Data Scientist Application

Muhammad Jauhar Hakim

PT ITSEC Asia Tbk.

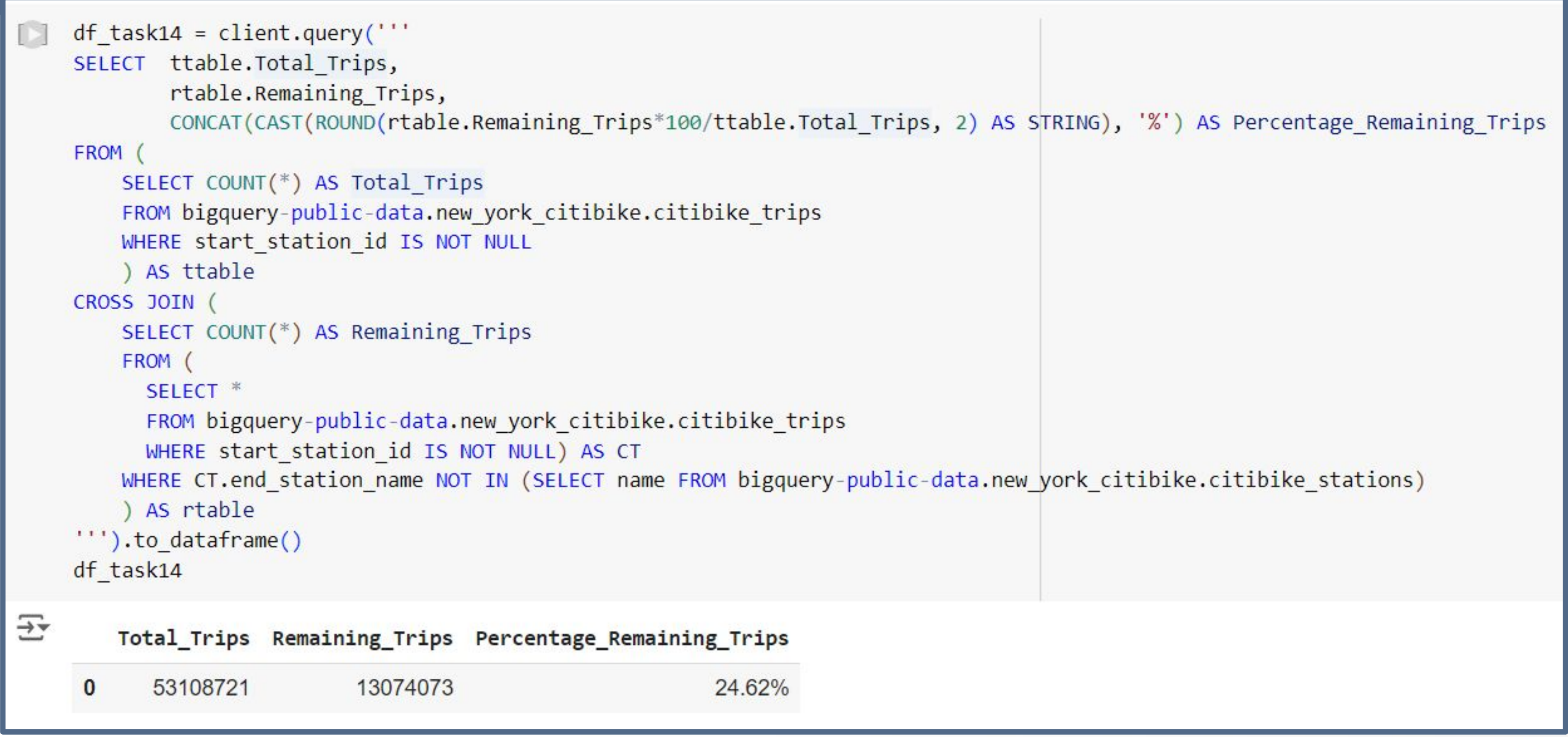
GitHub Link for the test :
https://github.com/Jauhar-Hakim/Big-Query-SQL

01

Task 1

- What proportion in percentage of remaining trips terminate at end_station_id that likewise do not exist in the station table?

- You should eliminate the journeys from the trip table that lack a start_station_id, in remaining trips, keep those with start_station_id that were note present at the station table.



```
df_task14 = client.query('''
SELECT  ttable.Total_Trips,
        rtable.Remaining_Trips,
        CONCAT(CAST(ROUND(rtable.Remaining_Trips*100/ttable.Total_Trips, 2) AS STRING), '%') AS Percentage_Remaining_Trips
FROM (
    SELECT COUNT(*) AS Total_Trips
    FROM bigquery-public-data.new_york_citibike.citibike_trips
    WHERE start_station_id IS NOT NULL
    ) AS ttable
CROSS JOIN (
    SELECT COUNT(*) AS Remaining_Trips
    FROM (
      SELECT *
      FROM bigquery-public-data.new_york_citibike.citibike_trips
      WHERE start_station_id IS NOT NULL) AS CT
    WHERE CT.end_station_name NOT IN (SELECT name FROM bigquery-public-data.new_york_citibike.citibike_stations)
    ) AS rtable
''').to_dataframe()
df_task14
```

	Total_Trips	Remaining_Trips	Percentage_Remaining_Trips
0	53108721	13074073	24.62%

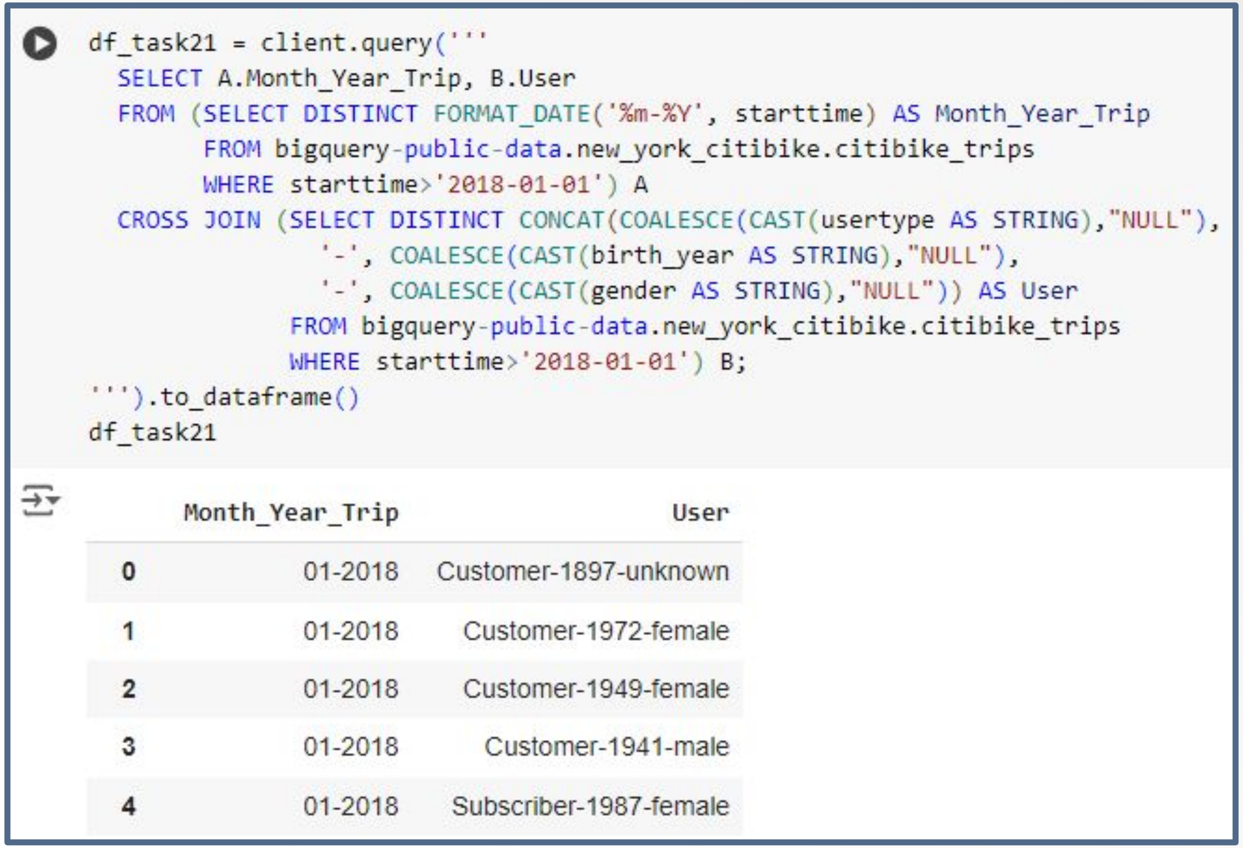
We can conclude the percentage of remaining trips terminate at end_station_id that likewise do not exist in the station table around 24.62%

