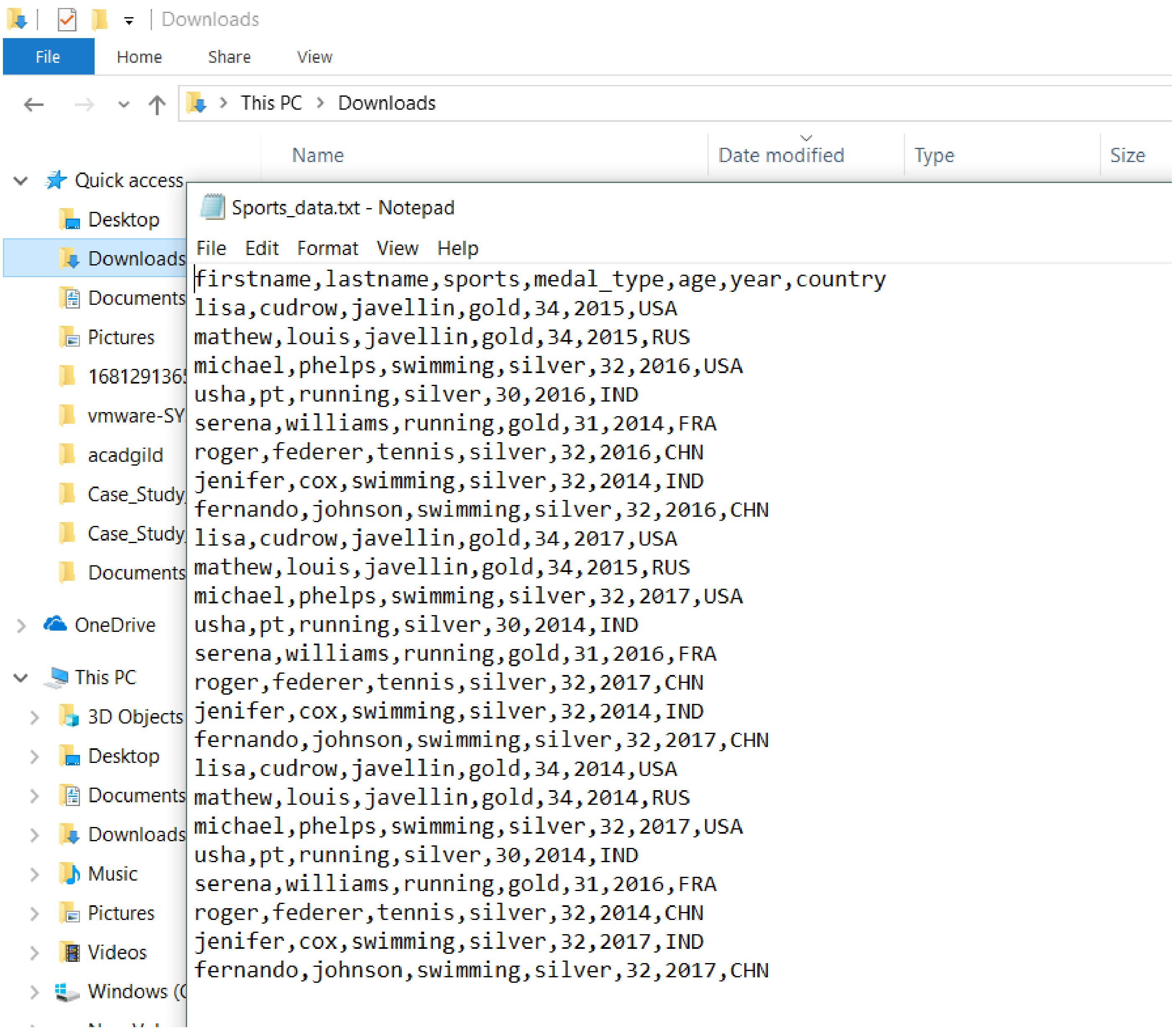
SPARK SQL – II ASSIGNMENT

Below provided Google Drive dataset is downloaded:

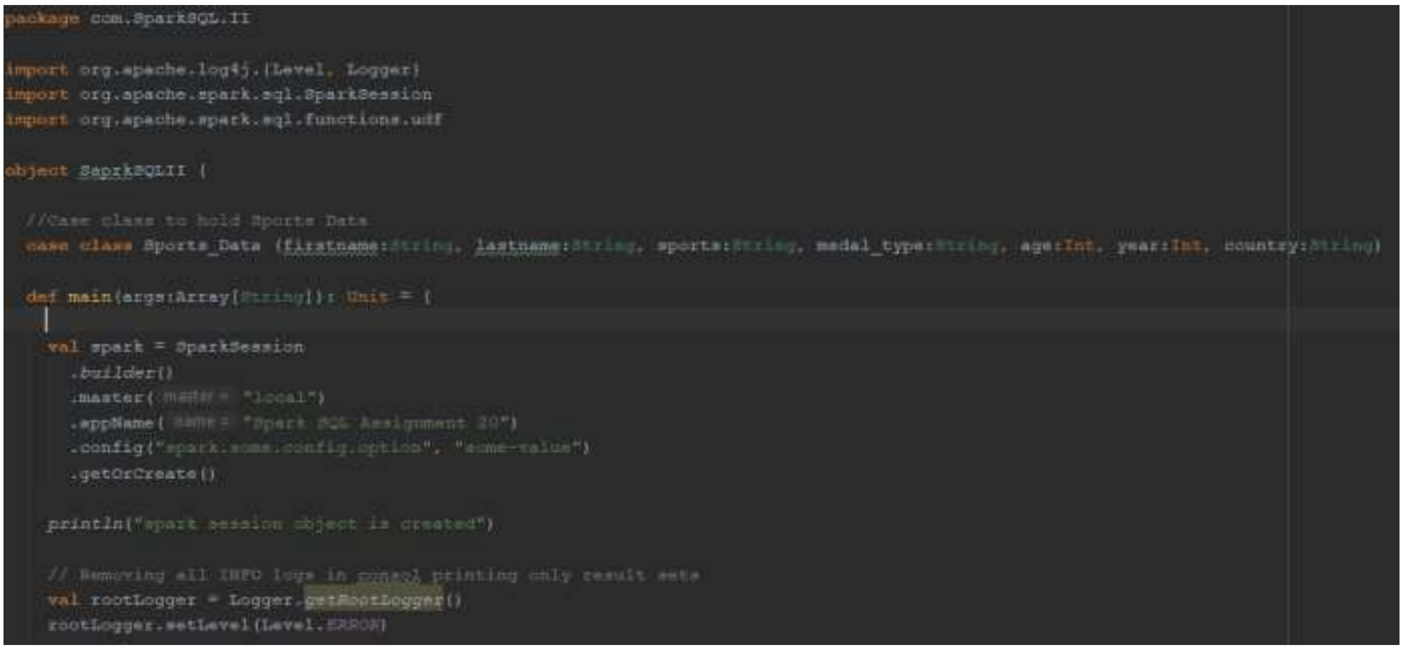


```
firstname,lastname,sports,medal_type,age,year,country
lisa,cudrow,javellin,gold,34,2015,USA
mathew,louis,javellin,gold,34,2015,RUS
michael,phelps,swimming,silver,32,2016,USA
usha,pt,running,silver,30,2016,IND
serena,williams,running,gold,31,2014,FRA
roger,federer,tennis,silver,32,2016,CHN
jenifer,cox,swimming,silver,32,2014,IND
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
```

