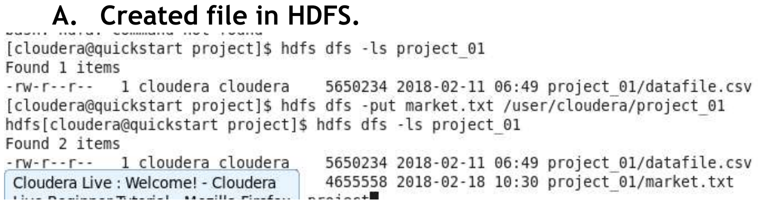
Project-1: Marketing Analysis

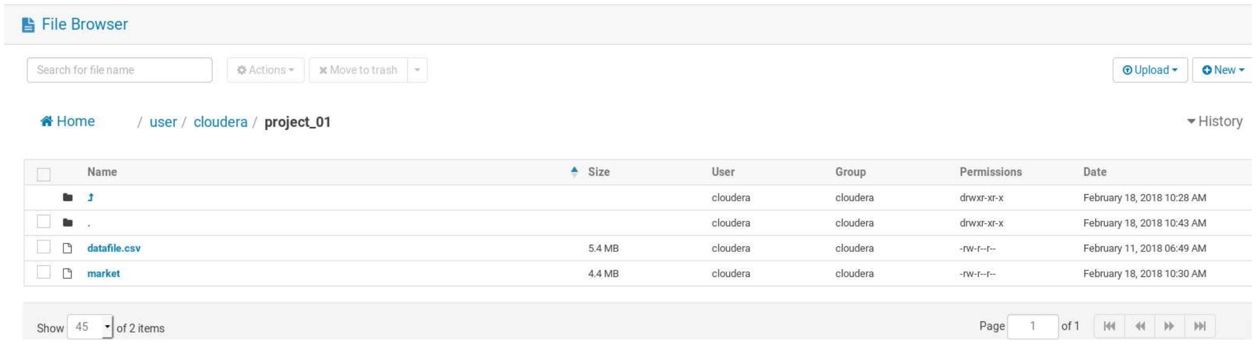
By Suprasanna Pradhan

1. Load data and create Spark data frame

A. Created file in HDFS.

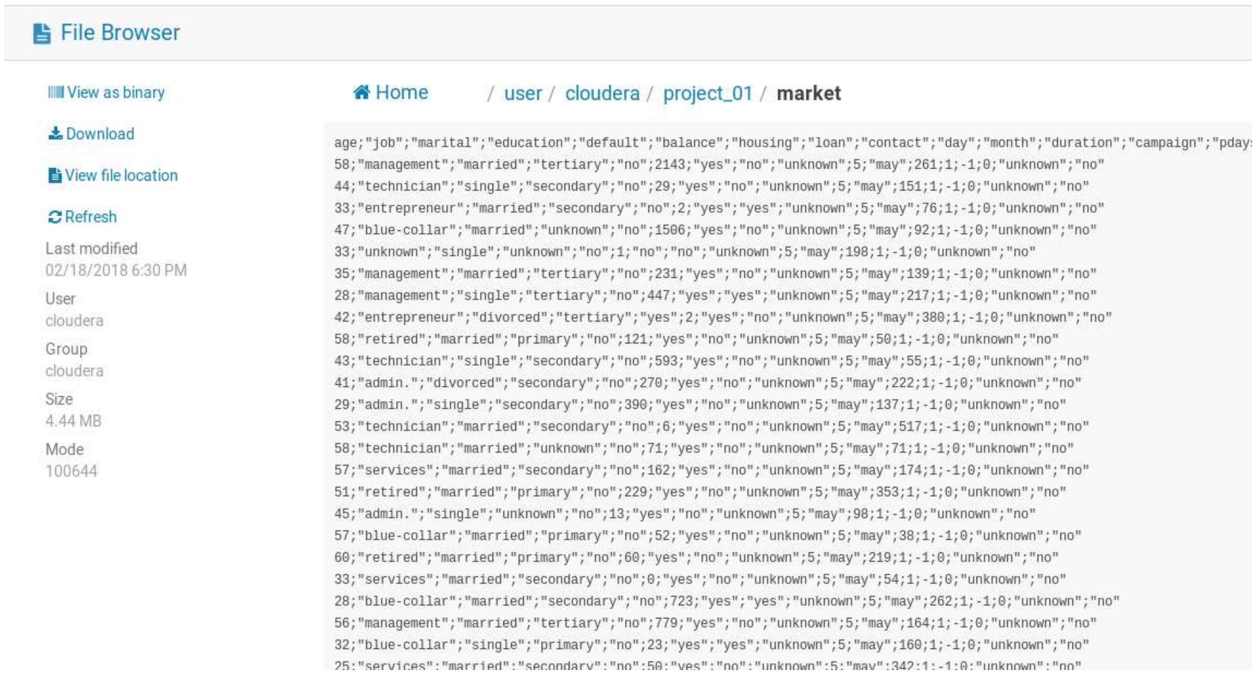


```
[cloudera@quickstart project]$ hdfs dfs -ls project_01
Found 1 items
-rw-r--r--   1 cloudera cloudera    5650234 2018-02-11 06:49 project_01/datafile.csv
[cloudera@quickstart project]$ hdfs dfs -put market.txt /user/cloudera/project_01
hdfs[cloudera@quickstart project]$ hdfs dfs -ls project_01
Found 2 items
-rw-r--r--   1 cloudera cloudera    5650234 2018-02-11 06:49 project_01/datafile.csv
 Cloudera Live : Welcome! - Cloudera    4655558 2018-02-18 10:30 project_01/market.txt
```



Name	Size	User	Group	Permissions	Date
↥		cloudera	cloudera	drwxr-xr-x	February 18, 2018 10:28 AM
.		cloudera	cloudera	drwxr-xr-x	February 18, 2018 10:43 AM
datafile.csv	5.4 MB	cloudera	cloudera	-rw-r--r--	February 11, 2018 06:49 AM
market	4.4 MB	cloudera	cloudera	-rw-r--r--	February 18, 2018 10:30 AM

