TheLook Sales Performance Analysis *Using Google Big Query (Advance)*

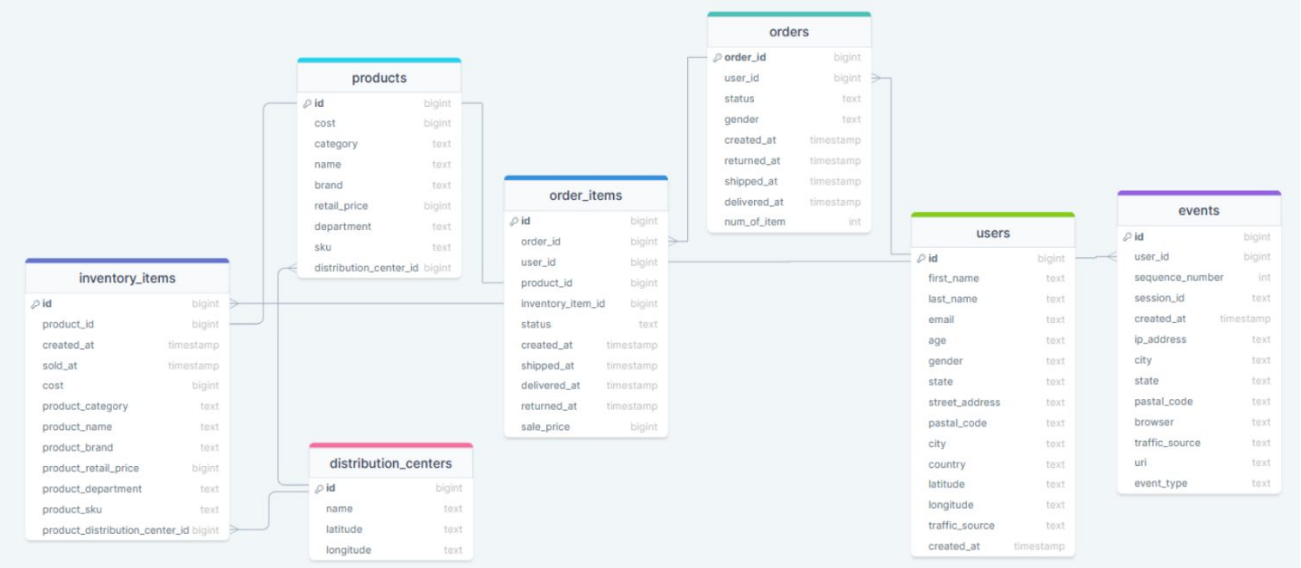
Santa Riyanti T
28 November 2025
RevoU FSDA Batch OCT25

COMPANY OVERVIEW

TheLook is a leading fashion e-commerce company operating in a highly competitive online market. With a commitment to innovation and customer satisfaction, the company leverages data to stay ahead of trends and consumer demands.

By utilizing data driven insights, TheLook continuously optimizes its product offerings, enhances the shopping experience, and maximizes profitability. Through market analysis and consumer behaviour tracking, the company refines its strategies to better serve its customers while maintaining a strong competitive edge in the e-commerce space.

DATASET OVERVIEW



products	
id	bigint
cost	bigint
category	text
name	text
brand	text
retail_price	bigint
department	text
sku	text
distribution_center_id	bigint

orders	
order_id	bigint
user_id	bigint
status	text
gender	text
created_at	timestamp
returned_at	timestamp
shipped_at	timestamp
delivered_at	timestamp
num_of_item	int

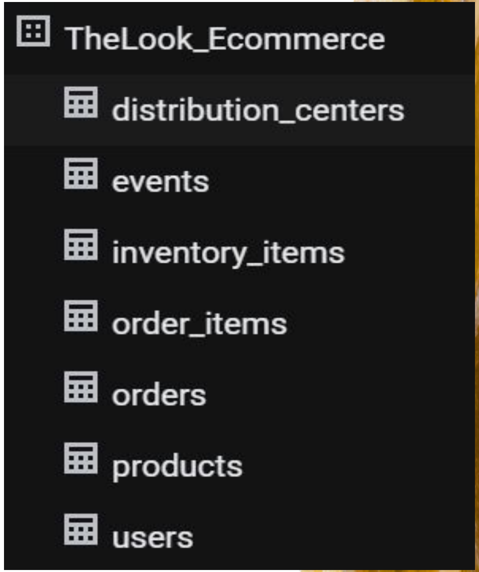
order_items	
id	bigint
order_id	bigint
user_id	bigint
product_id	bigint
inventory_item_id	bigint
status	text
created_at	timestamp
shipped_at	timestamp
delivered_at	timestamp
returned_at	timestamp
sale_price	bigint