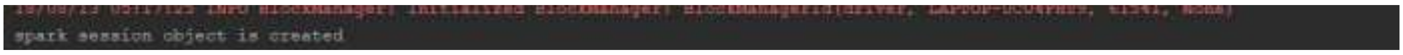
SPARK SQL – II ASSIGNMENT

- Created case class for each text files which represents the schema for the respective text files.
- Created a Spark Session Object
- Created a spark context to read the files from the local file system to spark by Matching the schema from case class
- Removed all INFO logs, set Log level to ERROR

Please find the screenshot below:



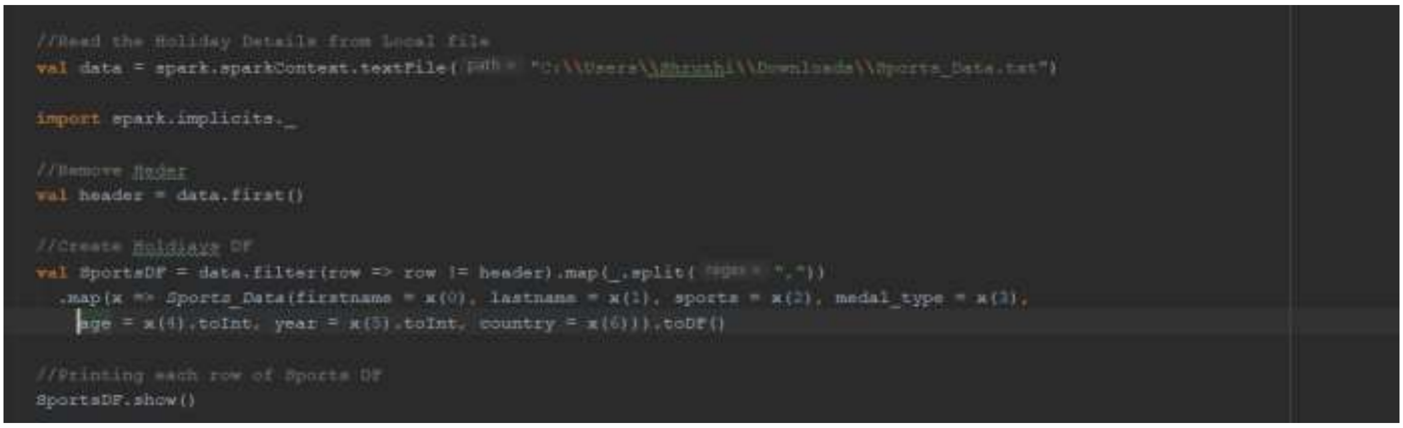
```
package com.SparkSQL.II

import org.apache.log4j.{Level, Logger}
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions.udf

object SaprkSQLII {

  //Case class to hold Sports Data
  case class Sports_Data (firstname:String, lastname:String, sports:String, medal_type:String, age:Int, year:Int, country:String)

  def main(args:Array[String]): Unit = {

    val spark = SparkSession
      .builder()
      .master( master = "local")
      .appName( name = "Spark SQL Assignment 20")
      .config("spark.some.config.option", "some-value")
      .getOrCreate()

    println("spark session object is created")

    // Removing all INFO logs in consol printing only result sets
    val rootLogger = Logger.getRootLogger()
    rootLogger.setLevel(Level.ERROR)
```



