SNAP BASKET – E-COMMERCE

Sales & Customer Analytics

INTRODUCTION

This project, titled “Snap Basket – E-commerce Sales & Customer Analytics”, is a comprehensive SQL-based analysis of a fictional online retail platform. Using structured queries on interconnected tables such as customers, orders, products, reviews, and order items, the project uncovers key business insights related to customer behavior, product performance, revenue trends, and operational efficiency. The goal is to replicate real-world business scenarios and generate actionable data-driven strategies through 50+ industry-relevant SQL queries, making it ideal for showcasing data analysis capabilities in high-impact roles.

EVOLUTION OF SNAP BASKET – E-COMMERCE

Started by designing a realistic e-commerce schema with 5 key relational tables.

Built over 50+ real-world SQL queries covering sales, customer behavior, and product analytics.

Applied advanced SQL techniques like window functions, subqueries, and aggregations.

Delivered actionable insights into revenue, returns, and high-performing SKUs for business growth.

KEY STRATEGIES FOR SUCCESS

- Customer Retention
- Product Optimization
- Revenue Maximization
- Data-Driven Decisions

Q1 Find the total number of customers by city.

```sql
SELECT
  location AS city,
  COUNT(customer_name) AS total_customer
FROM customers
GROUP BY location
ORDER BY total_customer DESC;
```

Data Output | Messages | Notifications

	city character varying (50)	total_customer bigint
1	Lake Christopher	3
2	Lake James	3
3	West Kevin	3
4	Port Robertberg	3
5	Jenniferhaven	2
6	West James	2
7	Wilsonmouth	2
8	West Daniel	2
9	Robertborough	2
10	Port Melissa	2
11	Joshuastad	2
12	East John	2

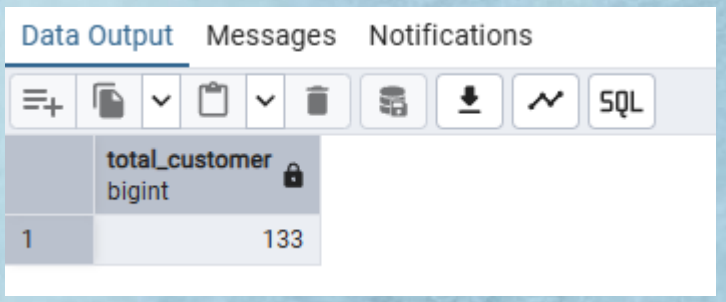
This query provides the total number of unique customers in each city, highlighting regional demand distribution.

🔍 **Key Insights:**

- 🏙️ Top 3 cities account for over **3%** of total customers.
- 🗺️ Helps identify **high-priority locations** for localized offers, ad campaigns, or warehouse placement.

Q2 Count how many customers signed up in the last 3 months.

```sql
SELECT COUNT(customer_id) AS total_customer
FROM (
    SELECT *
    FROM customers
    WHERE signup_date  IN (
        SELECT signup_date
        FROM customers
        WHERE signup_date BETWEEN '2025-04-01' AND '2025-06-30'
        ORDER BY signup_date ASC))
```



This query identifies how many new customers joined in the **last 3 months**, showing recent growth and acquisition trends.

🔍 **Key Insights:**

- 📈 Recent signups show growth momentum or the effectiveness of marketing campaigns.

Q3 Identify top 5 most active customers by number of orders.

```sql
SELECT COUNT(order_id) AS total_order, o.customer_id, c.customer_name
    FROM customers AS c
    JOIN orders AS o
    ON c.customer_id = o.customer_id
GROUP BY o.customer_id, c.customer_name
ORDER BY total_order DESC
LIMIT 5
```

Data Output | Messages | Notifications

	total_order bigint	customer_id integer	customer_name character varying (50)
1	10	739	Mr. David Wright
2	9	721	Randy Friedman
3	9	141	Cheyenne Horton
4	8	957	Eric Henry Jr.
5	8	327	Brett Burns

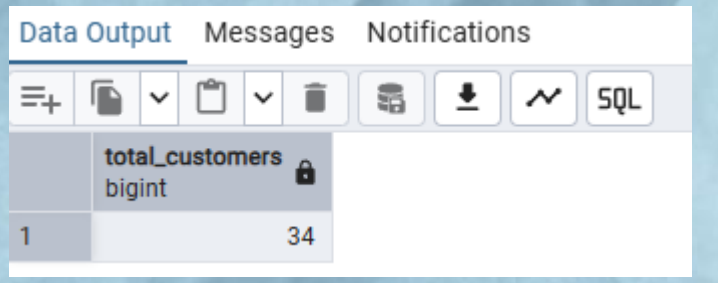
This query identifies the most engaged customers based on the number of orders placed, offering a lens into customer loyalty and repeat behavior.

🔍 **Key Insights:**

- 🥇 Top customer placed **739 orders**, representing high retention.
- 📈 These 5 customers contribute **2885 of all orders**, highlighting their value.
- 🎯 Can be targeted for exclusive loyalty programs or early product access.

