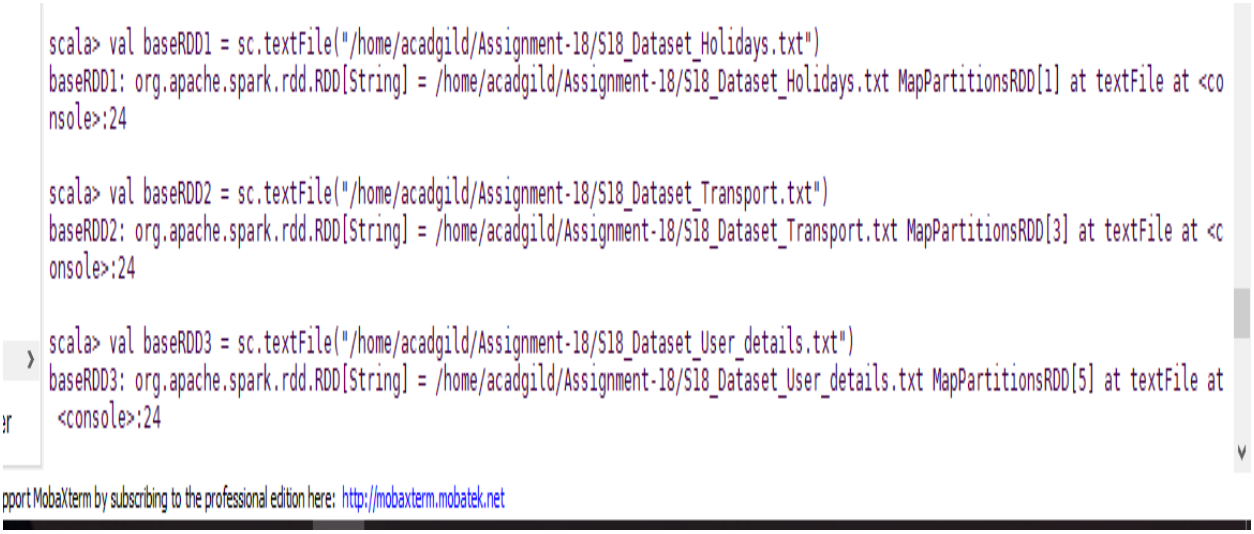
Problem Statement

1) Considering age groups of < 20 , 20-35, 35 > ,Which age group spends the most
amount of money travelling.
2) What is the amount spent by each age-group, every year in travelling?

Question 1 Solutions:



Accadgild_Serssion_18_Assignment_18.3_Solutions

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

scala> baseRDD1.persist(StorageLevel.MEMORY_ONLY)
res0: baseRDD1.type = /home/acadgild/Assignment-18/S18_Dataset_Holidays.txt MapPartitionsRDD[1] at textFile at <console>:24

scala> baseRDD2.persist(StorageLevel.MEMORY_ONLY)
res1: baseRDD2.type = /home/acadgild/Assignment-18/S18_Dataset_Transport.txt MapPartitionsRDD[3] at textFile at <console>:24

scala> baseRDD3.persist(StorageLevel.MEMORY_ONLY)
res2: baseRDD3.type = /home/acadgild/Assignment-18/S18_Dataset_User_details.txt MapPartitionsRDD[5] at textFile at <console>:24

scala>

scala> val travel = baseRDD1.map(x => (x.split(",")(0).toInt,x.split(",")(1),x.split(",")(2),x.split(",")(3),x.split(",")(4).toInt,x.s
t(",")(5).toInt))
travel: org.apache.spark.rdd.RDD[(Int, String, String, String, Int, Int)] = MapPartitionsRDD[6] at map at <console>:27

scala> val transport = baseRDD2.map(x => (x.split(",")(0),x.split(",")(1).toInt))
transport: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[7] at map at <console>:27

scala> val user = baseRDD3.map(x => (x.split(",")(0).toInt,x.split(",")(1),x.split(",")(2).toInt))
user: org.apache.spark.rdd.RDD[(Int, String, Int)] = MapPartitionsRDD[8] at map at <console>:27

scala>

scala> val AgeMap = user.map(x => x._1 ->
     |      {
     |      if(x._3<20)
     |      "20"
     |      else if(x._3>35)
     |      "35"
     |      else "20-35"
     |      })
AgeMap: org.apache.spark.rdd.RDD[(Int, String)] = MapPartitionsRDD[9] at map at <console>:29
```