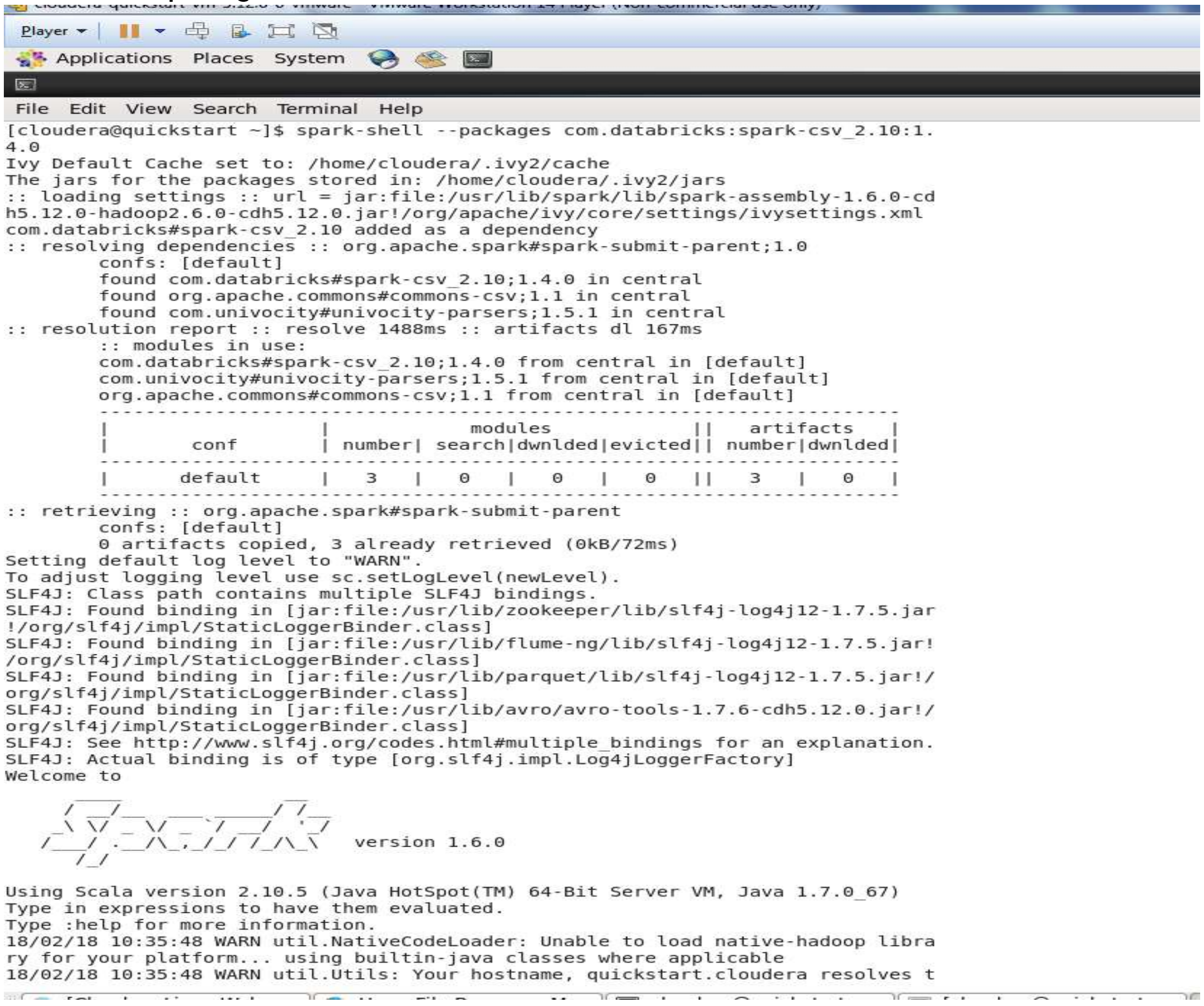
B. Input file view



Last modified
02/18/2018 6:30 PM
User
cloudera
Group
cloudera
Size
4.44 MB
Mode
100644

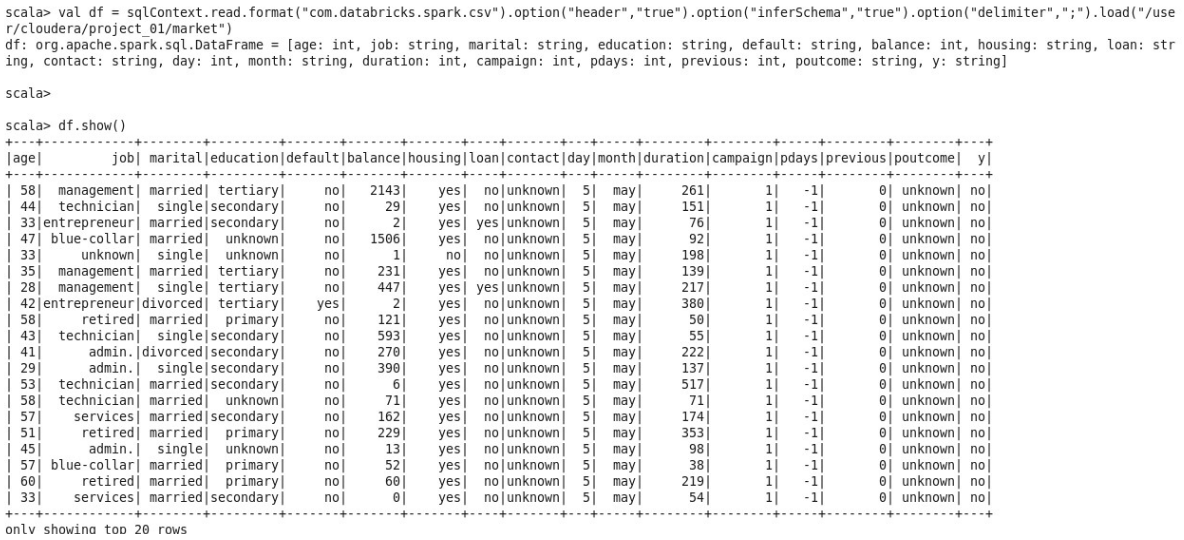
```
age;"job";"marital";"education";"default";"balance";"housing";"loan";"contact";"day";"month";"duration";"campaign";"pday
58;"management";"married";"tertiary";"no";2143;"yes";"no";"unknown";5;"may";261;1;-1;0;"unknown";"no"
44;"technician";"single";"secondary";"no";29;"yes";"no";"unknown";5;"may";151;1;-1;0;"unknown";"no"
33;"entrepreneur";"married";"secondary";"no";2;"yes";"yes";"unknown";5;"may";76;1;-1;0;"unknown";"no"
47;"blue-collar";"married";"unknown";"no";1506;"yes";"no";"unknown";5;"may";92;1;-1;0;"unknown";"no"
33;"unknown";"single";"unknown";"no";1;"no";"no";"unknown";5;"may";198;1;-1;0;"unknown";"no"
35;"management";"married";"tertiary";"no";231;"yes";"no";"unknown";5;"may";139;1;-1;0;"unknown";"no"
28;"management";"single";"tertiary";"no";447;"yes";"yes";"unknown";5;"may";217;1;-1;0;"unknown";"no"
42;"entrepreneur";"divorced";"tertiary";"yes";2;"yes";"no";"unknown";5;"may";380;1;-1;0;"unknown";"no"
58;"retired";"married";"primary";"no";121;"yes";"no";"unknown";5;"may";50;1;-1;0;"unknown";"no"
43;"technician";"single";"secondary";"no";593;"yes";"no";"unknown";5;"may";55;1;-1;0;"unknown";"no"
41;"admin.";"divorced";"secondary";"no";270;"yes";"no";"unknown";5;"may";222;1;-1;0;"unknown";"no"
29;"admin.";"single";"secondary";"no";390;"yes";"no";"unknown";5;"may";137;1;-1;0;"unknown";"no"
53;"technician";"married";"secondary";"no";6;"yes";"no";"unknown";5;"may";517;1;-1;0;"unknown";"no"
58;"technician";"married";"unknown";"no";71;"yes";"no";"unknown";5;"may";71;1;-1;0;"unknown";"no"
57;"services";"married";"secondary";"no";162;"yes";"no";"unknown";5;"may";174;1;-1;0;"unknown";"no"
51;"retired";"married";"primary";"no";229;"yes";"no";"unknown";5;"may";353;1;-1;0;"unknown";"no"
45;"admin.";"single";"unknown";"no";13;"yes";"no";"unknown";5;"may";98;1;-1;0;"unknown";"no"
57;"blue-collar";"married";"primary";"no";52;"yes";"no";"unknown";5;"may";38;1;-1;0;"unknown";"no"
60;"retired";"married";"primary";"no";60;"yes";"no";"unknown";5;"may";219;1;-1;0;"unknown";"no"
33;"services";"married";"secondary";"no";0;"yes";"no";"unknown";5;"may";54;1;-1;0;"unknown";"no"
28;"blue-collar";"married";"secondary";"no";723;"yes";"yes";"unknown";5;"may";262;1;-1;0;"unknown";"no"
56;"management";"married";"tertiary";"no";779;"yes";"no";"unknown";5;"may";164;1;-1;0;"unknown";"no"
32;"blue-collar";"single";"primary";"no";23;"yes";"yes";"unknown";5;"may";160;1;-1;0;"unknown";"no"
25;"services";"married";"secondary";"no";50;"yes";"no";"unknown";5;"may";342;1;-1;0;"unknown";"no"
```