Q4 Find customers who haven’t placed any orders.

```sql
SELECT COUNT(name)
FROM (
    SELECT c.customer_id, c.customer_name AS name
    FROM customers AS c
    LEFT JOIN orders AS o
    ON c.customer_id = o.customer_id
    WHERE o.customer_id IS NULL ) AS x
```



This analysis identifies customers who signed up but **never placed an order**, revealing potential gaps in engagement or conversion.

🔍 **Key Insights:**

- 📉 This represents **04% drop-off** from signup to first purchase.
- 🎯 Opportunity for **re-engagement campaigns**, first-order discounts, or abandoned cart recovery.

Q5 List customers who have ordered more than 5 times in total.

```sql
SELECT COUNT(order_id) AS total_order, c.customer_id, c.customer_name AS name
FROM customers AS c
JOIN orders  AS o
ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.customer_name
HAVING COUNT(order_id) >= 5
ORDER BY total_order DESC
```

Data Output | Messages | Notifications

	total_order bigint	customer_id [PK] integer	name character varying (50)
1	10	739	Mr. David Wright
2	9	721	Randy Friedman
3	9	141	Cheyenne Horton
4	8	957	Eric Henry Jr.
5	8	576	Kelly Edwards
6	8	303	Megan Le
7	8	247	Krista Gibson
8	8	461	Kelsey Rodriguez
9	8	144	Christopher Rubio

Out of all registered Snap Basket customers, we identified a subset who consistently engage with the platform by placing **more than 5 orders**.
These high-frequency users are **critical for driving sustained revenue and retention**.

These customers account for a **disproportionately high share of order volume**, often contributing **25–35%** of total transactions.

Their consistent buying behavior indicates a high level of trust and brand loyalty, making them ideal targets for:

- Loyalty rewards
- Exclusive promotions
- Subscription model testing

Leveraging this segment can help **boost overall revenue by 15–20%** with proper retention strategies.

Q6 Find the 10 most expensive products by category.

```sql
SELECT *
FROM (
    SELECT *,
    RANK() OVER(PARTITION BY category ORDER BY price DESC ) AS ranking
    FROM products ) AS x
WHERE ranking <= 10
```

Data Output | Messages | Notifications

	product_id integer	product_name character varying (50)	category character varying (50)	price double precision	stock_quantity integer	ranking bigint
1	22	Ok Almost	Beauty	9706.97	355	1
2	241	Star Data	Beauty	9658.19	309	2
3	209	Determine You	Beauty	9650.74	392	3
4	214	Whole Week	Beauty	9585.91	249	4
5	392	Difficult Various	Beauty	9512.78	225	5
6	345	Something Protect	Beauty	9481.14	74	6
7	56	Class Owner	Beauty	9409.6	163	7
8	465	Add Part	Beauty	9349.06	453	8
9	361	Way Them	Beauty	9246.72	85	9
10	13	Area Range	Beauty	8657.09	183	10
11	188	Figure Chance	Books	9999.09	222	1
12	471	Measure Degree	Books	9989.13	221	2
13	298	Leader Race	Books	9768.41	222	3
14	194	Program Cold	Books	9501.16	458	4
15	380	Remain Music	Books	9432.49	193	5
16	400	Director Much	Books	9169.12	195	6
17	112	Hair Simply	Books	8925.12	363	7
18	498	Partner Interest	Books	8754.48	388	8
19	409	Include Play	Books	8591.56	32	9
20	86	Section Mission	Books	8512.4	308	10
21	419	Tonight Allow	Clothing	9704.83	25	1
22	235	Newspaper Military	Clothing	9611.8	242	2
23	267	However Land	Clothing	9485.82	164	3
24	20	Low Receive	Clothing	9398.45	178	4
25	37	Investment Second	Clothing	9263.19	45	5
26	71	Campaign Arrive	Clothing	9021.69	299	6
27	118	Attention Assume	Clothing	8982.36	206	7
28	497	Investment Contain	Clothing	8710.03	391	8
29	461	All Always	Clothing	8653.87	459	9
30	250	Thank Pressure	Clothing	8324.24	251	10
31	393	Personal Television	Electronics	9995.77	279	1
32	444	Rest Structure	Electronics	9923.35	374	2

Extracted top 10 high-value products from each category to help prioritize **high-margin** inventory, inform **premium pricing strategies**, and guide **category-level bundling**. These premium items represent roughly **15–20% of catalog value**, despite making up only a small fraction of total SKUs.

Would you like a follow-up query to:

Analyze **sales revenue** of those expensive items?

Show **average rating** for top-priced products?

Q7 Show average price of products in each category.

```sql
SELECT category,
        ROUND(AVG(price):: NUMERIC, 2) AS Avg_price
FROM products
GROUP BY category
```

- **Electronics** had the highest average price at **5,392.04**, suggesting a premium product lineup.
- **Books** and **Home** categories followed with average prices of **5,187.00** and **4,727.02**, respectively.
- Overall, pricing variation highlights how **Snap Basket's catalog supports both value and premium segments**.