inventory_items	
id	bigint
product_id	bigint
created_at	timestamp
sold_at	timestamp
cost	bigint
product_category	text
product_name	text
product_brand	text
product_retail_price	bigint
product_department	text
product_sku	text
product_distribution_center_id	bigint

distribution_centers	
id	bigint
name	text
latitude	text
longitude	text

users	
id	bigint
first_name	text
last_name	text
email	text
age	text
gender	text
state	text
street_address	text
pastal_code	text
city	text
country	text
latitude	text
longitude	text
traffic_source	text
created_at	timestamp

events	
id	bigint
user_id	bigint
sequence_number	int
session_id	text
created_at	timestamp
ip_address	text
city	text
state	text
pastal_code	text
browser	text
traffic_source	text
uri	text
event_type	text



TheLook_Ecommerce

- distribution_centers
- events
- inventory_items
- order_items
- orders
- products
- users

DISCLAIMERS :

- This analysis is based on a publicly available Kaggle dataset and does not reflect real-world financial or business insights.
- TheLook is a fictional entity and the results presented here are purely for educational purposes.
- The queries and insights generated in this assignment should be personalized analysis results.

QUESTION 1 BUSINESS UNDERSTANDING

The company operates a multi-product e-commerce business with multiple distribution centers. The Entity Relationship Diagram (ERD) illustrates the e-commerce business flow, covering the process from user order placement, order processing, item-level transaction details, to product and inventory management at the distribution center. The complete business flow from order placement to fulfillment and inventory tracking is clearly represented in the ERD.

The data structure enables the business to address key strategic questions such as :

- Which product categories are the best sellers
- Who are the top customers by transaction value
- How inventory moves and how to forecast demand
- How each warehouse performs
- Revenue breakdown by product, brand, and region

This ERD is highly suitable for conducting :

- Sales analytics (revenue, order volume, repeat purchase rate)
- Inventory optimization
- Customer analytics (segmentation, retention, customer lifetime value)
- Operational efficiency analysis (fulfillment speed, distribution center performance)

QUESTION 2 IDENTIFY A KEY COLUMN PER TABLE THAT CAN USE WITH ANOTHER TABLE

TABLE	PRIMARY KEY	FOREIGN KEY	JOIN WITH
users	user_id	Master Data Table	events and orders
events	event_id	user_id	users
orders	orders_id	user_id	users and order_items
order_items	order_item_id	order_id and product_id	orders and products
product	product_id	distribution_center_id	order_items, inventory_items and distribution_center
distribution_centers	distribution_center_id	Master Data Table	products and inventory_items
inventory_items	inventory_item_id	product_id and distribution_center_id	products and distribution_centers

QUESTION 3 THE GOAL OF THIS EXERCISE AND HOW THE ERD SUPPORTS BUSINESS OBJECTIVES

The Goal of this exercise is, as data analyst, we should be able to :

- Connect SQL queries to their source tables
- Understand the relationships between tables
- Use the ERD to determine the correct joins

An ERD (Entity Relationship Diagram) gives a structured, visual model of how data is stored and connected across the business.

- Enables a single source of truth
  The ERD ensures that all teams access consistent, accurate data to reduces errors, improves reporting accuracy and ensures all departments rely on the same data definitions.
- Support End to End business operations tracking
  Can showing how marketing , sales and logistics all connect together.
- Enable data driven decision making
  The ERD makes it possible to create complex queries and analytics to be faster business insight dan forecasting.
- Optimizes operational workflows
  Because relationships are clearly mapped, so can lower operational cost and more efficient supply chain.

QUESTION 4 THE ERD SOURCE OF EACH COLUMN PER QUESTION.

