BIG DATA HADOOP & SPARK TRAINING

Assignment on RDD's Deep Dive

Rashmi Krishna 5/4/18

Contents

Input Dataset: 2

Task 1 2

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

2. Write a program to read a text file and print the number of words in the document. 4

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. 5

Task 2 6

Problem Statement 1: Read the text file, and create a tupled rdd. 6

2.Find the count of total number of rows present. 6

3.What is the distinct number of subjects present in the entire school 6

4. What is the count of the number of students in the school, whose name is Mathew and marks is 55 7

Problem Statement 2: 7

Problem Statement 3: 13

Input Dataset:

Newdataset.txt

newdataset - Notepad

File Edit Format View Help

```
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
```

Task 1

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

We can extract the contents of the text file stored in local file system, using the SparkContext:-

val textFileLocalTest = sc.textFile("file:///home/acadgild/newdataset.txt");

Now we will read the contents of the file using the below command:-

textFileLocalTest.foreach(println)

```
scala> val textFileLocalTest = sc.textFile("file:///home/acadgild/newdataset.txt");
textFileLocalTest: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/newdataset.txt MapPartitionsRDD[3] at textFile at <console>
:24

scala> textFileLocalTest.foreach(println)
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

scala>
```

contents of the file

We will count the number of rows in the file using the command below:

textFileLocalTest.count

```
scala> textFileLocalTest.count
res2: Long = 22

scala>
```

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

We can extract the contents of the text file stored in local file system, using the SparkContext:-

val textFileLocalTest = sc.textFile("file:///home/acadgild/newdataset.txt");

Now we will read the contents of the file using the below command:-

val words = textFileLocalTest.flatMap(_.split("-"))

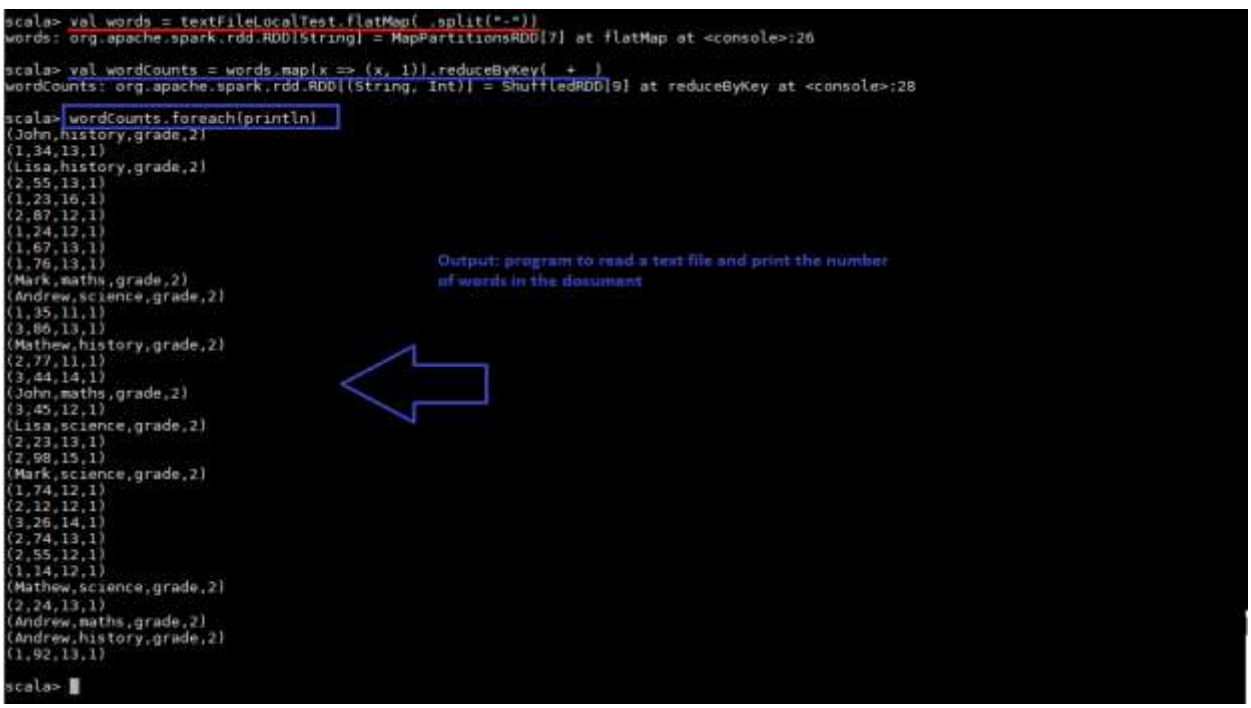
This command will split the text file based on the "-" separator.