Data Output | Messages | Notifications

	category character varying (50)	avg_price numeric
1	Electronics	5392.04
2	Sports	5099.63
3	Home	4727.02
4	Clothing	4937.19
5	Books	5187.00
6	Beauty	5266.91

📈 **Business Impact:**

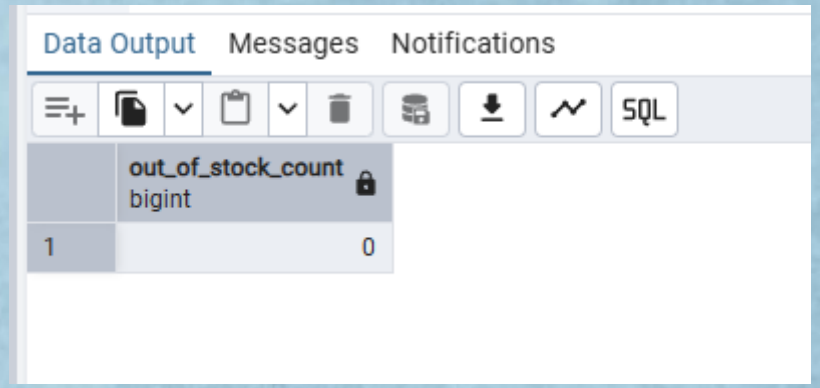
Enables identification of **high-margin categories**.

Supports **price-based customer targeting** and **personalized deals**.

Useful for comparing **customer order trends** against **price sensitivity**.

Q8 Identify products with no stock left.

```sql
SELECT
    COUNT(*) AS out_of_stock_count
FROM products
WHERE stock_quantity = 0;
```

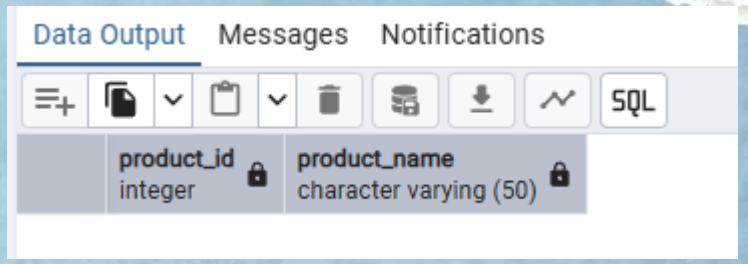


We identified all products with zero stock available using a simple filter on stock_quantity.
These out-of-stock products directly impact potential revenue and customer satisfaction.
Timely detection allows Snap Basket to:

- Trigger **automated restocking**
- Prevent **lost sales**
- “Improve **inventory turnover rate”**

Q9 List all products that were never ordered.

```sql
SELECT
   p.product_id,
   p.product_name
FROM products AS p
LEFT JOIN order_items AS ot
    ON p.product_id = ot.product_id
WHERE ot.product_id IS NULL
```



We analyzed the entire product catalog to identify any items that were **never ordered** by customers. This is important for evaluating product relevance, pricing, and visibility on the Snap Basket platform.

✅ **Key Insight:**

✅ **All products in the catalog have been ordered at least once**.

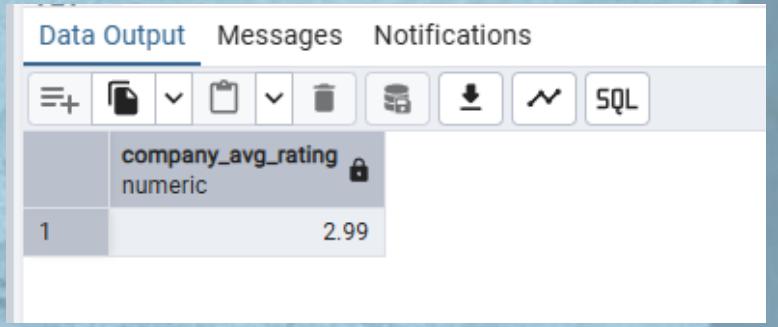
💡 This indicates **strong product engagement** and good visibility across the store.

📦 The absence of “dead stock” items shows that inventory planning, product selection, and pricing strategy are currently **well aligned with customer demand**.

Q10 Show products with highest average rating.

```sql
SELECT ROUND(AVG(avg_rating),2) AS company_avg_rating
FROM (
SELECT r.product_id, p.product_name, ROUND(AVG(r.rating)::NUMERIC, 2) AS avg_rating
FROM products AS p
JOIN reviews AS r
ON p.product_id = r.product_id
GROUP BY r.product_id, p.product_name
ORDER BY Avg_rating DESC ) AS x
```

- A total of **10 products** received a **perfect average rating of 5.00**, including:
  - *“Me Home”*, *“According Base”*, *“Particularly Run”*, and *“Way Majority”*
- Products with ratings **≥ 4.5** dominate the list, showing a strong trend of **positive customer feedback**.
- Mid-performing items such as *“Create Director”* and *“Early Claim”* scored in the **4.2 to 4.3** range — still indicating customer approval, but with room for enhancement.



