COHORT ANALYSIS SQL QUERIES REPORT

A. Exploratory Data Analysis (EDA) in SQL:

Query 1. The total number of buyers and the number of completed orders each month (from January 2019 to April 2022):

```
Select FORMAT_DATE('%Y-%m', t2.delivered_at) as month_year,

count(DISTINCT t1.user_id) as total_user,

count(t1.ORDER_id) as total_order

from bigquery-public-data.thelook_ecommerce.orders as t1

Join bigquery-public-data.thelook_ecommerce.order_items as t2

on t1.order_id=t2.order_id

Where t1.status='Complete' and

t2.delivered_at BETWEEN '2019-01-01 00:00:00' AND '2022-05-01 00:00:00'

Group by month_year

ORDER BY month_year
```

Row	month_year	total_user	total_order
1	2019-01	2	2
2	2019-02	7	10
3	2019-03	12	21
4	2019-04	21	28
5	2019-05	23	36
6	2019-06	33	53
7	2019-07	39	55
8	2019-08	50	66
9	2019-09	54	91
10	2019-10	57	85

Insights:

- Overall, the number of buyers and completed orders has gradually increased each month and year.

- Period from 2019 to January 2022: Buyers tended to shop more during the last three months of the year (October–December) and January of the following year due to increased year-end and New Year shopping demand, as well as various year-end promotions and discounts.

- First four months of 2022: A significant increase in the number of buyers was recorded compared to the last three months of 2021, possibly due to TheLook launching a new promotional campaign to stimulate shopping activity in the early months of the year.

- July 2021: An unusual surge in purchases was observed, in contrast to the decline in the same period of 2020. This may be attributed to TheLook implementing a special campaign to improve sales performance specifically for July.

***Query 2. Average Order Value (AOV) and the number of customers each month**:*

```
Select

FORMAT_DATE('%Y-%m', created_at) as month_year,

count(DISTINCT user_id) as distinct_users,

round(sum(sale_price)/count(distinct order_id),2) as average_order_value

from bigquery-public-data.thelook_ecommerce.order_items

Where created_at BETWEEN '2019-01-01 00:00:00' AND '2022-05-01 00:00:00'

Group by month_year

ORDER BY month_year
```

Row	month_year	distinct_users	average_order_value
1	2019-01	10	62.1
2	2019-02	37	80.9
3	2019-03	62	83.57
4	2019-04	85	82.26
5	2019-05	123	87.62
6	2019-06	140	80.13
7	2019-07	158	82.16
8	2019-08	200	81.71

Insight:

- In 2019, the low number of users led to high fluctuations in the average order value (AOV) across months.

- From late 2019 onwards, the number of users stabilized above 400 and generally continued to increase over the months, while the average order value remained stable at around $80–$90.

Query 3. Customer segments by age - the youngest and oldest customers by gender (From January 2019 to April 2022):

```
With female_age as

(select min(age) as min_age, max(age) as max_age

from bigquery-public-data.thelook_ecommerce.users

Where gender='F' and created_at BETWEEN '2019-01-01 00:00:00' AND '2022-05-01 00:00:00'),

male_age as

(select min(age) as min_age, max(age) as max_age

from bigquery-public-data.thelook_ecommerce.users

Where gender='M' and created_at BETWEEN '2019-01-01 00:00:00' AND '2022-05-01 00:00:00'),

young_old_group as

(select t1.first_name, t1.last_name, t1.gender, t1.age

from bigquery-public-data.thelook_ecommerce.users as t1

Join female_age as t2 on t1.age=t2.min_age or t1.age=t2.max_age

Where t1.gender='F'and created_at BETWEEN '2019-01-01 00:00:00' AND '2022-05-01 00:00:00'

UNION ALL

Select t3.first_name, t3.last_name, t3.gender, t3.age

from bigquery-public-data.thelook_ecommerce.users as t3

Join female_age as t4 on t3.age=t4.min_age or t3.age=t4.max_age

Where t3.gender='M' and created_at BETWEEN '2019-01-01 00:00:00' AND '2022-05-01 00:00:00'),
```

```
age_tag as

(Select *,

Case

When age in (select min(age) as min_age

from bigquery-public-data.thelook_ecommerce.users

Where gender='F' and created_at

BETWEEN '2019-01-01 00:00:00' AND '2022- 05-01 00:00:00') then 'Youngest'

When age in (select min(age) as min_age

from bigquery-public-data.thelook_ecommerce.users

Where gender='M'and created_at BETWEEN '2019-01-01 00:00:00' AND '2022-05-01
00:00:00') then 'Youngest'

Else 'Oldest'

END as tag from young_old_group )

Select gender, tag, count(*) as user_count from age_tag

group by gender, tag
```

