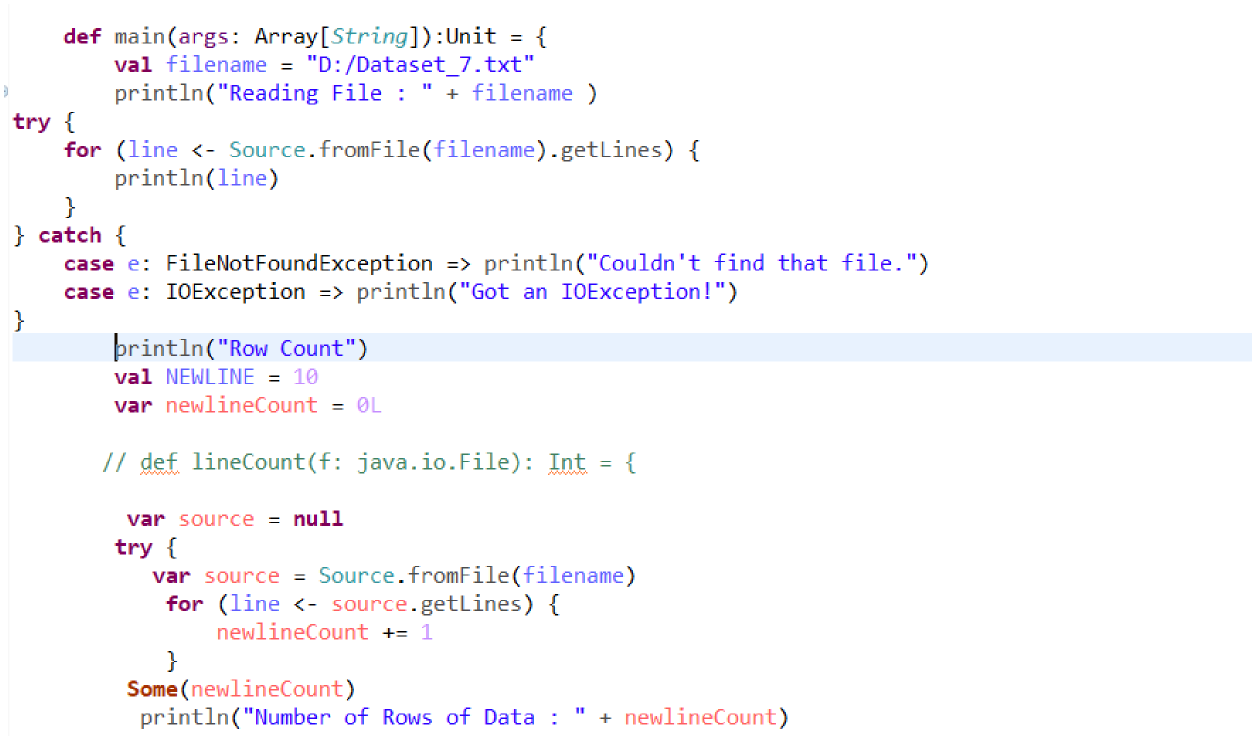
Session 7: RDD DEEP DIVE SPARK SQL 1 SPARK SQL 2 Assignment 1

Task 1

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

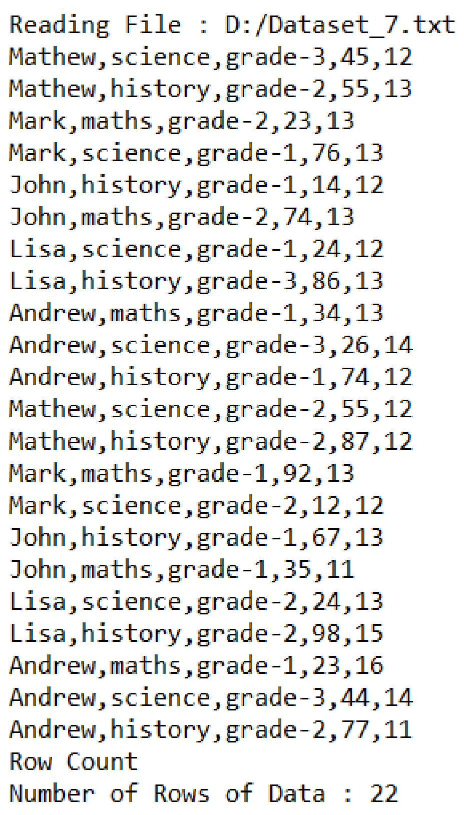
Comments: The below code reads the data from the file and counts the number of rows of data in document.

Code



```
    def main(args: Array[String]):Unit = {
        val filename = "D:/Dataset_7.txt"
        println("Reading File : " + filename )
try {
    for (line <- Source.fromFile(filename).getLines) {
        println(line)
    }
} catch {
    case e: FileNotFoundException => println("Couldn't find that file.")
    case e: IOException => println("Got an IOException!")
}
        println("Row Count")
        val NEWLINE = 10
        var newlineCount = 0L

       // def lineCount(f: java.io.File): Int = {

         var source = null
        try {
           var source = Source.fromFile(filename)
            for (line <- source.getLines) {
                newlineCount += 1
            }
         Some(newlineCount)
          println("Number of Rows of Data : " + newlineCount)
```