02

Task 2

- a. In each month of 2018, what is the user count in each segment?

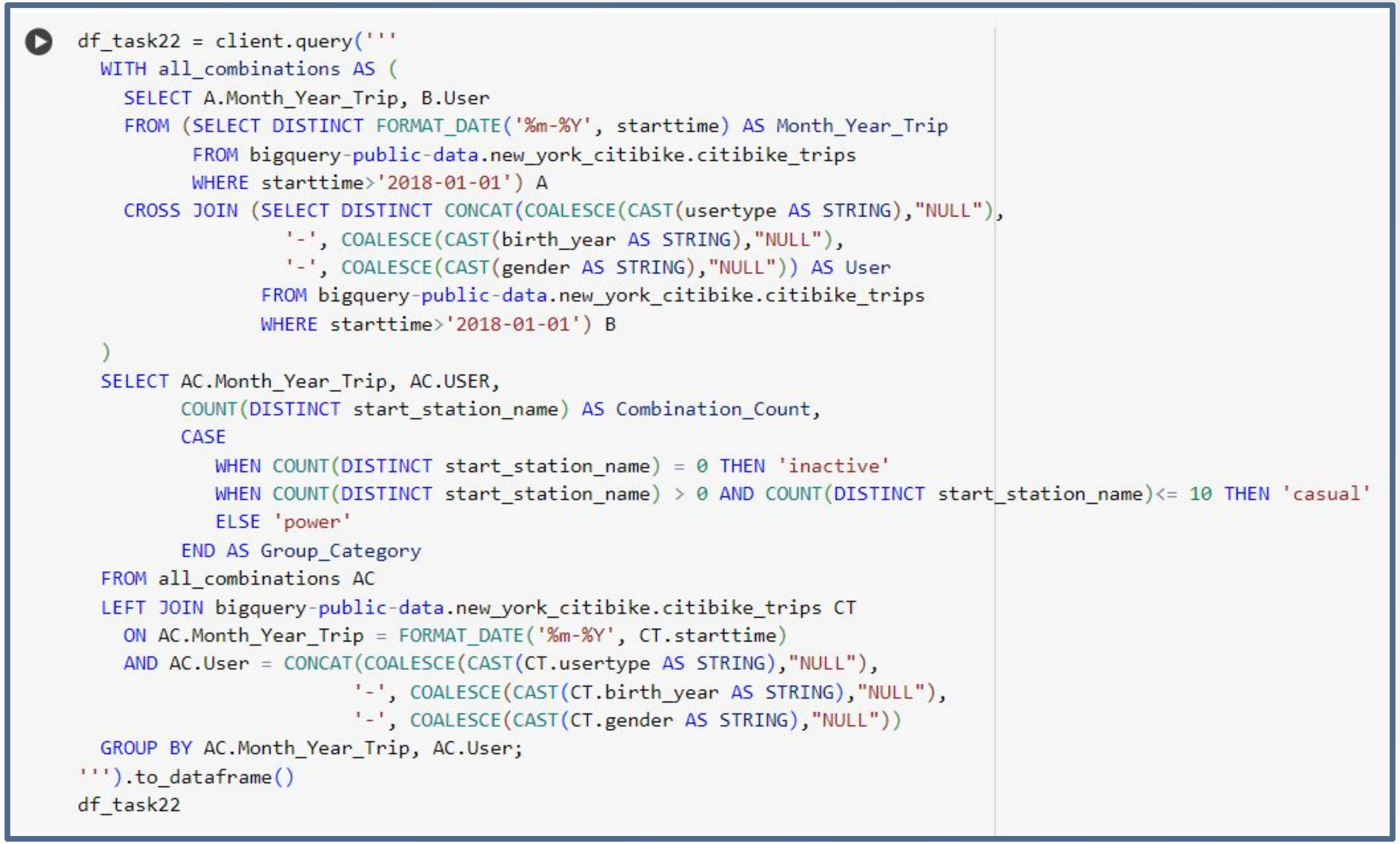
- b. For each month of 2018, determine the shift of users among the segments in the following month. As an example: From January 2018 to February 2018, quantify how many casual users remained casual, transitioned to power, or became inactive? Repeat this process for the other categories and for the rest of the months in 2018.



```
df_task21 = client.query('''
  SELECT A.Month_Year_Trip, B.User
  FROM (SELECT DISTINCT FORMAT_DATE('%m-%Y', starttime) AS Month_Year_Trip
        FROM bigquery-public-data.new_york_citibike.citibike_trips
        WHERE starttime>'2018-01-01') A
  CROSS JOIN (SELECT DISTINCT CONCAT(COALESCE(CAST(usertype AS STRING),"NULL"),
                '-', COALESCE(CAST(birth_year AS STRING),"NULL"),
                '-', COALESCE(CAST(gender AS STRING),"NULL")) AS User
              FROM bigquery-public-data.new_york_citibike.citibike_trips
              WHERE starttime>'2018-01-01') B;
''').to_dataframe()
df_task21
```

