Session 21:

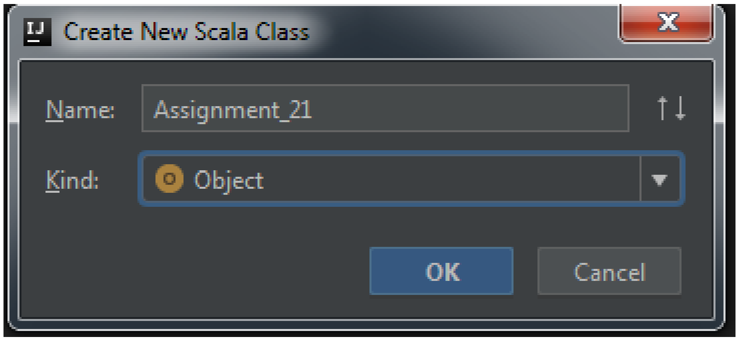
SPARK SQL 2

Assignment 1

- Prachi Mohite

In this Assignment we will be using IDEA IntelliJ to Complete the given Task

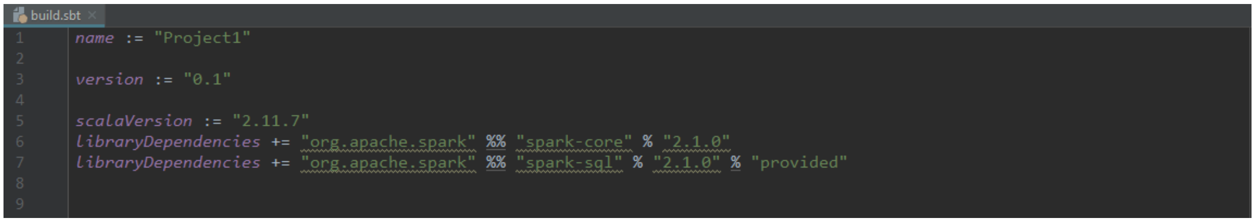
1. Created new Project and added scala object named as Assignment_21 as below



Name: Assignment_21

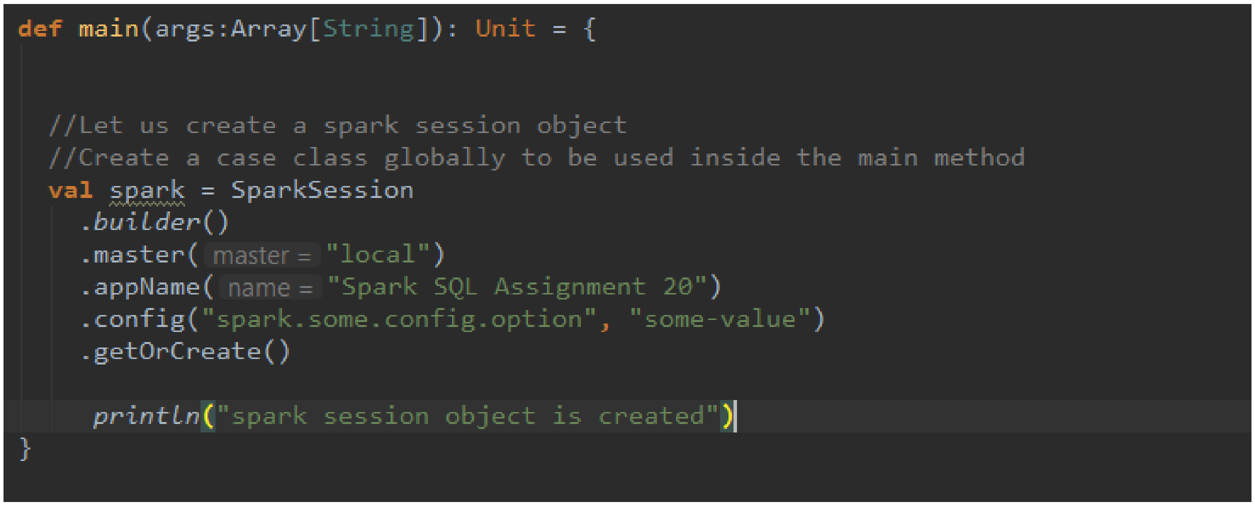
Kind: Object

2. To add the required dependencies we have created scala sbt project in IDEA and added library dependency from maven repository as below



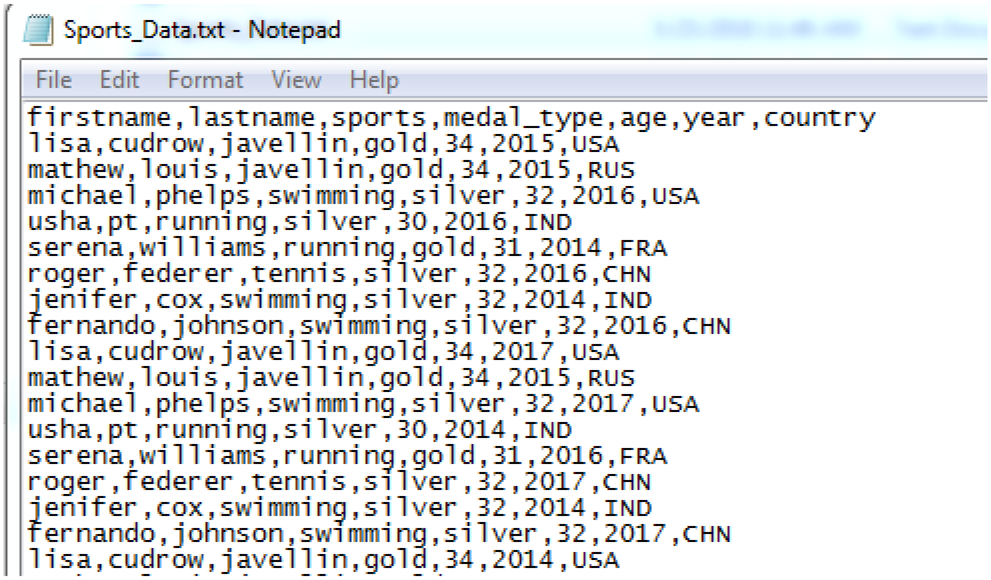
```
name := "Project1"

version := "0.1"

scalaVersion := "2.11.7"
libraryDependencies += "org.apache.spark" %% "spark-core" % "2.1.0"
libraryDependencies += "org.apache.spark" %% "spark-sql" % "2.1.0" % "provided"


