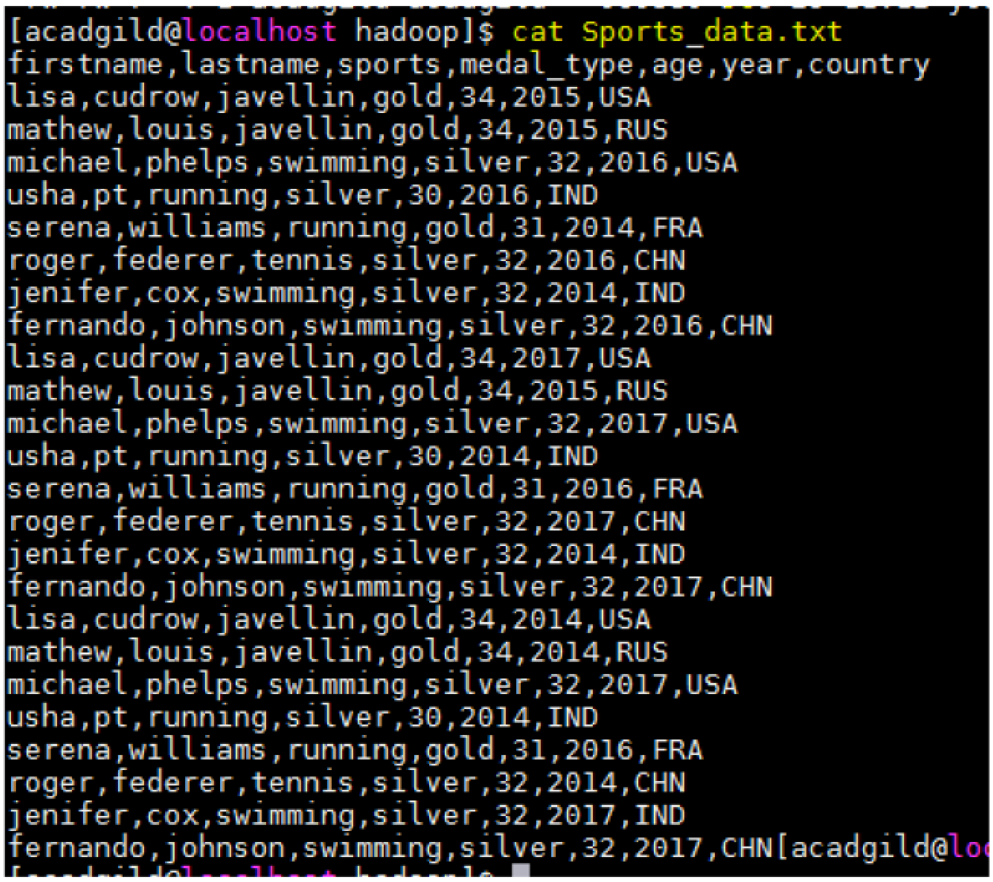
Session 21:
SPARK SQL 2
Assignment 1

Task 1
Using spark-sql, Find:
1. What are the total number of gold medal winners every year?
2. How many silver medals have been won by USA in each sport?

Task 2
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
2. 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

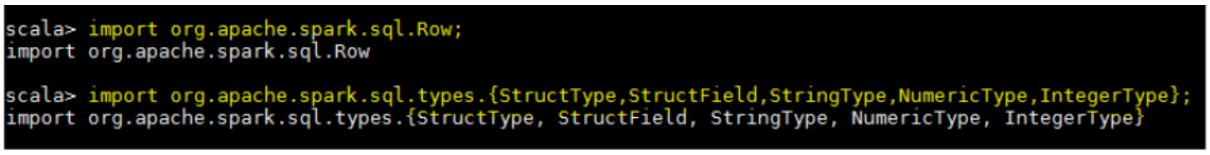
1. What are the total number of gold medal winners every year?



```
[acadgild@localhost hadoop]$ cat Sports_data.txt
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
serena,williams,running,gold,31,2016,FRA
roger,federer,tennis,silver,32,2014,CHN
jenifer,cox,swimming,silver,32,2017,IND
fernando,johnson,swimming,silver,32,2017,CHN[acadgild@lo
[acadgild@localhost hadoop]$
```

