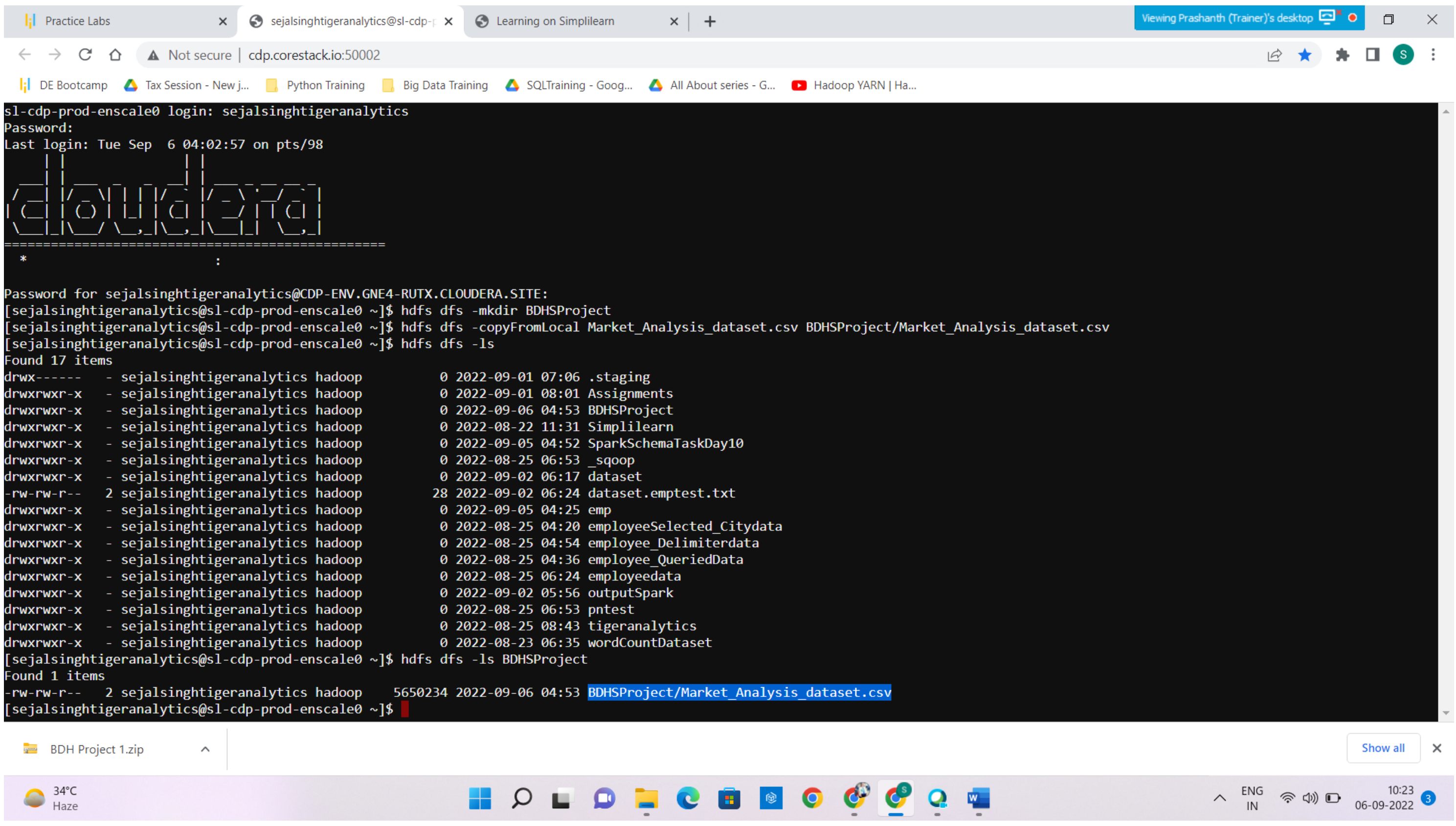
BDHS PROJECT

Load data and create a Spark data frame

```
hdfs dfs -mkdir BDHSProject
```

```
hdfs dfs -copyFromLocal Market_Analysis_dataset.csv
BDHSProject/Market_Analysis_dataset.csv
```