Output

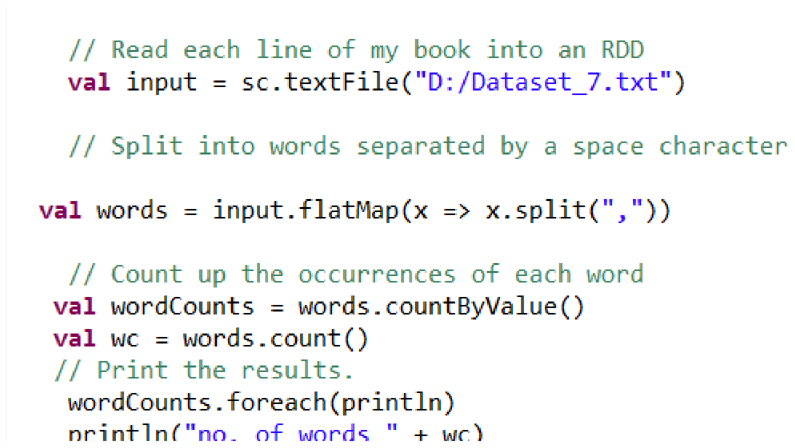


```
Reading File : D:/Dataset_7.txt
Mathew,science,grade-3,45,12
Mathew,history,grade-2,55,13
Mark,maths,grade-2,23,13
Mark,science,grade-1,76,13
John,history,grade-1,14,12
John,maths,grade-2,74,13
Lisa,science,grade-1,24,12
Lisa,history,grade-3,86,13
Andrew,maths,grade-1,34,13
Andrew,science,grade-3,26,14
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
Row Count
Number of Rows of Data : 22
```

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

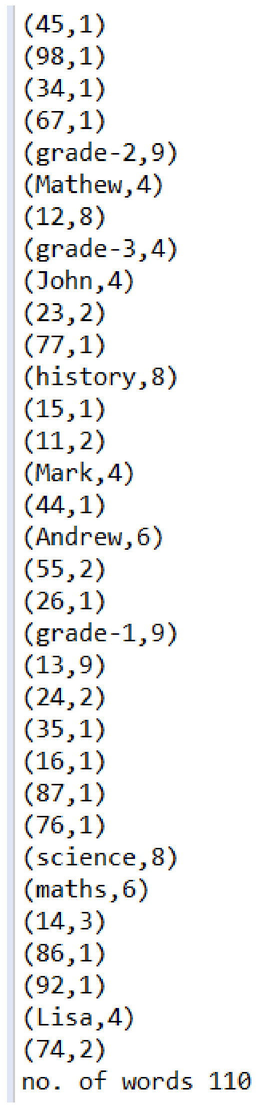
Comments: The below code counts the occurrence of each word and also counts the total number of words present in the document

Code



```
    // Read each line of my book into an RDD
    val input = sc.textFile("D:/Dataset_7.txt")

    // Split into words separated by a space character

  val words = input.flatMap(x => x.split(","))

    // Count up the occurrences of each word
   val wordCounts = words.countByValue()
   val wc = words.count()
   // Print the results.
    wordCounts.foreach(println)
    println("no. of words " + wc)
```

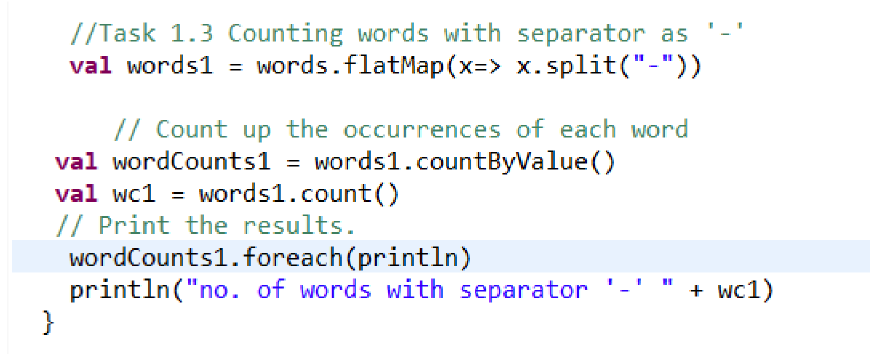
Output



```
(45,1)
(98,1)
(34,1)
(67,1)
(grade-2,9)
(Mathew,4)
(12,8)
(grade-3,4)
(John,4)
(23,2)
(77,1)
(history,8)
(15,1)
(11,2)
(Mark,4)
(44,1)
(Andrew,6)
(55,2)
(26,1)
(grade-1,9)
(13,9)
(24,2)
(35,1)
(16,1)
(87,1)
(76,1)
(science,8)
(maths,6)
(14,3)
(86,1)
(92,1)
(Lisa,4)
(74,2)
no. of words 110
```

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.

Code



```
    //Task 1.3 Counting words with separator as '-'
    val words1 = words.flatMap(x=> x.split("-"))

       // Count up the occurrences of each word
   val wordCounts1 = words1.countByValue()
   val wc1 = words1.count()
   // Print the results.
    wordCounts1.foreach(println)
    println("no. of words with separator '-' " + wc1)
  }
```

Output

```
(45,1)
(98,1)
(34,1)
(67,1)
(Mathew,4)
(12,8)
(John,4)
(23,2)
(77,1)
(history,8)
(15,1)
(11,2)
(Mark,4)
(44,1)
(Andrew,6)
(55,2)
(26,1)
(13,9)
(24,2)
(35,1)
(16,1)
(87,1)
(76,1)
(science,8)
(maths,6)
(1,9)
(14,3)
(grade,22)
(2,9)
(86,1)
(92,1)
(Lisa,4)
(3,4)
(74,2)
no. of words with separator '-' 132
```

Task 2

Problem Statement 1:

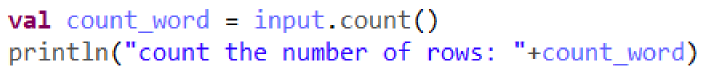
1. Read the text file, and create a tupled rdd.