C. Load package in scala

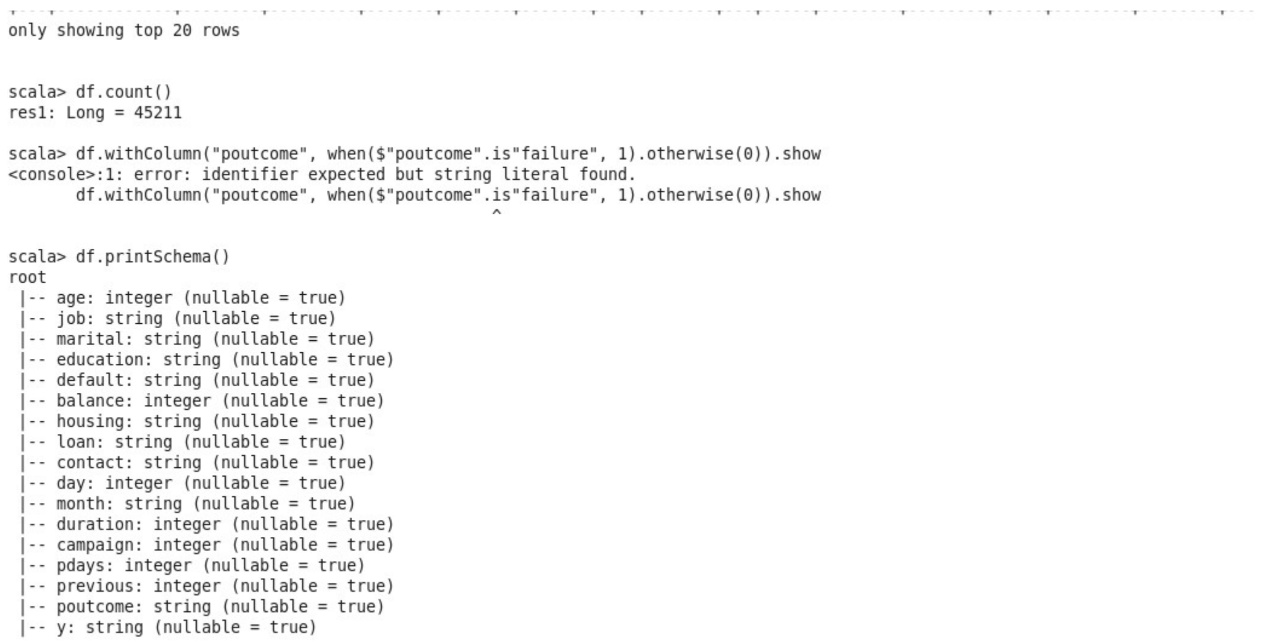

Player
Applications Places System
File Edit View Search Terminal Help

```
[cloudera@quickstart ~]$ spark-shell --packages com.databricks:spark-csv_2.10:1.
4.0
Ivy Default Cache set to: /home/cloudera/.ivy2/cache
The jars for the packages stored in: /home/cloudera/.ivy2/jars
:: loading settings :: url = jar:file:/usr/lib/spark/lib/spark-assembly-1.6.0-cd
h5.12.0-hadoop2.6.0-cdh5.12.0.jar!/org/apache/ivy/core/settings/ivysettings.xml
com.databricks#spark-csv_2.10 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent;1.0
        confs: [default]
        found com.databricks#spark-csv_2.10;1.4.0 in central
        found org.apache.commons#commons-csv;1.1 in central
        found com.univocity#univocity-parsers;1.5.1 in central
:: resolution report :: resolve 1488ms :: artifacts dl 167ms
        :: modules in use:
        com.databricks#spark-csv_2.10;1.4.0 from central in [default]
        com.univocity#univocity-parsers;1.5.1 from central in [default]
        org.apache.commons#commons-csv;1.1 from central in [default]
        ---------------------------------------------------------------------
        |                  |            modules            ||   artifacts   |
        |       conf       | number| search|dwnlded|evicted|| number|dwnlded|
        ---------------------------------------------------------------------
        |      default     |   3   |   0   |   0   |   0   ||   3   |   0   |
        ---------------------------------------------------------------------
:: retrieving :: org.apache.spark#spark-submit-parent
        confs: [default]
        0 artifacts copied, 3 already retrieved (0kB/72ms)
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/zookeeper/lib/slf4j-log4j12-1.7.5.jar
!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/flume-ng/lib/slf4j-log4j12-1.7.5.jar!
/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/parquet/lib/slf4j-log4j12-1.7.5.jar!/
org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/avro/avro-tools-1.7.6-cdh5.12.0.jar!/
org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.0
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_67)
Type in expressions to have them evaluated.
Type :help for more information.
18/02/18 10:35:48 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
18/02/18 10:35:48 WARN util.Utils: Your hostname, quickstart.cloudera resolves t
```

D. Create dataframe in Scala

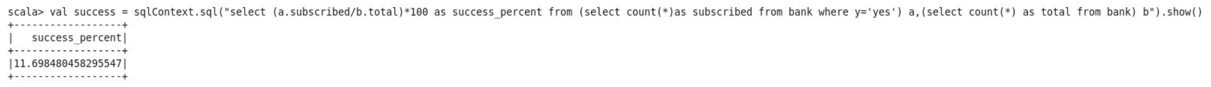


```
scala> val df = sqlContext.read.format("com.databricks.spark.csv").option("header","true").option("inferSchema","true").option("delimiter",";").load("/use
r/cloudera/project_01/market")
df: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, default: string, balance: int, housing: string, loan: str
ing, contact: string, day: int, month: string, duration: int, campaign: int, pdays: int, previous: int, poutcome: string, y: string]

scala>

scala> df.show()
+---+------------+--------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
|age|         job| marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|  y|
+---+------------+--------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
| 58|  management| married| tertiary|     no|   2143|    yes|  no|unknown|  5|  may|     261|       1|   -1|       0| unknown| no|
| 44|  technician|  single|secondary|     no|     29|    yes|  no|unknown|  5|  may|     151|       1|   -1|       0| unknown| no|
| 33|entrepreneur| married|secondary|     no|      2|    yes| yes|unknown|  5|  may|      76|       1|   -1|       0| unknown| no|
| 47| blue-collar| married|  unknown|     no|   1506|    yes|  no|unknown|  5|  may|      92|       1|   -1|       0| unknown| no|
| 33|     unknown|  single|  unknown|     no|      1|     no|  no|unknown|  5|  may|     198|       1|   -1|       0| unknown| no|
| 35|  management| married| tertiary|     no|    231|    yes|  no|unknown|  5|  may|     139|       1|   -1|       0| unknown| no|
| 28|  management|  single| tertiary|     no|    447|    yes| yes|unknown|  5|  may|     217|       1|   -1|       0| unknown| no|
| 42|entrepreneur|divorced| tertiary|    yes|      2|    yes|  no|unknown|  5|  may|     380|       1|   -1|       0| unknown| no|
| 58|     retired| married|  primary|     no|    121|    yes|  no|unknown|  5|  may|      50|       1|   -1|       0| unknown| no|
| 43|  technician|  single|secondary|     no|    593|    yes|  no|unknown|  5|  may|      55|       1|   -1|       0| unknown| no|
| 41|      admin.|divorced|secondary|     no|    270|    yes|  no|unknown|  5|  may|     222|       1|   -1|       0| unknown| no|
| 29|      admin.|  single|secondary|     no|    390|    yes|  no|unknown|  5|  may|     137|       1|   -1|       0| unknown| no|
| 53|  technician| married|secondary|     no|      6|    yes|  no|unknown|  5|  may|     517|       1|   -1|       0| unknown| no|
| 58|  technician| married|  unknown|     no|     71|    yes|  no|unknown|  5|  may|      71|       1|   -1|       0| unknown| no|
| 57|    services| married|secondary|     no|    162|    yes|  no|unknown|  5|  may|     174|       1|   -1|       0| unknown| no|
| 51|     retired| married|  primary|     no|    229|    yes|  no|unknown|  5|  may|     353|       1|   -1|       0| unknown| no|
| 45|      admin.|  single|  unknown|     no|     13|    yes|  no|unknown|  5|  may|      98|       1|   -1|       0| unknown| no|
| 57| blue-collar| married|  primary|     no|     52|    yes|  no|unknown|  5|  may|      38|       1|   -1|       0| unknown| no|
| 60|     retired| married|  primary|     no|     60|    yes|  no|unknown|  5|  may|     219|       1|   -1|       0| unknown| no|
| 33|    services| married|secondary|     no|      0|    yes|  no|unknown|  5|  may|      54|       1|   -1|       0| unknown| no|
+---+------------+--------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
only showing top 20 rows
```

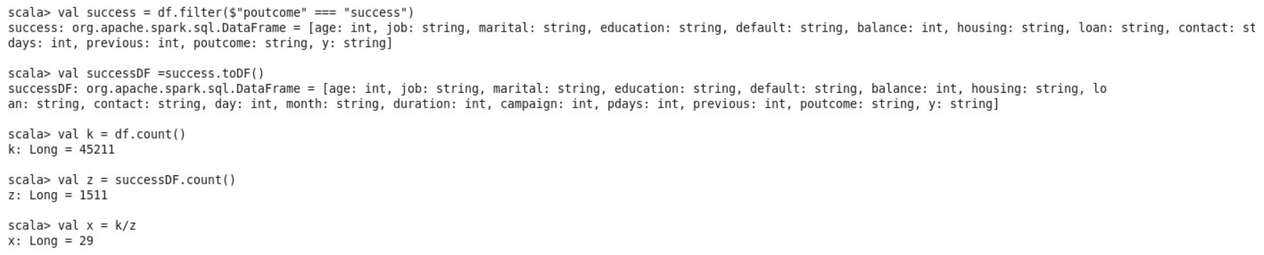


```
only showing top 20 rows


scala> df.count()
res1: Long = 45211

scala> df.withColumn("poutcome", when($"poutcome".is"failure", 1).otherwise(0)).show
<console>:1: error: identifier expected but string literal found.
       df.withColumn("poutcome", when($"poutcome".is"failure", 1).otherwise(0)).show
                                                  ^


scala> df.printSchema()
root
 |-- age: integer (nullable = true)
 |-- job: string (nullable = true)
 |-- marital: string (nullable = true)
 |-- education: string (nullable = true)
 |-- default: string (nullable = true)
 |-- balance: integer (nullable = true)
 |-- housing: string (nullable = true)
 |-- loan: string (nullable = true)
 |-- contact: string (nullable = true)
 |-- day: integer (nullable = true)
 |-- month: string (nullable = true)
 |-- duration: integer (nullable = true)
 |-- campaign: integer (nullable = true)
 |-- pdays: integer (nullable = true)
 |-- previous: integer (nullable = true)
 |-- poutcome: string (nullable = true)
 |-- y: string (nullable = true)
```

