Assignment 19 RDD Deep Drive

Task 1

1.Write a program to read a text file and print the number of rows of data in the document.

2. Write a program to read a text file and print the number of words in the document.

3. We have a document where the word separator is -, instead of space. Write a spark code, to obtain the count of the total number of words present in the document.

```
[acadgild@localhost ~]$ pwd
/home/acadgild
[acadgild@localhost ~]$ cat 19_file.txt
Mathew,science,grade-3,45,12
Mathew,history,grade-2,55,13
Mark,maths,grade-2,23,13
Mark,science,grade-1,76,13
John,history,grade-1,14,12
John,maths,grade-2,74,13
Lisa,science,grade-1,24,12
Lisa,history,grade-3,86,13
Andrew,maths,grade-1,34,13
Andrew,science,grade-3,26,14
Andrew,history,grade-1,74,12
Mathew,science,grade-2,55,12
Mathew,history,grade-2,87,12
Mark,maths,grade-1,92,13
Mark,science,grade-2,12,12
John,history,grade-1,67,13
John,maths,grade-1,35,11
Lisa,science,grade-2,24,13
Lisa,history,grade-2,98,15
Andrew,maths,grade-1,23,16
Andrew,science,grade-3,44,14
Andrew,history,grade-2,77,11
```

Answers

1. val file = sc.textFile("file:///home/acadgild/19_file.txt");

scala> file.count()

res17: Long = 22

2.scala> val file1 = file.flatMap(x => x.split(","))

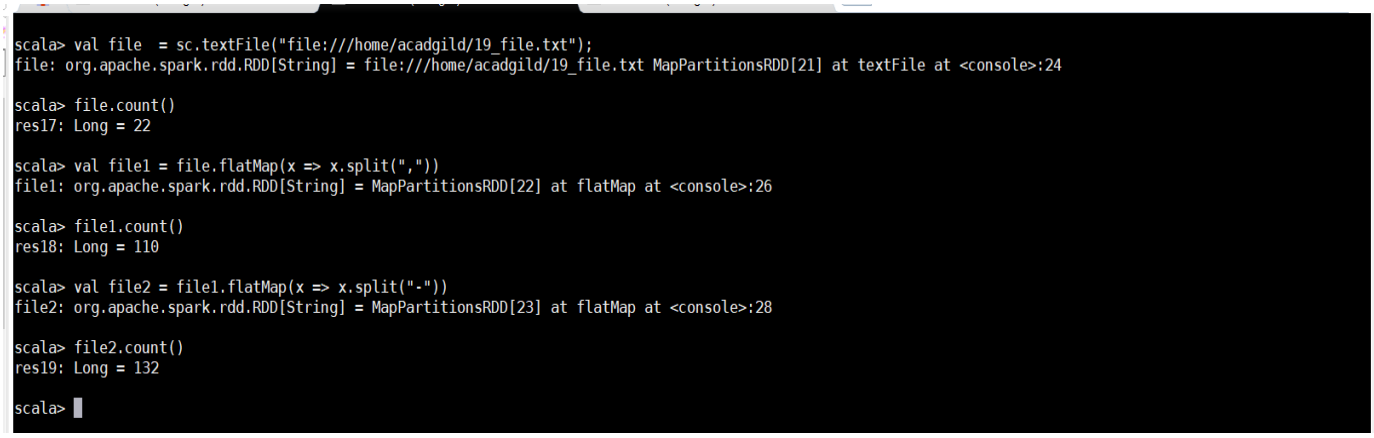
scala> file1.count()

