SQL Project Deck

Presented by Muhammad Rafie Darmawan

W4W5
SQL ASSIGNMENT

OVERVIEW

As part of W4W5 SQL assignment from Full Stack Data Analytics Program, we have to find potential problems from fictitious dataset developed by the Looker Team.

The data is called TheLook eCommerce, which contains information such as customers, product, order, logistics, and web event.

Potential Problems and Insights

Using the dataset, imagine we want to boost the profit, as well as finding some potential problems. The answer would be figured out from the questions, such as

01. During January 2019 until August 2022, how many unique customers that did shopping activity? How many order they made? how much sales' potential we could get? Divided by shopping's status and month.

02. Using the same timeframe, how many completed orders customers made? How many unique customers that made the completed order? How is the average order value? Divided by month.

03. During August 2022, many customers are refunding their orders. How is the list of customers that made refund during the month, detail it by their first name, last name, email, and user_id.

04. Find the top 5 profitable products as well as least 5 profitable products over all the time. Check what kind of products that generate high and least revenue over time?

05. Assuming the date when the analytical request came was 15 August 2022, how is the look of the MTD (Month to Date) of total profit from each product's category? Divided by month and category.

06. How is the condition of the monthly growth from each products? Divided by month and products, order it descending.

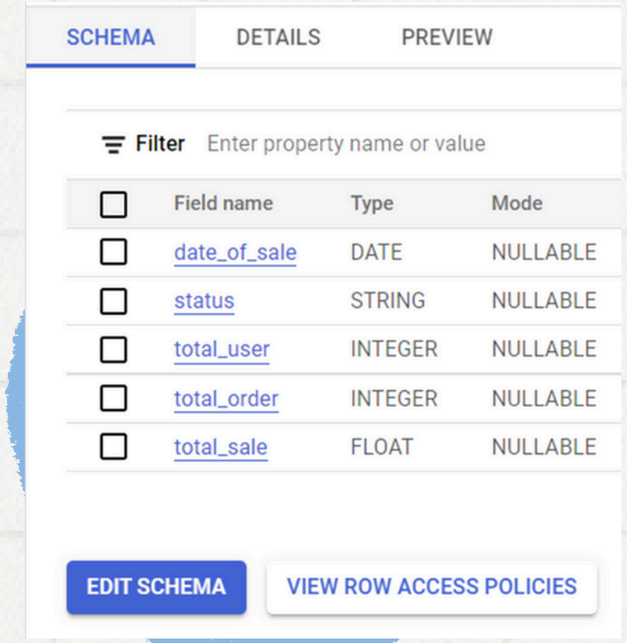
07. How does the monthly retention customers look across 2022? We need to find out using cohort analysis to see customers' retention in the following months in 2022.

Dataset

Question 1

The query here is to get the number of unique users, number of orders, and total sale price per status and month.

```
SELECT DATE_TRUNC(DATE(shipped_at), month) as
date_of_sale,
      status,
      COUNT(DISTINCT user_id) as total_user,
      COUNT(order_id) as total_order,
      ROUND(SUM(sale_price), 1) as total_sale
FROM `bigquery-public-
data.thelook_ecommerce.order_items`
WHERE shipped_at BETWEEN '2018-12-31' AND '2022-
09-01'
GROUP BY date_of_sale,
       status
ORDER BY date_of_sale;
```



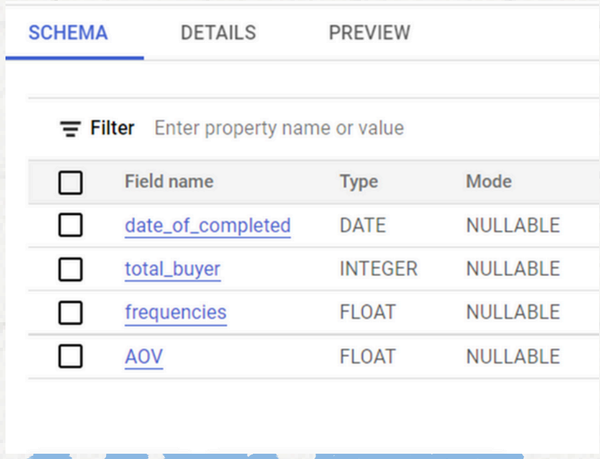
Row	date_of_sale	status	total_user	total_order	total_sale
1	2019-01-01	Complete	2	2	90.0
2	2019-01-01	Shipped	4	6	400.7
3	2019-01-01	Returned	1	2	114.9
4	2019-02-01	Complete	23	29	1544.0
5	2019-02-01	Returned	7	10	444.4
6	2019-02-01	Shipped	18	21	1598.3
7	2019-03-01	Complete	46	72	3787.8
8	2019-03-01	Shipped	51	85	5261.5
9	2019-03-01	Returned	8	10	444.8
10	2019-04-01	Shipped	69	110	6427.7
11	2019-04-01	Complete	55	82	6426.8
12	2019-04-01	Returned	21	29	1521.4
13	2019-05-01	Complete	72	112	6240.6

