M2_Lab_2

Sudheer Nimmagadda edited this page Jul 22, 2019 · 3 revisions Jump to bottom

This is the wiki page for Module-2 Lab-2.

1. Name: Sudheer Nimmagadda (class ID: 21)

2. Name: Jaya Prakash Ravella (class ID: 25)

Team ID: 5

Objective:

This Lab assignment gives hands on experience using spark in scala as well as pyspark where you will get to know from basic spark programming to spark graph algorithms.

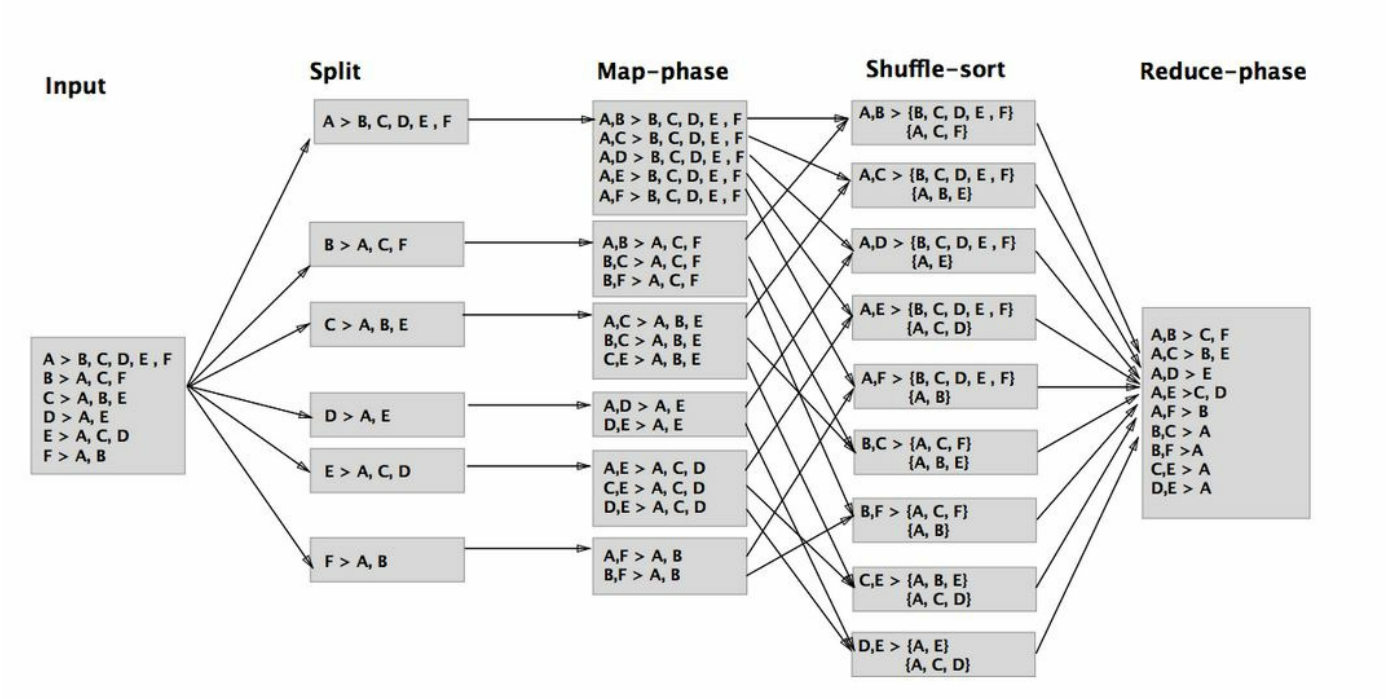
Tools Used:

1. Intellij Idea
2. Pycharm IDE
3. Windows cmd

Questions-Solutions:

Q1. Hadoop Map Reduce Algorithm to find facebook mutual friends.

Below is the Basic block diagram for map reduce,



The code is written in pyspark and the screenshots are attached below,

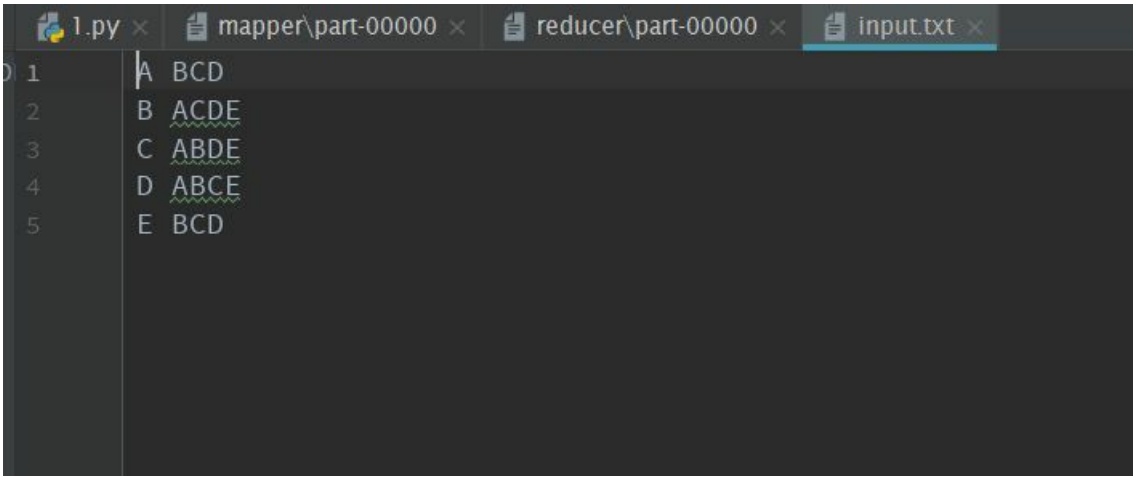
```
from pyspark import SparkContext

def mapper(theString):
    theString = theString.split(" ")
    user = theString[0]
    friends = theString[1]
    keyvalues = []

    for char in friends:
        keyvalues.append((''.join(sorted(user+char)), friends.replace(char, "")))

    return keyvalues


def reducer(a, b):
    newString = ''
    for char in a:
        if char in b:
            newString += char
    return newString


"""if __name__ == "__main__":
    mew = SparkContext.getOrCreate()
    lines = mew.textFile("C:\\Users\\Lenovo\\PycharmProjects\\BDP_Lab2\\input.txt", 1)
    newLines = lines.flatMap(mapper)
    newLines.saveAsTextFile("mapper")
    friends = newLines.reduceByKey(reducer)
    friends.coalesce(1).saveAsTextFile("reducer")
    mew.stop()"""
```

```
if __name__ == "__main__":
    facebook = SparkContext.getOrCreate()
    facebooklines = facebook.textFile("C:\\Users\\Lenovo\\PycharmProjects\\BDP_Lab2\\facebook_combined.txt", 1)
    facebookNewLines = facebooklines.flatMap(mapper)
    facebookNewLines.saveAsTextFile("facebookmapper")
    facebookfriends = facebookNewLines.reduceByKey(reducer)
    facebookfriends.coalesce(1).saveAsTextFile("facebookreducer")
    facebook.stop()
```

Dataset screenshot is attached below,

```
0 1
0 2
0 3
0 4
0 5
0 6
0 7
0 8
0 9
0 10
0 11
0 12
0 13
0 14
0 15
0 16
0 17
0 18
0 19
0 20
0 21
0 22
0 23
0 24
0 25
0 26
0 27
0 28
0 29
0 30
0 31
0 32
0 33
0 34
```

The above dataset given for us is hard to program and understand. so we create our own data as given in the first lab assignment.



The mapper output and reducer output for the given dataset are attached below,

1.py | facebook_combined.txt | facebookreducer\part-00000 | facebookmapper\part-00000

The file size (5.64 MB) exceeds configured limit (2.56 MB). Code insight features are not available.

```
('06', '2')
('02', '7')
('07', '2')
('02', '8')
('08', '2')
('02', '9')
('09', '2')
('03', '0')
('00', '3')
('03', '1')
('01', '3')
('03', '2')
('02', '3')
('03', '')
('03', '')
('03', '4')
('04', '3')
('03', '5')
('05', '3')
('03', '6')
('06', '3')
('03', '7')
('07', '3')
('03', '8')
('08', '3')
('03', '9')
('09', '3')
('04', '0')
('00', '4')
```

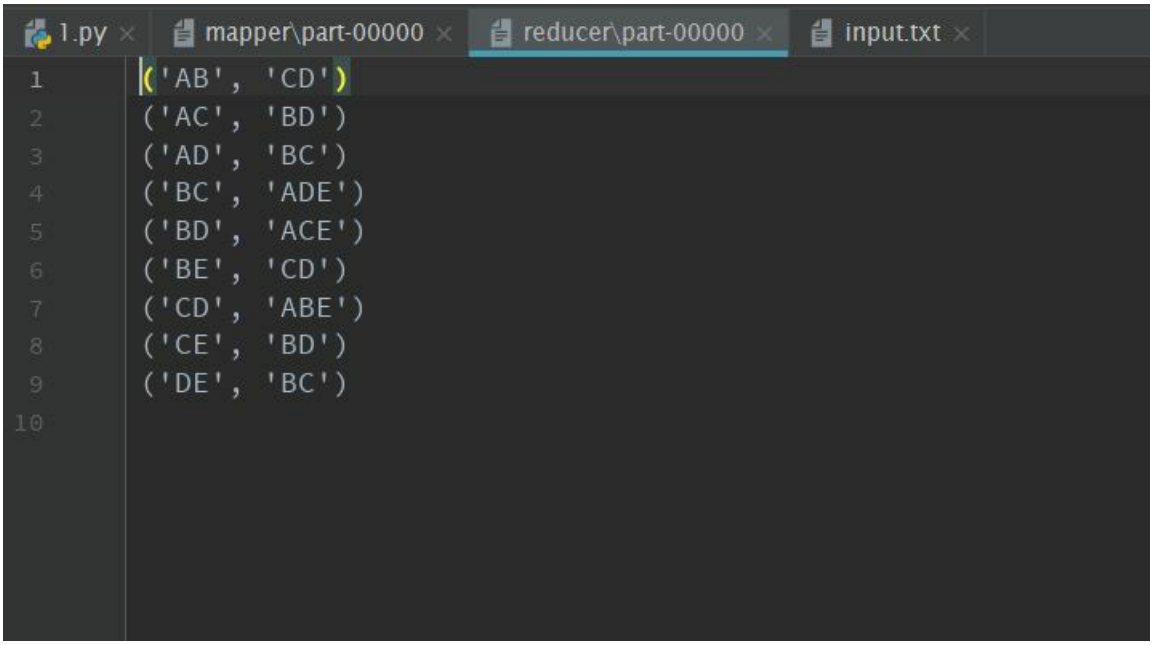
1.py | facebook_combined.txt | facebookreducer\part-00000 | facebookmapper\part-00000

```
('01', '')
('02', '')
('03', '')
('04', '')
('05', '')
('06', '')
('07', '')
('08', '')
('09', '')
('00', '')
('14', '')
('18', '')
('15', '')
('13', '')
('17', '')
('19', '')
('12', '')
('11', '')
('16', '')
('22', '')
('25', '')
('26', '')
('24', '')
('29', '')
('23', '')
('39', '')
('35', '')
('36', '')
('37', '')
```