Code

```
    // Create a SparkContext using every core of the local machine
   val sc = new SparkContext("local[*]", "RDD")

   // Read each line of my book into an RDD
   val input = sc.textFile("D:/Dataset_7.txt")

   // Split into words separated by a space character
    val words = input.map(x => {
     val row = x.split(",").toList
      (row.apply(0).toString, row.apply(1).toString, row.apply(2).toString , row.apply(3).toInt, row.apply(4).toInt)
           })
    val readRDD = input.collect().foreach(println)
```

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

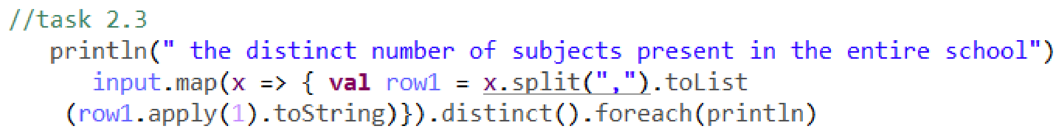
<u>Code</u>



```
val count_word = input.count()
println("count the number of rows: "+count_word)
```

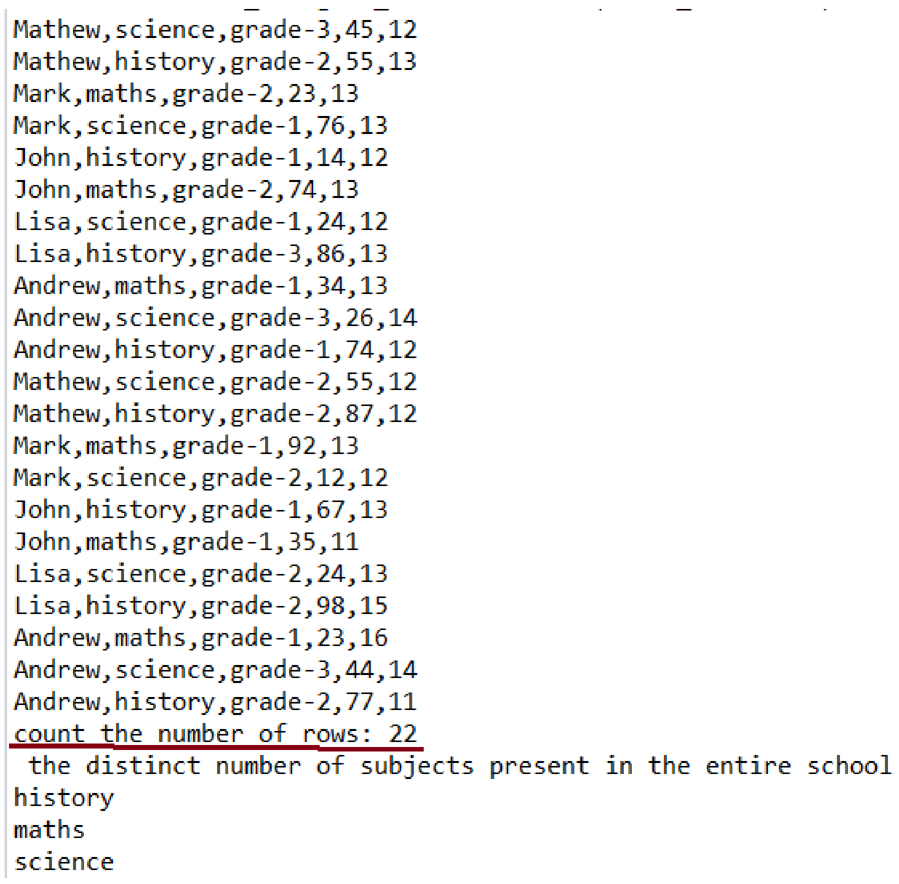
3. What is the distinct number of subjects present in the entire school

<u>Code</u>



```
//task 2.3
   println(" the distinct number of subjects present in the entire school")
      input.map(x => { val row1 = x.split(",").toList
    (row1.apply(1).toString)}).distinct().foreach(println)
```

<u>Output</u>



```
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
count the number of rows: 22
 the distinct number of subjects present in the entire school
history
maths
science
```

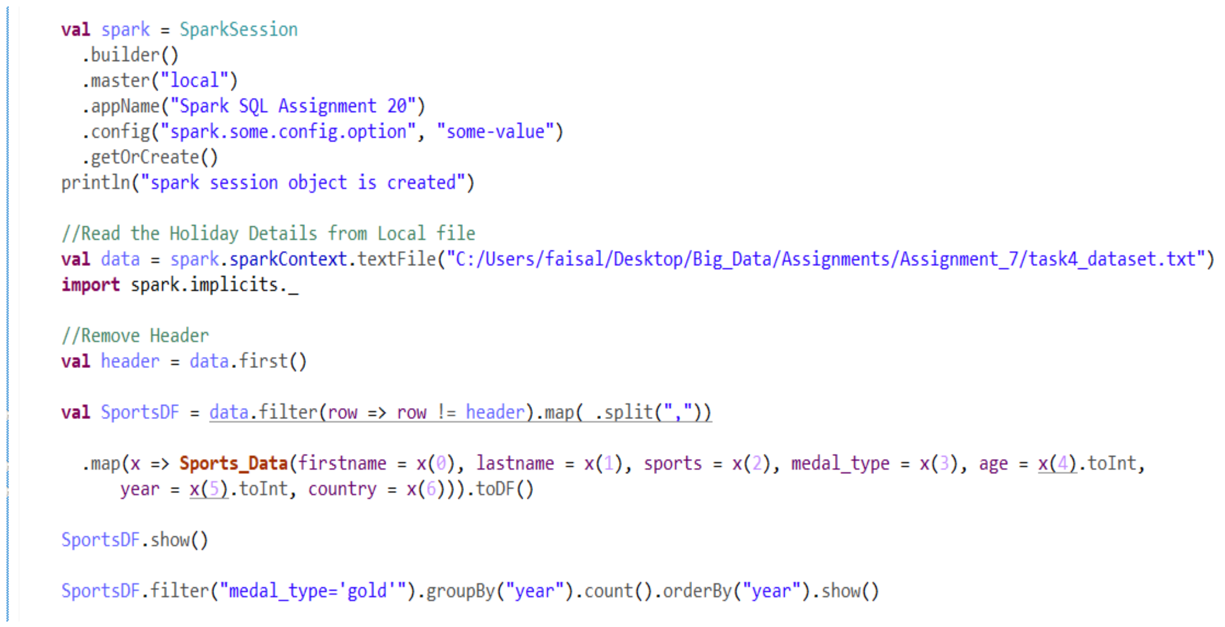
Task 4

1. What are the total number of gold medal winners every year

Comments:

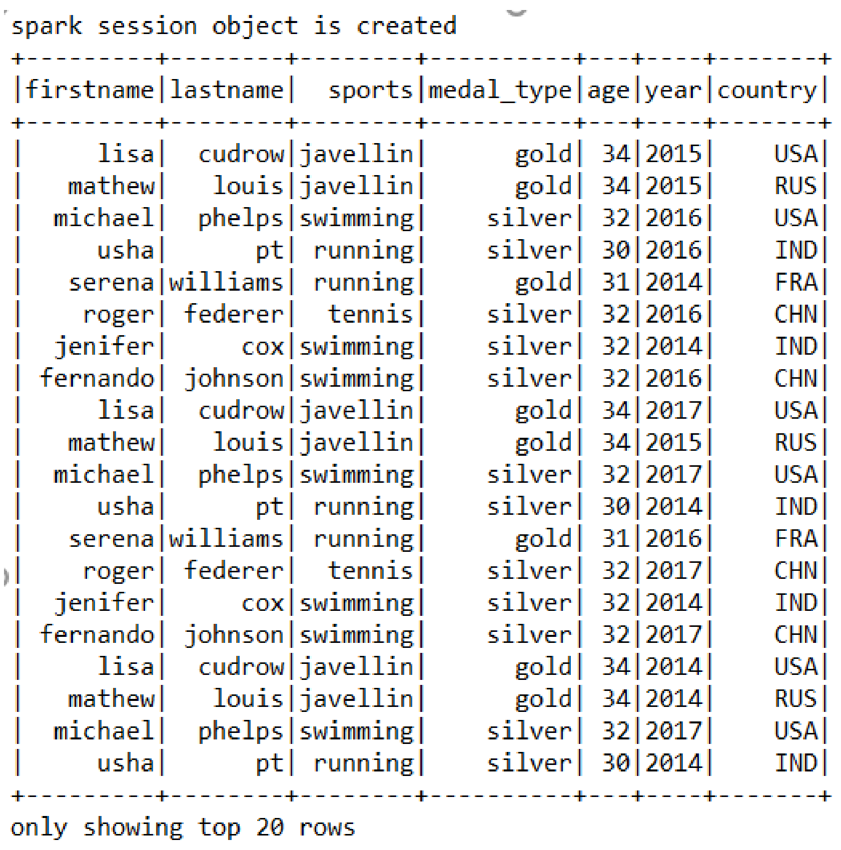
- **To load the data from these local file to Dataframe in Spark SQL.**
- **Load data from above created RDD in dataframe**

CODE

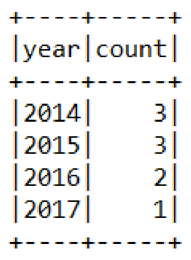


```
val spark = SparkSession
  .builder()
  .master("local")
  .appName("Spark SQL Assignment 20")
  .config("spark.some.config.option", "some-value")
  .getOrCreate()
println("spark session object is created")

//Read the Holiday Details from Local file
val data = spark.sparkContext.textFile("C:/Users/faisal/Desktop/Big_Data/Assignments/Assignment_7/task4_dataset.txt")
import spark.implicits._

//Remove Header
val header = data.first()

val SportsDF = data.filter(row => row != header).map(_.split(","))

  .map(x => Sports_Data(firstname = x(0), lastname = x(1), sports = x(2), medal_type = x(3), age = x(4).toInt,
      year = x(5).toInt, country = x(6))).toDF()

SportsDF.show()

SportsDF.filter("medal_type='gold'").groupBy("year").count().orderBy("year").show()
```

Output



```
spark session object is created
+---------+--------+--------+----------+---+----+-------+
|firstname|lastname|  sports|medal_type|age|year|country|
+---------+--------+--------+----------+---+----+-------+
|     lisa|  cudrow|javellin|      gold| 34|2015|    USA|
|   mathew|   louis|javellin|      gold| 34|2015|    RUS|
|  michael|  phelps|swimming|    silver| 32|2016|    USA|
|     usha|      pt| running|    silver| 30|2016|    IND|
|   serena|williams| running|      gold| 31|2014|    FRA|
|    roger| federer|  tennis|    silver| 32|2016|    CHN|
|  jenifer|     cox|swimming|    silver| 32|2014|    IND|
| fernando| johnson|swimming|    silver| 32|2016|    CHN|
|     lisa|  cudrow|javellin|      gold| 34|2017|    USA|
|   mathew|   louis|javellin|      gold| 34|2015|    RUS|
|  michael|  phelps|swimming|    silver| 32|2017|    USA|
|     usha|      pt| running|    silver| 30|2014|    IND|
|   serena|williams| running|      gold| 31|2016|    FRA|
|    roger| federer|  tennis|    silver| 32|2017|    CHN|
|  jenifer|     cox|swimming|    silver| 32|2014|    IND|
| fernando| johnson|swimming|    silver| 32|2017|    CHN|
|     lisa|  cudrow|javellin|      gold| 34|2014|    USA|
|   mathew|   louis|javellin|      gold| 34|2014|    RUS|
|  michael|  phelps|swimming|    silver| 32|2017|    USA|
|     usha|      pt| running|    silver| 30|2014|    IND|
+---------+--------+--------+----------+---+----+-------+
only showing top 20 rows
```

