Assignment 19

Dataset for Task1

File name: new.txt

Content

Mathew-science-45-12

Mark-maths-23-13

John-history-67-13

Lisa-science-24-13

Task 1

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

```
val data = sc.textFile("file:///home/acadgild/new.txt").count
```

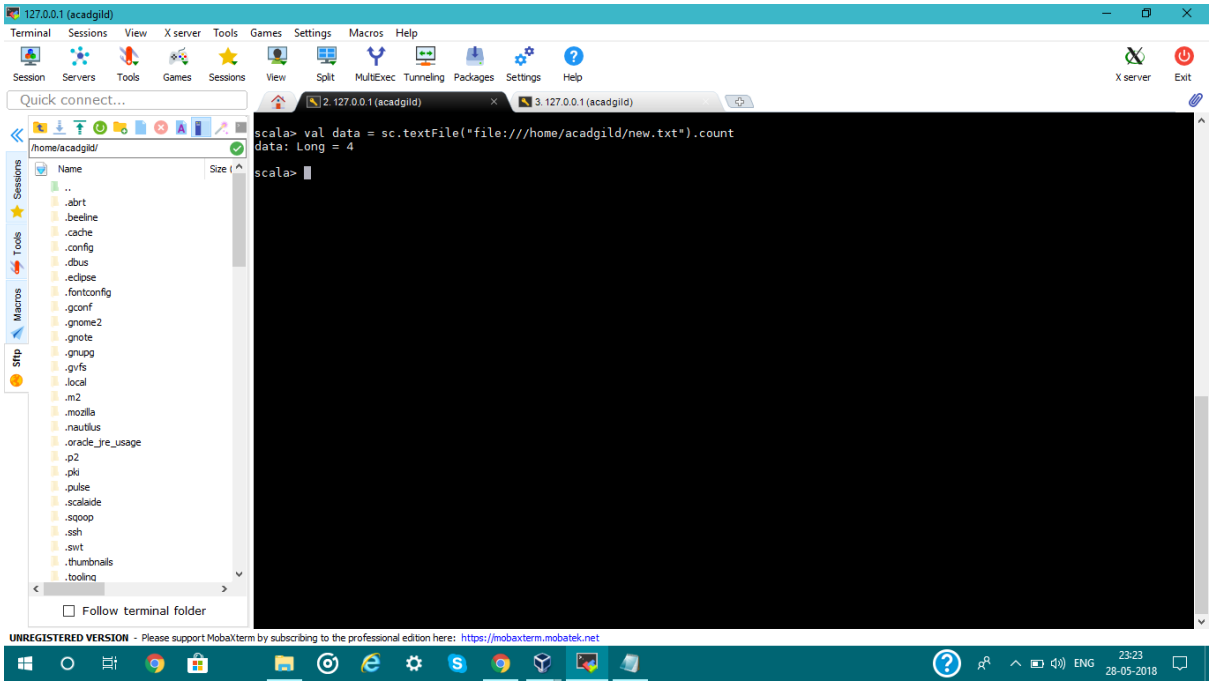
b. Write a program to read a text file and print the number of words in the document

```
val data = sc.textFile("file:///home/acadgild/new.txt").flatMap(x=>x.split("-"))
                                        //split the line into words
data.collect.foreach (println)
```