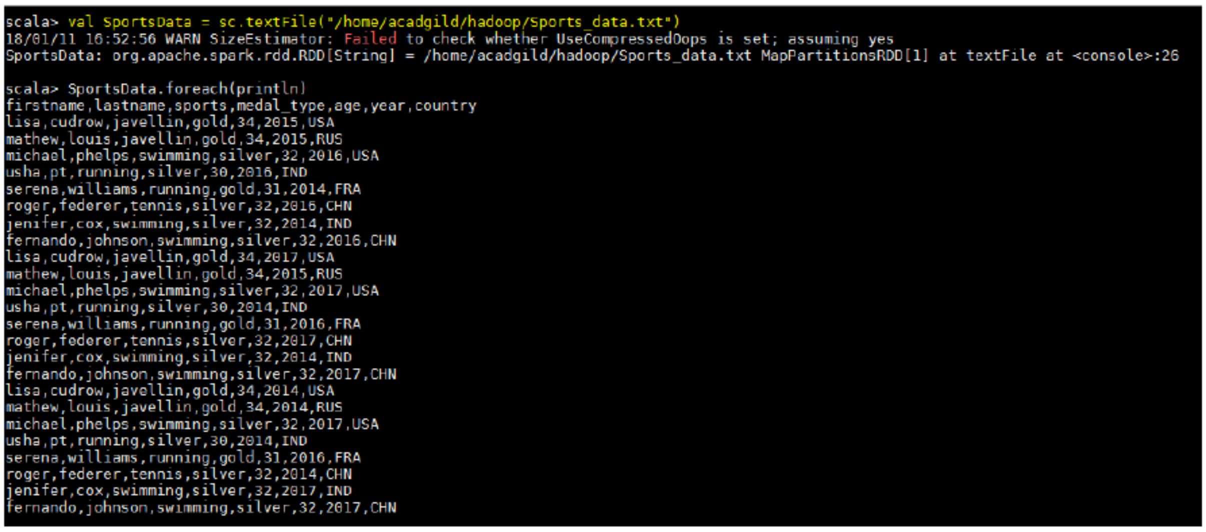
Code

```
import org.apache.spark.sql.Row;
import
org.apache.spark.sql.types.{StructType,StructField,StringType,NumericType,IntegerType};

val SportsData = sc.textFile("/home/acadgild/hadoop/Sports_data.txt")
val schemaString =
"firstname:string,lastname:string,sports:string,medal_type:string,age:string,year:string,count
ry:string"
val schema = StructType(schemaString.split(",").map(x =>
StructField(x.split(":")(0),if(x.split(":")(1).equals("string"))StringType else IntegerType, true)))
val rowRDD = SportsData.map(_.split(",")).map(r => Row(r(0), r(1), r(2), r(3), r(4), r(5), r(6)))
val SportsDataDF = spark.createDataFrame(rowRDD, schema)
SportsDataDF.createOrReplaceTempView("SportsData")
val resultDF = spark.sql("SELECT year,COUNT (*) FROM SportsData WHERE medal_type = 'gold'
GROUP BY year")
resultDF.show()
```

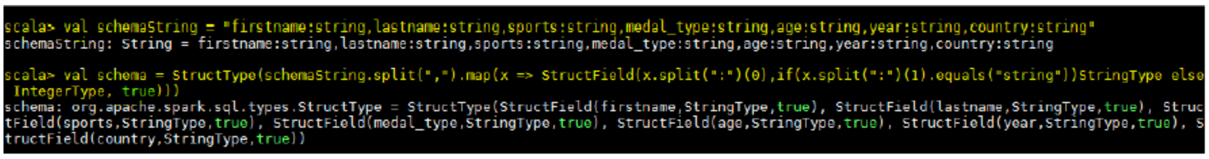


```
scala> import org.apache.spark.sql.Row;
import org.apache.spark.sql.Row

scala> import org.apache.spark.sql.types.{StructType,StructField,StringType,NumericType,IntegerType};
import org.apache.spark.sql.types.{StructType, StructField, StringType, NumericType, IntegerType}
```