2. Marketing Success Rate

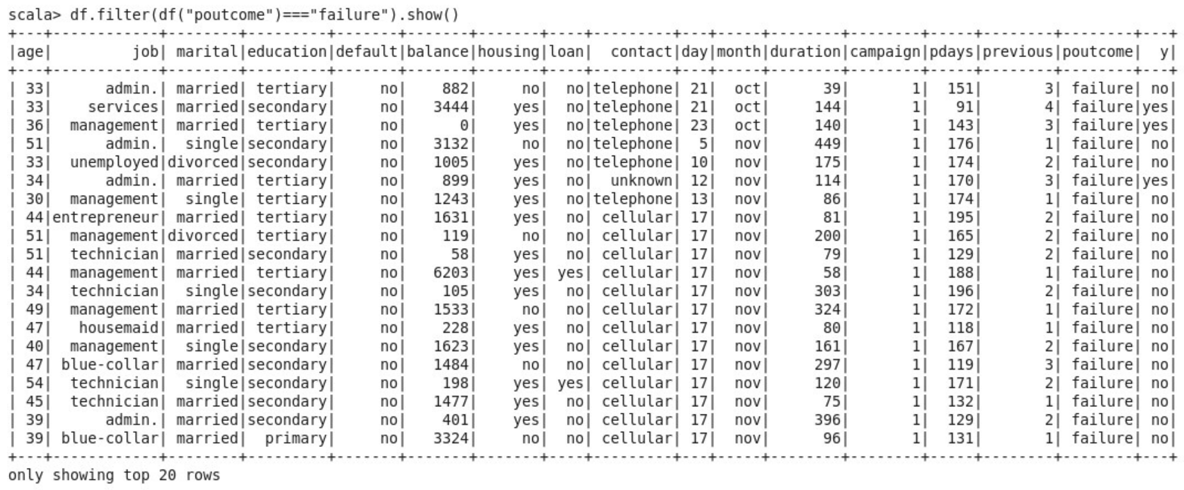


```
scala> val success = sqlContext.sql("select (a.subscribed/b.total)*100 as success_percent from (select count(*)as subscribed from bank where y='yes') a,(select count(*) as total from bank) b").show()
+------------------+
|   success_percent|
+------------------+
|11.698480458295547|
+------------------+
```



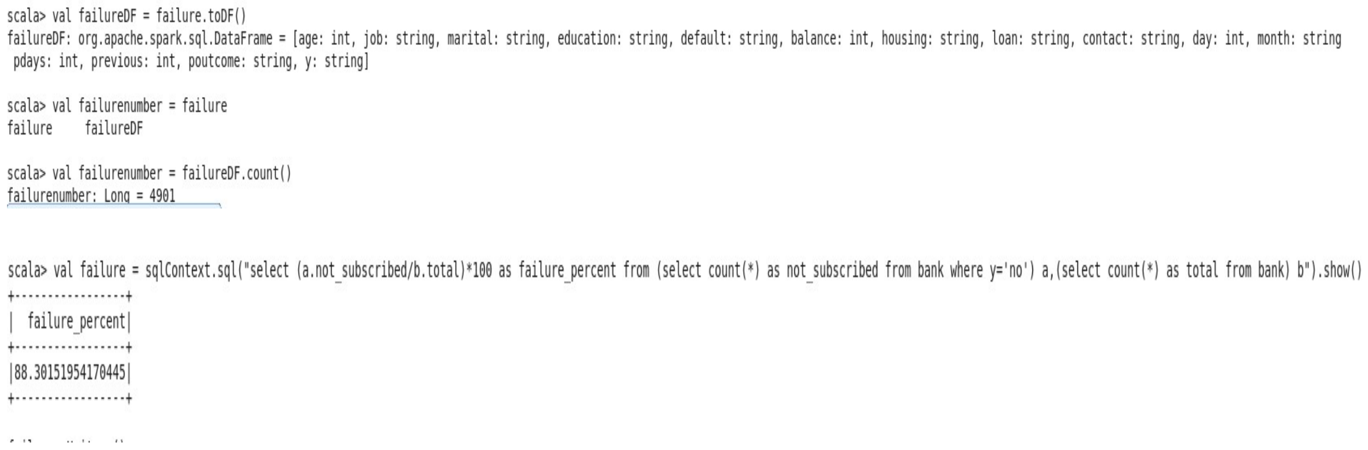
```
scala> val success = df.filter($"poutcome" === "success")
success: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, default: string, balance: int, housing: string, loan: string, contact: st
days: int, previous: int, poutcome: string, y: string]

scala> val successDF =success.toDF()
successDF: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, default: string, balance: int, housing: string, lo
an: string, contact: string, day: int, month: string, duration: int, campaign: int, pdays: int, previous: int, poutcome: string, y: string]

scala> val k = df.count()
k: Long = 45211

scala> val z = successDF.count()
z: Long = 1511

scala> val x = k/z
x: Long = 29
```

A. Marketing Failure Rate



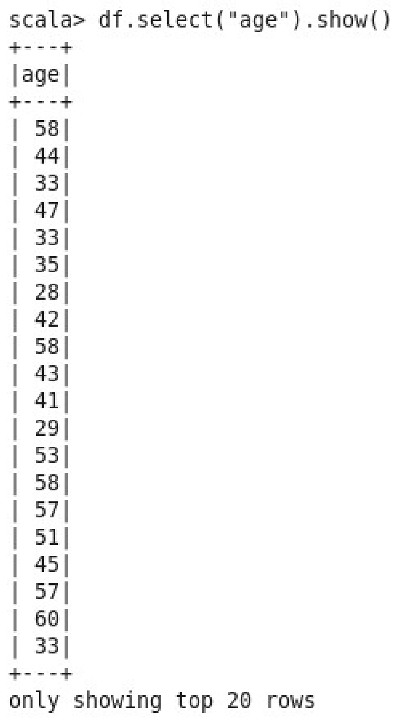
```
scala> df.filter(df("poutcome")==="failure").show()
+---+------------+--------+---------+-------+-------+-------+----+---------+---+-----+--------+--------+-----+--------+--------+---+
|age|         job| marital|education|default|balance|housing|loan|  contact|day|month|duration|campaign|pdays|previous|poutcome|  y|
+---+------------+--------+---------+-------+-------+-------+----+---------+---+-----+--------+--------+-----+--------+--------+---+
| 33|      admin.| married| tertiary|     no|    882|     no|  no|telephone| 21|  oct|      39|       1|  151|       3| failure| no|
| 33|    services| married|secondary|     no|   3444|    yes|  no|telephone| 21|  oct|     144|       1|   91|       4| failure|yes|
| 36|  management| married| tertiary|     no|      0|    yes|  no|telephone| 23|  oct|     140|       1|  143|       3| failure|yes|
| 51|      admin.|  single|secondary|     no|   3132|     no|  no|telephone|  5|  nov|     449|       1|  176|       1| failure| no|
| 33|  unemployed|divorced|secondary|     no|   1005|    yes|  no|telephone| 10|  nov|     175|       1|  174|       2| failure| no|
| 34|      admin.| married| tertiary|     no|    899|    yes|  no|  unknown| 12|  nov|     114|       1|  170|       3| failure|yes|
| 30|  management|  single| tertiary|     no|   1243|    yes|  no|telephone| 13|  nov|      86|       1|  174|       1| failure| no|
| 44|entrepreneur| married| tertiary|     no|   1631|    yes|  no| cellular| 17|  nov|      81|       1|  195|       2| failure| no|
| 51|  management|divorced| tertiary|     no|    119|     no|  no| cellular| 17|  nov|     200|       1|  165|       2| failure| no|
| 51|  technician| married|secondary|     no|     58|    yes|  no| cellular| 17|  nov|      79|       1|  129|       2| failure| no|
| 44|  management| married| tertiary|     no|   6203|    yes| yes| cellular| 17|  nov|      58|       1|  188|       1| failure| no|
| 34|  technician|  single|secondary|     no|    105|    yes|  no| cellular| 17|  nov|     303|       1|  196|       2| failure| no|
| 49|  management| married| tertiary|     no|   1533|     no|  no| cellular| 17|  nov|     324|       1|  172|       1| failure| no|
| 47|   housemaid| married| tertiary|     no|    228|    yes|  no| cellular| 17|  nov|      80|       1|  118|       1| failure| no|
| 40|  management|  single|secondary|     no|   1623|    yes|  no| cellular| 17|  nov|     161|       1|  167|       2| failure| no|
| 47| blue-collar| married|secondary|     no|   1484|     no|  no| cellular| 17|  nov|     297|       1|  119|       3| failure| no|
| 54|  technician|  single|secondary|     no|    198|    yes| yes| cellular| 17|  nov|     120|       1|  171|       2| failure| no|
| 45|  technician| married|secondary|     no|   1477|    yes|  no| cellular| 17|  nov|      75|       1|  132|       1| failure| no|
| 39|      admin.| married|secondary|     no|    401|    yes|  no| cellular| 17|  nov|     396|       1|  129|       2| failure| no|
| 39| blue-collar| married|  primary|     no|   3324|     no|  no| cellular| 17|  nov|      96|       1|  131|       1| failure| no|
+---+------------+--------+---------+-------+-------+-------+----+---------+---+-----+--------+--------+-----+--------+--------+---+
only showing top 20 rows
```

