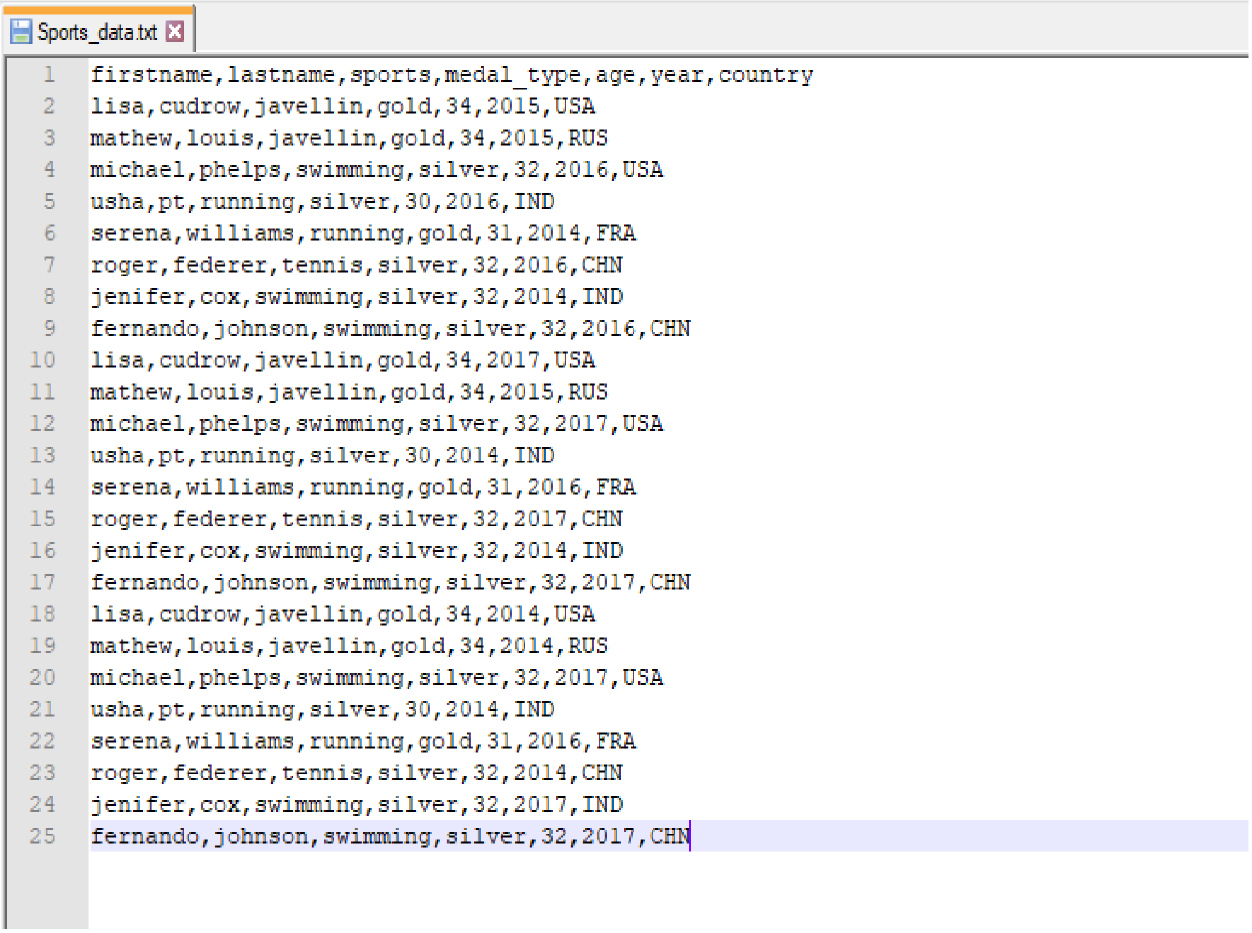
ASSIGNMENT 19.2

Input Dataset:

We have sports related data with us:

Sports_data

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
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
```

Problem Statement:

Using UDFs on data frame:

1. Change firstname, lastname columns into

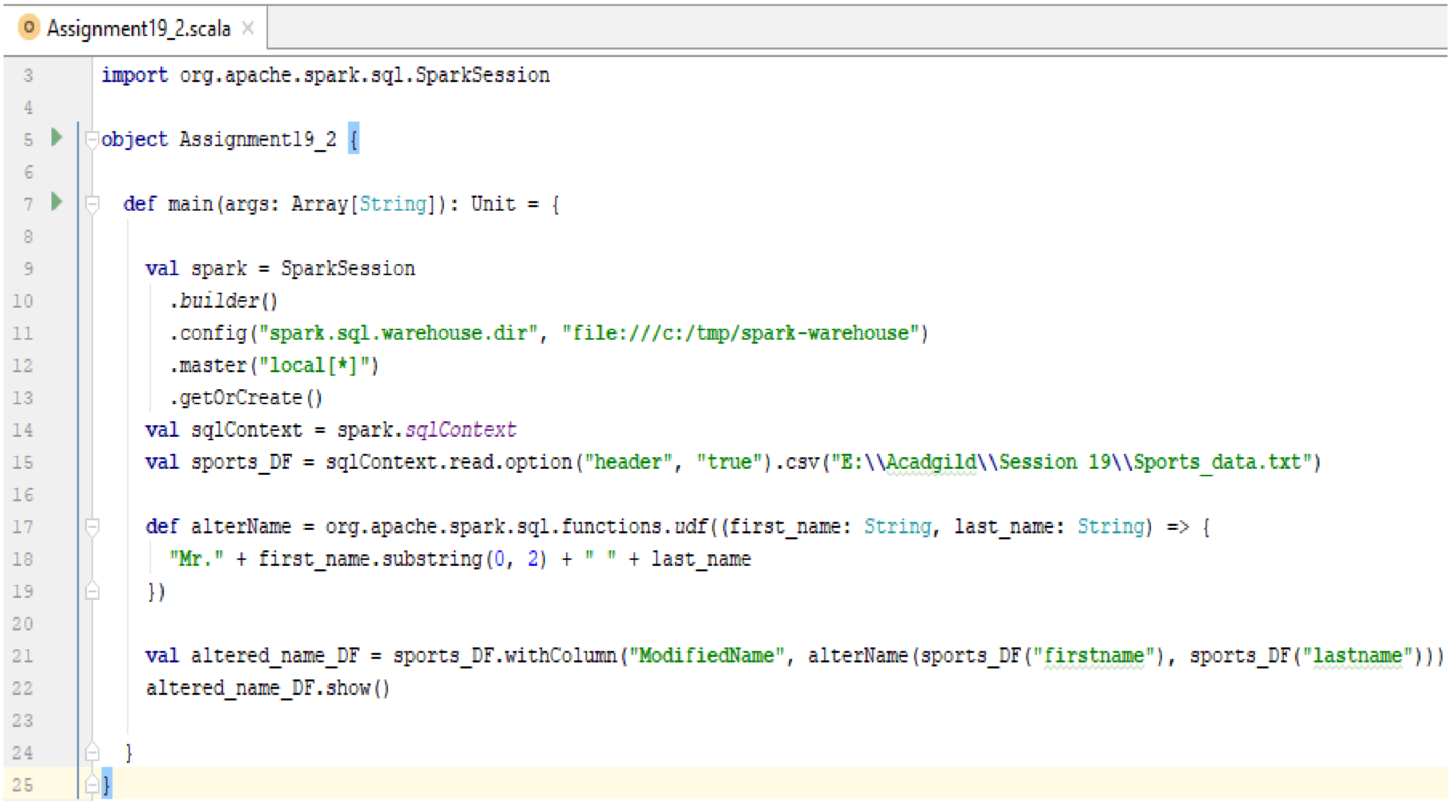
Mr.first_two_letters_of_firstname<space>lastname

for example - michael, phelps becomes Mr.mi phelps

Solution:

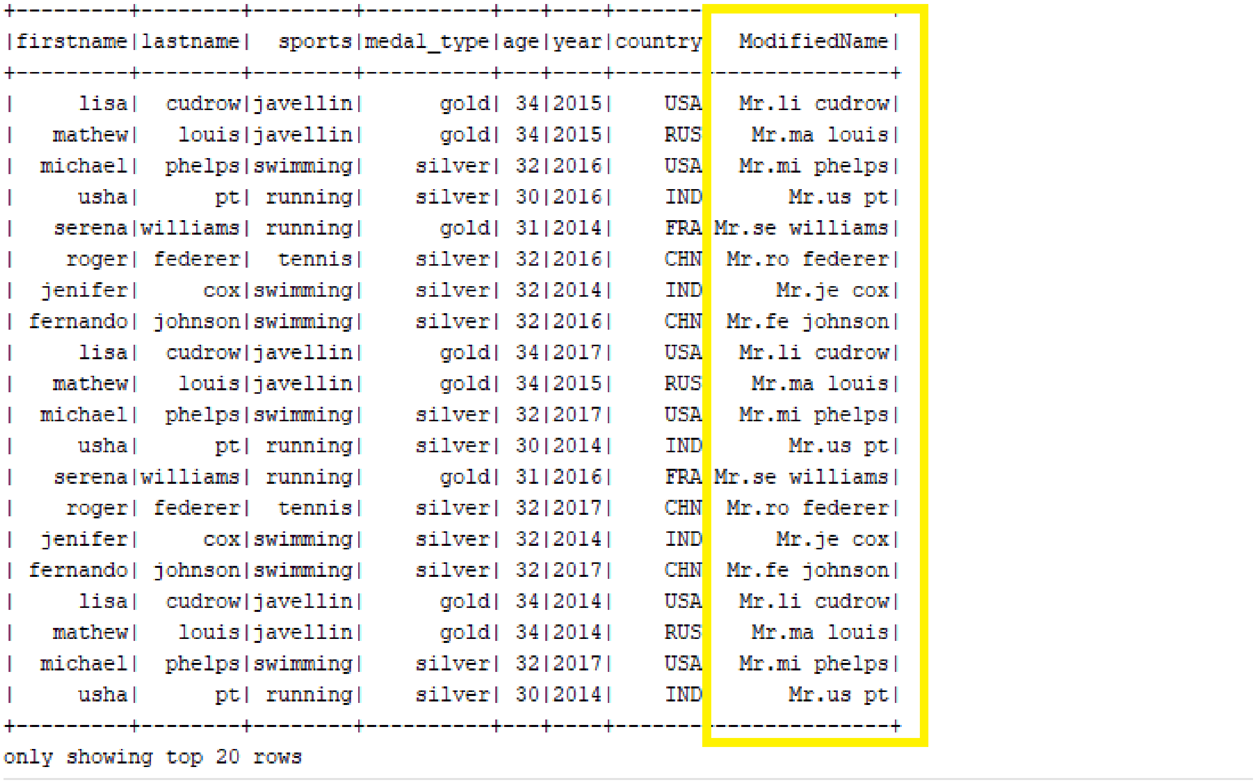
1. Here is the Spark code snippet to create UDF and its application as per the problem description:

```
// import required Spark packages
import org.apache.spark.sql.SparkSession

object Assignment19_2 {
  def main(args: Array[String]): Unit = {
// create a SparkSession object that can be used to create various contexts of Spark such as sqlContext
    val spark = SparkSession
      .builder()
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()

// initialize sqlContext
    val sqlContext = spark.sqlContext

// load input data file – Sports_data.txt
    val sports_DF = sqlContext.read.option("header", "true").csv("E:\\Acadgild\\Session 19\\Sports_data.txt")

// User Defined Function (UDF) to modify naming convention in the dataset as per instructions
def alterName = org.apache.spark.sql.functions.udf((first_name: String, last_name: String) => {

      "Mr." + first_name.substring(0, 2) + " " + last_name
    })
// call the UDF, pass first name and last name as arguments, create new column ‘ModifiedName’

// which will hold the new modified name

    val altered_name_DF = sports_DF.withColumn("ModifiedName", alterName(sports_DF("firstname"), sports_DF("lastname")))

// show the dataset with new column and its associated values on the console

    altered_name_DF.show()
```



```scala
import org.apache.spark.sql.SparkSession

object Assignment19_2 {

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

    val spark = SparkSession
      .builder()
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()
    val sqlContext = spark.sqlContext
    val sports_DF = sqlContext.read.option("header", "true").csv("E:\\Acadgild\\Session 19\\Sports_data.txt")

    def alterName = org.apache.spark.sql.functions.udf((first_name: String, last_name: String) => {
      "Mr." + first_name.substring(0, 2) + " " + last_name
    })

    val altered_name_DF = sports_DF.withColumn("ModifiedName", alterName(sports_DF("firstname"), sports_DF("lastname")))
    altered_name_DF.show()

  }
}
```

Output:



```
+---------+--------+--------+----------+---+----+-------+--------------+
|firstname|lastname|  sports|medal_type|age|year|country|  ModifiedName|
+---------+--------+--------+----------+---+----+-------+--------------+
|     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
```

2. Add a new column called ranking using UDFs on data frame, 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

Spark code in Scala:

```
// import required Spark packages
import org.apache.spark.sql.SparkSession

object Assignment19_2 {
  def main(args: Array[String]): Unit = {
// create a SparkSession object that can be used to create various contexts of Spark such as sqlContext
    val spark = SparkSession
      .builder()
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()

// initialize sqlContext
    val sqlContext = spark.sqlContext

// load input data file – Sports_data.txt
    val sports_DF = sqlContext.read.option("header", "true").csv("E:\\Acadgild\\Session 19\\Sports_data.txt")

// User Defined Function (UDF) to find ranking of a medalist based on his/her age and medal won
def computeRanking = org.apache.spark.sql.functions.udf((medal_type: String, age: Int) => {

      if(medal_type == "gold" && age >= 32 )

        "pro"

      else if(medal_type == "gold" && age <= 31)

        "amateur"
      else if(medal_type == "silver" && age >= 32)

        "expert"

      else if(medal_type == "silver" && age <= 31)

        "rookie"

    })
```

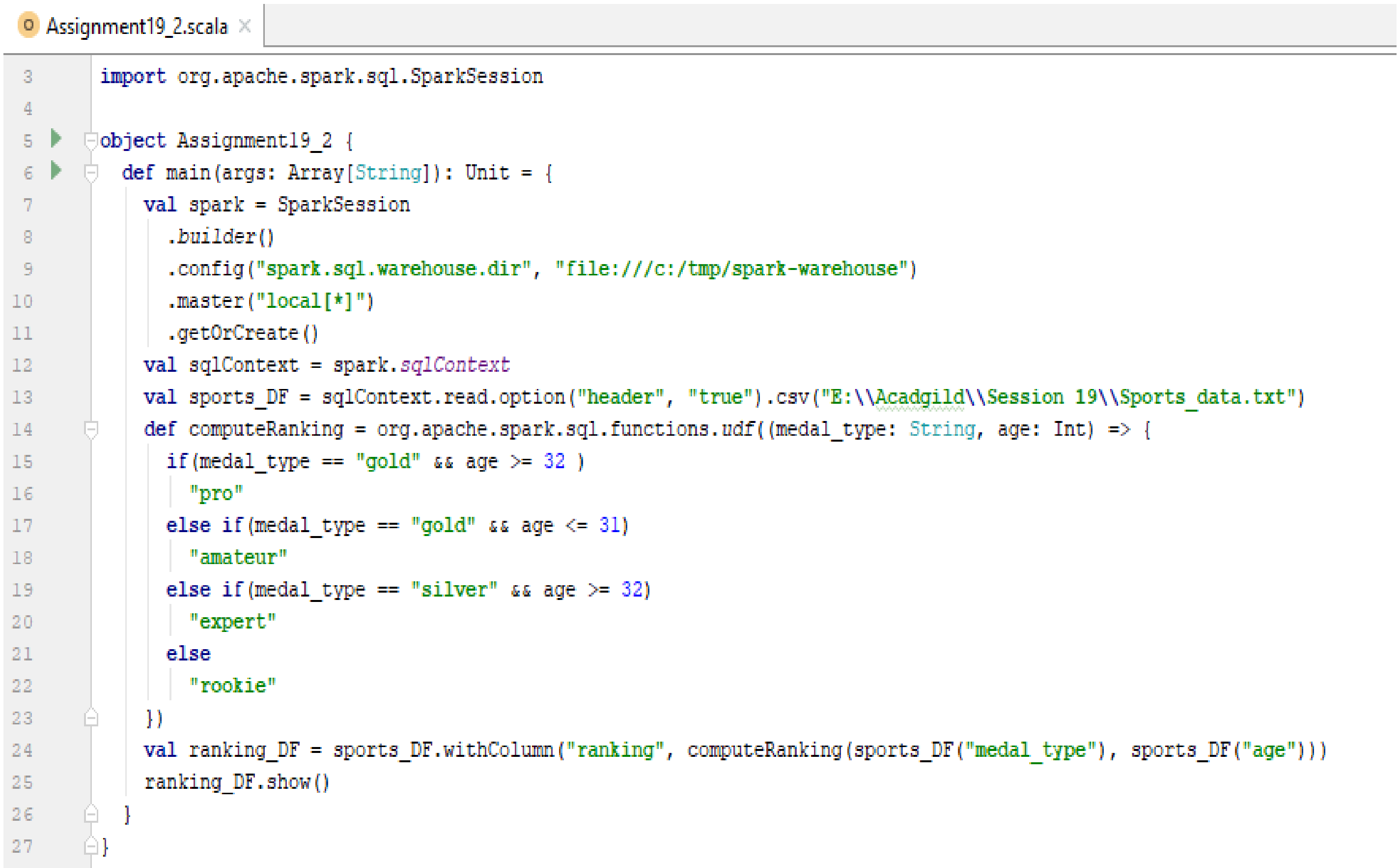
// call the UDF, pass medal type and age as arguments, create new column ‘ranking’ that stores

// value returned by UDF for each record