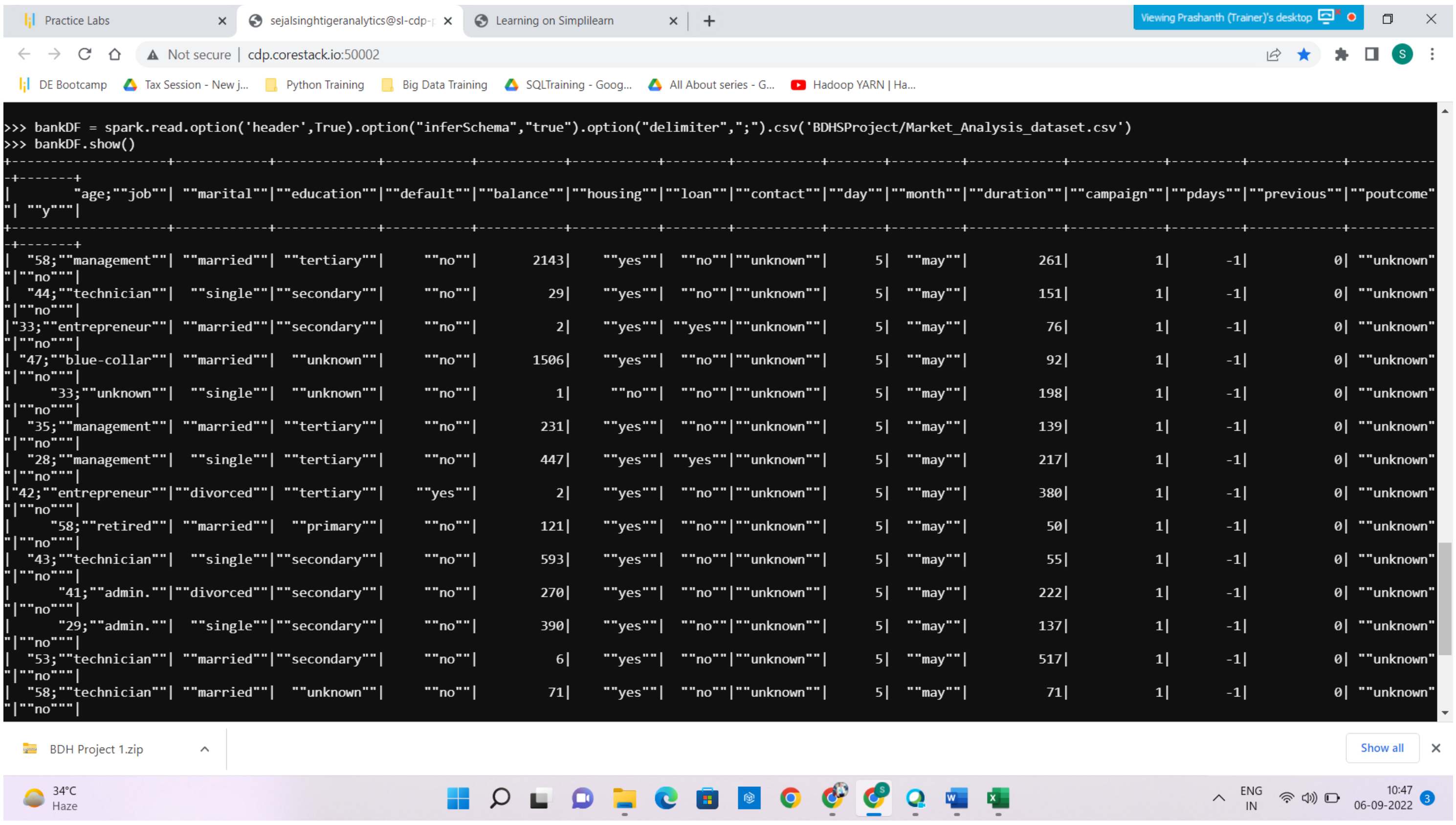

```
sl-cdp-prod-enscale0 login: sejalsinghtigeranalytics
Password:
Last login: Tue Sep  6 04:02:57 on pts/98
      | |               | |
   ___| | ___  _   _  __| | ___ _ __ __ _
  / __| |/ _ \| | | |/ _` |/ _ \ '__/ _` |
 | (__| | (_) | |_| | (_| |  __/ | | (_| |
  \___|_|\___/ \__,_|\__,_|\___|_|  \__,_|
==================================================
  *                         :

Password for sejalsinghtigeranalytics@CDP-ENV.GNE4-RUTX.CLOUDERA.SITE:
[sejalsinghtigeranalytics@sl-cdp-prod-enscale0 ~]$ hdfs dfs -mkdir BDHSProject
[sejalsinghtigeranalytics@sl-cdp-prod-enscale0 ~]$ hdfs dfs -copyFromLocal Market_Analysis_dataset.csv BDHSProject/Market_Analysis_dataset.csv
[sejalsinghtigeranalytics@sl-cdp-prod-enscale0 ~]$ hdfs dfs -ls
Found 17 items
drwx------   - sejalsinghtigeranalytics hadoop          0 2022-09-01 07:06 .staging
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-09-01 08:01 Assignments
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-09-06 04:53 BDHSProject
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-08-22 11:31 Simplilearn
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-09-05 04:52 SparkSchemaTaskDay10
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-08-25 06:53 _sqoop
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-09-02 06:17 dataset
-rw-rw-r--   2 sejalsinghtigeranalytics hadoop         28 2022-09-02 06:24 dataset.emptest.txt
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-09-05 04:25 emp
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-08-25 04:20 employeeSelected_Citydata
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-08-25 04:54 employee_Delimiterdata
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-08-25 04:36 employee_QueriedData
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-08-25 06:24 employeedata
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-09-02 05:56 outputSpark
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-08-25 06:53 pntest
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-08-25 08:43 tigeranalytics
drwxrwxr-x   - sejalsinghtigeranalytics hadoop          0 2022-08-23 06:35 wordCountDataset
[sejalsinghtigeranalytics@sl-cdp-prod-enscale0 ~]$ hdfs dfs -ls BDHSProject
Found 1 items
-rw-rw-r--   2 sejalsinghtigeranalytics hadoop    5650234 2022-09-06 04:53 BDHSProject/Market_Analysis_dataset.csv
[sejalsinghtigeranalytics@sl-cdp-prod-enscale0 ~]$
```

1. Load data and create a Spark data frame

```
bankDF =
spark.read.option('header',True).option("inferSchema","true").option("delimiter",";")
.csv('BDHSProject/Market_Analysis_dataset.csv')
```

```
bankDF.show()
```

```
bankDF.createOrReplaceTempView("bankT")
```

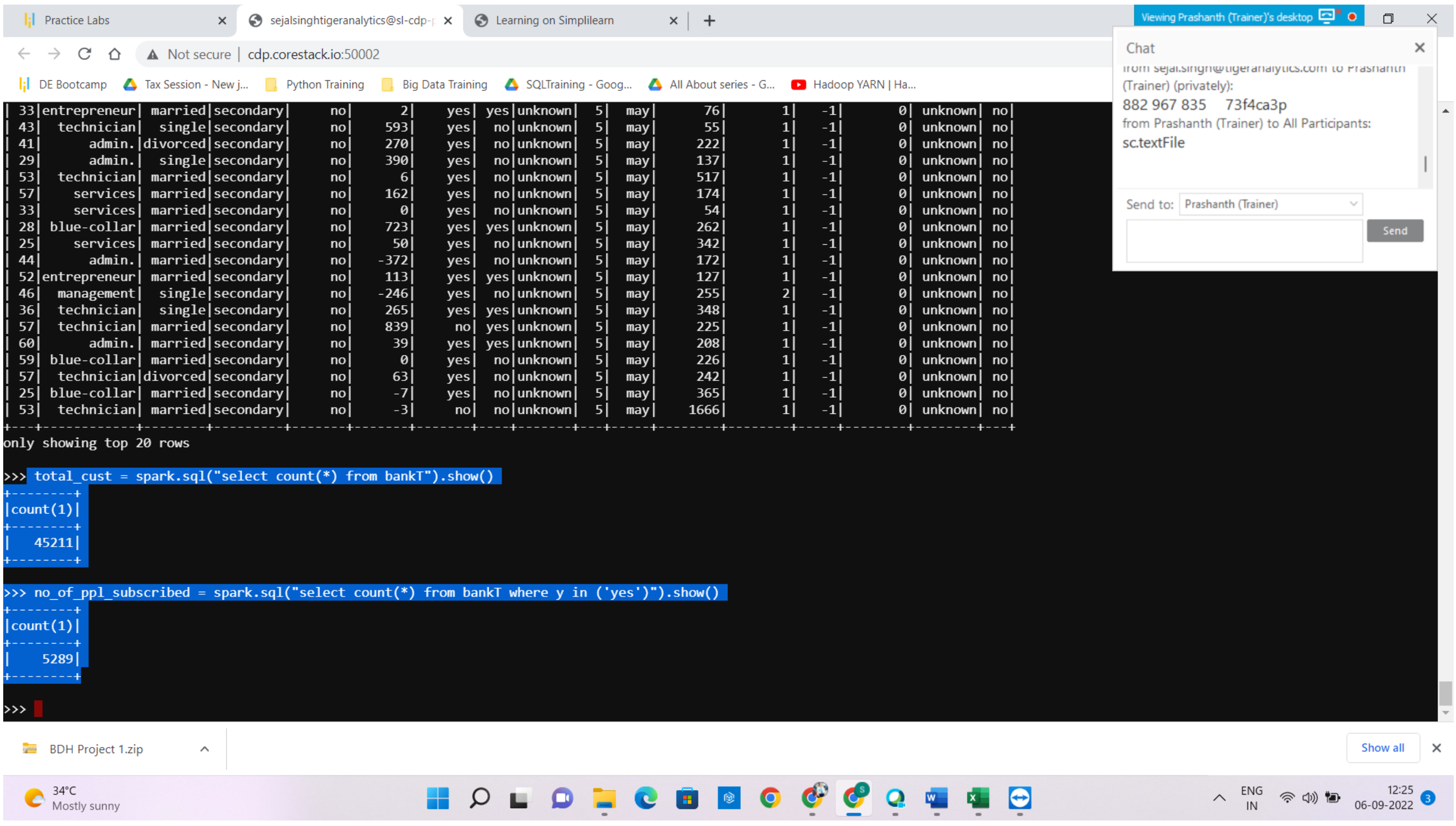


```
>>> bankDF = spark.read.option('header',True).option("inferSchema","true").option("delimiter",";").csv('BDHSProject/Market_Analysis_dataset.csv')
>>> bankDF.show()
+-------------------+-----------+-------------+-----------+-----------+-----------+--------+-------------+-------+---------+------------+------------+--------+------------+-----------
-+-------+
|        "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"""|
```