```

3. Added main function and created the spark object as below



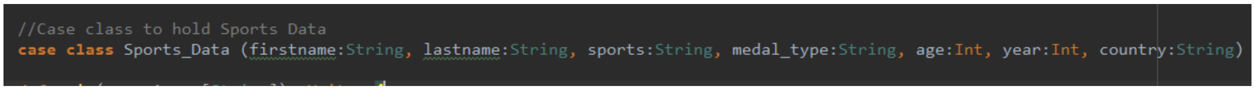
```
def main(args:Array[String]): Unit = {


  //Let us create a spark session object
  //Create a case class globally to be used inside the main method
  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")
}
```

4. We will be using below dataset for this assignment
    a. Sports_Data.txt
       Columns are – firstname,lastname,sports,medal_type,age,year,country



```
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
```

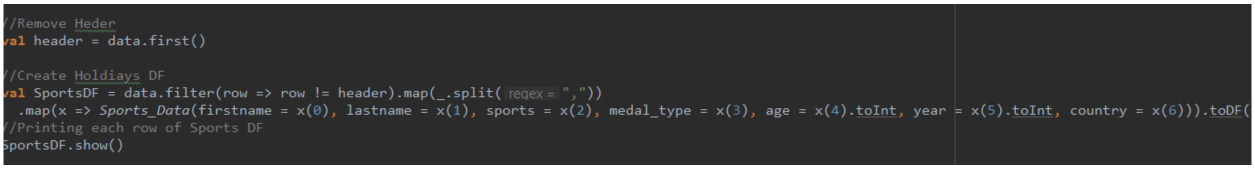
5. To Complete the assignment first we have to load the data from these local file to Dataframe in Spark SQL as below
    a. Created the case class to map the details in Dataframe from text file
        i. Case Class for Sports Data



```
//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)
```

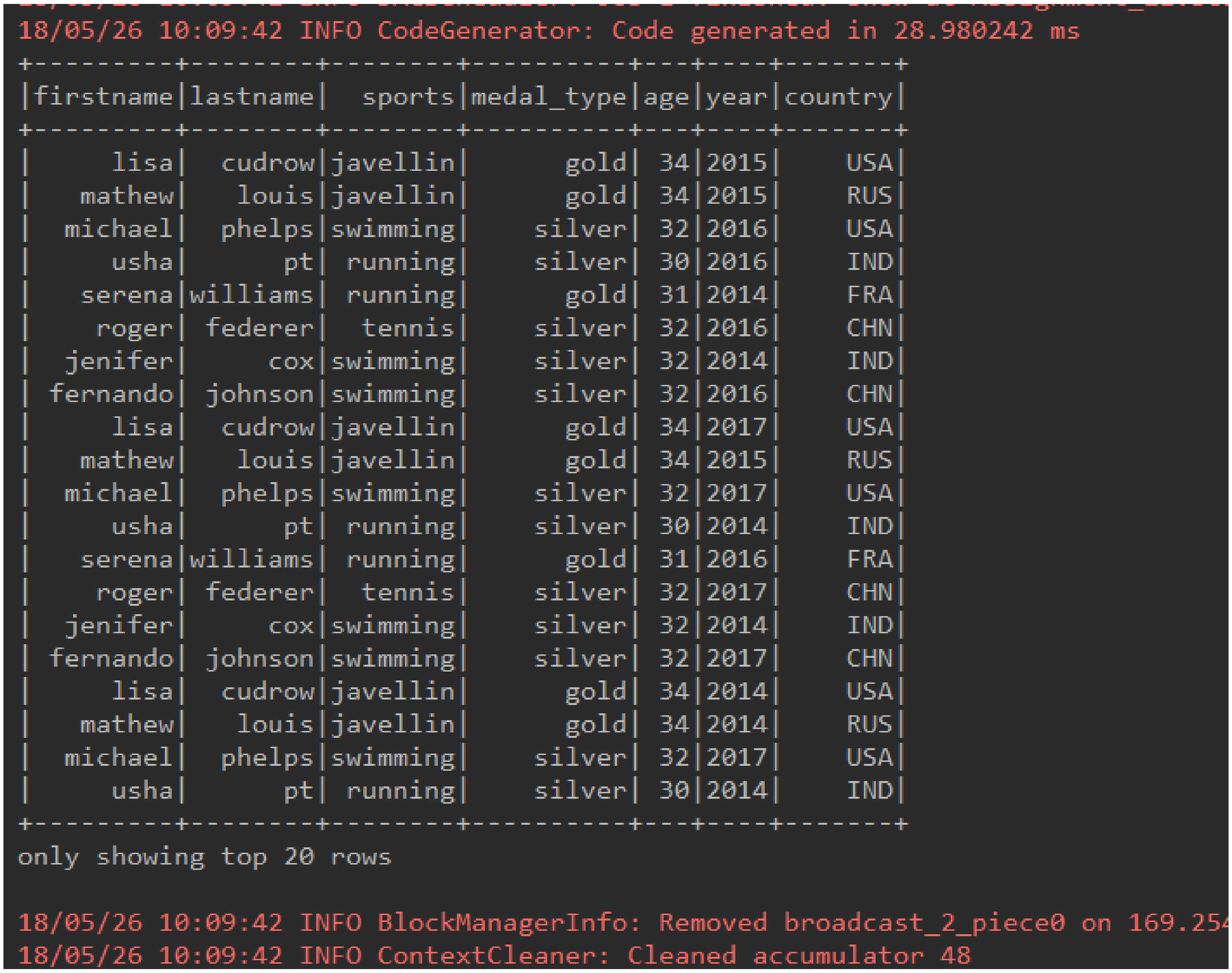
6. Load data from above created RDD in dataframe
    a. Before doing that we have to remove the header present in the sports data RDD. This will be achieved by using first method and get all the rows not same as the header
    b. Then DF is created from above dataset as below