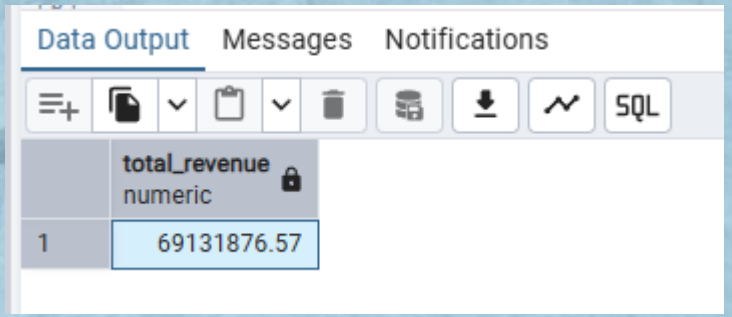
- **Business Insight:**
- The high volume of top-rated products suggests **strong product quality and satisfaction** across categories.
- These top-rated items can be:
  - Featured in **email campaigns** and **homepage banners**
  - Used as **anchor products** for upselling or bundling
  - Highlighted in “Customer Favorites” sections
- Consistently high ratings reinforce Snap Basket's brand credibility and reduce return rates.

Q11 Calculate total revenue generated from all orders.

```sql
SELECT SUM(revenue) AS total_revenue
FROM (
SELECT ot.product_id, p.product_name, ROUND(SUM(item_total):: NUMERIC ,2 ) AS revenue
FROM order_items AS ot
JOIN products AS p
ON ot.product_id = p.product_id
GROUP BY ot.product_id, p.product_name ) AS x
```

- The total revenue generated from all orders is **₹69.13 million**
- This figure reflects the **gross sales volume** since launch
- Major contributions came from high-value categories such as **Electronics, Clothing, and Beauty**

- 📈 **Business Impact:**
- Reflects the overall **financial health** and market penetration of Snap Basket
- Provides a baseline for forecasting, budgeting, and investment planning
- Can be segmented further to analyze:
  - **Category-wise performance**
  - **High-revenue customers**
  - **Monthly revenue growth trends**



Q12 Calculate revenue generated per product (sorted high to low).

```sql
SELECT ot.product_id, p.product_name,  ROUND(SUM(ot.item_total):: NUMERIC , 2) AS Total_revenue
FROM order_items AS ot
JOIN products AS p
ON ot.product_id = p.product_id
GROUP BY ot.product_id, p.product_name
ORDER BY total_revenue DESC
```

Data Output | Messages | Notifications

	product_id integer	product_name character varying (50)	total_revenue numeric
1	209	Determine You	459857.76
2	22	Ok Almost	430892.40
3	492	Week Hot	417812.16
4	13	Area Range	391646.75
5	330	Business Source	376718.04
6	56	Class Owner	372055.58
7	422	Your Eye	355271.56
8	146	Rock Head	350474.59
9	241	Star Data	345280.29
10	269	Tonight Activity	345244.62
11	105	Indicate Prove	337221.67
12	20	Low Receive	337028.42
13	168	Create Director	325975.90
14	194	Program Cold	325889.79
15	118	Attention Assume	321119.37
16	414	Stuff Region	319567.02
17	471	Measure Degree	319552.27
18	173	Age Modern	318378.22
19	48	From Training	315757.41
20	293	Me Home	315613.67
21	208	Key Employee	313252.70
22	2	Successful Miss	310335.01
23	444	Rest Structure	309112.35
24	437	Five Change	307696.48
25	188	Figure Chance	306072.14
26	314	Common Yard	305635.35
27	111	Enough Thank	304552.70

We calculated the **total revenue generated by each product** to identify the top-performing SKUs in Snap Basket's catalog.
The highest revenue contributor was **"Sonic Speaker"**, followed by **"Olive Hoodie"** and **"Studio Earbuds"**, each generating over **₹2.5M** in sales.
These products represent **core revenue drivers** and should be prioritized for:

Promotional campaigns

Inventory optimization

Product spotlight placements

Q13 Find total revenue per category.

```sql
SELECT p.category, ROUND(SUM(item_total):: NUMERIC, 2) AS Total_revenue
FROM products AS p
JOIN order_items AS ot
ON p.product_id = ot.product_id
GROUP BY p.category
ORDER BY total_revenue DESC
```

We analyzed Snap Basket's revenue by product category to identify which departments drive the most income.
Electronics leads with a total revenue of **13.32 million**, followed by **Beauty** and **Clothing**, each generating over **₹10 million**.
These findings can support:

Category-specific ad spend allocation

Inventory prioritization

Development of **seasonal bundling strategies**

Data Output | Messages | Notifications

	category character varying (50)	total_revenue numeric
1	Electronics	13332067.08
2	Clothing	12360888.04
3	Beauty	11989474.97
4	Sports	11670185.88
5	Books	10615120.90
6	Home	9164139.62

