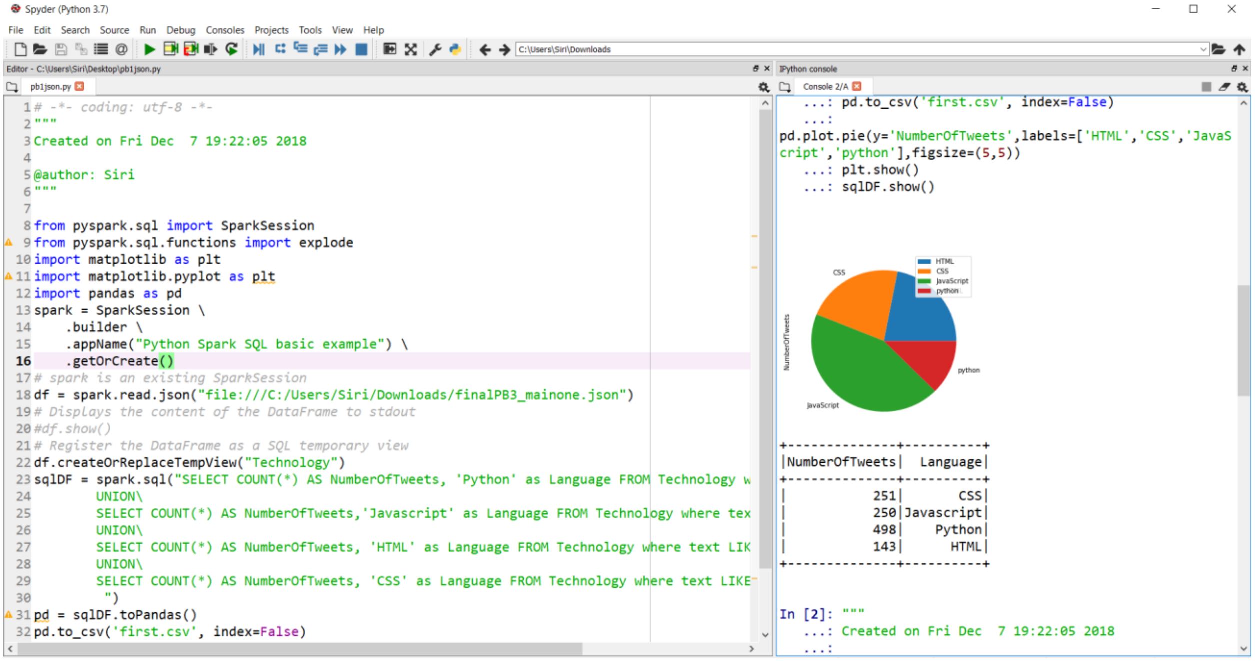


D9

	A	B	C	D	E	F
1	Movies	Count	name	text		
2	action	1	KRK	@kamaalrkhan No, probably 2.0 had better vfx and action scenes. That overshadow script and story. #2Point0FromToday		
3						
4						
5						
6						
7						

Testing Based on tools:

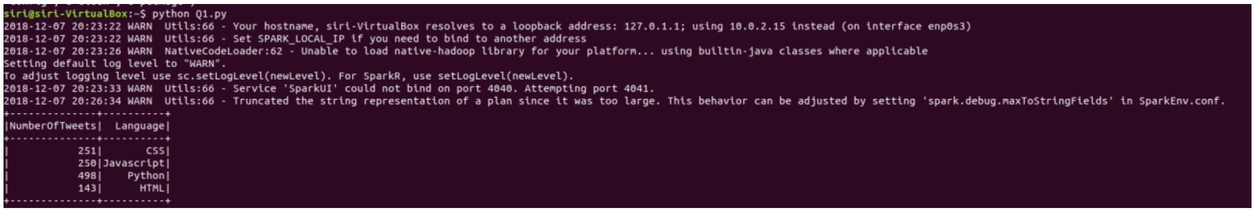
Running in SPYDER:



```
# -*- coding: utf-8 -*-
"""
Created on Fri Dec  7 19:22:05 2018

@author: Siri
"""

from pyspark.sql import SparkSession
from pyspark.sql.functions import explode
import matplotlib as plt
import matplotlib.pyplot as plt
import pandas as pd
spark = SparkSession \
    .builder \
    .appName("Python Spark SQL basic example") \
    .getOrCreate()
# spark is an existing SparkSession
df = spark.read.json("file:///C:/Users/Siri/Downloads/finalPB3_mainone.json")
# Displays the content of the DataFrame to stdout
#df.show()
# Register the DataFrame as a SQL temporary view
df.createOrReplaceTempView("Technology")
sqlDF = spark.sql("SELECT COUNT(*) AS NumberOfTweets, 'Python' as Language FROM Technology w
        UNION\
        SELECT COUNT(*) AS NumberOfTweets,'Javascript' as Language FROM Technology where tex
        UNION\
        SELECT COUNT(*) AS NumberOfTweets, 'HTML' as Language FROM Technology where text LIK
        UNION\
        SELECT COUNT(*) AS NumberOfTweets, 'CSS' as Language FROM Technology where text LIKE
         ")
pd = sqlDF.toPandas()
pd.to_csv('first.csv', index=False)
```

```
   ...: pd.to_csv('first.csv', index=False)
   ...:
pd.plot.pie(y='NumberOfTweets',labels=['HTML','CSS','JavaS
cript','python'],figsize=(5,5))
   ...: plt.show()
   ...: sqlDF.show()

+--------------+----------+
|NumberOfTweets|  Language|
+--------------+----------+
|           251|       CSS|
|           250|Javascript|
|           498|    Python|
|           143|      HTML|
+--------------+----------+


In [2]: """
   ...: Created on Fri Dec  7 19:22:05 2018
   ...:
```

Running in LINUX Terminal:



```
siri@siri-VirtualBox:~$ python Q1.py
2018-12-07 20:23:22 WARN  Utils:66 - Your hostname, siri-VirtualBox resolves to a loopback address: 127.0.1.1; using 10.0.2.15 instead (on interface enp0s3)
2018-12-07 20:23:22 WARN  Utils:66 - Set SPARK_LOCAL_IP if you need to bind to another address
2018-12-07 20:23:26 WARN  NativeCodeLoader:62 - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
2018-12-07 20:23:33 WARN  Utils:66 - Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
2018-12-07 20:26:34 WARN  Utils:66 - Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.debug.maxToStringFields' in SparkEnv.conf.
+--------------+----------+
|NumberOfTweets|  Language|
+--------------+----------+
|           251|       CSS|
|           250|Javascript|
|           498|    Python|
|           143|      HTML|
+--------------+----------+
```

As we can observe from the above two outputs both the outputs match each other therefore we confirm that the result is correct.

Code Link:

https://github.com/VineethDvv/Principles-of-Bigdata-