Code



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

//Create Holdiays 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()
```

Output of Sports Data DF Show method



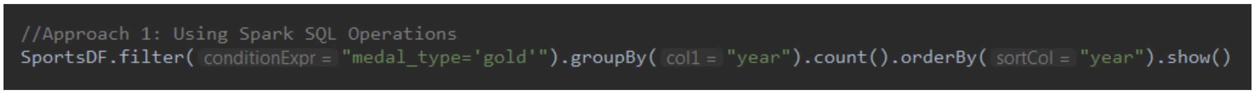
```
18/05/26 10:09:42 INFO CodeGenerator: Code generated in 28.980242 ms
+---------+--------+--------+----------+---+----+-------+
|firstname|lastname|  sports|medal_type|age|year|country|
+---------+--------+--------+----------+---+----+-------+
|     lisa|  cudrow|javellin|      gold| 34|2015|    USA|
|   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

18/05/26 10:09:42 INFO BlockManagerInfo: Removed broadcast_2_piece0 on 169.254
18/05/26 10:09:42 INFO ContextCleaner: Cleaned accumulator 48
```

Task 1.1 What are the total number of gold medal winners every year

Solution Approach -

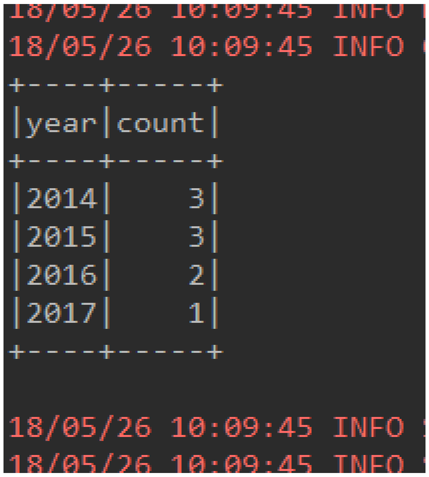
1. We have query the Sports Dataframe where medal_type is gold and group on year.
2. This will be achieved using filter , groupby and count operations of the Spark SQL.

Approach 1: Using SPARK SQL Operations → Filter , GroupBy and Count



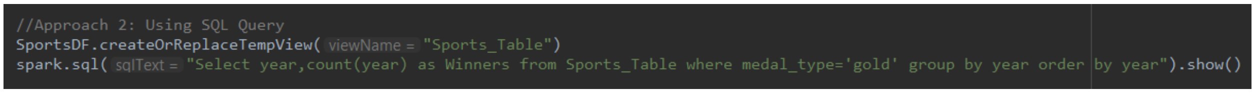
```
//Approach 1: Using Spark SQL Operations
SportsDF.filter( conditionExpr = "medal_type='gold'").groupBy( col1 = "year").count().orderBy( sortCol = "year").show()
```

Output



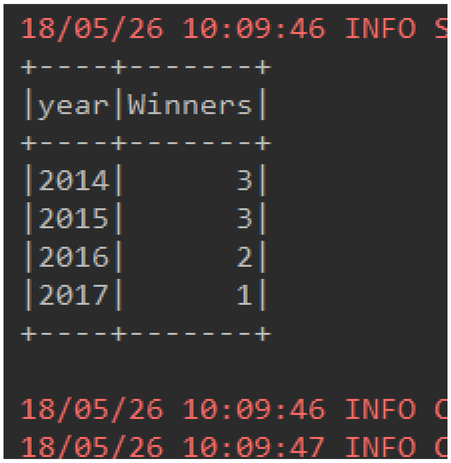
```
18/05/26 10:09:45 INFO
18/05/26 10:09:45 INFO
+----+-----+
|year|count|
+----+-----+
|2014|    3|
|2015|    3|
|2016|    2|
|2017|    1|
+----+-----+

18/05/26 10:09:45 INFO
18/05/26 10:09:45 INFO
```

Approach 2: Using SQL Queries



```
//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()
```

Output



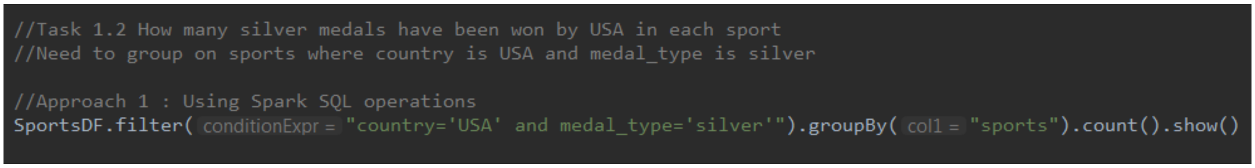
```
18/05/26 10:09:46 INFO S
+----+-------+
|year|Winners|
+----+-------+
|2014|      3|
|2015|      3|
|2016|      2|
|2017|      1|
+----+-------+