val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)

This command will sum up all the rows and sort them accordingly.

wordCounts.foreach(println)

This command will print the number of words in the text file.



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.

To obtain the count of total number of words present in the document, we simply run the ".count" command i.e.

wordCounts.count

```
scala> wordCounts.foreach(println)
(John,history,grade,2)
(1,34,13,1)
(Lisa,history,grade,2)
(2,55,13,1)
(1,23,16,1)
(2,87,12,1)
(1,24,12,1)
(1,67,13,1)
(1,76,13,1)
(Mark,maths,grade,2)
(Andrew,science,grade,2)
(1,35,11,1)
(3,86,13,1)
(Mathew,history,grade,2)
(2,77,11,1)
(3,44,14,1)
(John,maths,grade,2)
(3,45,12,1)
(Lisa,science,grade,2)
(2,23,13,1)
(2,98,15,1)
(Mark,science,grade,2)
(1,74,12,1)
(2,12,12,1)
(3,26,14,1)
(2,74,13,1)
(2,55,12,1)
(1,14,12,1)
(Mathew,science,grade,2)
(2,24,13,1)
(Andrew,maths,grade,2)
(Andrew,history,grade,2)
(1,92,13,1)

scala> wordCounts.count
res10: Long = 33
```

Output:count of the total number of words present in the document

Task 2

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

To create a tuple from the contents of the text file and display the contents of the tuple, we use the following commands:

val file = textFileLocalTest.map(x=> x.split(","))

file.collect

```
scala> val file = textFileLocalTest.map(x=> x.split(","))
file: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[10] at map at <console>:26

scala> file.collect
res12: Array[Array[String]] = Array(Array(Mathew, science, grade-3, 45, 12), Array(Mathew, history, grade-2, 55, 13), Array(Mark, maths
, grade-2, 23, 13), Array(Mark, science, grade-1, 76, 13), Array(John, history, grade-1, 14, 12), Array(John, maths, grade-2, 74, 13),
Array(Lisa, science, grade-1, 24, 12), Array(Lisa, history, grade-3, 86, 13), Array(Andrew, maths, grade-1, 34, 13), Array(Andrew, scie
nce, grade-3, 26, 14), Array(Andrew, history, grade-1, 74, 12), Array(Mathew, science, grade-2, 55, 12), Array(Mathew, history, grade-2
, 87, 12), Array(Mark, maths, grade-1, 92, 13), Array(Mark, science, grade-2, 12, 12), Array(John, history, grade-1, 67, 13), Array(Joh
n, maths, grade-1, 35, 11), Array(Lisa, science, grade-2, 24, 13), Array(Lisa, history, grade-2, 98, 15), Array(Andrew, ma...
scala>
```

Output: Read the text file, and create a tupled rdd.