The mapper and reducer output for own data are attached below,

1.py | mapper\part-00000 | reducer\part-00000 | input.txt

```
('AB', 'CD')
('AC', 'BD')
('AD', 'BC')
('AB', 'CDE')
('BC', 'ADE')
('BD', 'ACE')
('BE', 'ACD')
('AC', 'BDE')
('BC', 'ADE')
('CD', 'ABE')
('CE', 'ABD')
('AD', 'BCE')
('BD', 'ACE')
('CD', 'ABE')
('DE', 'ABC')
('BE', 'CD')
('CE', 'BD')
('DE', 'BC')
```



Q2. Spark Data Frames on FIFA world cup dataset

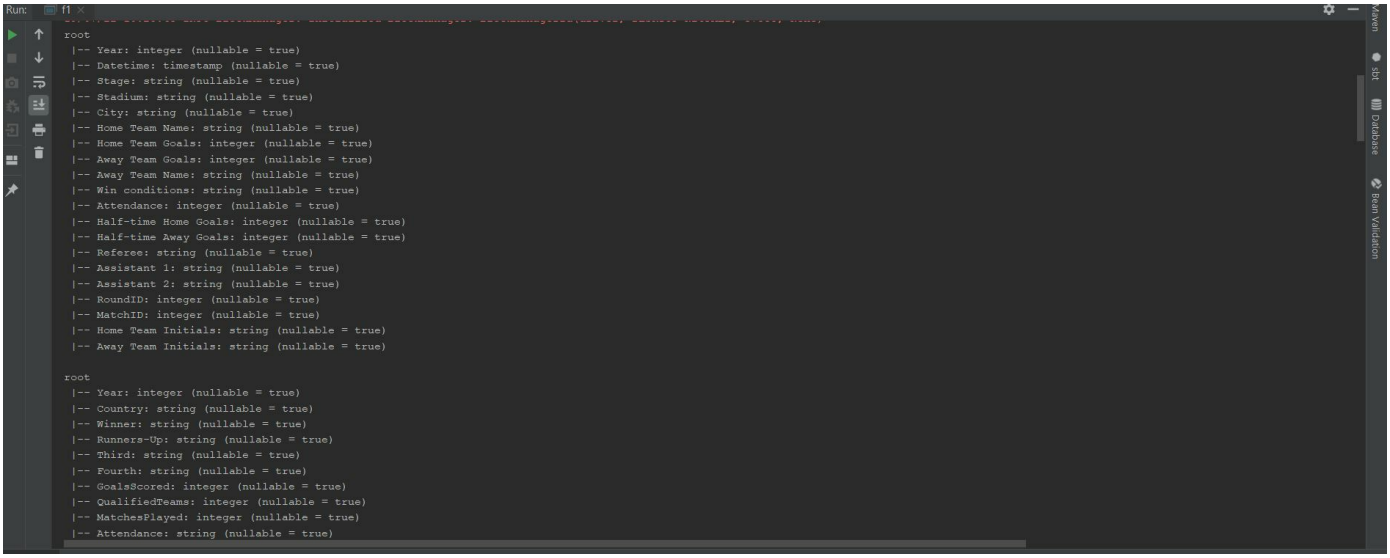
a) As specified in the question we first created a structfield type and then we used that schema on dataframe. The screenshots are attached below.

```scala
val customSchema = StructType(List(
  StructField("Year", IntegerType, true),
StructField("Datetime", TimestampType, true),
StructField("Stage", StringType, true),
StructField("Stadium", StringType, true),
StructField("City", StringType, true),
StructField("Home Team Name", StringType, true),
StructField("Home Team Goals", IntegerType, true),
StructField("Away Team Goals", IntegerType, true),
StructField("Away Team Name", StringType, true),
StructField("Win conditions", StringType, true),
StructField("Attendance", IntegerType, true),
StructField("Half-time Home Goals", IntegerType, true),
StructField("Half-time Away Goals", IntegerType, true),
StructField("Referee", StringType, true),
StructField("Assistant 1", StringType, true),
StructField("Assistant 2", StringType, true),
StructField("RoundID", IntegerType, true),
StructField("MatchID", IntegerType, true),
StructField("Home Team Initials", StringType, true),
StructField("Away Team Initials", StringType, true)))
```

```scala
val customSchema2= StructType( List(
  StructField("Year", IntegerType,true),
  StructField("Country", StringType,true),
  StructField("Winner", StringType,true),
  StructField("Runners-Up",StringType,true),
  StructField("Third", StringType,true),
  StructField("Fourth", StringType,true),
  StructField("GoalsScored", IntegerType,true),
  StructField("QualifiedTeams",IntegerType,true),
  StructField("MatchesPlayed",IntegerType,true),
  StructField("Attendance",StringType,true)
))
```

```scala
val df_match = spark.read.format( source = "csv").option("header","true").schema(customSchema).load( path = "C:\\Users\\Lenovo\\IdeaProjects\\M2_Lab2_2\\WorldCupMatches.csv")
val df_final = spark.read.format( source = "csv").option("header","true").schema(customSchema2).load( path = "C:\\Users\\Lenovo\\IdeaProjects\\M2_Lab2_2\\WorldCups.csv")
```

The output for the above schema look like below,



```
root
 |-- Year: integer (nullable = true)
 |-- Country: string (nullable = true)
 |-- Winner: string (nullable = true)
 |-- Runners-Up: string (nullable = true)
 |-- Third: string (nullable = true)
 |-- Fourth: string (nullable = true)
 |-- GoalsScored: integer (nullable = true)
 |-- QualifiedTeams: integer (nullable = true)
 |-- MatchesPlayed: integer (nullable = true)
 |-- Attendance: string (nullable = true)

root
 |-- RoundID: string (nullable = true)
 |-- MatchID: string (nullable = true)
 |-- Team Initials: string (nullable = true)
 |-- Coach Name: string (nullable = true)
 |-- Line-up: string (nullable = true)
 |-- Shirt Number: string (nullable = true)
 |-- Player Name: string (nullable = true)
 |-- Position: string (nullable = true)
 |-- Event: string (nullable = true)
```

b) Perform 10 intuitive questions in Dataset.

Query-1: Most Number of goals scored by teams in finals.

```
//Perform  10  intuitive  questions  in  Dataset
//Q1. Teams that scored most number of goals in finals
val df20 = df_match.filter(df_match("Stage").isin( list = "Final")).select(df_match("Home Team Name"),df_match("Home Team Goals")).groupBy(df_match("Home Team Name")).agg(sum(df_match("Home Te
```

```
+--------------+-----+
|Home Team Name|goals|
+--------------+-----+
|        Brazil|   12|
|         Italy|   10|
|     Argentina|    6|
|       Uruguay|    4|
|    Germany FR|    4|
|       England|    4|
|       Germany|    2|
|   Netherlands|    1|
+--------------+-----+
```

Query-2: Calculating the average no:of goals scored in worldcups by teams.

```
//Q2. Average number of goals scored by a team in worldcups
val Average_goals = spark.sql( sqlText = "SELECT Country AS Teams, ROUND(AVG(GoalsScored),0) AS average_goals FROM table2 GROUP BY Country")
Average_goals.show()
```

```
+------------+-------------+
|       Teams|average_goals|
+------------+-------------+
|      Sweden|        126.0|
|     Germany|        122.0|
|      France|        128.0|
|   Argentina|        102.0|
| Korea/Japan|        161.0|
|       Chile|         89.0|
|       Italy|         93.0|
|       Spain|        146.0|
|         USA|        141.0|
|     Uruguay|         70.0|
|      Mexico|        114.0|
| Switzerland|        140.0|
|      Brazil|        130.0|
|     England|         89.0|
|South Africa|        145.0|
+------------+-------------+
```

Query-3: Calculating the most no:of times countries hosted in world cups.

```
//Q3. Most number of times a country has hosted Worldcups
val df21 =df_final.groupBy( col1 = "Country").count().orderBy(desc( columnName = "count")).show()
```

```
+-------------+-----+
|      Country|count|
+-------------+-----+
|        Italy|    2|
|      Germany|    2|
|       Mexico|    2|
|       Brazil|    2|
|       France|    2|
|       Sweden|    1|
|    Argentina|    1|
|        Chile|    1|
|      Uruguay|    1|
| Korea/Japan|    1|
|      England|    1|
|        Spain|    1|
| Switzerland|    1|
|South Africa|    1|
|          USA|    1|
+-------------+-----+
```

Query-4: Most no:of matches won by players in world cups.

```
//Q4. Players who has won most matches in Worldcups
val df14 = df_match.filter(df_match("Stage").isin( list = "Final")).select(df_match("Year"),when(df_match("Home Team Goals") < df_match("Away Team Goals"),df_match("Away Team Name")).otherwise(df_mat
val df15 = df_player.select(df_player("RoundID"),df_player("MatchID"),df_player("Coach Name"),df_player("Team Initials"))
val df16 = df14.join(df15,(df14("RoundID") <=> df15("RoundID") && df14("MatchID") <=> df15("MatchID") && df14("initials") <=> df15("Team Initials"))).select(df15("Coach Name").alias( alias = "coach"),
val df17 = df16.distinct().groupBy( col1 = "coach").agg(count( columnName = "*").alias( alias = "cnt")).orderBy(desc( columnName = "cnt")).show()
```

```
+--------------------+---+
|               coach|cnt|
+--------------------+---+
|POZZO Vittorio (ITA)|  2|
|BILARDO Carlos (ARG)|  1|
|SUPPICI Alberto (...|  1|
|    RAMSEY Alf (ENG)|  1|
|MOREIRA Aymore (BRA)|  1|
|LIPPI Marcello (ITA)|  1|
|  LOEW Joachim (GER)|  1|
| SCHOEN Helmut (FRG)|  1|
|BECKENBAUER Franz...|  1|
| FEOLA Vicente (BRA)|  1|
|  JACQUET Aime (FRA)|  1|
|  BEARZOT Enzo (ITA)|  1|
|PARREIRA Carlos A...|  1|
|MENOTTI Cesar Lui...|  1|
|HERBERGER Sepp (FRG)|  1|
| ZAGALLO Mario (BRA)|  1|
|DEL BOSQUE Vicent...|  1|
|SCOLARI Luiz Feli...|  1|
+--------------------+---+
```