```
scala> val userMap = travel.map(x => x._4 -> (x._1,x._5))
userMap: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[10] at map at <console>:29

scala> val transportmap = transport.map(x=> x._1 -> x._2)
transportmap: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[11] at map at <console>:29

scala> val joinCost = userMap.join(transportmap)
joinCost: org.apache.spark.rdd.RDD[(String, ((Int, Int), Int))] = MapPartitionsRDD[14] at join at <console>:37

scala> val calCost = joinCost.map(x => x._2._1._1 -> x._2._1._2 * x._2._2)
calCost: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[15] at map at <console>:39

scala> val groupCost = calCost.groupByKey().map(x => x._1 -> x._2.sum)
groupCost: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[17] at map at <console>:41

scala> val groupAgeCost = AgeMap.join(groupCost).map(x => x._2._1 -> x._2._2)
groupAgeCost: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[21] at map at <console>:49

scala> val GroupSpend = groupAgeCost.group
<console>:51: error: value group is not a member of org.apache.spark.rdd.RDD[(String, Int)]
       val GroupSpend = groupAgeCost.group
                                     ^

scala> val finalCost = groupAgeCost.groupByKey().map(x => x._1 -> x._2.sum)
finalCost: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[23] at map at <console>:51

scala> val maxVal = finalCost.sortBy(x => -x._2).first()
maxVal: (String, Int) = (20-35,442000)

scala>
```

Question 2 Solutions:

```
scala> val UserYearMap = travel.map(x => x._4 -> (x._1,x._5,x._6))
UserYearMap: org.apache.spark.rdd.RDD[(String, (Int, Int, Int))] = MapPartitionsRDD[27] at map at <console>:29

scala> val transportmap = transport.map(x=> x._1 -> x._2)
transportmap: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[28] at map at <console>:29

scala> val UserCost = UserYearMap.join(transportmap)
UserCost: org.apache.spark.rdd.RDD[(String, ((Int, Int, Int), Int))] = MapPartitionsRDD[31] at join at <console>:37

scala> val CalcCost = UserCost.map(x => x._2._1._1 -> (x._2._1._3,x._2._1._2 * x._2._2))
CalcCost: org.apache.spark.rdd.RDD[(Int, (Int, Int))] = MapPartitionsRDD[32] at map at <console>:39

scala> val AgeMap = user.map(x => x._1 ->
     |      {
     |      if(x._3<20)
     |      "20"
     |      else if(x._3>35)
     |      "35"
     |      else "20-35"
     |      })
AgeMap: org.apache.spark.rdd.RDD[(Int, String)] = MapPartitionsRDD[33] at map at <console>:29

scala> val CostMap = AgeMap.join(CalcCost).map(x => (x._2._1,x._2._2._1) -> x._2._2._2)
CostMap: org.apache.spark.rdd.RDD[((String, Int), Int)] = MapPartitionsRDD[37] at map at <console>:47

scala> val ExpPeryear = CostMap.groupByKey().map(x => x._1 -> x._2.sum)
ExpPeryear: org.apache.spark.rdd.RDD[((String, Int), Int)] = MapPartitionsRDD[39] at map at <console>:49

scala>
```

```
scala> ExpPeryear.foreach(println)
((35,1992),136000)
((20,1991),102000)
((20,1993),170000)
((20-35,1990),170000)
((35,1993),34000)
((20-35,1991),136000)
((20-35,1994),34000)
((20,1990),34000)
((20-35,1993),34000)
((20,1992),34000)
((20-35,1992),68000)
((35,1990),68000)
((35,1991),68000)

scala>
```


Submitted By Shishir