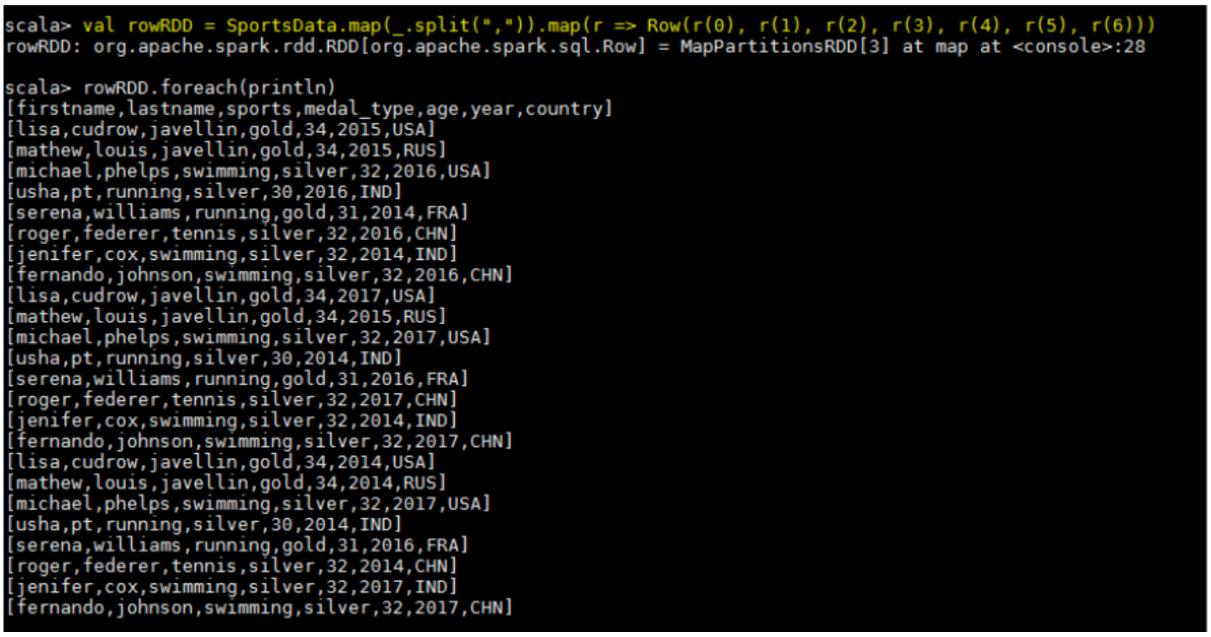
creating a RDD from Input DataSet,



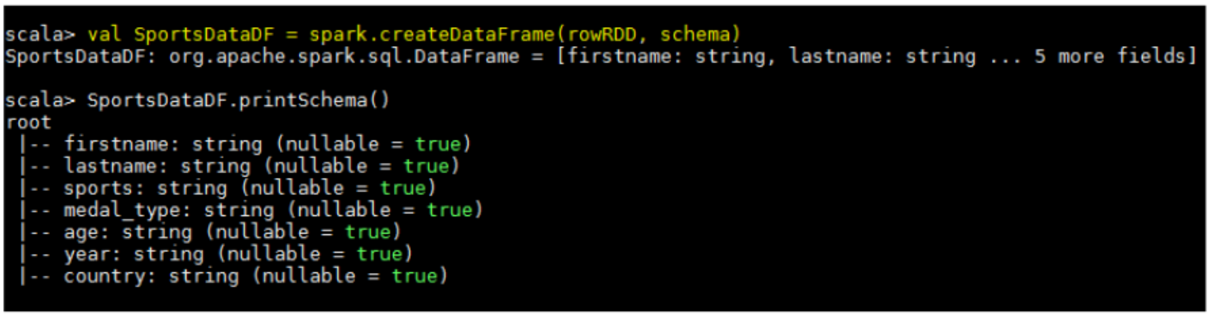
```
scala> val SportsData = sc.textFile("/home/acadgild/hadoop/Sports_data.txt")
18/01/11 16:52:56 WARN SizeEstimator: Failed to check whether UseCompressedOops is set; assuming yes
SportsData: org.apache.spark.rdd.RDD[String] = /home/acadgild/hadoop/Sports_data.txt MapPartitionsRDD[1] at textFile at <console>:26

scala> SportsData.foreach(println)
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
serena,williams,running,gold,31,2016,FRA
roger,federer,tennis,silver,32,2014,CHN
jenifer,cox,swimming,silver,32,2017,IND
fernando,johnson,swimming,silver,32,2017,CHN
```



```
scala> val schemaString = "firstname:string,lastname:string,sports:string,medal_type:string,age:string,year:string,country:string"
schemaString: String = firstname:string,lastname:string,sports:string,medal_type:string,age:string,year:string,country:string

scala> val schema = StructType(schemaString.split(",").map(x => StructField(x.split(":")(0),if(x.split(":")(1).equals("string"))StringType else
 IntegerType, true)))
schema: org.apache.spark.sql.types.StructType = StructType(StructField(firstname,StringType,true), StructField(lastname,StringType,true), Struc
tField(sports,StringType,true), StructField(medal_type,StringType,true), StructField(age,StringType,true), StructField(year,StringType,true), S
tructField(country,StringType,true))
```

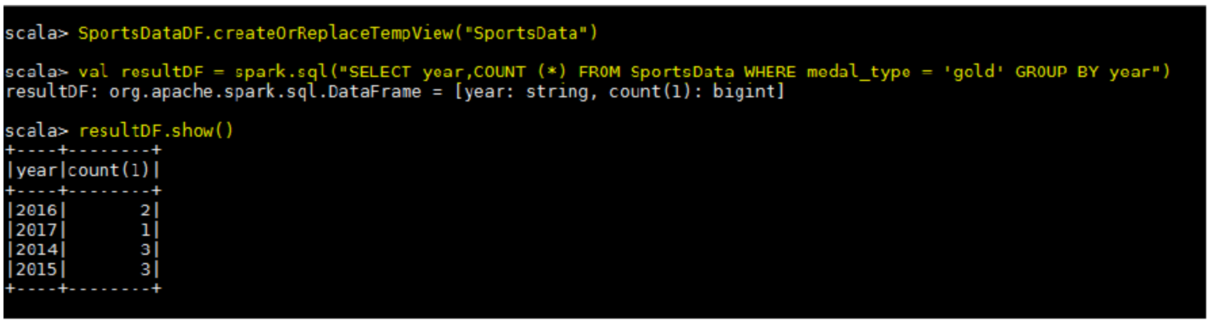


```
scala> val rowRDD = SportsData.map(_.split(",")).map(r => Row(r(0), r(1), r(2), r(3), r(4), r(5), r(6)))
rowRDD: org.apache.spark.rdd.RDD[org.apache.spark.sql.Row] = MapPartitionsRDD[3] at map at <console>:28

scala> rowRDD.foreach(println)
[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]
[serena,williams,running,gold,31,2016,FRA]
[roger,federer,tennis,silver,32,2014,CHN]
[jenifer,cox,swimming,silver,32,2017,IND]
[fernando,johnson,swimming,silver,32,2017,CHN]
```



```
scala> val SportsDataDF = spark.createDataFrame(rowRDD, schema)
SportsDataDF: org.apache.spark.sql.DataFrame = [firstname: string, lastname: string ... 5 more fields]

scala> SportsDataDF.printSchema()
root
 |-- firstname: string (nullable = true)
 |-- lastname: string (nullable = true)
 |-- sports: string (nullable = true)
 |-- medal_type: string (nullable = true)
 |-- age: string (nullable = true)
 |-- year: string (nullable = true)
 |-- country: string (nullable = true)
```

Result

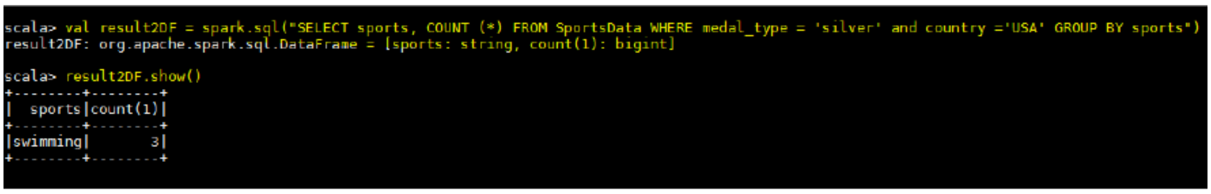


```
scala> SportsDataDF.createOrReplaceTempView("SportsData")

scala> val resultDF = spark.sql("SELECT year,COUNT (*) FROM SportsData WHERE medal_type = 'gold' GROUP BY year")
resultDF: org.apache.spark.sql.DataFrame = [year: string, count(1): bigint]

scala> resultDF.show()
+----+--------+
|year|count(1)|
+----+--------+
|2016|       2|
|2017|       1|
|2014|       3|
|2015|       3|
+----+--------+
```