Query-5: Most no:of worldcups won by players.

```
//Q5. Players who won most number of Worldcups
val df10 = df_match.filter(df_match("Stage").isin( list = "Final")).select(df_match("Year"),when(df_match("Home Team Goals") < df_match("Away Team Goals"),df_match("Away Team Name")).otherwise(d
val df11 = df_player.select(df_player("RoundID"),df_player("MatchID"),df_player("Player Name"),df_player("Team Initials"))
val df12 = df10.join(df11,(df10("RoundID") <=> df11("RoundID") && df10("MatchID") <=> df11("MatchID") && df10("initials") <=> df11("Team Initials"))).select(df11("Player Name").alias( alias = "P
val df13 = df12.distinct().groupBy( col1 = "player").agg(count( columnName = "*").alias( alias = "cnt")).orderBy(desc( columnName = "cnt")).show()
```

```
+--------------------+---+
|              player|cnt|
+--------------------+---+
|PEL (Edson Arant...|  3|
|                DIDI|  2|
|       NILTON SANTOS|  2|
|    Eraldo MONZEGLIO|  2|
|            CASTILHO|  2|
|             RONALDO|  2|
|       Mario ZAGALLO|  2|
|                PEPE|  2|
|                VAVA|  2|
|GILMAR (Gilmar Do...|  2|
|           GARRINCHA|  2|
|         MAURO RAMOS|  2|
|       Guido MASETTI|  2|
|             BELLINI|  2|
|    Giovanni FERRARI|  2|
|       DJALMA SANTOS|  2|
|                ZITO|  2|
|   Daniel PASSARELLA|  2|
|                DIDA|  2|
|                CAFU|  2|
+--------------------+---+
only showing top 20 rows
```

Query-6: Goals scored greater than 3 by home teams.

```
//Q6. Home teams who scored goals greater than 3
val Goals = spark.sql( sqlText = "SELECT Stage,Stadium,City,Home_Team_Name FROM table4 WHERE Home_Team_Goals >= 3 AND Home_Team_Goals <= 10")
Goals.show()
```

```
+-----------------+--------------------+-----------+--------------+
|            Stage|             Stadium|       City|Home_Team_Name|
+-----------------+--------------------+-----------+--------------+
|          Group 1|             Pocitos|Montevideo |        France|
|          Group 4|      Parque Central|Montevideo |           USA|
|          Group 3|             Pocitos|Montevideo |       Romania|
|          Group 1|      Parque Central|Montevideo |         Chile|
|          Group 2|      Parque Central|Montevideo |    Yugoslavia|
|          Group 4|      Parque Central|Montevideo |           USA|
|          Group 1|  Estadio Centenario|Montevideo |     Argentina|
|          Group 2|  Estadio Centenario|Montevideo |        Brazil|
|          Group 3|  Estadio Centenario|Montevideo |       Uruguay|
|          Group 1|  Estadio Centenario|Montevideo |     Argentina|
|      Semi-finals|  Estadio Centenario|Montevideo |     Argentina|
|      Semi-finals|  Estadio Centenario|Montevideo |       Uruguay|
|            Final|  Estadio Centenario|Montevideo |       Uruguay|
|Preliminary round|Stadio Benito Mus...|     Turin |       Austria|
|Preliminary round|   Giorgio Ascarelli|    Naples |       Hungary|
|Preliminary round|            San Siro|     Milan |   Switzerland|
|Preliminary round|           Littorale|   Bologna |        Sweden|
|Preliminary round|      Giovanni Berta|  Florence |       Germany|
|Preliminary round|      Luigi Ferraris|     Genoa |         Spain|
|Preliminary round|       Nazionale PNF|      Rome |         Italy|
+-----------------+--------------------+-----------+--------------+
only showing top 20 rows
```

Query-7: Most no:of goals scored by teams in world cups.

```
//Q7. Teams scored most number of goals in Worldcups
val df1 = df_match.select(df_match("Home Team Name").alias( alias = "team"),df_match("Home Team Goals").alias( alias = "goals"))
val df2 = df_match.select(df_match("Away Team Name").alias( alias = "team"),df_match("Away Team Goals").alias( alias = "goals"))
val df3 = df1.union(df2)
val df4=df3.groupBy(df3("team")).agg(count( columnName = "*").alias( alias = "cnt")).orderBy(desc( columnName = "cnt")).filter(col( colName = "team")isNotNull).show()
```

```
+---------------+---+
|           team|cnt|
+---------------+---+
|         Brazil|108|
|          Italy| 83|
|      Argentina| 81|
|     Germany FR| 62|
|        England| 62|
|         France| 61|
|          Spain| 59|
|         Mexico| 54|
|    Netherlands| 54|
|        Uruguay| 52|
|        Germany| 48|
|         Sweden| 46|
|        Belgium| 43|
|     Yugoslavia| 37|
|          Chile| 34|
|            USA| 34|
|    Switzerland| 34|
|        Hungary| 32|
|Korea Republic| 31|
|         Poland| 31|
+---------------+---+
only showing top 20 rows
```

Query-8: Most no:of world cups where players are part of.

```
//Q8.  Players part of most number of Worldcups
val df5 = df_player.select(df_player("RoundID"),df_player("MatchID"),df_player("Player Name"))
val df6 = df_match.select(df_match("RoundID"),df_match("MatchID"),df_match("Year"))
val df7 = df5.join(df6,(df5("RoundID") <=> df6("RoundID") && df5("MatchID") <=> df6("MatchID"))).select(df5("Player Name"),df6("Year"))
val df8 = df7.distinct().groupBy(df7("Player Name")).agg(count( columnName = "*").alias( alias = "cnt")).orderBy(desc( columnName = "cnt")).show()
```

```
+-----------------+---+
|      Player Name|cnt|
+-----------------+---+
|          RONALDO|  6|
|Antonio CARBAJAL|  5|
|Enrico ALBERTOSI|  4|
|         CASTILHO|  4|
|          PAREDES|  4|
|            JONES|  4|
|          GAMARRA|  4|
|       Uwe SEELER|  4|
|    NILTON SANTOS|  4|
|             XAVI|  4|
|             SONG|  4|
|       Sepp MAIER|  4|
|       Lev YASHIN|  4|
|   Diego MARADONA|  4|
|            OSCAR|  4|
|             DIDA|  4|
|              EDU|  4|
|             LEAO|  4|
|    DJALMA SANTOS|  4|
| Dobromir JECHEV|  4|
+-----------------+---+
only showing top 20 rows
```

Query-9: Pattern recognition for the country.

```
//Q9. pattern Recognition Germany
val Patternreg = spark.sql( sqlText = "SELECT * from table2 WHERE Third LIKE 'Germany'")
Patternreg.show()
```

```
+----+------------+------+--------------+-------+--------+-----------+--------------+-------------+----------+
|Year|     Country|Winner|    Runners-Up|  Third|  Fourth|GoalsScored|QualifiedTeams|MatchesPlayed|Attendance|
+----+------------+------+--------------+-------+--------+-----------+--------------+-------------+----------+
|1934|       Italy| Italy|Czechoslovakia|Germany| Austria|         70|            16|           17|   363.000|
|2006|     Germany| Italy|        France|Germany|Portugal|        147|            32|           64| 3.359.439|
|2010|South Africa| Spain|   Netherlands|Germany| Uruguay|        145|            32|           64| 3.178.856|
+----+------------+------+--------------+-------+--------+-----------+--------------+-------------+----------+
```

Query-10: World cups won by distinct countries.

```
//Q10. Number of distinct countries who had won the Worldcup
val df22=df_final.select( col = "Winner").distinct().count()
println("Number of distinct countries who had won the worldcup:" +df22);
```

```
Number of distinct countries who had won the worldcup:9

Process finished with exit code 0
```

c) Perform any 5 queries in Spark RDD's and Spark Data Frames. Compare the results.

Query-1:

```
//Q1. Find Highest Number of Goals

//RDD
val rddgoals = data.filter(line => line.split( regex = ",")(6) != "NULL").map(line => (line.split( regex = ",")(1),
  (line.split( regex = ",")(6)))).takeOrdered( num = 10)
rddgoals.foreach(println)

// Dataframe
wc_df.select( col = "Country", cols = "GoalsScored").orderBy( sortCol = "GoalsScored").show( numRows = 10)

// Dataframe SQL
val dfGoals = spark.sql( sqlText = "select Country,GoalsScored FROM WorldCup order by GoalsScored Desc Limit 10").show()
```

Output:

```
(Argentina,102)
(Brazil,171)
(Brazil,88)
(Chile,89)
(England,89)
(France,171)
(France,84)
(Germany,147)
(Germany,97)
(Italy,115)
```

```
+------------+-----------+
|     Country|GoalsScored|
+------------+-----------+
|   Argentina|        102|
|       Italy|        115|
|      Sweden|        126|
|      Mexico|        132|
| Switzerland|        140|
|         USA|        141|
|South Africa|        145|
|       Spain|        146|
|     Germany|        147|
| Korea/Japan|        161|
+------------+-----------+
only showing top 10 rows
```

```
+-------+-----------+
|Country|GoalsScored|
+-------+-----------+
|Germany|         97|
| Mexico|         95|
|  Chile|         89|
|England|         89|
| Brazil|         88|
| France|         84|
|  Italy|         70|
|Uruguay|         70|
| France|        171|
| Brazil|        171|
+-------+-----------+
```

Query-2:

```
//Q2. Retrieve all the hosting countries who are winning countries along with the year.

//RDD
val rddvenue = data.filter(line => line.split( regex = ",")(1)==line.split( regex = ",")(2))
  .map(line => (line.split( regex = ",")(0),line.split( regex = ",")(1), line.split( regex = ",")(2)))
  .collect()

rddvenue.foreach(println)

//Dataframe
wc_df.select( col = "Year", cols = "Country","Winner").filter( conditionExpr = "Country==Winner").show( numRows = 10)

//Spark SQL
val venueDF = spark.sql( sqlText = "select Year,Country,Winner from WorldCup where Country = Winner order by Year").show()
```