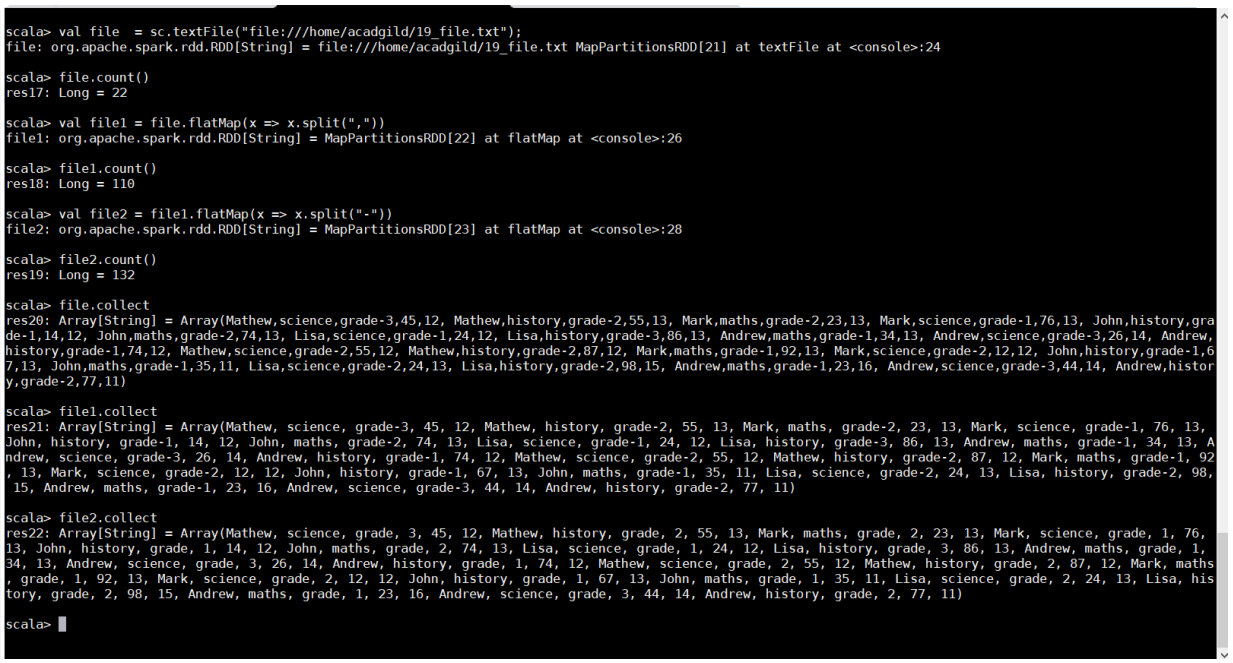
res18: Long = 110

3.scala> val file2 = file1.flatMap(x => x.split("-"))

scala> file2.count()

res19: Long = 132





Task2

Problem Statement 1:

1. Read the text file, and create a tupled rdd.

2. Find the count of total number of rows present.

3. What is the distinct number of subjects present in the entire school

4. What is the count of the number of students in the school, whose name is Mathew and marks is 55

scala> val data = sc.textFile("file:///home/acadgild/19_file.txt")

scala> val arrayTuples = data.map(line => line.split(",")).map(array => (array(0), array(1), array(2), array(3), array(4))).collect

scala> data.count()

res24: Long = 22

scala> val data2 = data.map(x => x.split(","))

data2: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[40] at map at <console>:26

scala> data3.collect

res26: Array[String] = Array(science, history, maths, science, history, maths, science, history, maths, science, history, science, history, maths, science, history, maths, science, history, maths, science, history)

scala> data3.distinct.collect

res27: Array[String] = Array(maths, history, science)

