CENG790 Big Data Analytics
Assignment #1

Emre Doğan

Part 1.

1. By using the following code snippet, I selected fields containing the identifier, GPS coordinates, and type of license of each picture.

```
originalFlickrMeta.createOrReplaceTempView("myDF")
val desiredDF = spark.sql("SELECT photo_id, longitude, latitude, license FROM myDF")
desiredDF.show()
```

By using .show() command, top 20 samples are shown below.

```
+----------+---------+----------+--------------------+
|  photo_id|longitude|  latitude|             license|
+----------+---------+----------+--------------------+
|5610122230|     -1.0|      -1.0|Attribution-NonCo...|
|5223747995|     -1.0|      -1.0|Attribution-NonCo...|
|5592678175|     -1.0|      -1.0|Attribution-NonCo...|
|8807058226|     -1.0|      -1.0|Attribution-NonCo...|
|2860980452| -0.02592| -0.036392|Attribution-NonCo...|
|6928499157|     -1.0|      -1.0|Attribution-NonCo...|
|5674323056|     -1.0|      -1.0|Attribution-NonCo...|
|5734258350|     -1.0|      -1.0|Attribution-NonCo...|
|6198509952|     -1.0|      -1.0|Attribution-NoDer...|
|7050886391|     -1.0|      -1.0|Attribution-Share...|
|2445790010| -0.83496| -54.99022|Attribution-NonCo...|
|4913556997|-0.005252|-81.434204|Attribution-NonCo...|
|1345730799|-0.002489|  -82.9847| Attribution License|
|1345733105|-0.040082| -82.98526| Attribution License|
|5052929796|-0.088212|   0.00773|Attribution-NonCo...|
|3116901547|  -1.2E-5|    3.0E-6| Attribution License|
|3117729084|  -1.2E-5|    3.0E-6| Attribution License|
|3737526549|-0.300909|  0.094894|Attribution-Share...|
|3117764790|  -1.2E-5|    3.0E-6| Attribution License|
|3117768410|  -1.2E-5|    3.0E-6| Attribution License|
+----------+---------+----------+--------------------+
only showing top 20 rows
```

2. I used the following code snippet to create a data frame consisting of interesting pictures.

```
// q2.
desiredDF.createOrReplaceTempView("desiredDF")
val desiredDF2 = spark.sql("SELECT * FROM desiredDF WHERE license IS NOT NULL AND latitude != -1 AND longitude != -1")
desiredDF2.show()
//desiredDF2.select("license").show()
```

20 samples of filtered data can be seen below.

```
+----------+---------+----------+--------------------+
|  photo_id|longitude|  latitude|             license|
+----------+---------+----------+--------------------+
|2860980452| -0.02592| -0.036392|Attribution-NonCo...|
|2445790010| -0.83496| -54.99022|Attribution-NonCo...|
|4913556997|-0.005252|-81.434204|Attribution-NonCo...|
|1345730799|-0.002489|  -82.9847| Attribution License|
|1345733105|-0.040082| -82.98526| Attribution License|
|5052929796|-0.088212|   0.00773|Attribution-NonCo...|
|3116901547|  -1.2E-5|    3.0E-6| Attribution License|
|3117729084|  -1.2E-5|    3.0E-6| Attribution License|
|3737526549|-0.300909|  0.094894|Attribution-Share...|
|3117764790|  -1.2E-5|    3.0E-6| Attribution License|
|3117768410|  -1.2E-5|    3.0E-6| Attribution License|
|4262750956|-0.050373|  0.342372|Attribution-Share...|
|3397220196|-0.001373|   8.58E-4|Attribution-NonCo...|
|3117773794|  -1.2E-5|    3.0E-6| Attribution License|
|3117761408|  -1.2E-5|    3.0E-6| Attribution License|
|4591167499|-0.851526| 10.785503|Attribution-NonCo...|
|4591166029|-0.851526| 10.785503|Attribution-NonCo...|
|3765897146|-0.627593| 10.797897|Attribution-NonCo...|
|3755727437| -0.52597| 10.991749|Attribution-NonCo...|
|8491558947|-0.972974| 10.822321|Attribution-NonCo...|
+----------+---------+----------+--------------------+
only showing top 20 rows
```