Output:

```
(1930,Uruguay,Uruguay)
(1934,Italy,Italy)
(1966,England,England)
(1978,Argentina,Argentina)
(1998,France,France)
+----+---------+---------+
|Year|  Country|   Winner|
+----+---------+---------+
|1930|  Uruguay|  Uruguay|
|1934|    Italy|    Italy|
|1966|  England|  England|
|1978|Argentina|Argentina|
|1998|   France|   France|
+----+---------+---------+

+----+---------+---------+
|Year|  Country|   Winner|
+----+---------+---------+
|1930|  Uruguay|  Uruguay|
|1934|    Italy|    Italy|
|1966|  England|  England|
|1978|Argentina|Argentina|
|1998|   France|   France|
+----+---------+---------+
```

Query-3:

```
//Q3. Details of years ending in ZERO

// RDD
var years = Array("1930","1950","1970","1990","2010")

val rddwinY = data.filter(line => (line.split( regex = ",")(0)=="1950" ))
  .map(line=> (line.split( regex = ",")(0),line.split( regex = ",")(2),line.split( regex = ",")(3))).collect()

rddwinY.foreach(println)

//DataFrame
wc_df.select( col = "Year", cols = "Runners-Up","Winner").filter( conditionExpr = "Year='1950' or Year='1930' or " +
  "Year='1990' or Year='1970' or Year='2010'").show( numRows = 10)

// SQL
val winYDF = spark.sql( sqlText = "SELECT * FROM WorldCup  WHERE " +
  " Year IN ('1950','1930','1990','1970','2010') ").show()
```

Output:

```
(1950,Uruguay,Brazil)
+----+-----------+----------+
|Year| Runners-Up|    Winner|
+----+-----------+----------+
|1930|  Argentina|   Uruguay|
|1950|     Brazil|   Uruguay|
|1970|      Italy|    Brazil|
|1990|  Argentina|Germany FR|
|2010|Netherlands|     Spain|
+----+-----------+----------+

+----+------------+----------+-----------+----------+----------+-----------+-------------+-------------+----------+
|Year|     Country|    Winner| Runners-Up|     Third|    Fourth|GoalsScored|QualifiedTeams|MatchesPlayed|Attendance|
+----+------------+----------+-----------+----------+----------+-----------+-------------+-------------+----------+
|1930|     Uruguay|   Uruguay|  Argentina|       USA|Yugoslavia|         70|           13|           18|   590.549|
|1950|      Brazil|   Uruguay|     Brazil|    Sweden|     Spain|         88|           13|           22| 1.045.246|
|1970|      Mexico|    Brazil|      Italy|Germany FR|   Uruguay|         95|           16|           32| 1.603.975|
|1990|       Italy|Germany FR|  Argentina|     Italy|   England|        115|           24|           52| 2.516.215|
|2010|South Africa|     Spain|Netherlands|   Germany|   Uruguay|        145|           32|           64| 3.178.856|
+----+------------+----------+-----------+----------+----------+-----------+-------------+-------------+----------+
```

Query-4:

```
//Q4. Retrieve all the details of the World Cup match organised in 2006.

//RDD

val rddStat = data.filter(line=>line.split( regex = ",")(0)=="2006")
  .map(line=> (line.split( regex = ",")(0),line.split( regex = ",")(2),line.split( regex = ",")(3))).collect()

rddStat.foreach(println)

//Dataframe
wc_df.filter( conditionExpr = "Year=2006").show()

//Sql
spark.sql( sqlText = " Select * from WorldCup where Year == 2006 ").show()
```

Output:

```
(2006,Italy,France)
+----+-------+------+----------+-------+--------+-----------+--------------+-------------+----------+
|Year|Country|Winner|Runners-Up|  Third|  Fourth|GoalsScored|QualifiedTeams|MatchesPlayed|Attendance|
+----+-------+------+----------+-------+--------+-----------+--------------+-------------+----------+
|2006|Germany| Italy|    France|Germany|Portugal|        147|            32|           64| 3.359.439|
+----+-------+------+----------+-------+--------+-----------+--------------+-------------+----------+

+----+-------+------+----------+-------+--------+-----------+--------------+-------------+----------+
|Year|Country|Winner|Runners-Up|  Third|  Fourth|GoalsScored|QualifiedTeams|MatchesPlayed|Attendance|
+----+-------+------+----------+-------+--------+-----------+--------------+-------------+----------+
|2006|Germany| Italy|    France|Germany|Portugal|        147|            32|           64| 3.359.439|
+----+-------+------+----------+-------+--------+-----------+--------------+-------------+----------+
```

Query-5:

```
//Q5. Maximum Matches Played

//RDD

val rddMax = data.filter(line=>line.split( regex = ",")(8) == "64")
  .map(line=> (line.split( regex = ",")(0),line.split( regex = ",")(2),line.split( regex = ",")(3))).collect()

rddMax.foreach(println)

// DataFrame
wc_df.filter( conditionExpr = "MatchesPlayed == 64").show()

// Spark SQL

spark.sql( sqlText = " Select * from WorldCup where MatchesPlayed in " +
  "(Select Max(MatchesPlayed) from WorldCup )" ).show()
```

Output:

```
(1998,France,Brazil)
(2002,Brazil,Germany)
(2006,Italy,France)
(2010,Spain,Netherlands)
(2014,Germany,Argentina)
+----+------------+-------+-----------+-----------+--------------+-----------+--------------+-------------+----------+
|Year|     Country| Winner| Runners-Up|      Third|        Fourth|GoalsScored|QualifiedTeams|MatchesPlayed|Attendance|
+----+------------+-------+-----------+-----------+--------------+-----------+--------------+-------------+----------+
|1998|      France| France|     Brazil|    Croatia|   Netherlands|        171|            32|           64| 2.785.100|
|2002| Korea/Japan| Brazil|    Germany|     Turkey|Korea Republic|        161|            32|           64| 2.705.197|
|2006|     Germany|  Italy|     France|    Germany|      Portugal|        147|            32|           64| 3.359.439|
|2010|South Africa|  Spain|Netherlands|    Germany|       Uruguay|        145|            32|           64| 3.178.856|
|2014|      Brazil|Germany|  Argentina|Netherlands|        Brazil|        171|            32|           64| 3.386.810|
+----+------------+-------+-----------+-----------+--------------+-----------+--------------+-------------+----------+

+----+------------+-------+-----------+-----------+--------------+-----------+--------------+-------------+----------+
|Year|     Country| Winner| Runners-Up|      Third|        Fourth|GoalsScored|QualifiedTeams|MatchesPlayed|Attendance|
+----+------------+-------+-----------+-----------+--------------+-----------+--------------+-------------+----------+
|1998|      France| France|     Brazil|    Croatia|   Netherlands|        171|            32|           64| 2.785.100|
|2002| Korea/Japan| Brazil|    Germany|     Turkey|Korea Republic|        161|            32|           64| 2.705.197|
|2006|     Germany|  Italy|     France|    Germany|      Portugal|        147|            32|           64| 3.359.439|
|2010|South Africa|  Spain|Netherlands|    Germany|       Uruguay|        145|            32|           64| 3.178.856|
|2014|      Brazil|Germany|  Argentina|Netherlands|        Brazil|        171|            32|           64| 3.386.810|
+----+------------+-------+-----------+-----------+--------------+-----------+--------------+-------------+----------+
```

Schema used are given below for the above 5 queries.

```
root
 |-- Year: string (nullable = true)
 |-- Country: string (nullable = true)
 |-- Winner: string (nullable = true)
 |-- Runners-Up: string (nullable = true)
 |-- Third: string (nullable = true)
 |-- Fourth: string (nullable = true)
 |-- GoalsScored: string (nullable = true)
 |-- QualifiedTeams: string (nullable = true)
 |-- MatchesPlayed: string (nullable = true)
 |-- Attendance: string (nullable = true)

root
 |-- Year: string (nullable = true)
 |-- Datetime: string (nullable = true)
 |-- Stage: string (nullable = true)
 |-- Stadium: string (nullable = true)
 |-- City: string (nullable = true)
 |-- Home Team Name: string (nullable = true)
 |-- Home Team Goals: string (nullable = true)
 |-- Away Team Goals: string (nullable = true)
 |-- Away Team Name: string (nullable = true)
 |-- Win conditions: string (nullable = true)
 |-- Attendance: string (nullable = true)
 |-- Half-time Home Goals: string (nullable = true)
 |-- Half-time Away Goals: string (nullable = true)
 |-- Referee: string (nullable = true)
 |-- Assistant 1: string (nullable = true)
 |-- Assistant 2: string (nullable = true)
 |-- RoundID: string (nullable = true)
 |-- MatchID: string (nullable = true)
 |-- Home Team Initials: string (nullable = true)
 |-- Away Team Initials: string (nullable = true)
```

```
root
 |-- RoundID: string (nullable = true)
 |-- MatchID: string (nullable = true)
 |-- Team Initials: string (nullable = true)
 |-- Coach Name: string (nullable = true)
 |-- Line-up: string (nullable = true)
 |-- Shirt Number: string (nullable = true)
 |-- Player Name: string (nullable = true)
 |-- Position: string (nullable = true)
 |-- Event: string (nullable = true)
```

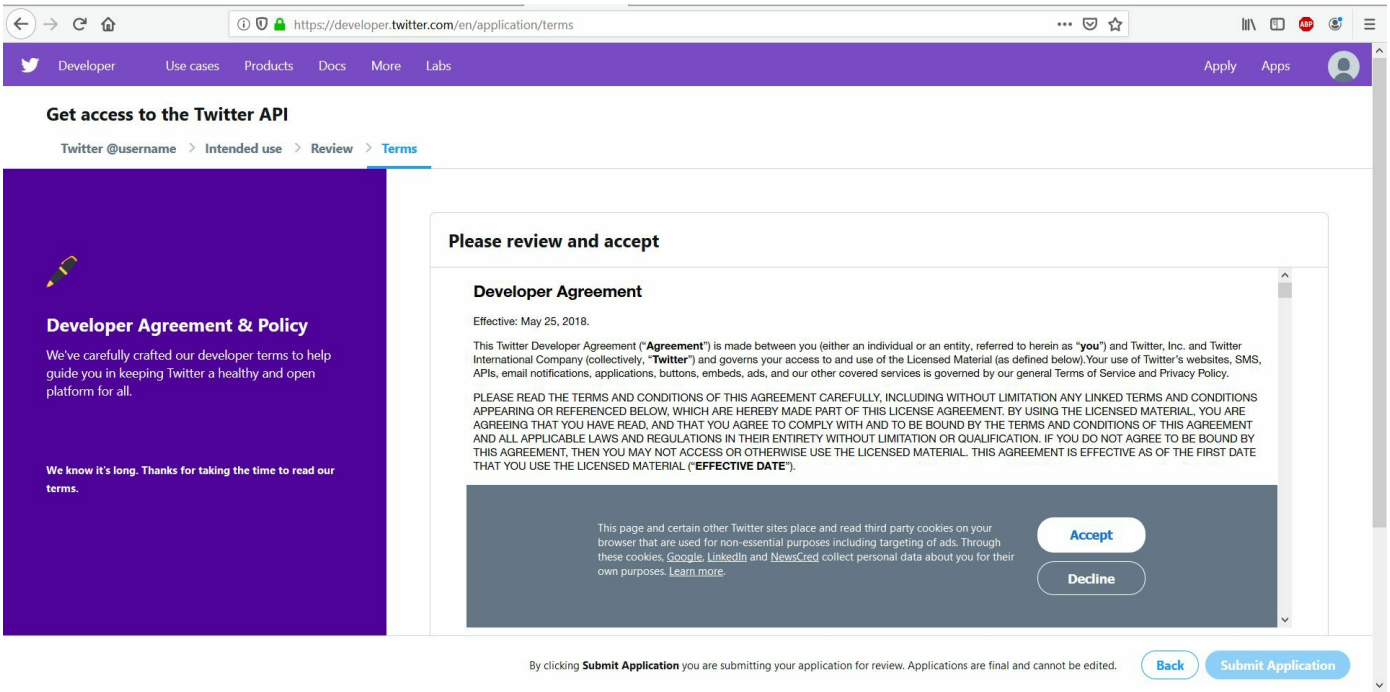
Comparison:

Comparison On	RDD	DataFrame
Data format	User has to specify schema	Organized into named Columns
Data representation	Distributed collection of data	Similar to Rdbms table
Type Safety	Compile time error	Run time error
Interoperability	Can covert from rdd to dataframe	Cannot convert from data frame to rdd
Aggregation operations	Slower	Faster

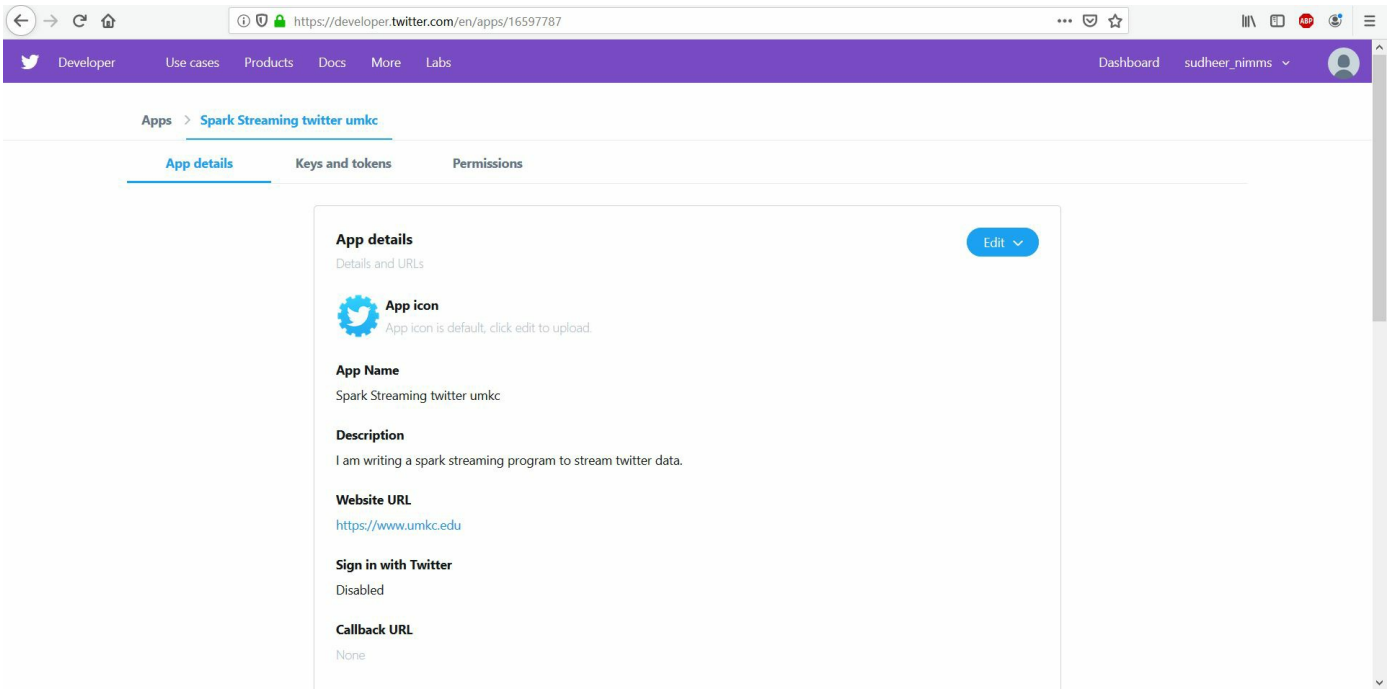
RDDs(Apache spark) offers low level functionality and control where as dataframes offer higher functionality.

Q3. Perform Word Count on Twitter Streaming Data using Spark.

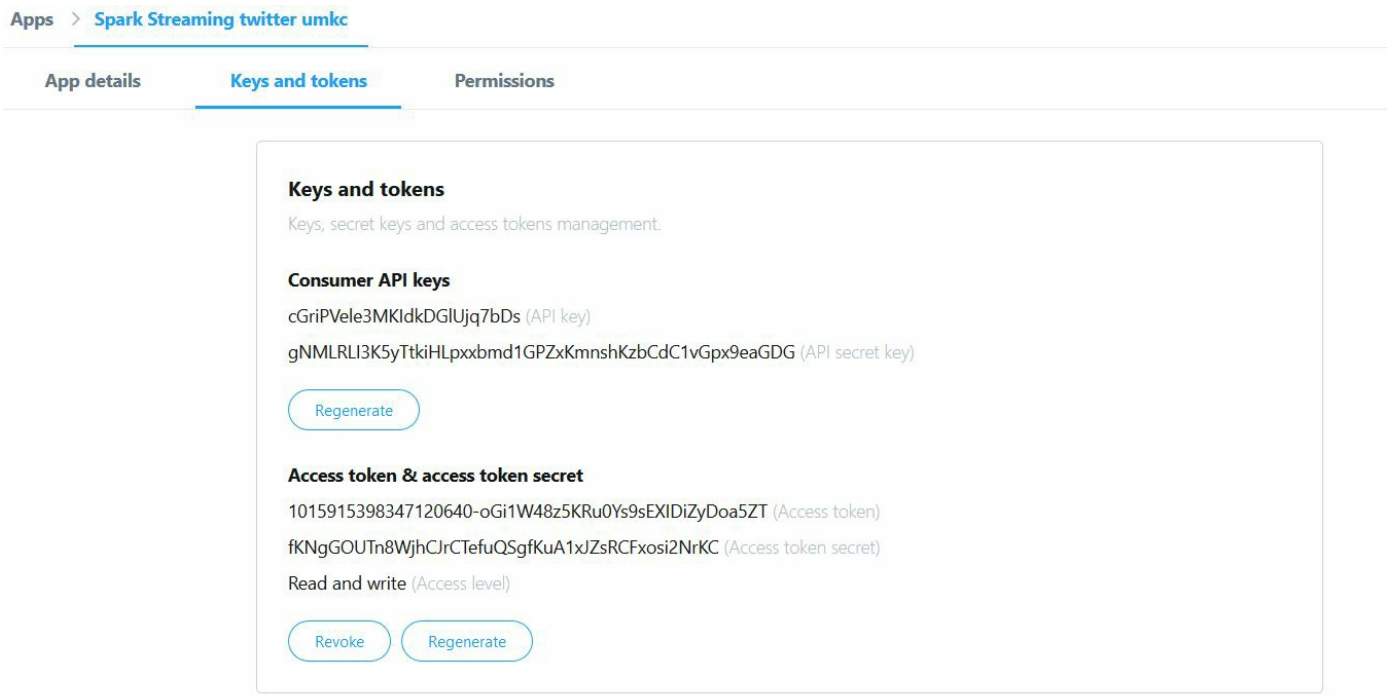
First of all you need to create twitter developer access account and fill in the required information as shown below.



After then create an app and name the app of your own as shown below.



After then at the 'Keys and tokens ' you can see the keys available to access the data.



Then below code is written in pyspark where you will write a socket binding program as well as the developer keys in python and then other program written to compute word count.

```python
import sys
import json
import time
import tweepy
import socket

CONSUMER_KEY = 'cGriPVele3MKIdkDGlUjq7bDs'
CONSUMER_SECRET = 'gNMLRLI3K5yTtkiHLpxxbmd1GPZxKmnshKzbCdC1vGpx9eaGDG'
ACCESS_TOKEN = '1015915398347120640-oGi1W48z5KRu0Ys9sEXIDiZyDoa5ZT'
ACCESS_TOKEN_SECRET = 'fKNgGOUTn8WjhCJrCTefuQSgfKuA1xJZsRCFxosj2NrKC'

def validTweet(str_tweet):
    json_tweet = json.loads(str_tweet)
    return False if list(json_tweet.keys())[0] == 'delete' or list(json_tweet.keys())[0] == 'limit' else True

class TwitterStreamListener(tweepy.StreamListener):
    def __init__(self, csocket):
        self.client_socket = csocket

    def on_data(self, data):
        if validTweet(data):
            tweet = json.loads(data)
            self.client_socket.send(tweet["text"].encode('utf-8'))

    def on_error(self, status):
        print(status)

def main():
    global CONSUMER_KEY
    global CONSUMER_SECRET
    global ACCESS_TOKEN
    global ACCESS_TOKEN_SECRET
```