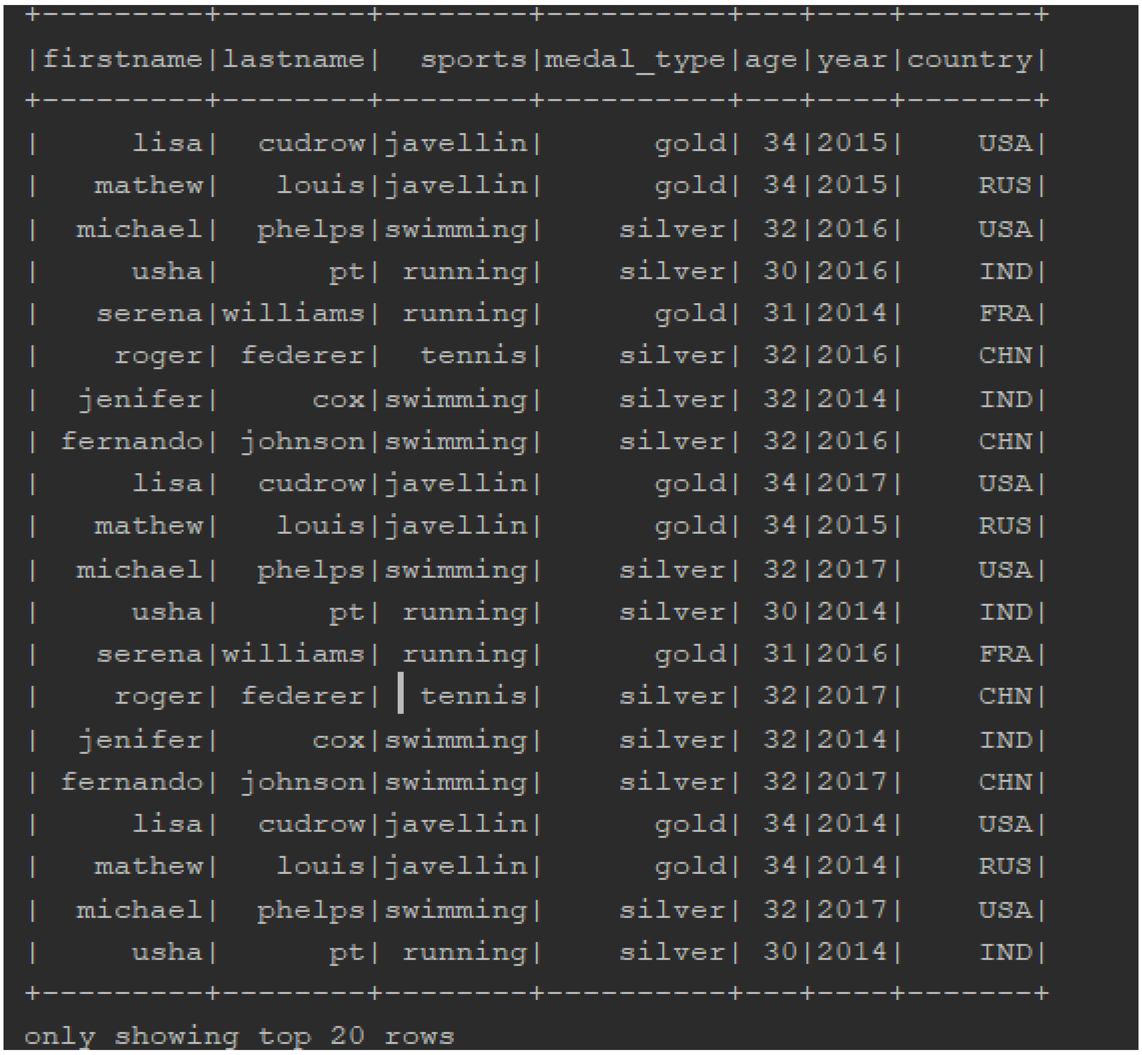
```
18/08/13 03:17:12 INFO BlockManager: Initialized BlockManager: BlockManagerId(driver, LAPTOP-[illegible], [illegible], None)
spark session object is created
```

SPARK SQL – II ASSIGNMENT

- Please find the code to load .txt file into spark
- Remove header
- Create data frame out of data using regex, comma separated
- Printing Data Frame



```
//Read the Holiday Details from Local file
val data = spark.sparkContext.textFile( path = "C:\\Users\\Shruthi\\Downloads\\Sports_Data.txt")

import spark.implicits._

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

//Create Holidays DF
val SportsDF = data.filter(row => row != header).map(_.split( regex = ","))
  .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()

//Printing each row of Sports DF
SportsDF.show()
```



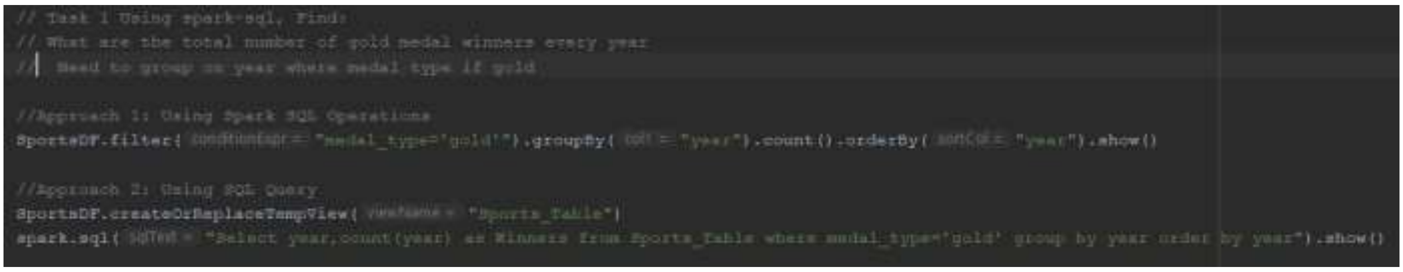
```
+---------+--------+--------+----------+---+----+-------+
|firstname|lastname|  sports|medal_type|age|year|country|
+---------+--------+--------+----------+---+----+-------+
|     lisa|  cudrow|javellin|      gold| 34|2015|    USA|
|   mathew|   louis|javellin|      gold| 34|2015|    RUS|
|  michael|  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
```