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

Code
val result2DF = spark.sql("SELECT sports, COUNT (*) FROM SportsData WHERE medal_type = 'silver' and country ='USA' GROUP BY sports")
result2DF.show()

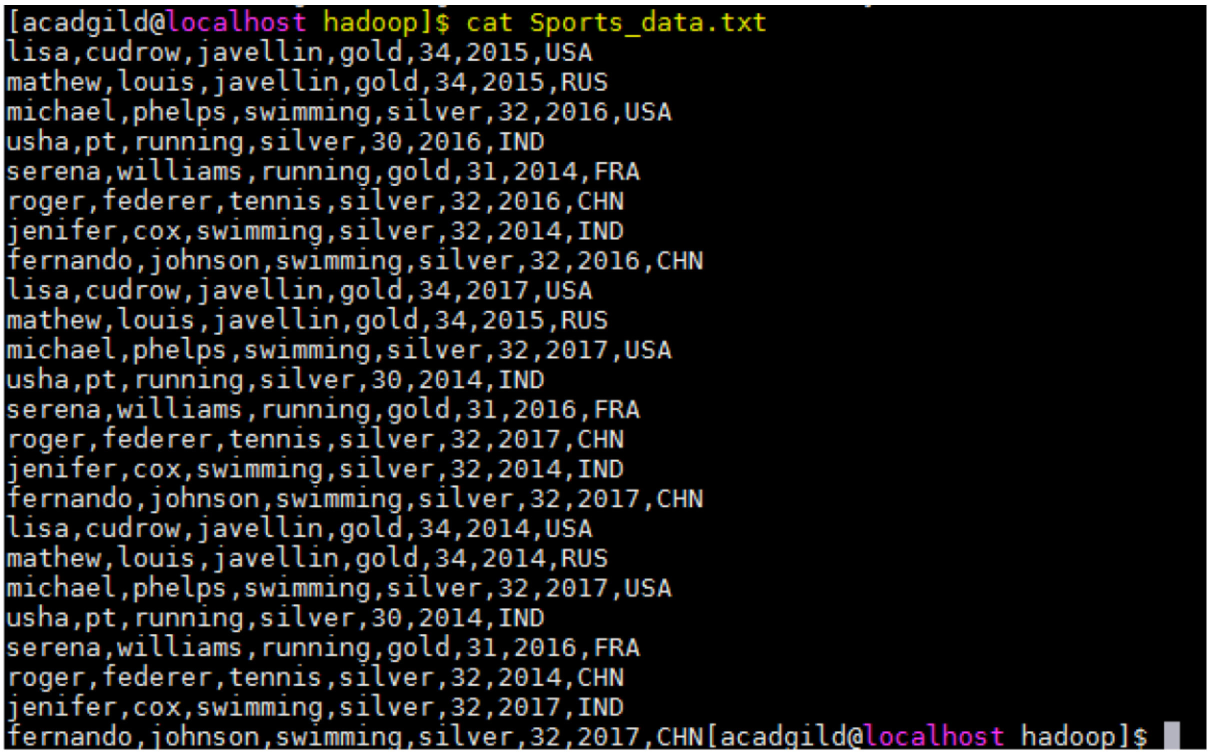
Result



```
scala> val result2DF = spark.sql("SELECT sports, COUNT (*) FROM SportsData WHERE medal_type = 'silver' and country ='USA' GROUP BY sports")
result2DF: org.apache.spark.sql.DataFrame = [sports: string, count(1): bigint]

scala> result2DF.show()
+--------+--------+
|  sports|count(1)|
+--------+--------+
|swimming|       3|
+--------+--------+
```

Task 2

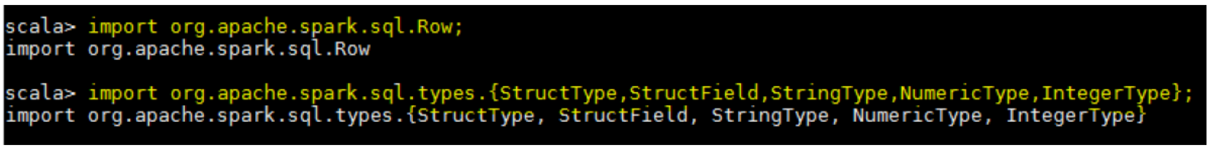
Using udfs on dataframe
1. Change firstname, lastname columns into
Mr.first_two_letters_of_firstname<space>lastname
for example - michael, phelps becomes Mr.mi phelps



