**Session 18 Assignment 3**

**// Importing and creating tuples for respective data sets**

scala> val traveldata = sc.textFile("/user/S18\_Dataset\_Holidays.txt")

traveldata: org.apache.spark.rdd.RDD[String] = /user/S18\_Dataset\_Holidays.txt MapPartitionsRDD[1] at textFile at <console>:27

scala> var travelTuple = traveldata.map( x=> {

| val row = x.split(",").toList

| (row.apply(0).toInt, row.apply(1), row.apply(2), row.apply(3), row.apply(4).toInt, row.apply(5).toInt)

| })

travelTuple: org.apache.spark.rdd.RDD[(Int, String, String, String, Int, Int)] = MapPartitionsRDD[2] at map at <console>:29

scala> val transport = sc.textFile("/user/S18\_Dataset\_Transport.txt")

transport: org.apache.spark.rdd.RDD[String] = /user/S18\_Dataset\_Transport.txt MapPartitionsRDD[4] at textFile at <console>:27

scala> var transportTuples = transport.map(x=> {

| val row = x.split(",").toList

| (row.apply(0), row.apply(1).toInt)

| })

transportTuples: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[5] at map at <console>:29

scala> val user = sc.textFile("/user/S18\_Dataset\_User\_details.txt")

user: org.apache.spark.rdd.RDD[String] = /user/S18\_Dataset\_User\_details.txt MapPartitionsRDD[7] at textFile at <console>:27

scala> var userTuples = user.map(x=> {

| val row = x.split(",").toList

| (row.apply(0).toInt, row.apply(1), row.apply(2).toInt)

| })

userTuples: org.apache.spark.rdd.RDD[(Int, String, Int)] = MapPartitionsRDD[8] at map at <console>:29

scala>

* **Task – 1**

scala> val travelrdd = travelTuple.map(x => (x.\_1 ,x.\_4)).join(userTuples.map(x => (x.\_1,x.\_3)))

travelrdd: org.apache.spark.rdd.RDD[(Int, (String, Int))] = MapPartitionsRDD[18] at join at <console>:35

scala> val agegroup = userTuples.map(x=> x.\_1->{

| if (x.\_3<20) "<20"

| else if (x.\_3 > 35) ">35"

| else "20-35"

| }).join(travelrdd)

agegroup: org.apache.spark.rdd.RDD[(Int, (String, (String, Int)))] = MapPartitionsRDD[22] at join at <console>:41

scala> val mostamtagegroup = agegroup.map(x => (x.\_2.\_1,x.\_2.\_2.\_2)).groupByKey().map(x => (x.\_1, x.\_2.sum)).sortBy(x => -x.\_2)

mostamtagegroup: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[28] at sortBy at <console>:39

scala> mostamtagegroup.foreach(println)

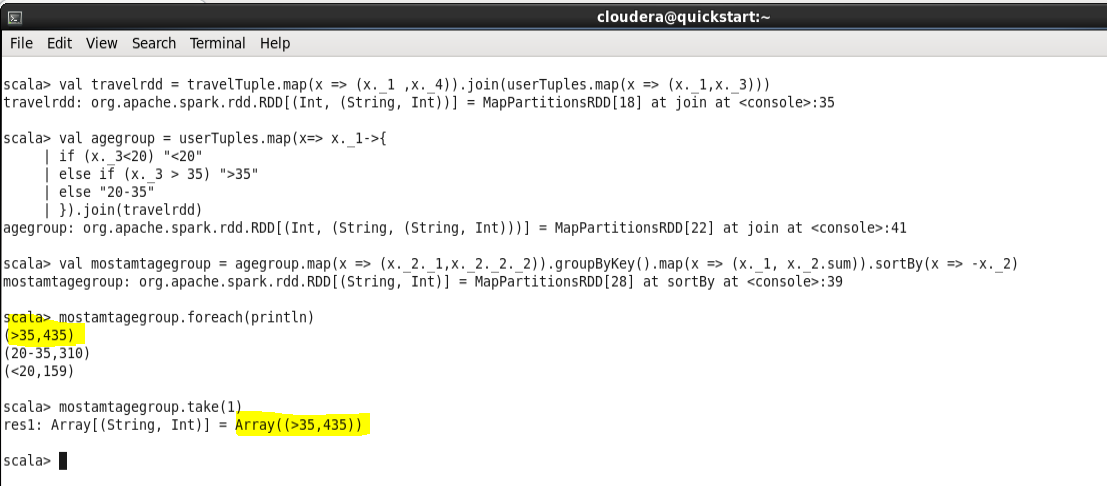
(>35,435)

(20-35,310)

(<20,159)

scala> mostamtagegroup.take(1)

res1: Array[(String, Int)] = Array((>35,435))



* **Task - 2**

scala> val traveldata = travelTuple.map(x => (x.\_1 ,x.\_6)).join(userTuples.map(x => (x.\_1,x.\_3)))

traveldata: org.apache.spark.rdd.RDD[(Int, (Int, Int))] = MapPartitionsRDD[33] at join at <console>:35

scala> val agegroup = userTuples.map(x=> x.\_1->{

| if (x.\_3<20) "<20"

| else if (x.\_3 > 35) ">35"

| else "20-35"

| }).join(traveldata)

agegroup: org.apache.spark.rdd.RDD[(Int, (String, (Int, Int)))] = MapPartitionsRDD[37] at join at <console>:42

scala> val amountPGPY = agegroup.map(x => ((x.\_2.\_1 ,x.\_2.\_2.\_1),x.\_2.\_2.\_2)).groupByKey().map(x => (x.\_1, x.\_2.sum)).sortBy(x => x.\_1.\_1)

amountPGPY: org.apache.spark.rdd.RDD[((String, Int), Int)] = MapPartitionsRDD[43] at sortBy at <console>:40

scala> amountPGPY.foreach(println)

((20-35,1990),117)

((20-35,1991),96)

((20-35,1994),25)

((20-35,1993),22)

((20-35,1992),50)

((<20,1991),49)

((<20,1993),78)

((<20,1992),17)

((<20,1990),15)

((>35,1991),101)

((>35,1990),99)

((>35,1992),191)

((>35,1993),44)

scala>

