Name – Satyam Kulkarni

HADOOP

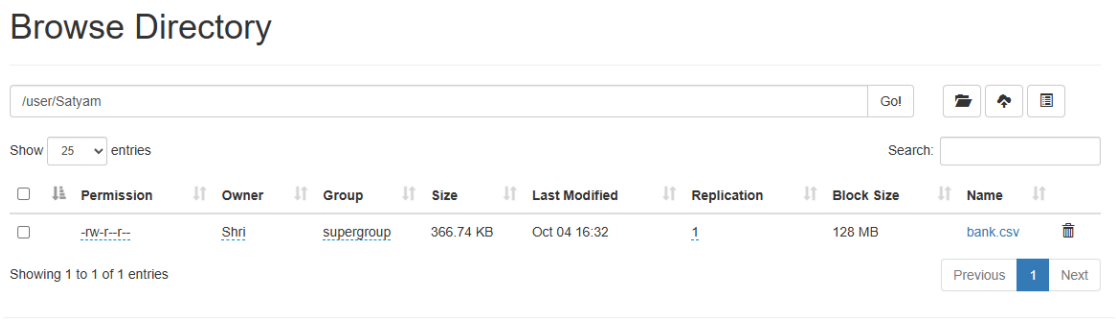
1. Data Ingestion:

- Create a directory in HDFS and transfer the banking dataset from the local system to the HDFS directory.
  Solution – Creation of directory

```
C:\hadoop\sbin>hdfs dfs -mkdir /user/Satyam

C:\hadoop\sbin>hdfs dfs -put C:/dataset/bank.csv /user/Satyam
```

Transfer the banking dataset from the local system to the HDFS directory.



2. Data Transformation with MapReduce:

- Write a MapReduce program in Python that calculates the average account balance for each job type.
  Solution –

```
C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem1/mapper.py /user/Satyam/input1

C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem1/reducer.py /user/Satyam/input1
```

```
C:\hadoop\sbin>hadoop jar C:/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.2.4.jar -files "file:///C:/dataset/Mapreduce/
Problem1/mapper.py,file:///C:/dataset/Mapreduce/Problem1/reducer.py" -mapper "python mapper.py" -reducer "python reducer.py
" -input /user/Satyam/bank.csv -output /user/Satyam/output1
```

```
C:\hadoop\sbin>hdfs dfs -cat /user/Satyam/output1/part-00000
admin.  1226.73640167364
blue-collar     1085.161733615222
entrepreneur    1645.125
housemaid       2083.8035714285716
management      1766.9287925696594
retired 2319.191304347826
self-employed   1392.4098360655737
services        1103.9568345323742
student 1543.8214285714287
technician      1330.99609375
unemployed      1089.421875
unknown 1501.7105263157894
```

Name – Satyam Kulkarni

- Write another MapReduce program that counts the number of individuals with and without a housing loan in each education category.

```
C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem2/mapper.py /user/Satyam/input2
put: `/user/Satyam/input2/mapper.py': File exists

C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem2/reducer.py /user/Satyam/input2
```

```
C:\hadoop\sbin>hadoop jar C:/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.2.4.jar -files "file:///C:/dataset/Mapreduce/
Problem2/mapper.py,file:///C:/dataset/Mapreduce/Problem2/reducer.py" -mapper "python mapper.py" -reducer "python reducer.py
" -input /user/Satyam/bank.csv -output /user/Satyam/output2
```

```
C:\hadoop\sbin>hdfs dfs -cat /user/Satyam/output2/part-00000
primary 94      583
secondary       416     1889
tertiary        173     1176
unknown 7       179
```

- Perform a MapReduce job to determine the number of clients contacted in each month and their subscription status to term deposits ('y' column).

```
C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem3/mapper.py /user/Satyam/input3

C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem3/reducer.py /user/Satyam/input3
```

```
C:\hadoop\sbin>hadoop jar C:/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.2.4.jar -files "file:///C:/dataset/Mapreduce/
Problem3/mapper.py,file:///C:/dataset/Mapreduce/Problem3/reducer.py" -mapper "python mapper.py" -reducer "python reducer.py
" -input /user/Satyam/bank.csv -output /user/Satyam/output3
```

```
C:\hadoop\sbin>hdfs dfs -cat /user/Satyam/output3/part-00000
apr     56      236
aug     79      553
dec     8       11
feb     38      183
jan     16      131
jul     61      644
jun     55      475
mar     21      27
may     93      1304
nov     38      350
oct     37      42
sep     16      35
```