Dataset

Question 2

The query here is to get frequencies, average order value and total number of unique users where status is complete grouped by month.

```
SELECT DATE_TRUNC(DATE(delivered_at), month) as
date_of_completed,
      COUNT(DISTINCT user_id) as total_buyer,
      ROUND(COUNT(order_id)/COUNT(DISTINCT
user_id), 3) as frequencies,
      ROUND(AVG(sale_price),2) as AOV
FROM `bigquery-public-
data.thelook_ecommerce.order_items`
WHERE status = "Complete" AND delivered_at BETWEEN
'2019-01-01' AND '2022-09-01'
GROUP BY date_of_completed,
        status
ORDER BY date_of_completed;
```



Row	date_of_co...	total_buyer	frequencies	AOV
1	2019-01-01	2	1.0	45.0
2	2019-02-01	20	1.15	48.52
3	2019-03-01	45	1.622	55.94
4	2019-04-01	54	1.481	74.05
5	2019-05-01	70	1.543	57.45
6	2019-06-01	89	1.382	55.94
7	2019-07-01	108	1.63	59.56
8	2019-08-01	118	1.39	62.23
9	2019-09-01	161	1.509	59.25
10	2019-10-01	176	1.477	59.83
11	2019-11-01	178	1.433	56.82
12	2019-12-01	169	1.396	61.03
13	2020-01-01	213	1.469	53.24
14	2020-02-01	200	1.385	59.36
15	2020-03-01	256	1.426	56.22
16	2020-04-01	271	1.413	59.45
17	2020-05-01	301	1.585	53.05
18	2020-06-01	312	1.449	66.59
19	2020-07-01	345	1.461	58.55
20	2020-08-01	367	1.436	58.48

Dataset

Question 3

The query here is to find the user id, email, first and last name of users whose status is refunded on Aug 22

```
SELECT u.id,
      u.first_name,
      u.last_name,
      u.email,
      o.status
FROM `bigquery-public-
data.thelook_ecommerce.users` as u
INNER JOIN `bigquery-public-
data.thelook_ecommerce.orders` as o
ON u.id = o.user_id
WHERE o.returned_at BETWEEN '2022-08-01' AND
'2022-09-01'
     AND o.status = "Returned"
ORDER BY o.returned_at;
```

SCHEMA | DETAILS | PREVIEW

Filter Enter property name or value

Field name	Type	Mode
id	INTEGER	NULLABLE
first_name	STRING	NULLABLE
last_name	STRING	NULLABLE
email	STRING	NULLABLE
status	STRING	NULLABLE

Row	id	first_name	last_name	email	status
1	62548	Christine	Boyle	christineboyle@example.org	Returned
2	6326	Jonathan	Hill	jonathanhill@example.com	Returned
3	81059	Gerald	Becker	geraldbecker@example.net	Returned
4	97391	Stephanie	Roy	stephanieroy@example.org	Returned
5	90364	David	Garrison	davidgarrison@example.org	Returned
6	43214	Erin	Oconnor	erinoconnor@example.org	Returned
7	35542	Steven	Gould	stevengould@example.org	Returned
8	30671	Clayton	Winters	claytonwinters@example.org	Returned
9	70336	John	King	johnking@example.net	Returned
10	97997	John	Brown	johnbrown@example.net	Returned
11	10814	Nicholas	Woods	nicholaswoods@example.org	Returned
12	43002	Cynthia	Watson	cynthiawatson@example.org	Returned
13	38115	Diana	Harrison	dianaharrison@example.com	Returned
14	62290	Jessica	Harvey	jessicaharvey@example.com	Returned
15	96018	Cindy	Serrano	cindyserrano@example.net	Returned
16	74639	Debra	Johnson	debrajohnson@example.com	Returned
17	49852	John	Burke	johnburke@example.net	Returned
18	20292	Christopher	Griffin	christophergriffin@example.org	Returned
19	94209	Steven	Cardenas	stevencardenas@example.net	Returned
20	76909	Nancy	Shields	nancyshields@example.org	Returned