3. The execution plan is given below,

```
== Physical Plan ==
*Project [photo_id#0L, longitude#10, latitude#11, license#15]
+- *Filter ((((isnotnull(longitude#10) && isnotnull(latitude#11)) && isnotnull(license#15)) && NOT (latitude#11 = -1.0)) && NOT (longitude#10 = -1.0))
   +- *FileScan csv [photo_id#0L,longitude#10,latitude#11,license#15] Batched: false, Format: CSV, Location: InMemoryFileIndex[file:/Users/emre/workspace/ceng790.hw1/flickrSample.txt], PartitionFilters: [], PushedFilters: [IsNotNull(longitude), IsNotNull(latitude), IsNotNull(license), Not(EqualTo(latitude,-1.0)), Not(..., ReadSchema: struct<photo_id:bigint,longitude:float,latitude:float,license:string
```

4. Data of pictures are displayed by .show() command,

```
+----------+---------+----------+--------------------+
|  photo_id|longitude|  latitude|             license|
+----------+---------+----------+--------------------+
|2860980452| -0.02592| -0.036392|Attribution-NonCo...|
|2445790010| -0.83496| -54.99022|Attribution-NonCo...|
|4913556997|-0.005252|-81.434204|Attribution-NonCo...|
|1345730799|-0.002489|  -82.9847| Attribution License|
|1345733105|-0.040082| -82.98526| Attribution License|
|5052929796|-0.088212|   0.00773|Attribution-NonCo...|
|3116901547|  -1.2E-5|    3.0E-6| Attribution License|
|3117729084|  -1.2E-5|    3.0E-6| Attribution License|
|3737526549|-0.300909|  0.094894|Attribution-Share...|
|3117764790|  -1.2E-5|    3.0E-6| Attribution License|
|3117768410|  -1.2E-5|    3.0E-6| Attribution License|
|4262750956|-0.050373|  0.342372|Attribution-Share...|
|3397220196|-0.001373|   8.58E-4|Attribution-NonCo...|
|3117773794|  -1.2E-5|    3.0E-6| Attribution License|
|3117761408|  -1.2E-5|    3.0E-6| Attribution License|
|4591167499|-0.851526| 10.785503|Attribution-NonCo...|
|4591166029|-0.851526| 10.785503|Attribution-NonCo...|
|3765897146|-0.627593| 10.797897|Attribution-NonCo...|
|3755727437| -0.52597| 10.991749|Attribution-NonCo...|
|8491558947|-0.972974| 10.822321|Attribution-NonCo...|
+----------+---------+----------+--------------------+
only showing top 20 rows
```

5. I used the following code snippet to apply JOIN operation to picture data table and license property table:

```
// q5.

// reading the license property file.
val licenceProperties = spark.sqlContext.read
  .format("csv")
  .option("delimiter", "\t")
  .option("header", "true")
  .load("flickrLicense.txt")

licenceProperties.createOrReplaceTempView("licenceProperties")
desiredDF2.createOrReplaceTempView("desiredDF2")
// creating tables so that we can join them.
val left = spark.sql("SELECT * FROM desiredDF2")
val right = spark.sql("SELECT * FROM licenceProperties")
// JOIN operation of left and right tables.
val desiredDF3 = left.join(right, "license")
desiredDF3.show()

// alternative JOIN
//val desiredDF3 = spark.sql("SELECT * FROM desiredDF2 LEFT OUTER JOIN licenceProperties ON Name=licence")

desiredDF3.createOrReplaceTempView("desiredDF3")
// selecting interesting and NonDerivative Licensed images.
val desiredDF4 = spark.sql("SELECT * FROM desiredDF3 WHERE NonDerivative=1 ")
desiredDF4.show()
desiredDF4.explain()
```