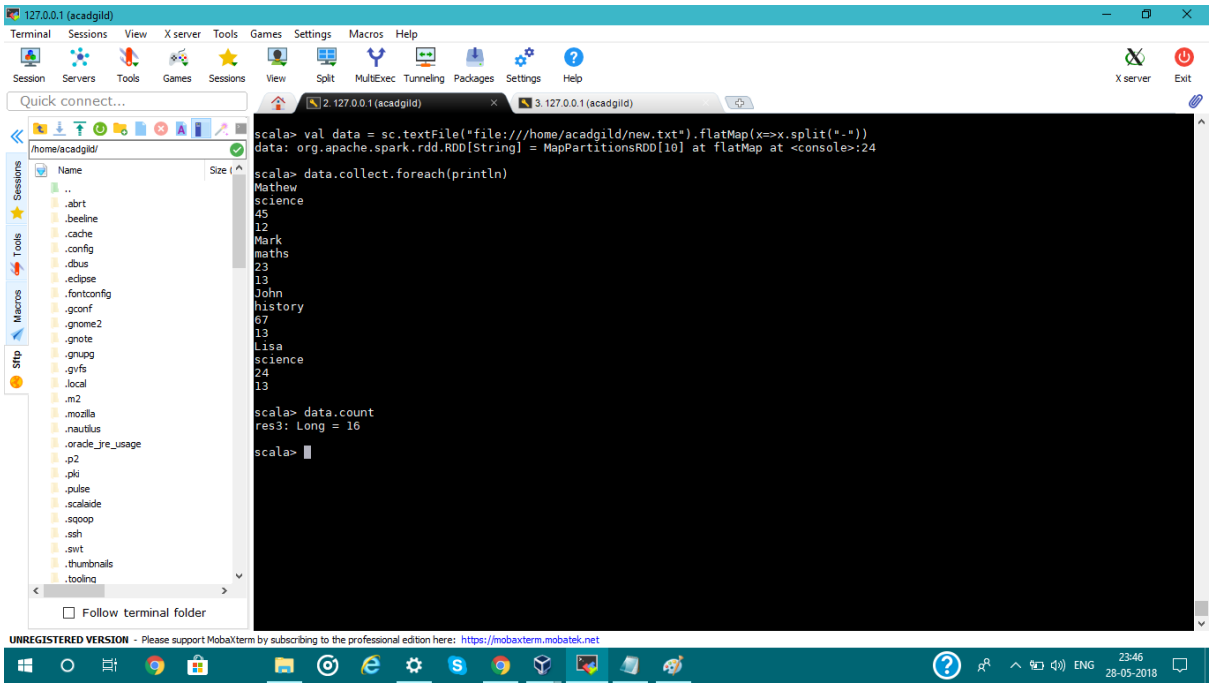
c. We have a document where the word separator is -, instead of space. Write a spark code to obtain the count of the total number of words present in the document

```
val data = sc.textFile("file:///home/acadgild/new.txt").flatMap(x=>x.split("-"))

data.count              //count the number of words
```

Screenshots


127.0.0.1 (acadgild)
scala> val data = sc.textFile("file:///home/acadgild/new.txt").count
data: Long = 4
scala>
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

a.


127.0.0.1 (acadgild)
scala> val data = sc.textFile("file:///home/acadgild/new.txt").flatMap(x=>x.split("-"))
data: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[10] at flatMap at <console>:24
scala> data.collect.foreach(println)
Mathew
science
45
12
Mark
maths
23
13
John
history
67
13
Lisa
science
24
13
scala> data.count
res3: Long = 16
scala>
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

b., c.

Task2

Problem Statement 1

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

```
val data = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x =>
(x.split(",")(0),(x.split(",")(1),x.split(",")(2),x.split(",")(3).toInt,x.split(",")(4).toInt)))    // a tuple
RDD with name as Key and the subject, grades and the marks as values

data.collect.foreach(println)                                    //printing each line
```

b. Find the count of total number of rows present

```
data.count()
```

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

```
val data= sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=> (x.split(",")(1),1))

val red = data.reduceByKey((x,y)=>(x+y))

red.collect.foreach(println)

//First we are creating a RDD to read the file and selecting only subject name and mapping
them with value 1 and counting the values of occurrences using reduceByKey to get distinct
number of subjects.
```

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

```
val data = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x =>
((x.split(",")(0),x.split(",")(3).toInt),1))

val fil = data.filter(x=>x._1._1 == "Mathew" && x._1._2 == 55)

val red = fil.reduceByKey((x,y)=> x+y).collect.foreach(println)

//In the first line code, we are reading the text file and creating a tuple RDD as "baseRDD"
with name & marks as key and mapping numerical 1 as value. Then, filter the tuple RDD by
providing the condition and then after that, we are counting each occurrences using the
reduceByKey.
```