Dataset

Question 4

The query here is to get the top 5 least and most profitable product over all time

```
WITH highest as (
SELECT "highest" AS datatype,
      p.id AS product_id,
      p.name AS product_name,
      ROUND(p.retail_price,2) AS ret_price,
      ROUND(p.cost,2) AS ret_cost,
      ROUND(SUM(o.sale_price - p.cost),2) as profit
FROM `bigquery-public-data.thelook_ecommerce.products` as p
INNER JOIN `bigquery-public-data.thelook_ecommerce.order_items` o
ON p.id = o.product_id
GROUP BY datatype,
        product_id,
        product_name,
        ret_price,
        ret_cost
ORDER BY profit DESC
limit 5
),
lowest as (
SELECT "lowest" AS datatype,
      p.id AS product_id,
      p.name AS product_name,
      ROUND(p.retail_price,2) AS ret_price,
      ROUND(p.cost,2) AS ret_cost,
      ROUND(SUM(o.sale_price - p.cost),2) as profit
FROM `bigquery-public-data.thelook_ecommerce.products` as p
INNER JOIN `bigquery-public-data.thelook_ecommerce.order_items` o
ON p.id = o.product_id
GROUP BY datatype,
        product_id,
        product_name,
        ret_price,
        ret_cost
ORDER BY profit
limit 5
)
SELECT *
FROM highest
UNION ALL
SELECT *
FROM lowest;
```

SCHEMA | DETAILS | PREVIEW

Filter Enter property name or value

Field name	Type	Mode
datatype	STRING	NULLABLE
product_id	INTEGER	NULLABLE
product_name	STRING	NULLABLE
ret_price	FLOAT	NULLABLE
ret_cost	FLOAT	NULLABLE
profit	FLOAT	NULLABLE

Row	datatype	product_id	product_name	ret_price	ret_cost	profit
1	highest	24447	Darla	999.0	404.6	7727.26
2	highest	2793	adidas Women's adiFIT Slim Pa...	903.0	375.65	6855.58
3	highest	20171	Robert Graham Men's Minstrel ...	598.0	209.9	5821.53
4	highest	23654	The North Face Apex Bionic So...	903.0	363.01	5399.94
5	highest	20228	Faconnable Men's Double Face...	487.5	185.74	4828.2
6	lowest	14235	Indestructable Aluminum Alum...	0.02	0.01	0.02
7	lowest	14159	Set of 2 - Replacement Insert For Checkbook Wallets Card Or Picture Insert	0.49	0.18	0.31
8	lowest	14202	GENUINE LEATHER SNAP ON STUDDED WHITE PIANO BELT FITS ANY BUCKLE	1.5	0.62	0.88
9	lowest	28913	TopTie Mens Black & White Checkerboard Pre-Tied Satin Formal Bow Tie	2.59	1.07	1.52
10	lowest	15531	Aviator Sunglasses Mirror Lens...	2.99	1.64	2.7

Dataset

Question 5

The query here is to get Month to Date of total profit in each product categories of past 3 months (current date 15 Aug 2022), breakdown by month and categories

```
WITH base as (
SELECT DATE_TRUNC(DATE(o.delivered_at), day) as mtd_date,
      p.category as categories,
      ROUND(SUM(o.sale_price - p.cost),2) as profit
FROM `bigquery-public-data.thelook_ecommerce.products` as p
INNER JOIN `bigquery-public-data.thelook_ecommerce.order_items` o
ON p.id = o.product_id
WHERE DATE_TRUNC(DATE(o.delivered_at), day) IS NOT NULL AND
    DATE_TRUNC(DATE(o.delivered_at), day) BETWEEN '2022-05-31' AND
'2022-08-15' AND
    EXTRACT (day from DATE_TRUNC(DATE(o.delivered_at), day)) BETWEEN 1
AND 15
GROUP BY mtd_date,
       categories
ORDER BY mtd_date
)
SELECT mtd_date,
      categories,
      profit,
      ROUND(SUM(profit)OVER(PARTITION BY EXTRACT(month from mtd_date),
categories ORDER BY mtd_date),2) as cumulative_profit
FROM base;
```