3. Data Analysis with MapReduce:

- Analyze the average duration of contact (in seconds) per campaign outcome ('poutcome').

```
C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem4/mapper.py /user/Satyam/input4

C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem4/reducer.py /user/Satyam/input4
```

```
C:\hadoop\sbin>hadoop jar C:/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.2.4.jar -files "file:///C:/dataset/Mapreduce/
Problem4/mapper.py,file:///C:/dataset/Mapreduce/Problem4/reducer.py" -mapper "python mapper.py" -reducer "python reducer.py
" -input /user/Satyam/bank.csv -output /user/Satyam/output4
```

Name – Satyam Kulkarni

```
C:\hadoop\sbin>hdfs dfs -cat /user/Satyam/output4/part-00000
failure 254.38367346938776
other   273.83248730964465
success 338.6356589147287
unknown 262.1031039136302
```

- Examine the relationship between the age of clients and their balance, and present findings in a summarized form.

```
C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem5/mapper.py /user/Satyam/input5

C:\hadoop\sbin>hdfs dfs -put C:/dataset/Mapreduce/Problem5/reducer.py /user/Satyam/input5
```

```
C:\hadoop\sbin>hadoop jar C:/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.2.4.jar -files "file:///C:/dataset/Mapreduce/
Problem5/mapper.py,file:///C:/dataset/Mapreduce/Problem5/reducer.py" -mapper "python mapper.py" -reducer "python reducer.py
" -input /user/Satyam/bank.csv -output /user/Satyam/output5
```

```
C:\hadoop\sbin>hdfs dfs -cat /user/Satyam/output5/part-00000
19      393.5
20      661.3333333333334
21      1774.2857142857142
22      1455.3333333333333
23      2117.95
24      634.625
25      1240.0681818181818
26      788.5584415584416
27      851.7765957446809
28      1025.0970873786407
29      1261.8762886597938
30      1113.0333333333333
31      1288.4824120603016
32      1256.549107142857
33      1545.4139784946237
34      1111.5367965367966
35      1192.8277777777778
36      1226.8936170212767
37      1463.9192546583852
38      1718.993710691824
39      1104.8615384615384
40      1399.5070422535211
41      1505.7925925925927
42      1612.3617021276596
43      1807.8347826086956
44      1836.5523809523809
45      1187.3660714285713
46      998.7731092436975
47      1363.0462962962963
48      1462.359649122807
49      1591.107142857143
50      1645.0594059405942
51      1528.5714285714287
52      782.2906976744187
53      1588.3085106382978
54      1656.661971830986
55      1244.9444444444443
56      2120.135135135135
```

Name – Satyam Kulkarni

HIVE

1. **Data Ingestion and Table Creation**:
    - Create a Hive database named **banking_data**.
    - Define and create a Hive table **client_info** with appropriate data types for the **bank.csv** dataset.

```
hive> CREATE TABLE client_info(
    > 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
    > )
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE;
2024-10-05T15:41:42,467 INFO [mai
40dcc72b-e2d0-48e9-97b4-9f1ef28cd
2024-10-05T15:41:42,469 INFO [mai
0-48e9-97b4-9f1ef28cdc23 main
OK
```

    - Load the data from the **bank.csv** file into the **client_info** table.

```
hive> LOAD DATA INPATH '/user/Satyam/bank.csv' INTO TABLE client_info;
2024-10-05T15:58:30,054 INFO [main] org.apache.hadoop.hive.conf.HiveConf
40dcc72b-e2d0-48e9-97b4-9f1ef28cdc23
2024-10-05T15:58:30,060 INFO [main] org.apache.hadoop.hive.ql.session.Ses
0-48e9-97b4-9f1ef28cdc23 main
Loading data to table banking_data.client_info
OK
```

```
hive> SELECT * FROM client_info LIMIT 10;
```