scala> data3.distinct.count

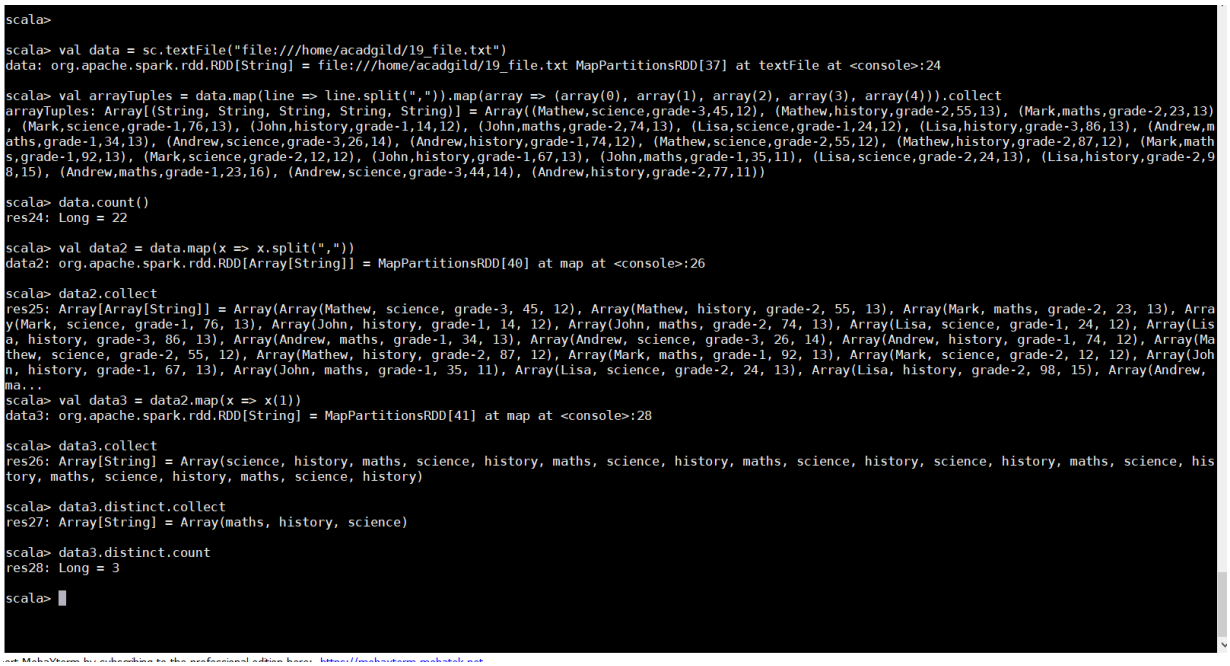
res28: Long = 3

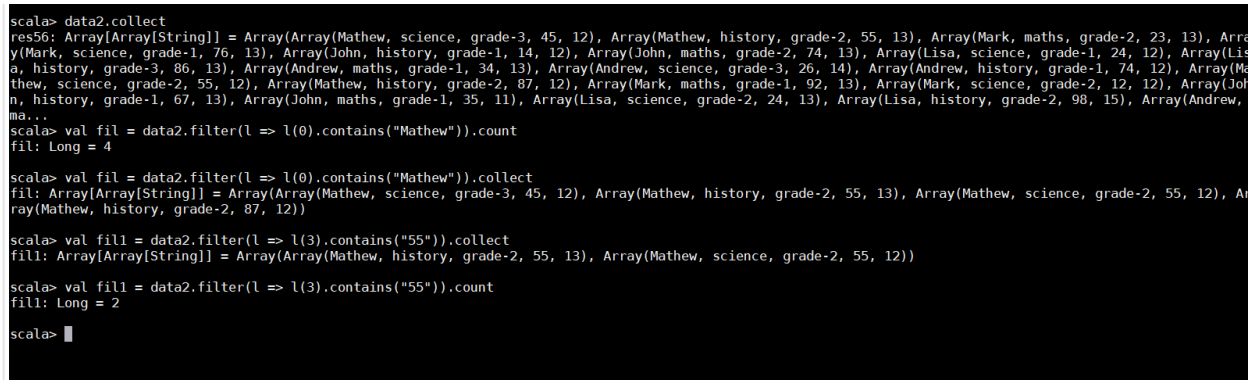
scala> val fil = data2.filter(l => l(0).contains("Mathew")).count

fil: Long = 4

scala> val fil1 = data2.filter(l => l(3).contains("55")).count

fil1: Long = 2





Problem Statement 2:

1. What is the count of students per grade in the school?

2. Find the average of each student (Note - Mathew is grade-1, is different from Mathew in some other grade!)

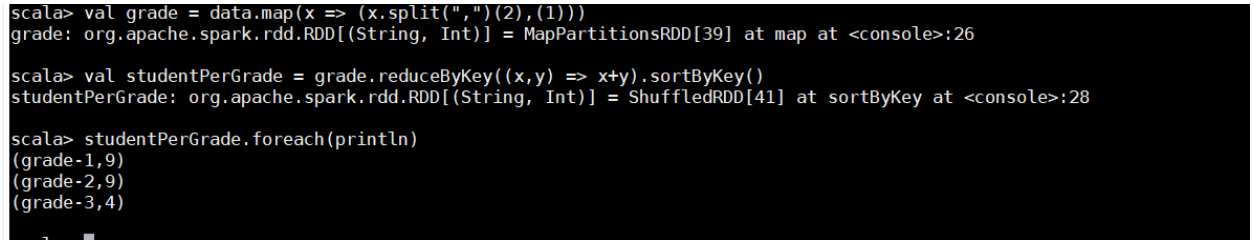
3. What is the average score of students in each subject across all grades?

4. What is the average score of students in each subject per grade?

5. For all students in grade-2, how many have average score greater than 50?

Val data = sc.textFile("file:///home/acadgild/19_file.txt")

1)To find no. of students per grade, we used map function on student to select grade column and mapped it value 1.Then we used reduceByKey to sum up all the values of each grade. Below screenshot shows number of students in every grade.



2)We used map function on student to create RDD subjectGrade for selecting the name and grade as key and marks as value. Each value of marks is mapped with the value 1. For that, we use the following code:

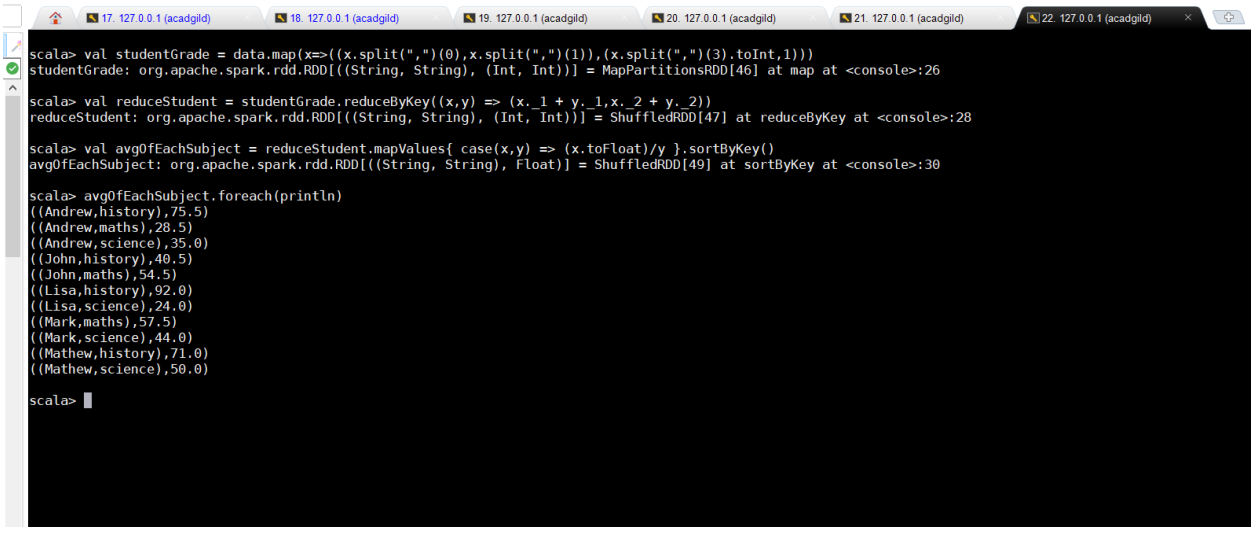
val subjectGrade = student.map(x =>((x.split(",")(0),x.split(",")(2)),(x.split(",")(3).toInt,1)))

Now, we used reduceByKey on subjectGrade to sum up the marks for each key of name and grade: val reduceSubject = subjectGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))