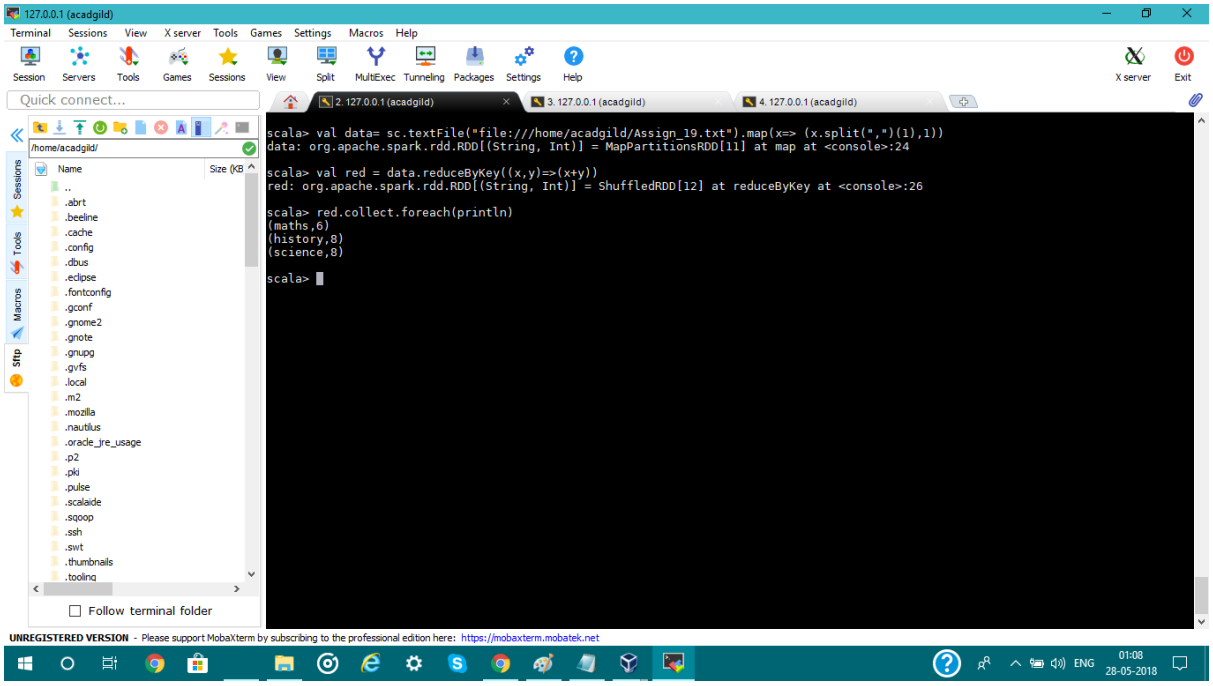
Screenshots

```
  at org.apache.spark.rdd.RDD.partitions(RDD.scala:250)
  at org.apache.spark.SparkContext.runJob(SparkContext.scala:2094)
  at org.apache.spark.rdd.RDD$anonfun$collect$1.apply(RDD.scala:936)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:151)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:112)
  at org.apache.spark.rdd.RDD.withScope(RDD.scala:362)
  at org.apache.spark.rdd.RDD.collect(RDD.scala:935)
  ... 48 elided

scala> val data = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x => (x.split(",")(0),(x.split(",")(1),x.split(",")(2),x.s
plit(",")(3).toInt,x.split(",")(4).toInt)))
data: org.apache.spark.rdd.RDD[(String, (String, String, Int, Int))] = MapPartitionsRDD[8] at map at <console>:24

scala> data.collect.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>
```