Q : Monthly growth of inventory in percentage by product categories, ordered by time descendingly

COLUMN IN QUERY	SOURCE TABLE	DESCRIPTION
category	products	Type Join : Inner Join FROM inventory_items JOIN product.p ON i.products_id = p.id
month / created_at	inventory_items	
total_inventory / cost	products	Aggregation : the results grouping by month (created_at) and category
total_inventory / product_id	inventory_items	

QUESTION 5 Find monthly growth of inventory in percentage breakdown by product categories, ordered by time descendingly

link

```sql
WITH monthly_inventory AS (
  SELECT
    date_trunc(created_at, month) AS month
    , p.category
    , SUM(p.cost * i.product_id) AS total_inventory
FROM fsda-sql-01.TheLook_Ecommerce.inventory_items i
JOIN fsda-sql-01.TheLook_Ecommerce.products p
  ON i.product_id = p.id
GROUP BY 1,2
),
growth_calculate AS (
  SELECT
    category
    , month
    , total_inventory
    , LAG(total_inventory) OVER(PARTITION BY category ORDER BY month) AS prev_month_inventory
  FROM monthly_inventory
)
SELECT
  category
  , month
  , total_inventory
  , prev_month_inventory
  , SAFE_DIVIDE((prev_month_inventory - total_inventory), prev_month_inventory) * 100 AS monthly_growth_percentage
FROM growth_calculate
ORDER BY 2 DESC, 1 ASC
```

The SQL query is designed to calculate the monthly inventory growth percentage for each product category. The logic is divided into two Common Table Expressions (CTE).

Step 1 CTE monthly_inventory (This CTE is used to aggregate the total inventory for each month and product category)

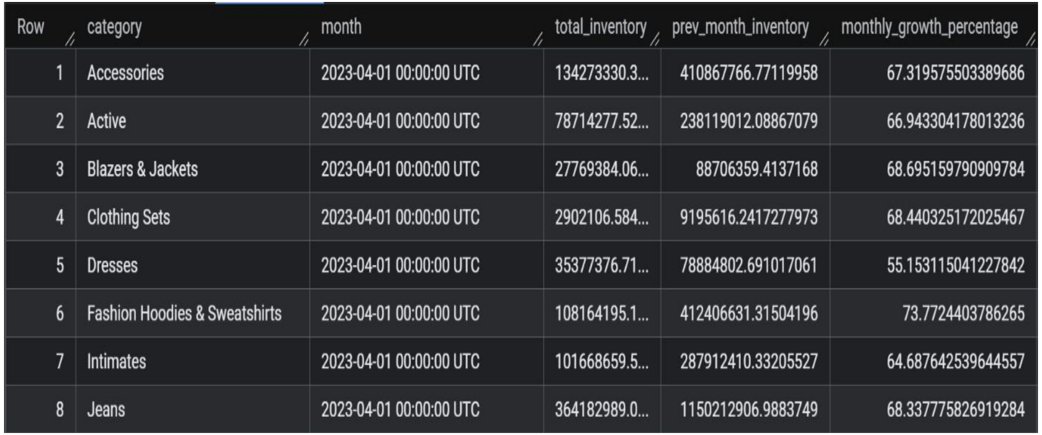
- date_trunc(created_at, month) AS month
- The inventory creation date (created_at) is truncated to the beginning of the month to group data on a monthly basis.
- SUM(p.cost * i.product_id) AS total_inventory calculates the total inventory value. FROM ...inventory_items i JOIN ...products p joins the inventory table with the products table based on product_id to retrieve category details.
- GROUP BY 1, 2 groups the results by month and category

Step 2 CTE growth_calculate (This CTE uses the aggregated monthly data to compare the current month inventory value with the previous month.

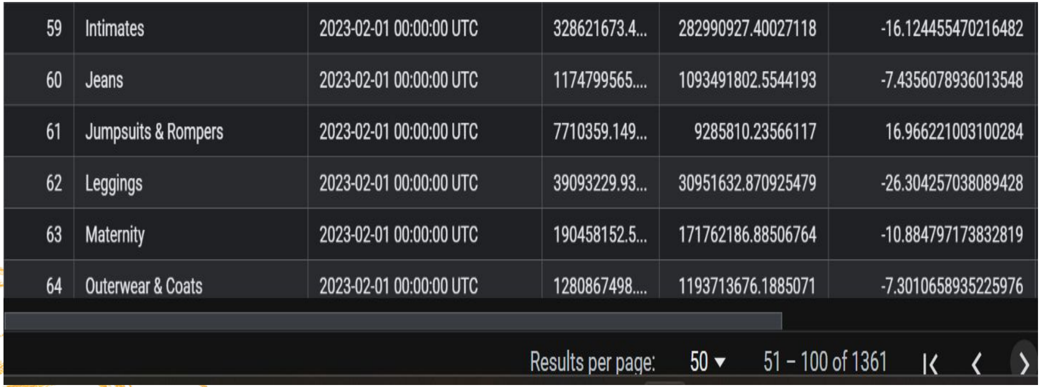
- total_inventory retrieves the inventory value for the current month for each category.
- LAG(total_inventory) OVER(PARTITION BY category ORDER BY month) AS prev_month_inventory, this is the key window function that fetches the total_inventory value from the previous row (the prior month) within the same partition (the same category).
- PARTITION BY category ensures that comparisons are only made across months within the same category.

