**Twitter Analysis on Movies Keyword**

**Introduction:**

**AIM:**

In this project we have analyzed the tweets on twitter data based on the keyword “movies” using Spark SQL DF, RDDs. We have drawn different conclusions and visualized the results.

**Technologies and** **Tools used:**

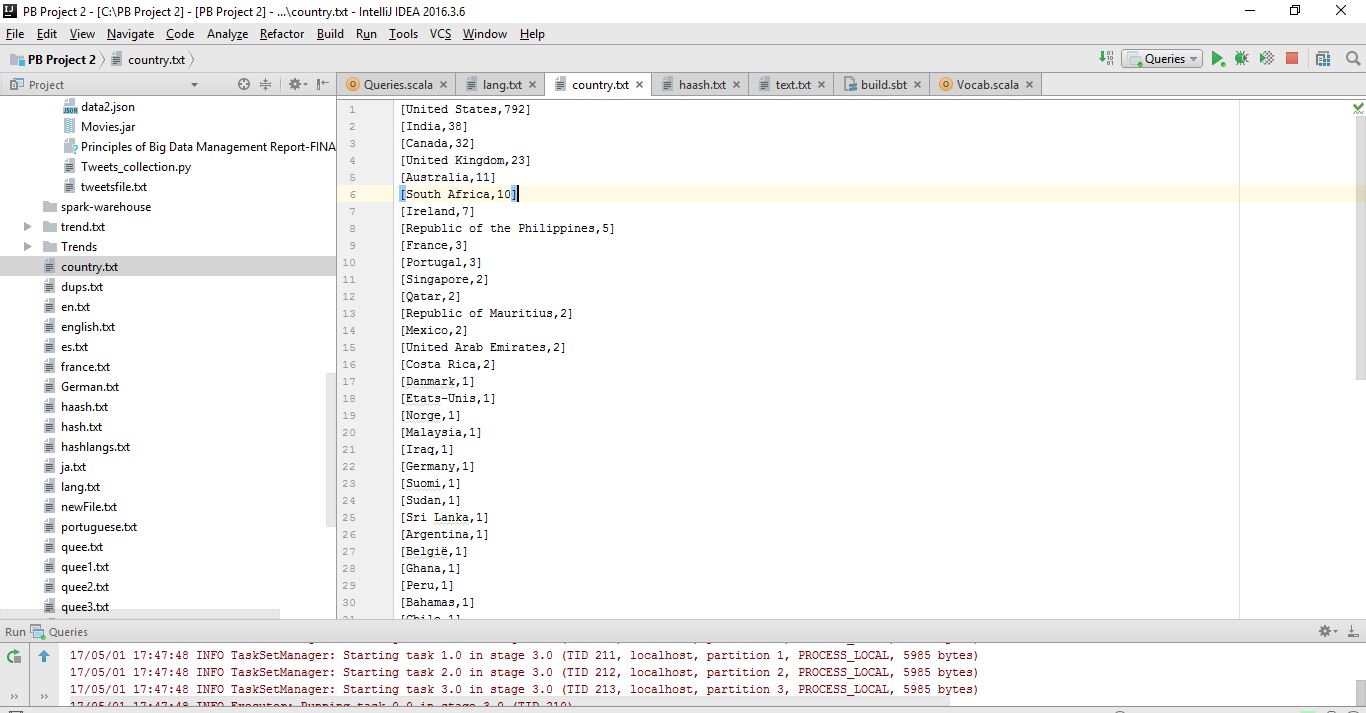
1. **Environment:** Windows 10
2. **Tools Used:** Intellij
3. **Technologies used:** Spark 2.0.1, Scala 2.11.8 and Java 1.8

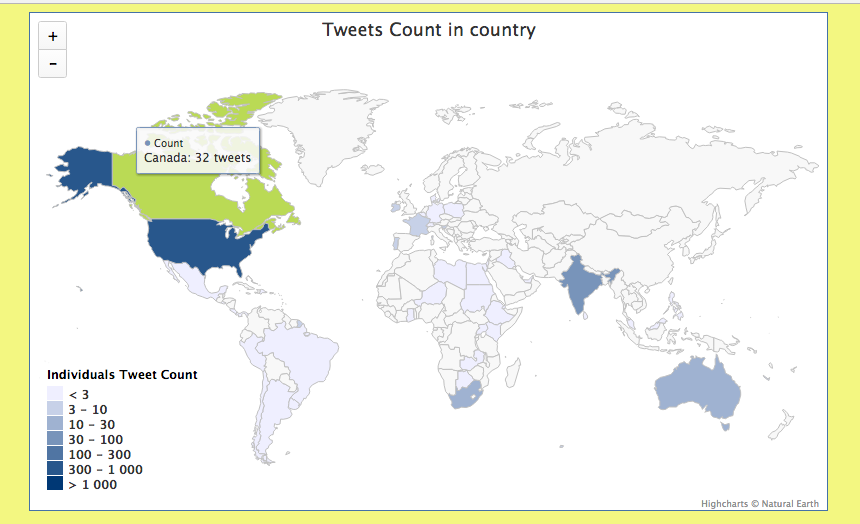
**Queries:**

**Query 1:**

Select place.country, count(\*) as country\_count from dftab where place.country IS NOT NULL GROUP BY place.country ORDER BY country\_count DESC

**Description:** This query finds the count of number of users from countries around the world who tweeted on the keyword “movies”.