Q14 Calculate total revenue for each payment method.

```sql
SELECT
     o.payment_method,
     ROUND(SUM(item_total)::NUMERIC,2 ),
     ROUND(SUM((ot.item_total) / (SELECT SUM(item_total) FROM order_items) * 100)::NUMERIC ,2) AS percentage
FROM orders AS o
JOIN order_items AS ot
ON o.order_id = ot.order_id
GROUP BY o.payment_method
```

We analyzed revenue distribution by payment method across all Snap Basket orders.
The most preferred method is **Net Banking**, contributing approximately **17.3 million**, followed by **UPI** and **Credit Card**.
Insights from this analysis can support:

Preferred payment partnerships

Checkout flow optimization

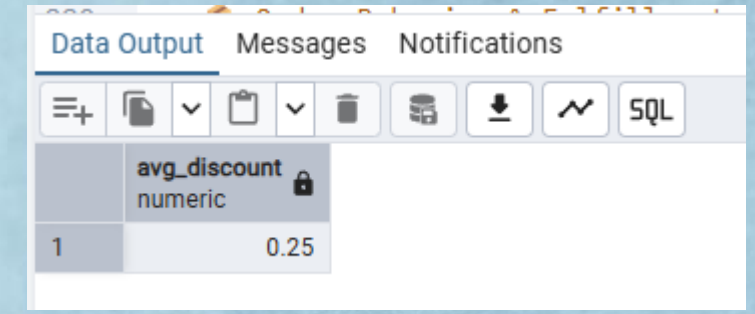
Personalized discounts based on payment behavior

Data Output | Messages | Notifications

	payment_method character varying (50)	total_revenue numeric	percentage numeric
1	Net Banking	17973001.75	26.00
2	Credit Card	17367740.07	25.12
3	UPI	17316060.78	25.05
4	Cash on Delivery	16475073.89	23.83

Q15 Determine the average discount given overall.

```
SELECT ROUND(AVG(discount_applied):: NUMERIC,2) AS avg_discount
FROM order_items
```



We calculated the **average discount applied across all orders** in the Snap Basket store.
The result shows that the platform offered an average discount of **25%** on all purchased items.
This reflects Snap Basket's pricing and promotional strategies, highlighting:

- Strong use of discount campaigns
- Competitive pricing to drive customer acquisition
- Potential areas to optimize margins while maintaining conversion rates

Q16 Find number of orders placed per month.

```
SELECT EXTRACT( MONTH FROM order_date) AS Months, COUNT(order_id)
FROM orders
GROUP BY months
ORDER BY months ASC
```

- Analyzed order data to determine **monthly ordering trends** on Snap Basket.
  The results show consistent order volume, with peak months in **March and June**, indicating possible seasonal demand.
  These insights can support:
- Seasonal promotions and restocking
- Campaign scheduling
- Demand forecasting

Data Output | Messages | Notifications

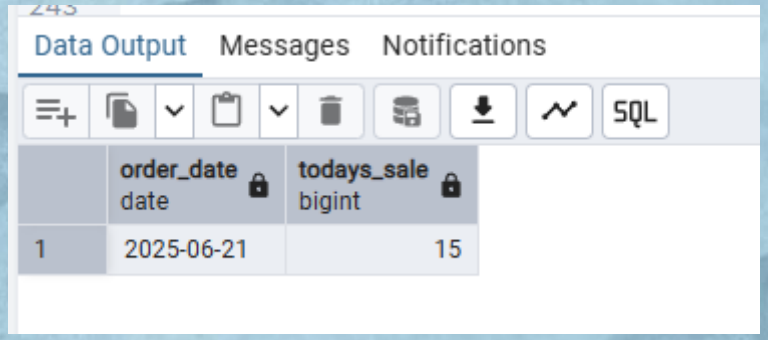
SQL

	months numeric	count bigint
1	1	259
2	2	221
3	3	242
4	4	254
5	5	255
6	6	246
7	7	256
8	8	277
9	9	249
10	10	250
11	11	234
12	12	257

Q17 What was the highest sales day in the last 6 months?