```
    val ranking_DF = sports_DF.withColumn("ranking", computeRanking (sports_DF ("medal_type"), sports_DF("age")))
```

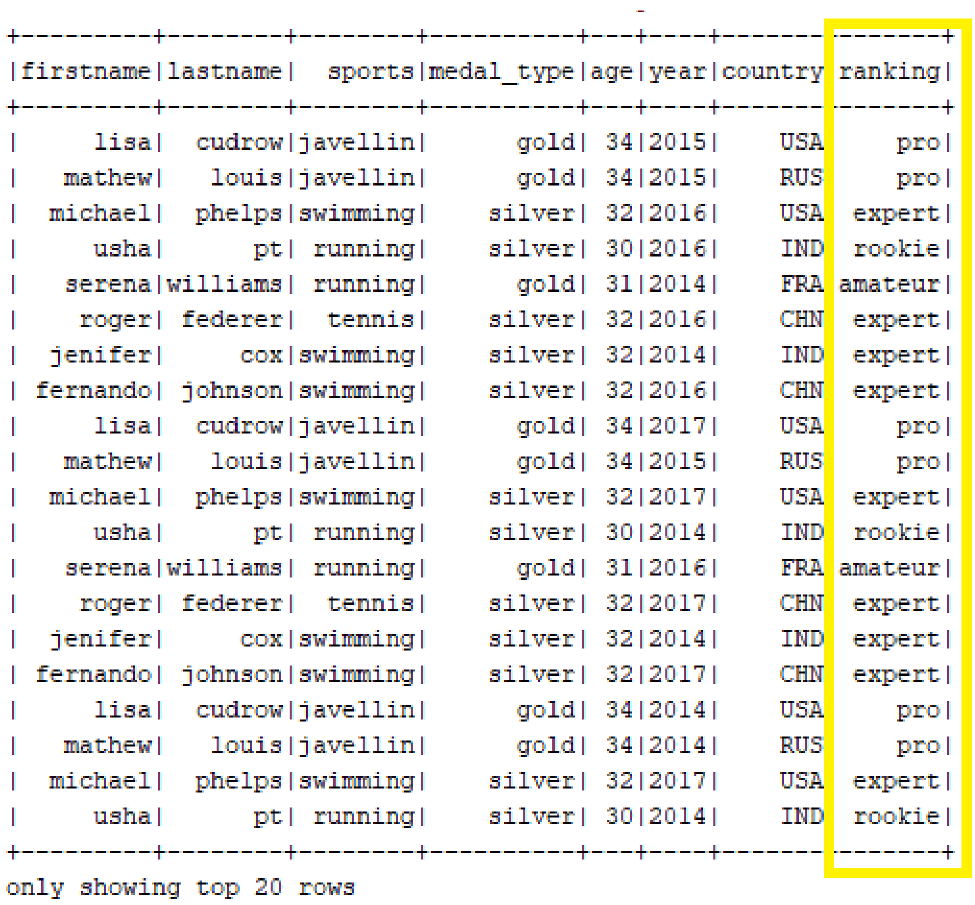
// show the dataset with new column and its associated values on the console

```
    ranking_DF.show()
```



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

object Assignment19_2 {
  def main(args: Array[String]): Unit = {
    val spark = SparkSession
      .builder()
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()
    val sqlContext = spark.sqlContext
    val sports_DF = sqlContext.read.option("header", "true").csv("E:\\Acadgild\\Session 19\\Sports_data.txt")
    def computeRanking = org.apache.spark.sql.functions.udf((medal_type: String, age: Int) => {
      if(medal_type == "gold" && age >= 32 )
        "pro"
      else if(medal_type == "gold" && age <= 31)
        "amateur"
      else if(medal_type == "silver" && age >= 32)
        "expert"
      else
        "rookie"
    })
    val ranking_DF = sports_DF.withColumn("ranking", computeRanking(sports_DF("medal_type"), sports_DF("age")))
    ranking_DF.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  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
```