18/05/26 10:09:46 INFO C
18/05/26 10:09:47 INFO C
```

Task 1.2 How many silver medals have been won by USA in each sports ?

Solution Approach -

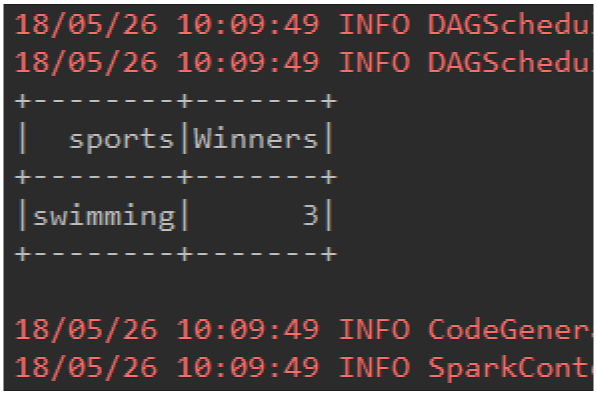
1. We have query the Sports Dataframe where country is USA , medal_type='silver' and group on sports.

Approach 1: Using SPARK SQL Operations → Filter , GroupBy and Count



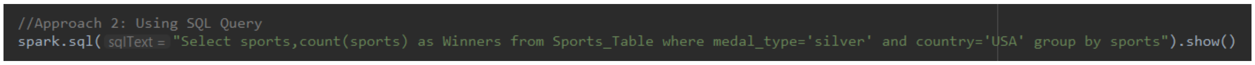
```
//Task 1.2 How many silver medals have been won by USA in each sport
//Need to group on sports where country is USA and medal_type is silver

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

Output



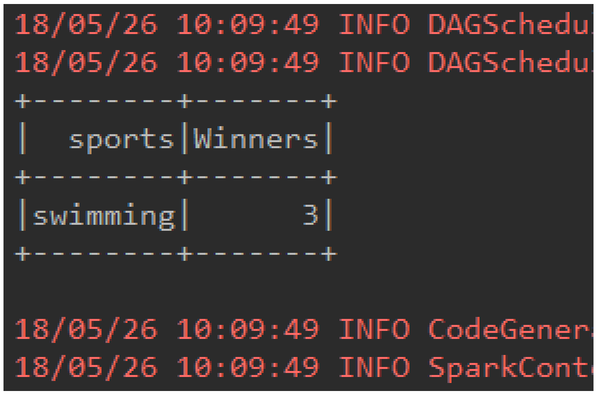
```
18/05/26 10:09:49 INFO DAGSchedu
18/05/26 10:09:49 INFO DAGSchedu
+--------+-------+
|  sports|Winners|
+--------+-------+
|swimming|      3|
+--------+-------+

18/05/26 10:09:49 INFO CodeGener
18/05/26 10:09:49 INFO SparkConte
```

Approach 2: Using SQL Queries



```
//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").show()
```

Output



```
18/05/26 10:09:49 INFO DAGSchedu
18/05/26 10:09:49 INFO DAGSchedu
+--------+-------+
|  sports|Winners|
+--------+-------+
|swimming|      3|
+--------+-------+

18/05/26 10:09:49 INFO CodeGener
18/05/26 10:09:49 INFO SparkConte
```

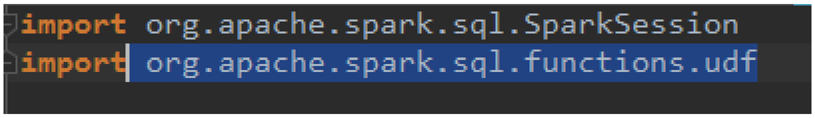
Task 2.1 Using udfs on dataframe
1. Change firstname, lastname columns into
Mr.first_two_letters_of_firstname<space>lastname
for example - michael, phelps becomes Mr.mi phelps

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.

Below are steps to create udfs in the Spark SQL

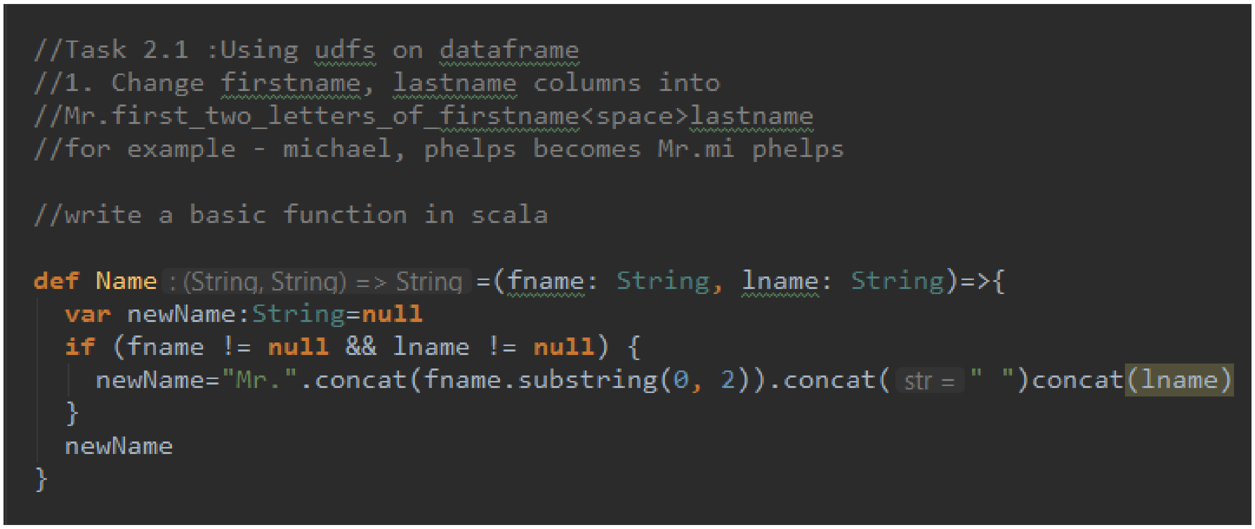
1. First we have to import namespace '*org.apache.spark.sql.functions.udf*' to extend the functionality / write the udfs.



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

2. Define a basic function scala which we would like perform the required functionality mentioned in task above. Here the function named as 'Name' is defined to accept two arguments first name and last name and returns the string output as asked.