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

```
total_cust = spark.sql("select * from bankT where education in ('secondary')").show()
```

```
no_of_ppl_subscribed = spark.sql("select count(*) from bankT where y in
('yes')").show()
```



```
| 33|entrepreneur| married|secondary|     no|      2|    yes| yes|unknown|  5|  may|       76|       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|
| 57|    services| married|secondary|     no|    162|    yes|  no|unknown|  5|  may|      174|       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|
| 25|    services| married|secondary|     no|     50|    yes|  no|unknown|  5|  may|      342|       1|   -1|       0| unknown| no|
| 44|      admin.| married|secondary|     no|   -372|    yes|  no|unknown|  5|  may|      172|       1|   -1|       0| unknown| no|
| 52|entrepreneur| married|secondary|     no|    113|    yes| yes|unknown|  5|  may|      127|       1|   -1|       0| unknown| no|
| 46|  management|  single|secondary|     no|   -246|    yes|  no|unknown|  5|  may|      255|       2|   -1|       0| unknown| no|
| 36|  technician|  single|secondary|     no|    265|    yes| yes|unknown|  5|  may|      348|       1|   -1|       0| unknown| no|
| 57|  technician| married|secondary|     no|    839|     no| yes|unknown|  5|  may|      225|       1|   -1|       0| unknown| no|
| 60|      admin.| married|secondary|     no|     39|    yes| yes|unknown|  5|  may|      208|       1|   -1|       0| unknown| no|
| 59| blue-collar| married|secondary|     no|      0|    yes|  no|unknown|  5|  may|      226|       1|   -1|       0| unknown| no|
| 57|  technician|divorced|secondary|     no|     63|    yes|  no|unknown|  5|  may|      242|       1|   -1|       0| unknown| no|
| 25| blue-collar| married|secondary|     no|     -7|    yes|  no|unknown|  5|  may|      365|       1|   -1|       0| unknown| no|
| 53|  technician| married|secondary|     no|     -3|     no|  no|unknown|  5|  may|     1666|       1|   -1|       0| unknown| no|
+---+------------+--------+---------+-------+-------+-------+----+-------+---+-----+---------+--------+-----+--------+--------+---+
only showing top 20 rows

>>> total_cust = spark.sql("select count(*) from bankT").show()
+--------+
|count(1)|
+--------+
|   45211|
+--------+

>>> no_of_ppl_subscribed = spark.sql("select count(*) from bankT where y in ('yes')").show()
+--------+
|count(1)|
+--------+
|    5289|
+--------+

>>> 
```

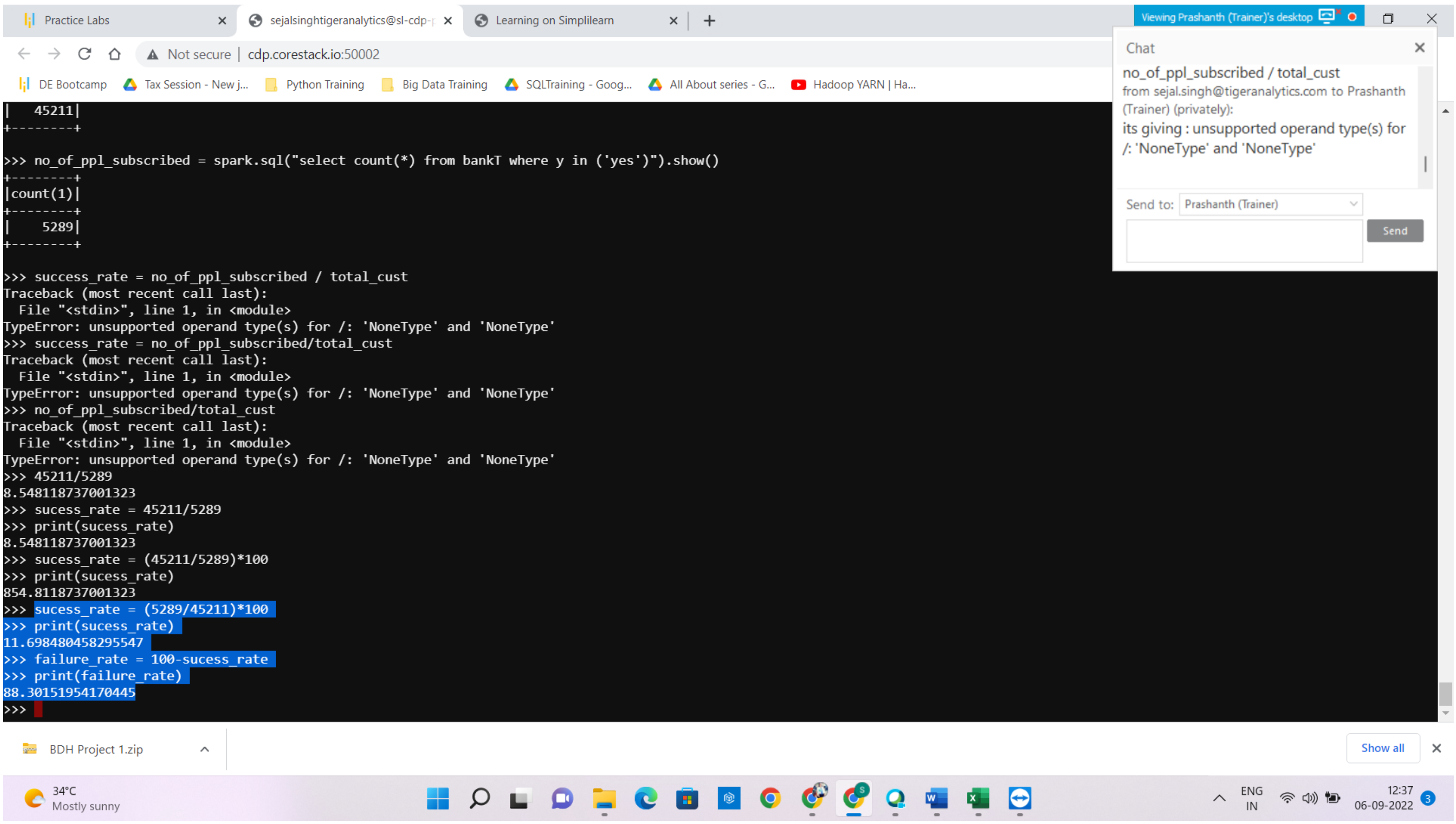
Chat
(Trainer) (privately):
882 967 835   73f4ca3p
from Prashanth (Trainer) to All Participants:
sc.textFile

```
sucess_rate =  (5289/45211)*100
```

```
print(success_rate)
```