**Query 2:**

//English

Select user.lang, count(\*) as en\_count from dftab where user.lang like 'en' group by user.lang

//Spanish

Select user.lang, count(\*) as es\_count from dftab where user.lang like 'es' group by user.lang

//German

Select user.lang, Count(\*) as ja\_count from dftab where user.lang like 'ja' group by user.lang

//Portuguese

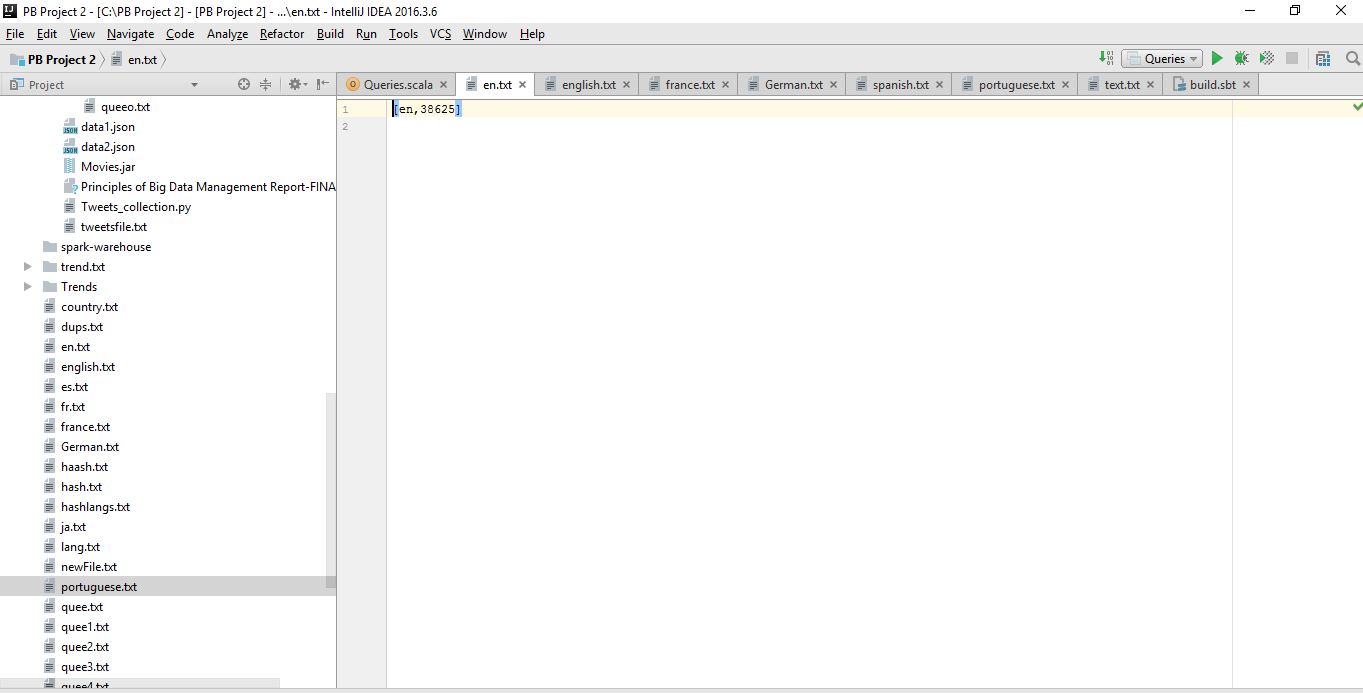
Select user.lang, count(\*) as pt\_count from dftab where user.lang like 'pt' group by user.lang