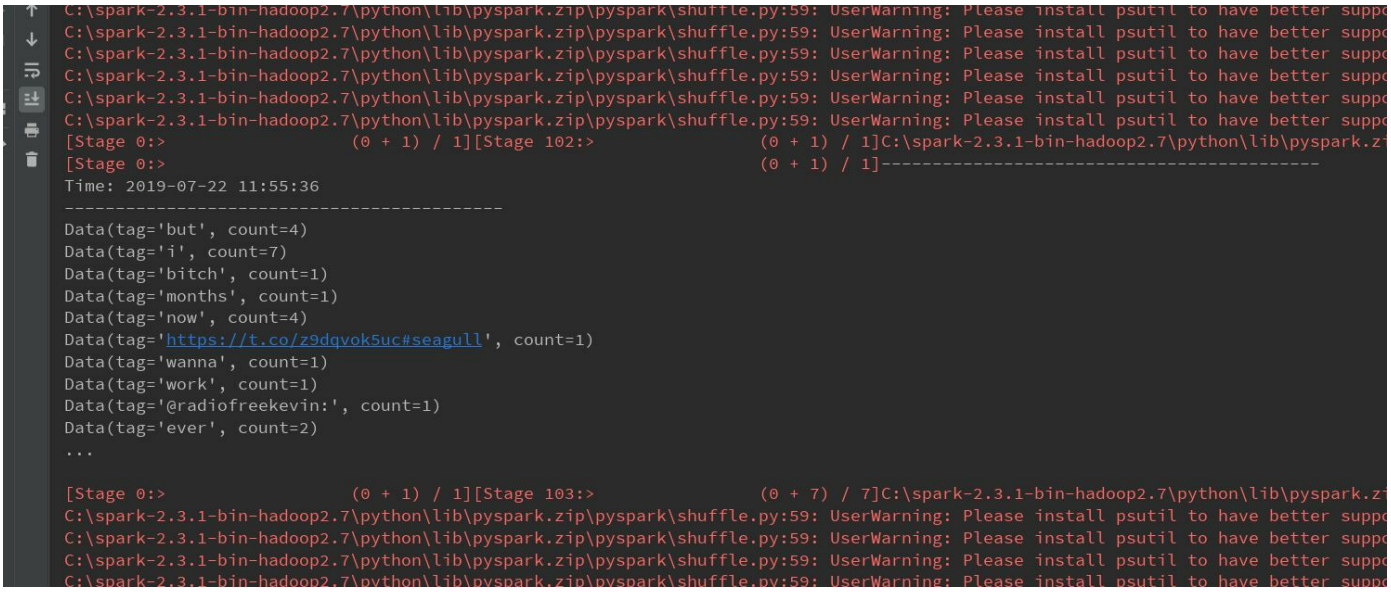
```python
def main():
    global CONSUMER_KEY
    global CONSUMER_SECRET
    global ACCESS_TOKEN
    global ACCESS_TOKEN_SECRET

    # Create socket
    s = socket.socket()
    host = 'localhost'
    port = 5000
    s.bind((host, port))
    s.listen(3)
    c_scoket, addr = s.accept()
    time.sleep(3)

    # Twitter streaming
    auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
    auth.set_access_token(ACCESS_TOKEN, ACCESS_TOKEN_SECRET)
    stream = tweepy.Stream(auth, TwitterStreamListener(c_scoket))
    stream.filter(languages=['en'], track=['spark', 'hadoop', 'python', 'hdfs', 'solr', 'cassandra', 'lucene', 'cloude

if __name__ == '__main__':
    main()
```

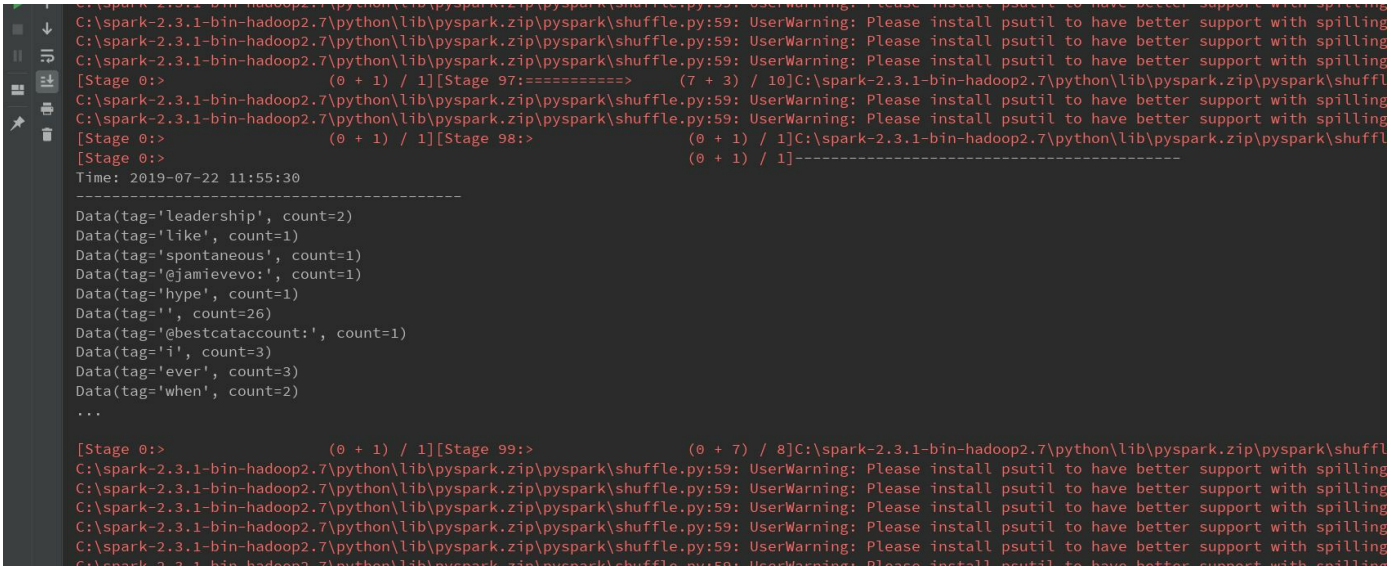
1.py 2.py

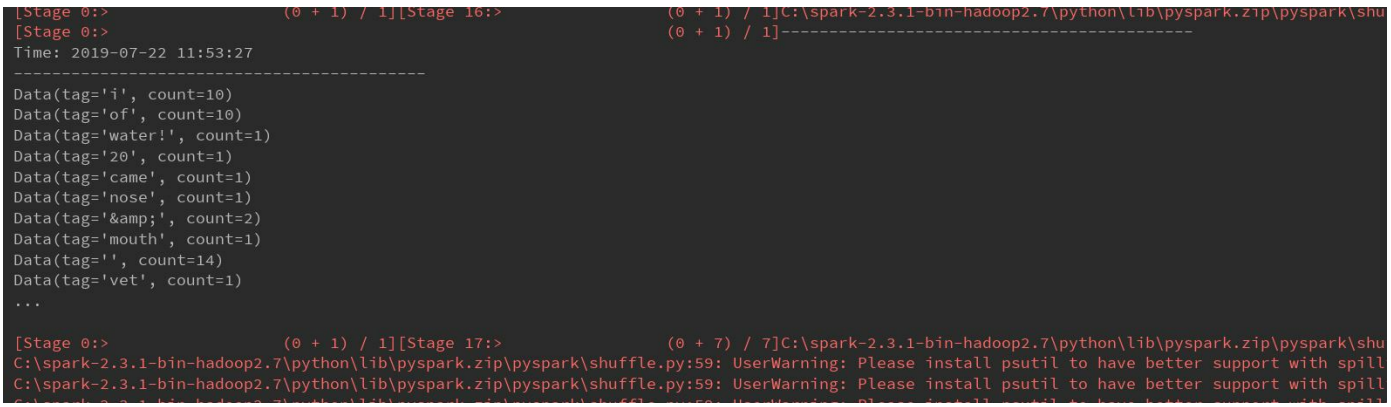
```python
from pyspark import SparkContext
from pyspark.streaming import StreamingContext
from collections import namedtuple

import os
os.environ["SPARK_HOME"] = "C:\\spark-2.3.1-bin-hadoop2.7"
os.environ["HADOOP_HOME"] = "C:\\winutils"

sc = SparkContext(appName="Lab 4")

# Change log level to error
logger = sc._jvm.org.apache.log4j
logger.LogManager.getRootLogger().setLevel(logger.Level.ERROR)

ssc = StreamingContext(sc, 3)

Tweet = namedtuple("Data", ("tag", "count"))

# Split each line into words and use map reduce to count occurance of token then print word count
ssc.socketTextStream("localhost", 5000).flatMap(lambda line: line.split(" ")).map(lambda word: (word.lower(), 1)).reduce

# Start spark streaming
ssc.start()
ssc.awaitTermination()
```

Output:

```
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
[Stage 0:>                  (0 + 1) / 1][Stage 105:==========>     (7 + 4) / 11]C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pys
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
[Stage 0:>                  (0 + 1) / 1][Stage 106:>                (0 + 1) / 1]C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pys
[Stage 0:>                                                          (0 + 1) / 1]-------------------------------------------
Time: 2019-07-22 11:55:42
-------------------------------------------
Data(tag='why', count=5)
Data(tag='he', count=3)
Data(tag='are', count=1)
Data(tag='an', count=2)
Data(tag='cunt?', count=1)
Data(tag='#trumpisaracistrapist', count=1)
Data(tag='kid', count=1)
Data(tag='set', count=1)
Data(tag='free', count=1)
Data(tag='@jamievevo:', count=2)
...

[Stage 0:>                  (0 + 1) / 1][Stage 107:>                (0 + 6) / 6]C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pys
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
C:\spark-2.3.1-bin-hadoop2.7\python\lib\pyspark.zip\pyspark\shuffle.py:59: UserWarning: Please install psutil to have better support w
```







Q4. Spark Graphx Task on Nashville meetup dataset.

a)

Below code written in pyspark where member-edges and member-meta data is taken from kaggle repository.

```python
from graphframes import *
from pyspark.sql import SparkSession
from pyspark.sql.functions import col
import sys
import os

os.environ["SPARK_HOME"] = "C:\\spark-2.3.1-bin-hadoop2.7"
os.environ["HADOOP_HOME"]="C:\\winutils"

# Create spark session
spark = SparkSession.builder.appName("Lab 4").getOrCreate()
spark.sparkContext.setLogLevel("ERROR")

# Define input path
input_path = "C:\\Users\\Lenovo\\Documents\\UMKC_SEMS\\UMKC_summer_sem\\M2_Lab2_4"

# Load vertics and edges
v = spark.read.format("csv").option("header", True).option("inferSchema", True).load(input_path + "\\meta-members.csv")\
    .select(col("member_id").alias("id"), col("name"))
e = spark.read.format("csv").option("header", True).option("inferSchema", True).load(input_path + "\\member-edges.csv")\
    .select(col("member1").alias("src"), col("member2").alias("dst"), col("weight").alias("relationship"))
# Construct graph
g = GraphFrame(v, e)
# Run PageRank until convergence to tolerance "tol"
results = g.pageRank(resetProbability=0.15, tol=0.01)
# Display resulting pageranks and final edge weights
results.vertices.select("id", "pagerank").show(10, False)
results.edges.select("src", "dst", "weight").show(10, False)
```