- Give marketing failure rate

```
failure_rate = 100-sucess_rate
 print(failure_rate)
```

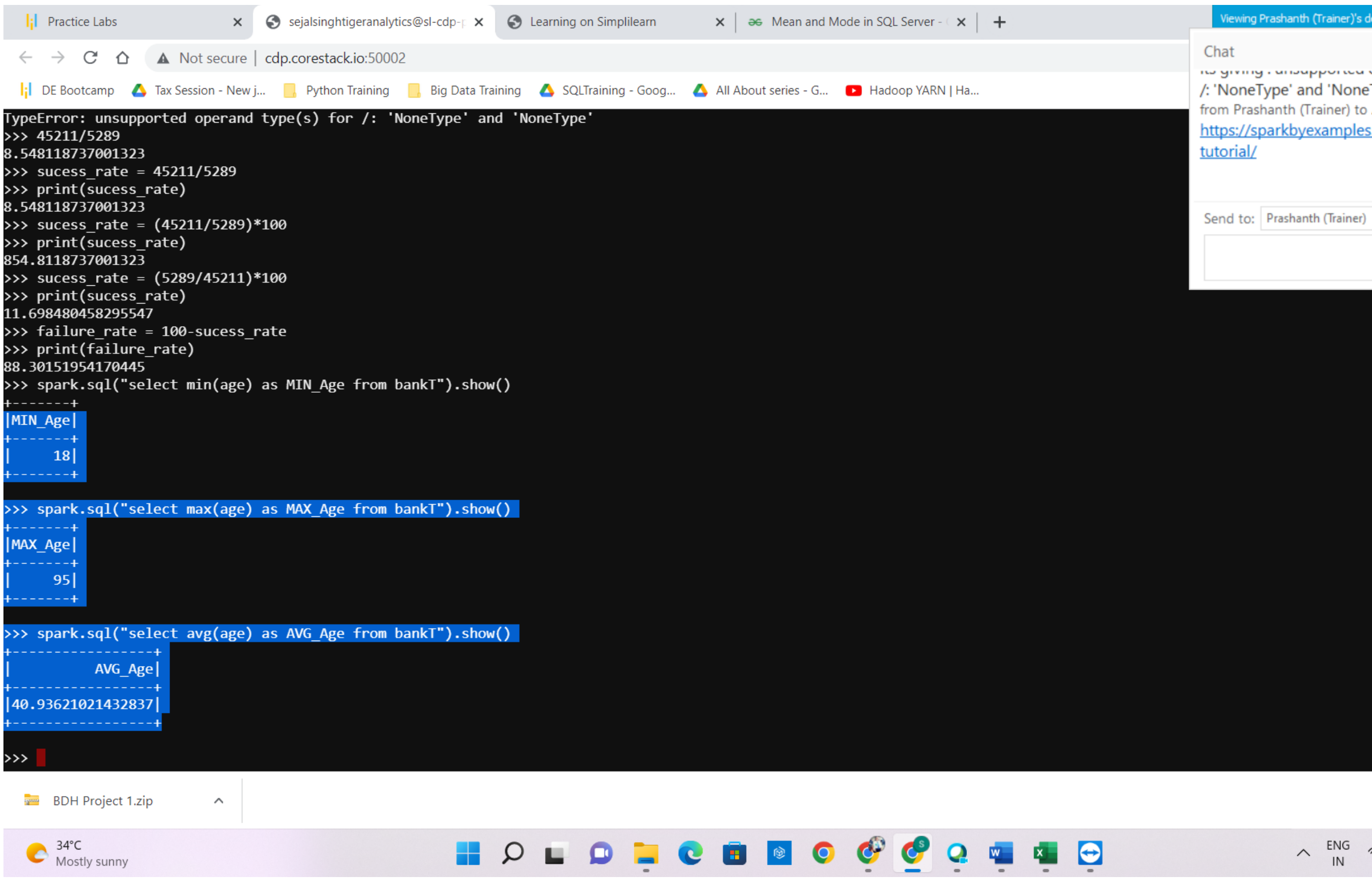


no_of_ppl_subscribed / total_cust
from sejal.singh@tigeranalytics.com to Prashanth (Trainer) (privately):
its giving : unsupported operand type(s) for /: 'NoneType' and 'NoneType'

```
|   45211|
+--------+

>>> no_of_ppl_subscribed = spark.sql("select count(*) from bankT where y in ('yes')").show()
+--------+
|count(1)|
+--------+
|    5289|
+--------+

>>> success_rate = no_of_ppl_subscribed / total_cust
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for /: 'NoneType' and 'NoneType'
>>> success_rate = no_of_ppl_subscribed/total_cust
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for /: 'NoneType' and 'NoneType'
>>> no_of_ppl_subscribed/total_cust
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for /: 'NoneType' and 'NoneType'
>>> 45211/5289
8.548118737001323
>>> sucess_rate = 45211/5289
>>> print(sucess_rate)
8.548118737001323
>>> sucess_rate = (45211/5289)*100
>>> print(sucess_rate)
854.8118737001323
>>> sucess_rate = (5289/45211)*100
>>> print(sucess_rate)
11.698480458295547
>>> failure_rate = 100-sucess_rate
>>> print(failure_rate)
88.30151954170445
>>>
```

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

```
spark.sql("select min(age) as MIN_Age from bankT").show()

spark.sql("select max(age) as MAX_Age from bankT").show()

spark.sql("select avg(age) as AVG_Age from bankT").show()
```



```
TypeError: unsupported operand type(s) for /: 'NoneType' and 'NoneType'
>>> 45211/5289
8.548118737001323
>>> sucess_rate = 45211/5289
>>> print(sucess_rate)
8.548118737001323
>>> sucess_rate = (45211/5289)*100
>>> print(sucess_rate)
854.8118737001323
>>> sucess_rate = (5289/45211)*100
>>> print(sucess_rate)
11.698480458295547
>>> failure_rate = 100-sucess_rate
>>> print(failure_rate)
88.30151954170445
>>> spark.sql("select min(age) as MIN_Age from bankT").show()
+-------+
|MIN_Age|
+-------+
|     18|
+-------+

>>> spark.sql("select max(age) as MAX_Age from bankT").show()
+-------+
|MAX_Age|
+-------+
|     95|
+-------+

>>> spark.sql("select avg(age) as AVG_Age from bankT").show()
+-----------------+
|          AVG_Age|
+-----------------+
|40.93621021432837|
+-----------------+

>>>
```