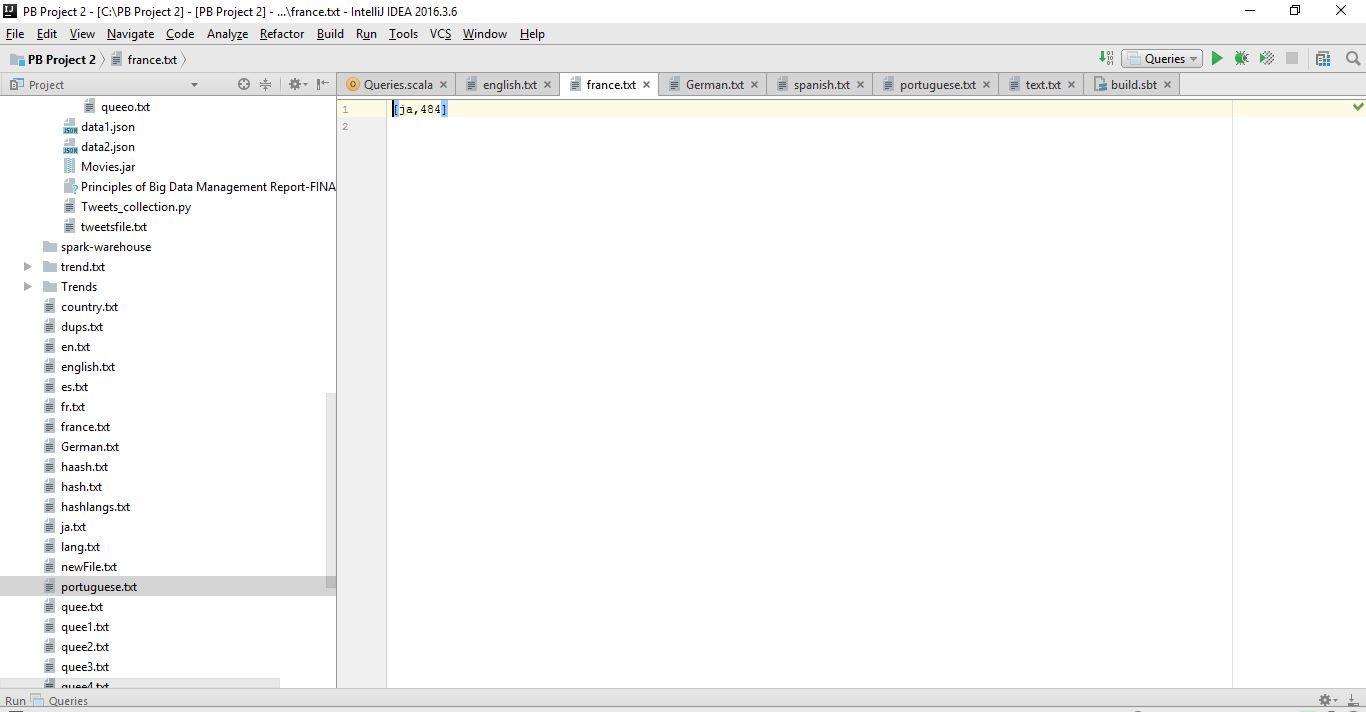
//France

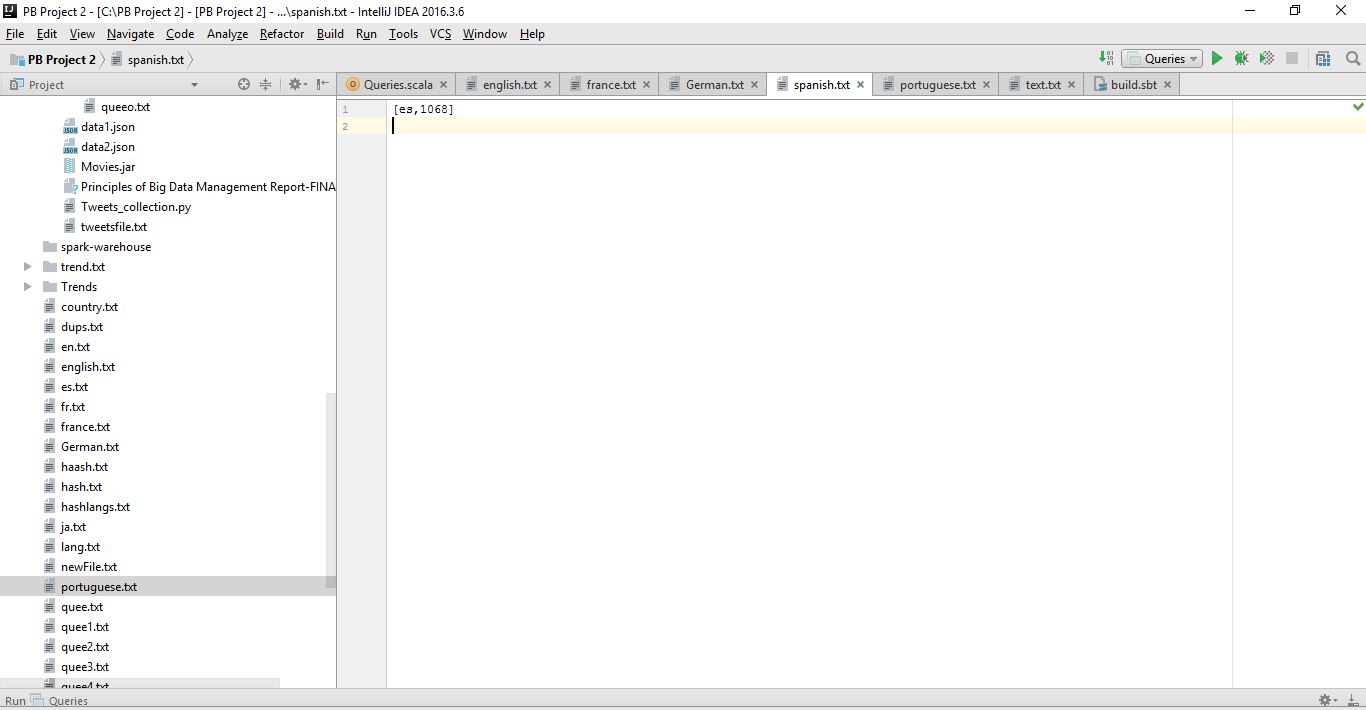
Select user.lang, Count(\*) as fr\_count from dftab where user.lang like 'fr' group by user.lang