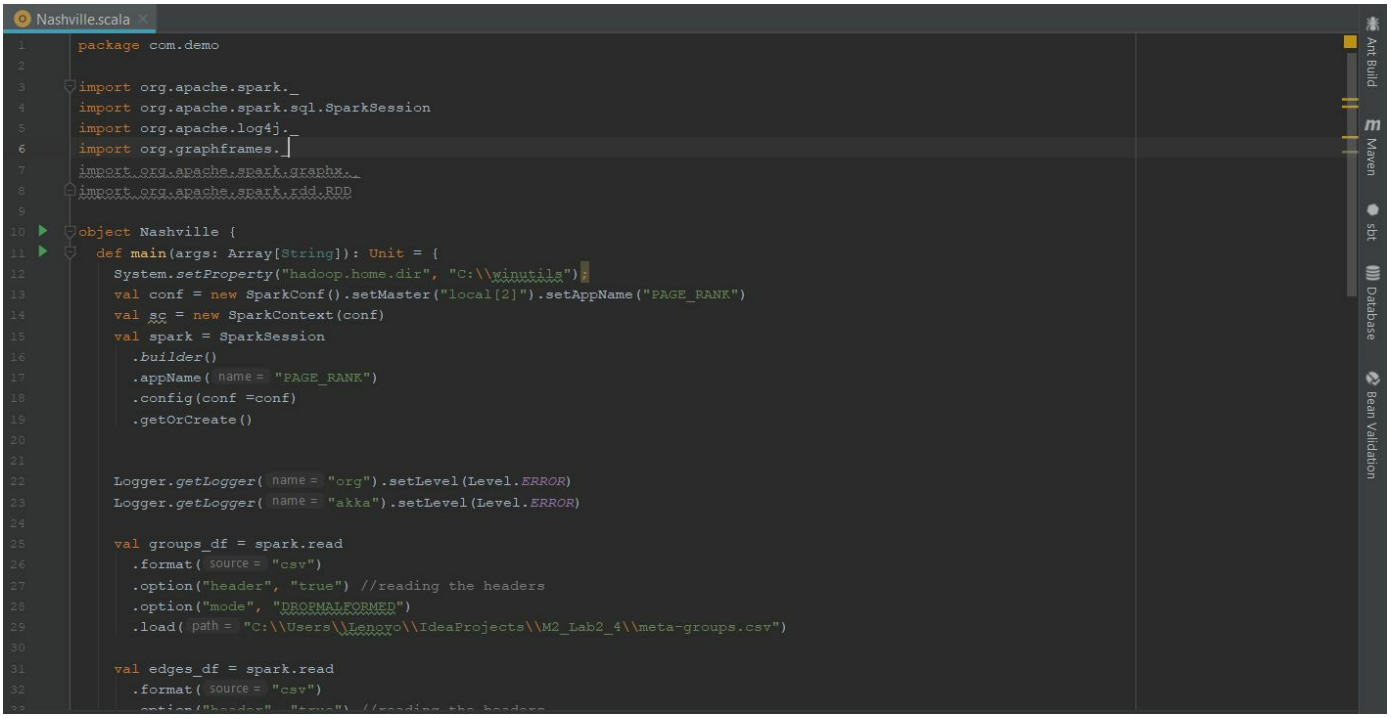
Output:

```
Administrator: Command Prompt

C:\spark-2.3.1-bin-hadoop2.7\bin>SET HADOOP_HOME="C:\winutils"

C:\spark-2.3.1-bin-hadoop2.7\bin>spark-submit --packages graphframes:graphframes:0.7.0-spark2.3-s_2.11 C:\\Users\\Lenovo\\Documents\\UMKC_SEMS\\UMKC_summer_sem\\M2_Lab2_4\\1.py
Ivy Default Cache set to: C:\Users\Lenovo\.ivy2\cache
The jars for the packages stored in: C:\Users\Lenovo\.ivy2\jars
:: loading settings :: url = jar:file:/C:/spark-2.3.1-bin-hadoop2.7/jars/ivy-2.4.0.jar!/org/apache/ivy/core/settings/ivysettings.xml
graphframes#graphframes added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-938cc7d9-b084-4342-a794-7d3380957e79;1.0
        confs: [default]
        found graphframes#graphframes;0.7.0-spark2.3-s_2.11 in spark-packages
        found org.slf4j#slf4j-api;1.7.16 in spark-list
:: resolution report :: resolve 432ms :: artifacts dl 17ms
        :: modules in use:
        graphframes#graphframes;0.7.0-spark2.3-s_2.11 from spark-packages in [default]
        org.slf4j#slf4j-api;1.7.16 from spark-list in [default]
        ---------------------------------------------------------------------
        |                  |            modules            ||   artifacts   |
        |       conf       | number| search|dwnlded|evicted|| number|dwnlded|
        ---------------------------------------------------------------------
        |      default     |   2   |   0   |   0   |   0   ||   2   |   0   |
        ---------------------------------------------------------------------

:: problems summary ::
:::: ERRORS
        unknown resolver sbt-chain


:: USE VERBOSE OR DEBUG MESSAGE LEVEL FOR MORE DETAILS
:: retrieving :: org.apache.spark#spark-submit-parent-938cc7d9-b084-4342-a794-7d3380957e79
        confs: [default]
        0 artifacts copied, 2 already retrieved (0kB/57ms)
2019-07-22 12:08:41 WARN  NativeCodeLoader:62 - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2019-07-22 12:08:44 INFO  SparkContext:54 - Running Spark version 2.3.1
2019-07-22 12:08:44 INFO  SparkContext:54 - Submitted application: Lab 4
2019-07-22 12:08:44 INFO  SecurityManager:54 - Changing view acls to: Lenovo
2019-07-22 12:08:44 INFO  SecurityManager:54 - Changing modify acls to: Lenovo
2019-07-22 12:08:44 INFO  SecurityManager:54 - Changing view acls groups to:
2019-07-22 12:08:44 INFO  SecurityManager:54 - Changing modify acls groups to:
2019-07-22 12:08:44 INFO  SecurityManager:54 - SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(Lenovo); groups with view permissions: Set(); users  with modify permi
ssions: Set(Lenovo); groups with modify permissions: Set()
2019-07-22 12:08:45 INFO  Utils:54 - Successfully started service 'sparkDriver' on port 40482.
2019-07-22 12:08:45 INFO  SparkEnv:54 - Registering MapOutputTracker
2019-07-22 12:08:45 INFO  SparkEnv:54 - Registering BlockManagerMaster
```

The page rank and the source to destination weight for the given member id's are shown below,

```
2019-07-22 12:08:46 INFO  BlockManager:54 - Initialized BlockManager: BlockManagerId(driver, DESKTOP-N2GPME1, 40523, None)
2019-07-22 12:08:46 INFO  ContextHandler:781 - Started o.s.j.s.ServletContextHandler@4af0c448{/metrics/json,null,AVAILABLE,@Spark}
2019-07-22 12:08:47 INFO  SharedState:54 - Setting hive.metastore.warehouse.dir ('null') to the value of spark.sql.warehouse.dir ('file:/C:/spark-2.3.1-bin-hadoop2.7/bin/spark-wa
2019-07-22 12:08:47 INFO  SharedState:54 - Warehouse path is 'file:/C:/spark-2.3.1-bin-hadoop2.7/bin/spark-warehouse/'.
2019-07-22 12:08:47 INFO  ContextHandler:781 - Started o.s.j.s.ServletContextHandler@29bc6937{/SQL,null,AVAILABLE,@Spark}
2019-07-22 12:08:47 INFO  ContextHandler:781 - Started o.s.j.s.ServletContextHandler@1f01cf8b{/SQL/json,null,AVAILABLE,@Spark}
2019-07-22 12:08:47 INFO  ContextHandler:781 - Started o.s.j.s.ServletContextHandler@77bfadf3{/SQL/execution,null,AVAILABLE,@Spark}
2019-07-22 12:08:47 INFO  ContextHandler:781 - Started o.s.j.s.ServletContextHandler@3b97383b{/SQL/execution/json,null,AVAILABLE,@Spark}
2019-07-22 12:08:47 INFO  ContextHandler:781 - Started o.s.j.s.ServletContextHandler@40e4f7fc{/static/sql,null,AVAILABLE,@Spark}
2019-07-22 12:08:49 INFO  StateStoreCoordinatorRef:54 - Registered StateStoreCoordinator endpoint
+---------+------------------+
|id       |pagerank          |
+---------+------------------+
|7848668  |0.7409105228055258|
|145573412|0.4191035572807095|
|211406247|0.4191035572807095|
|41985802 |0.4191035572807095|
|183125840|0.7673888142364493|
|219037159|0.7545570380374939|
|226811920|0.4191035572807095|
|188126267|0.4191035572807095|
|57507522 |4.170353085963747 |
|201765390|0.6686779835828749|
+---------+------------------+
only showing top 10 rows

+------+---------+--------------------+
|src   |dst      |weight              |
+------+---------+--------------------+
|2069  |113522242|0.0024154589371980675|
|2069  |184268293|0.0024154589371980675|
|2069  |211844922|0.0024154589371980675|
|9205  |5175848  |0.0020876826722338203|
|9205  |81196252 |0.0020876826722338203|
|9205  |207502711|0.0020876826722338203|
|9205  |210687500|0.0020876826722338203|
|9205  |219268686|0.0020876826722338203|
|106758|203089059|0.00574712643678161 |
|303336|16003291 |0.005714285714285714|
+------+---------+--------------------+
only showing top 10 rows
```

We have taken group-edges and meta-groups from kaggle repository where the code is written in scala. Below are the screenshots,



```
    val edges_df = spark.read
      .format( source = "csv")
      .option("header", "true") //reading the headers
      .option("mode", "DROPMALFORMED")
      .load( path = "C:\\Users\\Lenovo\\IdeaProjects\\M2_Lab2_4\\group-edges.csv")



    // Printing the Schema

    edges_df.printSchema()

    groups_df.printSchema()


    edges_df.createOrReplaceTempView( viewName = "e")

    groups_df.createOrReplaceTempView( viewName = "g")


    val g2 = spark.sql( sqlText = "select * from g")

    val e2 = spark.sql( sqlText = "select * from e")

    val vertices = g2
      .withColumnRenamed( existingName = "group_id",  newName = "id").limit(200)
      .distinct()

    val edges = e2
      .withColumnRenamed( existingName = "group1",  newName = "src").limit(600).distinct()
      .withColumnRenamed( existingName = "group2",  newName = "dst").limit(600).distinct()
```

```
    val vertices = g2
      .withColumnRenamed( existingName = "group_id",  newName = "id").limit(200)
      .distinct()

    val edges = e2
      .withColumnRenamed( existingName = "group1",  newName = "src").limit(600).distinct()
      .withColumnRenamed( existingName = "group2",  newName = "dst").limit(600).distinct()


    val graph = GraphFrame(vertices, edges)

    edges.cache()
    vertices.cache()
    graph.vertices.show()
    graph.edges.show()


    println("Total Number of vertices: " + graph.vertices.count)
    println("Total Number of edges: " + graph.edges.count)


    val stationPageRank = graph.pageRank.resetProbability( value = 0.15).tol( value = 0.01).run()
    stationPageRank.vertices.show()
    stationPageRank.edges.show()
  }

}
```