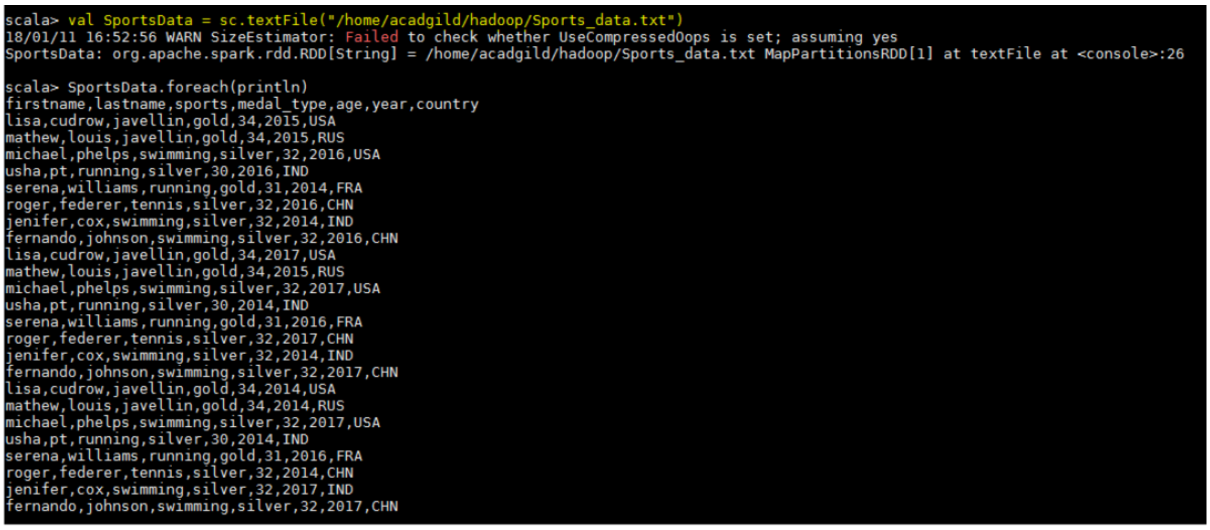
```
[acadgild@localhost hadoop]$ cat Sports_data.txt
lisa,cudrow,javellin,gold,34,2015,USA
mathew,louis,javellin,gold,34,2015,RUS
michael,phelps,swimming,silver,32,2016,USA
usha,pt,running,silver,30,2016,IND
serena,williams,running,gold,31,2014,FRA
roger,federer,tennis,silver,32,2016,CHN
jenifer,cox,swimming,silver,32,2014,IND
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
serena,williams,running,gold,31,2016,FRA
roger,federer,tennis,silver,32,2014,CHN
jenifer,cox,swimming,silver,32,2017,IND
fernando,johnson,swimming,silver,32,2017,CHN[acadgild@localhost hadoop]$
```

Code
import org.apache.spark.sql.Row;
import org.apache.spark.sql.types.{StructType,StructField,StringType,NumericType,IntegerType};
import org.apache.spark.sql.functions.udf

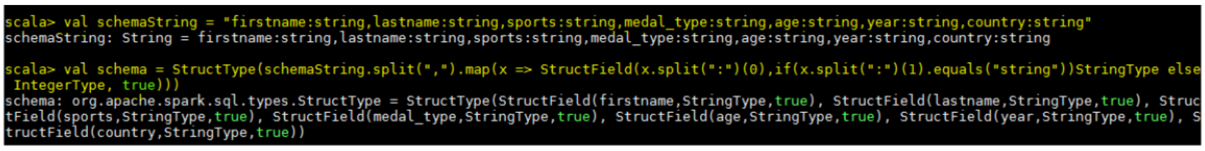
```
val SportsData = sc.textFile("/home/acadgild/hadoop/Sports_data.txt")
val schemaString =
"firstname:string,lastname:string,sports:string,medal_type:string,age:string,year:string,country:st
ring"
 val schema = StructType(schemaString.split(",").map(x =>
StructField(x.split(":")(0),if(x.split(":")(1).equals("string"))StringType else IntegerType, true)))
val rowRDD = SportsData.map(_.split(",")).map(r => Row(r(0), r(1), r(2), r(3), r(4), r(5), r(6)))
val SportsDataDF = spark.createDataFrame(rowRDD, schema)
SportsDataDF.createOrReplaceTempView("Sports_Data")
val Name = udf((firstname:String,
lastname:String)=>"Mr.".concat(firstname.substring(0,2)).concat(" ")concat(lastname))
spark.udf.register("Full_Name", Name)
val fname = spark.sql("SELECT Full_Name(firstname, lastname) FROM SportsData").show()
```



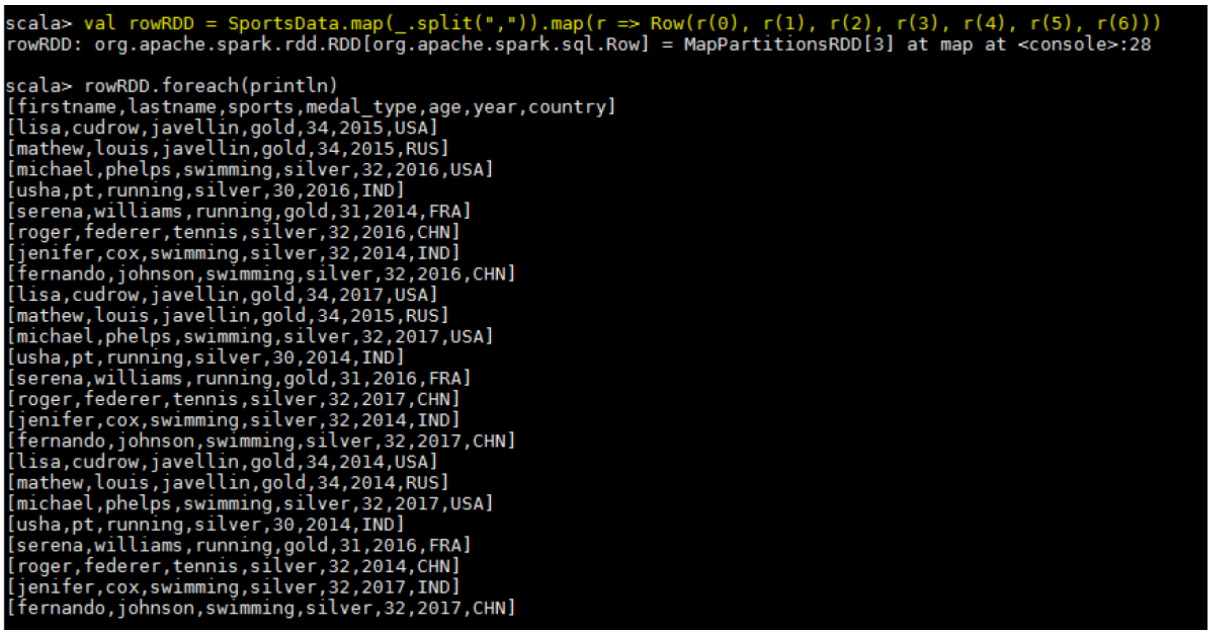
```
scala> import org.apache.spark.sql.Row;
import org.apache.spark.sql.Row

scala> import org.apache.spark.sql.types.{StructType,StructField,StringType,NumericType,IntegerType};
import org.apache.spark.sql.types.{StructType, StructField, StringType, NumericType, IntegerType}
```