And I achieved resulting data in the following shape:

```
+--------------------+----------+---------+----------+-----------+-------------+-------------+---------+----------------------+---------------+
|             license|  photo_id|longitude|  latitude|Attribution|Noncommercial|NonDerivative|ShareAlike|PublicDomainDedication|PublicDomainWork|
+--------------------+----------+---------+----------+-----------+-------------+-------------+---------+----------------------+---------------+
|Attribution-NonCo...|2860980452| -0.02592| -0.036392|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...|4913556997|-0.005252|-81.434204|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...|8491558947|-0.972974| 10.822321|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...|3725078884|-0.906372| 11.497519|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...|3724276245|-0.906372| 11.497519|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...|3725109474|-0.906372| 11.497519|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...|3725065346|-0.906372| 11.497519|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...|3725062966|-0.906372| 11.497519|          1|            1|            1|        0|                     0|              0|
|Attribution-NoDer...|4105402596|-0.527343| 12.287764|          1|            0|            1|        0|                     0|              0|
|Attribution-NonCo...|5512012382|-0.109863| 13.781568|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...| 104791081|-0.274744| 13.578083|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...|5511312835|-0.109863| 13.781568|          1|            1|            1|        0|                     0|              0|
|Attribution-NonCo...| 104778685|-0.274744| 13.578083|          1|            1|            1|        0|                     0|              0|
|Attribution-NoDer...|6442481127|-0.067377| 16.330847|          1|            0|            1|        0|                     0|              0|
|Attribution-NonCo...| 259199471|-0.791015| 16.997038|          1|            1|            1|        0|                     0|              0|
|Attribution-NoDer...|6442477951|-0.050554| 16.270626|          1|            0|            1|        0|                     0|              0|
+--------------------+----------+---------+----------+-----------+-------------+-------------+---------+----------------------+---------------+
```

Notice that NonDerivative attribute is 1 for all image samples.

And the new execution plan is given below:

```
== Physical Plan ==
*Project [license#15, photo_id#0L, longitude#10, latitude#11, Attribution#151, Noncommercial#152,
NonDerivative#153, ShareAlike#154, PublicDomainDedication#155, PublicDomainWork#156]
+- *BroadcastHashJoin [license#15], [license#150], Inner, BuildRight
   :- *Project [photo_id#0L, longitude#10, latitude#11, license#15]
   :  +- *Filter ((((isnotnull(longitude#10) && isnotnull(latitude#11)) && isnotnull(license#15)) && NOT
(latitude#11 = -1.0)) && NOT (longitude#10 = -1.0))
   :     +- *FileScan csv [photo_id#0L,longitude#10,latitude#11,license#15] Batched: false, Format: CSV,
Location: InMemoryFileIndex[file:/Users/emre/workspace/ceng790.hw1/flickrSample.txt], PartitionFilters: [],
PushedFilters: [IsNotNull(longitude), IsNotNull(latitude), IsNotNull(license), Not(EqualTo(latitude,-1.0)),
Not(..., ReadSchema: struct<photo_id:bigint,longitude:float,latitude:float,license:string>
   +- BroadcastExchange HashedRelationBroadcastMode(List(input[0, string, true]))
      +- *Project [license#150, Attribution#151, Noncommercial#152, NonDerivative#153, ShareAlike#154,
PublicDomainDedication#155, PublicDomainWork#156]
         +- *Filter ((isnotnull(NonDerivative#153) && (cast(NonDerivative#153 as int) = 1)) &&
isnotnull(license#150))
            +- *FileScan csv
[license#150,Attribution#151,Noncommercial#152,NonDerivative#153,ShareAlike#154,PublicDomainDedication#155,
PublicDomainWork#156] Batched: false, Format: CSV, Location:
InMemoryFileIndex[file:/Users/emre/workspace/ceng790.hw1/flickrLicense.txt], PartitionFilters: [],
PushedFilters: [IsNotNull(NonDerivative), IsNotNull(license)], ReadSchema:
struct<license:string,Attribution:string,Noncommercial:string,NonDerivative:string,ShareAlike:str...
```

It is noticeable that the new execution plan is more complicated than the first version. And it is seen that SQL SELECT statements are converted to Filter() operation.