B. Count failure cases

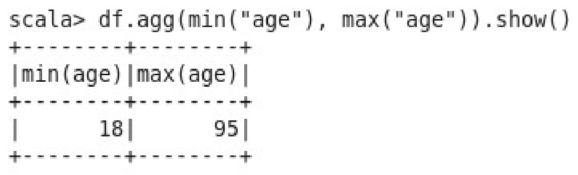


```
scala> val failureDF = failure.toDF()
failureDF: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, default: string, balance: int, housing: string, loan: string, contact: string, day: int, month: string
 pdays: int, previous: int, poutcome: string, y: string]

scala> val failurenumber = failure
failure     failureDF

scala> val failurenumber = failureDF.count()
failurenumber: Long = 4901


scala> val failure = sqlContext.sql("select (a.not_subscribed/b.total)*100 as failure_percent from (select count(*) as not_subscribed from bank where y='no') a,(select count(*) as total from bank) b").show()
+-----------------+
|  failure_percent|
+-----------------+
|88.30151954170445|
+-----------------+
```

3. **Maximum, Mean, and Minimum age of average targeted customer**

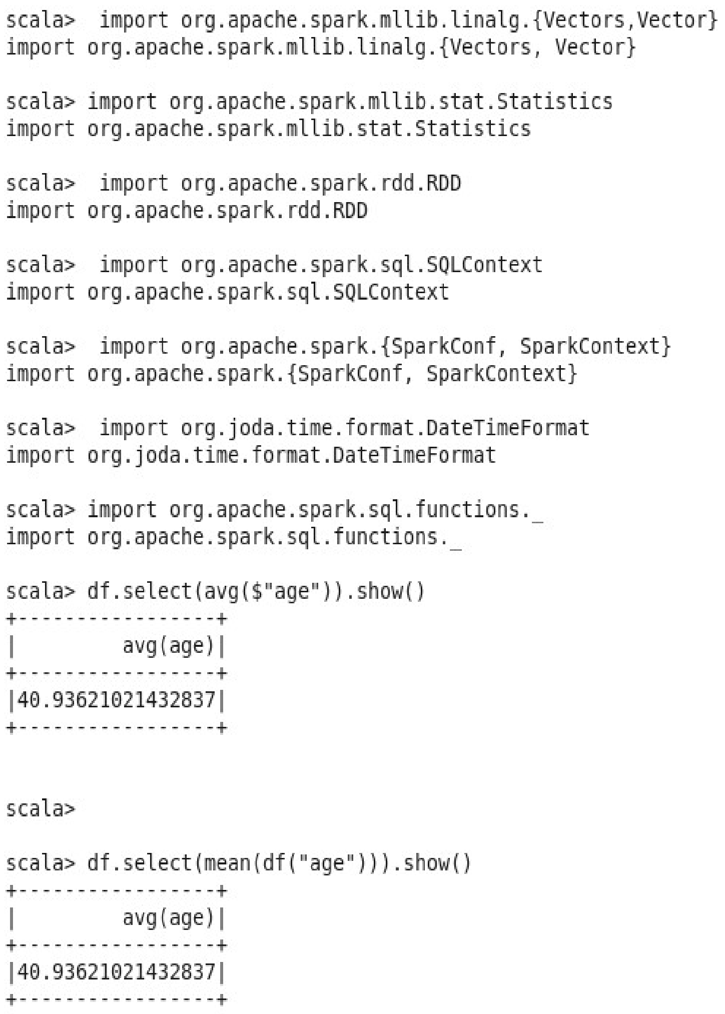


```
scala> df.select("age").show()
+---+
|age|
+---+
| 58|
| 44|
| 33|
| 47|
| 33|
| 35|
| 28|
| 42|
| 58|
| 43|
| 41|
| 29|
| 53|
| 58|
| 57|
| 51|
| 45|
| 57|
| 60|
| 33|
+---+
only showing top 20 rows
```



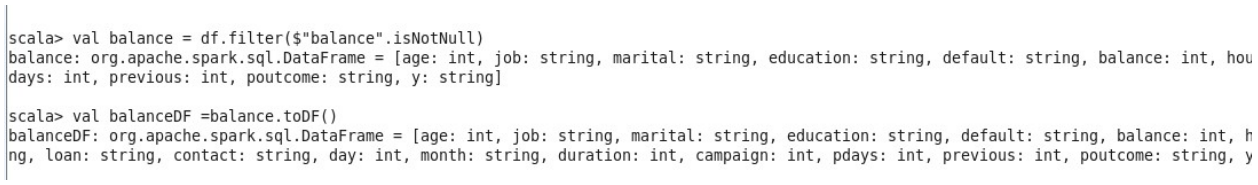
```
scala> df.agg(min("age"), max("age")).show()
+--------+--------+
|min(age)|max(age)|
+--------+--------+
|      18|      95|
+--------+--------+
```

A. Average of ages

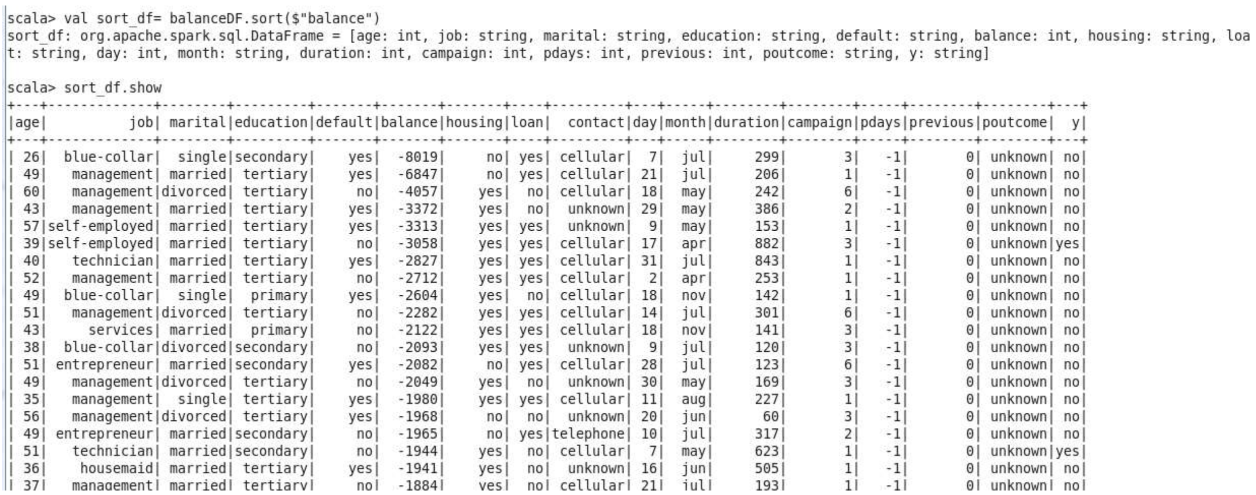


```
scala>  import org.apache.spark.mllib.linalg.{Vectors,Vector}
import org.apache.spark.mllib.linalg.{Vectors, Vector}

scala> import org.apache.spark.mllib.stat.Statistics
import org.apache.spark.mllib.stat.Statistics

scala>  import org.apache.spark.rdd.RDD
import org.apache.spark.rdd.RDD

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

scala>  import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.{SparkConf, SparkContext}

scala>  import org.joda.time.format.DateTimeFormat
import org.joda.time.format.DateTimeFormat

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

scala> df.select(avg($"age")).show()
+-----------------+
|         avg(age)|
+-----------------+
|40.93621021432837|
+-----------------+


scala>

scala> df.select(mean(df("age"))).show()
+-----------------+
|         avg(age)|
+-----------------+
|40.93621021432837|
+-----------------+
```

