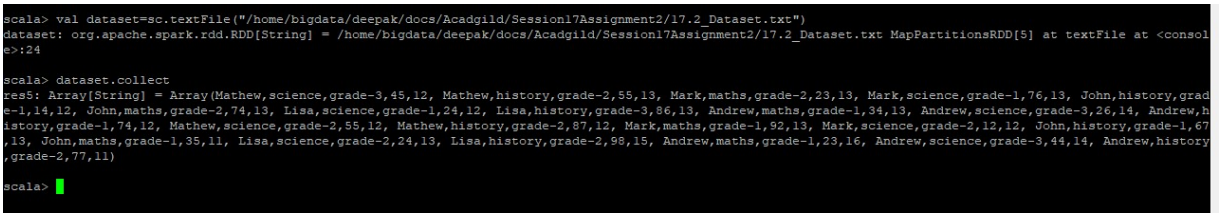
Session 17 Assignment 2

Problem 1

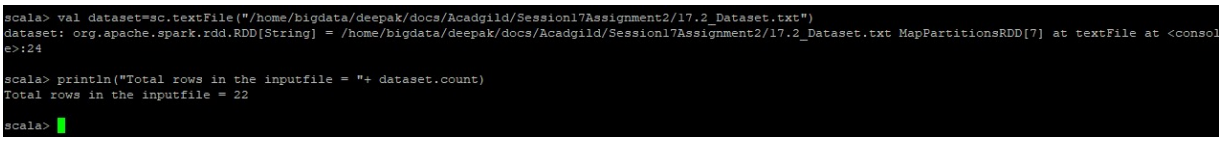
1. Read the file and create and tupled RDD

```
val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
```



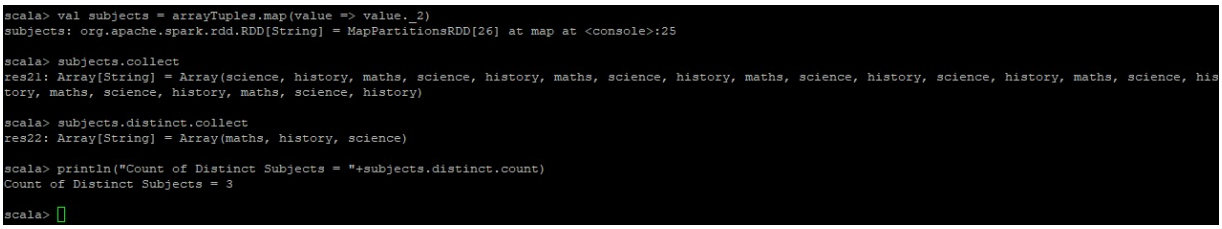
2. Find the total number of rows

```
val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")

println("Total rows in the inputfile = "+ dataset.count)
```



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

```
val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
val subjects = arrayTuples.map(value => value._2)
subjects.collect

println("Count of Distinct Subjects = "+subjects.distinct.count)
```



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

```
val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
val filenameandmarks = arrayTuples.filter(value => value._1=="Mathew" && value._4=="55")
filenameandmarks.collect
println("Total count of students whose name is Mathew and Marks is 55 = "+filenameandmarks.count)
```

```
scala> val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
dataset: org.apache.spark.rdd.RDD[String] = /home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt MapPartitionsRDD[20] at textFile at <conso
le>:24

scala> val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
arrayTuples: org.apache.spark.rdd.RDD[(String, String, String, String, String)] = MapPartitionsRDD[22] at map at <console>:25

scala> val filenameandmarks = arrayTuples.filter(value => value._1=="Mathew" && value._4=="55")
filenameandmarks: org.apache.spark.rdd.RDD[(String, String, String, String, String)] = MapPartitionsRDD[23] at filter at <console>:25

scala> filenameandmarks.collect
res10: Array[(String, String, String, String, String)] = Array((Mathew,history,grade-2,55,13), (Mathew,science,grade-2,55,12))

scala> println("Total count of students whose name is Mathew and Marks is 55 = "+filenameandmarks.count)
Total count of students whose name is Mathew and Marks is 55 = 2

scala>
```

Problem 2

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

```
val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")

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

val groupStudentByGrade=arrayTuples.map(value => (value._3,1))

val countStudentByGrade=groupStudentByGrade.reduceByKey(_+_)

countStudentByGrade.foreach(println)
```

```
scala> val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
dataset: org.apache.spark.rdd.RDD[String] = /home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt MapPartitionsRDD[9] at textFile at <consol
e>:24

scala> val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
arrayTuples: org.apache.spark.rdd.RDD[(String, String, String, String, String)] = MapPartitionsRDD[11] at map at <console>:25

scala> val groupStudentByGrade=arrayTuples.map(value => (value._3,1))
groupStudentByGrade: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[12] at map at <console>:25

scala> val countStudentByGrade=groupStudentByGrade.reduceByKey(_+_)
countStudentByGrade: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[13] at reduceByKey at <console>:25

scala> countStudentByGrade.foreach(println)
(grade-3,4)
(grade-1,9)
(grade-2,9)

scala>
```

