**PRINCIPLES OF BIG DATA**

**TWITTER DATA ANALYSIS - PHASE 2**

**Team:**

1. **Name:** Salkuti Sandeep Reddy

**Student id:** 16296868

**Email:** [sswf7@mail.umkc.edu](mailto:sswf7@mail.umkc.edu)

1. **Name:** Saichand Patchala

**Student id:** 16292087

**Email:** [spznc@mail.umkc.edu](mailto:spznc@mail.umkc.edu)

1. **Name:** Pardhasaradhi Ramineni

**Student id:** 16300893

**Email:** [prd6f@mail.missouri.edu](mailto:prd6f@mail.missouri.edu)

**Objective:**

The main goal in this phase is to achieve the following tasks

1. Analyze the twitter data downloaded of tweets around 200000 tweets.
2. Write atleast 10 queries from the analyzed data using different sql functions.
3. Visualize the data using any softwares and save them.

**Softwares used:**

Softwares used mainly are:

1. Apache Spark
2. IntelliJ IDE
3. IDLE ( for writing python plotting code)
4. Visual Studio Code

**Working Functionality:**

**Spark:** Installed spark and practiced basic queries in scala to understand twitter data and their results upon running queries.

**IntelliJ IDE:** Created project in IntelliJ IDE and downloaded maven dependencies required like spark context and spark sql and added them to build.sbt. Wrote small scala program by importing spark sql.

Read json data which is downloaded tweets and save in a temporary file named parquetFile and write 10 queries planned. Save each query using coalesce into .csv file format. So by running the scala program all 10 queries get executed at a time and data is saved as each individual folders say q1, q2….

**matplot:** Used matplotlib for plotting which is used for python programming language.

Wrote small python program where we imported matplotlib, pandas, numpy, glob which are used for plotting then using glob we read .csv file where our query results are stored and store them in a dataframe using pandas. Now read data from data frame and plot required diagram like bar, pie etc. based on need and label them and show diagram using show() function.

**VisualStudio:**

For showing the query results all at one place created a web application using python with Django .

This is basic web page where user clicks a button and it executes query internally and saves the query results as plot image in folder and displays plot to user. So user clicks each query button and sees visualization part without any internal data.

**Drive link for downloaded tweets data:**

<https://drive.google.com/drive/folders/12OlHVribo-1nnzG8E-b_DSSp4AmWrtce>

**Github link for project code and query results and visualizations:**

<https://github.com/sandeepsalkuti/PrinciplesofBigdataProject>

**IntelliJ IDE queries execution code:**

import org.apache.spark.sql.{SQLContext, SparkSession}  
object test {  
 def main(args: Array[String]): Unit = {  
 val spark = SparkSession.*builder*.appName("SparkSQL").master("local[\*]").getOrCreate()  
 System.*setProperty*("hadoop.home.dir", "C:\\spark-3.0.0-preview2-bin-hadoop2.7\\")  
  
 val PBDF=spark.read.json("C:\\sandeep\\tweetsdata.txt")  
  
 PBDF.createOrReplaceTempView("parquetFile")  
 //query1 what day retweet counts received  
 val day=spark.sql(sqlText="SELECT distinct substring(user.created\_at,1,3) as day,count(user.description)as count from parquetFile where user.description like '%Apple%'GROUP BY day ORDER BY day DESC LIMIT 10" )  
 day.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q1")  
  
 //query2 apple product with more likes  
 val likes=spark.sql(sqlText="SELECT user.name as name,user.statuses\_count as cunt from parquetFile where user.description like '%Apple%' ORDER BY user.statuses\_count DESC LIMIT 10")  
 likes.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q2")  
  
 //query3 number of languages comments received  
 val langcomm=spark.sql(sqlText="SELECT lang, Count(lang) as Count from parquetFile where id is NOT NULL and lang is not null and text like '%Apple%' group by lang order by Count DESC")  
 langcomm.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q3")  
  