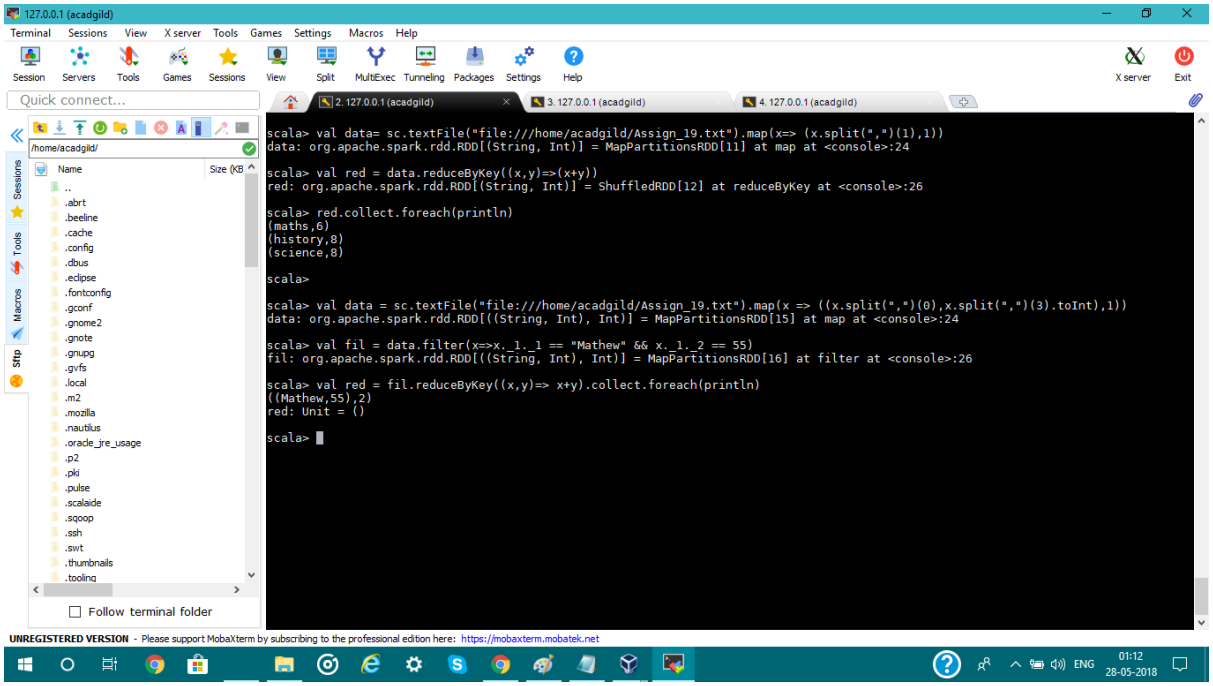
a.

```
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:151)
  at org.apache.spark.rdd.RDDOperationScope$.withScope(RDDOperationScope.scala:112)
  at org.apache.spark.rdd.RDD.withScope(RDD.scala:362)
  at org.apache.spark.rdd.RDD.collect(RDD.scala:935)
  ... 48 elided

scala> val data = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x => (x.split(",")(0),(x.split(",")(1),x.split(",")(2),x.s
plit(",")(3).toInt,x.split(",")(4).toInt)))
data: org.apache.spark.rdd.RDD[(String, (String, String, Int, Int))] = MapPartitionsRDD[8] at map at <console>:24

scala> data.collect.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> data.count()
res8: Long = 22

scala>
```

b.


127.0.0.1 (acadgild)
scala> val data= sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=> (x.split(",")(1),1))
data: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[11] at map at <console>:24
scala> val red = data.reduceByKey((x,y)=>(x+y))
red: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[12] at reduceByKey at <console>:26
scala> red.collect.foreach(println)
(maths,6)
(history,8)
(science,8)
scala>
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

c.


127.0.0.1 (acadgild)
scala> val data= sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=> (x.split(",")(1),1))
data: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[11] at map at <console>:24
scala> val red = data.reduceByKey((x,y)=>(x+y))
red: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[12] at reduceByKey at <console>:26
scala> red.collect.foreach(println)
(maths,6)
(history,8)
(science,8)
scala>
scala> val data = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x => ((x.split(",")(0),x.split(",")(3).toInt),1))
data: org.apache.spark.rdd.RDD[((String, Int), Int)] = MapPartitionsRDD[15] at map at <console>:24
scala> val fil = data.filter(x=>x._1._1 == "Mathew" && x._1._2 == 55)
fil: org.apache.spark.rdd.RDD[((String, Int), Int)] = MapPartitionsRDD[16] at filter at <console>:26
scala> val red = fil.reduceByKey((x,y)=> x+y).collect.foreach(println)
((Mathew,55),2)
red: Unit = ()
scala>
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

d.