SPARK SQL – II ASSIGNMENT

Task 1

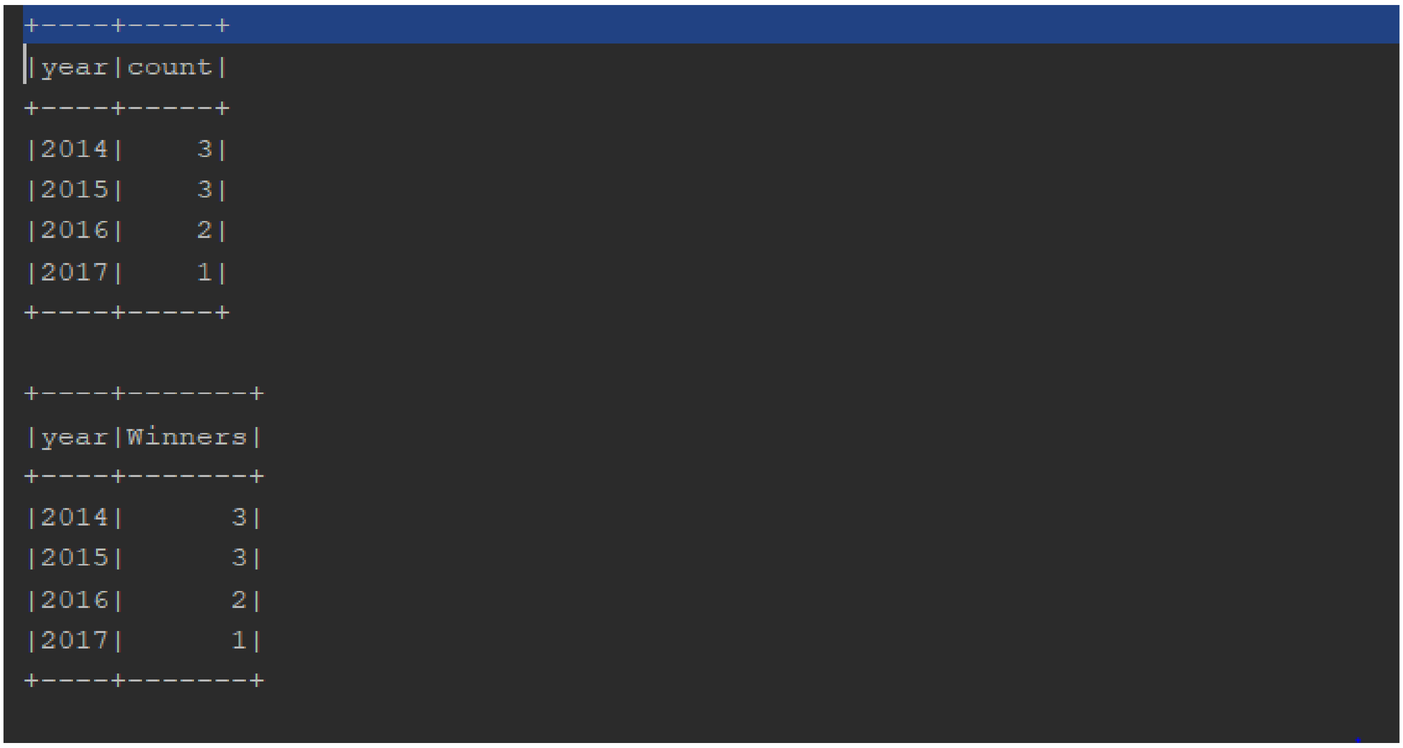
Using spark-sql, Find:

1. What are the total number of gold medal winners every year
2. By using below approaches we will be able to find the gold medal winners every year



```
// Task 1 Using spark-sql, Find:
// What are the total number of gold medal winners every year
//  Need to group on year where medal type if gold

//Approach 1: Using Spark SQL Operations
SportsDF.filter( conditionExpr = "medal_type='gold'").groupBy( col1 = "year").count().orderBy( sortCol = "year").show()

//Approach 2: Using SQL Query
SportsDF.createOrReplaceTempView( viewName = "Sports_Table")
spark.sql( sqlText = "Select year,count(year) as Winners from Sports_Table where medal_type='gold' group by year order by year").show()
```

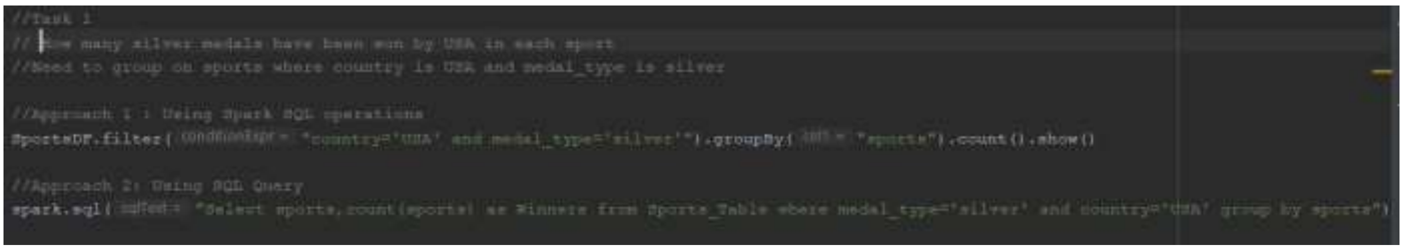
O/P:-



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

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

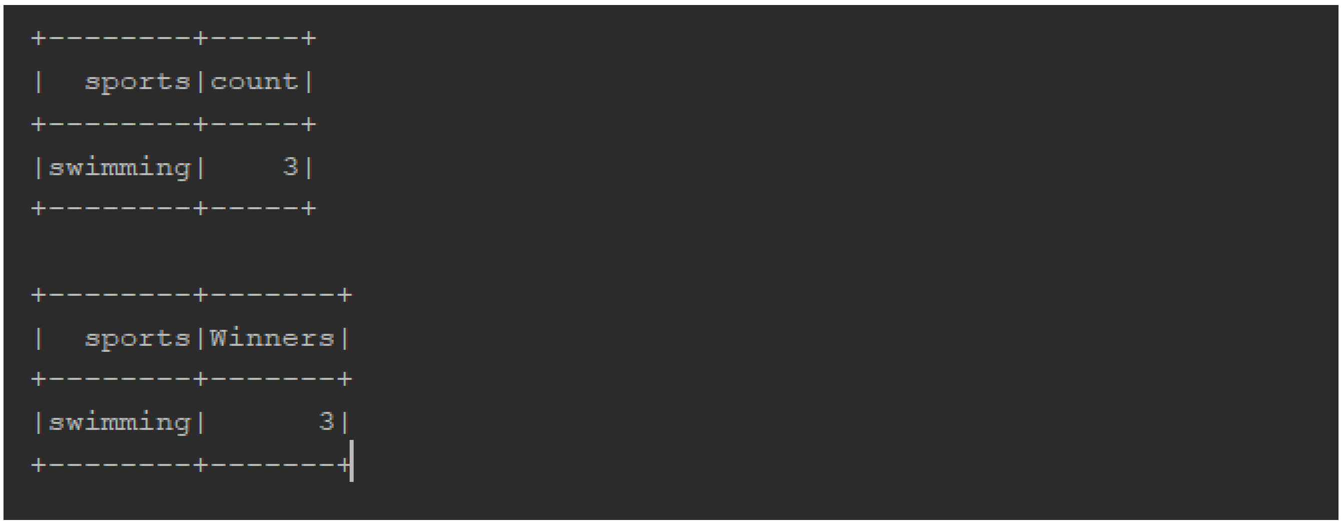
SPARK SQL – II ASSIGNMENT

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



```
//Task 1
//How many silver medals have been won by USA in each sport
//Need to group on sports where country is USA and medal_type is silver

//Approach 1 : Using Spark SQL operations
SportsDF.filter( conditionExpr = "country='USA' and medal_type='silver'").groupBy( col1 = "sports").count().show()

//Approach 2: Using SQL Query
spark.sql( sqlText = "Select sports,count(sports) as Winners from Sports_Table where medal_type='silver' and country='USA' group by sports")
```