Code

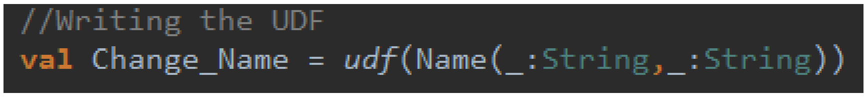


```
//Task 2.1 :Using udfs on dataframe
//1. Change firstname, lastname columns into
//Mr.first_two_letters_of_firstname<space>lastname
//for example - michael, phelps becomes Mr.mi phelps

//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
}
```

3. Once a basic function is created in scala we have can call this newly add method in Spark SQL as udf in two ways

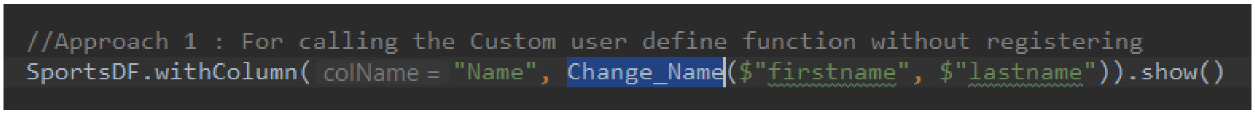
Approach 1: Create udf for above function in scala and use it with SPARK SQL Operations

Created udf – Code



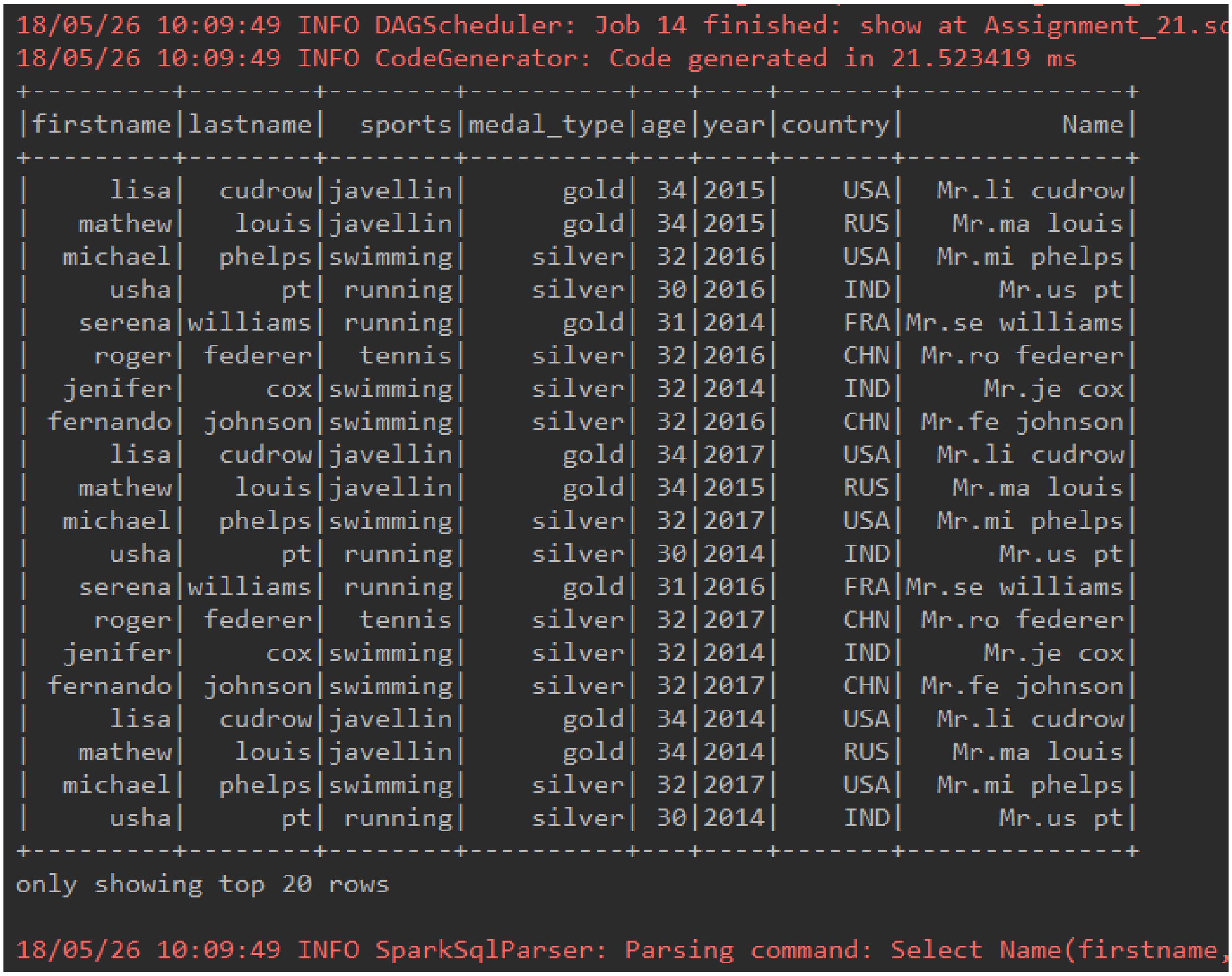
```
//Writing the UDF
val Change_Name = udf(Name(_:String,_:String))
```

Calling above created udf inside spark sql opearations



```
//Approach 1 : For calling the Custom user define function without registering
SportsDF.withColumn( colName = "Name", Change_Name($"firstname", $"lastname")).show()
```

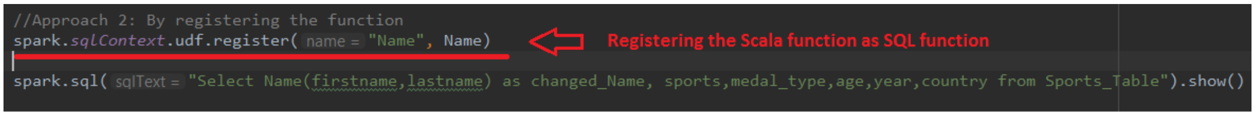
Output



```
18/05/26 10:09:49 INFO DAGScheduler: Job 14 finished: show at Assignment_21.sc
18/05/26 10:09:49 INFO CodeGenerator: Code generated in 21.523419 ms
+---------+--------+--------+----------+---+----+-------+--------------+
|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

18/05/26 10:09:49 INFO SparkSqlParser: Parsing command: Select Name(firstname,
```

