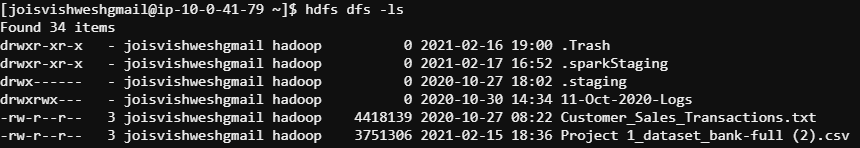
Market Analysis in Banking Domain

Domain: Banking (Market Analysis)

**Background and Objective:**

Your client, a Portuguese banking institution, ran a marketing campaign to convince potential customers to invest in a bank term deposit scheme.   
The marketing campaigns were based on phone calls. Often, the same customer was contacted more than once through phone, in order to assess if they would want to subscribe to the bank term deposit or not. You have to perform the marketing analysis of the data generated by this campaign.

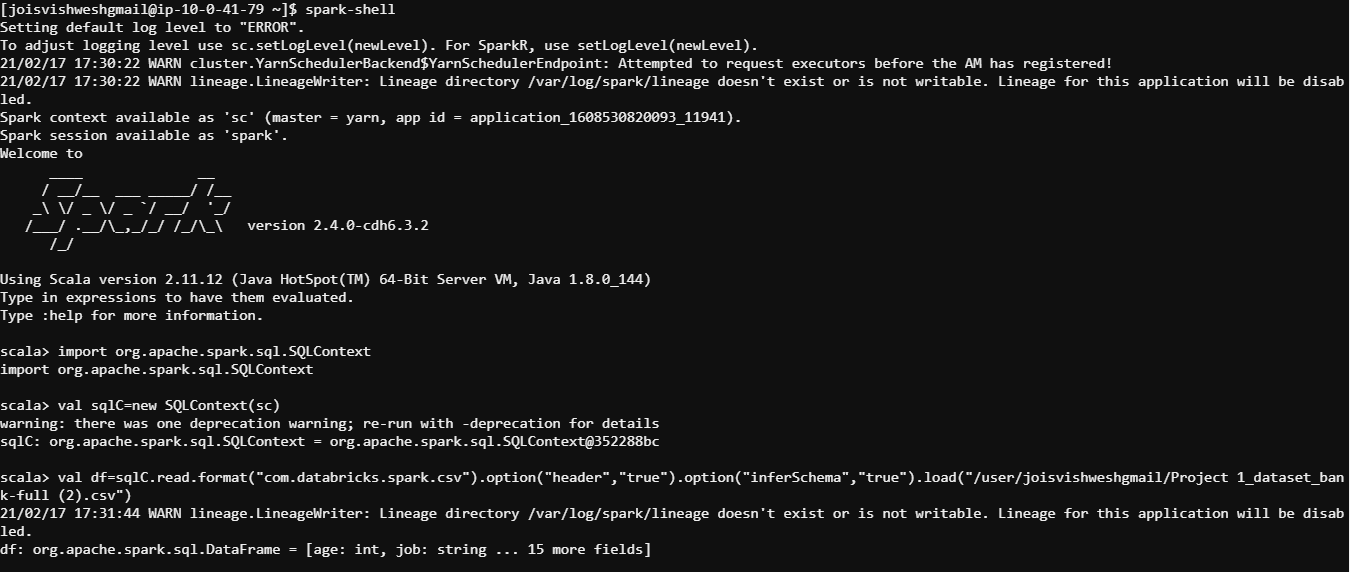
**Source Code:**

* + Load data and create a Spark data frame

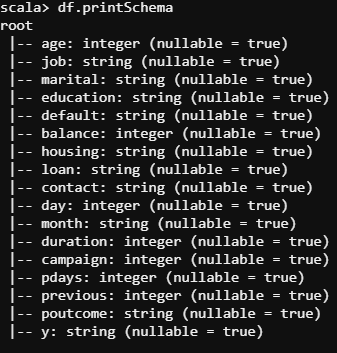
import org.apache.spark.sql.SQLContext

val sqlC=new SQLContext(sc)