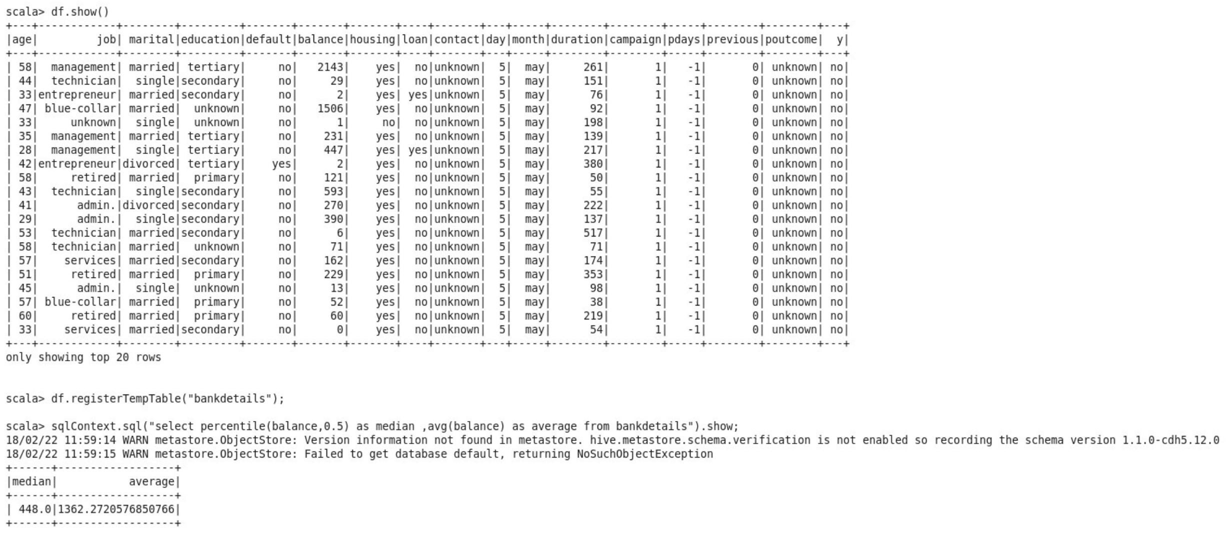
4. Check quality of customers by checking average balance, median balance of customers



```
scala> val balance = df.filter($"balance".isNotNull)
balance: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, default: string, balance: int, hou
days: int, previous: int, poutcome: string, y: string]

scala> val balanceDF =balance.toDF()
balanceDF: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, default: string, balance: int, h
ng, loan: string, contact: string, day: int, month: string, duration: int, campaign: int, pdays: int, previous: int, poutcome: string, y
```

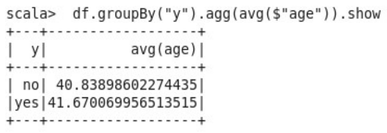


```
scala> val sort_df= balanceDF.sort($"balance")
sort_df: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, default: string, balance: int, housing: string, loa
t: string, day: int, month: string, duration: int, campaign: int, pdays: int, previous: int, poutcome: string, y: string]

scala> sort_df.show
+---+-------------+--------+---------+-------+-------+-------+----+---------+---+-----+--------+--------+-----+--------+--------+---+
|age|          job| marital|education|default|balance|housing|loan|  contact|day|month|duration|campaign|pdays|previous|poutcome|  y|
+---+-------------+--------+---------+-------+-------+-------+----+---------+---+-----+--------+--------+-----+--------+--------+---+
| 26|  blue-collar|  single|secondary|    yes|  -8019|     no| yes| cellular|  7|  jul|     299|       3|   -1|       0| unknown| no|
| 49|   management| married| tertiary|    yes|  -6847|     no| yes| cellular| 21|  jul|     206|       1|   -1|       0| unknown| no|
| 60|   management|divorced| tertiary|     no|  -4057|    yes|  no| cellular| 18|  may|     242|       6|   -1|       0| unknown| no|
| 43|   management| married| tertiary|    yes|  -3372|    yes|  no|  unknown| 29|  may|     386|       2|   -1|       0| unknown| no|
| 57|self-employed| married| tertiary|    yes|  -3313|    yes| yes|  unknown|  9|  may|     153|       1|   -1|       0| unknown| no|
| 39|self-employed| married| tertiary|     no|  -3058|    yes| yes| cellular| 17|  apr|     882|       3|   -1|       0| unknown|yes|
| 40|   technician| married| tertiary|    yes|  -2827|    yes| yes| cellular| 31|  jul|     843|       1|   -1|       0| unknown| no|
| 52|   management| married| tertiary|     no|  -2712|    yes| yes| cellular|  2|  apr|     253|       1|   -1|       0| unknown| no|
| 49|  blue-collar|  single|  primary|    yes|  -2604|    yes|  no| cellular| 18|  nov|     142|       1|   -1|       0| unknown| no|
| 51|   management|divorced| tertiary|     no|  -2282|    yes| yes| cellular| 14|  jul|     301|       6|   -1|       0| unknown| no|
| 43|     services| married|  primary|     no|  -2122|    yes| yes| cellular| 18|  nov|     141|       3|   -1|       0| unknown| no|
| 38|  blue-collar|divorced|secondary|     no|  -2093|    yes| yes|  unknown|  9|  jul|     120|       3|   -1|       0| unknown| no|
| 51| entrepreneur| married|secondary|    yes|  -2082|     no| yes| cellular| 28|  jul|     123|       6|   -1|       0| unknown| no|
| 49|   management|divorced| tertiary|     no|  -2049|    yes|  no|  unknown| 30|  may|     169|       3|   -1|       0| unknown| no|
| 35|   management|  single| tertiary|    yes|  -1980|    yes| yes| cellular| 11|  aug|     227|       1|   -1|       0| unknown| no|
| 56|   management|divorced| tertiary|    yes|  -1968|     no|  no|  unknown| 20|  jun|      60|       3|   -1|       0| unknown| no|
| 49| entrepreneur| married|secondary|     no|  -1965|     no| yes|telephone| 10|  jul|     317|       2|   -1|       0| unknown| no|
| 51|   technician| married|secondary|     no|  -1944|    yes|  no| cellular|  7|  may|     623|       1|   -1|       0| unknown|yes|
| 36|    housemaid| married| tertiary|    yes|  -1941|    yes|  no|  unknown| 16|  jun|     505|       1|   -1|       0| unknown| no|
| 37|   management| married| tertiary|     no|  -1884|    yes|  no| cellular| 21|  jul|     193|       1|   -1|       0| unknown| no|
```