Step 3 Main Query (The final query selects all columns from growth_calculate and applies the percentage growth formula)

- SAFE_DIVIDE((prev_month_inventory - total_inventory), prev_month_inventory) * 100 AS monthly_growth_percentage, calculates the monthly growth percentage using the standard formula.
- SAFE_DIVIDE is used to handle cases where prev_month_inventory may be zero (such as the first month in each category), preventing errors.
- ORDER BY 2 DESC, 1 ASC, sorts the results by month in descending order (latest first), and then by category in ascending order (alphabetical).



Row	category	month	total_inventory	prev_month_inventory	monthly_growth_percentage
1	Accessories	2023-04-01 00:00:00 UTC	134273330.3...	410867766.77119958	67.319575503389686
2	Active	2023-04-01 00:00:00 UTC	78714277.52...	238119012.08867079	66.943304178013236
3	Blazers & Jackets	2023-04-01 00:00:00 UTC	27769384.06...	88706359.4137168	68.695159790909784
4	Clothing Sets	2023-04-01 00:00:00 UTC	2902106.584...	9195616.2417277973	68.440325172025467
5	Dresses	2023-04-01 00:00:00 UTC	35377376.71...	78884802.691017061	55.153115041227842
6	Fashion Hoodies & Sweatshirts	2023-04-01 00:00:00 UTC	108164195.1...	412406631.31504196	73.7724403786265
7	Intimates	2023-04-01 00:00:00 UTC	101668659.5...	287912410.33205527	64.687642539644557
8	Jeans	2023-04-01 00:00:00 UTC	364182989.0...	1150212906.9883749	68.337775826919284



Row	category	month	total_inventory	prev_month_inventory	monthly_growth_percentage
59	Intimates	2023-02-01 00:00:00 UTC	328621673.4...	282990927.40027118	-16.124455470216482
60	Jeans	2023-02-01 00:00:00 UTC	1174799565....	1093491802.5544193	-7.4356078936013548
61	Jumpsuits & Rompers	2023-02-01 00:00:00 UTC	7710359.149...	9285810.23566117	16.966221003100284
62	Leggings	2023-02-01 00:00:00 UTC	39093229.93...	30951632.870925479	-26.304257038089428
63	Maternity	2023-02-01 00:00:00 UTC	190458152.5...	171762186.88506764	-10.884797173832819
64	Outerwear & Coats	2023-02-01 00:00:00 UTC	1280867498....	1193713676.1885071	-7.3010658935225976

The provided results table offers clear insights

1. Majority of Categories Show Positive Growth (Apr vs. Mar 2023).
   Almost all displayed categories (Accessories, Active, Blazers & Jackets, Clothing Sets, Dresses, Fashion Hoodies & Sweatshirts, Intimates, and Jeans) recorded significantly positive monthly growth percentages, ranging from 55% to nearly 74%. This indicates either an increase in demand or an aggressive procurement strategy for April.
2. Declines in Several Categories (Feb vs. Jan 2023).
   In contrast, the data for February 2023 reveals a notable negative trend across several categories:
   - Intimates (Row 59) : Decline of approximately -16%
   - Jeans (Row 60) : Decline of approximately -7.4%
   - Leggings : Sharp decline of approximately -26.3%
   - Outerwear & Coats : Decline of approximately -7.3%
3. These declines may suggest seasonal factors (for example, reduced demand for winter clothing after January) or potential overstock issues carried over from January.

QUESTION 6 Create monthly retention cohorts and then how many of them (%) coming back for the following months 2022