Chat

/: 'NoneType' and 'None

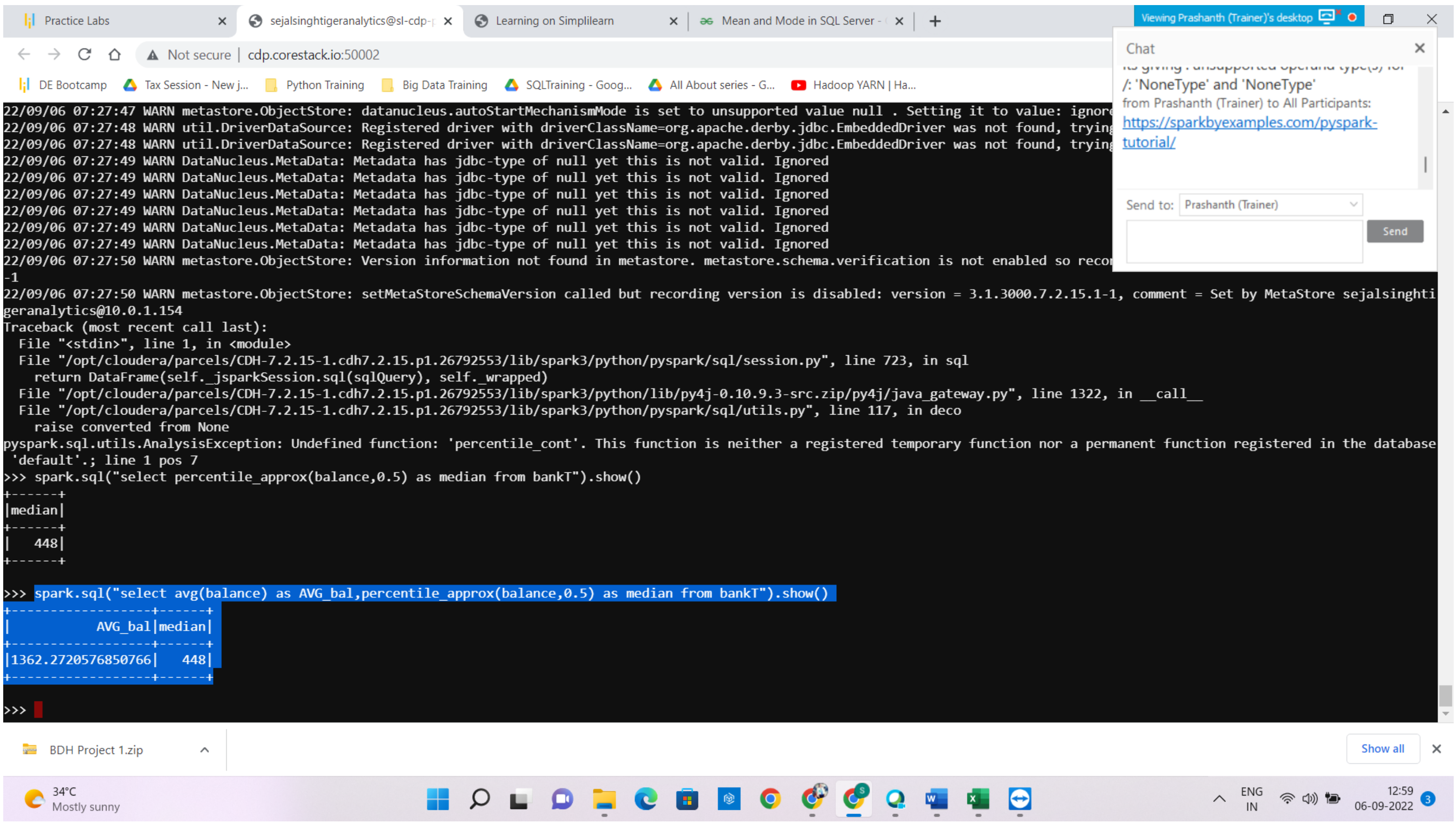
from Prashanth (Trainer) to

https://sparkbyexamples
tutorial/

Send to: Prashanth (Trainer)

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

```
spark.sql("select avg(balance) as AVG_bal,percentile_approx(balance,0.5) as median
from bankT").show()
```



```
22/09/06 07:27:47 WARN metastore.ObjectStore: datanucleus.autoStartMechanismMode is set to unsupported value null . Setting it to value: ignore
22/09/06 07:27:48 WARN util.DriverDataSource: Registered driver with driverClassName=org.apache.derby.jdbc.EmbeddedDriver was not found, trying
22/09/06 07:27:48 WARN util.DriverDataSource: Registered driver with driverClassName=org.apache.derby.jdbc.EmbeddedDriver was not found, trying
22/09/06 07:27:49 WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
22/09/06 07:27:49 WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
22/09/06 07:27:49 WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
22/09/06 07:27:49 WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
22/09/06 07:27:49 WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
22/09/06 07:27:49 WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
22/09/06 07:27:50 WARN metastore.ObjectStore: Version information not found in metastore. metastore.schema.verification is not enabled so reco
-1
22/09/06 07:27:50 WARN metastore.ObjectStore: setMetaStoreSchemaVersion called but recording version is disabled: version = 3.1.3000.7.2.15.1-1, comment = Set by MetaStore sejalsinghti
geranalytics@10.0.1.154
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/opt/cloudera/parcels/CDH-7.2.15-1.cdh7.2.15.p1.26792553/lib/spark3/python/pyspark/sql/session.py", line 723, in sql
    return DataFrame(self._jsparkSession.sql(sqlQuery), self._wrapped)
  File "/opt/cloudera/parcels/CDH-7.2.15-1.cdh7.2.15.p1.26792553/lib/spark3/python/lib/py4j-0.10.9.3-src.zip/py4j/java_gateway.py", line 1322, in __call__
  File "/opt/cloudera/parcels/CDH-7.2.15-1.cdh7.2.15.p1.26792553/lib/spark3/python/pyspark/sql/utils.py", line 117, in deco
    raise converted from None
pyspark.sql.utils.AnalysisException: Undefined function: 'percentile_cont'. This function is neither a registered temporary function nor a permanent function registered in the database
 'default'.; line 1 pos 7
>>> spark.sql("select percentile_approx(balance,0.5) as median from bankT").show()
+------+
|median|
+------+
|   448|
+------+

>>> spark.sql("select avg(balance) as AVG_bal,percentile_approx(balance,0.5) as median from bankT").show()
+------------------+------+
|           AVG_bal|median|
+------------------+------+
|1362.2720576850766|   448|
+------------------+------+

>>>
```

Chat

/: 'NoneType' and 'NoneType'

from Prashanth (Trainer) to All Participants:

https://sparkbyexamples.com/pyspark-tutorial/

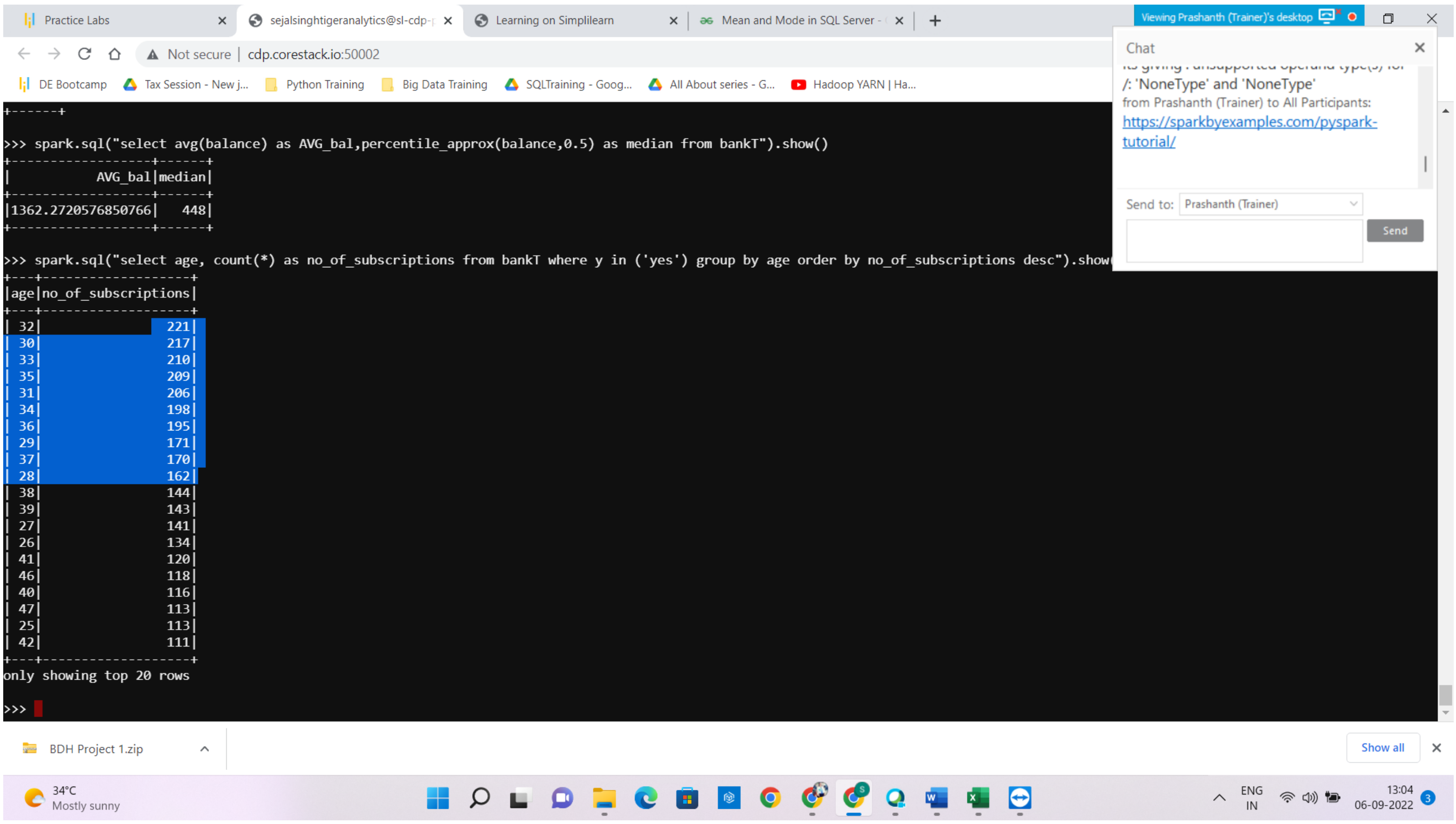
Send to: Prashanth (Trainer)

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

```
spark.sql("select age, count(*) as no_of_subscriptions from bankT where y in ('yes')
group by age order by no_of_subscriptions desc").show()
```

ANALYSIS : Max subscriptions are by people in the age of 32 (i.e. people in their 30s).

As age is deviates from 30s, subscription decreases



```
+------+

>>> spark.sql("select avg(balance) as AVG_bal,percentile_approx(balance,0.5) as median from bankT").show()
+-----------------+------+
|          AVG_bal|median|
+-----------------+------+
|1362.2720576850766|   448|
+-----------------+------+

>>> spark.sql("select age, count(*) as no_of_subscriptions from bankT where y in ('yes') group by age order by no_of_subscriptions desc").show
+---+-------------------+
|age|no_of_subscriptions|
+---+-------------------+
| 32|                221|
| 30|                217|
| 33|                210|
| 35|                209|
| 31|                206|
| 34|                198|
| 36|                195|
| 29|                171|
| 37|                170|
| 28|                162|
| 38|                144|
| 39|                143|
| 27|                141|
| 26|                134|
| 41|                120|
| 46|                118|
| 40|                116|
| 47|                113|
| 25|                113|
| 42|                111|
+---+-------------------+
only showing top 20 rows

>>>
```

Chat

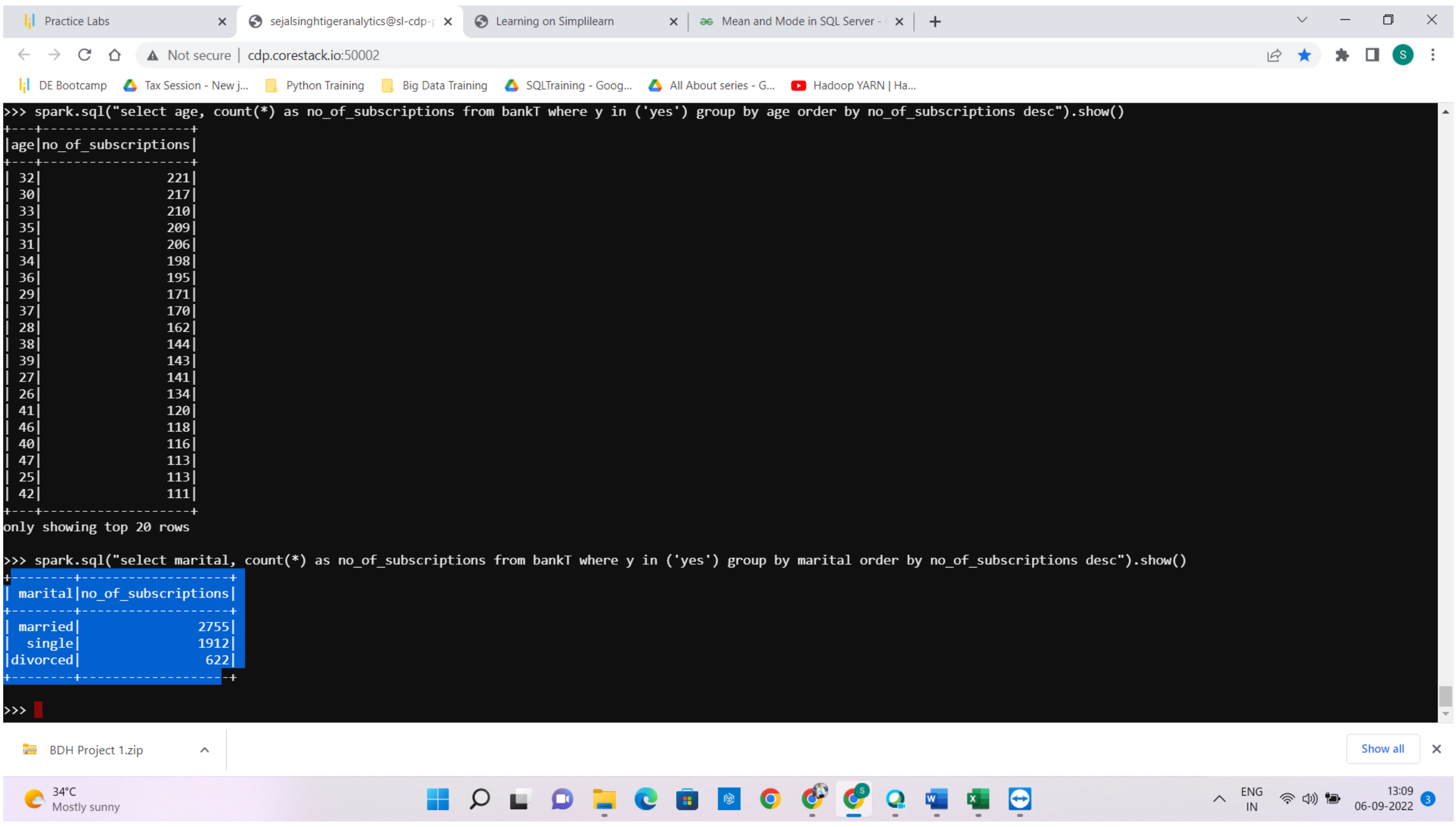
its giving : unsupported operand type(s) for /: 'NoneType' and 'NoneType'

from Prashanth (Trainer) to All Participants:
https://sparkbyexamples.com/pyspark-tutorial/