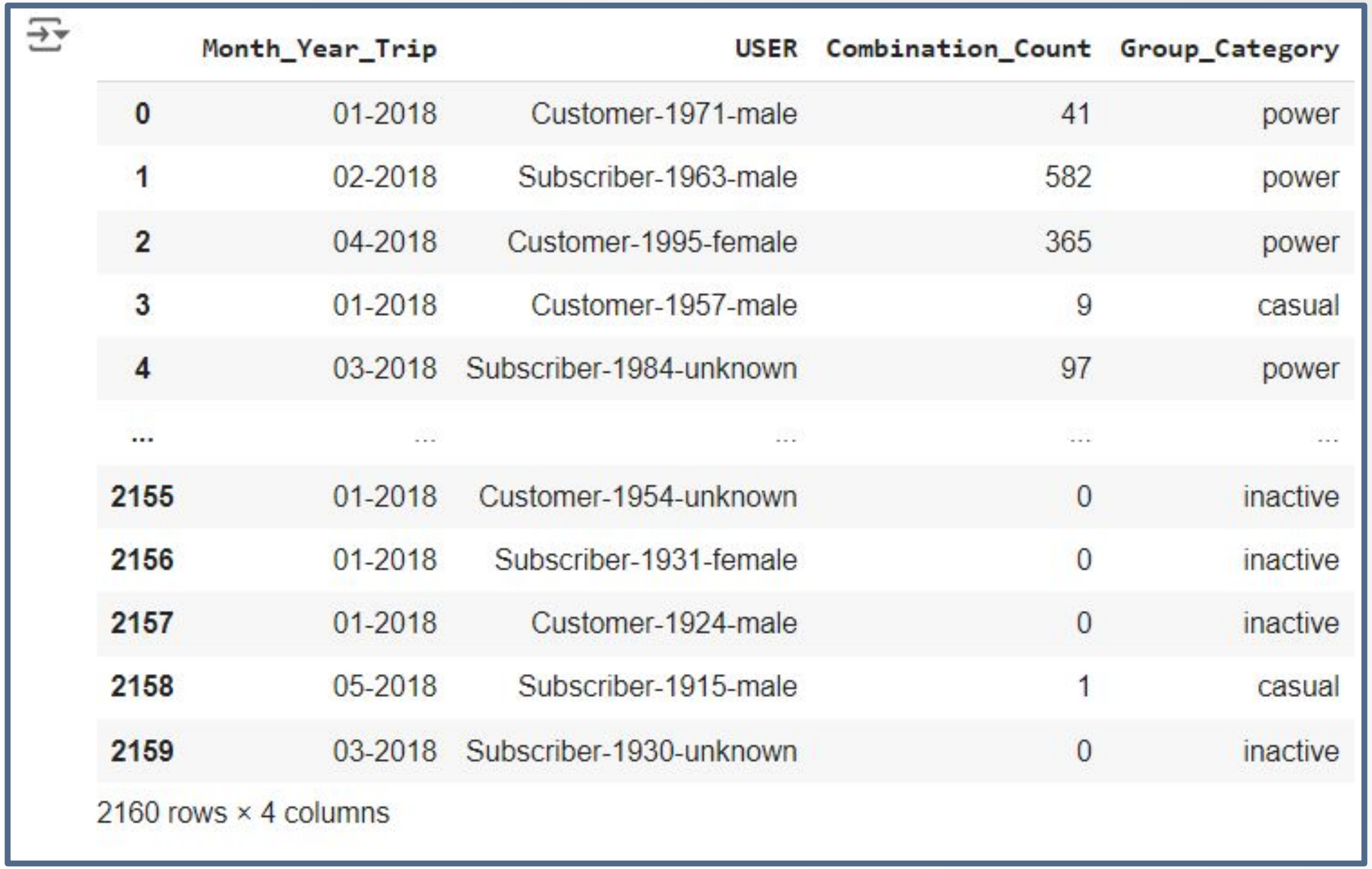
	Month_Year_Trip	User
0	01-2018	Customer-1897-unknown
1	01-2018	Customer-1972-female
2	01-2018	Customer-1949-female
3	01-2018	Customer-1941-male
4	01-2018	Subscriber-1987-female

2.1. Make combination User name and find month trip from that user



```
df_task22 = client.query('''
  WITH all_combinations AS (
    SELECT A.Month_Year_Trip, B.User
    FROM (SELECT DISTINCT FORMAT_DATE('%m-%Y', starttime) AS Month_Year_Trip
          FROM bigquery-public-data.new_york_citibike.citibike_trips
          WHERE starttime>'2018-01-01') A
    CROSS JOIN (SELECT DISTINCT CONCAT(COALESCE(CAST(usertype AS STRING),"NULL"),
                  '-', COALESCE(CAST(birth_year AS STRING),"NULL"),
                  '-', COALESCE(CAST(gender AS STRING),"NULL")) AS User
                FROM bigquery-public-data.new_york_citibike.citibike_trips
                WHERE starttime>'2018-01-01') B
  )
  SELECT AC.Month_Year_Trip, AC.USER,
         COUNT(DISTINCT start_station_name) AS Combination_Count,
         CASE
            WHEN COUNT(DISTINCT start_station_name) = 0 THEN 'inactive'
            WHEN COUNT(DISTINCT start_station_name) > 0 AND COUNT(DISTINCT start_station_name)<= 10 THEN 'casual'
            ELSE 'power'
         END AS Group_Category
  FROM all_combinations AC
  LEFT JOIN bigquery-public-data.new_york_citibike.citibike_trips CT
    ON AC.Month_Year_Trip = FORMAT_DATE('%m-%Y', CT.starttime)
    AND AC.User = CONCAT(COALESCE(CAST(CT.usertype AS STRING),"NULL"),
                      '-', COALESCE(CAST(CT.birth_year AS STRING),"NULL"),
                      '-', COALESCE(CAST(CT.gender AS STRING),"NULL"))
  GROUP BY AC.Month_Year_Trip, AC.User;
''').to_dataframe()
df_task22
```

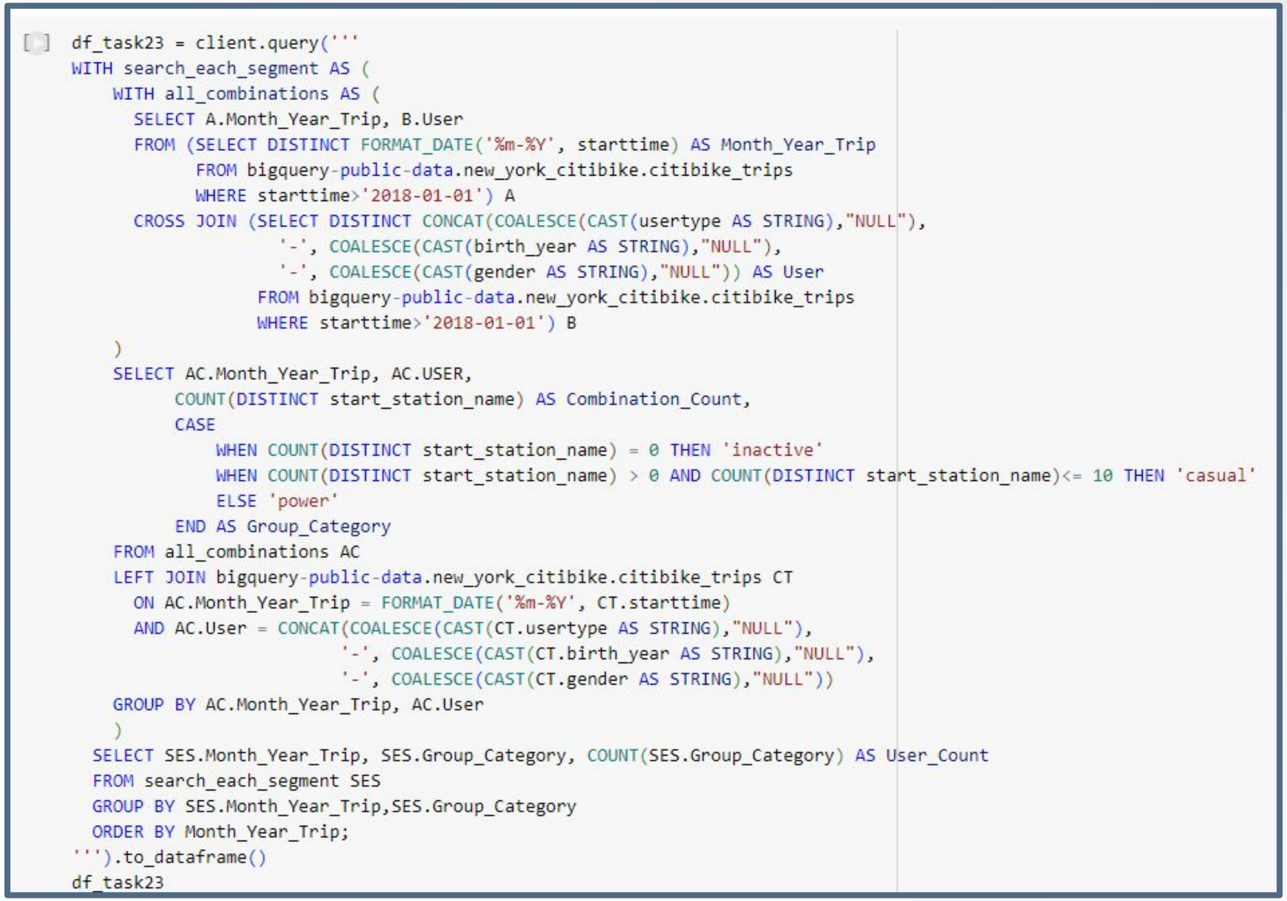
2.2 Code for generating active category per user