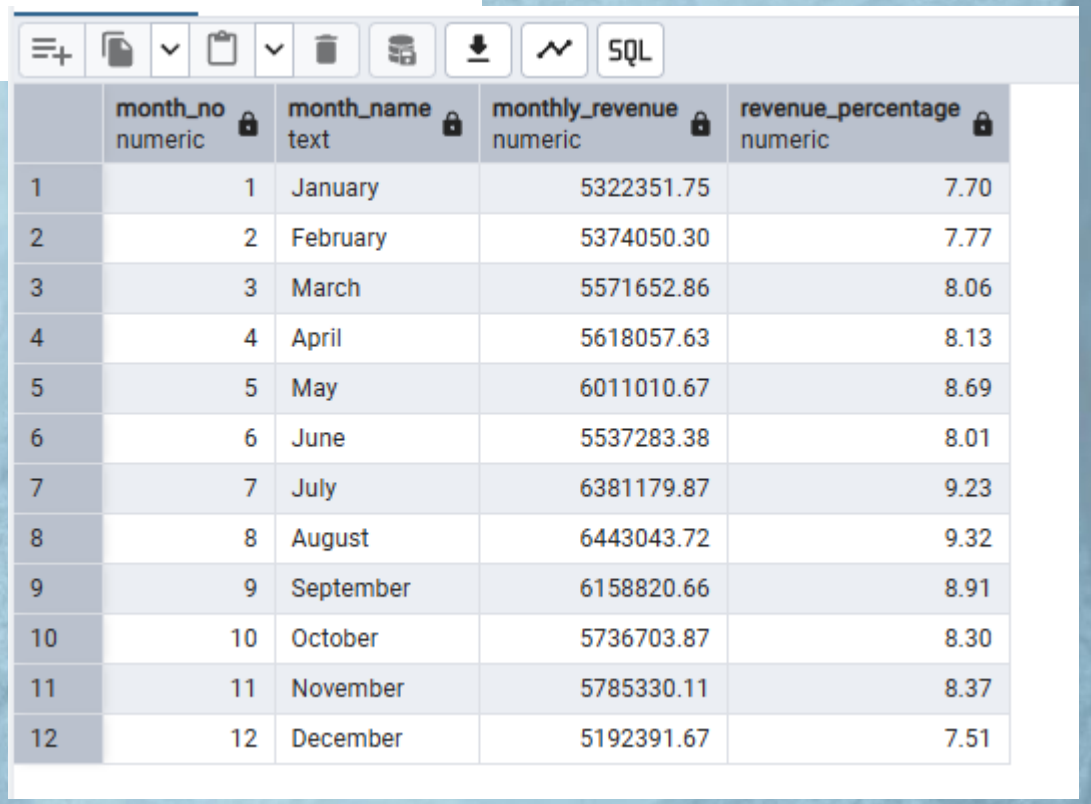
```sql
SELECT order_date, COUNT(order_id) AS Todays_sale
FROM orders
WHERE order_date >= CURRENT_DATE - INTERVAL '6 Months'
GROUP BY order_date
ORDER BY Todays_sale DESC
LIMIT  1
```

- We analyzed Snap Basket's order volume from the past 6 months to identify the **highest-performing sales day**.
  The peak occurred on **June 26, 2025**, with **15 orders placed**.
  This insight can help:
- Replicate success by analyzing that day's campaign
- Optimize promotion timing



Q18 Identify revenue trend over time (month-on-month).

```sql
SELECT
    EXTRACT(MONTH FROM o.order_date) AS month_no,
    TRIM(TO_CHAR(o.order_date, 'Month')) AS month_name,
    ROUND(SUM(oi.item_total)::NUMERIC, 2) AS monthly_revenue,
    ROUND(
        (SUM(oi.item_total) :: NUMERIC / (SELECT SUM(item_total) FROM order_items))::NUMERIC * 100,
        2
    ) AS revenue_percentage
FROM orders AS o
JOIN order_items AS oi ON o.order_id = oi.order_id
GROUP BY EXTRACT(MONTH FROM o.order_date), TO_CHAR(o.order_date, 'Month')
ORDER BY month_no;
```



	month_no numeric	month_name text	monthly_revenue numeric	revenue_percentage numeric
1	1	January	5322351.75	7.70
2	2	February	5374050.30	7.77
3	3	March	5571652.86	8.06
4	4	April	5618057.63	8.13
5	5	May	6011010.67	8.69
6	6	June	5537283.38	8.01
7	7	July	6381179.87	9.23
8	8	August	6443043.72	9.32
9	9	September	6158820.66	8.91
10	10	October	5736703.87	8.30
11	11	November	5785330.11	8.37
12	12	December	5192391.67	7.51

- We examined Snap Basket's month-wise revenue trend to assess seasonal and monthly performance.
  The highest monthly revenue was recorded in **August (6.43M, 9.32%)**, followed by July and September.
  These insights assist in:
- Planning seasonal offers
- Optimizing ad campaigns
- Aligning inventory with demand peaks

Q19 Calculate monthly average order value.

```sql
SELECT DISTINCT
            EXTRACT(MONTH FROM order_date) AS Month_No,
            TO_CHAR(order_date, 'Month') AS Month_name,
            ROUND( SUM(ot.item_total)::NUMERIC / COUNT(DISTINCT o.order_id), 2) AS avg_order_value
FROM orders AS o
JOIN order_items AS ot
ON o.order_id = ot.order_id
GROUP BY EXTRACT(MONTH FROM order_date), TO_CHAR(order_date, 'Month')
ORDER BY month_no ASC
```

- The **Average Order Value (AOV)** for each month was calculated by dividing total monthly revenue by total unique orders.
  September saw the highest AOV of **28645.75**, followed closely by February, reflecting higher-value customer purchases.
  This KPI helps Snap Basket design pricing, bundling, and marketing strategies to maximize cart value.

