Assignment 18.1

Below is the dataset which we will be using for this Assignment in all problems. It has been kept in local file system:-

```
[acadgild@localhost Assignment-18]$ ls -lrt
total 12
-rw-rw-r--. 1 acadgild acadgild 929 Jan  3 19:49 S18_Dataset_Holidays.txt
-rw-rw-r--. 1 acadgild acadgild  42 Jan  3 19:49 S18_Dataset_Transport.txt
-rw-rw-r--. 1 acadgild acadgild 116 Jan  3 19:49 S18_Dataset_User_details.txt
[acadgild@localhost Assignment-18]$ cat S18_Dataset_Transport.txt
airplane,170
car,140
train,120
ship,200[acadgild@localhost Assignment-18]$ cat S18_Dataset_User_details.txt
1,mark,15
2,john,16
3,luke,17
4,lisa,27
5,mark,25
6,peter,22
7,james,21
8,andrew,55
9,thomas,46
```

```
10,annie,44[acadgild@localhost Assignment-18]$ cat S18_Dataset_Holidays.txt
1,CHN,IND,airplane,200,1990
2,IND,CHN,airplane,200,1991
3,IND,CHN,airplane,200,1992
4,RUS,IND,airplane,200,1990
5,CHN,RUS,airplane,200,1992
6,AUS,PAK,airplane,200,1991
7,RUS,AUS,airplane,200,1990
8,IND,RUS,airplane,200,1991
9,CHN,RUS,airplane,200,1992
10,AUS,CHN,airplane,200,1993
1,AUS,CHN,airplane,200,1993
2,CHN,IND,airplane,200,1993
3,CHN,IND,airplane,200,1993
4,IND,AUS,airplane,200,1991
5,AUS,IND,airplane,200,1992
6,RUS,CHN,airplane,200,1993
7,CHN,RUS,airplane,200,1990
8,AUS,CHN,airplane,200,1990
9,IND,AUS,airplane,200,1991
10,RUS,CHN,airplane,200,1992
1,PAK,IND,airplane,200,1993
2,IND,RUS,airplane,200,1991
3,CHN,PAK,airplane,200,1991
4,CHN,PAK,airplane,200,1990
5,IND,PAK,airplane,200,1991
6,PAK,RUS,airplane,200,1991
7,CHN,IND,airplane,200,1990
8,RUS,IND,airplane,200,1992
9,RUS,IND,airplane,200,1992
10,CHN,AUS,airplane,200,1990
1,PAK,AUS,airplane,200,1993
5,CHN,PAK,airplane,200,1994[acadgild@localhost Assignment-18]$
```

DataSet is uploaded in baseRDD:-

- val baseRDD = sc.textFile("/home/acadgild/Assignment-18/S18_Dataset_Holidays.txt")
- import org.apache.spark.storage.StorageLevel
- baseRDD.persist(StorageLevel.MEMORY_ONLY)

```
scala> val baseRDD = sc.textFile("/home/acadgild/Assignment-18/S18_Dataset_Holidays.txt")
baseRDD: org.apache.spark.rdd.RDD[String] = /home/acadgild/Assignment-18/S18_Dataset_Holidays.txt MapPartitionsRDD[3] at textFile at <console>:25

scala> import org.apache.spark.storage.StorageLevel
import org.apache.spark.storage.StorageLevel

scala> baseRDD.persist(StorageLevel.MEMORY_ONLY)
res2: baseRDD.type = /home/acadgild/Assignment-18/S18_Dataset_Holidays.txt MapPartitionsRDD[3] at textFile at <console>:25
```

```
scala> baseRDD.foreach(println)
1,CHN,IND,airplane,200,1990
2,IND,CHN,airplane,200,1991
3,IND,CHN,airplane,200,1992
4,RUS,IND,airplane,200,1990
5,CHN,RUS,airplane,200,1992
6,AUS,PAK,airplane,200,1991
7,RUS,AUS,airplane,200,1990
8,IND,RUS,airplane,200,1991
9,CHN,RUS,airplane,200,1992
10,AUS,CHN,airplane,200,1993
1,AUS,CHN,airplane,200,1993
2,CHN,IND,airplane,200,1993
3,CHN,IND,airplane,200,1993
4,IND,AUS,airplane,200,1991
5,AUS,IND,airplane,200,1992
6,RUS,CHN,airplane,200,1993
7,CHN,RUS,airplane,200,1990
8,AUS,CHN,airplane,200,1990
9,IND,AUS,airplane,200,1991
10,RUS,CHN,airplane,200,1992
1,PAK,IND,airplane,200,1993
2,IND,RUS,airplane,200,1991
3,CHN,PAK,airplane,200,1991
4,CHN,PAK,airplane,200,1990
5,IND,PAK,airplane,200,1991
6,PAK,RUS,airplane,200,1991
7,CHN,IND,airplane,200,1990
8,RUS,IND,airplane,200,1992
9,RUS,IND,airplane,200,1992
10,CHN,AUS,airplane,200,1990
1,PAK,AUS,airplane,200,1993
5,CHN,PAK,airplane,200,1994
```