Approach 2:

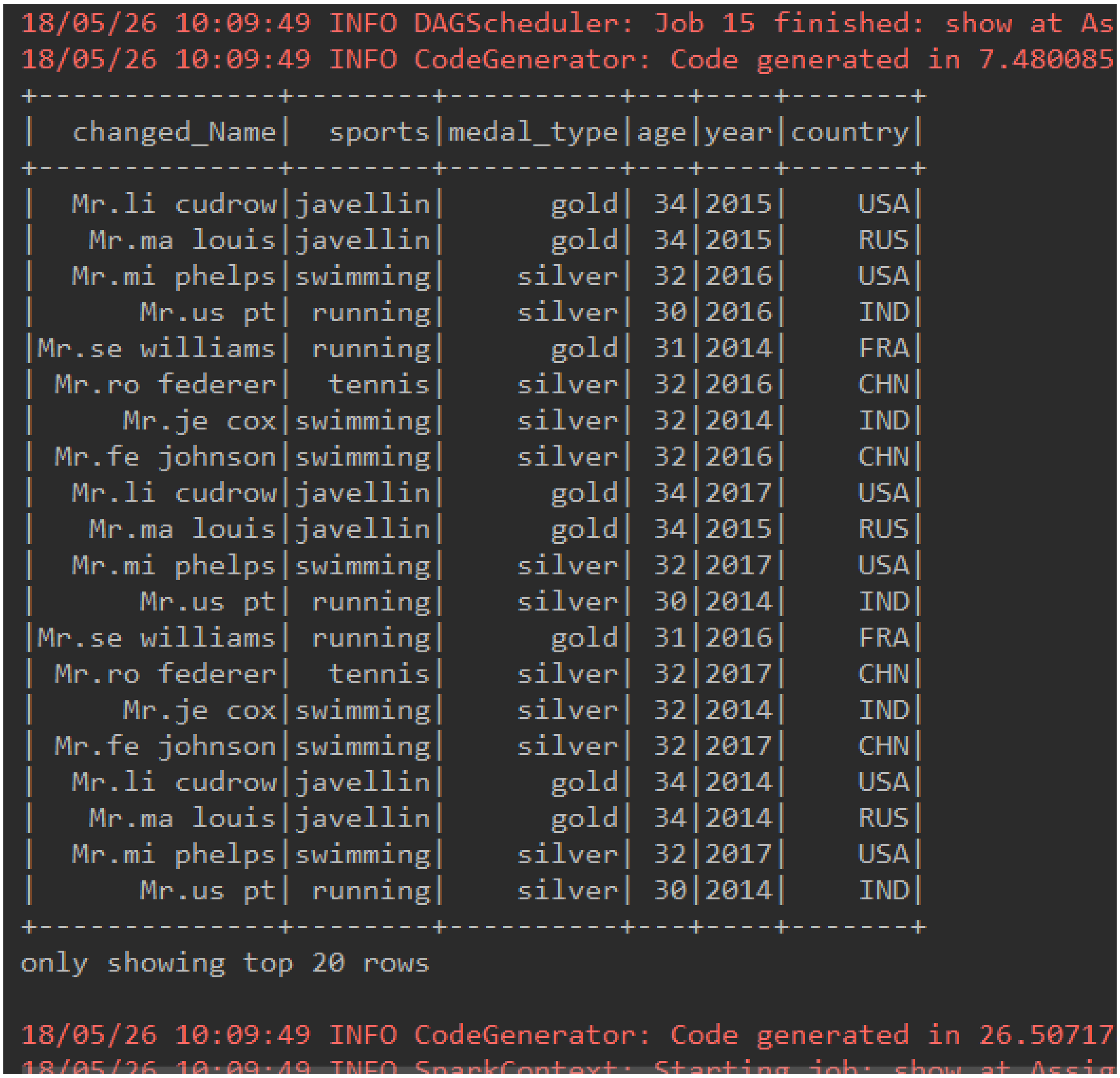
By registering the udf so that it can be used wih sql queries as well.



```
//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()
```

Registering the Scala function as SQL function

Output

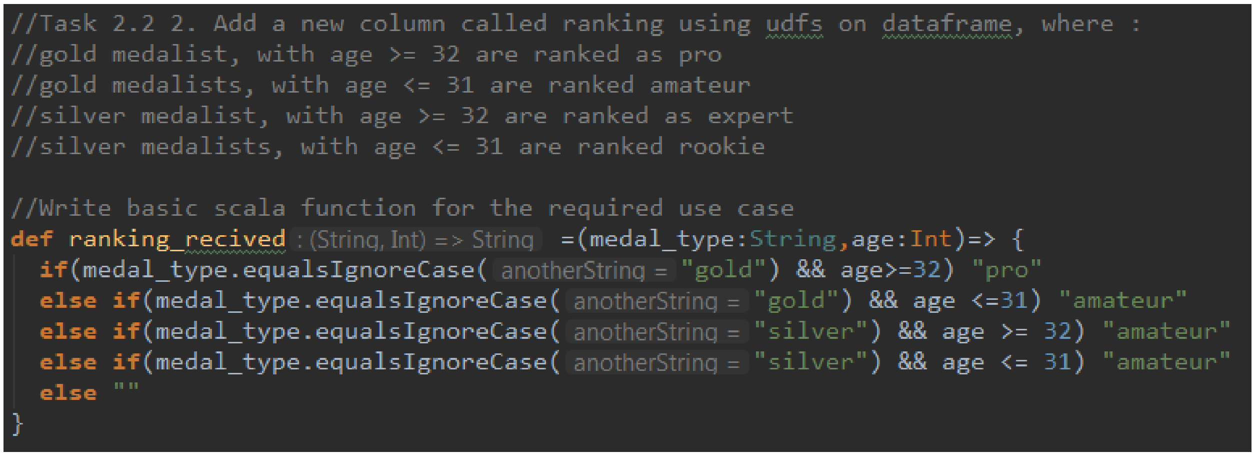


```
18/05/26 10:09:49 INFO DAGScheduler: Job 15 finished: show at As
18/05/26 10:09:49 INFO CodeGenerator: Code generated in 7.480085
+--------------+--------+----------+---+----+-------+
|  changed_Name|  sports|medal_type|age|year|country|
+--------------+--------+----------+---+----+-------+
|  Mr.li 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

18/05/26 10:09:49 INFO CodeGenerator: Code generated in 26.50717
18/05/26 10:09:49 INFO SparkContext: Starting job: show at Assig
```