Row	gender	tag	user_count
1	F	Youngest	471
2	F	Oldest	489
3	M	Oldest	469
4	M	Youngest	458

Insight:

- In the period from January 2019 to April 2022

+ Gender Female: the oldest is 70 years old (489 users); the youngest is 12 years old (471 users)

+ Gender Male: the oldest is 70 years old (469 users); the youngest is 12 years old (458 users)

***Query 4. Top 5 products with the highest profit each month (ranking for each product)*:**

```
  WITH product_profit AS ( SELECT
CAST(FORMAT_DATE('%Y-%m', t1.delivered_at) AS STRING) AS month_year,
 t1.product_id AS product_id, t2.name AS product_name,
 ROUND(SUM(t1.sale_price), 2) AS sales, ROUND(SUM(t2.cost), 2) AS cost,
 ROUND(SUM(t1.sale_price) - SUM(t2.cost), 2) AS profit
   FROM bigquery-public-data.thelook_ecommerce.order_items AS t1
  JOIN bigquery-public-data.thelook_ecommerce.products AS t2
    ON t1.product_id = t2.id
    WHERE t1.status = 'Complete'
 GROUP BY month_year, t1.product_id, t2.name)
 SELECT * FROM ( SELECT *,
    DENSE_RANK() OVER (PARTITION BY month_year ORDER BY profit DESC) AS  rank
    FROM product_profit) AS ranked_table
  WHERE ranked_table.rank <= 5
  ORDER BY ranked_table.month_year, ranked_table.rank;
```

Row	month_year	product_id	product_name	sales	cost	profit	rank
1	2019-01	26257	ck one Men's Ck One Slim Fit B...	26.0	12.3	13.7	1
2	2019-01	19368	Allegra K Mens Stylish Deep V Neck Button Down Pure Color Stretchy Fall Cardigan Blue S	13.2	6.92	6.28	2
3	2019-02	26893	Hanro Men's City Pajama Set	230.0	82.11	147.89	1
4	2019-02	14840	Layered look formal Nursing an...	168.0	81.48	86.52	2
5	2019-02	12762	TYR Sport Women's Solid Duraf...	45.45	20.82	24.63	3
6	2019-02	10183	Marshmallow Robe - Lavender	65.99	41.38	24.61	4
7	2019-02	13899	Outdoor Research Women's Gri...	35.67	13.95	21.72	5
8	2019-03	19805	Tommy Hilfiger Men's Two	229.99	99.36	130.63	1

Query 5. Revenue to date for each category Statistics of total daily revenue for each product category in the past 3 months (assuming the current date is April 15, 2022):

```
Select

 CAST(FORMAT_DATE('%Y-%m-%d', t1.delivered_at) AS STRING) as dates,

 t2.category as product_categories,

 round(sum(t1.sale_price),2) as revenue,

 from bigquery-public-data.thelook_ecommerce.order_items as t1

 Join bigquery-public-data.thelook_ecommerce.products

 as t2 on t1.product_id=t2.id

Where t1.status='Complete' and t1.delivered_at BETWEEN '2022-01-15 00:00:00'
AND '2022-04-16 00:00:00'

 Group by dates, product_categories

 Order by dates
```

Row	dates	product_categories	revenue
1	2022-01-15	Leggings	9.95
2	2022-01-15	Shorts	24.97
3	2022-01-15	Underwear	54.5
4	2022-01-15	Sleep & Lounge	166.0
5	2022-01-15	Intimates	152.98
6	2022-01-15	Jeans	92.16
7	2022-01-15	Suits	54.0
8	2022-01-15	Accessories	185.88
9	2022-01-15	Tops & Tees	99.0
10	2022-01-15	Dresses	106.7