6. New physical plan is given below. By applying cache() command, I cached two tables to be joined 'left' and 'right' tables. Yellow highlighted lines show that instead of file scanning or reading, data is directly taken from cache which decreases the execution time.

```
// q6.

left.cache()
right.cache()
licenceProperties.createOrReplaceTempView("licenceProperties")
val desiredDF5 = left.join(right, "license")
//val desiredDF3 = left.join(right, left.col("license") === right.col("Name"))
//val desiredDF3 = spark.sql("SELECT * FROM desiredDF2 LEFT OUTER JOIN licenceProperties ON Name=licence")
desiredDF5.explain()
```

```
== Physical Plan ==
*Project [license#15, photo_id#0L, longitude#10, latitude#11, Attribution#151, Noncommercial#152,
NonDerivative#153, ShareAlike#154, PublicDomainDedication#155, PublicDomainWork#156]
+- *SortMergeJoin [license#15], [license#150], Inner
   :- *Sort [license#15 ASC NULLS FIRST], false, 0
   :  +- Exchange hashpartitioning(license#15, 200)
   :     +- *Filter isnotnull(license#15)
   :        +- InMemoryTableScan [photo_id#0L, longitude#10, latitude#11, license#15], [isnotnull(license#15)]
   :              +- InMemoryRelation [photo_id#0L, longitude#10, latitude#11, license#15], true, 10000,
StorageLevel(disk, memory, deserialized, 1 replicas)
   :                    +- *Project [photo_id#0L, longitude#10, latitude#11, license#15]
   :                       +- *Filter ((((isnotnull(longitude#10) && isnotnull(latitude#11)) &&
isnotnull(license#15)) && NOT (latitude#11 = -1.0)) && NOT (longitude#10 = -1.0))
   :                          +- *FileScan csv [photo_id#0L,longitude#10,latitude#11,license#15] Batched:
false, Format: CSV, Location: InMemoryFileIndex[file:/Users/emre/workspace/ceng790.hw1/flickrSample.txt],
PartitionFilters: [], PushedFilters: [IsNotNull(longitude), IsNotNull(latitude), IsNotNull(license),
Not(EqualTo(latitude,-1.0)), Not(..., ReadSchema:
struct<photo_id:bigint,longitude:float,latitude:float,license:string>
   +- *Sort [license#150 ASC NULLS FIRST], false, 0
      +- Exchange hashpartitioning(license#150, 200)
         +- *Filter isnotnull(license#150)
            +- InMemoryTableScan [license#150, Attribution#151, Noncommercial#152, NonDerivative#153,
ShareAlike#154, PublicDomainDedication#155, PublicDomainWork#156], [isnotnull(license#150)]
                  +- InMemoryRelation [license#150, Attribution#151, Noncommercial#152, NonDerivative#153,
ShareAlike#154, PublicDomainDedication#155, PublicDomainWork#156], true, 10000, StorageLevel(disk, memory,
deserialized, 1 replicas)
                        +- *FileScan csv
[license#150,Attribution#151,Noncommercial#152,NonDerivative#153,ShareAlike#154,PublicDomainDedication#155,
PublicDomainWork#156] Batched: false, Format: CSV, Location:
InMemoryFileIndex[file:/Users/emre/workspace/ceng790.hw1/flickrLicense.txt], PartitionFilters: [],
PushedFilters: [], ReadSchema:
struct<license:string,Attribution:string,Noncommercial:string,NonDerivative:string,ShareAlike:str...
```

7. The .csv file is available attached in the project in the `/project_directory/ceng790.hw1/part1Result.csv/` directory.

Part 2.

1. The number of elements and five sample data is given below.

```
count is 100


5610122230      54345792@N00    higginskurt     2011-04-11 10:20:13.0   1302531613              SROTAROX%3Ca+href%3D%22http%3A%2F%2Fwww.serotta.com%2F%22+rel%3D%22nofollow%22%3ESerrot
5223747995      54345792@N00    higginskurt     2010-12-01 12:58:33.0   1291226313              Titan   I+won+this+one...+albeit+using+abbreviated-limit+of+one+hour-Risk-type-battle+
5592678175      54345792@N00    higginskurt     2011-04-05 15:51:03.0   1302033063              Jenny+snoozing+on+the+ride                                      -1.0    -1.0    16      http:/
8807058226      34427466499@N01 Padre+Denny     2013-05-23 16:32:53.0   1369344773                                              -1.0    -1.0    16      http://www.flickr.com/photos/3
2860980452      23516515@N07    50mm-traveller  2008-06-19 12:25:30.0   1221513976      FUJIFILM+FinePix+F30    leaves+in+motionleaves+in+motiongallery2flickr          -0.02592-0.036
```

As there are many attributes of data, I could not make all data fit into the screenshot.