link

```
WITH first_purchase AS (
  SELECT
    user_id
    , DATE_TRUNC(MIN(created_at), MONTH) AS cohort_month
  FROM fsda-sql-01.TheLook_Ecommerce.orders
  GROUP BY 1
),
user_activity AS (
  SELECT
    user_id
    , DATE_TRUNC(created_at, MONTH) AS activity_month
  FROM fsda-sql-01.TheLook_Ecommerce.orders
),
cohort_activity AS (
  SELECT
    fp.cohort_month
    , ua.activity_month
    , COUNT(DISTINCT ua.user_id) AS active_users
  FROM first_purchase fp
  JOIN user_activity ua
    ON fp.user_id = ua.user_id
  WHERE EXTRACT(YEAR FROM ua.activity_month) = 2022
  GROUP BY 1,2
),
```

```
cohort_size AS (
  SELECT
    cohort_month
    , COUNT(DISTINCT user_id) AS cohort_users
  FROM first_purchase
  GROUP BY 1
)
SELECT
  ca.cohort_month
  , ca.activity_month
  , cs.cohort_users
  , ca.active_users
  , SAFE_DIVIDE(ca.active_users,
cs.cohort_users) * 100 AS
retention_rate_percentage
FROM cohort_activity ca
JOIN cohort_size cs
  ON ca.cohort_month = cs.cohort_month
ORDER BY 1,2
```

Step-by-Step Explanation of the Query

1. Defining First Purchase (first_purchase)

- The Common Table Expression (CTE) first_purchase identifies the first purchase date for each user_id.
- It rounds the date to the beginning of the month (cohort_month).
- This defines the cohort group for each user.

2. Defining User Activity (user_activity)

- The CTE user_activity collects all user transactions.
- Each transaction date is rounded to the beginning of the month (activity_month) for monthly analysis.

3. Calculating Cohort Activity (cohort_activity)

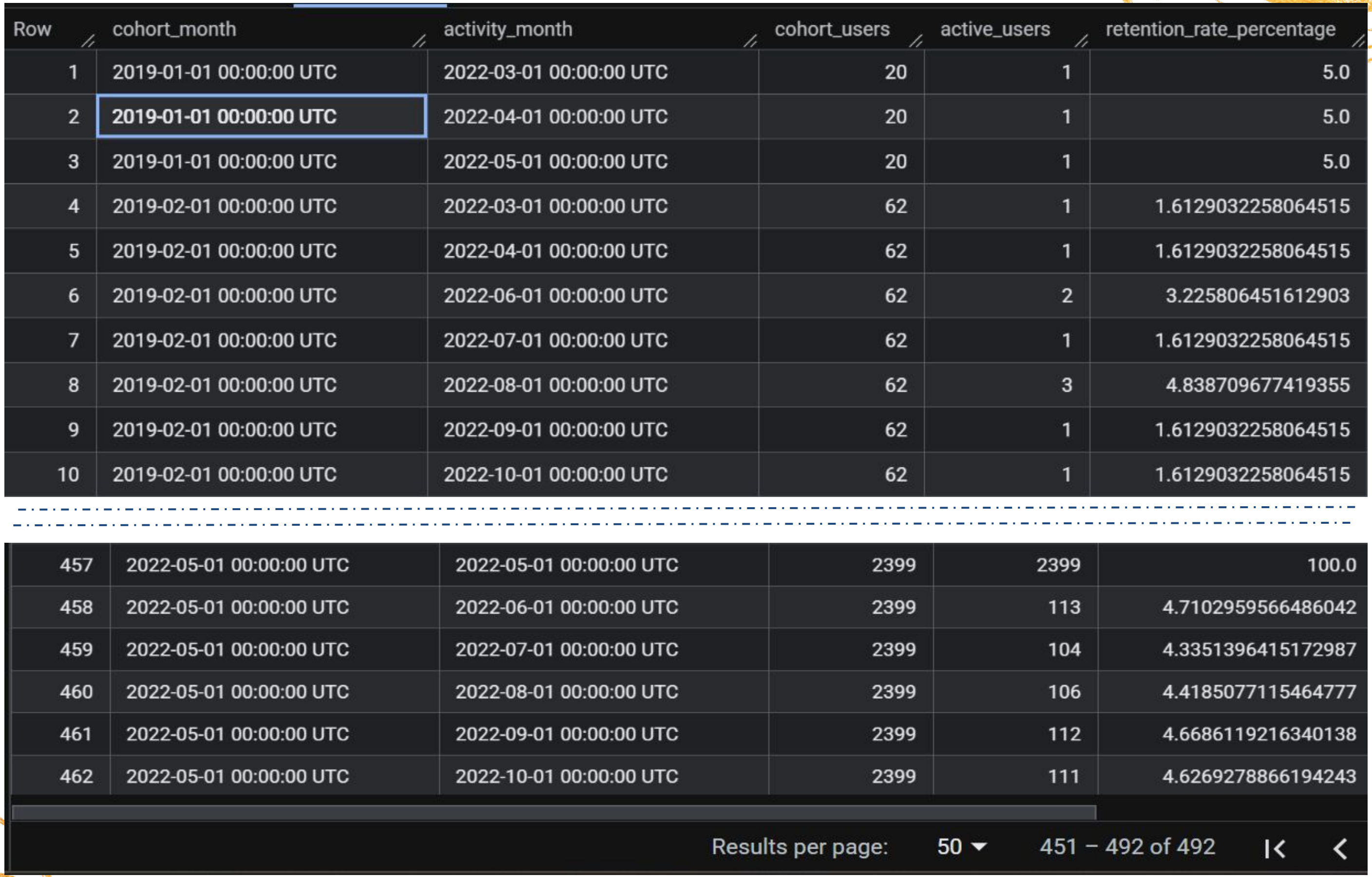
- The CTE cohort_activity joins first_purchase and user_activity.
- This matches each transaction with the user's original cohort month.
- The query then counts the number of unique active users for each combination of cohort_month and activity_month.
- Only activities occurring in 2022 are included, as specified in the requirement.