B. Cohort Analysis in SQL:

1. Create a dataset includes the following variables: Month, Year, Product_category, TPV, TPO, Revenue_growth, Order_growth, Total_cost, Total_profit, Profit_to_cost_ratio and save that dataset into a VIEW named vw_ecommerce_analyst:

```
With category_data as

(Select FORMAT_DATE('%Y-%m', t1.created_at) as Month,

FORMAT_DATE('%Y', t1.created_at) as Year, t2.category as Product_category,
```

```
round(sum(t3.sale_price),2) as TPV, count(t3.order_id) as TPO,

round(sum(t2.cost),2) as Total_cost

from bigquery-public-data.thelook_ecommerce.orders as t1

Join bigquery-public-data.thelook_ecommerce.products as t2 on t1.order_id=t2.id

Join bigquery-public-data.thelook_ecommerce.order_items as t3 on t2.id=t3.id

Group by Month, Year, Product_category)

Select Month, Year, Product_category, TPV, TPO,

round(cast((TPV - lag(TPV) OVER(PARTITION BY Product_category ORDER BY Year, Month))

      /lag(TPV) OVER(PARTITION BY Product_category ORDER BY Year, Month) as Decimal)*100.00,2) || '%'

      as Revenue_growth,

round(cast((TPO - lag(TPO) OVER(PARTITION BY Product_category ORDER BY Year, Month))

  /lag(TPO) OVER(PARTITION BY Product_category ORDER BY Year, Month) as Decimal)*100.00,2) || '%'

      as Order_growth,

Total_cost, round(TPV - Total_cost,2) as Total_profit,

round((TPV - Total_cost)/Total_cost,2) as Profit_to_cost_ratio

from category_data

Order by Product_category, Year, Month
```

2. Cohort Chart:

```
 With a as

  (Select user_id, amount, FORMAT_DATE('%Y-%m', first_purchase_date) as cohort_month,   created_at,
```

```
      (Extract(year     from     created_at)     -     extract(year     from
first_purchase_date))*12

       +    Extract(MONTH    from    created_at)    -    extract(MONTH    from
first_purchase_date) +1 as index

 from (Select user_id, round(sale_price,2) as amount,

   Min(created_at)  OVER  (PARTITION  BY  user_id)  as  first_purchase_date,
created_at from bigquery-public-data.thelook_ecommerce.order_items ) as b),

  cohort_data as

   (Select   cohort_month,   index,COUNT(DISTINCT   user_id)   as   user_count,
round(SUM(amount),2) as revenue from a

  Group by cohort_month, index

  ORDER BY INDEX),

 --CUSTOMER COHORT--

 Customer_cohort as

 (Select cohort_month,

Sum(case when index=1 then user_count else 0 end) as m1,

Sum(case when index=2 then user_count else 0 end) as m2,

Sum(case when index=3 then user_count else 0 end) as m3,

Sum(case when index=4 then user_count else 0 end) as m4

from cohort_data

Group by cohort_month

Order by cohort_month),

--RETENTION COHORT--

retention_cohort as

(Select cohort_month,

round(100.00* m1/m1,2) || '%' as m1,
```

```
round(100.00* m2/m1,2) || '%' as m2,

round(100.00* m3/m1,2) || '%' as m3,

round(100.00* m4/m1,2) || '%' as m4

from customer_cohort)

--CHURN COHORT--

Select cohort_month,

(100.00 - round(100.00* m1/m1,2)) || '%' as m1,

(100.00 - round(100.00* m2/m1,2)) || '%' as m2,

(100.00 - round(100.00* m3/m1,2)) || '%' as m3,

(100.00 - round(100.00* m4/m1,2))|| '%' as m4

from customer_cohort
```

Cohort Analysis: Cohort Chart

Overall Insights:

Overall, TheLook has recorded a consistent increase in the number of new users each month, indicating the effectiveness of the advertising campaign targeting new users.

However, during the first 4 months after making a purchase or using TheLook's e-commerce site, the rate of returning users in the following month is quite low: it fluctuated below 10% from January 2019 to July 2023

and increased to above 10% in the remaining months of 2023, with the highest being in the first month after October 2023 at 18.28%.

The customer retention rate is low, and TheLook should consider promotional strategies to establish and engage a loyal customer base in order to increase revenue from this group and save on marketing costs.