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

```
val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")

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

val groupDatasetByGradeAndName=arrayTuples.map(x => (x._3+"-"+x._1,(x._4.toInt,1)))

val getAveragePerStudentPerGrade=groupDatasetByGradeAndName.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2._2.toInt))

getAveragePerStudentPerGrade.foreach(println)
```

```
scala> val groupDatasetByGradeAndName=arrayTuples.map(x => (x._3+"-"+x._1,(x._4.toInt,1)))
groupDatasetByGradeAndName: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[26] at map at <console>:25

scala> val getAveragePerStudentPerGrade=groupDatasetByGradeAndName.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2
._2.toInt))
getAveragePerStudentPerGrade: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[28] at map at <console>:25

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

scala>
```

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

```
val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
val groupDatasetBySubjectAndName=arrayTuples.map(x => (x._2,(x._4.toInt,1)))
val getAveragePerPerSubject=groupDatasetBySubjectAndName.reduceByKey{(x, y) => (x._1 + y._1, x._2
+y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2._2.toInt))
getAveragePerStudentPerSubject.foreach(println)
```

```
scala> val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
dataset: org.apache.spark.rdd.RDD[String] = /home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt MapPartitionsRDD[40] at textFile at <conso
le>:24

scala> val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
arrayTuples: org.apache.spark.rdd.RDD[(String, String, String, String, String)] = MapPartitionsRDD[42] at map at <console>:25

scala> val groupDatasetBySubjectAndName=arrayTuples.map(x => (x._2,(x._4.toInt,1)))
groupDatasetBySubjectAndName: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[43] at map at <console>:25

scala> groupDatasetBySubjectAndName.collect
res18: Array[(String, (Int, Int))] = Array((science,(45,1)), (history,(55,1)), (maths,(23,1)), (science,(76,1)), (history,(14,1)), (maths,(74,1)), (science,(24
,1)), (history,(86,1)), (maths,(34,1)), (science,(26,1)), (history,(74,1)), (science,(55,1)), (history,(87,1)), (maths,(92,1)), (science,(12,1)), (history,(67,
1)), (maths,(35,1)), (science,(24,1)), (history,(98,1)), (maths,(23,1)), (science,(44,1)), (history,(77,1)))

scala> val getAveragePerPerSubject=groupDatasetBySubjectAndName.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2._2
.toInt))
getAveragePerPerSubject: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[45] at map at <console>:25

scala> getAveragePerPerSubject.collect
res19: Array[(String, Double)] = Array((maths,46.833333333333336), (history,69.75), (science,38.25))

scala> getAveragePerPerSubject.foreach(println)
(maths,46.833333333333336)
(history,69.75)
(science,38.25)
```

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

```
val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
val groupDatasetBySubjectAndGrade=arrayTuples.map(x => (x._2+"-"+x._3,(x._4.toInt,1)))
val getAveragePerSubjectGrade=groupDatasetBySubjectAndGrade.reduceByKey{(x, y) => (x._1 + y._1, x._2
+y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2._2.toInt))
getAveragePerSubjectGrade.foreach(println)
```

```
scala> val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
dataset: org.apache.spark.rdd.RDD[String] = /home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt MapPartitionsRDD[47] at textFile at <conso
le>:24

scala> val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
arrayTuples: org.apache.spark.rdd.RDD[(String, String, String, String, String)] = MapPartitionsRDD[49] at map at <console>:25

scala> val groupDatasetBySubjectAndGrade=arrayTuples.map(x => (x._2+"-"+x._3,(x._4.toInt,1)))
groupDatasetBySubjectAndGrade: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[50] at map at <console>:25

scala> val getAveragePerSubjectGrade=groupDatasetBySubjectAndGrade.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2
._2.toInt))
getAveragePerSubjectGrade: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[52] at map at <console>:25

scala> getAveragePerSubjectGrade.foreach(println)
(maths-grade-2,48.5)
(science-grade-1,50.0)
(history-grade-2,79.25)
(science-grade-2,30.333333333333332)
(science-grade-3,38.333333333333336)
(maths-grade-1,46.0)
(history-grade-3,86.0)
(history-grade-1,51.666666666666664)

scala>
```