Task 2.2 Using udfs on dataframe

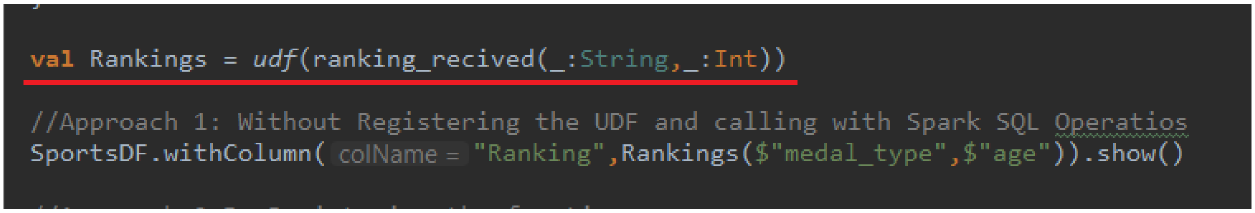
Add a new column called ranking using udfs on dataframe, where :
gold medalist, with age >= 32 are ranked as pro
gold medalists, with age <= 31 are ranked amateur
silver medalist, with age >= 32 are ranked as expert
silver medalists, with age <= 31 are ranked rookie

Basic scala function to perform the required above task



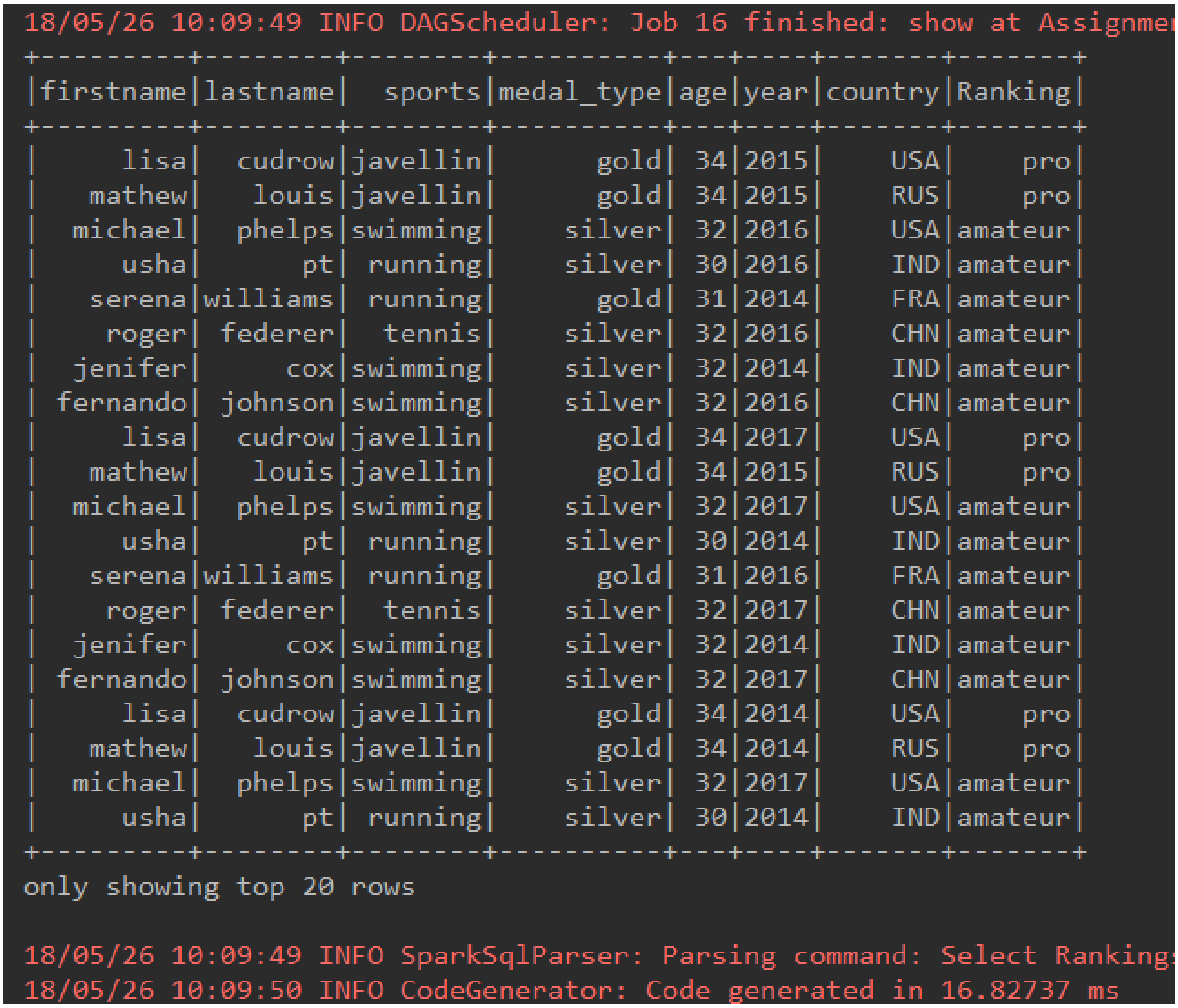
```
//Task 2.2 2. Add a new column called ranking using udfs on dataframe, where :
//gold medalist, with age >= 32 are ranked as pro
//gold medalists, with age <= 31 are ranked amateur
//silver medalist, with age >= 32 are ranked as expert
//silver medalists, with age <= 31 are ranked rookie

//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 ""
}
```