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

```
spark.sql("select marital, count(*) as no_of_subscriptions from bankT where y in
('yes') group by marital order by no_of_subscriptions desc").show()
```

ANALYSIS : Married people have most subscriptions to deposit

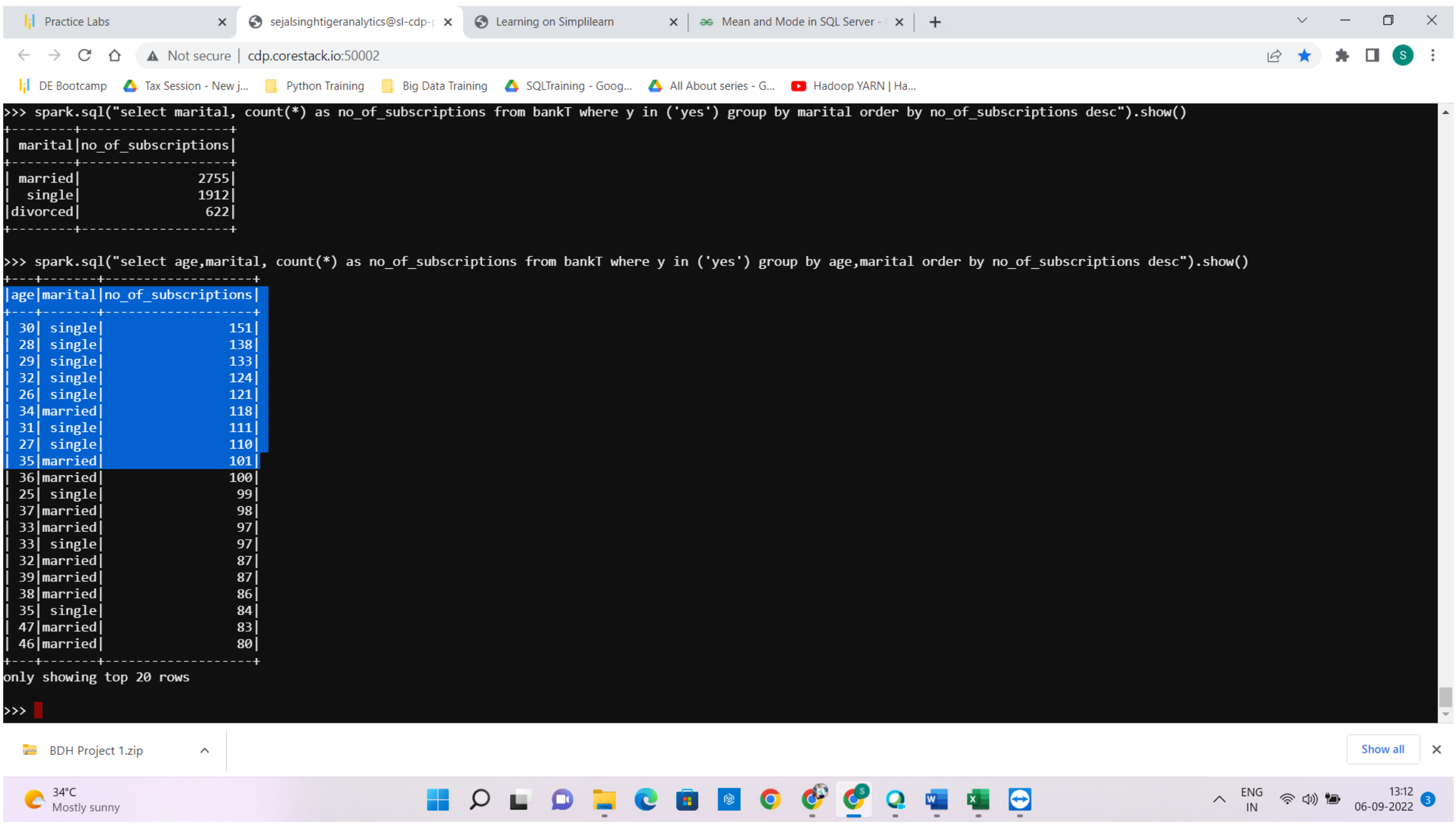


```
>>> spark.sql("select age, count(*) as no_of_subscriptions from bankT where y in ('yes') group by age order by no_of_subscriptions desc").show()
+---+-------------------+
|age|no_of_subscriptions|
+---+-------------------+
| 32|                221|
| 30|                217|
| 33|                210|
| 35|                209|
| 31|                206|
| 34|                198|
| 36|                195|
| 29|                171|
| 37|                170|
| 28|                162|
| 38|                144|
| 39|                143|
| 27|                141|
| 26|                134|
| 41|                120|
| 46|                118|
| 40|                116|
| 47|                113|
| 25|                113|
| 42|                111|
+---+-------------------+
only showing top 20 rows

>>> spark.sql("select marital, count(*) as no_of_subscriptions from bankT where y in ('yes') group by marital order by no_of_subscriptions desc").show()
+--------+-------------------+
| marital|no_of_subscriptions|
+--------+-------------------+
| married|               2755|
|  single|               1912|
|divorced|                622|
+--------+-------------------+

>>>
```

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

ANALYSIS : The analysis shows that most deposits of subscriptions are made by people who are SINGLE in their 30s.

```
spark.sql("select age,marital, count(*) as no_of_subscriptions from bankT where y in
('yes') group by age,marital order by no_of_subscriptions desc").show()
```



```
>>> spark.sql("select marital, count(*) as no_of_subscriptions from bankT where y in ('yes') group by marital order by no_of_subscriptions desc").show()
+--------+-------------------+
| marital|no_of_subscriptions|
+--------+-------------------+
| married|               2755|
|  single|               1912|
|divorced|                622|
+--------+-------------------+

>>> spark.sql("select age,marital, count(*) as no_of_subscriptions from bankT where y in ('yes') group by age,marital order by no_of_subscriptions desc").show()
+---+-------+-------------------+
|age|marital|no_of_subscriptions|
+---+-------+-------------------+
| 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|
| 46|married|                 80|
+---+-------+-------------------+
only showing top 20 rows

>>>
```