2. Before starting, I want to mention a problem I faced. Attributes of data are tab separated in flickrSample.txt. And, attributes without a value are stated just as an empty string(""). The problem of this notation is that while creating the picture object, empty strings cannot be pass as a parameter to Picture object. To solve this problem, I used a very simple RegEx trick such that whenever two consecutive tabs are seen in data ("\t\t"), it is replaced with a ("\t"+"null"+"\t") character set. I saved the new txt file to be used in this part as "flickrSampleManipulated.txt" to my project directory so that you can check it.

   The code snippet below illustrates the creation of RDD of Picture objects.

```
// question 2:
// created an array so that I can directly feed data to the Picture objects.
val rddAsArray = originalFlickrMeta.map(x => x.split("\t"))

// creation of RDD of Picture objects from rddAsArray.
val pictureRDD: RDD[Picture] = rddAsArray.map(x => new Picture(x))

// filtering data to achieve interesting pictures with non-null usertags and country attributes.
val interestingPictureRDD: RDD[Picture] = pictureRDD.filter(f => f.c != null && f.userTags != "null" && f.userTags.size > 0  && f.lat != -1 && f.lon != -1 )
interestingPictureRDD.take(5).foreach(f => println(f))
```

   The resulting 5 samples are given below,

```
(UV, aids, art education, ghana, hiv, hiv/aids, hiv prevention, lotos collective, malina de carlo, roberto sanchez-camus, youth visions)
(UV, aids, art education, ghana, hiv, hiv/aids, hiv prevention, lotos collective, malina de carlo, roberto sanchez-camus, youth visions)
(BN, africa, ghana, idds, navrongo)
(UV, africa, ghana, idds, night)
(UV, dhf, ghana, gspd)
```

3. Code snippet to group images by country.

```
// question 3:

// grouping interestingPictureRDD by their country attribute.
val groupedByCountryRDD = interestingPictureRDD.groupBy(_.c)
groupedByCountryRDD.foreach(f => println(f._2))

println("\ntype is equal to : val groupedByCountryRDD: RDD[(Country, Iterable[Picture])]")
```

Notice that type of RDD is equal to RDD[(Country, Iterable[Picture].

The first country is ‘AG’ in the list. List of images is given below.

CompactBuffer(
(AG, ثقافة أمازيغية, تمنراست, الهقار, الطوارق, الجزائر, alger, algeria, amazigh culture, hoggar, la culture amazighe, tamanrasset, touareg),
(AG, ثقافة أمازيغية, تمنراست, الهقار, الطوارق, الجزائر, alger, algeria, amazigh culture, hoggar, la culture amazighe, tamanrasset),
(AG, ثقافة أمازيغية, تمنراست, الهقار, الطوارق, الجزائر, alger, algeria, amazigh culture, hoggar))

4.

From the grouped RDD from Question 3, I achieved the concatenated version of RDD (type of RDD[(Country, List[String])]) by using the following code snippet.

```
// question 4:

 // transforming RDD[(Country, Iterable[Picture])] to RDD[(Country, List[List[String]])]
 val com_ = groupedByCountryRDD.map(x => (x._1, x._2.toList.map(f => f.userTags.toList)))

 // transforming RDD[(Country, List[List[String]])] to RDD[(Country, List[String])].
 val com_2 = com_.map(f =>(f._1, f._2.flatMap(f => f)))
 // this is the concatenated version.
 com_2.foreach(println)
```

And the concatenated version of lists are given below.

(ML,List(yosemite, yosemite, yosemite, yosemite, yosemite, yosemite, yosemite, yosemite, canada square park, canary wharf, jiving lindy hoppers, pasadena roof orchestra, twilight delights, boat, dune, gao, mali, niger, river, sahara, sand, mali, yosemite, canada square park, canary wharf, jiving lindy hoppers, pasadena roof orchestra, twilight delights, canada square park, canary wharf, jiving lindy hoppers, pasadena roof orchestra, twilight delights, mali, canada square park, canary wharf, jiving lindy hoppers, pasadena roof orchestra, twilight delights, mali, gao, mali, man, nomad, sahara, tuareg, africa, desierto, islam, mali, mezquitas, niger, rio niger, tombuctú, viajes, africa, desierto, islam, mali, mezquitas, niger, rio niger, tombuctú, viajes, africa, desierto, islam, mali, mercado tombuctú, mezquitas, niger, rio niger, viajes, africa, desierto, islam, mali, mezquitas, niger, rio niger, tombuctú, viajes, africa, desierto, islam, mali, mezquitas, niger, rio niger, tombuctú, viajes, africa, desierto, islam, mali, mezquitas, niger, rio niger, tuaregs tombuctú, viajes, africa, desierto, islam, mali, mezquitas, niger, rio niger, tuaregs tombuctú, viajes, 4x4, africa, desierto, islam, mali, mezquitas, niger, pescados, rio niger, transbordador tombuctú, viajes, África, desierto, islam, mali, mezquita, niger, rio niger, tombuctú, tuaregs, viajes, africa, desierto, islam, mali, mezquitas, niger, rio niger, tombuctú, viajes))

