**Session 18 Assignment 1**

**// Importing data and creating tuples**

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

* **TASK – 1 // total number of air-travellers per year**

scala> val travelersPYrTask1 = travelTuple.map(x=>(x.\_6,1)).groupByKey().map(x => (x.\_1,(x.\_2.sum)))

travelersPYrTask1: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[5] at map at <console>:31

scala> travelersPYrTask1.count()

res0: Long = 5

scala> travelersPYrTask1.foreach(println)

(1994,1)

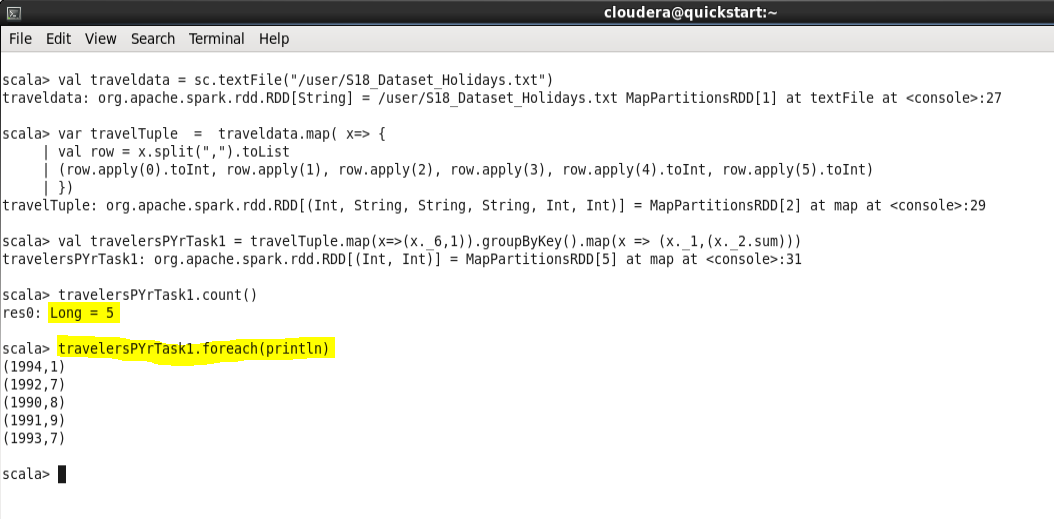
(1992,7)

(1990,8)

(1991,9)

(1993,7)

scala>



* **TASK – 2 // total air distance covered by each user per year**

scala> val groupTask2 = travelTuple.map(x => ((x.\_1,x.\_6),x.\_5)).groupByKey()

groupTask2: org.apache.spark.rdd.RDD[((Int, Int), Iterable[Int])] = ShuffledRDD[26] at groupByKey at <console>:31

scala> val airdistanceTask2 = groupTask2.map(x => (x.\_1,x.\_2.sum)).sortBy(x => (x.\_1,x.\_2))

airdistanceTask2: org.apache.spark.rdd.RDD[((Int, Int), Int)] = MapPartitionsRDD[30] at sortBy at <console>:33

scala> airdistanceTask2.count()

res5: Long = 23

scala> airdistanceTask2.foreach(println)

((1,1990),200)

((1,1993),600)

((2,1991),400)

((2,1993),200)

((3,1991),200)

((3,1992),200)

((3,1993),200)

((4,1990),400)

((4,1991),200)

((5,1991),200)

((5,1992),400)

((5,1994),200)

((6,1991),400)

((6,1993),200)

((7,1990),600)

((8,1990),200)

((8,1991),200)

((8,1992),200)

((9,1991),200)

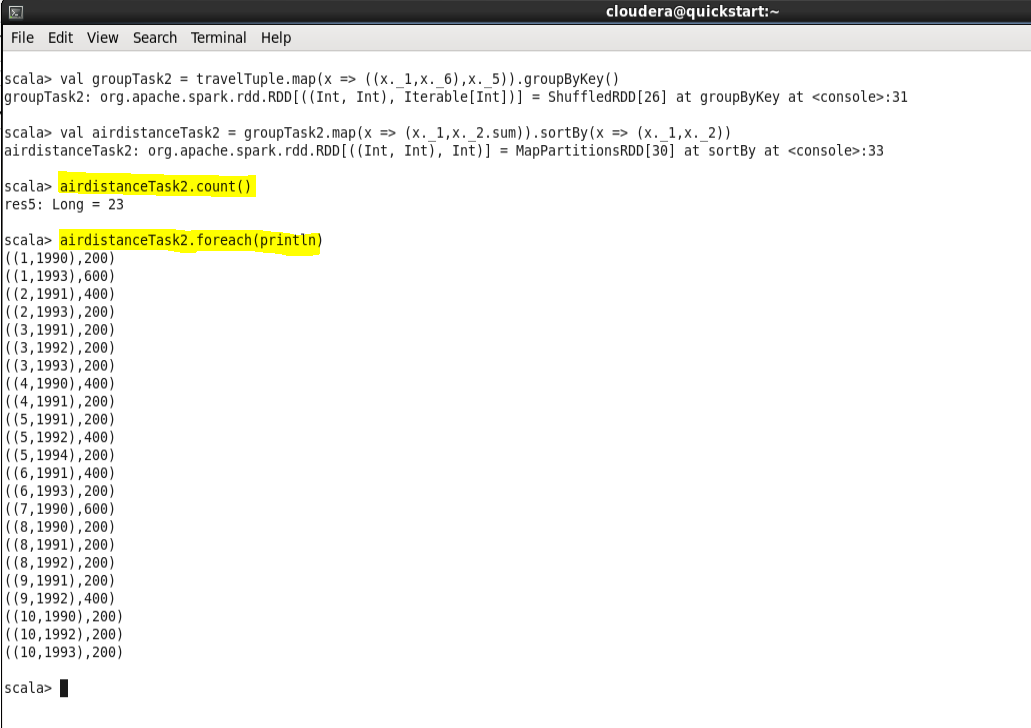
((9,1992),400)