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

To count total number of rows present we run the below command:

File.count

```
scala> file.count
res13: Long = 22

scala>
```

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

To find the distinct number of subjects in the school we use RDD Action i.e. "distinct", this removes all the duplicate records.

val newfile = file.map(subj =>(subj(1)))

This command will map the subject part of the tuple "file"

newfile.distinct.collect

This will remove all the duplicates and return only unique values.

```
scala> val newfile = file.map(subj =>(subj(1)))
newfile: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[12] at map at <console>:28

scala> newfile.distinct.collect
res15: Array[String] = Array(maths, history, science)

scala>
```

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

To accomplish this task, we split the text file by "," separator

val file = textFileLocalTest.map(x=> x.split(","))

then, we filter the variable "file" where marks are equal to "55" and student name is "Mathew"

val fil = file.filter(x =>(x(3).toInt.equals(55)&& x(0).contains("Mathew")))

```
scala> val file = textFileLocalTest.map(x=> x.split(","))
file: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[23] at map at <console>:26

scala> val fil = file.filter(x =>( x(3).toInt.equals(55)&& x(0).contains("Mathew"))).collect
fil: Array[Array[String]] = Array(Array(Mathew, history, grade-2, 55, 13), Array(Mathew, science, grade-2, 55, 12))

scala>
```

Output: Count of the number of students in the school, whose name is Mathew and marks is 55

Problem Statement 2:

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

To count the students per grade in the school, we have to perform the following tasks:

- We map the tuples with only two values, i.e. grade(2) which indicates the grades and grade(0) which indicates the name of the student.

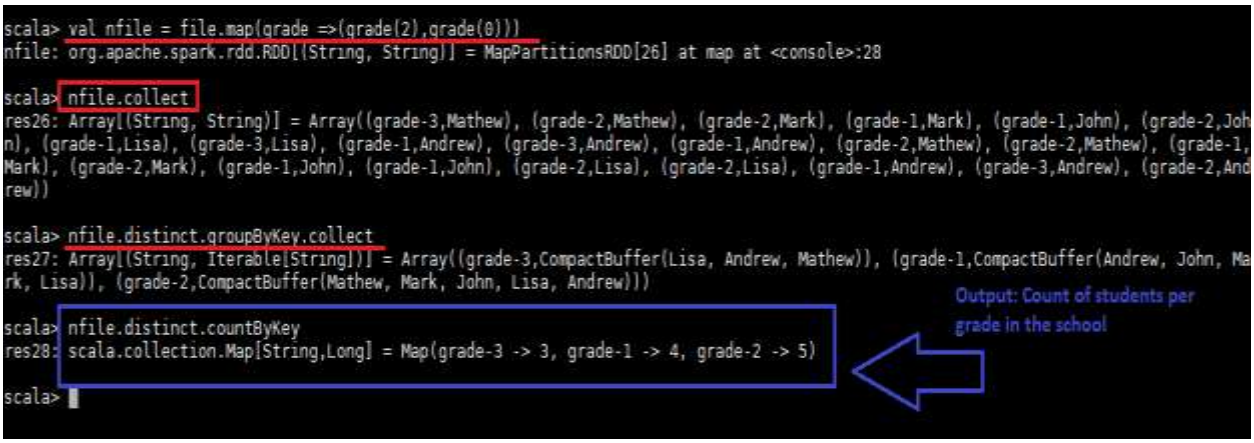
  val nfile = file.map(grade =>(grade(2),grade(0))).collect

- We perform groupByKey, to know how many students fall under each grade

  nfile.distinct.groupByKey.collect

- We find the distinct values and calculate a count of it.

  nfile.distinct.countByKey



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

To calculate average we need to do the following things:-

- We extract the contents of the text file to spark and then split the tuple based on "," separator and extract only 3 fields i.e. student_name, subject and marks

val student = sc.textFile("file:///home/acadgild/newdataset.txt");

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

- We group them by adding 1 to each of the row and calculate a sum of them and sort them base on the key, this is to calculate the countsof number of records. In the same we calculate the total of all marks for each student and subject

val total_len = avg_each_stu.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()

val total_marks = avg_each_stu.reduceByKey(_+_).sortByKey()

- We now join both total marks and counts

val join_data = total_marks.join(total_len)