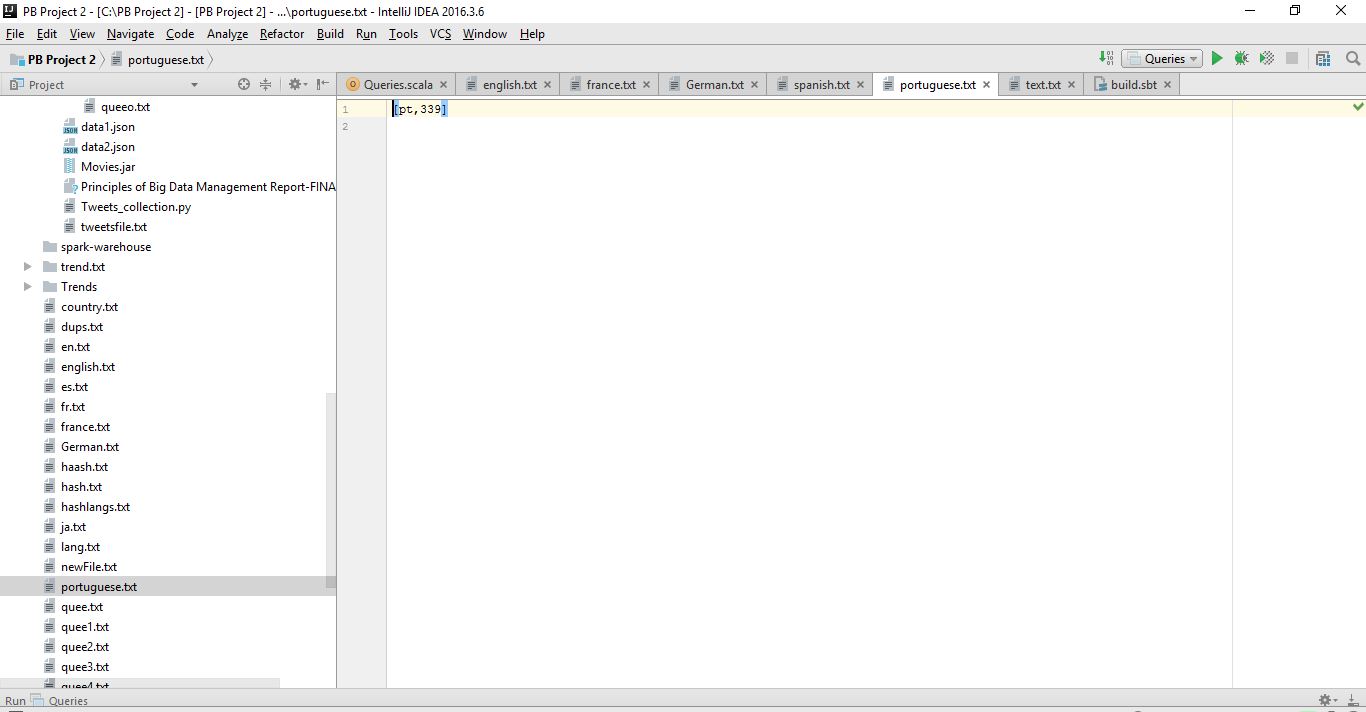
**Description:** This query finds the count of users from different languages namely Spanish,

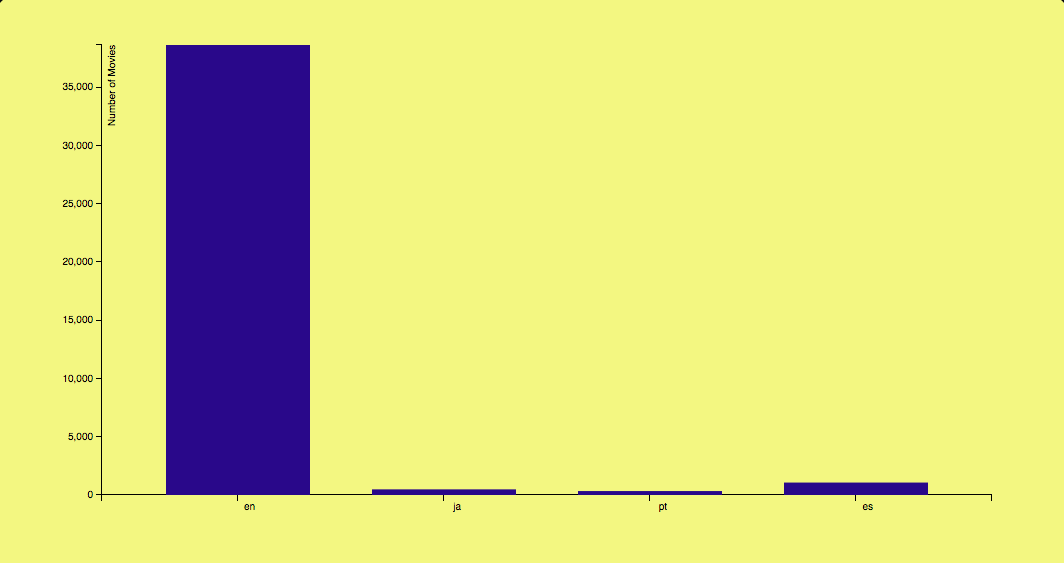
German, Portuguese, English, France who tweeted on the movies keyword.