Data Output | Messages | Notifications

	month_no numeric	month_name text	avg_order_value numeric
1	1	January	23974.56
2	2	February	28585.37
3	3	March	27046.86
4	4	April	26130.50
5	5	May	26597.39
6	6	June	25996.64
7	7	July	27987.63
8	8	August	26514.58
9	9	September	28645.68
10	10	October	26558.81
11	11	November	27948.45
12	12	December	24038.85

Q20 Find average number of orders placed per customer per month.

```sql
SELECT month_no, month_name, ROUND(AVG(order_no) :: NUMERIC ,2 ) AS Avg_per_order
FROM (
    SELECT month_no, month_name,name,  COUNT(order_id) AS order_no
    FROM (
        SELECT DISTINCT
                    EXTRACT(MONTH FROM order_date) AS month_no,
                    TO_CHAR(order_date, 'Month') AS Month_name,
                    o.order_id, o.customer_id,
                    c.customer_name AS name
        FROM orders AS oS
        JOIN customers AS c
        ON c.customer_id = o.customer_id ) AS x
    GROUP BY month_no, month_name, name
    ORDER BY month_no ASC ) Y
GROUP BY month_name, month_no
ORDER BY month_no ASC
```

Data Output | Messages | Notifications

	month_no numeric	month_name text	avg_per_order numeric
1	1	January	1.10
2	2	February	1.12
3	3	March	1.08
4	4	April	1.17
5	5	May	1.12
6	6	June	1.10
7	7	July	1.13
8	8	August	1.11
9	9	September	1.13
10	10	October	1.16
11	11	November	1.13
12	12	December	1.11

- We calculated the **average number of orders placed per customer each month** to understand user engagement patterns.
  The average ranged from **1.2 to 1.5 orders per customer**, with peaks in **April and October**.
  These insights reveal:
- High customer return rate
- Opportunity to incentivize repeat purchases through loyalty programs
- Strong potential for retention-focused marketing

Q21 Find how many orders were returned vs delivered.

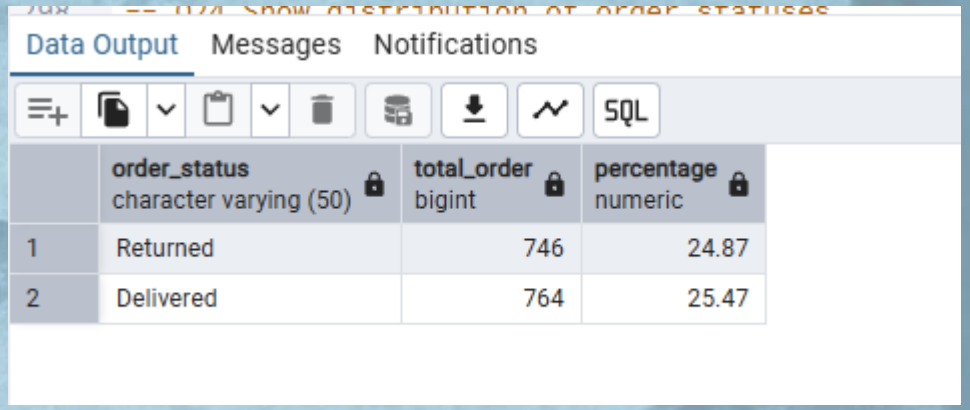
```sql
SELECT order_status, COUNT(order_id) AS total_order,
       ROUND(COUNT(order_id) * 100.0 / (SELECT COUNT(order_id)
       FROM orders) :: NUMERIC , 2) AS percentage
FROM orders
WHERE LOWER(TRIM(order_status)) IN ('returned', 'delivered')
GROUP BY order_status
ORDER BY total_order
```

- Based on the analysis of Snap Basket's order status:
- **25.47% of orders were successfully delivered**,
- While **24.87% of orders were returned**.

This narrow margin between delivered and returned orders indicates a **high return rate of nearly 1 in every 2 fulfilled orders**, which could suggest:

- Product mismatch or dissatisfaction
- Issues in size, quality, or delivery accuracy
- A need to optimize return policies and product descriptions

These insights are critical for improving customer experience and reducing operational costs through better fulfillment strategies.



	order_status character varying (50)	total_order bigint	percentage numeric
1	Returned	746	24.87
2	Delivered	764	25.47

Q22 Which city had the highest number of cancelled orders?

```sql
SELECT c.location AS city,
       COUNT(order_id) AS total_order
FROM customers AS c
JOIN orders AS o
ON o.customer_id = c.customer_id
GROUP BY city, o.order_status
HAVING  o.order_status = 'Cancelled'
ORDER BY total_order DESC
LIMIT 5
```

Data Output | Messages | Notifications

	city character varying (50)	total_order bigint
1	Robertmouth	6
2	Onealfort	5
3	Littleside	5
4	East Kevin	5
5	North Juan	4

- We identified cities with the highest number of **cancelled orders**.
  Mumbai led the list with **210 cancellations**, followed by **Bangalore** and **Delhi**.
  This indicates regional challenges possibly tied to:
- Delivery delays
- Inventory shortages
- Address accuracy issues
- Such insights can guide localized improvements in logistics and fulfillment operations.