Schema for the vertices and edges are shown below,

```
root
 |-- _c0: string (nullable = true)
 |-- group1: string (nullable = true)
 |-- group2: string (nullable = true)
 |-- weight: string (nullable = true)

root
 |-- group_id: string (nullable = true)
 |-- group_name: string (nullable = true)
 |-- num_members: string (nullable = true)
 |-- category_id: string (nullable = true)
 |-- category_name: string (nullable = true)
 |-- organizer_id: string (nullable = true)
 |-- group_urlname: string (nullable = true)
```

Output:

id	group_name	num_members	category_id	category_name	organizer_id	group_urlname
339011	Nashville Hiking ...	15838	23	Outdoors & Adventure	4353803	nashville-hiking
19728145	Stepping Out Soci...	1778	5	Dancing	118484462	steppingoutsocial...
6335372	Nashville soccer	2869	32	Sports & Recreation	108448302	Nashville-soccer
10016242	NashJS	1975	34	Tech	8111102	nashjs
21174496	20's & 30's Women...	2782	31	Socializing	184580248	new-friends-in-Na...
11077852	Sunday Assembly N...	918	28	Religion & Beliefs	4765912	Sunday-Assembly-N...
22197221	Team Green Advent...	1812	23	Outdoors & Adventure	199336381	TeamGreenAdventures
1585196	Tennessee Hiking ...	4828	23	Outdoors & Adventure	13537265	TennesseeHikingGroup
526316	¡Diablos Que Bail...	3472	5	Dancing	12229328	diablos-que-bailan
1763190	Nashville Tennis ...	1563	32	Sports & Recreation	9890725	Nashville-Tennis-...
18243826	Middle TN 40+ sin...	2583	30	Singles	198309808	MTN-40
11625832	PyNash	1442	34	Tech	215201845	PyNash
168014	The Nashville Wri...	3286	36	Writing	1281684	nashvillewriters
19218850	Greater Nashville...	764	34	Tech	12825115	Greater-Nashville...
1526075	Nashville Area Ga...	2730	11	Games	10764011	NAGACentral
1102353	Nashville Backpac...	3861	23	Outdoors & Adventure	7528310	NashvilleBackpacker
18955830	Eat Love Nash	5008	31	Socializing	13814459	EatLoveNash
18495240	Middle Tennessee ...	1576	23	Outdoors & Adventure	183268581	Middle-Tennessee-...
18562307	Nashville Young P...	3210	2	Career & Business	8736052	Nashville-Young-P...
18589616	Agile Nashville U...	862	34	Tech	126249582	Agile-Nashville-U...

only showing top 20 rows

_c0	src	dst	weight
0	19292162	535553	2
1	19292162	19194894	1
2	19292162	19728145	1
3	19292162	18850080	2
4	19292162	1728035	1
5	19292162	22817838	2
6	19292162	19997487	2
7	19292162	18855476	2
8	19292162	18955830	1
9	19292162	11294262	1
10	19292162	1360698	2
11	19292162	1179719	1
12	19292162	1457232	1
13	19292162	13560402	1
14	19292162	7151442	1
15	19292162	18506072	1
16	19292162	16477792	2
17	19292162	20947040	1
18	19292162	5618532	1
19	19292162	11625832	5

only showing top 20 rows

```
Total Number of vertices: 200
Total Number of edges: 600
+--------+--------------------+-----------+-----------+--------------------+------------+--------------------+------------------+
|      id|          group_name|num_members|category_id|       category_name|organizer_id|       group_urlname|          pagerank|
+--------+--------------------+-----------+-----------+--------------------+------------+--------------------+------------------+
|  405938|MTRAS ~ MidTn Rob...|        525|         34|                Tech|     3246917|         robotics-71|0.9711094925952919|
|18616278|Franklin Develope...|        629|         34|                Tech|   170855672|franklin-develope...|1.0049854051276859|
|19416348|Bellevue Business...|        298|          2|   Career & Business|    83272622|Bellevue-Business...|1.0219798212481033|
|24125934|Murfreesboro Web ...|         43|         34|                Tech|   178742432|Murfreesboro-Web-...|0.9711094925952919|
|22736876|Business Connecti...|        126|          2|   Career & Business|   191532521|Brentwood-Rowdy-R...|0.9807076678128035|
|18494105|The Iron Yard - N...|       1491|         34|                Tech|   104388972|The-Iron-Yard-Nas...|1.0013437445304536|
|18529135|Franklin AM - Net...|        360|          2|   Career & Business|    34583172|Franklin-AM-Netwo...|1.0013437445304536|
|11625832|              PyNash|       1442|         34|                Tech|   215201845|              PyNash|1.0211705633376074|
|20135961|20s/30s Nashville...|       1124|         31|        Socializing|   198403977|Nashville-Online-...|0.9711094925952919|
|  535553|             Nash.rb|        881|         34|                Tech|    14344641|              nashrb|1.0115723881200958|
|19528743|Nashville Real Es...|        441|          2|   Career & Business|   144256692|Nashville-Real-Es...| 0.991745569312942|
|15335602|Brentwood TN Conv...|        315|         16|Language & Ethnic...|   153513242|Brentwood-TN-Conv...| 1.017528902740375|
|18314164|              NashBI|        784|         34|                Tech|   183427754|              NashBI|0.9807076678128035|
| 6707902|Data Science Nash...|       1046|         34|                Tech|    14589429|Data-Science-Nash...|1.0049854051276859|
|  541319|The Nashville Son...|       2644|         21|               Music|     2984170|       vocalists-164|1.0219798212481033|
|20493986|  R-Ladies Nashville|        210|          2|   Career & Business|   213434886|   rladies-nashville|0.9807076678128035|
|  339011|Nashville Hiking ...|      15838|         23|Outdoors & Adventure|     4353803|    nashville-hiking|1.0725584406541084|
|20583464|Mediumship and In...|        234|         22|New Age & Spiritu...|     5212354|Mediumship-and-In...|0.9807076678128035|
| 4126912|Nashville Online ...|       1532|          2|   Career & Business|    44942272|    nashville-online|1.0219798212481033|
|22197221|Team Green Advent...|       1812|         23|Outdoors & Adventure|   199336381| TeamGreenAdventures|1.0563732824441867|
+--------+--------------------+-----------+-----------+--------------------+------------+--------------------+------------------+
only showing top 20 rows
```

```
+---+--------+--------+------+--------------------+
|_c0|     src|     dst|weight|              weight|
+---+--------+--------+------+--------------------+
|268| 1585196|18506072|     6|0.011627906976744186|
|441| 1417288| 1498076|     1|               0.025|
|324| 1179719|19030621|     1|  0.0196078431372549|
|388| 1417288|18955830|     1|               0.025|
|364| 1179719|19934054|     1|  0.0196078431372549|
|555| 3047512|18955830|     1| 0.06666666666666667|
|371| 1179719|23674770|     3|  0.0196078431372549|
|215| 1585196|20166757|     1|0.011627906976744186|
| 23|19292162|16487812|     5|0.029411764705882353|
|495|  168014| 4126912|     1|               0.025|
|391| 1417288| 1772099|     1|               0.025|
|394| 1417288|  168014|     1|               0.025|
|417| 1417288| 1358081|     2|               0.025|
|241| 1585196| 1307837|     1|0.011627906976744186|
| 36|19292162|19654655|     1|0.029411764705882353|
|162|20135961|18506072|     3|                0.25|
|239| 1585196|15335602|     1|0.011627906976744186|
|205| 1585196|22023226|     1|0.011627906976744186|
|317| 1179719|18855476|     1|  0.0196078431372549|
|234| 1585196|11131552|     2|0.011627906976744186|
+---+--------+--------+------+--------------------+
only showing top 20 rows
```

b) State importance of using graphx on the chosen dataset.

For member-data in pyspark,

In real time page rank is used to rank web sites. It is a way of evaluating the importance of website pages. For the data set we used, we first made a vertex for each member. So in network graph each member is a node. Then we create the edge b/w two vertices and allot a weight as a relationship. With this result ready, the page rank is determined to which destination and source, it is the most popular and significant (path,node) in the network graph.

For group-data in scala,

Graphx are mainly used for distributed processing of graphs. For example, where a graph is very large with huge no:of vertices and edges then it is difficult to process on a single state machine. Then we need to use parallel computation.

Here, we used group-id to produce vertices and group-1,group-2 taken from the dataset which is used to produce edges for the graphs. In the final step a graph is produced as shown above in the code screenshots.

References:

1. https://docs.databricks.com/spark/latest/graph-analysis/graphframes/graph-analysis-tutorial.html#graph-processing-primer
2. https://docs.databricks.com/spark/latest/graph-analysis/graphframes/user-guide-scala.html#basic-graph-and-dataframe-queries
3. https://data-flair.training/blogs/apache-spark-rdd-vs-dataframe-vs-dataset/
4. https://stackoverflow.com/questions/52563151/pyspark-finding-friend-recommendations-for-users-based-on-mutual-friends
5. https://www.toptal.com/apache/apache-spark-streaming-twitter

Contributions:

Sudheer Nimmagadda-

Wiki documentation, code upload and screenshots, mapper code for facebook mutual friends and its corresponding mapper output for both datasets, creating a struct schema and processing it to a dataframe for FIFA worldcup data and first 5 queries on the FIFA world cup data, first 3 queries on FIFA worldcup data for spark RDD's vs data frames, twitter word count program and the developer access keys, pyspark program graph on member data from nashville and also its importance, half video dubbing (52%)

Prakash Ravella-

Reducer code for facebook mutual friends and its corresponding reducer output for both datasets, next 5 intuitive queries on the FIFA world cup data, next 2 queries on FIFA worldcup data for spark RDD's vs data frames, socket listening program for twitter word count, scala program graph on group data from nashville and also its importance, half video dubbing (48%)

M2_ICP_6

Show 1 more pages…

Clone this wiki locally

```
https://github.com/nimms9/2019_Summer_CSEE5590_Big_Data_Programming.wiki.git
```