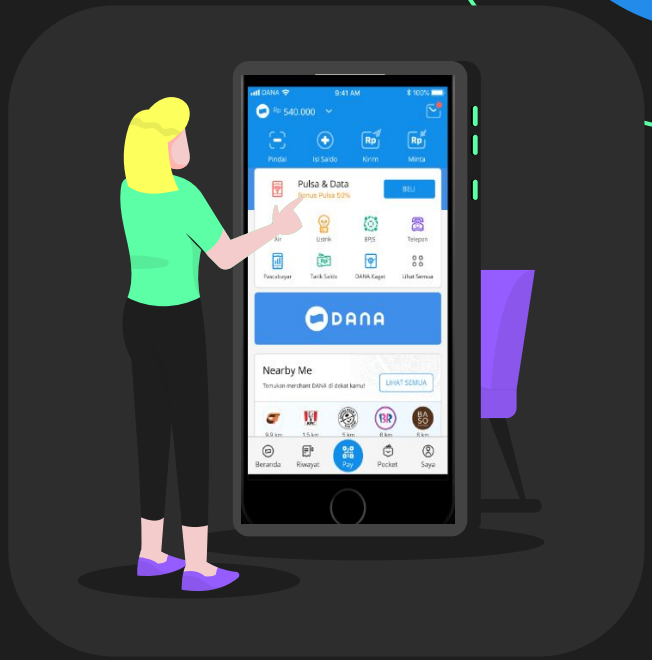
Challenge from DANA!

Business Intelligence Intern

By Khalisul Akbar



Business Intelligence Technical Test

Name : Khalisul Akbar
Date of Test : 1 - 4th September 2022

1. DANA Growth Team would like to request data for its weekly business strategy meeting. Business Intelligence team is tasked to provide data summary of DANA Users' performances in Q3 2019 (July, August, September) which include:
    a. Number of Daily Unique DANA users who transacted on Q3 2019
    b. How many users transact in a certain frequency each week? (Transact 1 Time, Transact 2 Times, Transact 3 Times, Transact more than 3 Times)
    c. What merchant is the most used as first transaction in each month during Q3 2019 (provide merchant_id).
    d. Monthly transaction percentages between current Premium and Non Premium Users.
    e. Monthly Retention Rate (how many users back transacting again next month after transacting this month).

    Please provide the SQL query. Data sample is provided as a reference to give an example of what the source table would look like (Assume data is available from January 2019 until Today).

2. As Business Intelligence, you're responsible for exploring, determining, and measuring business metrics to help improve the business. Give us at least 2 metrics that you feel would give the best possible impact to DANA as a rapidly growing financial technology company. Give your reasoning and how you would measure those metrics.

SQL Query Test

NOTES :

- In writing the SQL QUERY for this challenge, I am using PostreSQL Query language and PGAdmin 4 ver 6.10 GUI to interact with the Database.
- There are two tables (as provided) I am using in this challenge, which are :
    1. Order_table
    2. Users_table

a. Number of Daily Unique DANA users who transacted on Q3 2019 ?

```sql
-- Daily Unique DANA users who transacted on Q3 2019 (July, August, September)
SELECT COUNT(DISTINCT(user_id)) AS Daily_Unique_DANA_users_on_Q3
FROM order_table
WHERE created_date >= '2019-07-01' AND created_date < '2019-10-1' ;
```

OR

```sql
-- Daily Unique DANA users who transacted on Q3 2019 (July, August, September)
SELECT DISTINCT(user_id), COUNT(user_id) AS frequency
FROM order_table
WHERE created_date >= '2019-07-01' AND created_date < '2019-10-1'
GROUP BY user_id
ORDER BY frequency DESC ;
```