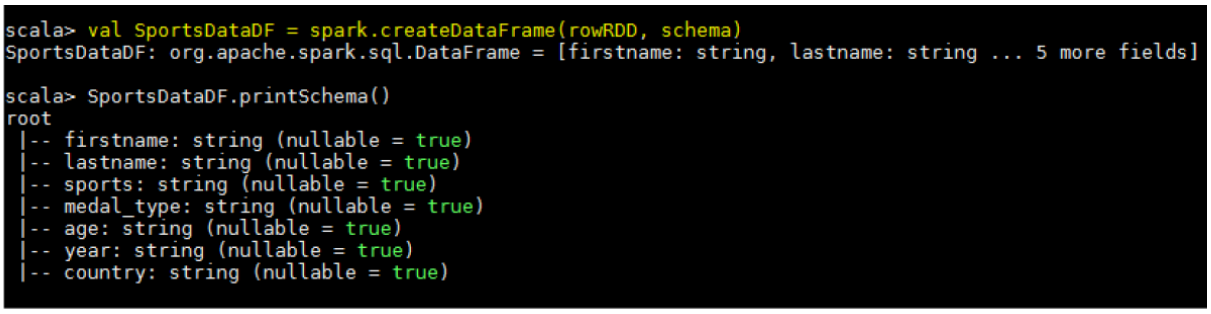
```
scala> val SportsData = sc.textFile("/home/acadgild/hadoop/Sports_data.txt")
18/01/11 16:52:56 WARN SizeEstimator: Failed to check whether UseCompressedOops is set; assuming yes
SportsData: org.apache.spark.rdd.RDD[String] = /home/acadgild/hadoop/Sports_data.txt MapPartitionsRDD[1] at textFile at <console>:26

scala> SportsData.foreach(println)
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
serena,williams,running,gold,31,2016,FRA
roger,federer,tennis,silver,32,2014,CHN
jenifer,cox,swimming,silver,32,2017,IND
fernando,johnson,swimming,silver,32,2017,CHN
```



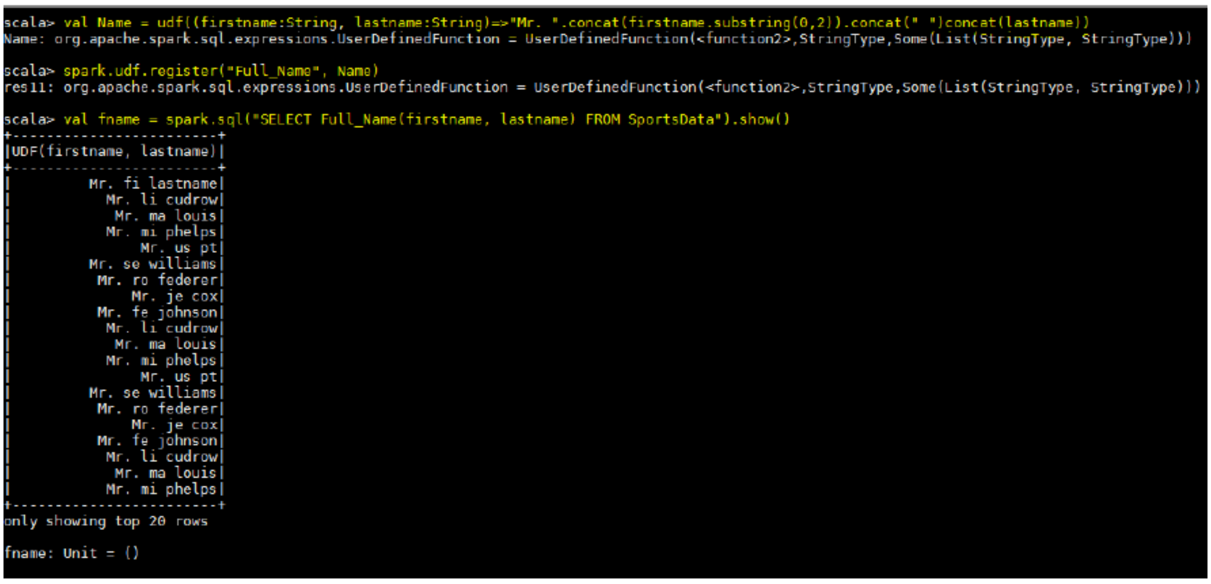
```
scala> val schemaString = "firstname:string,lastname:string,sports:string,medal_type:string,age:string,year:string,country:string"
schemaString: String = firstname:string,lastname:string,sports:string,medal_type:string,age:string,year:string,country:string

scala> val schema = StructType(schemaString.split(",").map(x => StructField(x.split(":")(0),if(x.split(":")(1).equals("string"))StringType else
 IntegerType, true)))
schema: org.apache.spark.sql.types.StructType = StructType(StructField(firstname,StringType,true), StructField(lastname,StringType,true), Struc
tField(sports,StringType,true), StructField(medal_type,StringType,true), StructField(age,StringType,true), StructField(year,StringType,true), S
tructField(country,StringType,true))
```



```
scala> val rowRDD = SportsData.map(_.split(",")).map(r => Row(r(0), r(1), r(2), r(3), r(4), r(5), r(6)))
rowRDD: org.apache.spark.rdd.RDD[org.apache.spark.sql.Row] = MapPartitionsRDD[3] at map at <console>:28

scala> rowRDD.foreach(println)
[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]
[serena,williams,running,gold,31,2016,FRA]
[roger,federer,tennis,silver,32,2014,CHN]
[jenifer,cox,swimming,silver,32,2017,IND]
[fernando,johnson,swimming,silver,32,2017,CHN]
```



```
scala> val SportsDataDF = spark.createDataFrame(rowRDD, schema)
SportsDataDF: org.apache.spark.sql.DataFrame = [firstname: string, lastname: string ... 5 more fields]

scala> SportsDataDF.printSchema()
root
 |-- firstname: string (nullable = true)
 |-- lastname: string (nullable = true)
 |-- sports: string (nullable = true)
 |-- medal_type: string (nullable = true)
 |-- age: string (nullable = true)
 |-- year: string (nullable = true)
 |-- country: string (nullable = true)
```