- Now, we calculate average, as we know average = sum/counts. We could compute the same:

val result_avg = join_data.map(x => ((x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)

```
scala> val student = sc.textFile("file:///home/acadgild/newdataset.txt");
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/newdataset.txt MapPartitionsRDD[37] at textFile at <console>:24

scala> val avg_each_stu = student.map( x => (x.toString.split(","))).map(x => ((x(0),x(1)),x(3).toInt))
avg_each_stu: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[39] at map at <console>:26

scala> val total_len = avg_each_stu.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()
total_len: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[42] at sortByKey at <console>:28

scala> val total_marks = avg_each_stu.reduceByKey(_+_).sortByKey()
total_marks: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[44] at sortByKey at <console>:28

scala> val join_data = total_marks.join(total_len)
join_data: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[47] at join at <console>:32

scala> val result_avg = join_data.map(x => ( ( x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)
(Lisa,history) ===> 92
(Mark,maths) ===> 57
(Andrew,science) ===> 35
(Mark,science) ===> 44
(Mathew,science) ===> 50
(Andrew,maths) ===> 28
(Mathew,history) ===> 71
(John,maths) ===> 54
(John,history) ===> 40
(Lisa,science) ===> 24
(Andrew,history) ===> 75
result_avg: Unit = ()

scala> total_marks.foreach(println)
((Andrew,history),151)
((Andrew,maths),57)
((Andrew,science),70)
((John,history),81)
((John,maths),109)
((Lisa,history),184)
((Lisa,science),48)
((Mark,maths),115)
((Mark,science),88)
((Mathew,history),142)
((Mathew,science),100)

scala>
```

Output: Average of each student

This is total marks

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

To calculate average we need to do the following things:-

- We extract the contents of the text file to spark and then split the tuple based on "," separator and extract only 4 fields i.e. student_name, subject, grades and marks

val student = sc.textFile("file:///home/acadgild/newdataset.txt");

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

- We group them by adding 1 to each of the row and calculate a sum of them and sort them base on the key, this is to calculate the counts of number of records. In the same we calculate the total of all marks for each student and subject

val total_len = avg_each_stu.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()

val total_marks = avg_each_stu.reduceByKey(_+_).sortByKey()

- We now join both total marks and counts

val join_data = total_marks.join(total_len)

- Now, we calculate average, as we know average = sum/counts. We could compute the same:

val result_avg = join_data.map(x => ((x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)

```
scala> val student = sc.textFile("file:///home/acadgild/newdataset.txt");
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/newdataset.txt MapPartitionsRDD[350] at textFile at <console>:24

scala> val avg_each_stu = student.map( x => (x.toString.split(","))).map(x => ((x(1),x(2)),x(3).toInt))
avg_each_stu: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[352] at map at <console>:26

scala> val total_len = avg_each_stu.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()
total_len: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[355] at sortByKey at <console>:28

scala> val total_marks = avg_each_stu.reduceByKey(_+_).sortByKey()
total_marks: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[357] at sortByKey at <console>:28

scala> val join_data = total_marks.join(total_len)
join_data: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[360] at join at <console>:32

scala> val result_avg = join_data.map(x => ( ( x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)
(history,grade-2) ===> 79
(history,grade-3) ===> 86
(maths,grade-1) ===> 46
(science,grade-3) ===> 38
(science,grade-1) ===> 50
(science,grade-2) ===> 30
(history,grade-1) ===> 51
(maths,grade-2) ===> 48
result_avg: Unit = ()

scala> total_marks.foreach(println)
((history,grade-1),155)
((history,grade-2),317)
((history,grade-3),86)
((maths,grade-1),184)
((maths,grade-2),97)
((science,grade-1),100)
((science,grade-2),91)
((science,grade-3),115)
```

Output: Average score of students in each subject across all grades

Sum of marks

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

To calculate average we need to do the following things:-

- We extract the contents of the text file to spark and then split the tuple based on "," separator and extract only 4 fields i.e. student_name, subject, grades and marks

val student = sc.textFile("file:///home/acadgild/newdataset.txt");