	Month_Year_Trip	USER	Combination_Count	Group_Category
0	01-2018	Customer-1971-male	41	power
1	02-2018	Subscriber-1963-male	582	power
2	04-2018	Customer-1995-female	365	power
3	01-2018	Customer-1957-male	9	casual
4	03-2018	Subscriber-1984-unknown	97	power
...	...	...	...	...
2155	01-2018	Customer-1954-unknown	0	inactive
2156	01-2018	Subscriber-1931-female	0	inactive
2157	01-2018	Customer-1924-male	0	inactive
2158	05-2018	Subscriber-1915-male	1	casual
2159	03-2018	Subscriber-1930-unknown	0	inactive

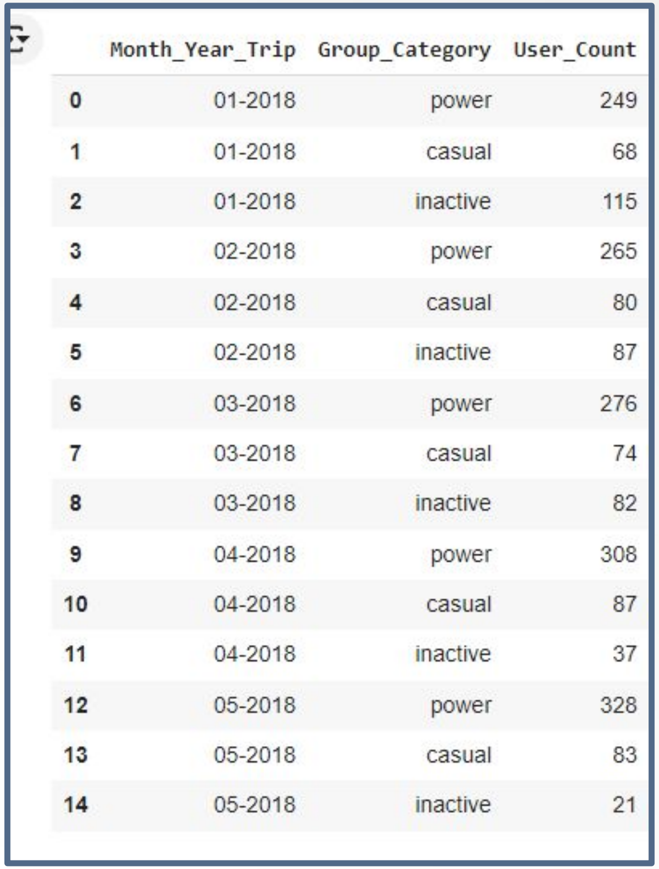
2160 rows × 4 columns

2.2 Result for generating active category per user



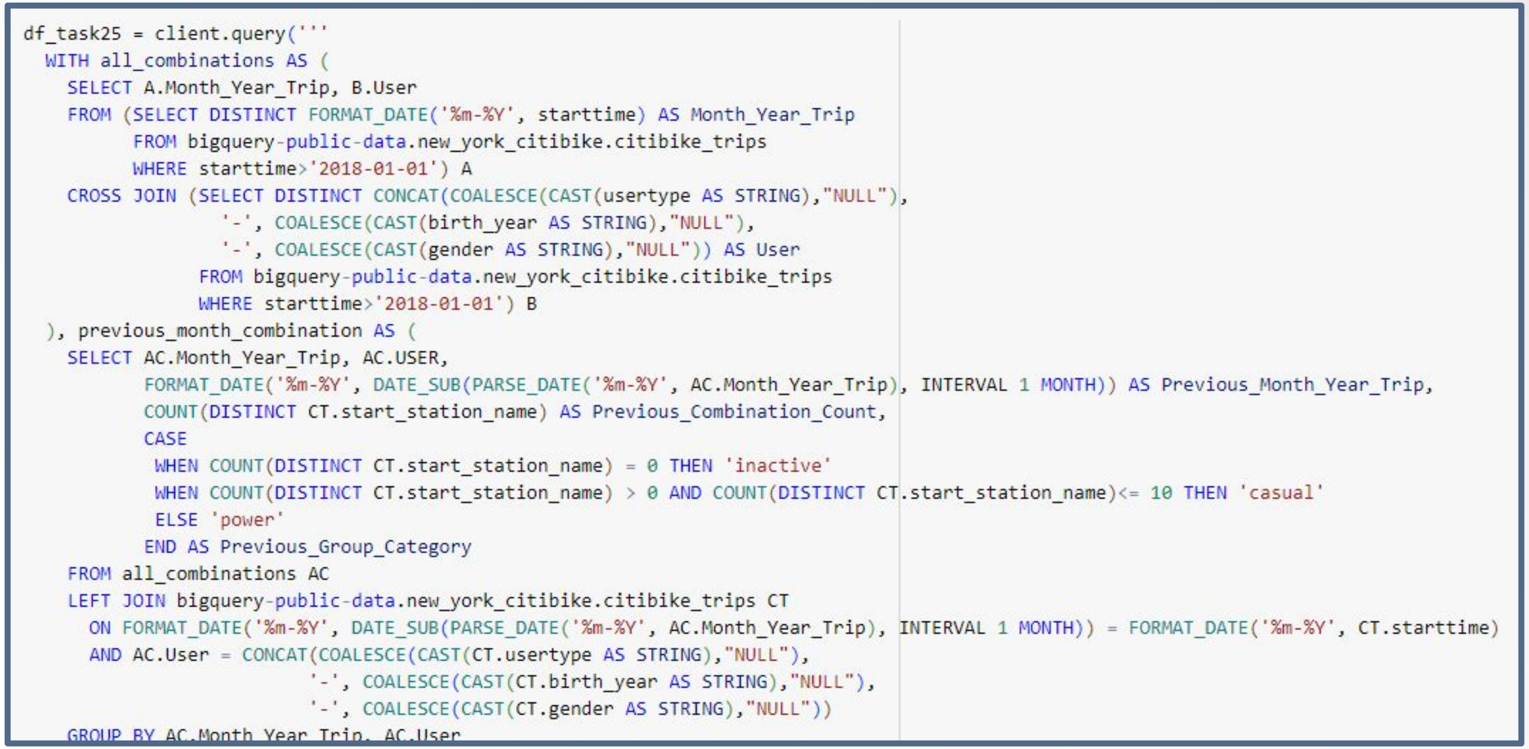
```
df_task23 = client.query('''
WITH search_each_segment AS (
    WITH all_combinations AS (
      SELECT A.Month_Year_Trip, B.User
      FROM (SELECT DISTINCT FORMAT_DATE('%m-%Y', starttime) AS Month_Year_Trip
            FROM bigquery-public-data.new_york_citibike.citibike_trips
            WHERE starttime>'2018-01-01') A
      CROSS JOIN (SELECT DISTINCT CONCAT(COALESCE(CAST(usertype AS STRING),"NULL"),
                   '-', COALESCE(CAST(birth_year AS STRING),"NULL"),
                   '-', COALESCE(CAST(gender AS STRING),"NULL")) AS User
                 FROM bigquery-public-data.new_york_citibike.citibike_trips
                 WHERE starttime>'2018-01-01') B
    )
    SELECT AC.Month_Year_Trip, AC.USER,
          COUNT(DISTINCT start_station_name) AS Combination_Count,
          CASE
              WHEN COUNT(DISTINCT start_station_name) = 0 THEN 'inactive'
              WHEN COUNT(DISTINCT start_station_name) > 0 AND COUNT(DISTINCT start_station_name)<= 10 THEN 'casual'
              ELSE 'power'
          END AS Group_Category
    FROM all_combinations AC
    LEFT JOIN bigquery-public-data.new_york_citibike.citibike_trips CT
      ON AC.Month_Year_Trip = FORMAT_DATE('%m-%Y', CT.starttime)
      AND AC.User = CONCAT(COALESCE(CAST(CT.usertype AS STRING),"NULL"),
                    '-', COALESCE(CAST(CT.birth_year AS STRING),"NULL"),
                    '-', COALESCE(CAST(CT.gender AS STRING),"NULL"))
    GROUP BY AC.Month_Year_Trip, AC.User
    )
  SELECT SES.Month_Year_Trip, SES.Group_Category, COUNT(SES.Group_Category) AS User_Count
  FROM search_each_segment SES
  GROUP BY SES.Month_Year_Trip,SES.Group_Category
  ORDER BY Month_Year_Trip;
''').to_dataframe()
df_task23
```