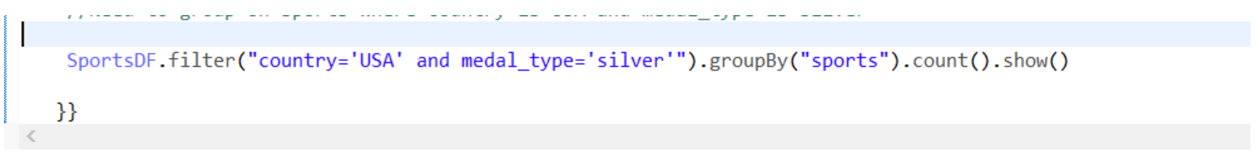


```
+----+-----+
|year|count|
+----+-----+
|2014|    3|
|2015|    3|
|2016|    2|
|2017|    1|
+----+-----+
```

2. How many silver medals have been won by USA in each sport

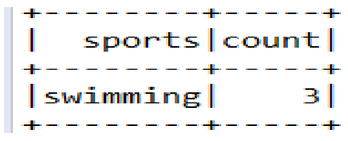
Comments: Need to group on sports where country is USA and medal_type is silver

CODE



```
SportsDF.filter("country='USA' and medal_type='silver'").groupBy("sports").count().show()

}}
```

Output



```
+--------+-----+
|  sports|count|
+--------+-----+
|swimming|    3|
+--------+-----+
```

Task 5 :

Task 5.1 : Using udfs on dataframe

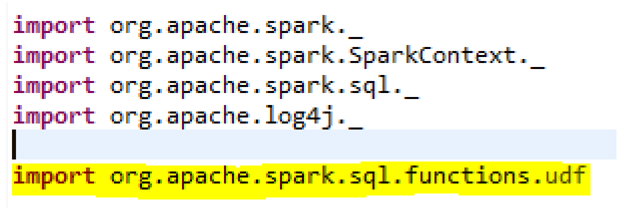
1. Change firstname, lastname columns into Mr.first_two_letters_of_firstname<space>lastname for example - michael, phelps becomes Mr.mi phelps

UDFs in Spark SQL:

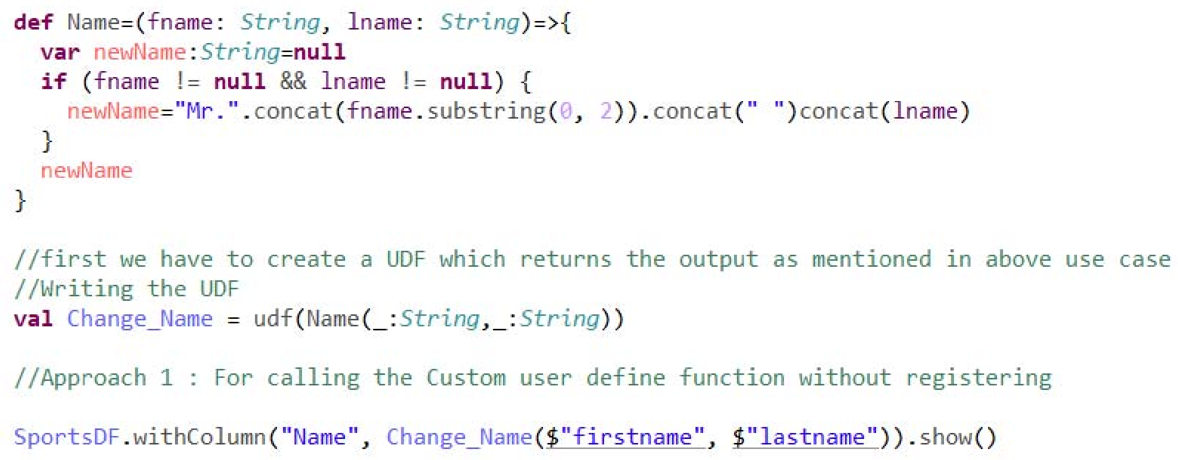
User-Defined Functions (aka UDF) is a feature of Spark SQL to define new Column-based functions that extend the vocabulary of Spark SQL's DSL for transforming Datasets.

- Import namespace 'org.apache.spark.sql.functions.udf' to extend the functionality / write the udfs.
- Define a basic function scala to perform the above task of changing the columns