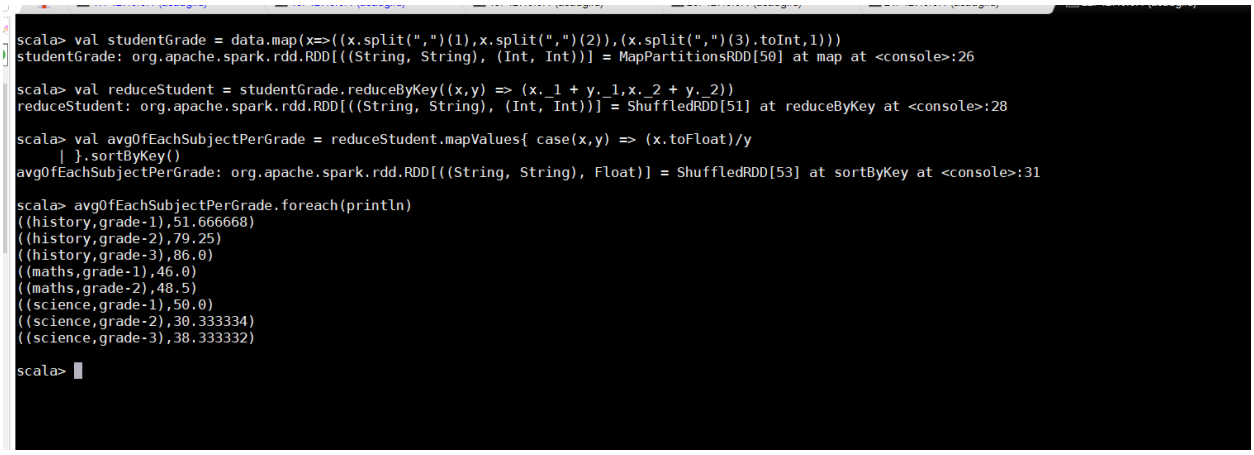
Now, we calculate the average marks by dividing the marks by its count for each key. val avgOfEachStudent = reduceSubject.mapValues{ case (x,y) =>(x.toFloat)/y}.sortByKey() avgOfEachStudent.foreach(println)

```
scala> val subjectGrade = data.map(x =>((x.split(",")(0),x.split(",")(2)),(x.split(",")(3).toInt,1)))
subjectGrade: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[42] at map at <console>:26

scala> val reduceSubject = subjectGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
reduceSubject: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[43] at reduceByKey at <console>:28

scala> reduceSubject.foreach(println)
((Lisa,grade-1),(24,1))
((Mark,grade-2),(35,2))
((Lisa,grade-2),(122,2))
((Mathew,grade-3),(45,1))
((Andrew,grade-2),(77,1))
((Andrew,grade-1),(131,3))
((Lisa,grade-3),(86,1))
((John,grade-1),(116,3))
((John,grade-2),(74,1))
((Mark,grade-1),(168,2))
((Andrew,grade-3),(70,2))
((Mathew,grade-2),(197,3))

scala> val avgOfEachStudent = reduceSubject.mapValues{ case (x,y) =>(x.toFloat)/y}.sortByKey()
avgOfEachStudent: org.apache.spark.rdd.RDD[((String, String), Float)] = ShuffledRDD[45] at sortByKey at <console>:30

scala>

scala> avgOfEachStudent.foreach(println)
((Andrew,grade-1),43.666668)
((Andrew,grade-2),77.0)
((Andrew,grade-3),35.0)
((John,grade-1),38.666668)
((John,grade-2),74.0)
((Lisa,grade-1),24.0)
((Lisa,grade-2),61.0)
((Lisa,grade-3),86.0)
((Mark,grade-1),84.0)
((Mark,grade-2),17.5)
((Mathew,grade-2),65.666664)
((Mathew,grade-3),45.0)

scala>
```

3)To find the average score of student in each subject we follow the same procedure as shown in above question. First, we create the RDD student to read the text file, and then we used map functions on student by creating RDD subjectGrade to select name and subject as key and marks as value and mapped the value 1 to it. Now, we create RDD reduceStudent to sum all the marks with respect to keys name and subject. Now, we calculate the average by dividing the sum of marks by its no of count.