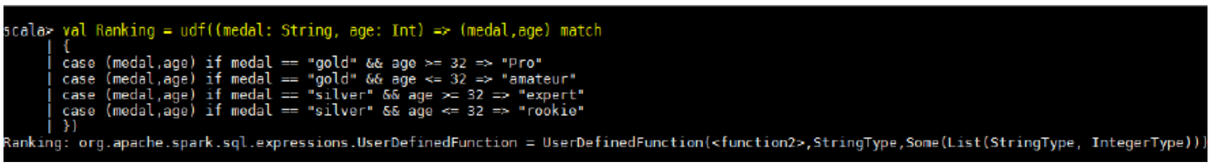


```
scala> val Name = udf((firstname:String, lastname:String)=>"Mr. ".concat(firstname.substring(0,2)).concat(" ")concat(lastname))
Name: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFunction(<function2>,StringType,Some(List(StringType, StringType)))

scala> spark.udf.register("Full_Name", Name)
res11: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFunction(<function2>,StringType,Some(List(StringType, StringType)))

scala> val fname = spark.sql("SELECT Full_Name(firstname, lastname) FROM SportsData").show()
+------------------------+
|UDF(firstname, lastname)|
+------------------------+
|         Mr. fi lastname|
|           Mr. li cudrow|
|            Mr. ma louis|
|           Mr. mi phelps|
|               Mr. us pt|
|         Mr. se williams|
|          Mr. ro federer|
|              Mr. je cox|
|          Mr. fe johnson|
|           Mr. li cudrow|
|            Mr. ma louis|
|           Mr. mi phelps|
|               Mr. us pt|
|         Mr. se williams|
|          Mr. ro federer|
|              Mr. je cox|
|          Mr. fe johnson|
|           Mr. li cudrow|
|            Mr. ma louis|
|           Mr. mi phelps|
+------------------------+
only showing top 20 rows

fname: Unit = ()
```

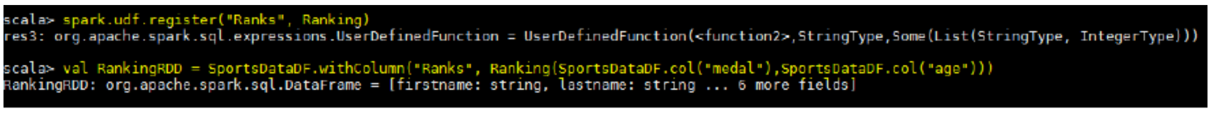
2. Add a new column called ranking using udfs on dataframe, where :
gold medalist, with age >= 32 are ranked as pro
gold medalists, with age <= 31 are ranked amateur
silver medalist, with age >= 32 are ranked as expert
silver medalists, with age <= 31 are ranked rookie

Code

```
val Ranking = udf((medal: String, age: Int) => (medal,age) match
{
case (medal,age) if medal == "gold" && age >= 32 => "Pro"
case (medal,age) if medal == "gold" && age <= 32 => "amateur"
case (medal,age) if medal == "silver" && age >= 32 => "expert"
case (medal,age) if medal == "silver" && age <= 32 => "rookie"
})

spark.udf.register("Ranks", Ranking)

val RankingRDD = SportsDataDF.withColumn("Ranks",
Ranking(SportsDataDF.col("medal"),SportsDataDF.col("age")))
```