Name – Satyam Kulkarni

```
NULL    job     marital education       default NULL    housing loan    contact NULL    month   NULL    NULL    NULL    N
ULL     poutcome        y
30      unemployed      married primary no      1787    no      no      cellular        19      oct     79      1       -
1       0       unknown no
33      services        married secondary       no      4789    yes     yes     cellular        11      may     220     1
        339     4       failure no
35      management      single  tertiary        no      1350    yes     no      cellular        16      apr     185     1
        330     1       failure no
30      management      married tertiary        no      1476    yes     yes     unknown 3       jun     199     4       -
1       0       unknown no
59      blue-collar     married secondary       no      0       yes     no      unknown 5       may     226     1       -
1       0       unknown no
35      management      single  tertiary        no      747     no      no      cellular        23      feb     141     2
        176     3       failure no
36      self-employed   married tertiary        no      307     yes     no      cellular        14      may     341     1
        330     2       other   no
39      technician      married secondary       no      147     yes     no      cellular        6       may     151     2
        -1      0       unknown no
41      entrepreneur    married tertiary        no      221     yes     no      unknown 14      may     57      2       -
1       0       unknown no
Time taken: 6.784 seconds, Fetched: 10 row(s)
```

2. **Basic Data Exploration**:
   - Write a HiveQL query to count the total number of clients in the dataset.
     Ans:

```
hive> select count(*) as total_clients from client_info;
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU
Total MapReduce CPU Time Spent: 12 seconds 246 ms
OK
4522
Time taken: 107.457 seconds, Fetched: 1 row(s)
```

   - Display the first 10 rows of the dataset.
     Ans:

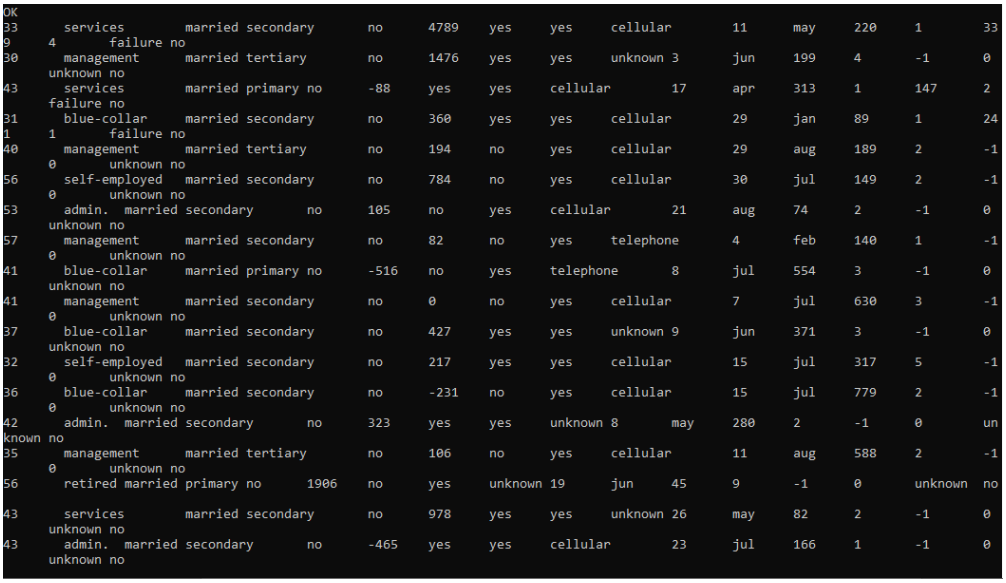
```
NULL    job     marital education       default NULL    housing loan    contact NULL    month   NULL    NULL    NULL    N
ULL     poutcome        y
30      unemployed      married primary no      1787    no      no      cellular        19      oct     79      1       -
1       0       unknown no
33      services        married secondary       no      4789    yes     yes     cellular        11      may     220     1
        339     4       failure no
35      management      single  tertiary        no      1350    yes     no      cellular        16      apr     185     1
        330     1       failure no
30      management      married tertiary        no      1476    yes     yes     unknown 3       jun     199     4       -
1       0       unknown no
59      blue-collar     married secondary       no      0       yes     no      unknown 5       may     226     1       -
1       0       unknown no
35      management      single  tertiary        no      747     no      no      cellular        23      feb     141     2
        176     3       failure no
36      self-employed   married tertiary        no      307     yes     no      cellular        14      may     341     1
        330     2       other   no
39      technician      married secondary       no      147     yes     no      cellular        6       may     151     2
        -1      0       unknown no
41      entrepreneur    married tertiary        no      221     yes     no      unknown 14      may     57      2       -
1       0       unknown no
Time taken: 6.784 seconds, Fetched: 10 row(s)
```