O/P:-

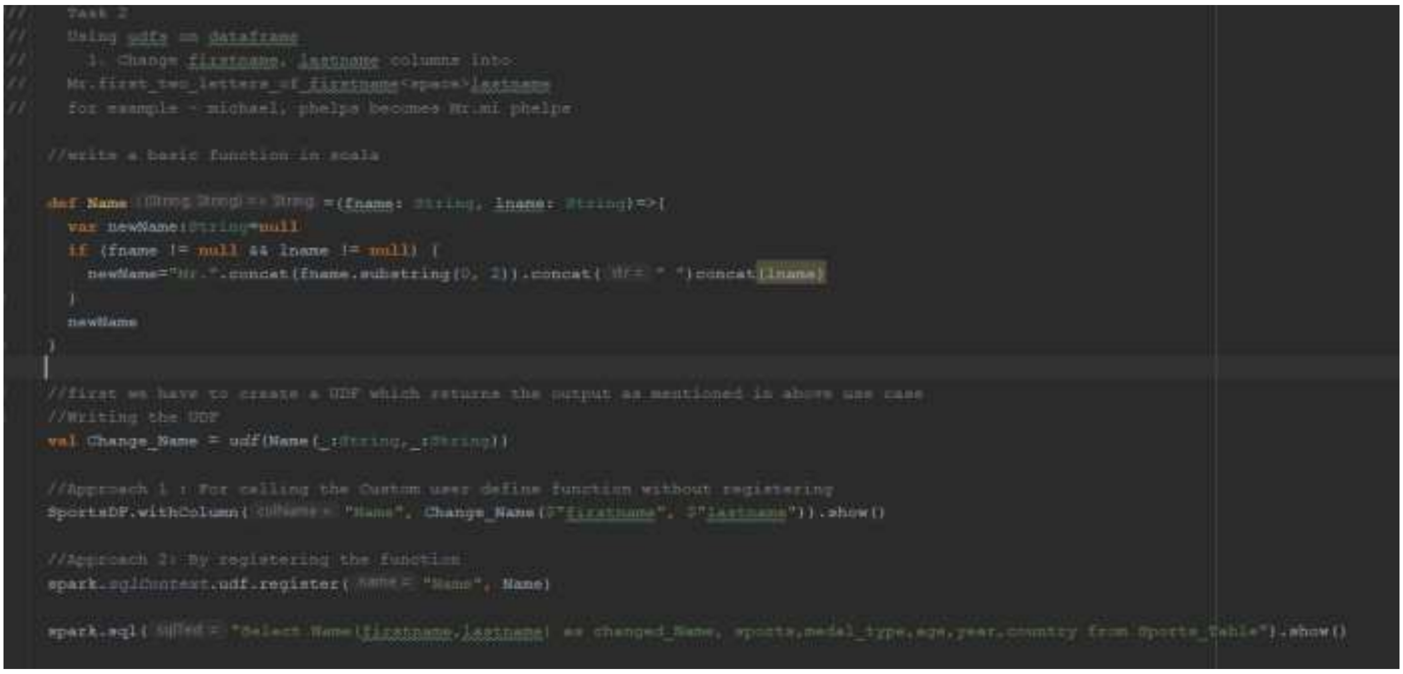


```
+--------+-----+
|  sports|count|
+--------+-----+
|swimming|    3|
+--------+-----+

+--------+-------+
|  sports|Winners|
+--------+-------+
|swimming|      3|
+--------+-------+
```