(BN,List(africa, ghana, idds, navrongo, africa, ghana, idds, rice, single mothers, ghana, lab, ghana, lab, ghana, lab, ghana, lab, ghana, lab))

(AG,List(ثقافة أمازيغية, تمنراست, الهقار, الطوارق, الجزائر, alger, algeria, amazigh culture, hoggar, la culture amazighe, tamanrasset, touareg, ثقافة أمازيغية, تمنراست, الهقار, الطوارق, الجزائر, alger, algeria, amazigh culture, hoggar, la culture amazighe, tamanrasset, ثقافة أمازيغية, تمنراست, الهقار, الطوارق, الجزائر, alger, algeria, amazigh culture, hoggar))

(UV,List(aids, art education, ghana, hiv, hiv/aids, hiv prevention, lotos collective, malina de carlo, roberto sanchez-camus, youth visions, aids, art education, ghana, hiv, hiv/aids, hiv prevention, lotos collective, malina de carlo, roberto sanchez-camus, youth visions, africa, ghana, idds, night, dhf, ghana, gspd, aids, art education, ghana, hiv, hiv/aids, hiv prevention, lotos collective, malina de carlo, roberto sanchez-camus, youth visions, africa, bedroom, bolga, ghana, hazwan, africa, ghana, aids, art education, ghana, hiv, hiv/aids, hiv prevention, lotos collective, malina de carlo, roberto sanchez-camus, youth visions, burkinafaso, electricity, informal, mfp, moulin, burkinafaso, electricity, informal, mfp, moulin, burkinafaso, electricity, informal, mfp, moulin, burkinafaso, electricity, informal, mfp, moulin, burkinafaso, electricity, informal, mfp, moulin, afrika, burkina faso, entwicklungshilfe, patenschaft, afrika, burkina faso, entwicklungshilfe, patenschaft, adaptation, agriculture, burkina faso, ccafs, cgiar, cgiarclimate, community, demi-lune, desert, drought, drylands, education, farmer, farmers organisations, gourcy, innovation, participation, sahel, west africa, zai, zodoma, null, 12scatti, burkina faso, 2007, africa, afrika, afrique, afrique de l'ouest, burkina, burkina faso, dori, faso, img_8641.jpg, travel, west africa, westafrika, adobe, africa, afrique, banco, bani, burkina, burkina_faso, faso, fulani,

```
guillaume_colin, mudbrick, peul, sahel, afrique, blanc, burkinafaso, bw, et, noir, ouagadougou, 2007,
africa, afrika, afrique, afrique de l'ouest, burkina, burkina faso, dori, faso, img_8643.jpg, travel, west
africa, westafrika, yosemite, adobe, africa, afrique, banco, bani, burkina, burkina_faso, faso,
guillaume_colin, mosquée, yosemite, afrique, blanc, burkinafaso, bw, et, noir, ouagadougou, 2007, africa,
afrika, afrique, afrique de l'ouest, beggar, burkina, burkina faso, dori, faso, fulani, travel, west
africa, westafrika, 2007, africa, afrika, afrique, afrique de l'ouest, burkina, burkina faso, dori, faso,
img_8649.jpg, travel, west africa, westafrika, 2007, africa, afrika, afrique, afrique de l'ouest, burkina,
burkina faso, dori, faso, img_8602.jpg, travel, west africa, westafrika, burkina-faso, gorom-gorom,
burkina-faso, gorom-gorom, burkina-faso, oursi, burkina-faso, oursi, burkina-faso, oursi, burkina-faso,
oursi, burkina-faso, gorom-gorom, burkina-faso, gorom-gorom, burkina-faso, oursi))
```

5. Map function maps each RDD[(Country, List[String])] structure to RDD[(Country, Map[String, Int])] structure by using groupBy(identity) and mapValues() functions.