Problem Statement:-

1. What is the distribution of the total number of air-travelers per year
2. What is the total air distance covered by each user per year
3. Which user has travelled the largest distance till date
4. What is the most preferred destination for all users.

Solution:-

- **The distribution of the total number of air-travelers per year**

Below is the code used:-

- val splitRDD = baseRDD.map(x => (x.split(",")(5).toInt,1))
- val countSplit = splitRDD.reduceByKey((x,y) => (x + y))
- countSplit.foreach(println)

Output:-

```
scala> val splitRDD = baseRDD.map(x => (x.split(",")(5).toInt,1))
splitRDD: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[4] at map at <console>:28

scala> val countSplit = splitRDD.reduceByKey((x,y) => (x + y))
countSplit: org.apache.spark.rdd.RDD[(Int, Int)] = ShuffledRDD[5] at reduceByKey at <console>:30

scala> countSplit.foreach(println)
(1994,1)
(1992,7)
(1990,8)
(1991,9)
(1993,7)

scala>
```

- **The total air distance covered by each user per year**

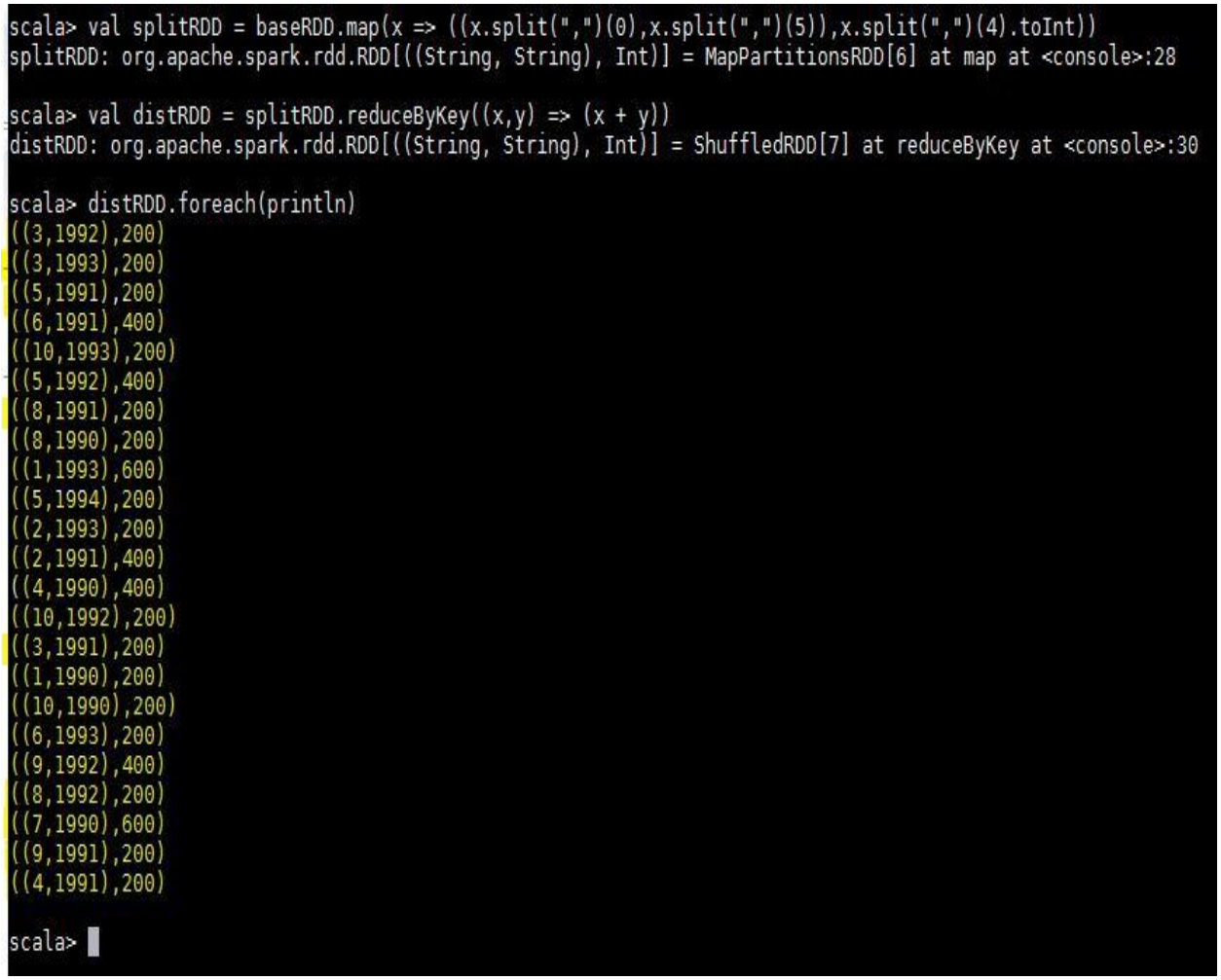
Below is the code used:-

```
val splitRDD = baseRDD.map(x =>
((x.split(",")(0),x.split(",")(5)),x.split(",")(4).toInt))
```

```
val distRDD = splitRDD.reduceByKey((x,y) => (x + y))
```

```
distRDD.foreach(println)
```

Output:-



- **User has travelled the largest distance till date**

Below is the code used:-

- val userRDD = baseRDD.map(x=> (x.split(",")(0),x.split(",")(4).toInt))