 //query 4 country and tweets count  
 val country=spark.sql(sqlText="SELECT distinct place.country, count(\*) as count FROM parquetFile where place.country is not null " + "GROUP BY place.country ORDER BY count DESC LIMIT 10")  
 country.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q4")  
  
 //query 5 user with followers count  
 val morefollowers =spark.sql(sqlText="SELECT user.name , user.followers\_count from parquetFile where retweeted\_status.text like '%iphone%' ORDER BY user.followers\_count DESC LIMIT 10")  
 morefollowers.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q5")  
  
 //query 6 common hashtags used by iphone users  
 //val hashtags=spark.sql(sqlText="SELECT entities.hashtags as name,count(entities.hashtags) as coun from parquetFile where entities.hashtags is not null GROUP BY entities.hashtags")  
 //hashtags.show()  
 //hashtags.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q8")  
 //query6 location with count of users  
 val location=spark.sql("SELECT user.location,user.listed\_count from parquetFile where user.followers\_count>1000 AND user.location is not null LIMIT 10")  
 location.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q6")  
  
 //query 7 number of tweets on particular data  
 val numdate = spark.sql(sqlText="SELECT SUBSTR(created\_at, 0, 10) tweet\_date, COUNT(1) tweet\_count FROM parquetFile GROUP BY SUBSTR(created\_at, 0, 10) ORDER BY COUNT(1) DESC LIMIT 5")  
 numdate.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q7")  
  
 //query8 count of keyword used in hashtags  
 //val key=spark.sql(sqlText="SELECT 'twitter' as name,count(urls.expanded\_url) as cunt from parquetFile where url.expanded\_url like '%twitter%'")  
 //val key=spark.sql(sqlText="SELECT 'Apple' as name,count(entities.hashtags[0].text) as cunt from parquetFile where text like '%Apple%'")  
 //key.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q18")  
 val x=spark.sql("SELECT CASE WHEN entities.hashtags[0].text like '%Apple%' THEN 'Apple' WHEN entities.hashtags[0].text like '%iphone%' THEN 'iphone' WHEN entities.hashtags[0].text like '%nike%' THEN 'nike' WHEN entities.hashtags[0].text like '%amazon%' THEN 'amazon'WHEN entities.hashtags[0].text like '%trump%' THEN 'trump'END AS names FROM parquetFile")  
 x.createOrReplaceTempView("stringnames")  
 val y=spark.sql("SELECT names,count(names) as count from stringnames where names is NOT NULL group by names order by count DESC")  
 y.coalesce(1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q8-1")  
  
 //query9 count of keywords  
 val a=spark.sql("SELECT 'iphone' as brand1,count(text) as count from parquetFile where text like '%iphone%'")  
 val b=spark.sql("SELECT 'fossil' as brand1,count(text) as count from parquetFile where text like '%fossil%'")  
 val c=spark.sql("SELECT 'nike' as brand1,count(text) as count from parquetFile where text like '%nike%'")  
 val d=spark.sql("SELECT 'Apple' as brand1,count(text) as count from parquetFile where text like '%Apple%'")  
 val e=a.union(b).union(c).union(d)  
 e.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q9")  
  
 //query10 tweeter accounts created ina month  
 val month=spark.sql("SELECT substring(created\_at,1,3) as month, count(1) as count from parquetFile GROUP BY month")  
 month.coalesce(numPartitions=1).write.csv(path="C:\\Users\\sathw\\OneDrive\\Desktop\\sandeep\\q10")  
 }  
  
}

**Queries and Plot Diagrams:**

1. **Query 1:** Display day and retweet counts who used ‘apple’ in their description

This query shows day and count of retweets done on that particular day and below are attached results of scala query execution log.

**LOG:**

A screen shot of a computer

Description automatically generated

**Visualization:**

**A screenshot of a cell phone

Description automatically generated**

**Query 2:** Display username and his statuses counts who used ‘apple’ in their description

This query shows username and their statuses count who used apple as keyword in their description and below are attached results of scala query execution log.