Problem Statement 2

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

```
val pr1 =data.map(x =>(x.split(",")(2),1)).reduceByKey((x,y)=>x+y).collect.foreach(println)
```

// we are reading the text file by creating a tuple RDD with grade as key and mapping numerical 1 as values and reducing the number occurrences using reduceByKey

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

```
val data =
sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>((x.split(",")(0),x.split(",")(2)),x
.split( ",")(3).toInt))        // creating RDD to read the file and selecting name and grade as
key and marks as value

val rmap = data.mapValues(x=>(x,1))

val red = rmap.reduceByKey((x,y) => (x._1 + y._1, x._2 + y._2))     // add the occurrences
of marks for each key

red.collect.foreach(println)

val ravg = red.mapValues{case(sum,count)=>(1.0*sum)/count}                    // calculating
average by summing the marks and dividing by its count for each key
```

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

```
val data =
sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>((x.split(",")(0),x.split(",")(1)),x.spl
it( ",")(3).toInt))    // creating RDD to read the text file and we are extracting name and
subject as key and marks as value

val rmap = data.mapValues(x=>(x,1))

val red = rmap.reduceByKey((x,y) => (x._1 + y._1, x._2 + y._2))      // add the
occurrences of marks for each key

red.collect.foreach(println)

val ravg = red.mapValues{case(sum,count)=>(1.0*sum)/count}
```

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

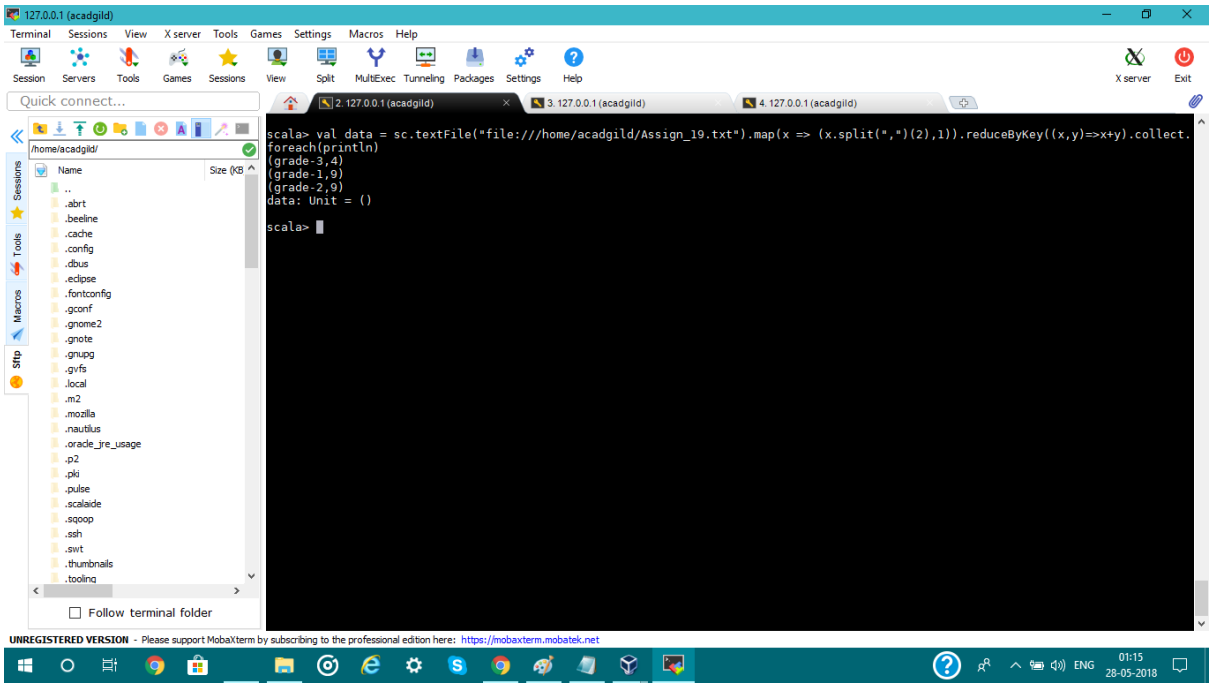
```
val data =
sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>((x.split(",")(1),x.split(",")(2)),x.spl
it( ",")(3).toInt))                    //creating RDD to read the file and extracting subject and
grade as key and marks as value

val rmap = data.mapValues(x=>(x,1))
```

```
val red = rmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))

val Avg_Grade = red.mapValues{case(sum,count)=>(1.0*sum)/count}.foreach(println)

// calculating average by dividing the sum of marks with number of occurrences
```