   -

3. **Data Filtering and Sorting**:
   - Retrieve all records of clients who are married and have a personal loan.
     Ans:

```
hive> select * from client_info where marital = 'married' and loan = 'yes';
```

Name – Satyam Kulkarni



```
OK
33      services        married secondary       no      4789    yes     yes     cellular        11      may     220     1       33
9       4       failure no
30      management      married tertiary        no      1476    yes     yes     unknown 3       jun     199     4       -1      0
      unknown no
43      services        married primary no      -88     yes     yes     cellular        17      apr     313     1       147     2
      failure no
31      blue-collar     married secondary       no      360     yes     yes     cellular        29      jan     89      1       24
1       1       failure no
40      management      married tertiary        no      194     no      yes     cellular        29      aug     189     2       -1
      0       unknown no
56      self-employed   married secondary       no      784     no      yes     cellular        30      jul     149     2       -1
      0       unknown no
53      admin.  married secondary       no      105     no      yes     cellular        21      aug     74      2       -1      0
      unknown no
57      management      married secondary       no      82      no      yes     telephone       4       feb     140     1       -1
      0       unknown no
41      blue-collar     married primary no      -516    no      yes     telephone       8       jul     554     3       -1      0
      unknown no
41      management      married secondary       no      0       no      yes     cellular        7       jul     630     3       -1
      0       unknown no
37      blue-collar     married secondary       no      427     yes     yes     unknown 9       jun     371     3       -1      0
      unknown no
32      self-employed   married secondary       no      217     yes     yes     cellular        15      jul     317     5       -1
      0       unknown no
36      blue-collar     married secondary       no      -231    no      yes     cellular        15      jul     779     2       -1
      0       unknown no
42      admin.  married secondary       no      323     yes     yes     unknown 8       may     280     2       -1      0       un
known no
35      management      married tertiary        no      106     no      yes     cellular        11      aug     588     2       -1
      0       unknown no
56      retired married primary no      1906    no      yes     unknown 19      jun     45      9       -1      0       unknown  no

43      services        married secondary       no      978     yes     yes     unknown 26      may     82      2       -1      0
      unknown no
43      admin.  married secondary       no      -465    yes     yes     cellular        23      jul     166     1       -1      0
      unknown no
Time taken: 1.168 seconds, Fetched: 453 row(s)
```

- List the top 10 clients with the highest balance, displaying their job, marital status, and balance.
  Ans:

```
hive> select job,marital,balance from client_info order by balance desc limit 10;
```

```
OK
retired married 71188
entrepreneur    married 42045
technician      single  27733
management      married 27359
technician      married 27069
housemaid       single  26965
retired married 26452
services        married 26394
management      divorced        26306
retired single  25824
Time taken: 85.493 seconds, Fetched: 10 row(s)
```

4. **Data Aggregation and Grouping**:
   - Calculate the average age of clients for each job category.
     Ans:

```
hive> select job, avg(age) as average_age from client_info group by job;
```

Name – Satyam Kulkarni

```
OK
admin.  39.68200836820084
blue-collar     40.15644820295983
entrepreneur    42.01190476190476
housemaid       47.339285714285715
job     NULL
management      40.54076367389061
retired 61.869565217391305
self-employed   41.45355191256831
services        38.57074340527578
student 26.821428571428573
technician      39.470052083333336
unemployed      40.90625
unknown 48.10526315789474
Time taken: 123.358 seconds, Fetched: 13 row(s)
```

- Find the total number of clients for each education level who have defaulted on credit.
  Ans:

```
hive> select education, default, count(*) as total_defaulted_client from client_info where default = 'yes' group by education, defa
ult
    > ;
```

```
OK
primary yes     10
secondary       yes     46
tertiary        yes     17
unknown yes     3
Time taken: 95.775 seconds, Fetched: 4 row(s)
```

5. **Complex Queries for Insights**:
   - Identify the top 5 job categories with the highest average balance and the percentage of clients in each of these job categories who have subscribed to a term deposit.
     Ans:
     We directly calculate the average balance (AVG(ci.balance)) for each job category.
     We also calculate the total number of clients (COUNT(*)) and the number of clients who have subscribed to a term deposit (SUM(CASE WHEN ci.y = 'yes' THEN 1 ELSE 0 END)).
     We calculate the subscription rate as the percentage of subscribed clients out of the total number of clients.
     The results are grouped by job category and sorted in descending order of average balance.
     We limit the output to the top 5 job categories with the highest average balance using the LIMIT clause.
     This query will give you the top 5 job categories with the highest average balance and the percentage of clients in each of these job categories who have subscribed to a term deposit.

```
hive> select s.job, s.avg_balance, (s.subscribed_clients/s.total_clients)*100 as subscription_percentage from (select job, avg(bala
nce) as avg_balance, count(*) as total_clients, sum(case when y = 'yes' then 1 else 0 end) as subscribed_clients from client_info g
roup by job order by avg_balance desc limit 5) s;
```