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

- We group them by adding 1 to each of the row and calculate a sum of them and sort them base on the key, this is to calculate the counts of number of records. In the same we calculate the total of all marks for each student and subject

val total_len = avg_each_stu.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()

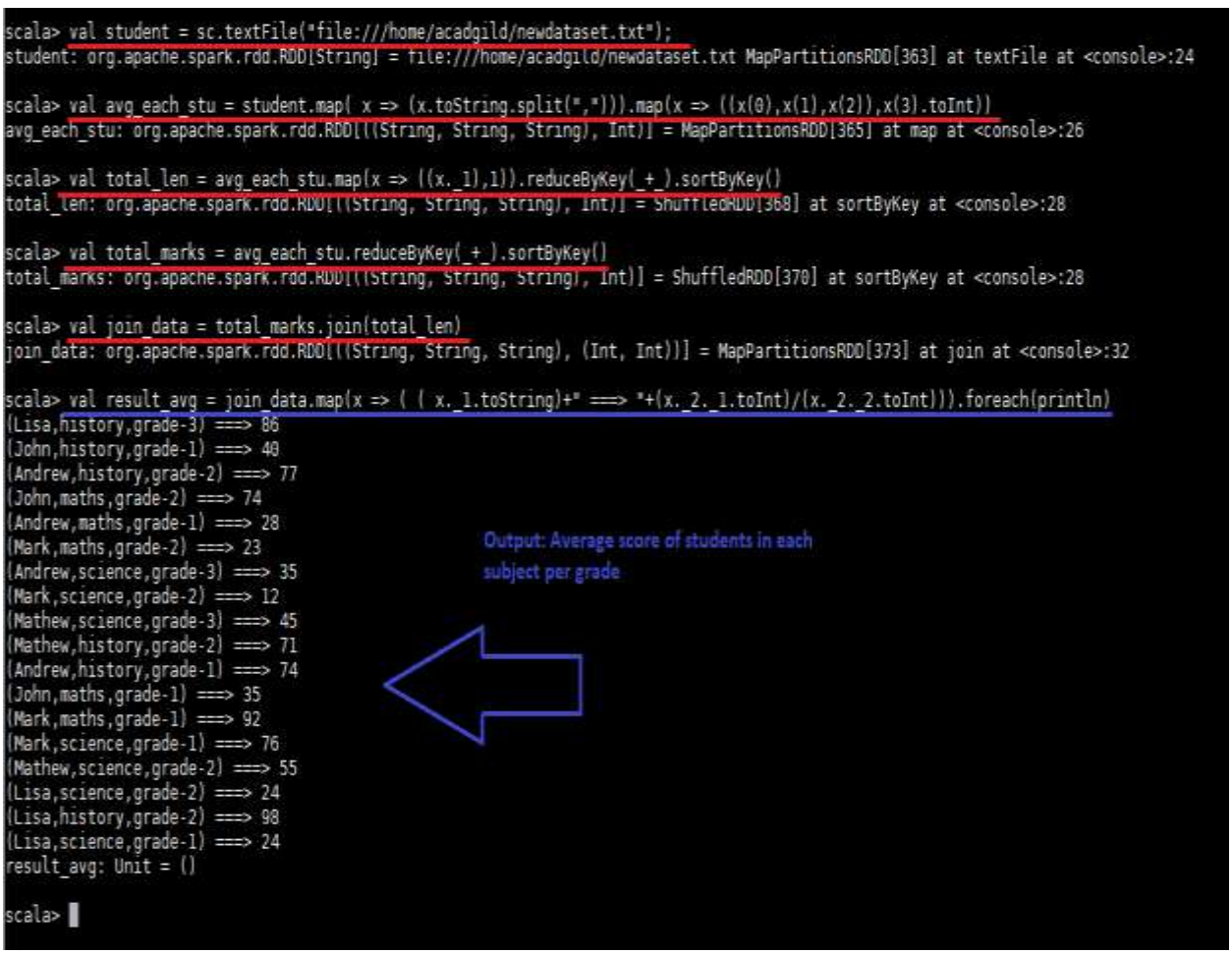
val total_marks = avg_each_stu.reduceByKey(_+_).sortByKey()