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

```
val data =
sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>((x.split(",")(0),x.split(",")(2)),x.spl
it( ",")(3).toInt))

val rmap = data.mapValues(x=>(x,1))

val red = rmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))

val ravg = red.mapValues{case(sum,count)=>(1.0*sum)/count}

val rfiltermap = ravg.filter(x=>x._1._2 == "grade-2" && x._2>50).count()          // we are
filtering the above result with student belonging to grade-2 and having marks greater than
50

val rfiltermap = ravg.filter(x=>x._1._2 == "grade-2" && x._2>50).foreach(println)
```

Screenshots



a.

```
data: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[24] at map at <console>:24

scala> val rmap = data.mapValues(x=>(x,1))
rmap: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[25] at mapValues at <console>:26

scala> val red = rmap.reduceByKey((x,y) => (x._1 + y._1, x._2 + y._2))
red: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[26] at reduceByKey at <console>:28

scala> red.collect.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 savg = red.mapValues{case(sum,count)=>(1.0*sum)/count}
savg: org.apache.spark.rdd.RDD[((String, String), Double)] = MapPartitionsRDD[27] at mapValues at <console>:30

scala> savg.collect.foreach(println)
((Lisa,grade-1),24.0)
((Mark,grade-2),17.5)
((Lisa,grade-2),61.0)
((Mathew,grade-3),45.0)
((Andrew,grade-2),77.0)
((Andrew,grade-1),43.666666666666664)
((Lisa,grade-3),86.0)
((John,grade-1),38.666666666666664)
((John,grade-2),74.0)
((Mark,grade-1),84.0)
((Andrew,grade-3),35.0)
((Mathew,grade-2),65.66666666666667)
```

b.

```
scala> val data = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>((x.split(",")(0),x.split(",")(1)),x.split(",")(3).toInt
))
data: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[30] at map at <console>:24

scala> val rmap = data.mapValues(x=>(x,1))
rmap: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[31] at mapValues at <console>:26

scala> rmap.collect.foreach(println)
((Mathew,science),(45,1))
((Mathew,history),(55,1))
((Mark,maths),(23,1))
((Mark,science),(76,1))
((John,history),(14,1))
((John,maths),(74,1))
((Lisa,science),(24,1))
((Lisa,history),(86,1))
((Andrew,maths),(34,1))
((Andrew,science),(26,1))
((Andrew,history),(74,1))
((Mathew,science),(55,1))
((Mathew,history),(87,1))
((Mark,maths),(92,1))
((Mark,science),(12,1))
((John,history),(67,1))
((John,maths),(35,1))
((Lisa,science),(24,1))
((Lisa,history),(98,1))
((Andrew,maths),(23,1))
((Andrew,science),(44,1))
((Andrew,history),(77,1))

scala> val red = rmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
red: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[32] at reduceByKey at <console>:28

scala> red.collect.foreach(println)
((Lisa,history),(184,2))
((Mark,maths),(115,2))
((Andrew,science),(70,2))
```

c.1

```
((Lisa,history),(98,1))
((Andrew,maths),(23,1))
((Andrew,science),(44,1))
((Andrew,history),(77,1))

scala> val red = rmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
red: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[32] at reduceByKey at <console>:28

scala> red.collect.foreach(println)
((Lisa,history),(184,2))
((Mark,maths),(115,2))
((Andrew,science),(70,2))
((Mark,science),(88,2))
((Mathew,science),(100,2))
((Andrew,maths),(57,2))
((Mathew,history),(142,2))
((John,maths),(109,2))
((John,history),(81,2))
((Lisa,science),(48,2))
((Andrew,history),(151,2))

scala> val avg_sc = red.mapValues{case(sum,count)=>(1.0*sum)/count}
avg_sc: org.apache.spark.rdd.RDD[((String, String), Double)] = MapPartitionsRDD[33] at mapValues at <console>:30

scala> avg_sc.collect.foreach(println)
((Lisa,history),92.0)
((Mark,maths),57.5)
((Andrew,science),35.0)
((Mark,science),44.0)
((Mathew,science),50.0)
((Andrew,maths),28.5)
((Mathew,history),71.0)
((John,maths),54.5)
((John,history),40.5)
((Lisa,science),24.0)
((Andrew,history),75.5)

scala>
```