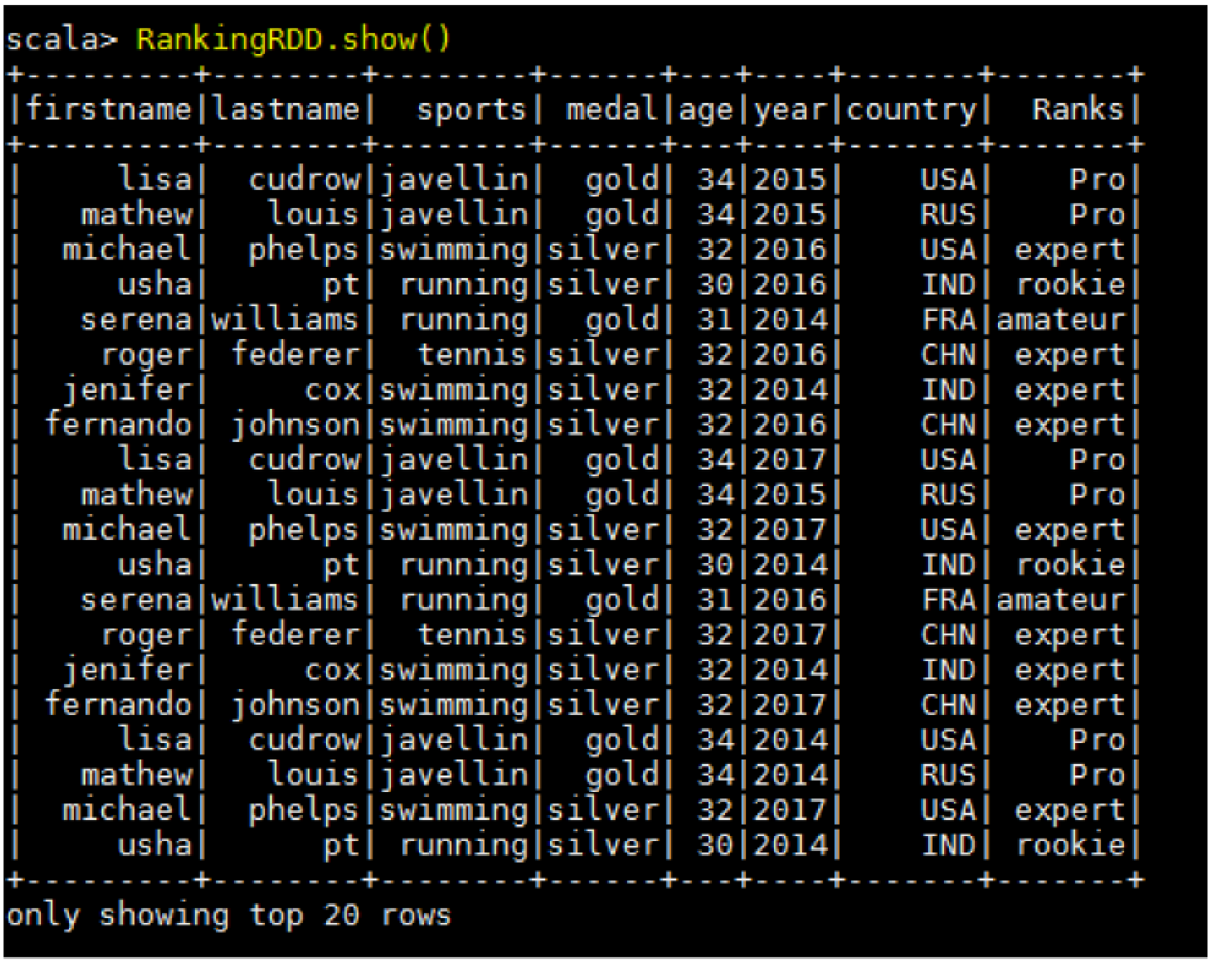


```
scala> val Ranking = udf((medal: String, age: Int) => (medal,age) match
     | {
     | case (medal,age) if medal == "gold" && age >= 32 => "Pro"
     | case (medal,age) if medal == "gold" && age <= 32 => "amateur"
     | case (medal,age) if medal == "silver" && age >= 32 => "expert"
     | case (medal,age) if medal == "silver" && age <= 32 => "rookie"
     | })
Ranking: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFunction(<function2>,StringType,Some(List(StringType, IntegerType)))
```



```
scala> spark.udf.register("Ranks", Ranking)
res3: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFunction(<function2>,StringType,Some(List(StringType, IntegerType)))

scala> val RankingRDD = SportsDataDF.withColumn("Ranks", Ranking(SportsDataDF.col("medal"),SportsDataDF.col("age")))
RankingRDD: org.apache.spark.sql.DataFrame = [firstname: string, lastname: string ... 6 more fields]
```

Result



```
scala> RankingRDD.show()
+---------+--------+--------+------+---+----+-------+-------+
|firstname|lastname|  sports| medal|age|year|country|  Ranks|
+---------+--------+--------+------+---+----+-------+-------+
|     lisa|  cudrow|javellin|  gold| 34|2015|    USA|    Pro|
|   mathew|   louis|javellin|  gold| 34|2015|    RUS|    Pro|
|  michael|  phelps|swimming|silver| 32|2016|    USA| expert|
|     usha|      pt| running|silver| 30|2016|    IND| rookie|
|   serena|williams| running|  gold| 31|2014|    FRA|amateur|
|    roger| federer|  tennis|silver| 32|2016|    CHN| expert|
|  jenifer|     cox|swimming|silver| 32|2014|    IND| expert|
| fernando| johnson|swimming|silver| 32|2016|    CHN| expert|
|     lisa|  cudrow|javellin|  gold| 34|2017|    USA|    Pro|
|   mathew|   louis|javellin|  gold| 34|2015|    RUS|    Pro|
|  michael|  phelps|swimming|silver| 32|2017|    USA| expert|
|     usha|      pt| running|silver| 30|2014|    IND| rookie|
|   serena|williams| running|  gold| 31|2016|    FRA|amateur|
|    roger| federer|  tennis|silver| 32|2017|    CHN| expert|
|  jenifer|     cox|swimming|silver| 32|2014|    IND| expert|
| fernando| johnson|swimming|silver| 32|2017|    CHN| expert|
|     lisa|  cudrow|javellin|  gold| 34|2014|    USA|    Pro|
|   mathew|   louis|javellin|  gold| 34|2014|    RUS|    Pro|
|  michael|  phelps|swimming|silver| 32|2017|    USA| expert|
|     usha|      pt| running|silver| 30|2014|    IND| rookie|
+---------+--------+--------+------+---+----+-------+-------+
only showing top 20 rows
```