SPARK SQL ASSIGNMENT 19.2

PROBLEM STATEMENT:

Using udfs on dataframe

19.2.1. Change firstname, lastname columns into

Mr.first_two_letters_of_firstname<space>lastname

for example - michael, phelps becomes Mr.miphelps

19.2.2. 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

==========

//Schema Definition for input test file Sports_data.txt

Sports_data.txt ->

firstname:String,

lastname:String,

sports:String,

medal_type:String,

age:Integer,

year:Integer,

country:String

==========

//moving to Downloads folder

[acadgild@localhost]$ cd Downloads

//Browsing the files present in directory

[acadgild@localhostDownloads]$ ls

//putting the file to HDFS from local filesystem

[acadgild@localhostDownloads]$ hadoop fs -put Sports_data.txt /user/acadgild/spark/

//Browsing the contents of file kept at HDFS

[acadgild@localhostDownloads]$ hadoop fs -cat /user/acadgild/spark/Sports_data.txt

//Initiating the spark shell

[acadgild@localhostDownloads]$ spark-shell

//Importing the spark sql packages

import org.apache.spark.sql._

import sqlContext.implicits._

import org.apache.spark.sql.functions

//conversion of text file into RDD with the help of SPARK CONTEXT object.

valsportsRDD = sc.textFile("hdfs://localhost:9000//user/acadgild/spark/Sports_data.txt")

//putting the first line as a header from the RDD

val header = sportsRDD.first()

//Removing the header from RDD for the purpose of querying data

valsdRDD =sportsRDD.filter(firstrecord => (firstrecord != header))

//printing the data of rdd without the header

sdRDD.foreach(println)

//sportsdata class definition for defining the schema of class

case class

sportsdata(firstname:String,

lastname:String,

sports:String,

medal_type:String,

age:Integer,

year:Integer,

country:String)

//mapping the data with comma delimiter and mapping it with the schema of sportsdata case class and creating the dataframe after map operation

val rec = sdRDD.map(x=>x.split(",")).map(x=>sportsdata(x(0),x(1),x(2),x(3),x(4).toInt,x(5).toInt,x(6))).toDF

//registering the temporary table named as sports for query execution

rec.registerTempTable("sports")

==========

Problem Statement 19.2.1

Using udfs on dataframe , Change firstname, lastname columns into

Mr.first_two_letters_of_firstname<space>lastname

for example - michael, phelps becomes Mr.miphelps

Input Commands:

def combinefirstlast=org.apache.spark.sql.functions.udf((firstname:String,lastname:String)=>{"Mr."+firstname +" "+lastname})

val newname = rec.withColumn("newfirstlast",combinefirstlast(substring(rec("firstname"),1,2),rec("lastname")))

newname.show()

newname.select("newfirstlast").show()

Screenshots:

```
scala> import org.apache.spark.sql._
import org.apache.spark.sql._

scala> import sqlContext.implicits._
import sqlContext.implicits._

scala> import org.apache.spark.sql.functions
import org.apache.spark.sql.functions

scala> val sportsRDD = sc.textFile("hdfs://localhost:9000//user/acadgild/spark/Sports_data.txt")
sportsRDD: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[114] at textFile at <console>:65

scala> val header = sportsRDD.first()
header: String = firstname,lastname,sports,medal_type,age,year,country

scala> val sdRDD  = sportsRDD.filter(firstrecord => (firstrecord != header))
sdRDD: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[115] at filter at <console>:69

scala> case class
     | sportsdata(firstname:String,
     | lastname:String,
     | sports:String,
     | medal_type:String,
     | age:Integer,
     | year:Integer,
     | country:String)
defined class sportsdata

scala> val rec = sdRDD.map(x=> x.split(",")).map(x=>sportsdata(x(0),x(1),x(2),x(3),x(4).toInt,x(5).t
oInt,x(6))).toDF
rec: org.apache.spark.sql.DataFrame = [firstname: string, lastname: string, sports: string, medal_ty
pe: string, age: int, year: int, country: string]

scala> rec.registerTempTable("sports")
```

```
scala> def combinefirstlast=org.apache.spark.sql.functions.udf((firstname:String,lastname:String)=>{
"Mr."+firstname +" "+lastname})
combinefirstlast: org.apache.spark.sql.UserDefinedFunction

scala> val newname = rec.withColumn("newfirstlast",combinefirstlast(substring(rec("firstname"),1,2),
rec("lastname")))
newname: org.apache.spark.sql.DataFrame = [firstname: string, lastname: string, sports: string, meda
l_type: string, age: int, year: int, country: string, newfirstlast: string]

scala> newname.select("newfirstlast").show()
+--------------+
|  newfirstlast|
+--------------+
|  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|
|      Mr.us pt|
+--------------+
only showing top 20 rows
```

==========

Problem Statement19.2.2.

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

Input Commands:

//defining the user defined UDF and passing age and medal type to determine the ranking value and return the same.

```
def
findRankUDF=org.apache.spark.sql.functions.udf((age:Integer,medal_type:St
ring)=>{
    if(age>=32 &&medal_type == "gold") "pro"
else if(age<=31 &&medal_type == "gold") "amateur"
else if(age>=32 &&medal_type == "silver") "expert"
else if(age<=31 &&medal_type == "silver") "rookie"
else "No Ranking defined"})
```

//creating a new column ranking and putting value as determined by the return value of user defined function where age and medal type are passed.

```
valrankdf =
rec.withColumn("ranking",findRankUDF(rec("age"),rec("medal_type")))
rankdf.show()
```

Screenshots:

```
scala> def findRankUDF=org.apache.spark.sql.functions.udf((age:Integer,medal_type:String)=>{
     |      if(age>=32 && medal_type == "gold") "pro"
     | else if(age<=31 && medal_type == "gold") "amateur"
     | else if(age>=32 && medal_type == "silver") "expert"
     | else if(age<=31 && medal_type == "silver") "rookie"
     | else "No Ranking defined"})
findRankUDF: org.apache.spark.sql.UserDefinedFunction

scala> val rankdf = rec.withColumn("ranking",findRankUDF(rec("age"),rec("medal_type")))
rankdf: org.apache.spark.sql.DataFrame = [firstname: string, lastname: string, sports: string, medal
_type: string, age: int, year: int, country: string, ranking: string]
```

```
scala> rankdf.show()
+---------+--------+--------+----------+---+----+-------+-------+
|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
```