c.2

```
scala> val data = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>((x.split(",")(1),x.split(",")(2)),x.split( ",")(3).toInt
))
data: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[36] at map at <console>:24

scala> val rmap = data.mapValues(x=>(x,1))
rmap: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[37] at mapValues at <console>:26

scala> val red = rmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
red: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[38] at reduceByKey at <console>:28

scala> red.collect.foreach(println)
((history,grade-2),(317,4))
((history,grade-3),(86,1))
((maths,grade-1),(184,4))
((science,grade-3),(115,3))
((science,grade-1),(100,2))
((science,grade-2),(91,3))
((history,grade-1),(155,3))
((maths,grade-2),(97,2))

scala> val Avg_Grade = red.mapValues{case(sum,count)=>(1.0*sum)/count}.foreach(println)
((history,grade-2),79.25)
((history,grade-3),86.0)
((maths,grade-1),46.0)
((science,grade-3),38.333333333333336)
((science,grade-1),50.0)
((science,grade-2),30.333333333333332)
((history,grade-1),51.666666666666664)
((maths,grade-2),48.5)
Avg_Grade: Unit = ()

scala>
```

d.

```
scala> val data = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>((x.split(",")(0),x.split(",")(2)),x.split( ",")(3).toInt
))
data: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[42] at map at <console>:24

scala> val rmap = data.mapValues(x=>(x,1))
rmap: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[43] at mapValues at <console>:26

scala> val red = rmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
red: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[44] at reduceByKey at <console>:28

scala> red.collect.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 ravg = red.mapValues{case(sum,count)=>(1.0*sum)/count}
ravg: org.apache.spark.rdd.RDD[((String, String), Double)] = MapPartitionsRDD[45] at mapValues at <console>:30

scala> ravg.collect.foreach(println)
((Lisa,grade-1),24.0)
((Mark,grade-2),17.5)
((Lisa,grade-2),61.0)
((Mathew,grade-3),45.0)
((Andrew,grade-2),77.0)
((Andrew,grade-1),43.666666666666664)
((Lisa,grade-3),86.0)
((John,grade-1),38.666666666666664)
((John,grade-2),74.0)
((Mark,grade-1),84.0)
```

e.1

```
scala> val ravg = red.mapValues{case(sum,count)=>(1.0*sum)/count}
ravg: org.apache.spark.rdd.RDD[((String, String), Double)] = MapPartitionsRDD[45] at mapValues at <console>:30

scala> ravg.collect.foreach(println)
((Lisa,grade-1),24.0)
((Mark,grade-2),17.5)
((Lisa,grade-2),61.0)
((Mathew,grade-3),45.0)
((Andrew,grade-2),77.0)
((Andrew,grade-1),43.666666666666664)
((Lisa,grade-3),86.0)
((John,grade-1),38.666666666666664)
((John,grade-2),74.0)
((Mark,grade-1),84.0)
((Andrew,grade-3),35.0)
((Mathew,grade-2),65.66666666666667)

scala> val rfiltermap = ravg.filter(x=>x._1._2 == "grade-2" && x._2>50).count().foreach(println)
<console>:32: error: value foreach is not a member of Long
       val rfiltermap = ravg.filter(x=>x._1._2 == "grade-2" && x._2>50).count().foreach(println)
                                                                              ^

scala> val rfiltermap = ravg.filter(x=>x._1._2 == "grade-2" && x._2>50).count()
rfiltermap: Long = 4

scala> rfiltermap.collect.foreach(println)
<console>:35: error: value collect is not a member of Long
       rfiltermap.collect.foreach(println)
                 ^

scala> val rfiltermap = ravg.filter(x=>x._1._2 == "grade-2" && x._2>50).foreach(println)
((Lisa,grade-2),61.0)
((Andrew,grade-2),77.0)
((John,grade-2),74.0)
((Mathew,grade-2),65.66666666666667)
rfiltermap: Unit = ()

scala>
```