SCHEMA | DETAILS | PREVIEW

Filter Enter property name or value

Field name	Type	Mode	C
mtd_date	DATE	NULLABLE	
categories	STRING	NULLABLE	
profit	FLOAT	NULLABLE	
cumulative_profit	FLOAT	NULLABLE	

Row	mtd_date	categories	profit	cumulative_...
1	2022-06-01	Accessories	185.55	185.55
2	2022-06-02	Accessories	160.07	345.62
3	2022-06-03	Accessories	41.52	387.14
4	2022-06-04	Accessories	19.32	406.46
5	2022-06-05	Accessories	78.98	485.44
6	2022-06-06	Accessories	25.35	510.79
7	2022-06-07	Accessories	83.74	594.53
8	2022-06-08	Accessories	60.27	654.8
9	2022-06-09	Accessories	123.86	778.66
10	2022-06-10	Accessories	162.42	941.08
11	2022-06-11	Accessories	89.67	1030.75
12	2022-06-12	Accessories	98.75	1129.5
13	2022-06-13	Accessories	226.89	1356.39
14	2022-06-14	Accessories	40.86	1397.25
15	2022-06-15	Accessories	339.31	1736.56
16	2022-06-01	Active	45.06	45.06
17	2022-06-02	Active	78.33	123.39
18	2022-06-03	Active	115.72	239.11
19	2022-06-04	Active	187.48	426.59
20	2022-06-05	Active	98.51	525.1

Dataset

Question 6

The query here is to get monthly growth of inventory in percentage breakdown by product categories, ordered by time descending. After analyzing the monthly growth, is there any interesting insight that we can get?

```
WITH base as (
SELECT DATE_TRUNC(date(created_at), month) as monthly_growth,
      product_category as category,
      COUNT(product_id) as inventory,
FROM `bigquery-public-data.thelook_ecommerce.inventory_items` as i
WHERE sold_at IS NOT NULL AND created_at BETWEEN '2018-12-31' AND '2022-04-30'
GROUP BY monthly_growth,
       category
ORDER BY monthly_growth DESC
),
base_2 as (
SELECT *,
      LEAD(inventory)OVER(PARTITION BY category ORDER BY monthly_growth DESC)
as previous_inventory
FROM base
)
SELECT *,
      IFNULL(ROUND((inventory - previous_inventory)/previous_inventory*100,2),0) as
percentage
FROM base_2;
```

SCHEMA | DETAILS | PREVIEW

Filter Enter property name or value

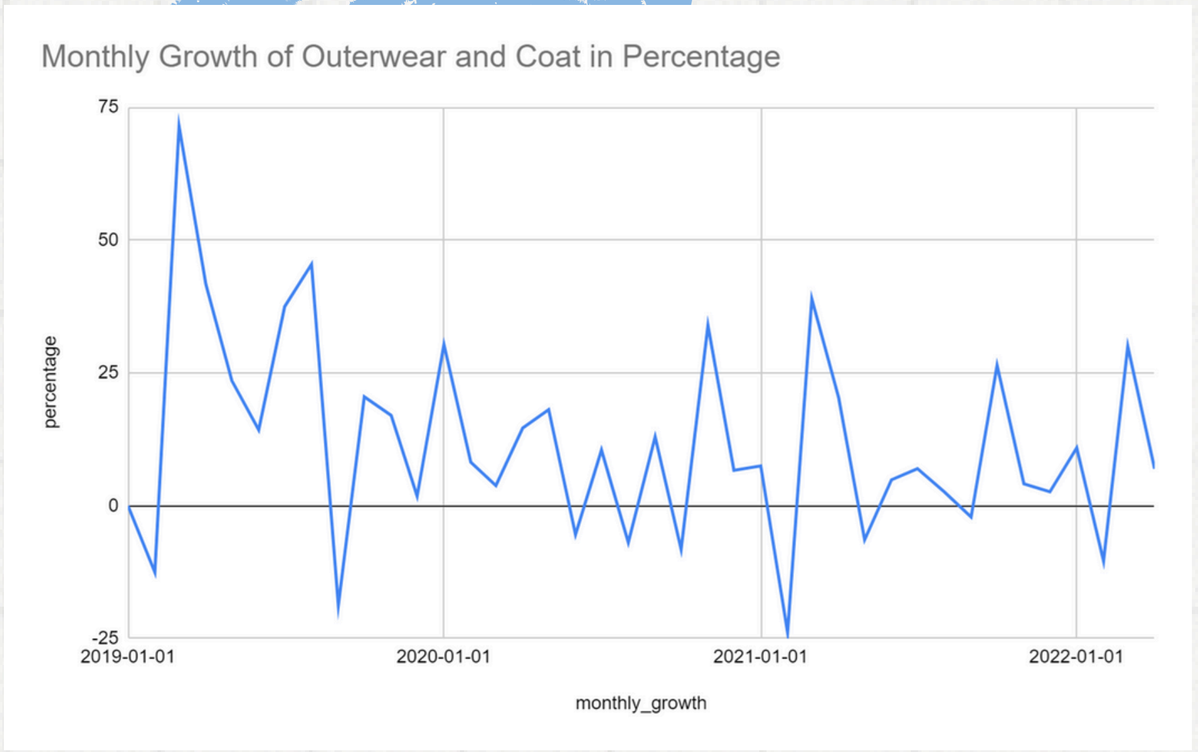
Field name	Type	Mode	Collatio
monthly_growth	DATE	NULLABLE	
category	STRING	NULLABLE	
inventory	INTEGER	NULLABLE	
previous_inventory	INTEGER	NULLABLE	
percentage	FLOAT	NULLABLE	