2.3 Code for find user count per category per month in 2018



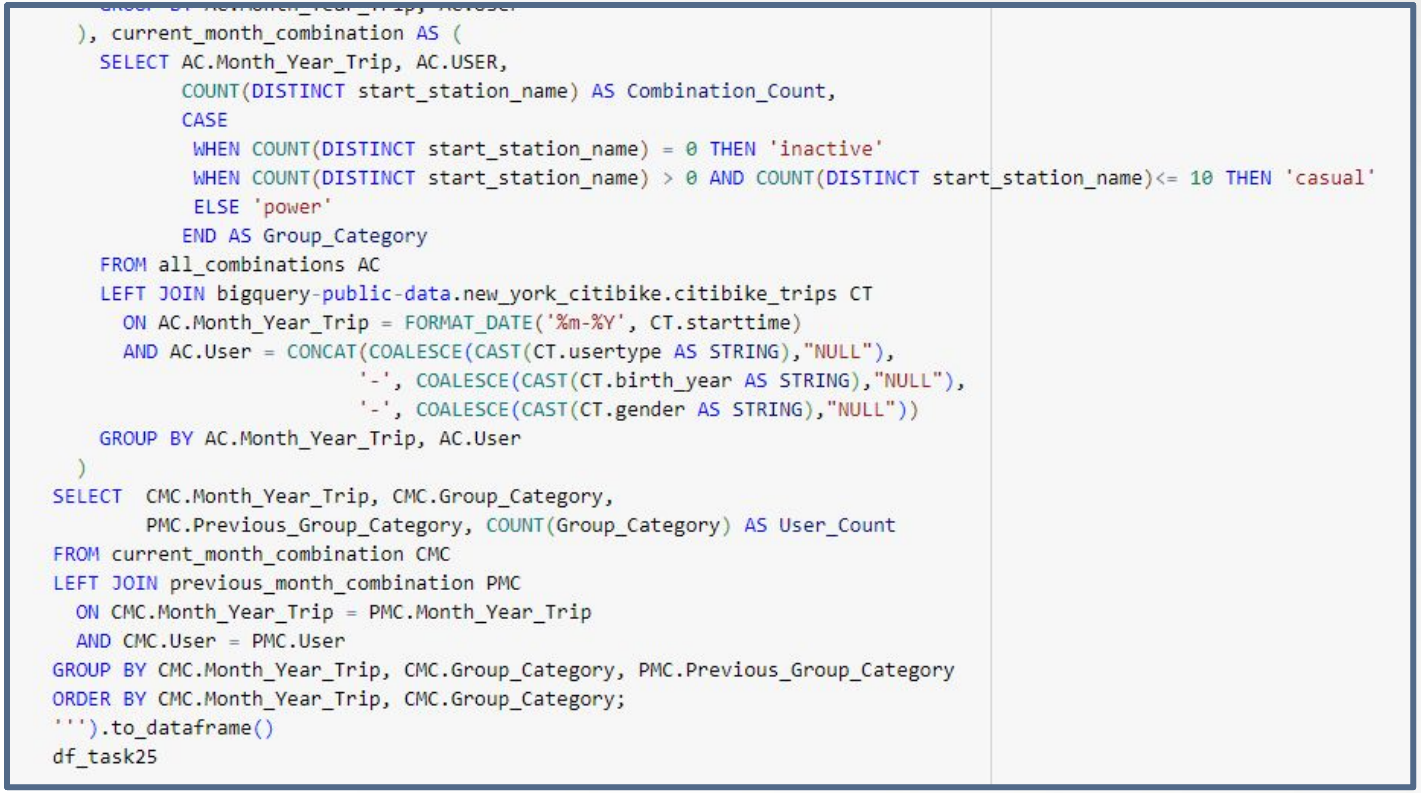
	Month_Year_Trip	Group_Category	User_Count
0	01-2018	power	249
1	01-2018	casual	68
2	01-2018	inactive	115
3	02-2018	power	265
4	02-2018	casual	80
5	02-2018	inactive	87
6	03-2018	power	276
7	03-2018	casual	74
8	03-2018	inactive	82
9	04-2018	power	308
10	04-2018	casual	87
11	04-2018	inactive	37
12	05-2018	power	328
13	05-2018	casual	83
14	05-2018	inactive	21