4. Calculating Cohort Size (cohort_size)

- The CTE cohort_size calculates the total number of unique users in each original cohort (cohort_month) from the first_purchase table.

5. Calculating Final Retention Rate

- The main query joins cohort_activity and cohort_size by cohort_month.
- It computes the retention rate percentage (retention_rate_percentage) by dividing the number of active users in a given month (active_users) by the total users in their original cohort (cohort_users), then multiplying by 100.
- The results are ordered by cohort month and activity month for clear presentation.



Row	cohort_month	activity_month	cohort_users	active_users	retention_rate_percentage
1	2019-01-01 00:00:00 UTC	2022-03-01 00:00:00 UTC	20	1	5.0
2	2019-01-01 00:00:00 UTC	2022-04-01 00:00:00 UTC	20	1	5.0
3	2019-01-01 00:00:00 UTC	2022-05-01 00:00:00 UTC	20	1	5.0
4	2019-02-01 00:00:00 UTC	2022-03-01 00:00:00 UTC	62	1	1.6129032258064515
5	2019-02-01 00:00:00 UTC	2022-04-01 00:00:00 UTC	62	1	1.6129032258064515
6	2019-02-01 00:00:00 UTC	2022-06-01 00:00:00 UTC	62	2	3.225806451612903
7	2019-02-01 00:00:00 UTC	2022-07-01 00:00:00 UTC	62	1	1.6129032258064515
8	2019-02-01 00:00:00 UTC	2022-08-01 00:00:00 UTC	62	3	4.838709677419355
9	2019-02-01 00:00:00 UTC	2022-09-01 00:00:00 UTC	62	1	1.6129032258064515
10	2019-02-01 00:00:00 UTC	2022-10-01 00:00:00 UTC	62	1	1.6129032258064515
457	2022-05-01 00:00:00 UTC	2022-05-01 00:00:00 UTC	2399	2399	100.0
458	2022-05-01 00:00:00 UTC	2022-06-01 00:00:00 UTC	2399	113	4.7102959566486042
459	2022-05-01 00:00:00 UTC	2022-07-01 00:00:00 UTC	2399	104	4.3351396415172987
460	2022-05-01 00:00:00 UTC	2022-08-01 00:00:00 UTC	2399	106	4.4185077115464777
461	2022-05-01 00:00:00 UTC	2022-09-01 00:00:00 UTC	2399	112	4.6686119216340138
462	2022-05-01 00:00:00 UTC	2022-10-01 00:00:00 UTC	2399	111	4.6269278866194243

Insights from the Retention Cohort Table

The retention cohort table provides deep understanding of user behavior and their loyalty to the product or service over time.

Key Insights :

- Low Long Term Retention : Overall, retention rates for older cohorts (January and February 2019) are very low, often with only 1 or 2 active users remaining after several months. This highlights significant challenges in sustaining long-term user engagement.
- Rapid Decline in Early Months : The data shows a sharp drop in active users shortly after the first month, suggesting that users may not perceive lasting value from the product or service.
- Cohort Comparison :
    - May 2022 Cohort (rows 457 onward) : Demonstrates much stronger retention in the first month (100% in the same activity month), but quickly declines in subsequent months, dropping to around 4–5% between June and October 2022. This may indicate that promotions or new features initially attracted users, but the appeal did not sustain.
    - February 2019 Cohort (rows 4 onward) : Shows very low retention from the start (around 1.6% to 4.8% in following months), which may point to issues with user acquisition or product quality at that time.

...Thank You...