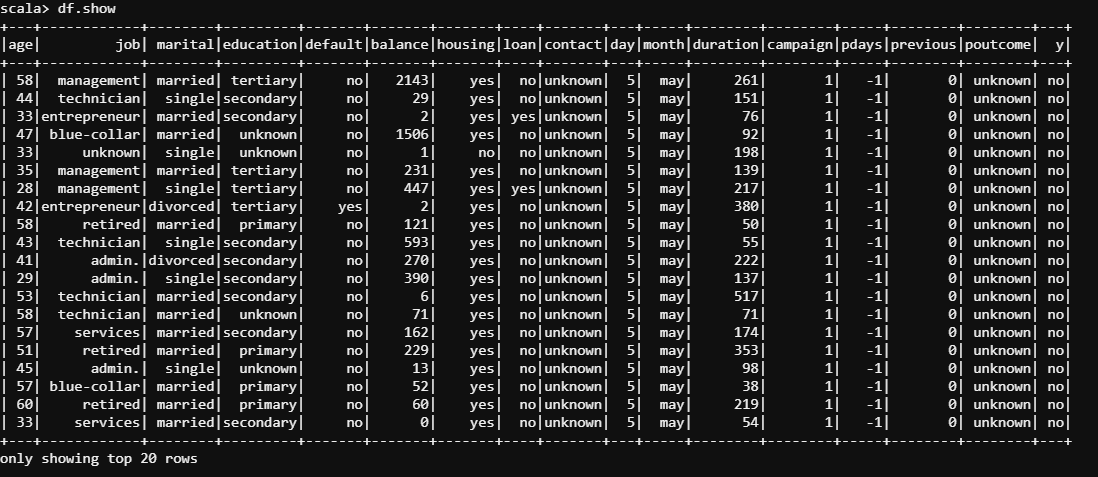
val df = sqlC.read.format("com.databricks.spark.csv").option("header","true").option("inferSchema","true").load("/user/joisvishweshgmail/Project 1\_dataset\_bank-full (2).csv")



df.printSchema

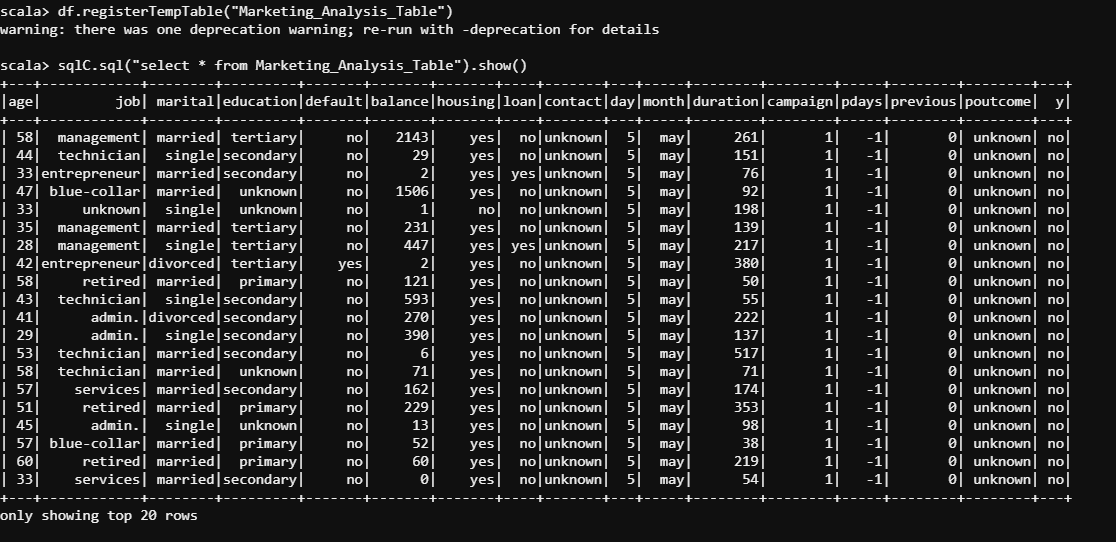


df.show



df.registerTempTable("Marketing\_Analysis\_Table")

sqlC.sql("select \* from Marketing\_Analysis\_Table").show()

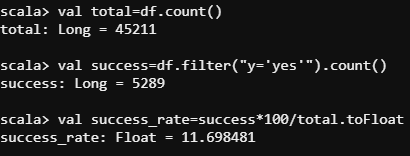


* Give marketing success rate (No. of people subscribed / total no. of entries)

val total=df.count()

val success=df.filter("y='yes'").count()

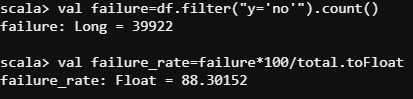
val success\_rate=success\*100/total.toFloat



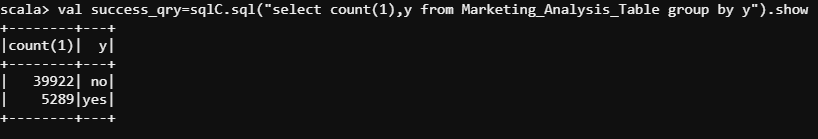
* Give marketing failure rate

val failure=df.filter("y='no'").count()

val failure\_rate=failure\*100/total.toFloat

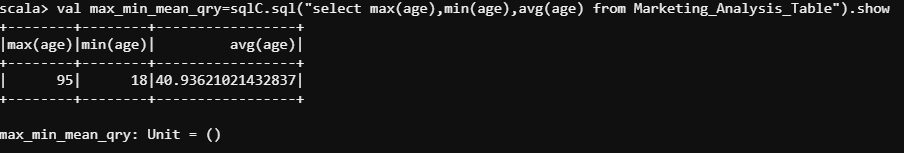


val success\_qry=sqlC.sql("select count(1),y from Marketing\_Analysis\_Table group by y").show