2.3 Result for find user count per category per month in 2018



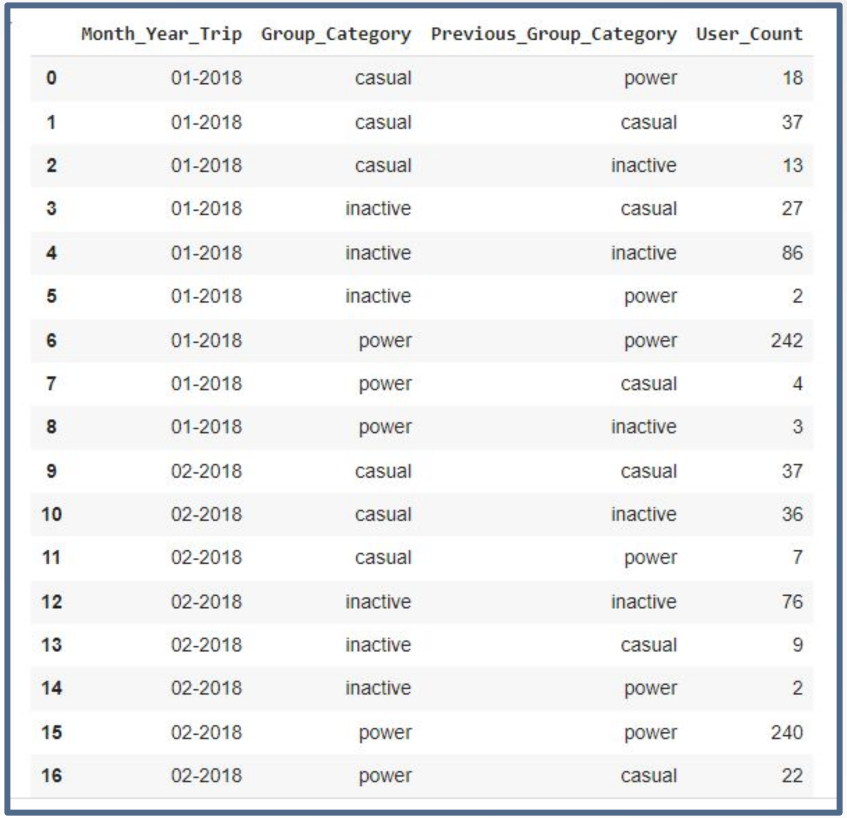
```
df_task25 = client.query('''
  WITH all_combinations AS (
    SELECT A.Month_Year_Trip, B.User
    FROM (SELECT DISTINCT FORMAT_DATE('%m-%Y', starttime) AS Month_Year_Trip
          FROM bigquery-public-data.new_york_citibike.citibike_trips
          WHERE starttime>'2018-01-01') A
    CROSS JOIN (SELECT DISTINCT CONCAT(COALESCE(CAST(usertype AS STRING),"NULL"),
                 '-', COALESCE(CAST(birth_year AS STRING),"NULL"),
                 '-', COALESCE(CAST(gender AS STRING),"NULL")) AS User
                FROM bigquery-public-data.new_york_citibike.citibike_trips
                WHERE starttime>'2018-01-01') B
  ), previous_month_combination AS (
    SELECT AC.Month_Year_Trip, AC.USER,
           FORMAT_DATE('%m-%Y', DATE_SUB(PARSE_DATE('%m-%Y', AC.Month_Year_Trip), INTERVAL 1 MONTH)) AS Previous_Month_Year_Trip,
           COUNT(DISTINCT CT.start_station_name) AS Previous_Combination_Count,
           CASE
            WHEN COUNT(DISTINCT CT.start_station_name) = 0 THEN 'inactive'
            WHEN COUNT(DISTINCT CT.start_station_name) > 0 AND COUNT(DISTINCT CT.start_station_name)<= 10 THEN 'casual'
            ELSE 'power'
           END AS Previous_Group_Category
    FROM all_combinations AC
    LEFT JOIN bigquery-public-data.new_york_citibike.citibike_trips CT
      ON FORMAT_DATE('%m-%Y', DATE_SUB(PARSE_DATE('%m-%Y', AC.Month_Year_Trip), INTERVAL 1 MONTH)) = FORMAT_DATE('%m-%Y', CT.starttime)
      AND AC.User = CONCAT(COALESCE(CAST(CT.usertype AS STRING),"NULL"),
                    '-', COALESCE(CAST(CT.birth_year AS STRING),"NULL"),
                    '-', COALESCE(CAST(CT.gender AS STRING),"NULL"))
    GROUP BY AC.Month_Year_Trip, AC.User
```

2.4 Code for find user count per category that shifting from previous month in 2018 - Part 1



```
  ), current_month_combination AS (
    SELECT AC.Month_Year_Trip, AC.USER,
           COUNT(DISTINCT start_station_name) AS Combination_Count,
           CASE
            WHEN COUNT(DISTINCT start_station_name) = 0 THEN 'inactive'
            WHEN COUNT(DISTINCT start_station_name) > 0 AND COUNT(DISTINCT start_station_name)<= 10 THEN 'casual'
            ELSE 'power'
           END AS Group_Category
    FROM all_combinations AC
    LEFT JOIN bigquery-public-data.new_york_citibike.citibike_trips CT
      ON AC.Month_Year_Trip = FORMAT_DATE('%m-%Y', CT.starttime)
      AND AC.User = CONCAT(COALESCE(CAST(CT.usertype AS STRING),"NULL"),
                          '-', COALESCE(CAST(CT.birth_year AS STRING),"NULL"),
                          '-', COALESCE(CAST(CT.gender AS STRING),"NULL"))
    GROUP BY AC.Month_Year_Trip, AC.User
  )
SELECT  CMC.Month_Year_Trip, CMC.Group_Category,
        PMC.Previous_Group_Category, COUNT(Group_Category) AS User_Count
FROM current_month_combination CMC
LEFT JOIN previous_month_combination PMC
  ON CMC.Month_Year_Trip = PMC.Month_Year_Trip
  AND CMC.User = PMC.User
GROUP BY CMC.Month_Year_Trip, CMC.Group_Category, PMC.Previous_Group_Category
ORDER BY CMC.Month_Year_Trip, CMC.Group_Category;
''').to_dataframe()
df_task25
```

2.4 Code for find user count per category that shifting from previous month in 2018 - Part 2



	Month_Year_Trip	Group_Category	Previous_Group_Category	User_Count
0	01-2018	casual	power	18
1	01-2018	casual	casual	37
2	01-2018	casual	inactive	13
3	01-2018	inactive	casual	27
4	01-2018	inactive	inactive	86
5	01-2018	inactive	power	2
6	01-2018	power	power	242
7	01-2018	power	casual	4
8	01-2018	power	inactive	3
9	02-2018	casual	casual	37
10	02-2018	casual	inactive	36
11	02-2018	casual	power	7
12	02-2018	inactive	inactive	76
13	02-2018	inactive	casual	9
14	02-2018	inactive	power	2
15	02-2018	power	power	240
16	02-2018	power	casual	22

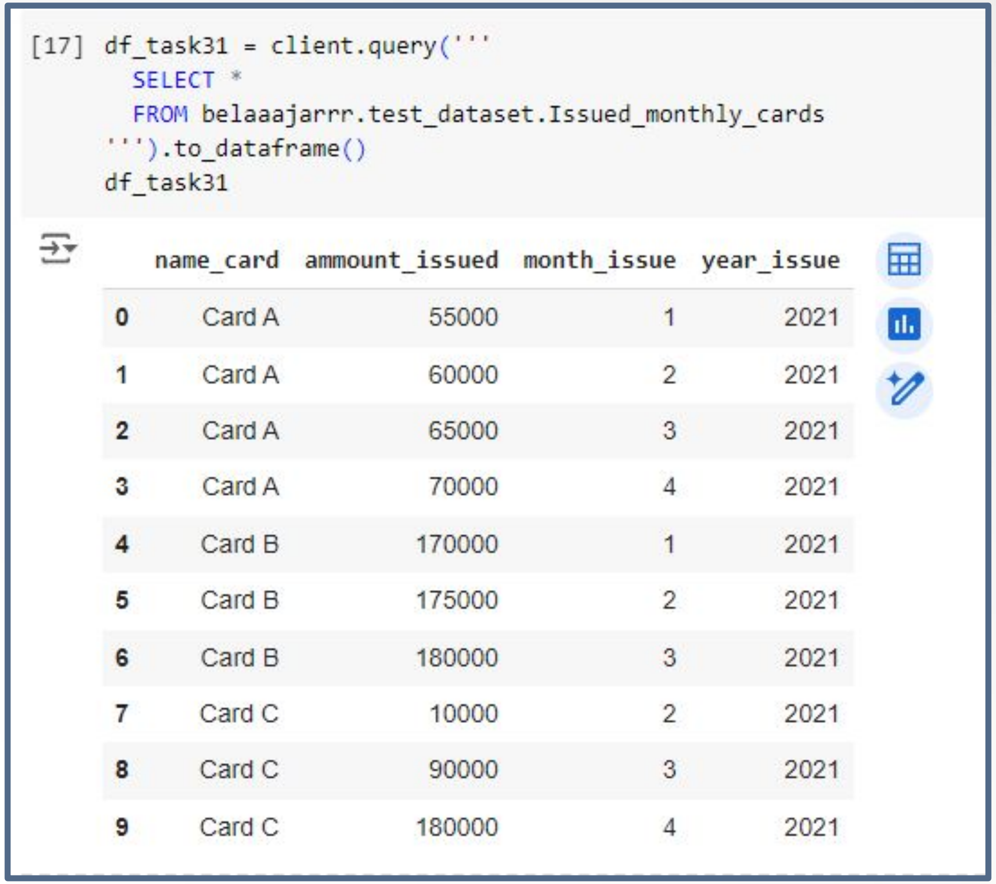
2.4 Result for find user count per category that shifting from previous month in 2018

