Assignment 19.2:

Problem Statement:

Initial Steps:

val sports_data_with_header = sc.textFile("/home/acadgild/spark/Sports_data.txt")

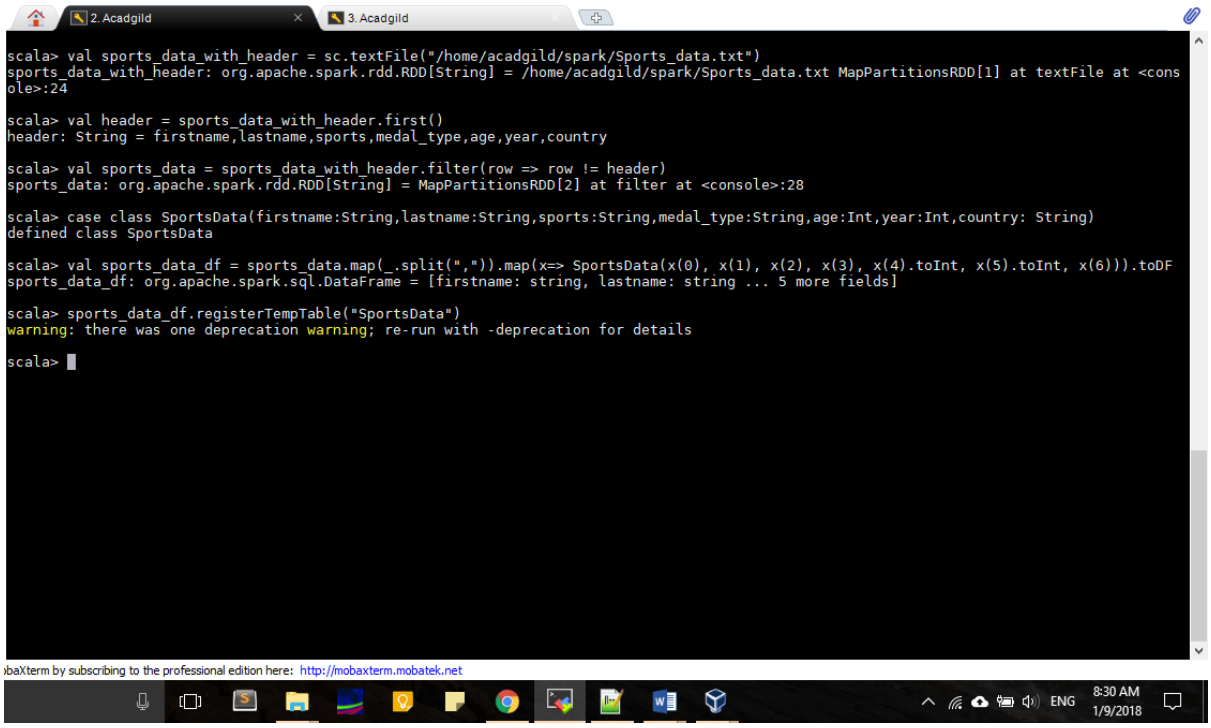
val header = sports_data_with_header.first()

val sports_data = sports_data_with_header.filter(row => row != header)

case class SportsData(firstname:String,lastname:String,sports:String,medal_type:String,age:Int,year:Int,country : String)

val sports_data_df = sports_data.map(_.split(",")).map(x=> SportsData(x(0), x(1), x(2), x(3), x(4).toInt, x(5).toInt, x(6))).toDF

sports_data_df.registerTempTable("SportsData")



1. Change firstname, lastname columns into Mr.first_two_letters_of_firstname<space>lastname

for example - michael, phelps becomes Mr.mi phelps

Step1: Create a UDF fullName which returns firstname,lastname column into Mr.first_two_letters_of_first name<space>lastname

Code is as below:

def fullName = org.apache.spark.sql.functions.udf((x:String, y:String) => "Mr." + x.substring(0,2) + " " + y)

Step2: Add a column fullName to sports_data_df and call the UDF fullName as created in step1 and taking arguments value of firstname and lastname and show the result

Code is as below:

sports_data_df.withColumn("fullName", fullName(sports_data_df("firstname"), sports_data_df("lastname"))).show()

```
scala> def fullName = org.apache.spark.sql.functions.udf((x:String, y:String) => "Mr." + x.substring(0,2) + " " + y)
fullName: org.apache.spark.sql.expressions.UserDefinedFunction

scala> sports_data_df.withColumn("fullName", fullName(sports_data_df("firstname"), sports_data_df("lastname"))).show()
+---------+--------+--------+----------+---+----+-------+--------------+
|firstname|lastname|  sports|medal_type|age|year|country|      fullName|
+---------+--------+--------+----------+---+----+-------+--------------+
|     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


scala>
```

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

Step1: Create a UDF findRanking which take two parameters x represenring medal_type and y representing age. Based on value of medal_type and age, following values are returned “pro”, “amateur”, “expert”, “”

Code is as below:

```
def findRanking = org.apache.spark.sql.functions.udf((x:String, y:Int) => {
                if (x == "gold" && y >=32) "pro"
                else if (x == "gold" && y <=31) "amateur"
                else if (x == "silver" && y >= 32) "expert"
                else if (x == "silver" && y <= 31) "rookie"
                else ""
        })
```

Step2: Add a column ranking to sports_data_df and call the UDF findRanking as created in step1 taking arguments, value of medal_type, age and show the result

Code is as below:

sports_data_df.withColumn("ranking", findRanking(sports_data_df("medal_type"), sports_data_df("age"))).show()