**Query 3 using RDD:**

val horror=lines.filter(line=>line.contains("#horror")).count()

val comedy=lines.filter(line=>line.contains("#comedy")).count()

val thriller=lines.filter(line=>line.contains("#thriller")).count()

val action=lines.filter(line=>line.contains("#action")).count()

println("Horror Movies %s".format(horror))

println("Comedy Movies %s".format(comedy))

println("Thriller Movies %s".format(thriller))

println("Action Movies %s".format(action))

val movieslist = List(("horror", horror),("comedy", comedy),("thriller", thriller),("action", action))

val rdd1 = sc.parallelize(movieslist)

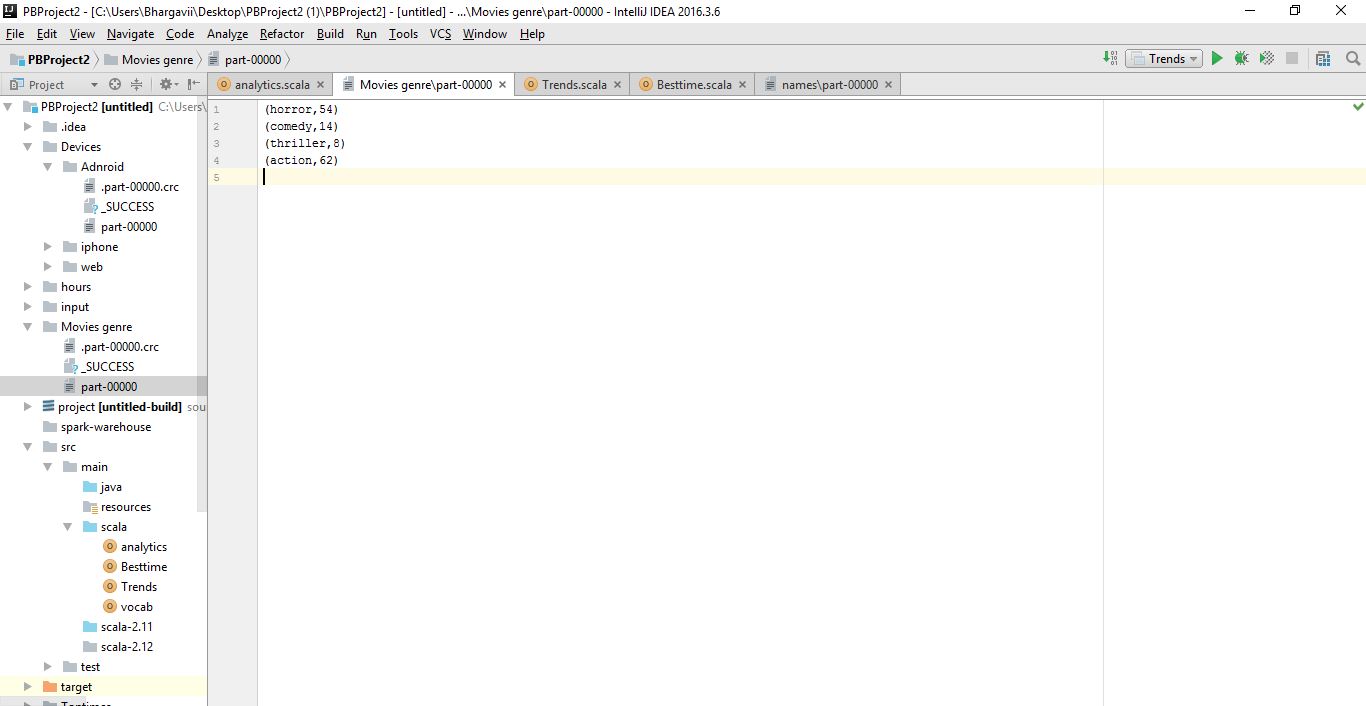
//rdd1.collect().foreach(println)

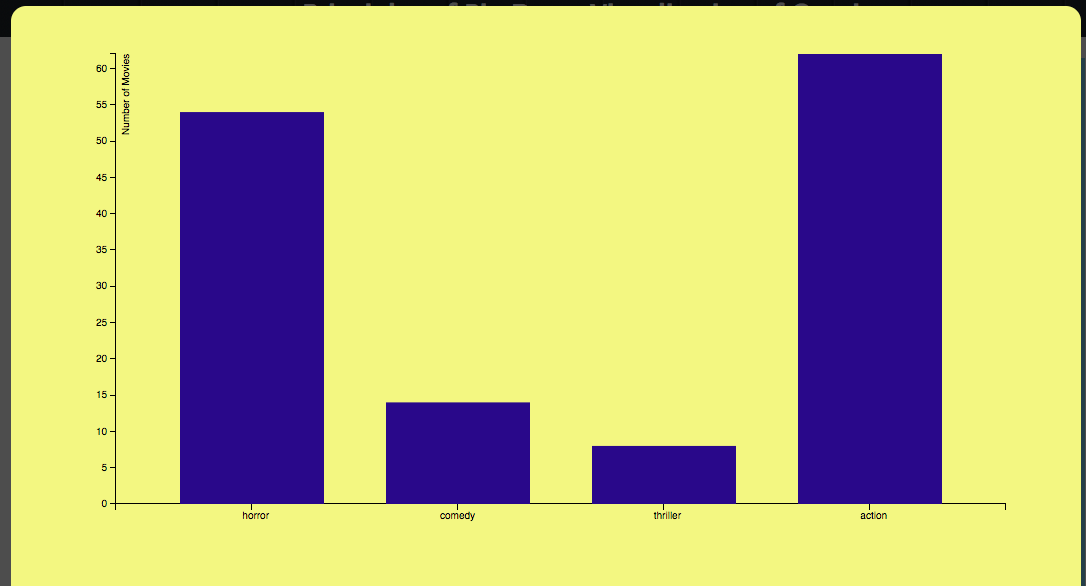
rdd1.coalesce(1).saveAsTextFile("Movies genre")

**Description:** This query finds the total number of tweets tweeted on different movie genres

namely Thriller, Horror, Action and Comedy from the collected “movies”

data set.