03

Task 3

- Please write a query that generates the name of each credit card along with the discrepancy in the number of cards issued between the month with the maximum issuance and the month with minimum issuance

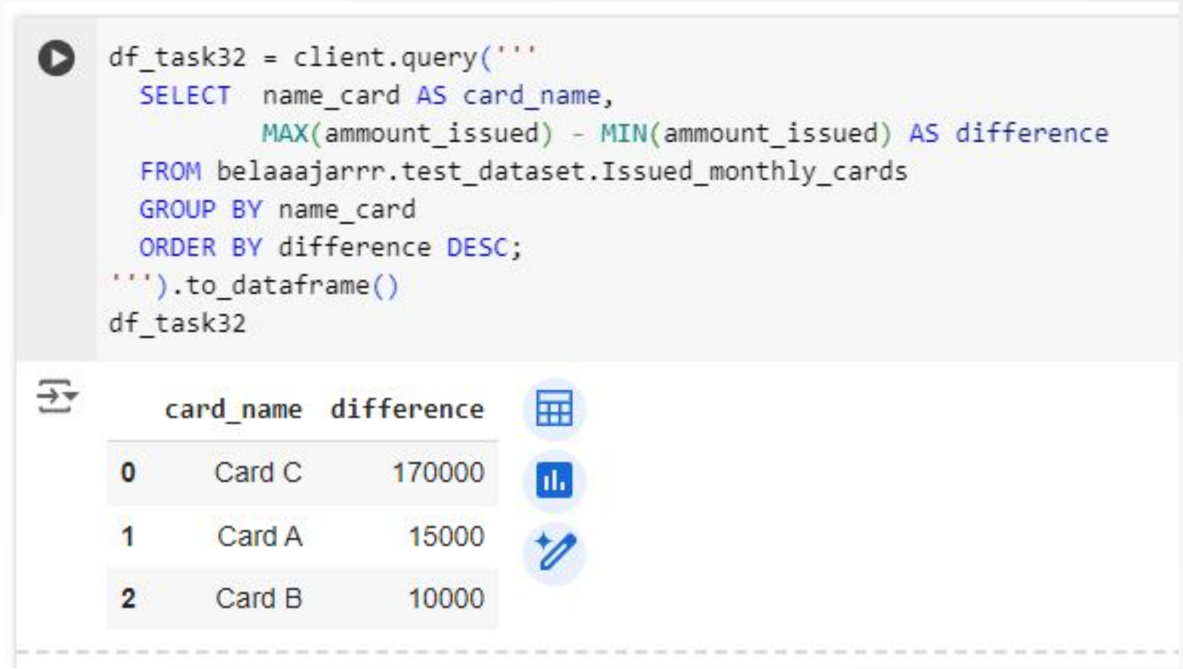
- Arrange the outcome in descending order



```
[17] df_task31 = client.query('''
       SELECT *
       FROM belaaajarrr.test_dataset.Issued_monthly_cards
     ''').to_dataframe()
     df_task31
```

	name_card	ammount_issued	month_issue	year_issue
0	Card A	55000	1	2021
1	Card A	60000	2	2021
2	Card A	65000	3	2021
3	Card A	70000	4	2021
4	Card B	170000	1	2021
5	Card B	175000	2	2021
6	Card B	180000	3	2021
7	Card C	10000	2	2021
8	Card C	90000	3	2021
9	Card C	180000	4	2021

3.1 Issued Monthly Cards Table with Some Addition Data



```
df_task32 = client.query('''
  SELECT  name_card AS card_name,
          MAX(ammount_issued) - MIN(ammount_issued) AS difference
  FROM belaaajarrr.test_dataset.Issued_monthly_cards
  GROUP BY name_card
  ORDER BY difference DESC;
''').to_dataframe()
df_task32
```

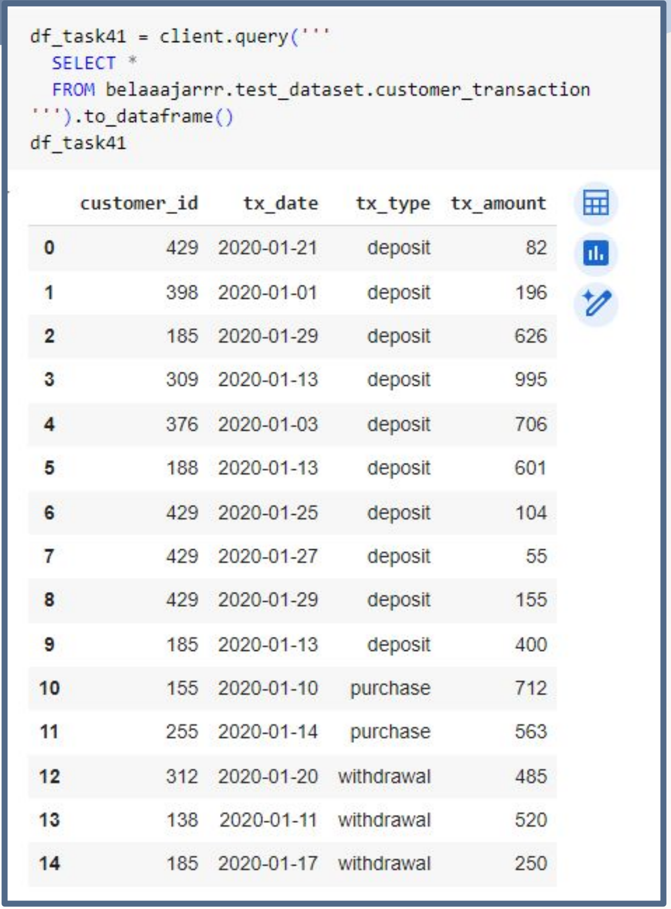
	card_name	difference
0	Card C	170000
1	Card A	15000
2	Card B	10000