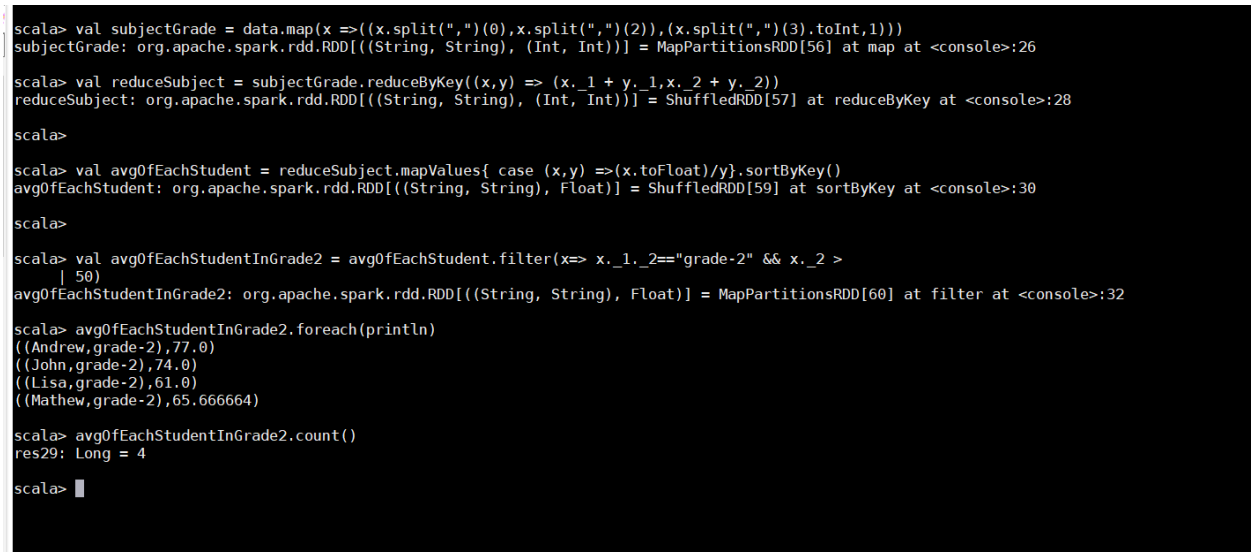


4)To find the average score of student in each subject we follow the same procedure as shown in above question. First, we create the RDD student to read the text file, and then we used map functions on student by creating RDD subjectGrade to select subject and grade as key and marks as value and mapped the value 1 to it. Now, we create RDD reduceStudent to sum all the marks with respect to keys subject and grade. Now, we calculate the average by dividing the sum of marks by its no of count.



```
scala> val studentGrade = data.map(x=>((x.split(",")(1),x.split(",")(2)),(x.split(",")(3).toInt,1)))
studentGrade: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[50] at map at <console>:26

scala> val reduceStudent = studentGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
reduceStudent: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[51] at reduceByKey at <console>:28

scala> val avgOfEachSubjectPerGrade = reduceStudent.mapValues{ case(x,y) => (x.toFloat)/y
     | }.sortByKey()
avgOfEachSubjectPerGrade: org.apache.spark.rdd.RDD[((String, String), Float)] = ShuffledRDD[53] at sortByKey at <console>:31

scala> avgOfEachSubjectPerGrade.foreach(println)
((history,grade-1),51.666668)
((history,grade-2),79.25)
((history,grade-3),86.0)
((maths,grade-1),46.0)
((maths,grade-2),48.5)
((science,grade-1),50.0)
((science,grade-2),30.333334)
((science,grade-3),38.333332)

scala>
```