- We now join both total marks and counts

val join_data = total_marks.join(total_len)

- Now, we calculate average, as we know average = sum/counts. We could compute the same:

val result_avg = join_data.map(x => ((x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)



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

To calculate average we need to do the following things:-

- We extract the contents of the text file to spark and then split the tuple based on "," separator and extract only 4 fields i.e. student_name, subject, grades and marks

val student = sc.textFile("file:///home/acadgild/newdataset.txt");

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

- We group them by adding 1 to each of the row and calculate a sum of them and sort them base on the key, this is to calculate the counts of number of records. In the same we calculate the total of all marks for each student and subject.

  val total_len = avg_each_stu.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()

  val total_marks = avg_each_stu.reduceByKey(_+_).sortByKey()

- We now join both total marks and counts

  val join_data = total_marks.join(total_len)

- Now, we calculate average, as we know average = sum/counts. We could compute the same:

  val result_avg = join_data.map(x => ((x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)

- Now, we filter the average so that we get to see only the averages of "grade-2" , of which whose averages are >=50

val res_fil = result_avg.filter(x => (x._1.contains("grade-2")) && (x._2 >= 50)).foreach(println)

```
scala> val student = sc.textFile("file:///home/acadgild/newdataset.txt");
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/newdataset.txt MapPartitionsRDD[418] at textFile at <console>:24

scala> val avg_each_stu = student.map( x => (x.toString.split(",")) ).map(x => ((x(0),x(2)),x(3).toInt))
avg_each_stu: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[420] at map at <console>:26

scala> val total_len = avg_each_stu.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()
total_len: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[423] at sortByKey at <console>:28

scala> val total_marks = avg_each_stu.reduceByKey(_+_).sortByKey()
total_marks: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[425] at sortByKey at <console>:28

scala> val join_data = total_marks.join(total_len)
join_data: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[428] at join at <console>:32

scala> val result_avg = join_data.map(x => ( ( x._1.toString),(x._2._1.toInt)/(x._2._2.toInt)))
result_avg: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[429] at map at <console>:34

scala> result_avg.foreach(println)
((Lisa,grade-1),24)
((Mark,grade-2),17)
((Lisa,grade-2),61)
((Mathew,grade-3),45)
((Andrew,grade-2),77)
((Andrew,grade-1),43)
((Lisa,grade-3),86)
((John,grade-1),38)
((John,grade-2),74)
((Mark,grade-1),84)
((Andrew,grade-3),35)
((Mathew,grade-2),65)

scala> val res_fil = result_avg.filter( x =>  (x._1.contains("grade-2")) && (x._2 >= 50)).foreach(println)
((Lisa,grade-2),61)
((Andrew,grade-2),77)
((John,grade-2),74)
((Mathew,grade-2),65)
```

Output: Students in grade-2, with average score greater than 50

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 student_name per grade Hint - Use Intersection Property**

To accomplish this task we perform the following steps:-

We first calculate average per grade:-

- We extract the contents of the text file to spark and then split the tuple based on "," separator and extract only 4 fields i.e. student_name, subject, grades and marks

val student = sc.textFile("file:///home/acadgild/newdataset.txt");

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

- We group them by adding 1 to each of the row and calculate a sum of them and sort them base on the key, this is to calculate the counts of number of records. In the same we calculate the total of all marks for each student and subject.

  val total_len1 = avg_each_grade.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()

  val total_mark1s = avg_each_grade.reduceByKey(_+_).sortByKey()