Name – Satyam Kulkarni

```
OK
retired 2319.191304347826       23.47826086956522
housemaid       2083.8035714285716      12.5
management      1766.9287925696594      13.519091847265221
entrepreneur    1645.125        8.928571428571429
student 1543.8214285714287      22.61904761904762
Time taken: 172.02 seconds, Fetched: 5 row(s)
```

- Determine the month with the highest number of contacts and the success rate of the campaign in that month (percentage of clients who subscribed to a term deposit).
  Ans:
  We directly calculate the success rate for each month by counting the total number of contacts and the number of clients who subscribed to a term deposit within each month.
  We use the GROUP BY clause to group the data by month.
  The results are sorted in descending order of the number of contacts, and we limit the output to only the first row, which represents the month with the highest number of contacts.
  This query will give you the month with the highest number of contacts and the success rate of the campaign in that month.

```
hive> select month, total_contacts, (successful_contacts/total_contacts)*100 as success_rate from (select month, count(*) as total_
contacts, sum(case when y = 'yes' then 1 else 0 end) as successful_contacts from client_info group by month order by total_contacts
 desc limit 1) as top_month;
```

```
OK
may     1398    6.652360515021459
Time taken: 172.717 seconds, Fetched: 1 row(s)
```

6. **Correlation Analysis**:
   - Calculate the correlation between age and balance for the clients.
     Ans:

     In this simplified query:
     1. We calculate the numerator of the correlation formula: the sum of the products of age and balance minus the count of records times the average of age times the average of balance.
     2. We calculate the denominator of the correlation formula: the square root of the difference between the sum of the squares of age and the count times the square of the average of age, multiplied by the square root of the difference between the sum of the squares of balance and the count times the square of the average of balance.
     3. We divide the numerator by the denominator to get the correlation coefficient.

```
hive> select corr(age, balance) as age_balance_correlation from client_info;
```

```
OK
0.08382014224477742
Time taken: 94.424 seconds, Fetched: 1 row(s)
```

Name – Satyam Kulkarni

7. **Trend Analysis**:
   - Analyze the year-over-year trend in the number of clients contacted.
     Ans:

     In this query:
     We use the SUBSTRING function to extract the year from the month column. Assuming the month column is in the format "YYYY-MM", we extract the first four characters to get the year.
     We count the number of clients contacted (COUNT(*)) for each year.
     We group the results by the extracted year.
     Finally, we order the results by year to see the trend over time.
     This query will give you the year-over-year trend in the number of clients contacted based on the data in your client_info table.

```
hive> select substring(month,1,4) as year, count(*) as num_clients_contacted from client_info group by substring(month,1,4) order b
y year;
```

```
OK
apr     293
aug     633
dec     20
feb     222
jan     148
jul     706
jun     531
mar     49
may     1398
mont    1
nov     389
oct     80
sep     52
Time taken: 163.069 seconds, Fetched: 13 row(s)
```

8. **Anomaly Detection**:
   - Identify any unusual patterns in the average yearly balance across different education levels.
     Ans:
     We calculate the average yearly balance for each education level.
     We also calculate the overall average yearly balance for each year using the window function AVG(AVG(balance)) OVER (PARTITION BY SUBSTRING(month, 1, 4)).
     We calculate the z-score for each average yearly balance within each education level by subtracting the overall average yearly balance and dividing by the standard deviation, both calculated over the partition of years.
     By examining the z-scores, you can identify any unusual patterns in the average yearly balance across different education levels. A z-score significantly higher or lower than zero indicates that the average yearly balance for a particular education level is unusually high or low compared to the overall average yearly balance.

```
hive> select year, education, (avg_yearly_balance - overall_avg_balance) / stddev_balance as z_score from (select substring(month,1
,4) as year, education, avg(balance) as avg_yearly_balance, avg(avg(balance)) over (partition by substring(month,1,4)) as overall_a
vg_balance, STDDEV(avg(balance)) over (partition by substring(month,1,4)) as stddev_balance from client_info group by substring(mon
th,1,4), education) as subquery;
```