**LOG:A screenshot of a computer screen

Description automatically generated**

**Visualization:**

**A screenshot of a social media post

Description automatically generated**

**Query 3:** Display language and their counts where text like ‘apple’ is used.

This query shows language and their count where text is like apple and ordered based on count and below are attached results of scala query execution log.

**LOG:**

**A screenshot of a computer

Description automatically generated**

**Visualization:**

**A screenshot of a cell phone

Description automatically generated**

**Query 4:** Display country and their tweets counts .

This query shows country name and their count and grouped based on location and below are attached results of scala query execution log and visualization.

**LOG:**

**A screenshot of a computer

Description automatically generated**

**Visualization:**

**A screenshot of a social media post

Description automatically generated**

**Query 5:** Display username and followers counts where text like ‘iphone’ in retweets.

This query shows username with their followers count where they used iphone as word in retweet and below are attached results of scala query execution log and visualization.

**LOG:**

**A screenshot of a computer screen

Description automatically generated**

**Visualization:**

**A screenshot of a cell phone

Description automatically generated**

**Query 6:** Display location and the user listed count where followers greater than 1000.

This query shows location name and their count of listed count field where followers greater than 1000 and location is not null and below are attached results of scala query execution log and visualization.

**LOG:**

**A screenshot of a computer

Description automatically generated**

**Visualization:**

**A screenshot of a cell phone

Description automatically generated**

**Query 7:** Display date and number of tweets count on that day.

This query shows date and their tweet count on that date as data is extracted on some dates results are bit lesser and below are attached results of scala query execution log and visualization.

**LOG:**

**A screen shot of a computer

Description automatically generated**

**Visualization:**

**A screenshot of a social media post

Description automatically generated**

**Query 8:** Display keyword appeared in hashtag and their count of total appearance .

This query shows keyword and their count based on its appearance in hashtag where used case to consider keywords and below are attached results of scala query execution log and visualization.

**LOG:**

**A screenshot of a computer

Description automatically generated**

**Visualization:**

**A screenshot of a cell phone

Description automatically generated**

**Query 9:** Display some keywords and their count of appearance in complete json data.

This query shows keyword and their count of existence in complete file where used union to combine individual queries and below are attached results of scala query execution log and visualization.

**LOG:**

**A screenshot of a cell phone

Description automatically generated**

**Visualization:**

**A screenshot of a cell phone

Description automatically generated**

**Query 10:** Display month and count of tweeter accounts created in that month

This query shows day and month name and their counts and group them by month as I have data extracted limited we have lesser results and below are attached results of scala query execution log and visualization.

**LOG:**

**Visualization:**

A screenshot of a cell phone

Description automatically generated

**WebSiteView:** Arranged each three queries in a row with button clickable on it to execute that query and display the visualization. After visualization we can click button to go back to front page. Below is front page screenshot after launching application and sample query visualization.

**Frontpage:**

**A screenshot of a computer screen

Description automatically generated**

**Query visualization after clicking the button:**

**A screenshot of a cell phone

Description automatically generated**

**Sample Visualization code:**

import matplotlib.pyplot as plt

import matplotlib

import pandas as pd

import numpy as np

import glob

f = glob.glob("q1\\*.csv")

df = pd.read\_csv(f[0],delimiter=',',names=['day','count'])

all\_data = pd.DataFrame()

all\_data = all\_data.append(df,ignore\_index=True)

labels = all\_data['day']

print(labels)

sizes = all\_data['count']

# Data to plot

#labels = ['Non-verified-accounts', 'Verified-Accounts']

#sizes = [204260, 1008]

colors = ['green', 'orange','yellow','red','blue','pink','violet']

# Plot

plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', shadow=True, startangle=140)

plt.axis('equal')

plt.title('Count of users used apple in their description on that day')

plt.show()