- val totaldistRDD = userRDD.reduceByKey((x,y) => (x+y))
- val maxRDD = totaldistRDD.takeOrdered(1)
- maxRDD.foreach(println)

Output:-

```
scala> val userRDD = baseRDD.map(x=> (x.split(",")(0),x.split(",")(4).toInt))
userRDD: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[11] at map at <console>:28

scala> val totaldistRDD = userRDD.reduceByKey((x,y) => (x+y))
totaldistRDD: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[12] at reduceByKey at <console>:30

scala> val maxRDD = totaldistRDD.takeOrdered(1)
maxRDD: Array[(String, Int)] = Array((1,800))

scala> maxRDD.foreach(println)
(1,800)
```

This shows that Mark has travelled the largest distance till date.

- **The most preferred destination for all users.**

Below is the code used:-

- val destRDD = baseRDD.map(x => (x.split(",")(2),1))
- val destreduceRDD = destRDD.reduceByKey((x,y) => (x + y))
- val maxRDD = destreduceRDD.takeOrdered(1)(Ordering[Int].reverse.on(_._2))
- maxRDD.foreach(println)

Output:-

```
scala> val destRDD = baseRDD.map(x => (x.split(",")(2),1))
destRDD: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[14] at map at <console>:28

scala> val destreduceRDD = destRDD.reduceByKey((x,y) => (x + y))
destreduceRDD: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[15] at reduceByKey at <console>:30

scala> val maxRDD = destreduceRDD.takeOrdered(1)(Ordering[Int].reverse.on(_._2))
maxRDD: Array[(String, Int)] = Array((IND,9))

scala> maxRDD.foreach(println)
(IND,9)

scala>
```