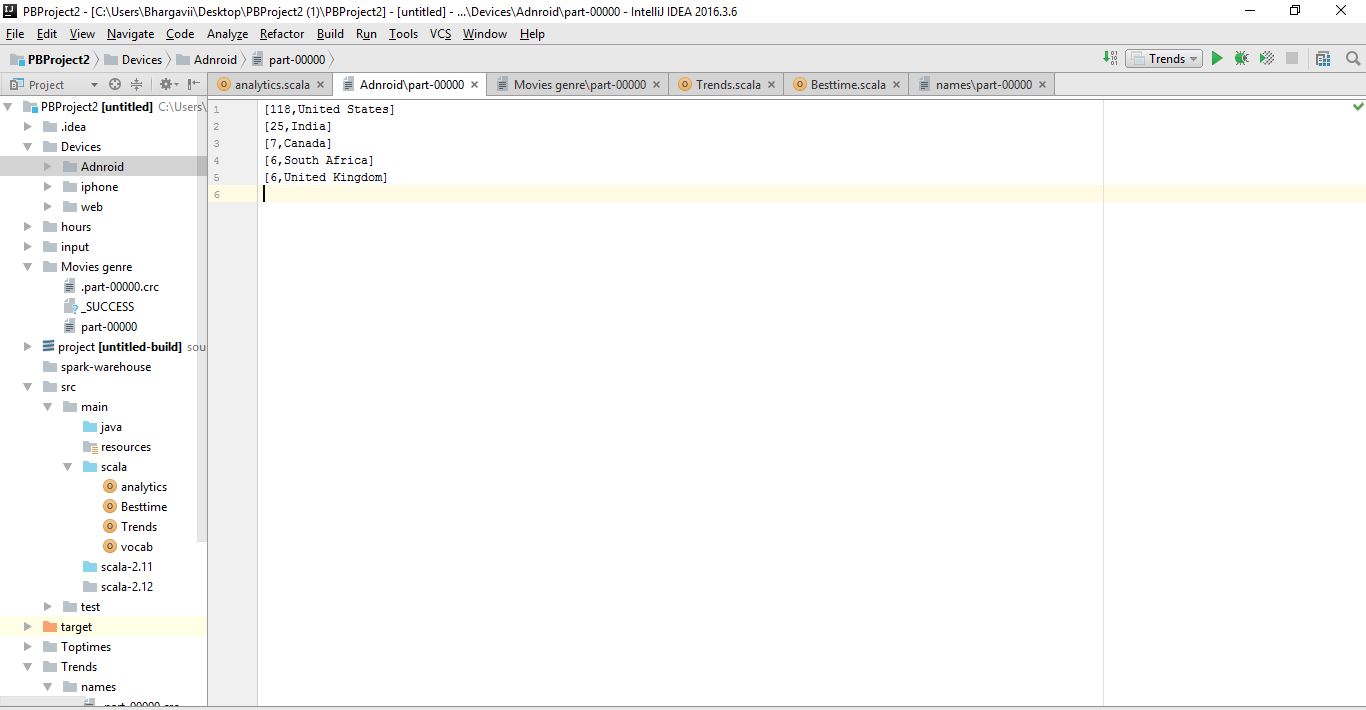


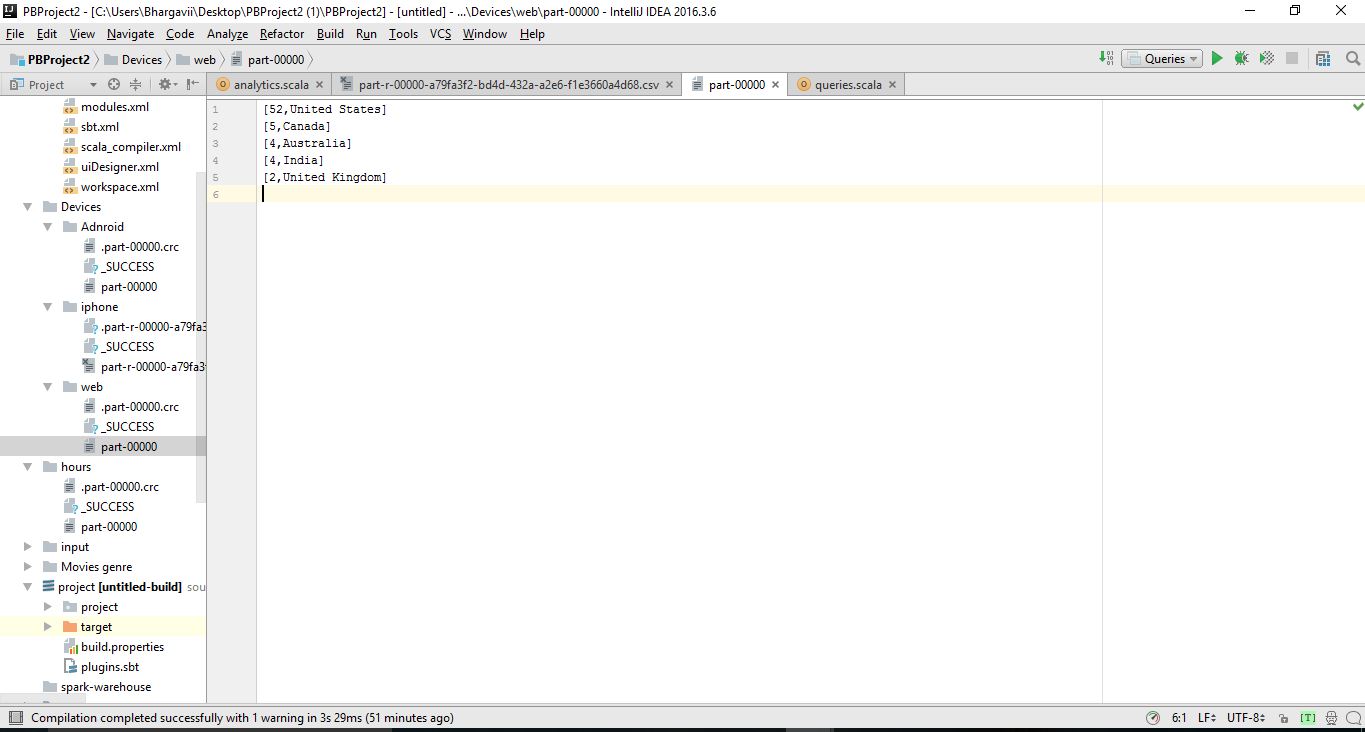
**Query 4:**

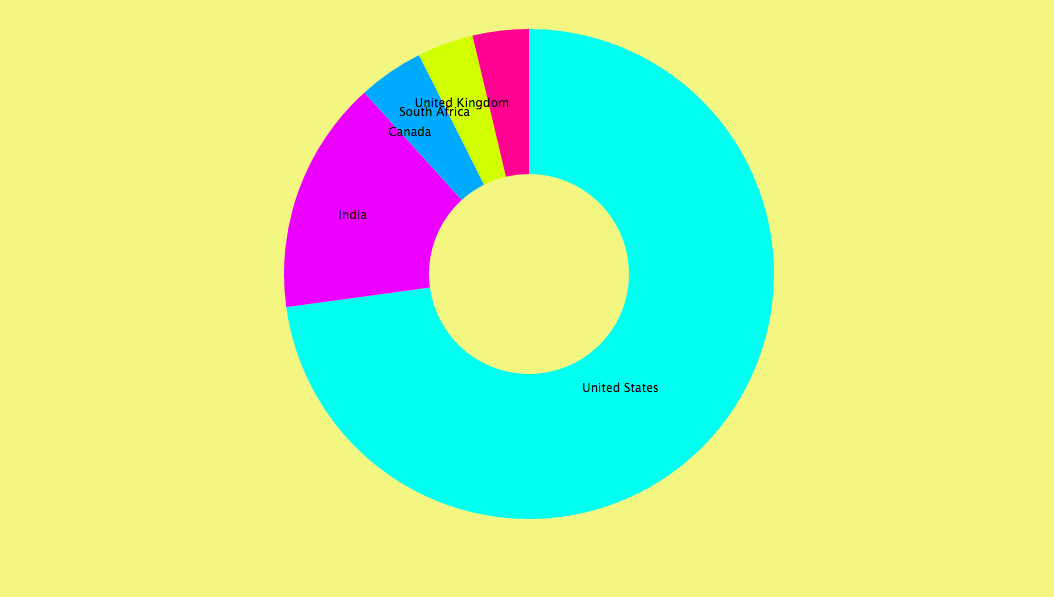
Select count(place.country),place.country from dftab" + " where source like '%Web Client%' group by place.country,source order by count(place.country) DESC