Name – Satyam Kulkarni

```
OK
apr     primary 1.267794378462834
apr     secondary       -0.7235770026264813
apr     tertiary        0.6555354947097891
apr     unknown -1.1997528705461413
aug     primary -0.8113812295586758
aug     secondary       -0.5775732890112949
aug     tertiary        -0.31633238372712924
aug     unknown 1.7052869022970998
dec     primary -0.8088635843923103
dec     secondary       -0.397568378390461
dec     tertiary        -0.5058241544774784
dec     unknown 1.7122561172602497
feb     primary -1.0701923004119014
feb     secondary       -0.9216091477916096
feb     tertiary        0.8917649610011248
feb     unknown 1.1000364872023862
jan     primary 1.1404156384061253
jan     secondary       -0.4369029752604172
jan     tertiary        0.7115176049750513
jan     unknown -1.41503026812076
jul     primary 0.9649291030842655
jul     secondary       -0.4258456259680814
jul     tertiary        0.9014112560610348
jul     unknown -1.4404947331772213
jun     primary -0.15682779176634196
jun     secondary       -1.0365919570822748
jun     tertiary        1.6428352994501707
jun     unknown -0.44941555060155386
mar     primary -1.0722876073529977
mar     secondary       -0.2462824810074777
mar     tertiary        1.6391398006892093
mar     unknown -0.3205697123287344
may     primary -1.0048248682462826
may     secondary       -0.9457054758622359
may     tertiary        0.6640708393343474
may     unknown 1.2864595047741691
```

```
mont    education       NULL
nov     primary -0.8137802862217836
nov     secondary       0.2770762379753206
nov     tertiary        1.5167439470393023
nov     unknown -0.9800398987928378
oct     primary 1.7097889755337383
oct     secondary       -0.37052787477325194
oct     tertiary        -0.5238214197432853
oct     unknown -0.8154396810172013
sep     primary -1.3086820436917062
sep     secondary       -0.4707154583719365
sep     tertiary        0.3984500713422833
sep     unknown 1.38094743072136
Time taken: 94.281 seconds, Fetched: 49 row(s)
```

Name – Satyam Kulkarni

9. **Advanced Analysis**:
    - Analyze the impact of previous campaign outcomes (**poutcome**) on the current campaign's success. Calculate the subscription rate (to term deposits) for each **poutcome** category.
      Ans:

      In this query:

      1. We group the data by the poutcome column to analyze the impact of previous
      campaign outcomes.
      2. We count the total number of clients (total_clients) and the number of clients who
      subscribed to term deposits (subscribed_clients) for each poutcome category.
      3. We calculate the subscription rate (subscription_rate) as the percentage of clients
      who subscribed to term deposits out of the total number of clients for each poutcome
      category.

```
hive> select poutcome, count(*) as total_clients, sum(case when y = 'yes' then 1 else 0 end) as subscribed_clients, round(sum(case
when y = 'yes' then 1 else 0 end) * 100.0/count(*),2) as subscription_rate from client_info group by poutcome order by subscription
_rate desc;
```

```
OK
success 129     83      64.34
other   197     38      19.29
failure 490     63      12.86
unknown 3705    337     9.10
poutcome        1       0       0.00
Time taken: 170.274 seconds, Fetched: 5 row(s)
```

    - Compare the average contact duration for clients who subscribed and who did not subscribe to a term deposit.

      Ans:

      In this query:

      1. We group the data by the subscription status (y column), which indicates

      whether the client subscribed to a term deposit.

      2. We calculate the average contact duration (avg_contact_duration) for each

      group separately.

      This query will provide you with the average contact duration for clients who

Name – Satyam Kulkarni

subscribed and who did not subscribe to a term deposit, allowing you to compare the

contact durations between the two groups.

```
hive> select y as subscription_status, avg(duration) as avg_contact_duration from client_info group by y;
```

```
OK
no      226.3475
y       NULL
yes     552.7428023032629
Time taken: 87.218 seconds, Fetched: 3 row(s)
```

Submission Guidelines:

- Make a copy of this doc file.
- Perform the analysis in your local system using Hadoop and Hive and provide screenshots of both the **code** and the **output** under each question.
- Upload the doc file with other files and submit it in the submission dashboard.