

→ Uploaded Data in HDFS as bank_edited.json and Loaded Data (from HDFS) and Created Spark Dataframe

```
scala> val df = spark.read.option("multiline","true").json("/user/vishalcanada1316gmail/bank_edited.json")
df: org.apache.spark.sql.DataFrame = [age: bigint, balance: bigint ... 15 more fields]

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

scala>
```

→ Oldest Customer targeted is 95 years of age while the youngest one is 18 years of age and average or mean age of the customer is approx. 41 years

→ Max. Balance of customer is 102127 while Min. Balance is -8019 and Median balance is 448

```
scala> df.agg(max($"age"),min($"age"),avg($"age")).show()
+-----+-----+-----+
|max(age)|min(age)|    avg(age)|
+-----+-----+-----+
|    95|    18|40.93621021432837|
+-----+-----+-----+

scala> df.createOrReplaceTempView("df1")

scala> val df2 = sql("SELECT max(balance) as max, min(balance) as min, avg(balance) as average, percentile_approx(balance, 0.5) as median FROM df1");
df2: org.apache.spark.sql.DataFrame = [max: bigint, min: bigint ... 2 more fields]

scala> df2.show()
+-----+-----+-----+-----+
|max|    min|    average|median|
+-----+-----+-----+-----+
|102127|-8019|1362.2720576850766|    448|
+-----+-----+-----+-----+

scala>
```

→ Age is dominating factor in subscription for term deposit as it is evident from the data that the interest is more from the younger age customers, compared to older age customers.

```
scala> df.registerTempTable("age")
warning: there was one deprecation warning; re-run with -deprecation for details

scala> val agedf = spark.sql("select age, count(*) as count from age where y='yes' group by age order by count desc")
agedf: org.apache.spark.sql.DataFrame = [age: bigint, count: bigint]

scala> agedf.show()
+-----+-----+
|age|count|
+-----+-----+
| 32|   221|
| 30|   217|
| 33|   210|
| 35|   209|
| 31|   206|
| 34|   188|
| 36|   195|
| 29|   171|
| 37|   170|
| 28|   162|
| 38|   144|
| 39|   143|
| 27|   141|
| 26|   134|
| 41|   120|
| 46|   118|
| 40|   116|
| 25|   113|
| 47|   113|
| 42|   111|
+-----+-----+
only showing top 20 rows

scala>
```

→ Looking at the data it can be concluded that more married customers subscribed than single customers for term deposit

```
Assessment x vishalcanada1316: x Word Online - Ch... x Market Analysis - x Alinity - Advanced x Property Search | x Property Search | x For sale: LT 2 RUS... x + - □ x
slbdh-webconsole.corestack.io
Apps JENI Whois search - Wh... My Work Banking Gmail YouTube Maps Network Solutions... Login Network Solutions... News Translate Hadoop - Google D... Hadoop

scala> val marital_subscribed = spark.sql("select marital, count(*) as count from age where y='yes' group by marital order by count desc")
marital_subscribed: org.apache.spark.sql.DataFrame = [marital: string, count: bigint]

scala> marital_subscribed.show()
+-----+-----+
| marital|count|
+-----+-----+
| married| 2755|
| single | 1912|
| divorced| 622|
+-----+-----+

scala>
```

→ Below table clearly suggests that age is prominent factor than marital status for subscrib

```
Assessment x vishalcanada1316: x Word Online - Ch... x Market Analysis - x Alinity - Advanced x Property Search | x Property Search | x For sale: LT 2 RUS... x + - □ x
slbdh-webconsole.corestack.io
Apps JENI Whois search - Wh... My Work Banking Gmail YouTube Maps Network Solutions... Login Network Solutions... News Translate Hadoop - Google D... Hadoop

scala> val age_marital = spark.sql("select age, marital, count(*) as count from age where y='yes' group by age,marital order by count desc")
age_marital: org.apache.spark.sql.DataFrame = [age: bigint, marital: string ... 1 more field]

scala> age_marital.show()
+-----+-----+-----+
| age|marital|count|
+-----+-----+-----+
| 30| single | 151|
| 28| single | 138|
| 29| single | 133|
| 32| single | 124|
| 26| single | 121|
| 34| married | 118|
| 31| single | 111|
| 27| single | 110|
| 35| married | 101|
| 36| married | 100|
| 25| single | 99|
| 37| married | 98|
| 33| married | 97|
| 33| single | 97|
| 32| married | 87|
| 39| married | 87|
| 38| married | 86|
| 35| single | 84|
| 47| married | 83|
| 31| married | 80|
+-----+-----+-----+
only showing top 20 rows

scala>
```

? Below analysis mentions that the subscription was highest amongst Adult age category

```
Practice L... vishalcan... Word Onl... Market A... Microsoft... Alinity - A... Property... Property... For sale... age 31 to... W Young ad... + - □ ×
slbdh-webconsole.corestack.io
Apps JENI Whois search - Wh... My Work Banking Gmail YouTube Maps Network Solutions... Login Network Solutions... News Translate Hadoop - Google D... Hadoop

scala> newdf.registerTempTable("newdf1")
warning: there was one deprecation warning; re-run with -deprecation for details

scala> val agecategorized = spark.udf.register("agecategorized", (age: Int) => {
  if (age < 17)
    "Minor"
  else if (age > 17 && age <= 45)
    "Adult"
  else if (age > 46 && age <= 55)
    "Middle Aged"
  else
    "old"
})

21/03/20 21:28:04 WARN analysis.SimpleFunctionRegistry: The function agecategorized replaced a previously registered function.
agecategorized: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFunction(<function1>, StringType, Some(List(IntegerType)))

scala> val newdf = df.withColumn("age_category", agecategorized(df("age")))
newdf: org.apache.spark.sql.DataFrame = [age: bigint, balance: bigint ... 16 more fields]

scala> newdf.show(5)

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|age|balance|campaign|contact|day|default|duration|education|housing|job|loan|marital|month|pdays|poutcome|previous|y|age_category|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|58|2143|1|unknown|5|no|261|tertiary|yes|management|no|married|may|-1|unknown|0|no|old|
|44|29|1|unknown|5|no|151|secondary|yes|technician|no|single|may|-1|unknown|0|no|Adult|
|33|2|1|unknown|5|no|76|secondary|yes|entrepreneur|yes|married|may|-1|unknown|0|no|Adult|
|47|1506|1|unknown|5|no|92|unknown|yes|blue-collar|no|married|may|-1|unknown|0|no|Middle Aged|
|33|1|1|unknown|5|no|198|unknown|no|unknown|no|single|may|-1|unknown|0|no|Adult|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

only showing top 5 rows

scala> val age_category_count = spark.sql("select age_category, count(*) as count from newdf1 where y = 'yes' group by age_category order by count desc")
age_category_count: org.apache.spark.sql.DataFrame = [age_category: string, count: bigint]

scala> age_category_count.show()

+-----+-----+
|age_category|count|
+-----+-----+
|Adult|3490|
|old|1024|
|Middle Aged|775|
+-----+-----+
```