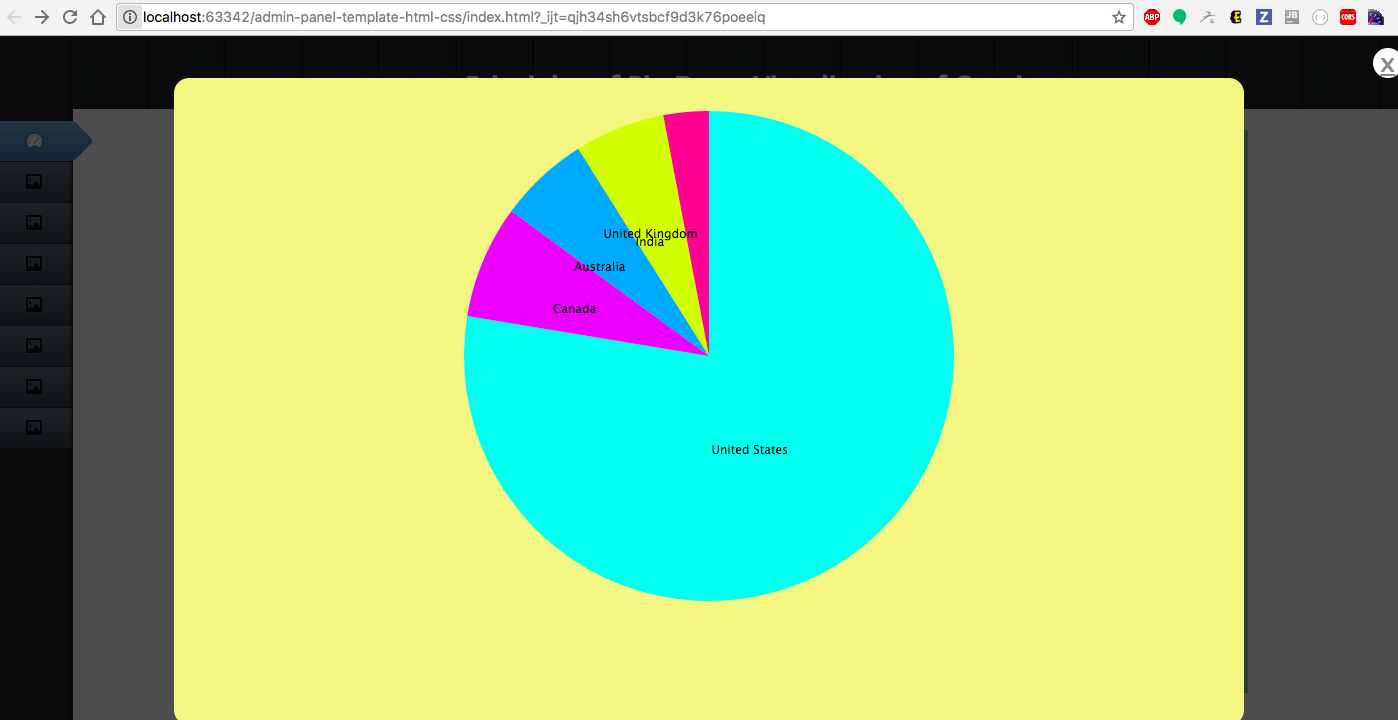
Select count(place.country),place.country from dftab " + "where source like '%Android%' and place.country !='null' group by place.country,source order by count(place.country) DESC