   The code snippet for this part is given below.

```
// question 5:

// to have each tag with its frequency in the form of RDD[(Country, Map[String, Int])].

val freqAddedVersion = com_2.map(f =>(f._1, (f._2.groupBy(identity).mapValues(_.size))))
freqAddedVersion.foreach(println)
```

Results in the form of RDD[(Country, Map[String, Int])] is given below for each country.

```
(ML,Map(sand -> 1, canary wharf -> 4, dune -> 1, mezquitas -> 9, tuaregs -> 1, gao -> 2, nomad -> 1,
transbordador tombuctú -> 1, river -> 1, yosemite -> 9, rio niger -> 10, man -> 1, boat -> 1, mali -> 15,
pasadena roof orchestra -> 4, mezquita -> 1, africa -> 9, twilight delights -> 4, viajes -> 10, jiving
lindy hoppers -> 4, 4x4 -> 1, tombuctú -> 6, África -> 1, desierto -> 10, mercado tombuctú -> 1, niger ->
11, tuaregs tombuctú -> 2, islam -> 10, canada square park -> 4, pescados -> 1, tuareg -> 1, sahara -> 2))

(BN,Map(lab -> 5, ghana -> 7, rice -> 1, single mothers -> 1, africa -> 2, idds -> 2, navrongo -> 1))

(AG,Map(3> - ثقافة أمازيغية, 3> - الطوارق, tamanrasset -> 2, 3> - الهقار, 3> - تمنراست, algeria -> 3, touareg -> 1, alger
-> 3, hoggar -> 3, la culture amazighe -> 2, 3> - الجزائر, amazigh culture -> 3))

(UV,Map(burkina_faso -> 2, patenschaft -> 2, img_8602.jpg -> 1, community -> 1, zai -> 1, drylands -> 1,
westafrika -> 5, burkina-faso -> 9, aids -> 4, bani -> 2, img_8643.jpg -> 1, hiv prevention -> 4, moulin ->
5, dori -> 5, mfp -> 5, desert -> 1, entwicklungshilfe -> 2, noir -> 2, 2007 -> 5, oursi -> 5, 12scatti ->
1, bw -> 2, gspd -> 1, beggar -> 1, farmers organisations -> 1, peul -> 1, bedroom -> 1, mosquée -> 1,
burkina faso -> 9, hazwan -> 1, adobe -> 2, night -> 1, malina de carlo -> 4, zodoma -> 1, yosemite -> 2,
burkinafaso -> 7, afrique -> 9, ghana -> 8, null -> 1, burkina -> 7, et -> 2, west africa -> 6, mudbrick ->
1, demi-lune -> 1, img_8649.jpg -> 1, participation -> 1, agriculture -> 1, travel -> 5, banco -> 2, africa
-> 10, ouagadougou -> 2, blanc -> 2, education -> 1, lotos collective -> 4, cgiarclimate -> 1, faso -> 7,
roberto sanchez-camus -> 4, idds -> 1, innovation -> 1, adaptation -> 1, drought -> 1, sahel -> 2, ccafs ->
1, bolga -> 1, cgiar -> 1, informal -> 5, farmer -> 1, gourcy -> 1, hiv/aids -> 4, dhf -> 1, youth visions
-> 4, art education -> 4, img_8641.jpg -> 1, gorom-gorom -> 4, afrika -> 7, electricity -> 5,
guillaume_colin -> 2, afrique de l'ouest -> 5, hiv -> 4, fulani -> 2))
```

6. If Map-Reduce Algorithm with a Combiner is applied for this problem, we can reach the same result without decreasing the size until the end of process. A Combiner[1],also known as mini-reducer, is defined as an optional class that summarizes the Mapper output record with the same Key before passing to the Reducer. This approach allows us to start counting process before the Reduce operation. So, the same operation is executed with a higher level of parallelism.

[1] https://data-flair.training/blogs/hadoop-combiner-tutorial/