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

```
val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")

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

val filteredData=arrayTuples.filter(values => values._3=="grade-2")

val groupDatasetByGradeAndName= filteredData.map(x => (x._3+"-"+x._1,(x._4.toInt,1)))

val getAveragePerStudentPerGrade=groupDatasetByGradeAndName.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv =>
(kv._1, kv._2._1.toDouble / kv._2._2.toInt))

getAveragePerStudentPerGrade.foreach(println)

val finalFiltereddata=getAveragePerStudentPerGrade.filter(values => values._2>50)

println("Count of students having average greater than 50 in grade-2 = "+finalFiltereddata.count)
```

```
scala> val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
dataset: org.apache.spark.rdd.RDD[String] = /home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt MapPartitionsRDD[83] at textFile at <conso
le>:24

scala> val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
arrayTuples: org.apache.spark.rdd.RDD[(String, String, String, String, String)] = MapPartitionsRDD[85] at map at <console>:25

scala> val filteredData=arrayTuples.filter(values => values._3=="grade-2")
filteredData: org.apache.spark.rdd.RDD[(String, String, String, String, String)] = MapPartitionsRDD[86] at filter at <console>:25

scala> val groupDatasetByGradeAndName=filteredData.map(x => (x._3+"-"+x._1,(x._4.toInt,1)))
groupDatasetByGradeAndName: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[87] at map at <console>:25

scala> val getAveragePerStudentPerGrade=groupDatasetByGradeAndName.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2
._2.toInt))
getAveragePerStudentPerGrade: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[89] at map at <console>:25

scala> getAveragePerStudentPerGrade.foreach(println)
(grade-2-Mathew,65.66666666666667)
(grade-2-Lisa,61.0)
(grade-2-John,74.0)
(grade-2-Mark,17.5)
(grade-2-Andrew,77.0)

scala> val finalFiltereddata=getAveragePerStudentPerGrade.filter(values => values._2>50)
finalFiltereddata: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[90] at filter at <console>:25

scala> finalFiltereddata.foreach(println)
(grade-2-Mathew,65.66666666666667)
(grade-2-Lisa,61.0)
(grade-2-John,74.0)
(grade-2-Andrew,77.0)

scala> println("Count of students having average greater than 50 in grade-2 = "+finalFiltereddata.count)
Count of students having average greater than 50 in grade-2 = 4

scala>
```

Problem Statement 3

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

student_name per grade

val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")

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

val groupDatasetByName=arrayTuples.map(x => (x._1,(x._4.toInt,1)))

val getAverage=groupDatasetByName.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2._2.toInt))

getAverage.foreach(println)

```
scala> val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
dataset: org.apache.spark.rdd.RDD[String] = /home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt MapPartitionsRDD[32] at textFile at <conso
le>:24

scala> val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
arrayTuples: org.apache.spark.rdd.RDD[(String, String, String, String, String)] = MapPartitionsRDD[34] at map at <console>:25

scala> val groupDatasetByName=arrayTuples.map(x => (x._1,(x._4.toInt,1)))
groupDatasetByName: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[35] at map at <console>:25

scala> val getAverage=groupDatasetByName.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2._2.toInt))
getAverage: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[37] at map at <console>:25

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

val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")

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

val groupDatasetByNameAndGrade=arrayTuples.map(x => (x._1+"-"+x._3,(x._4.toInt,1)))

val getAveragePerStudentPerGrade=groupDatasetByNameAndGrade.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2._2.toInt))

getAveragePerStudentPerGrade.foreach(println)

```
scala> val dataset=sc.textFile("/home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt")
dataset: org.apache.spark.rdd.RDD[String] = /home/bigdata/deepak/docs/Acadgild/Session17Assignment2/17.2_Dataset.txt MapPartitionsRDD[17] at textFile at <conso
le>:24

scala> val arrayTuples = dataset.map(line => line.split(",")).map(array => (array(0),array(1),array(2),array(3),array(4)))
arrayTuples: org.apache.spark.rdd.RDD[(String, String, String, String, String)] = MapPartitionsRDD[19] at map at <console>:25

scala> val groupDatasetByNameAndGrade=arrayTuples.map(x => (x._1+"-"+x._3,(x._4.toInt,1)))
groupDatasetByNameAndGrade: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[20] at map at <console>:25

scala> val getAveragePerStudentPerGrade=groupDatasetByNameAndGrade.reduceByKey{(x, y) => (x._1 + y._1, x._2 +y._2)}.map(kv => (kv._1, kv._2._1.toDouble / kv._2
._2.toInt))
getAveragePerStudentPerGrade: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[22] at map at <console>:25

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

scala>
```

getAveragePerStudentPerGrade.intersection(getAverage)

getAveragePerStudentPerGrade.intersection(getAverage).foreach(println)

```
scala> getAveragePerStudentPerGrade.intersection(getAverage)
res6: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[43] at intersection at <console>:28

scala> getAveragePerStudentPerGrade.intersection(getAverage).foreach(println)

scala>
```