Code

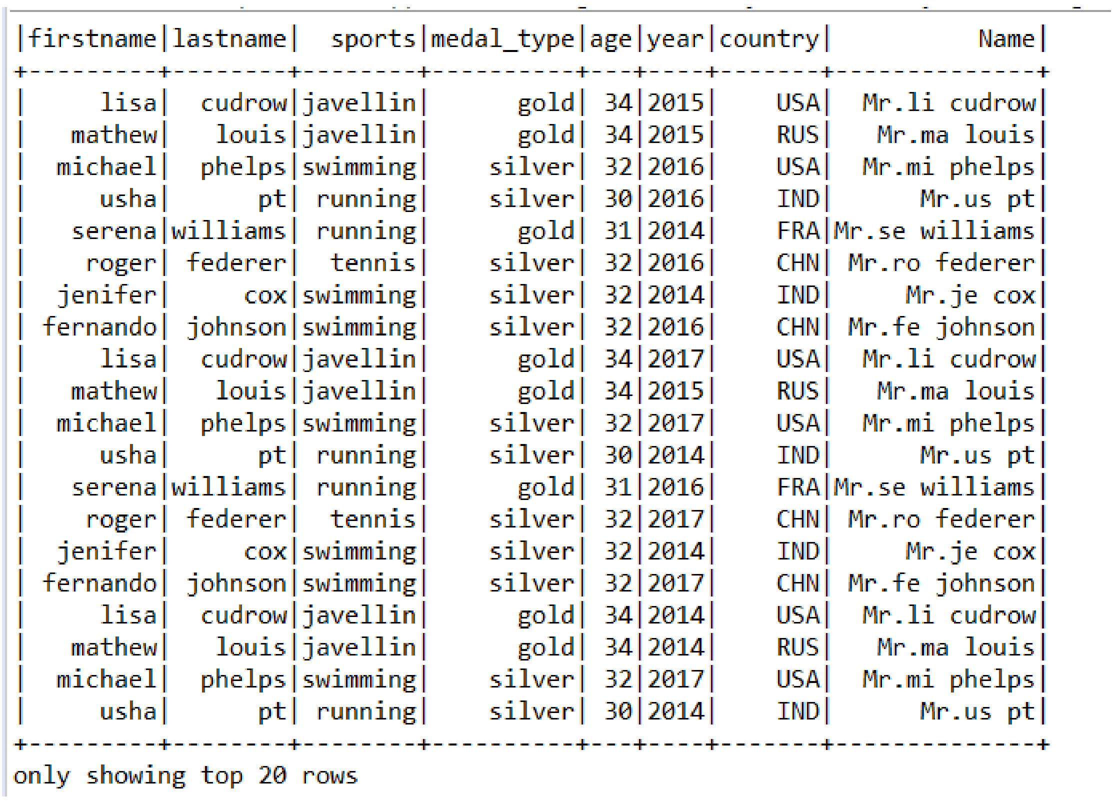


```scala
import org.apache.spark._
import org.apache.spark.SparkContext._
import org.apache.spark.sql._
import org.apache.log4j._

import org.apache.spark.sql.functions.udf
```



```scala
def Name=(fname: String, lname: String)=>{
  var newName:String=null
  if (fname != null && lname != null) {
    newName="Mr.".concat(fname.substring(0, 2)).concat(" ")concat(lname)
  }
  newName
}

//first we have to create a UDF which returns the output as mentioned in above use case
//Writing the UDF
val Change_Name = udf(Name(_:String,_:String))

//Approach 1 : For calling the Custom user define function without registering

SportsDF.withColumn("Name", Change_Name($"firstname", $"lastname")).show()
```

Output



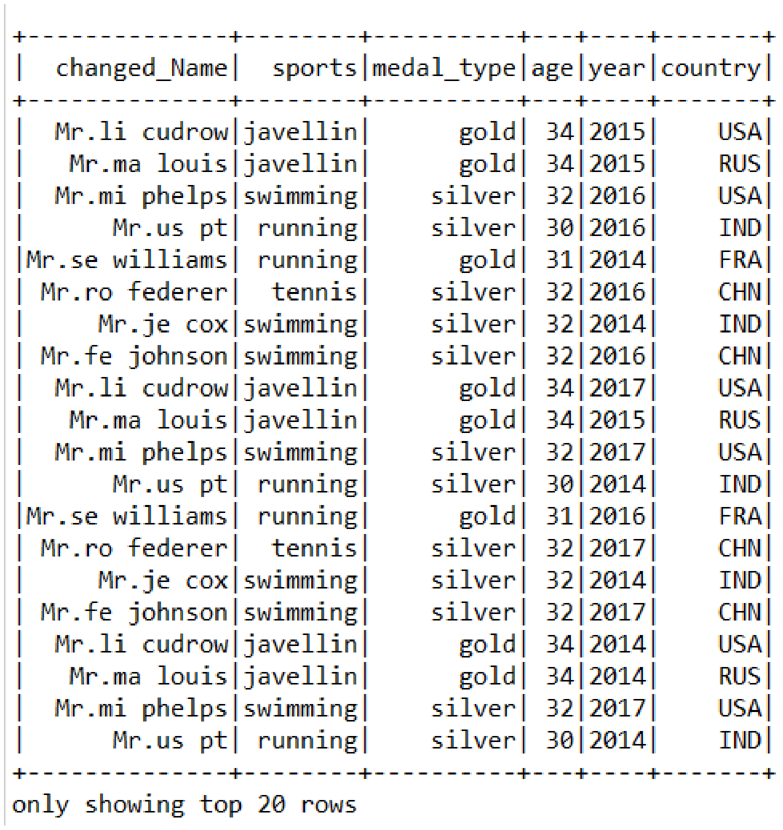
```
|firstname|lastname|  sports|medal_type|age|year|country|          Name|
+---------+--------+--------+----------+---+----+-------+--------------+
|     lisa|  cudrow|javellin|      gold| 34|2015|    USA|  Mr.li cudrow|
|   mathew|   louis|javellin|      gold| 34|2015|    RUS|   Mr.ma louis|
|  michael|  phelps|swimming|    silver| 32|2016|    USA|  Mr.mi phelps|
|     usha|      pt| running|    silver| 30|2016|    IND|      Mr.us pt|
|   serena|williams| running|      gold| 31|2014|    FRA|Mr.se williams|
|    roger| federer|  tennis|    silver| 32|2016|    CHN| Mr.ro federer|
|  jenifer|     cox|swimming|    silver| 32|2014|    IND|     Mr.je cox|
| fernando| johnson|swimming|    silver| 32|2016|    CHN| Mr.fe johnson|
|     lisa|  cudrow|javellin|      gold| 34|2017|    USA|  Mr.li cudrow|
|   mathew|   louis|javellin|      gold| 34|2015|    RUS|   Mr.ma louis|
|  michael|  phelps|swimming|    silver| 32|2017|    USA|  Mr.mi phelps|
|     usha|      pt| running|    silver| 30|2014|    IND|      Mr.us pt|
|   serena|williams| running|      gold| 31|2016|    FRA|Mr.se williams|
|    roger| federer|  tennis|    silver| 32|2017|    CHN| Mr.ro federer|
|  jenifer|     cox|swimming|    silver| 32|2014|    IND|     Mr.je cox|
| fernando| johnson|swimming|    silver| 32|2017|    CHN| Mr.fe johnson|
|     lisa|  cudrow|javellin|      gold| 34|2014|    USA|  Mr.li cudrow|
|   mathew|   louis|javellin|      gold| 34|2014|    RUS|   Mr.ma louis|
|  michael|  phelps|swimming|    silver| 32|2017|    USA|  Mr.mi phelps|
|     usha|      pt| running|    silver| 30|2014|    IND|      Mr.us pt|
+---------+--------+--------+----------+---+----+-------+--------------+
only showing top 20 rows
```