Row	monthly_gro...	category	inventory	previous_inv...	percentage
20	2020-09-01	Accessories	130	131	-0.76
21	2020-08-01	Accessories	131	117	11.97
22	2020-07-01	Accessories	117	88	32.95
23	2020-06-01	Accessories	88	113	-22.12
24	2020-05-01	Accessories	113	92	22.83
25	2020-04-01	Accessories	92	90	2.22
26	2020-03-01	Accessories	90	90	0.0
27	2020-02-01	Accessories	90	67	34.33
28	2020-01-01	Accessories	67	65	3.08
29	2019-12-01	Accessories	65	61	6.56
30	2019-11-01	Accessories	61	60	1.67
31	2019-10-01	Accessories	60	53	13.21
32	2019-09-01	Accessories	53	44	20.45
33	2019-08-01	Accessories	44	42	4.76
34	2019-07-01	Accessories	42	41	2.44
35	2019-06-01	Accessories	41	37	10.81
36	2019-05-01	Accessories	37	23	60.87
37	2019-04-01	Accessories	23	22	4.55
38	2019-03-01	Accessories	22	10	120.0
39	2019-02-01	Accessories	10	9	11.11
40	2019-01-01	Accessories	9	null	0.0

Dataset

Insights from #6

As we see here I attached the growth percentage of Outerwear and Coat. Now the thing that I interest to point out is how we see the graph of inventory stock had spiked up every start of the year and near end of the year. As example when the graph had spiked up almost 75% near the start of 2019, and then they decreased below zero when summer came up. Quick hypothesis from my side, if the inventory graph had grown up (in percentage as the chart shows), it means that the category was searched and bought by many people that the store came up with the conclusion to increase the inventory of said category. The reason this category increase at the start and end of the year? One of my early conclusion is because the winter season comes.



Dataset

Question 7

The query in the next slide is to get monthly retention cohorts (the groups, or cohorts, can be defined based upon the date that a user completely purchased a product) and then how many of them (%) coming back for the following months in 2022. After analyzing the retention cohort, is there any interesting insight that we can get?

SCHEMA | DETAILS | PREVIEW

Filter Enter property name or value

Field name	Type	Mode
first_purchase_month	DATE	NULLABLE
num_users	INTEGER	NULLABLE
month_number	INTEGER	NULLABLE
user_id	INTEGER	NULLABLE
pct_retention	FLOAT	NULLABLE

Row	first_purcha...	num_users	month_num...	user_id	pct_retention
1	2022-01-01	2863	0	2863	1.0
2	2022-01-01	2863	5	112	0.04
3	2022-01-01	2863	2	94	0.03
4	2022-01-01	2863	4	112	0.04
5	2022-01-01	2863	3	116	0.04
6	2022-01-01	2863	7	121	0.04
7	2022-01-01	2863	8	95	0.03
8	2022-01-01	2863	1	95	0.03
9	2022-01-01	2863	6	98	0.03
10	2022-01-01	2863	9	28	0.01
11	2022-02-01	2727	0	2727	1.0
12	2022-02-01	2727	7	106	0.04
13	2022-02-01	2727	4	95	0.03
14	2022-02-01	2727	5	110	0.04
15	2022-02-01	2727	2	98	0.04
16	2022-02-01	2727	1	104	0.04
17	2022-02-01	2727	6	105	0.04
18	2022-02-01	2727	3	99	0.04
19	2022-02-01	2727	8	20	0.01
20	2022-03-01	3068	2	119	0.04
21	2022-03-01	3068	0	3068	1.0
22	2022-03-01	3068	5	121	0.04