- We now join both total marks and counts

  val join_data = total_marks1.join(total_len1)

- Now, we calculate average, as we know average = sum/counts. We could compute the same:

  val result_avg = join_data.map(x => ((x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)

```
scala> val grade = sc.textFile("file:///home/acadgild/newdataset.txt");
grade: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/newdataset.txt MapPartitionsRDD[312] at textFile at <console>:24

scala> val avg_each_grade = grade.map( x => (x.toString.split(",")) ).map(x => ((x(0),x(2)),x(3).toInt))
avg_each_grade: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[314] at map at <console>:26

scala> val total_len1 = avg_each_grade.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()
total_len1: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[317] at sortByKey at <console>:28

scala> val total_marks1 = avg_each_grade.reduceByKey(_+_).sortByKey()
total_marks1: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[319] at sortByKey at <console>:28

scala> val join_data = total_marks1.join(total_len1)
join_data: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[322] at join at <console>:32

scala> val result_avg_grade = join_data.map(x => ( ( x._1.toString),(x._2._1.toInt)/(x._2._2.toInt)))
result_avg_grade: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[323] at map at <console>:34

scala> result_avg_grade.foreach(println)
((Lisa,grade-1),24)
((Mark,grade-2),17)
((Lisa,grade-2),61)
((Mathew,grade-3),45)
((Andrew,grade-2),77)
((Andrew,grade-1),43)
((Lisa,grade-3),86)
((John,grade-1),38)
((John,grade-2),74)
((Mark,grade-1),84)
((Andrew,grade-3),35)
((Mathew,grade-2),65)
```

Average of students across all grades

We then calculate average base on students:-

- We extract the contents of the text file to spark and then split the tuple based on "," separator and extract only 4 fields i.e. student_name, subject, grades and marks

val student = sc.textFile("file:///home/acadgild/newdataset.txt");

val avg_each_stu = student.map(x => (x.toString.split(",")).map(x => (x(0),x(3).toInt))

- We group them by adding 1 to each of the row and calculate a sum of them and sort them base on the key, this is to calculate the counts of number of records. In the same we calculate the total of all marks for each student and subject.

  val total_len = avg_each_stu.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()

  val total_marks = avg_each_stu.reduceByKey(_+_).sortByKey()

- We now join both total marks and counts

  val join_data = total_marks.join(total_len)

- Now, we calculate average, as we know average = sum/counts. We could compute the same:

  val result_avg = join_data.map(x => ((x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)

```
scala> val student = sc.textFile("file:///home/acadgild/newdataset.txt");
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/newdataset.txt MapPartitionsRDD[325] at textFile at <console>:24

scala> val avg_each_stu = student.map( x => (x.toString.split(",")) ).map(x => ((x(0)),x(3).toInt))
avg_each_stu: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[327] at map at <console>:26

scala> val total_len = avg_each_stu.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()
total_len: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[330] at sortByKey at <console>:28

scala> val total_marks = avg_each_stu.reduceByKey(_+_).sortByKey()
total_marks: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[332] at sortByKey at <console>:28

scala> val join_data = total_marks.join(total_len)
join_data: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[335] at join at <console>:32

scala> val result_avg_stu = join_data.map(x => ( ( x._1.toString),(x._2._1.toInt)/(x._2._2.toInt)))
result_avg_stu: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[336] at map at <console>:34

scala> result_avg_stu.foreach(println)
(Mark,50)
(Andrew,46)
(Mathew,60)
(John,47)
(Lisa,58)
```

Average of each students across all grades

Now we find the intersection of average of marks across grades and average of marks across students.

We can observe that, intersection action returned null values this indicates that average score per student across all grades is not same as average score per student.

```
scala> result_avg_grade.intersection(result_avg_stu)
res71: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[342] at intersection at <console>:47

scala> result_avg_grade.intersection(result_avg_stu).foreach(println)

scala>
```

This indicates that, average score per student_name across all grades is not same as average score per student_name per grade