Approach 1: Create udf for above function in scala and use it with SPARK SQL Operations



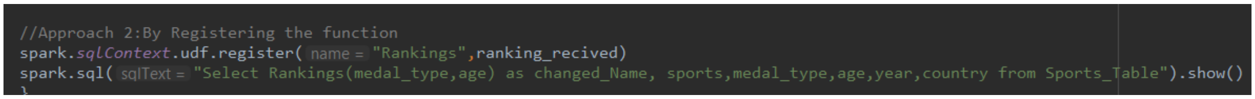
```
 val Rankings = udf(ranking_recived(_:String,_:Int))

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

Output



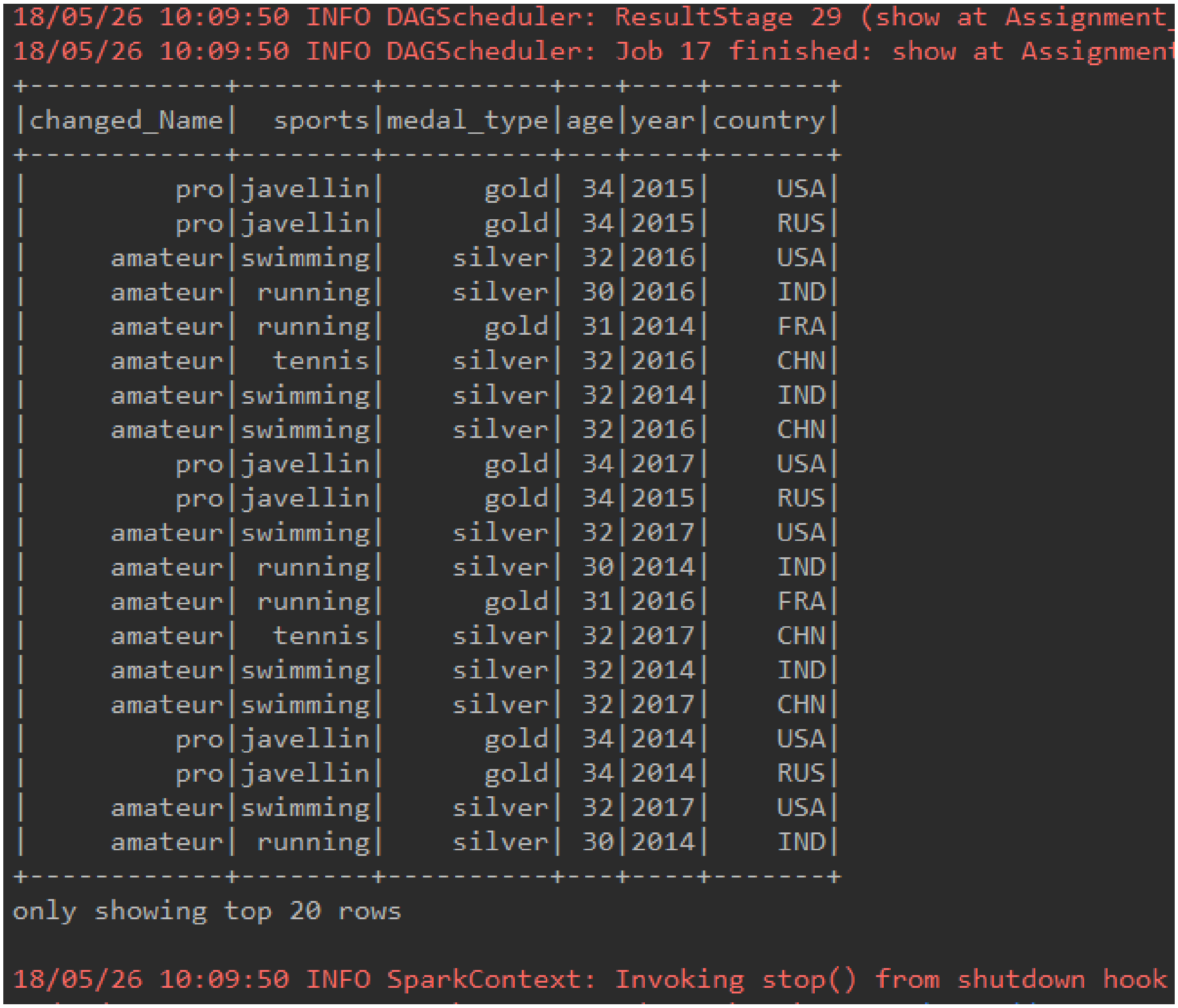
```
18/05/26 10:09:49 INFO DAGScheduler: Job 16 finished: show at Assignmen
+---------+--------+--------+----------+---+----+-------+-------+
|firstname|lastname|  sports|medal_type|age|year|country|Ranking|
+---------+--------+--------+----------+---+----+-------+-------+
|     lisa|  cudrow|javellin|      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

18/05/26 10:09:49 INFO SparkSqlParser: Parsing command: Select Ranking
18/05/26 10:09:50 INFO CodeGenerator: Code generated in 16.82737 ms
```

Approach 2: By registering the udf so that it can be used wih sql queries as well.



```
//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()
}
```

Output



```
18/05/26 10:09:50 INFO DAGScheduler: ResultStage 29 (show at Assignment_
18/05/26 10:09:50 INFO DAGScheduler: Job 17 finished: show at 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

18/05/26 10:09:50 INFO SparkContext: Invoking stop() from shutdown hook
```