The result example will be :

	daily_unique_dana_users_on_q3 bigint
1	22

OR

	user_id character varying	frequency bigint
1	USR001	17
2	USR004	11
3	USR005	9
4	USR020	8
5	USR002	7
6	USR017	5
7	USR030	5
8	USR013	4
9	USR025	4
10	USR007	3
11	USR027	3
12	USR003	2
13	USR029	2
14	USR006	1
15	USR008	1
16	USR009	1
17	USR010	1
18	USR011	1
19	USR015	1
20	USR021	1
21	USR022	1
22	USR024	1

b. How many users transact in a certain frequency each week?

```sql
-- How many users transact in a certain frequency each week? (Transact 1 Time, Transact 2 Times, Transact 3 Times, Transact more than 3 Times)
WITH week_r AS (SELECT
    DISTINCT to_char(created_date, '"WEEK NO :"IW (Month)') AS month_week,
    user_id,
    COUNT(user_id) AS frequency,
    CASE
        WHEN COUNT(user_id) = 1 THEN 'Transact 1 time'
        WHEN COUNT(user_id) = 2 THEN 'Transact 2 times'
        WHEN COUNT(user_id) = 3 THEN 'Transact 3 times'
        ELSE 'Transact more than 3 times'
    END AS cs
    FROM order_table
    WHERE created_date >= '2019-07-01' AND created_date < '2019-10-1'
    GROUP BY 1,2
    ORDER BY 1)

SELECT month_week,  COUNT(frequency) AS total_users, SUM(frequency) AS total_transaction, cs AS transaction_, ROW_NUMBER ( ) OVER (PARTITION BY month_week ORDER BY cs)
FROM week_r
GROUP BY 1,4 ;
```

The result example will be :

	month_week text	total_users bigint	total_transaction numeric	transaction_ text	row_number bigint
1	WEEK NO :27 (July)	3	3	Transact 1 time	1
2	WEEK NO :27 (July)	1	3	Transact 3 times	2
3	WEEK NO :30 (July)	6	6	Transact 1 time	1
4	WEEK NO :30 (July)	1	2	Transact 2 times	2
5	WEEK NO :31 (August)	5	5	Transact 1 time	1
6	WEEK NO :33 (August)	2	2	Transact 1 time	1
7	WEEK NO :34 (August)	5	5	Transact 1 time	1
8	WEEK NO :34 (August)	1	2	Transact 2 times	2
9	WEEK NO :35 (August)	4	4	Transact 1 time	1
10	WEEK NO :35 (August)	1	2	Transact 2 times	2
11	WEEK NO :35 (September)	2	2	Transact 1 time	1
12	WEEK NO :35 (September)	1	2	Transact 2 times	2
13	WEEK NO :36 (September)	4	4	Transact 1 time	1
14	WEEK NO :36 (September)	4	8	Transact 2 times	2
15	WEEK NO :36 (September)	2	6	Transact 3 times	3
16	WEEK NO :36 (September)	3	18	Transact more than 3 times	4
17	WEEK NO :37 (September)	4	4	Transact 1 time	1
18	WEEK NO :37 (September)	1	3	Transact 3 times	2
19	WEEK NO :38 (September)	2	2	Transact 1 time	1
20	WEEK NO :39 (September)	5	5	Transact 1 time	1
21	WEEK NO :40 (September)	1	1	Transact 1 time	1

c. What merchant is the most used as first transaction in each month during Q3 2019 ?

```sql
-- What merchant is the most used as first transaction in each month during Q3 2019 (provide merchant_id).
SELECT month_, merchant_id, total_first_transaction
FROM (
    SELECT DISTINCT (merchant_id),
           SUM(cnt) AS total_first_transaction,
           MAX(EXTRACT(month from first_trans)) AS Month_,
           ROW_NUMBER () OVER(PARTITION BY MAX(EXTRACT(month from first_trans)) ORDER BY SUM(cnt) DESC) AS row_num
    FROM (
          SELECT DISTINCT ON (user_id) created_date AS "first_trans", user_id, merchant_id, COUNT(merchant_id) AS cnt
          FROM order_table
          WHERE created_date >= '2019-07-01' AND created_date < '2019-10-1'
          GROUP BY created_date, user_id, merchant_id
          ORDER BY user_id, created_date
          ) AS q3 -- create table for q3 data only
    GROUP BY merchant_id
    ORDER BY month_, row_num ASC
    ) AS nested_table
WHERE row_num = 1 ;
```