SPARK SQL – II ASSIGNMENT

Task 2

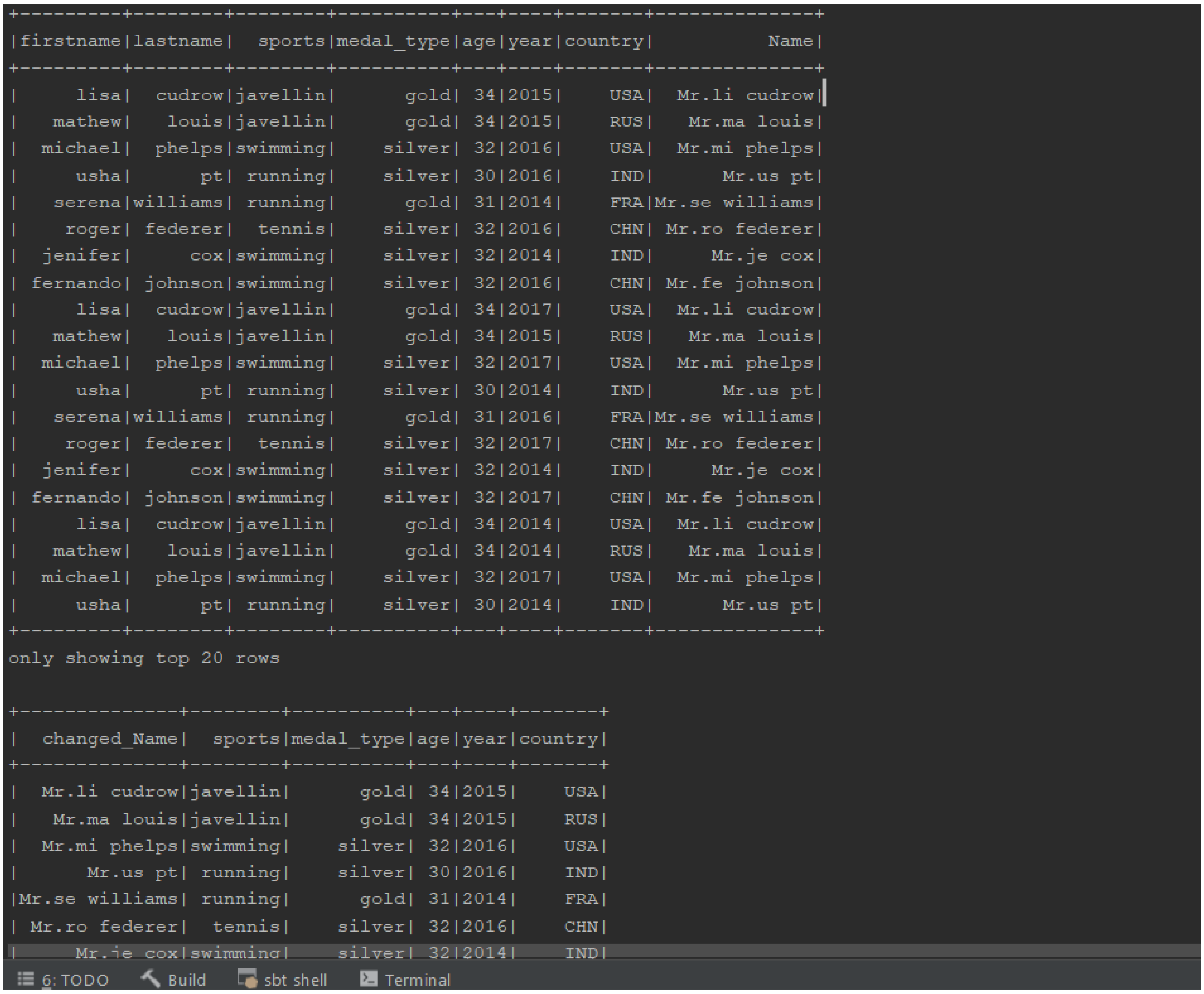
Using udfs on dataframe
1. Change firstname, lastname columns into
Mr.first_two_letters_of_firstname<space>lastname
for example - michael, phelps becomes Mr.mi phelps



```scala
//    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

  //write a basic function in scala

  def Name (String, String) => String =(fname: String, lname: String)=>{
    var newName:String=null
    if (fname != null && lname != null) {
      newName="Mr.".concat(fname.substring(0, 2)).concat( str = " ")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( colName = "Name", Change_Name($"firstname", $"lastname")).show()

  //Approach 2: By registering the function
  spark.sqlContext.udf.register( name = "Name", Name)

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

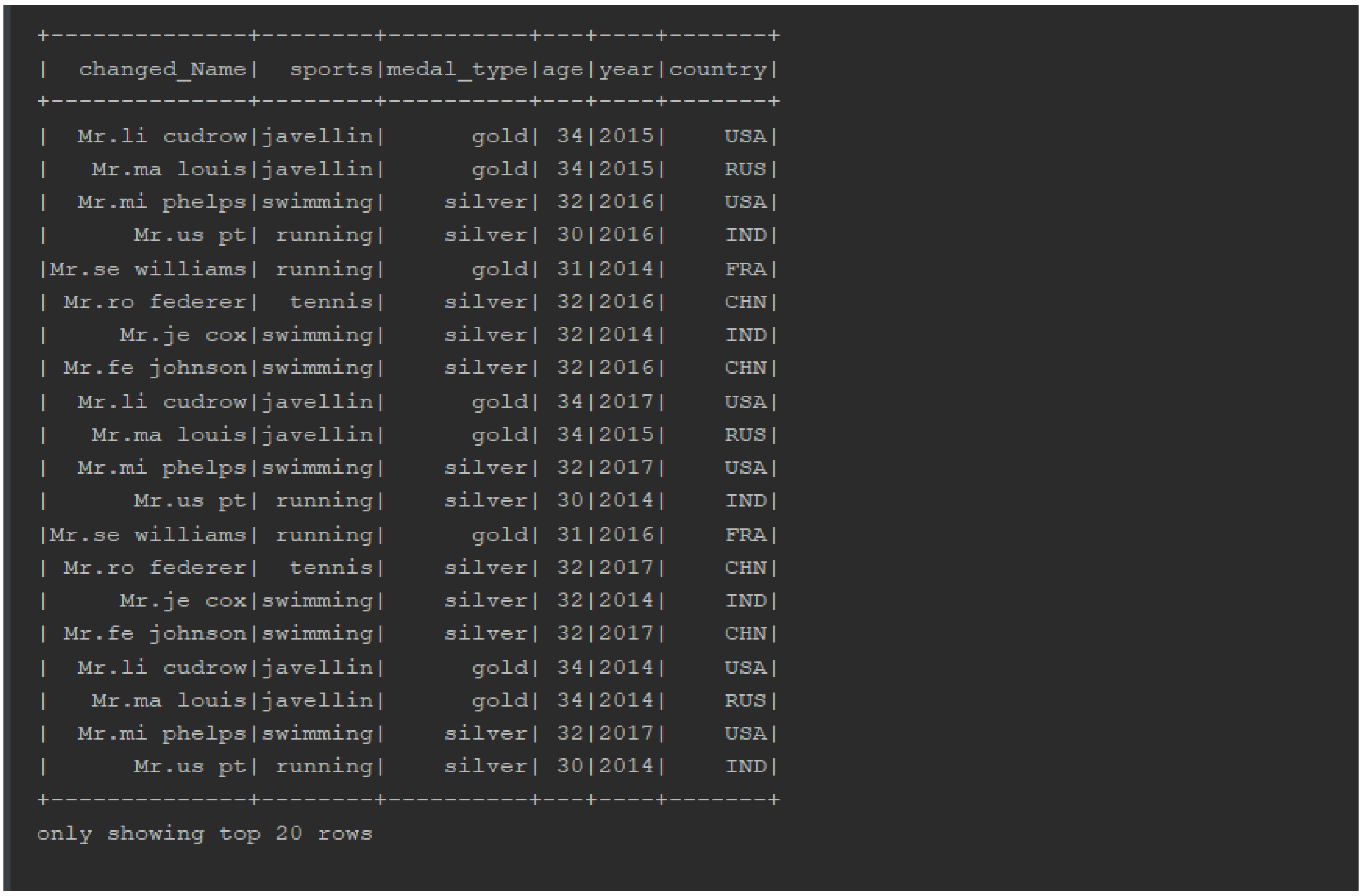
O/P:-

SPARK SQL – II ASSIGNMENT



```
+---------+--------+--------+----------+---+----+-------+--------------+
|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