((10,1990),200)

((10,1992),200)

((10,1993),200)

scala>



* **TASK – 3 // The user who travelled the largest distance till date**

scala> val largestdistanceTask3 = travelTuple.map(x => (x.\_1, x.\_5)).groupByKey.map(x => (x.\_1, x.\_2.sum))

largestdistanceTask3: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[33] at map at <console>:31

scala> val sort = largestdistanceTask3.sortBy(x => -x.\_2)

sort: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[36] at sortBy at <console>:33

scala> sort.foreach(println)

(1,800)

(5,800)

(4,600)

(6,600)

(3,600)

(7,600)

(9,600)

(8,600)

(10,600)

(2,600)

scala> val largedistuserTask3 = sort.take(1)

largedistuserTask3: Array[(Int, Int)] = Array((1,800))

scala> largedistuserTask3.foreach(println)

(1,800)

scala> val largedistuserTask3 = sort.take(2)

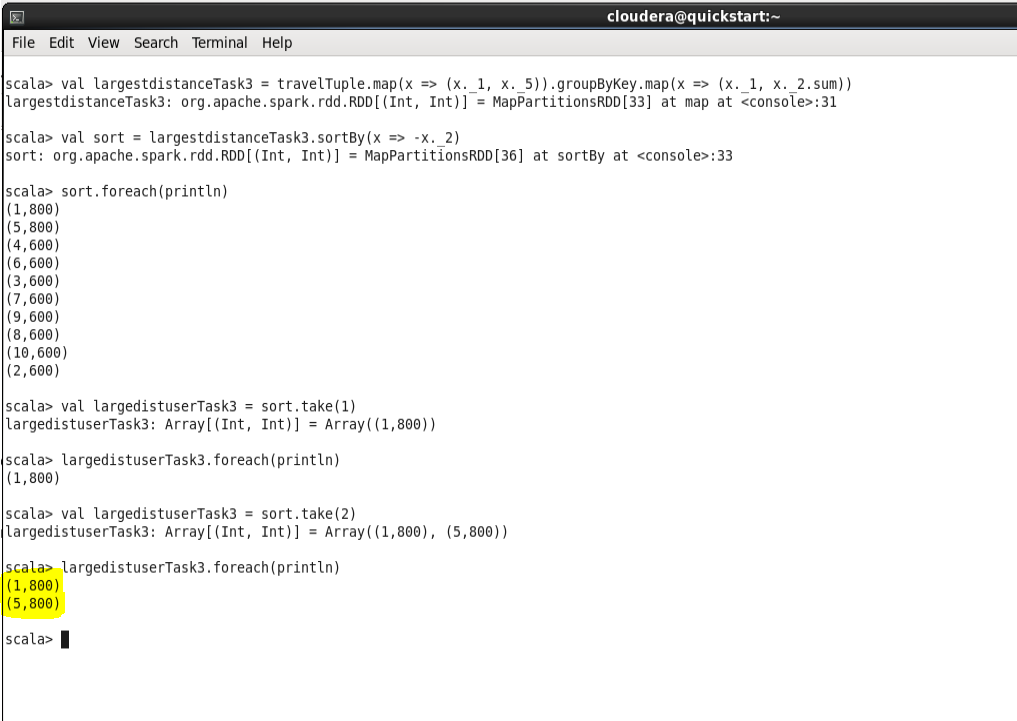
largedistuserTask3: Array[(Int, Int)] = Array((1,800), (5,800))

scala> largedistuserTask3.foreach(println)

(1,800)

(5,800)

scala>



* **TASK – 4 // most preferred destination for all users.**

scala> val preferlocation = travelTuple.map( x=> (x.\_3,1)).groupByKey().map(x =>( x.\_1 ,x.\_2.sum))

preferlocation: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[39] at map at <console>:31

scala> preferlocation.foreach(println)

(CHN,7)

(IND,9)

(PAK,5)

(RUS,6)

(AUS,5)

scala> val sortTask4 = preferlocation.sortBy(x => -x.\_2)

sortTask4: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[42] at sortBy at <console>:33

scala> sortTask4.foreach(println)

(IND,9)

(CHN,7)

(RUS,6)

(PAK,5)

(AUS,5)

scala> val preferdest = sortTask4.take(1)

preferdest: Array[(String, Int)] = Array((IND,9))

scala> preferdest.foreach(println)

(IND,9)

scala>