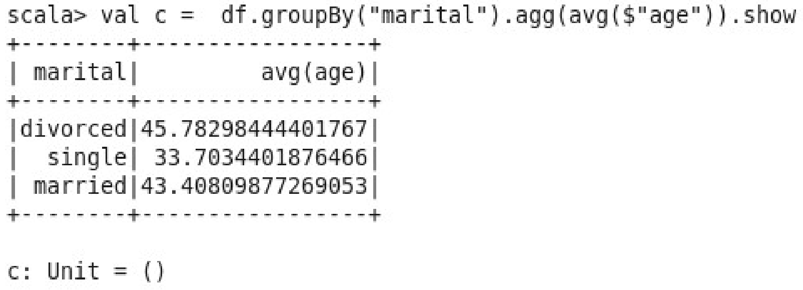
```
scala> df.show()
+---+------------+--------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
|age|         job| marital|education|default|balance|housing|loan|contact|day|month|duration|campaign|pdays|previous|poutcome|  y|
+---+------------+--------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
| 58|  management| married| tertiary|     no|   2143|    yes|  no|unknown|  5|  may|     261|       1|   -1|       0| unknown| no|
| 44|  technician|  single|secondary|     no|     29|    yes|  no|unknown|  5|  may|     151|       1|   -1|       0| unknown| no|
| 33|entrepreneur| married|secondary|     no|      2|    yes| yes|unknown|  5|  may|      76|       1|   -1|       0| unknown| no|
| 47| blue-collar| married|  unknown|     no|   1506|    yes|  no|unknown|  5|  may|      92|       1|   -1|       0| unknown| no|
| 33|     unknown|  single|  unknown|     no|      1|     no|  no|unknown|  5|  may|     198|       1|   -1|       0| unknown| no|
| 35|  management| married| tertiary|     no|    231|    yes|  no|unknown|  5|  may|     139|       1|   -1|       0| unknown| no|
| 28|  management|  single| tertiary|     no|    447|    yes| yes|unknown|  5|  may|     217|       1|   -1|       0| unknown| no|
| 42|entrepreneur|divorced| tertiary|    yes|      2|    yes|  no|unknown|  5|  may|     380|       1|   -1|       0| unknown| no|
| 58|     retired| married|  primary|     no|    121|    yes|  no|unknown|  5|  may|      50|       1|   -1|       0| unknown| no|
| 43|  technician|  single|secondary|     no|    593|    yes|  no|unknown|  5|  may|      55|       1|   -1|       0| unknown| no|
| 41|      admin.|divorced|secondary|     no|    270|    yes|  no|unknown|  5|  may|     222|       1|   -1|       0| unknown| no|
| 29|      admin.|  single|secondary|     no|    390|    yes|  no|unknown|  5|  may|     137|       1|   -1|       0| unknown| no|
| 53|  technician| married|secondary|     no|      6|    yes|  no|unknown|  5|  may|     517|       1|   -1|       0| unknown| no|
| 58|  technician| married|  unknown|     no|     71|    yes|  no|unknown|  5|  may|      71|       1|   -1|       0| unknown| no|
| 57|    services| married|secondary|     no|    162|    yes|  no|unknown|  5|  may|     174|       1|   -1|       0| unknown| no|
| 51|     retired| married|  primary|     no|    229|    yes|  no|unknown|  5|  may|     353|       1|   -1|       0| unknown| no|
| 45|      admin.|  single|  unknown|     no|     13|    yes|  no|unknown|  5|  may|      98|       1|   -1|       0| unknown| no|
| 57| blue-collar| married|  primary|     no|     52|    yes|  no|unknown|  5|  may|      38|       1|   -1|       0| unknown| no|
| 60|     retired| married|  primary|     no|     60|    yes|  no|unknown|  5|  may|     219|       1|   -1|       0| unknown| no|
| 33|    services| married|secondary|     no|      0|    yes|  no|unknown|  5|  may|      54|       1|   -1|       0| unknown| no|
+---+------------+--------+---------+-------+-------+-------+----+-------+---+-----+--------+--------+-----+--------+--------+---+
only showing top 20 rows


scala> df.registerTempTable("bankdetails");

scala> sqlContext.sql("select percentile(balance,0.5) as median ,avg(balance) as average from bankdetails").show;
18/02/22 11:59:14 WARN metastore.ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not enabled so recording the schema version 1.1.0-cdh5.12.0
18/02/22 11:59:15 WARN metastore.ObjectStore: Failed to get database default, returning NoSuchObjectException
+------+------------------+
|median|           average|
+------+------------------+
| 448.0|1362.2720576850766|
+------+------------------+
```

5. **Check if age matters in marketing subscription for deposit**



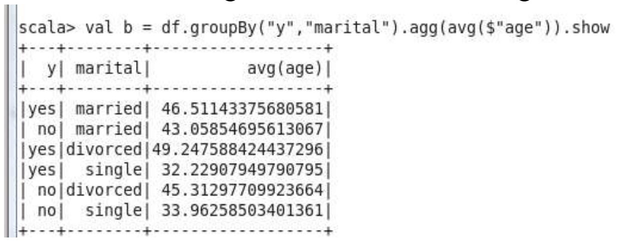
```
scala>  df.groupBy("y").agg(avg($"age")).show
+---+------------------+
|  y|          avg(age)|
+---+------------------+
| no| 40.83898602274435|
|yes|41.670069956513515|
+---+------------------+
```

6. **Check if marital status mattered for subscription to deposit**



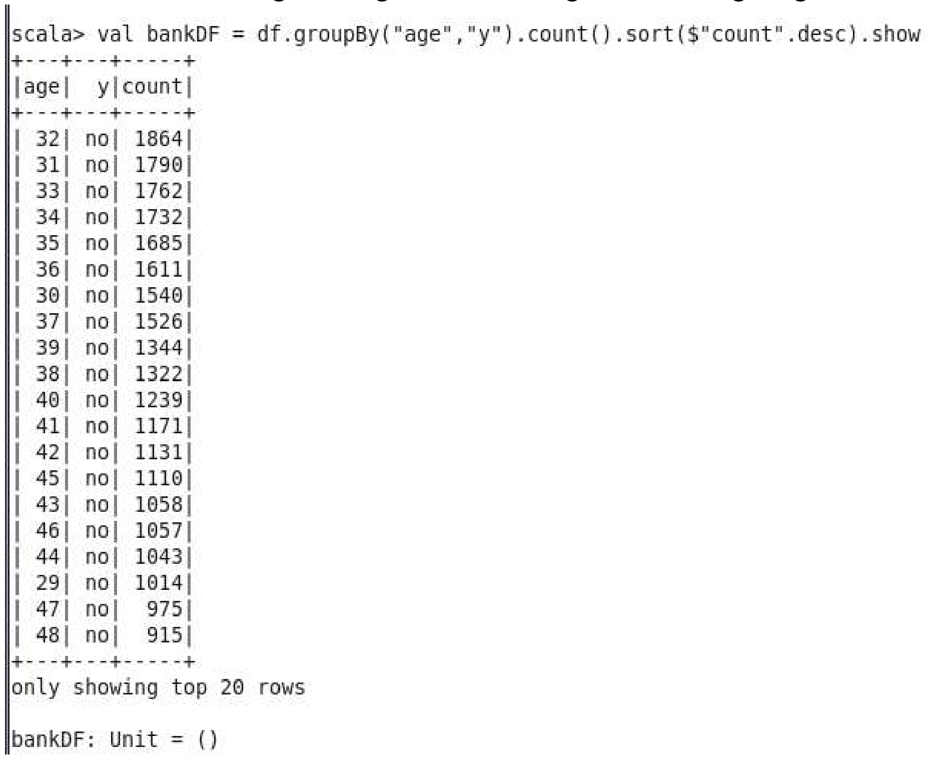
```
scala> val c =  df.groupBy("marital").agg(avg($"age")).show
+--------+-----------------+
| marital|         avg(age)|
+--------+-----------------+
|divorced|45.78298444401767|
|  single| 33.7034401876466|
| married|43.40809877269053|
+--------+-----------------+

c: Unit = ()
```

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

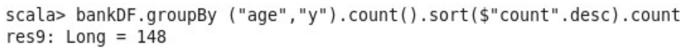


```
scala> val b = df.groupBy("y","marital").agg(avg($"age")).show
+---+--------+------------------+
|  y| marital|          avg(age)|
+---+--------+------------------+
|yes| married| 46.51143375680581|
| no| married| 43.05854695613067|
|yes|divorced|49.247588424437296|
|yes|  single| 32.22907949790795|
| no|divorced| 45.31297709923664|
| no|  single| 33.96258503401361|
+---+--------+------------------+
```

8. **Do feature engineering for column—age and find right age effect on campaign**