+--------------+--------+----------+---+----+-------+
|  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|
```

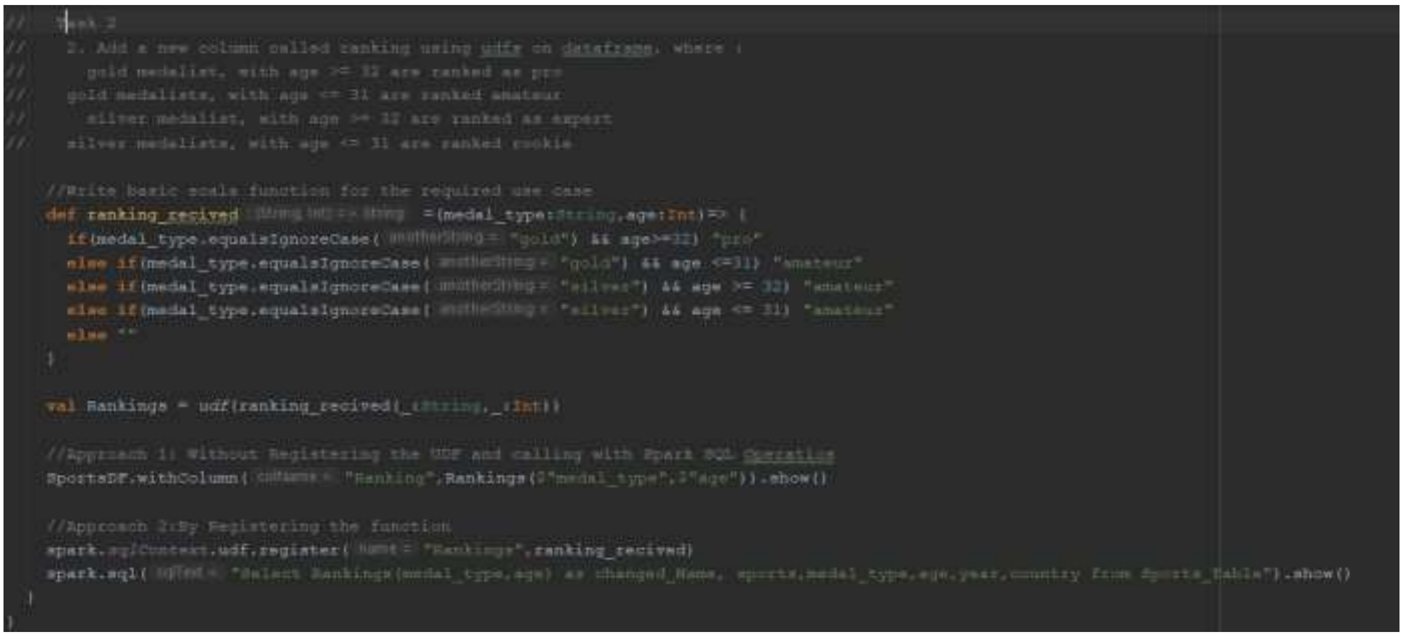
SPARK SQL – II ASSIGNMENT