Dataset

Query from #7

```
WITH first_purchase as (
SELECT o.user_id as user_id,
      MIN(DATE(DATE_TRUNC(shipped_at, month))) as first_purchase_month
FROM `bigquery-public-data.thelook_ecommerce.orders` as o
WHERE shipped_at BETWEEN '2022-01-01' AND '2022-12-31' AND shipped_at IS NOT NULL
GROUP BY o.user_id
),
next_purchase as (
SELECT DISTINCT o.user_id as user_id,
      DATE_DIFF(DATE(DATE_TRUNC(shipped_at, month)),first_purchase_month, month) as
month_number,
FROM `bigquery-public-data.thelook_ecommerce.orders` as o
LEFT JOIN first_purchase as first
ON first.user_id = o.user_id
WHERE shipped_at BETWEEN '2022-01-01' AND '2022-12-31' AND shipped_at IS NOT NULL
),
cohort_size as (
SELECT first_purchase_month,
      COUNT(user_id) as num_users
FROM first_purchase
GROUP BY first_purchase_month
ORDER BY first_purchase_month
),
retention as(
SELECT first.first_purchase_month,
      month_number,
      COUNT(next.user_id)user_id
FROM next_purchase as next
LEFT JOIN first_purchase as first
ON next.user_id=first.user_id
GROUP BY first_purchase_month,
       month_number
ORDER BY first_purchase_month,
       month_number
)
SELECT cohort.first_purchase_month,
      num_users,
      month_number,
      user_id,
      ROUND(user_id/num_users,2) as pct_retention
FROM cohort_size cohort
LEFT JOIN retention
ON cohort.first_purchase_month=retention.first_purchase_month
ORDER BY cohort.first_purchase_month;
```

Dataset

Insights from #7

first_purchase_month	num_users	0	1	2	3	4	5	6	7	8	9	Average of Grouping
1/1/2022	2,863	100.00%	3.32%	3.28%	4.05%	3.91%	3.91%	3.42%	4.23%	3.32%	0.98%	3.38%
2/1/2022	2,727	100.00%	3.81%	3.59%	3.63%	3.48%	4.03%	3.85%	3.89%	0.73%		3.38%
3/1/2022	3,068	100.00%	5.22%	3.88%	4.53%	4.50%	3.94%	3.81%	1.21%			3.87%
4/1/2022	3,084	100.00%	4.51%	4.12%	4.51%	5.45%	4.80%	1.01%				4.06%
5/1/2022	3,409	100.00%	4.69%	6.10%	5.87%	5.28%	1.26%					4.64%
6/1/2022	3,543	100.00%	6.97%	5.42%	5.45%	1.27%						4.78%
7/1/2022	4,026	100.00%	7.95%	7.92%	1.86%							5.91%
8/1/2022	4,434	100.00%	10.53%	2.57%								6.55%
9/1/2022	5,048	100.00%	5.41%									5.41%
10/1/2022	3,109	100.00%										0
Average of month ongoing		100%	5.82%	4.61%	4.27%	3.98%	3.59%	3.02%	3.11%	2.03%	0.98%	4.62%

Above right here is the retention cohort that was made by the result of user purchased the products. Basically, this cohort shows us how many of the users that comeback to purchase the products from us in the span of 9 months in 2022. For example, one group of users that purchased the product for the first time in January 2022 had their biggest comeback in 7th month to buy the products again, and then we also see massive decrease of comeback users in 9th month, to the point that it reached only 0,98% from total of 2.863 users that first purchased the products in January.

After we got many group of users that purchased the products in certain number of months' periods, we get the average of the retention user from many groups. Now I can say, the store really did a great job to retain the user for average of 4,62%, but maybe they could do it better by adjusting the trend of how user likes its product, and maybe we could also provide them with such a membership, subscription, etc. that could give them some discount to product's price if they buy it for several times.