e.2

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.

```
val data =
sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>(x.split(",")(0),x.split(",")(3
).toInt))
val stu_avg = data.mapValues(x=>(x,1)).foreach(println)
val studred = stu_avg.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))
val Avg_Stud = studred.mapValues{case (sum,count) => (1.0 * sum)/count}
Avg_Stud.collect.foreach(println)
val flatAvg_St=Avg_Stud.map(x=>x._1+","+x._2)
flatAvg_St.collect.foreach(println)


val data2 =
sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>((x.split(",")(0),x.split(",")(
2)),x.split( ",")(3).toInt)).foreach(println)
val grade = data2.mapValues(x=>(x,1))
val gradeReduce = grade.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))
gradeReduce.collect.foreach(println)
val gradeAvg = gradeReduce.mapValues{case(sum,count) => (1.0*sum)/count}
gradeAvg.collect.foreach(println)
val flatAvg = gradeAvg.map(x=> x._1._1 + "," + x._2.toDouble)
flatAvg.collect.foreach(println)

val common= flatAvg.intersection(flatAvg_Stud)          //intersection to find the
common student
```

Screenshots

```
(Mark,(92,1))
(Mark,(12,1))
(John,(67,1))
(John,(35,1))
(Lisa,(24,1))
(Lisa,(98,1))
(Andrew,(23,1))
(Andrew,(44,1))
(Andrew,(77,1))
stu_avg: Unit = ()

scala> val studred = stu_avg.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))
<console>:28: error: value reduceByKey is not a member of Unit
       val studred = stu_avg.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))
                             ^

scala>

scala> val data = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>(x.split(",")(0),x.split(",")(3).toInt))
data: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[54] at map at <console>:24

scala> val stu_avg = data.mapValues(x=>(x,1))
stu_avg: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[55] at mapValues at <console>:26

scala> val studred = stu_avg.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))
studred: org.apache.spark.rdd.RDD[(String, (Int, Int))] = ShuffledRDD[56] at reduceByKey at <console>:28

scala> val Avg_Stud = studred.mapValues{case (sum,count) => (1.0 * sum)/count}
Avg_Stud: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[57] at mapValues at <console>:30

scala> Avg_Stud.collect.foreach(println)
(Mark,50.75)
(Andrew,46.333333333333336)
(Mathew,60.5)
(John,47.5)
(Lisa,58.0)

scala> val data2 = sc.textFile("file:///home/acadgild/Assign_19.txt").map(x=>((x.split(",")(0),x.split(",")(2)),x.split( ",")(3).toIn
```

Data1

```
data2: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[66] at map at <console>:24

scala> val grade = data2.mapValues(x=>(x,1))
grade: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[67] at mapValues at <console>:26

scala> val gradeReduce = grade.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))
gradeReduce: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[68] at reduceByKey at <console>:28

scala> gradeReduce.collect.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 gradeAvg = gradeReduce.mapValues{case(sum,count) => (1.0*sum)/count}
gradeAvg: org.apache.spark.rdd.RDD[((String, String), Double)] = MapPartitionsRDD[69] at mapValues at <console>:30

scala> gradeAvg.collect.foreach(println)
((Lisa,grade-1),24.0)
((Mark,grade-2),17.5)
((Lisa,grade-2),61.0)
((Mathew,grade-3),45.0)
((Andrew,grade-2),77.0)
((Andrew,grade-1),43.666666666666664)
((Lisa,grade-3),86.0)
((John,grade-1),38.666666666666664)
((John,grade-2),74.0)
((Mark,grade-1),84.0)
((Andrew,grade-3),35.0)
((Mathew,grade-2),65.66666666666667)
```

Data2

```
scala> val flatAvg = gradeAvg.map(x=> x._1._1 + "," + x._2.toDouble)
flatAvg: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[72] at map at <console>:32

scala> flatAvg.collect.foreach(println)
Lisa,24.0
Mark,17.5
Lisa,61.0
Mathew,45.0
Andrew,77.0
Andrew,43.666666666666664
Lisa,86.0
John,38.666666666666664
John,74.0
Mark,84.0
Andrew,35.0
Mathew,65.66666666666667

scala> val flatAvg_St=Avg_Stud.map(x=>x._1+","+x._2)
flatAvg_St: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[73] at map at <console>:32

scala> flatAvg_St.collect.foreach(println)
Mark,50.75
Andrew,46.333333333333336
Mathew,60.5
John,47.5
Lisa,58.0

scala> val common= flatAvg.intersection(flatAvg_St)
common: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[79] at intersection at <console>:44

scala> common.collect.foreach(println)

scala>
```

Intersection