```
+--------------+--------+----------+---+----+-------+
|  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
```

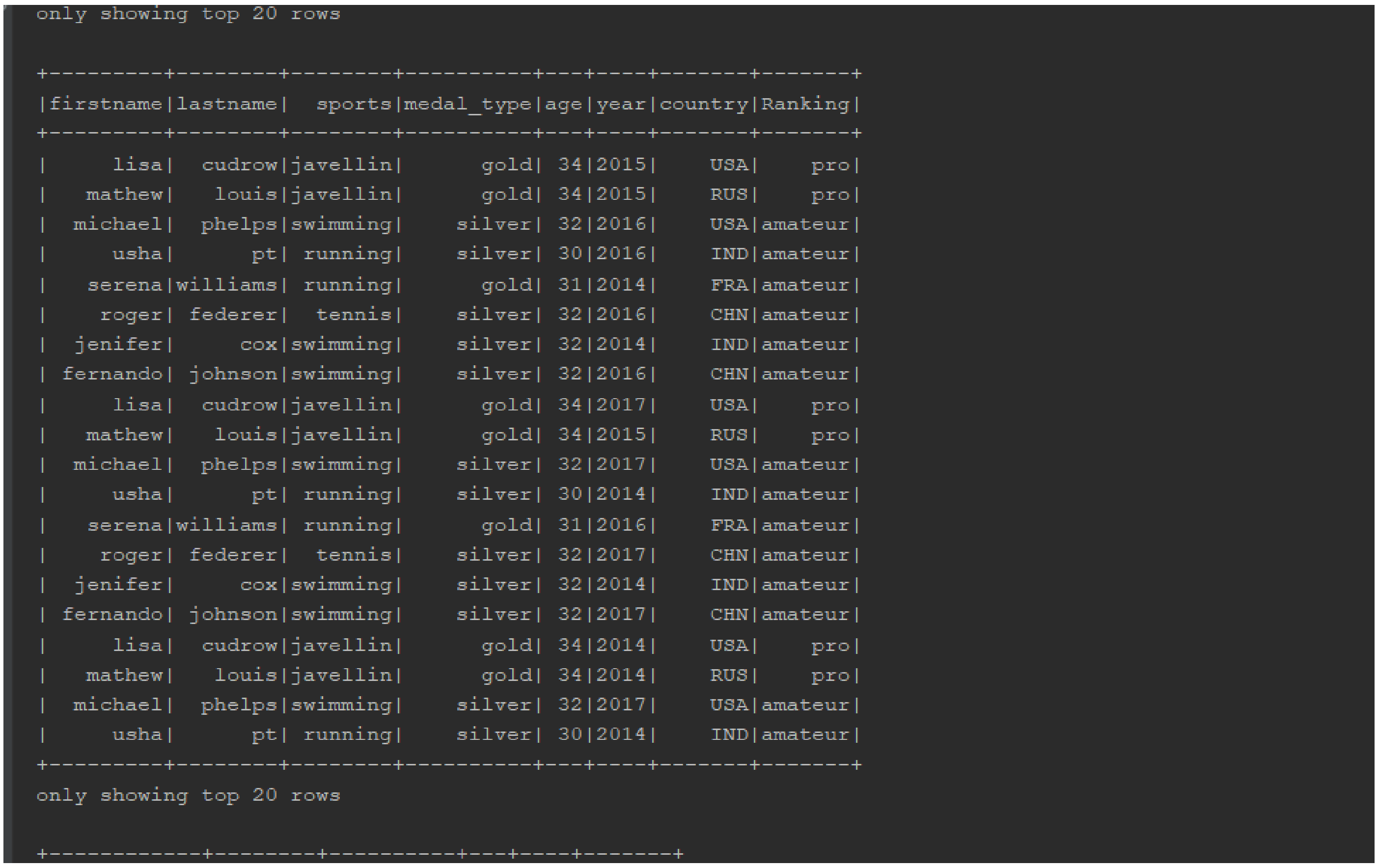
SPARK SQL – II ASSIGNMENT

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



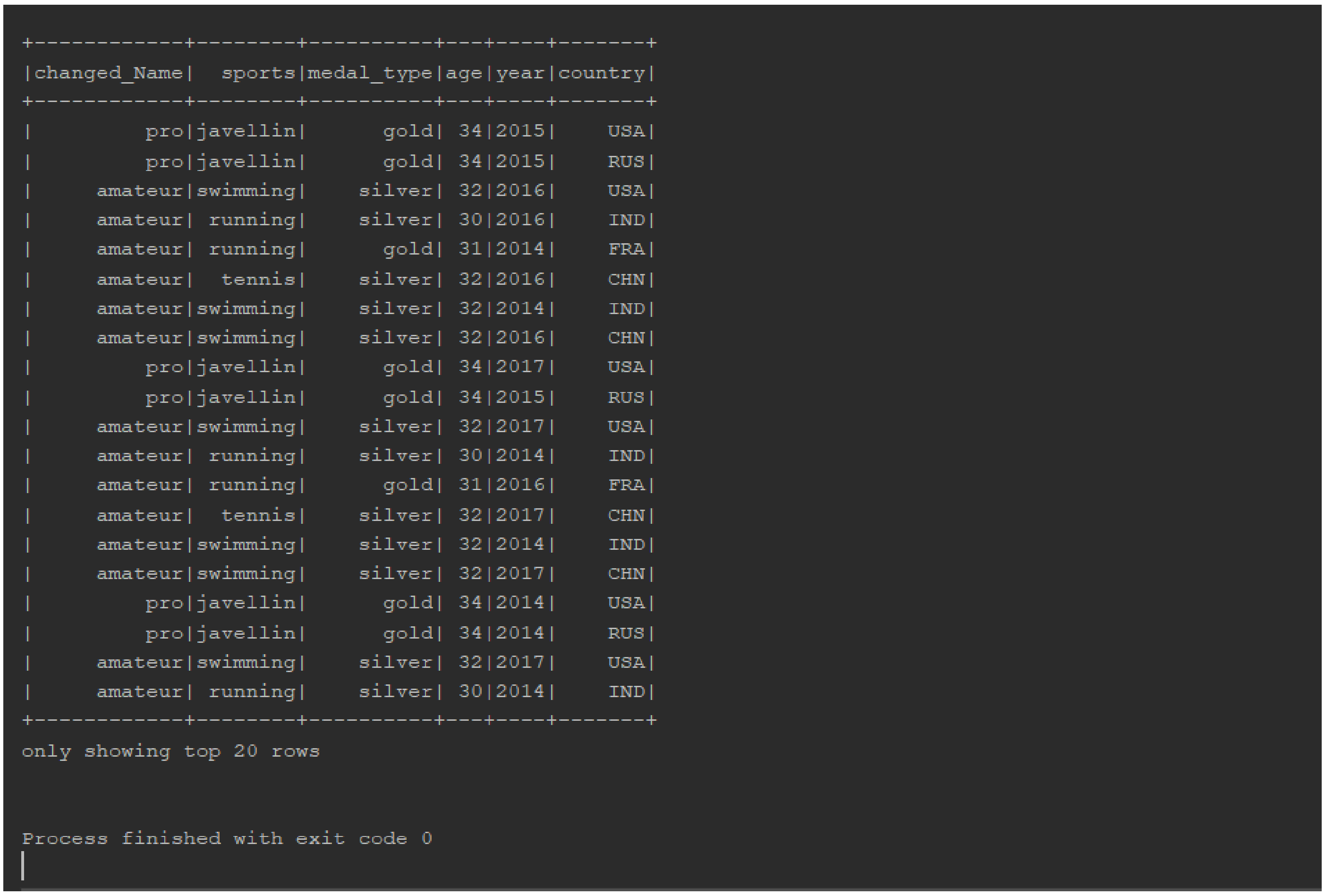
```
//   Task 2
//    2. Add a new column called ranking using udfs on dataframe, where :
//      gold medalist, with age >= 32 are ranked as pro
//    gold medalists, with age <= 31 are ranked amateur
//      silver medalist, with age >= 32 are ranked as expert
//    silver medalists, with age <= 31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

  //Write basic scala function for the required use case
  def ranking_recived (String, Int) => String  =(medal_type:String,age:Int)=> {
    if(medal_type.equalsIgnoreCase( anotherString = "gold") && age>=32) "pro"
    else if(medal_type.equalsIgnoreCase( anotherString = "gold") && age <=31) "amateur"
    else if(medal_type.equalsIgnoreCase( anotherString = "silver") && age >= 32) "amateur"
    else if(medal_type.equalsIgnoreCase( anotherString = "silver") && age <= 31) "amateur"
    else ""
  }

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

  //Approach 1: Without Registering the UDF and calling with Spark SQL Operatios
  SportsDF.withColumn( colName = "Ranking",Rankings($"medal_type",$"age")).show()

  //Approach 2:By Registering the function
  spark.sqlContext.udf.register( name = "Rankings",ranking_recived)
  spark.sql( sqlText = "Select Rankings(medal_type,age) as changed_Name, sports,medal_type,age,year,country from Sports_Table").show()
 }
}
```

O/P:-



```
only showing top 20 rows

+---------+--------+--------+----------+---+----+-------+-------+
|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

+------------+--------+----------+---+----+-------+
```

SPARK SQL – II ASSIGNMENT



```
+------------+--------+----------+---+----+-------+
|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


Process finished with exit code 0
```