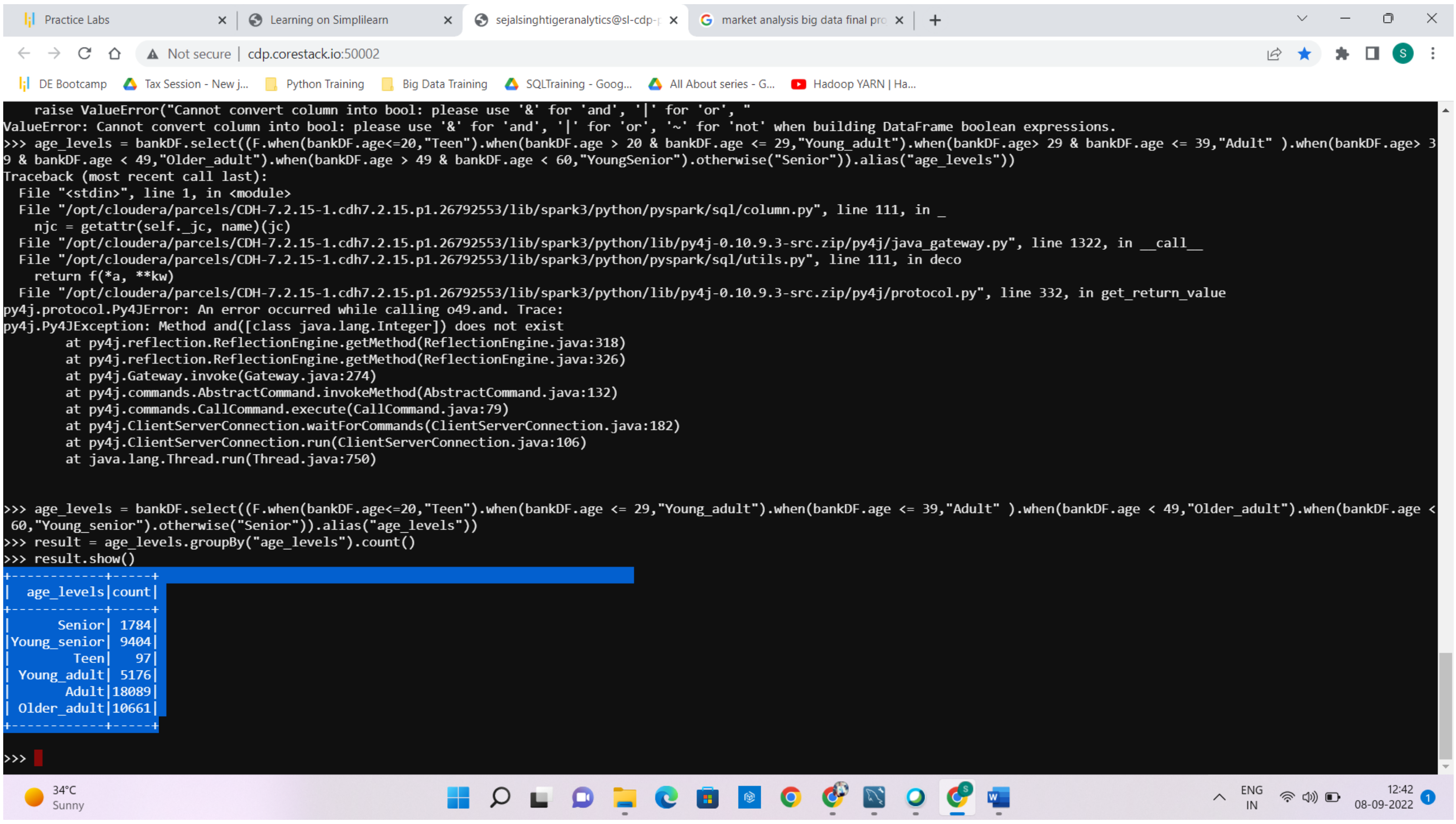
8. Do feature engineering for the bank and find the right age effect on the campaign.

```
from pyspark.sql import functions as F
```

```
age_levels = bankDF.select((F.when(bankDF.age<=20,"Teen").when(bankDF.age <=
29,"Young_adult").when(bankDF.age <= 39,"Adult" ).when(bankDF.age < 49,"Older_adult")
.when(bankDF.age < 60,"Young_senior").otherwise("Senior")).alias("age_levels"))
```

```
result = age_levels.groupBy("age_levels").count()
```

```
result.show()
```



```
    raise ValueError("Cannot convert column into bool: please use '&' for 'and', '|' for 'or', "
ValueError: Cannot convert column into bool: please use '&' for 'and', '|' for 'or', '~' for 'not' when building DataFrame boolean expressions.
>>> age_levels = bankDF.select((F.when(bankDF.age<=20,"Teen").when(bankDF.age > 20 & bankDF.age <= 29,"Young_adult").when(bankDF.age> 29 & bankDF.age <= 39,"Adult" ).when(bankDF.age> 3
9 & bankDF.age < 49,"Older_adult").when(bankDF.age > 49 & bankDF.age < 60,"YoungSenior").otherwise("Senior")).alias("age_levels"))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/opt/cloudera/parcels/CDH-7.2.15-1.cdh7.2.15.p1.26792553/lib/spark3/python/pyspark/sql/column.py", line 111, in _
    njc = getattr(self._jc, name)(jc)
  File "/opt/cloudera/parcels/CDH-7.2.15-1.cdh7.2.15.p1.26792553/lib/spark3/python/lib/py4j-0.10.9.3-src.zip/py4j/java_gateway.py", line 1322, in __call__
  File "/opt/cloudera/parcels/CDH-7.2.15-1.cdh7.2.15.p1.26792553/lib/spark3/python/pyspark/sql/utils.py", line 111, in deco
    return f(*a, **kw)
  File "/opt/cloudera/parcels/CDH-7.2.15-1.cdh7.2.15.p1.26792553/lib/spark3/python/lib/py4j-0.10.9.3-src.zip/py4j/protocol.py", line 332, in get_return_value
py4j.protocol.Py4JError: An error occurred while calling o49.and. Trace:
py4j.Py4JException: Method and([class java.lang.Integer]) does not exist
        at py4j.reflection.ReflectionEngine.getMethod(ReflectionEngine.java:318)
        at py4j.reflection.ReflectionEngine.getMethod(ReflectionEngine.java:326)
        at py4j.Gateway.invoke(Gateway.java:274)
        at py4j.commands.AbstractCommand.invokeMethod(AbstractCommand.java:132)
        at py4j.commands.CallCommand.execute(CallCommand.java:79)
        at py4j.ClientServerConnection.waitForCommands(ClientServerConnection.java:182)
        at py4j.ClientServerConnection.run(ClientServerConnection.java:106)
        at java.lang.Thread.run(Thread.java:750)


>>> age_levels = bankDF.select((F.when(bankDF.age<=20,"Teen").when(bankDF.age <= 29,"Young_adult").when(bankDF.age <= 39,"Adult" ).when(bankDF.age < 49,"Older_adult").when(bankDF.age <
 60,"Young_senior").otherwise("Senior")).alias("age_levels"))
>>> result = age_levels.groupBy("age_levels").count()
>>> result.show()
+------------+-----+
|  age_levels|count|
+------------+-----+
|      Senior| 1784|
|Young_senior| 9404|
|        Teen|   97|
| Young_adult| 5176|
|       Adult|18089|
| Older_adult|10661|
+------------+-----+

>>> 
```