5)To find the average score of student in each subject we follow the below procedure First, we create the RDD student to read the text file, and then we used map functions on student by creating RDD subjectGrade to select name and grade as key and marks as value and mapped the value 1 to it. Now, we create RDD reduceStudent to sum all the marks with respect to keys subject and grade. Now, we calculate the average by dividing the sum of marks by its no of count. After getting the average value, we used filter function to filter the marks greater than the average value 50 and grade is grade-2.



```
scala> val subjectGrade = data.map(x =>((x.split(",")(0),x.split(",")(2)),(x.split(",")(3).toInt,1)))
subjectGrade: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[56] at map at <console>:26

scala> val reduceSubject = subjectGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
reduceSubject: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[57] at reduceByKey at <console>:28

scala>

scala> val avgOfEachStudent = reduceSubject.mapValues{ case (x,y) =>(x.toFloat)/y}.sortByKey()
avgOfEachStudent: org.apache.spark.rdd.RDD[((String, String), Float)] = ShuffledRDD[59] at sortByKey at <console>:30

scala>

scala> val avgOfEachStudentInGrade2 = avgOfEachStudent.filter(x=> x._1._2=="grade-2" && x._2 >
     | 50)
avgOfEachStudentInGrade2: org.apache.spark.rdd.RDD[((String, String), Float)] = MapPartitionsRDD[60] at filter at <console>:32

scala> avgOfEachStudentInGrade2.foreach(println)
((Andrew,grade-2),77.0)
((John,grade-2),74.0)
((Lisa,grade-2),61.0)
((Mathew,grade-2),65.666664)

scala> avgOfEachStudentInGrade2.count()
res29: Long = 4

scala>
```

Problem Statement 3:

Are there any students in the college that satisfy the below criteria:

1. Average score per student_name across all grades is same as average score per

Big Data Hadoop and Spark Development

student_name per grade

Hint - Use Intersection Property

Answer

we have to calculate the average score per student across all grades first, second, we have to calculate the average score per student per grade and the we have to check intersection between these two scores

```
scala> val studentMarks = data.map(x =>(x.split(",")(0),(x.split(",")(3).toInt,1)))
studentMarks: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[61] at map at <console>:26

scala> val studentMarksReduce = studentMarks.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
studentMarksReduce: org.apache.spark.rdd.RDD[(String, (Int, Int))] = ShuffledRDD[62] at reduceByKey at <console>:28

scala> val studentMarksCount = studentMarksReduce.mapValues{case (x,y)
     | =>(x.toFloat)/y}.sortByKey()
studentMarksCount: org.apache.spark.rdd.RDD[(String, Float)] = ShuffledRDD[64] at sortByKey at <console>:31

scala> val studentMarksAllGrade = studentMarksCount.map(x => x._1+ "," + x._2)
studentMarksAllGrade: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[65] at map at <console>:32

scala> studentMarksAllGrade.foreach(println)
Andrew,46.333332
John,47.5
Lisa,58.0
Mark,50.75
Mathew,60.5
```

```
scala> val subjectGrade = data.map(x =>((x.split(",")(0),x.split(",")(2)),(x.split(",")(3).toInt,1)))
subjectGrade: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[71] at map at <console>:26

scala> val subjectGradeReduce = subjectGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
subjectGradeReduce: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[72] at reduceByKey at <console>:28

scala> val subjectGradeReduceCount = subjectGradeReduce.mapValues{case (x,y)
     | =>(x.toFloat)/y}.sortByKey()
subjectGradeReduceCount: org.apache.spark.rdd.RDD[((String, String), Float)] = ShuffledRDD[74] at sortByKey at <console>:31

scala> val studentMarksPerGrade = subjectGradeReduceCount.map(x => x._1._1 + "," +
     | x._2.toDouble)
studentMarksPerGrade: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[75] at map at <console>:32

scala> studentMarksPerGrade.foreach(println)
Andrew,43.66666793823242
Andrew,77.0
Andrew,35.0
John,38.66666793823242
John,74.0
Lisa,24.0
Lisa,61.0
Lisa,86.0
Mark,84.0
Mark,17.5
Mathew,65.66666412353516
Mathew,45.0

scala> val commanStudents = studentMarksAllGrade.intersection(studentMarksPerGrade)
commanStudents: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[81] at intersection at <console>:42

scala> commanStudents.foreach(println)

scala>
```