**Description:** This query returns number of the top five countries of users who tweeted on movies using Android and Web.









**Query 5 on Trends.txt file:**

val trends=hashtags.select(explode($"details.trends").as("trend"))

trends.show()

val names=trends.select(explode($"trend.name").as("names"))

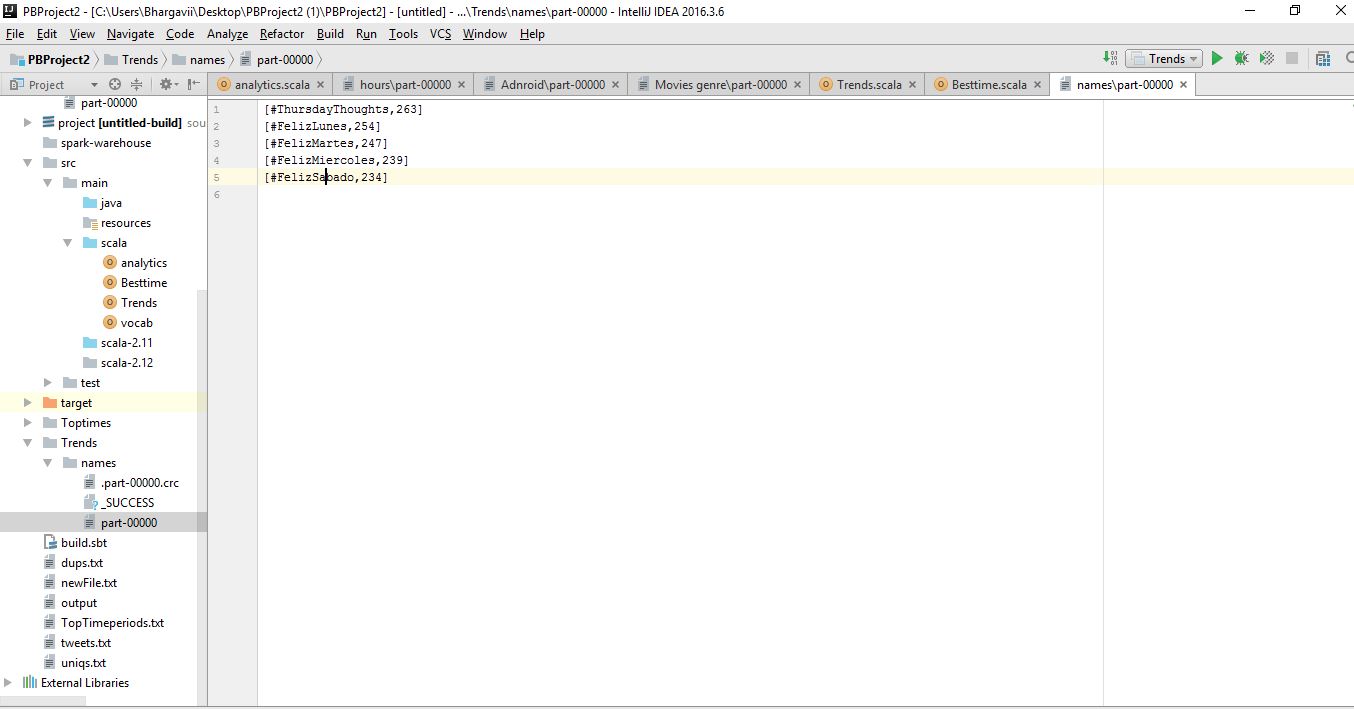
names.show()

//names.groupBy($"names").count().orderBy($"count".desc).limit(5).show()

var l = names.groupBy($"names").count().orderBy($"count".desc).limit(5)

l.rdd.coalesce(1).saveAsTextFile("Trends/names")

**Description:** This query returns the names of top five users in the trends.txt file.





**Query 6:**

Select substring(created\_at, 12,2), count(\*)" +" as count from dftab where created\_at is not null " + "group by substring(created\_at, 12,2) ORDER BY count DESC ")

**Description:** This query returns the count of number of tweets tweeted per hour.