By registering the udf so that it can be used wih sql queries

<u>Code</u>

```
//Approach 2: By registering the function

spark.sqlContext.udf.register("Name", Name)


spark.sql("Select Name(firstname,lastname) as changed_Name, sports,medal_type,age,year,country from Sports_Table").show()
```

<u>Output</u>



```
+--------------+--------+----------+---+----+-------+
|  changed_Name|  sports|medal_type|age|year|country|
+--------------+--------+----------+---+----+-------+
|  Mr.li cudrow|javellin|      gold| 34|2015|    USA|
|   Mr.ma louis|javellin|      gold| 34|2015|    RUS|
|  Mr.mi phelps|swimming|    silver| 32|2016|    USA|
|      Mr.us pt| running|    silver| 30|2016|    IND|
|Mr.se williams| running|      gold| 31|2014|    FRA|
| Mr.ro federer|  tennis|    silver| 32|2016|    CHN|
|     Mr.je cox|swimming|    silver| 32|2014|    IND|
| Mr.fe johnson|swimming|    silver| 32|2016|    CHN|
|  Mr.li cudrow|javellin|      gold| 34|2017|    USA|
|   Mr.ma louis|javellin|      gold| 34|2015|    RUS|
|  Mr.mi phelps|swimming|    silver| 32|2017|    USA|
|      Mr.us pt| running|    silver| 30|2014|    IND|
|Mr.se williams| running|      gold| 31|2016|    FRA|
| Mr.ro federer|  tennis|    silver| 32|2017|    CHN|
|     Mr.je cox|swimming|    silver| 32|2014|    IND|
| Mr.fe johnson|swimming|    silver| 32|2017|    CHN|
|  Mr.li cudrow|javellin|      gold| 34|2014|    USA|
|   Mr.ma louis|javellin|      gold| 34|2014|    RUS|
|  Mr.mi phelps|swimming|    silver| 32|2017|    USA|
|      Mr.us pt| running|    silver| 30|2014|    IND|
+--------------+--------+----------+---+----+-------+
only showing top 20 rows
```

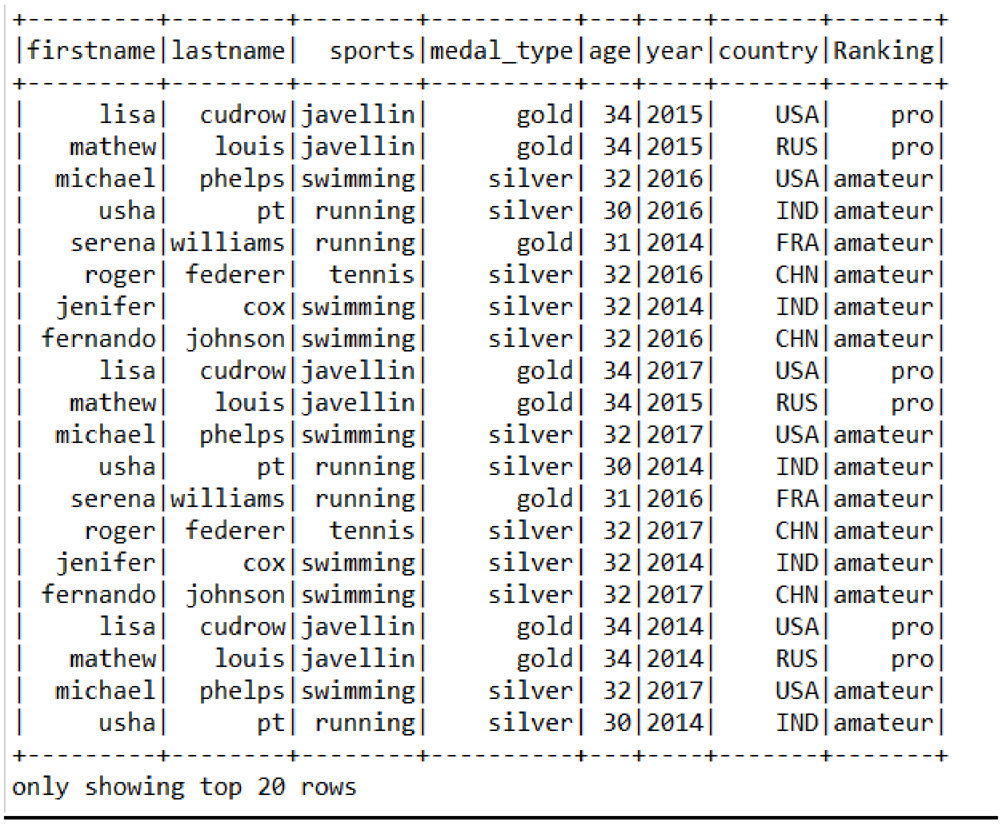
Task 5.2 Using udfs on dataframe

Add a new column called ranking using udfs on dataframe,
where : gold medalist, with age >= 32 are ranked as pro
gold medalists with age <= 31 are ranked amateur
silver medalist, with age >= 32 are ranked as expert
silver medalists, with age <= 31 are ranked rookie