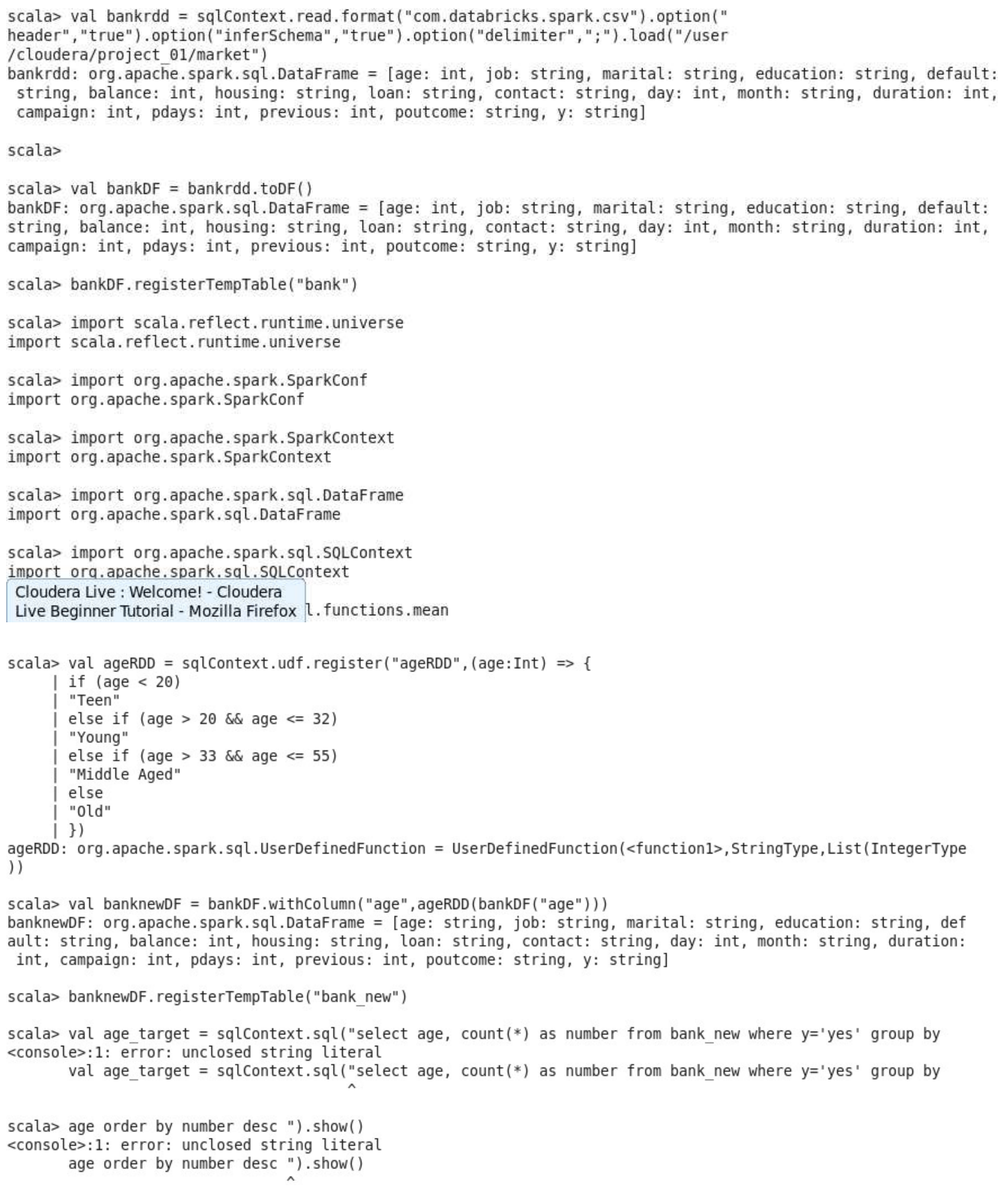


```
scala> val bankDF = df.groupBy("age","y").count().sort($"count".desc).show
+---+---+-----+
|age|  y|count|
+---+---+-----+
| 32| no| 1864|
| 31| no| 1790|
| 33| no| 1762|
| 34| no| 1732|
| 35| no| 1685|
| 36| no| 1611|
| 30| no| 1540|
| 37| no| 1526|
| 39| no| 1344|
| 38| no| 1322|
| 40| no| 1239|
| 41| no| 1171|
| 42| no| 1131|
| 45| no| 1110|
| 43| no| 1058|
| 46| no| 1057|
| 44| no| 1043|
| 29| no| 1014|
| 47| no|  975|
| 48| no|  915|
+---+---+-----+
only showing top 20 rows

bankDF: Unit = ()
```



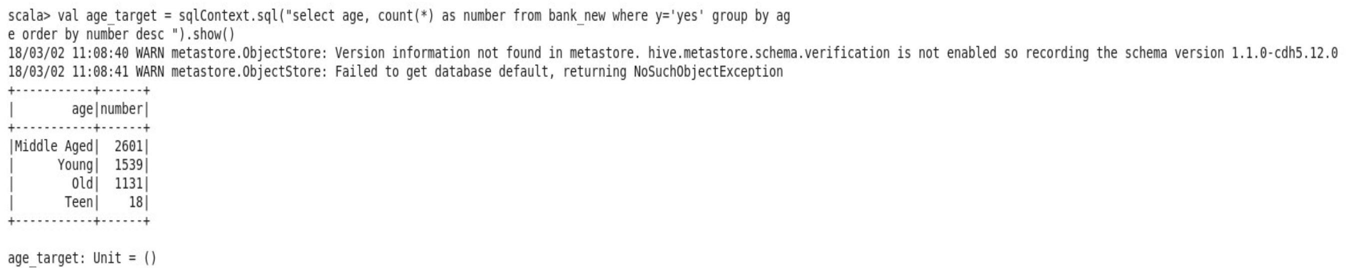
```
scala> bankDF.groupBy ("age","y").count().sort($"count".desc).count
res9: Long = 148
```

A. Create another categorical new data frame to find age category



```
scala> val bankrdd = sqlContext.read.format("com.databricks.spark.csv").option("
header","true").option("inferSchema","true").option("delimiter",";").load("/user
/cloudera/project_01/market")
bankrdd: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, default:
 string, balance: int, housing: string, loan: string, contact: string, day: int, month: string, duration: int,
 campaign: int, pdays: int, previous: int, poutcome: string, y: string]

scala>

scala> val bankDF = bankrdd.toDF()
bankDF: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string, default:
string, balance: int, housing: string, loan: string, contact: string, day: int, month: string, duration: int,
campaign: int, pdays: int, previous: int, poutcome: string, y: string]

scala> bankDF.registerTempTable("bank")

scala> import scala.reflect.runtime.universe
import scala.reflect.runtime.universe

scala> import org.apache.spark.SparkConf
import org.apache.spark.SparkConf

scala> import org.apache.spark.SparkContext
import org.apache.spark.SparkContext

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

scala> import org.apache.spark.sql.SQLContext
import org.apache.spark.sql.SQLContext
Cloudera Live : Welcome! - Cloudera
Live Beginner Tutorial - Mozilla Firefox l.functions.mean


scala> val ageRDD = sqlContext.udf.register("ageRDD",(age:Int) => {
     | if (age < 20)
     | "Teen"
     | else if (age > 20 && age <= 32)
     | "Young"
     | else if (age > 33 && age <= 55)
     | "Middle Aged"
     | else
     | "Old"
     | })
ageRDD: org.apache.spark.sql.UserDefinedFunction = UserDefinedFunction(<function1>,StringType,List(IntegerType
))

scala> val banknewDF = bankDF.withColumn("age",ageRDD(bankDF("age")))
banknewDF: org.apache.spark.sql.DataFrame = [age: string, job: string, marital: string, education: string, def
ault: string, balance: int, housing: string, loan: string, contact: string, day: int, month: string, duration:
 int, campaign: int, pdays: int, previous: int, poutcome: string, y: string]

scala> banknewDF.registerTempTable("bank_new")

scala> val age_target = sqlContext.sql("select age, count(*) as number from bank_new where y='yes' group by
<console>:1: error: unclosed string literal
       val age_target = sqlContext.sql("select age, count(*) as number from bank_new where y='yes' group by
                                       ^

scala> age order by number desc ").show()
<console>:1: error: unclosed string literal
       age order by number desc ").show()
                                ^
```

B. Check which age category shown more interest on deposits



```
scala> val age_target = sqlContext.sql("select age, count(*) as number from bank_new where y='yes' group by ag
e order by number desc ").show()
18/03/02 11:08:40 WARN metastore.ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not enabled so recording the schema version 1.1.0-cdh5.12.0
18/03/02 11:08:41 WARN metastore.ObjectStore: Failed to get database default, returning NoSuchObjectException
+-----------+------+
|        age|number|
+-----------+------+
|Middle Aged|  2601|
|      Young|  1539|
|        Old|  1131|
|       Teen|    18|
+-----------+------+

age_target: Unit = ()
```

Conclusion: Middle age people are showing more interest on deposits