3.2 Max different of amount issued by card c with 170000 difference

04

Task 4

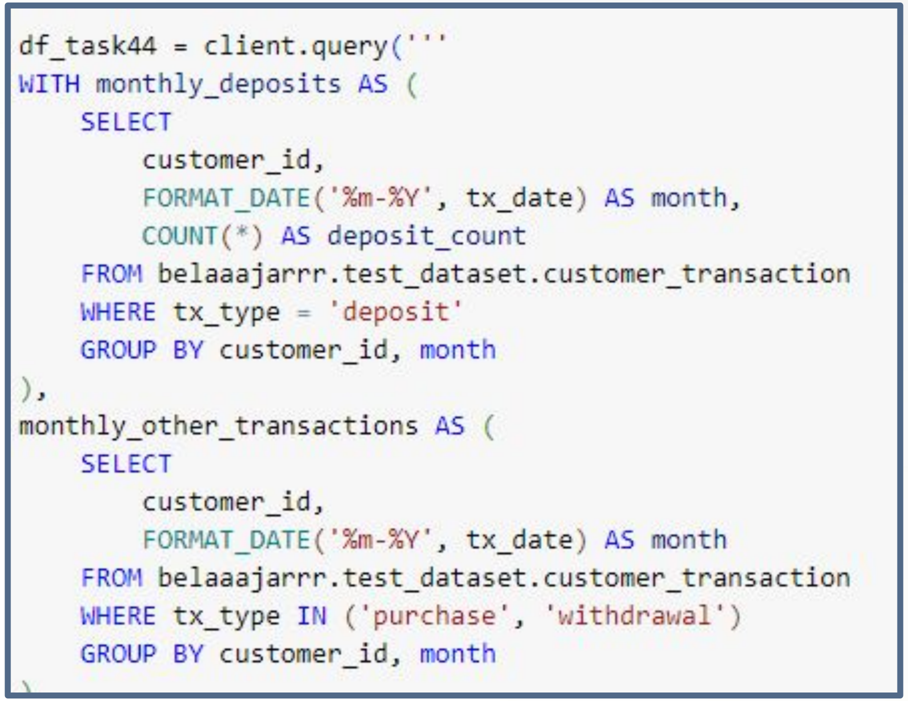
- How many customers, each month, make more than one deposit and at least one transaction (either a purchase or a withdrawal) within the same month?



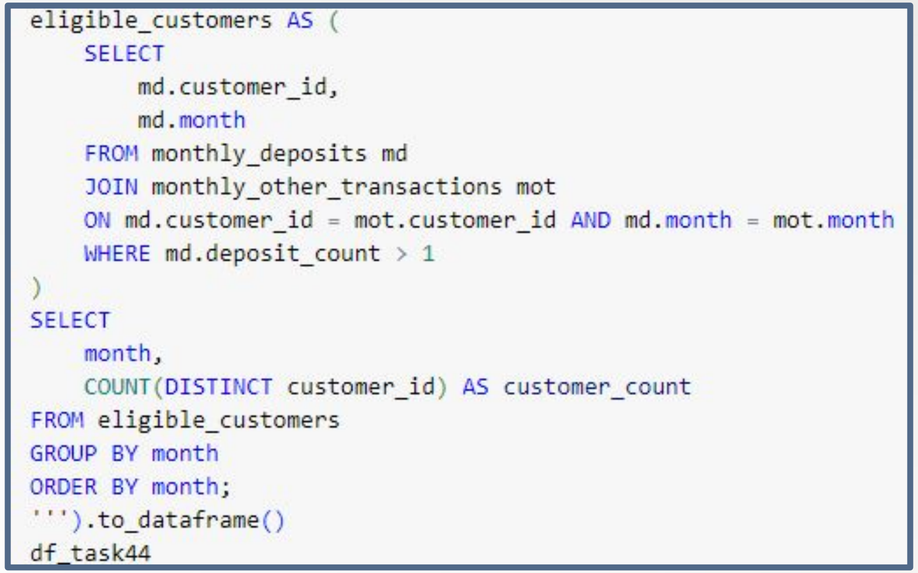
```
df_task41 = client.query('''
  SELECT *
  FROM belaaajarrr.test_dataset.customer_transaction
''').to_dataframe()
df_task41
```

```
    customer_id    tx_date     tx_type  tx_amount
0           429 2020-01-21     deposit         82
1           398 2020-01-01     deposit        196
2           185 2020-01-29     deposit        626
3           309 2020-01-13     deposit        995
4           376 2020-01-03     deposit        706
5           188 2020-01-13     deposit        601
6           429 2020-01-25     deposit        104
7           429 2020-01-27     deposit         55
8           429 2020-01-29     deposit        155
9           185 2020-01-13     deposit        400
10          155 2020-01-10    purchase        712
11          255 2020-01-14    purchase        563
12          312 2020-01-20  withdrawal        485
13          138 2020-01-11  withdrawal        520
14          185 2020-01-17  withdrawal        250
```

4.1 Customer Transaction Table with Some Addition

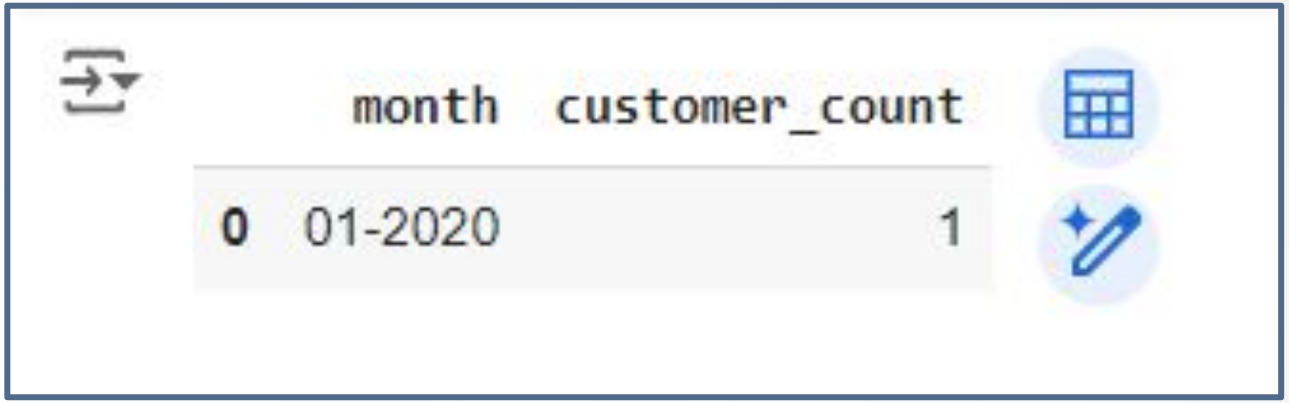


```
df_task44 = client.query('''
WITH monthly_deposits AS (
    SELECT
        customer_id,
        FORMAT_DATE('%m-%Y', tx_date) AS month,
        COUNT(*) AS deposit_count
    FROM belaaajarrr.test_dataset.customer_transaction
    WHERE tx_type = 'deposit'
    GROUP BY customer_id, month
),
monthly_other_transactions AS (
    SELECT
        customer_id,
        FORMAT_DATE('%m-%Y', tx_date) AS month
    FROM belaaajarrr.test_dataset.customer_transaction
    WHERE tx_type IN ('purchase', 'withdrawal')
    GROUP BY customer_id, month
)
```



```
eligible_customers AS (
    SELECT
        md.customer_id,
        md.month
    FROM monthly_deposits md
    JOIN monthly_other_transactions mot
    ON md.customer_id = mot.customer_id AND md.month = mot.month
    WHERE md.deposit_count > 1
)
SELECT
    month,
    COUNT(DISTINCT customer_id) AS customer_count
FROM eligible_customers
GROUP BY month
ORDER BY month;
''').to_dataframe()
df_task44
```

4.2 Preparation Code For checking Customer Tables



```
   month    customer_count
0  01-2020               1
```

4.2 Result For checking Customer Tables That have Multiple Criteria

Thank You