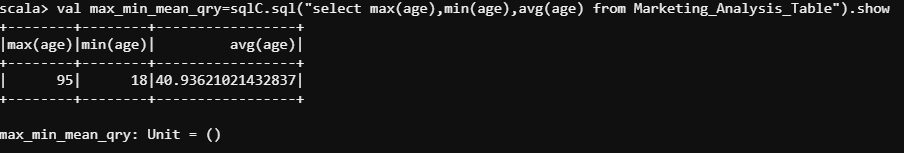
* Give the maximum, mean, and minimum age of the average targeted customer

val max\_min\_mean\_qry=sqlC.sql("select max(age),min(age),avg(age) from Marketing\_Analysis\_Table").show



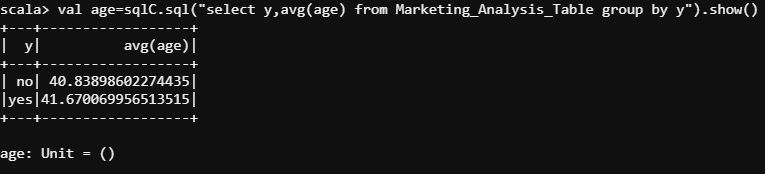
* Check the quality of customers by checking average balance, median balance of customers

val quality=sqlC.sql("select avg(balance),percentile\_approx(balance,0.5) from Marketing\_Analysis\_Table").show()



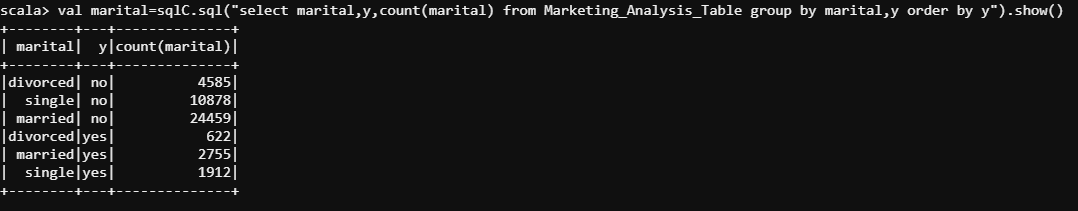
* Check if age matters in marketing subscription for deposit

val age=sqlC.sql("select y,avg(age) from Marketing\_Analysis\_Table group by y").show()



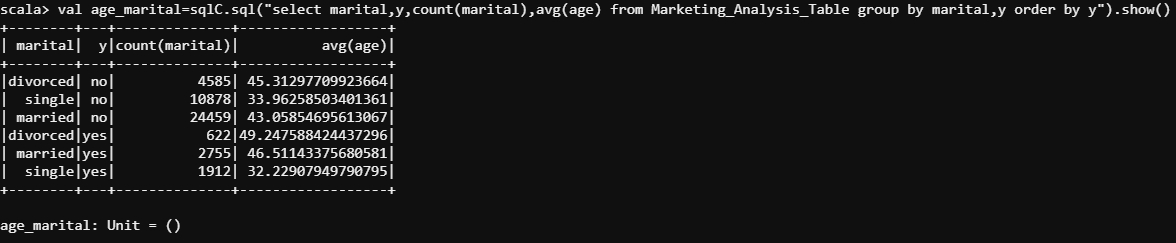
* Check if marital status mattered for a subscription to deposit

val marital=sqlC.sql("select marital,y,count(marital) from Marketing\_Analysis\_Table group by marital,y order by y").show()



* Check if age and marital status together mattered for a subscription to deposit scheme

val age\_marital=sqlC.sql("select marital,y,count(marital),avg(age) from Marketing\_Analysis\_Table group by marital,y order by y").show()



* Do feature engineering for the bank and find the right age effect on the campaign.

val age\_group=sqlC.sql("select 'Output','Age','Count' from Marketing\_Analysis\_Table where 1=0 union select y,'Teen',Count(y) from Marketing\_Analysis\_Table where y='yes' and age<20 group by y union select y,'Young',Count(y) from Marketing\_Analysis\_Table where y='yes' and age>=20 and age<30 group by y union select y,'Middle Aged',Count(y) from Marketing\_Analysis\_Table where y='yes' and age>=30 and age<45 group by y union select y,'Senior',Count(y) from Marketing\_Analysis\_Table where y='yes' and age>=45 and age<60 group by y union select y,'Old',Count(y) from Marketing\_Analysis\_Table where y='yes' and age<=60 group by y").show()