Code

```
val Rankings = udf(ranking_recived(_:String,_:Int))


//Approach 1: Without Registering the UDF and calling with Spark SQL Operatios

SportsDF.withColumn("Ranking",Rankings($"medal type",$"age")).show()
```

Output



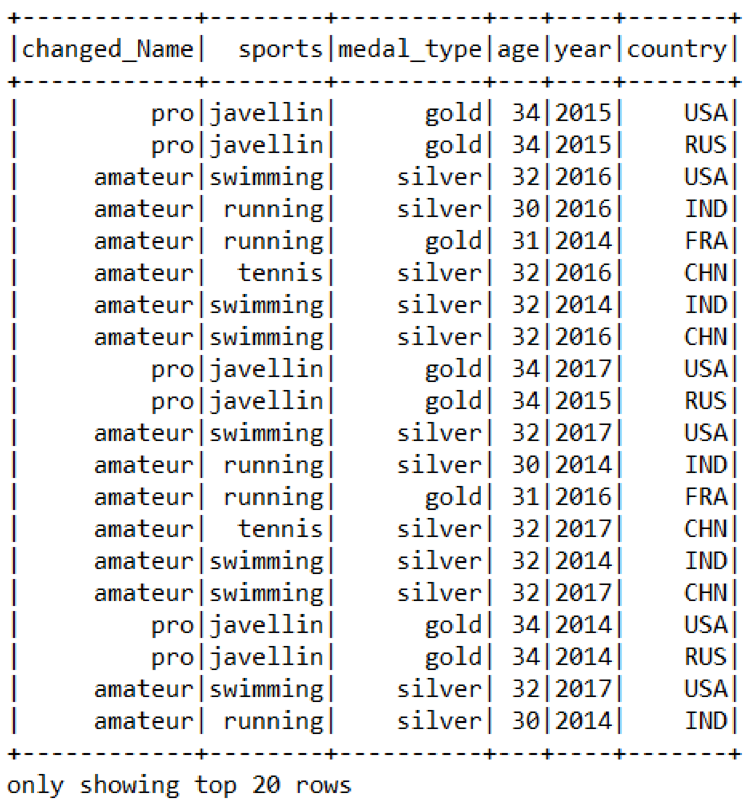
```
+---------+--------+--------+----------+---+----+-------+--------+
|firstname|lastname|  sports|medal_type|age|year|country| Ranking|
+---------+--------+--------+----------+---+----+-------+--------+
|     lisa|  cudrow|javellin|      gold| 34|2015|    USA|     pro|
|   mathew|   louis|javellin|      gold| 34|2015|    RUS|     pro|
|  michael|  phelps|swimming|    silver| 32|2016|    USA| amateur|
|     usha|      pt| running|    silver| 30|2016|    IND| amateur|
|   serena|williams| running|      gold| 31|2014|    FRA| amateur|
|    roger| federer|  tennis|    silver| 32|2016|    CHN| amateur|
|  jenifer|     cox|swimming|    silver| 32|2014|    IND| amateur|
| fernando| johnson|swimming|    silver| 32|2016|    CHN| amateur|
|     lisa|  cudrow|javellin|      gold| 34|2017|    USA|     pro|
|   mathew|   louis|javellin|      gold| 34|2015|    RUS|     pro|
|  michael|  phelps|swimming|    silver| 32|2017|    USA| amateur|
|     usha|      pt| running|    silver| 30|2014|    IND| amateur|
|   serena|williams| running|      gold| 31|2016|    FRA| amateur|
|    roger| federer|  tennis|    silver| 32|2017|    CHN| amateur|
|  jenifer|     cox|swimming|    silver| 32|2014|    IND| amateur|
| fernando| johnson|swimming|    silver| 32|2017|    CHN| amateur|
|     lisa|  cudrow|javellin|      gold| 34|2014|    USA|     pro|
|   mathew|   louis|javellin|      gold| 34|2014|    RUS|     pro|
|  michael|  phelps|swimming|    silver| 32|2017|    USA| amateur|
|     usha|      pt| running|    silver| 30|2014|    IND| amateur|
+---------+--------+--------+----------+---+----+-------+--------+
only showing top 20 rows
```

By registering the udf so that it can be used wih sql queries

Code

```
//Approach 2:By Registering the function

spark.sqlContext.udf.register("Rankings",ranking_recived)

spark.sql("Select Rankings(medal_type,age) as changed_Name, sports,medal_type,age,year,country from Sports_Table").show()
```

Output



```
+------------+--------+----------+---+----+-------+
|changed_Name|  sports|medal_type|age|year|country|
+------------+--------+----------+---+----+-------+
|         pro|javellin|      gold| 34|2015|    USA|
|         pro|javellin|      gold| 34|2015|    RUS|
|     amateur|swimming|    silver| 32|2016|    USA|
|     amateur| running|    silver| 30|2016|    IND|
|     amateur| running|      gold| 31|2014|    FRA|
|     amateur|  tennis|    silver| 32|2016|    CHN|
|     amateur|swimming|    silver| 32|2014|    IND|
|     amateur|swimming|    silver| 32|2016|    CHN|
|         pro|javellin|      gold| 34|2017|    USA|
|         pro|javellin|      gold| 34|2015|    RUS|
|     amateur|swimming|    silver| 32|2017|    USA|
|     amateur| running|    silver| 30|2014|    IND|
|     amateur| running|      gold| 31|2016|    FRA|
|     amateur|  tennis|    silver| 32|2017|    CHN|
|     amateur|swimming|    silver| 32|2014|    IND|
|     amateur|swimming|    silver| 32|2017|    CHN|
|         pro|javellin|      gold| 34|2014|    USA|
|         pro|javellin|      gold| 34|2014|    RUS|
|     amateur|swimming|    silver| 32|2017|    USA|
|     amateur| running|    silver| 30|2014|    IND|
+------------+--------+----------+---+----+-------+
only showing top 20 rows
```