The result example will be :

	month_ numeric	merchant_id character varying	total_first_transaction numeric
1	7	MCT001	5
2	8	MCT886	2
3	9	MCT192	6

d. Monthly transaction percentages between current Premium and Non Premium Users on Q3 2019 ?

```sql
-- Monthly transaction percentages between current Premium and Non Premium Users.
WITH joined_table AS (
    SELECT u.is_premium, CASE WHEN is_premium = 'true' THEN 1 ELSE 0 END AS bool_to_int, o.*
    FROM users_table AS u
    LEFT JOIN order_table AS o
    ON u.user_id = o.user_id
    WHERE o.user_id IS NOT NULL AND (o.created_date >= '2019-07-01' AND o.created_date < '2019-10-1')
    ) -- CREATE TEMPORARY TABLE

SELECT
    to_char(created_date, 'IYYY-Month') AS month_,
    count(nullif(is_premium, true)) * 100/ count(is_premium) AS not_premium_pctg,  -- count false values
    count(nullif(is_premium, false)) * 100/ count(is_premium) AS premium_pctg, -- count true values
    count(is_premium) AS total_users
from joined_table
GROUP BY month_ ;
```

The result example will be :

	month_ text	not_premium_pctg bigint	premium_pctg bigint	total_users bigint
1	2019-July	50	50	14
2	2019-August	75	25	20
3	2019-September	69	30	55

e. Monthly Retention Rate on Q3 2019 ?

```sql
WITH temp_table AS (
    SELECT user_id,
           DATE_TRUNC('Month', created_date) AS month_,
           count(*) AS item_transactions,
           LAG(DATE_TRUNC('Month', created_date)) OVER (PARTITION BY user_id ORDER BY DATE_TRUNC('Month',
created_date)) = date_trunc('month', created_date) - interval '1 month'
           OR NULL AS repeat_transaction
    FROM order_table
    WHERE created_date >= '2019-07-01' AND created_date < '2019-10-1'
    GROUP BY user_id, created_date
    ORDER BY 1,2
    )

SELECT month_,
       SUM(item_transactions) AS number_transaction,
       COUNT(DISTINCT user_id) AS number_users,
       COUNT(repeat_transaction) AS repeat_buyers,
       ROUND(CASE
                WHEN sum(item_transactions) > 0 THEN COUNT(repeat_transaction) * 100/ SUM(item_transactions)
                ELSE 0
            END, 2) AS monthly_retention_rate
FROM temp_table
GROUP BY month_
ORDER BY month_
```

The result example will be :

	month_ timestamp with time zone	number_transaction numeric	number_users bigint	repeat_buyers bigint	monthly_retention_rate numeric
1	2019-07-01 00:00:00+07	14	10	0	0.00
2	2019-08-01 00:00:00+07	20	12	6	30.00
3	2019-09-01 00:00:00+07	55	16	9	16.36

Business Intelligence Metrics

As a Business Intelligence, We are responsible for exploring, determining, and measuring business metrics to help improve the business. With Business metrics, a company can track the progress of its business goals in a planned time frame. There are hundreds of different business metrics available out there, however there is no use in measuring all of it. Business metrics should be used and measured depending on each of companies business goals. As one of the best financial technology company, I consider these two metrics to be critical for DANA to measure the company's performance, which are :

1. Sales Revenue
2. Customer Retention

1. Sales Revenue

This metric can tell a lot of things about how the company performs to generate revenue using its current product/service, marketing strategy, innovation, etc. From the sales metrics, a company can track daily sales/transactions, average revenue per user, average transaction value, and Month-over-Month sales result. As a result, company can use the metric's result as a evaluation tool to determine whether the company should stick to the planned strategy or they should try a different approach.

How to measure :

- **Sales revenue** is calculated by summing up all the income from client transactions, minus the associated cost (use daily transaction for daily sales revenue and use monthly transaction data to display month-to-month sales revenue)
- **average revenue per user**, Total Revenue/ number of user who does transaction
- **average transaction value,** Total Revenue/ number of transaction

2. Customer Retention

This metric is focused on developing a better relationship with existing customers with the goal of increasing the customers loyalty and driving them to repeat the purchases/transactions/etc. Moreover, having loyal customers also beneficial not only to grow the company sales but also to help in spreading a good word about the company product.

How to measure :

Retention Rate = (((CE-CN)/CS)) X 100

Where :

CE = number of customers at the end of a certain time period (1 year, for example)
CN = number of new customers acquired during the same time period
CS = number of clients at the start of the time period

References :

[1] https://www.scoro.com/blog/12-business-metrics/
[2] https://business-apprentice.com/finance/metrics-for-a-fintech-company/
[3] https://business-apprentice.com/business/introduction-to-metrics/

THANK YOU