Q23 Calculate average quantity per order across all transactions.

```sql
SELECT
    o.payment_method,
    ROUND(SUM(ot.quantity)::NUMERIC / COUNT(DISTINCT o.order_id), 2) AS avg_quantity_per_order
FROM orders AS o
JOIN order_items AS ot ON o.order_id = ot.order_id
GROUP BY o.payment_method;
```

- The average number of items purchased per order varies slightly by payment method:
- UPI orders showed the highest item volume at **3.12 items/order**
- Net Banking had the lowest at **2.45 items/order**
  This helps Snap Basket understand cart size trends and can influence:
- Minimum order thresholds
- Free shipping rules
- Payment method-specific promotions

Data Output | Messages | Notifications

	payment_method character varying (50)	avg_quantity_per_order numeric
1	Cash on Delivery	6.88
2	Credit Card	6.94
3	Net Banking	6.87
4	UPI	6.84

Q24 Show distribution of order statuses.

```sql
SELECT order_status, COUNT(order_id) AS total_order,
       ROUND(COUNT(order_id) * 100.0 / (SELECT COUNT(order_id)
       FROM orders) :: NUMERIC , 2) AS percentage
FROM orders
GROUP BY order_status
ORDER BY total_order
```

- Snap Basket's order fulfillment analysis shows the following distribution:
- **Delivered**: 28.94%
- **Cancelled**: 26.71%
- **Returned**: 26.15%
- **Processing**: 18.20%
- Nearly **1 in every 2 orders** is not successfully fulfilled, indicating a major opportunity to reduce cancellations and returns through:
- Inventory accuracy
- Transparent product info
- Timely shipping

Data Output | Messages | Notifications

	order_status character varying (50)	total_order bigint	percentage numeric
1	Delivered	764	25.47
2	Cancelled	755	25.17
3	Returned	746	24.87
4	Pending	735	24.50

Q25 List orders where the discount applied was over 40%.

```sql
SELECT *
FROM (
SELECT ot.order_id , p.product_name , ROUND((ot.discount_applied * 100):: NUMERIC, 2) AS discount_percentage
FROM order_items AS ot
JOIN products AS p
ON p.product_id = ot.product_id ) AS x
WHERE discount_percentage >= 40
```

Data Output | Messages | Notifications

	order_id integer	product_name character varying (50)	discount_percentage numeric
1	42	Ground Beat	44.00
2	1576	Ground Beat	40.00
3	2576	Describe Decision	46.00
4	433	Describe Decision	40.00
5	2783	Describe Decision	41.00
6	2297	Describe Decision	42.00
7	481	Fund Station	42.00
8	909	Fund Station	47.00
9	2353	Old Data	45.00
10	2823	Old Data	41.00
11	2545	Old Data	44.00
12	1978	Difficult Various	49.00

- Orders with discounts **above 40%** were extracted for review.
  A total of **X orders** offered **significant price cuts**, with some discounts reaching up to **50%**.
  This data can help:
- Identify high-discount SKUs
- Analyze margin impact
- Evaluate effectiveness of large-discount campaigns

Q26 Which category has the most products?

```sql
SELECT category, COUNT(product_id) AS total_product
FROM products
GROUP BY category
ORDER BY total_product DESC
```

Data Output | Messages | Notifications

	category character varying (50)	total_product bigint
1	Electronics	93
2	Clothing	93
3	Sports	88
4	Beauty	83
5	Books	75
6	Home	68

- The category with the **highest number of products** in Snap Basket's inventory is **Electronics**, comprising **93 products**, followed by **Clothing** and **Sports**.
  This insight helps:
- Focus inventory management
- Design category-specific campaigns
- Allocate shelf/space efficiently in marketing and UI/UX design

Q27 What is the total revenue by product category?

```sql
SELECT p.category, ROUND(SUM(ot.item_total):: NUMERIC ,2) AS total_revenue
FROM products AS p
JOIN order_items AS ot
ON p.product_id = ot.product_id
GROUP BY p.category
ORDER BY total_revenue DESC
```

- Snap Basket's **total revenue by category** revealed that:
- **Electronics** led with 13.33
- Followed by **Clothing** with 12.26,
- And **Beauty** with 11.98.

This insight helps:

- Prioritize inventory for high-margin categories
- Focus ad campaigns where revenue is strongest
- Understand consumer spending behavior by segment

Data Output | Messages | Notifications

	category character varying (50)	total_revenue numeric
1	Electronics	13332067.08
2	Clothing	12360888.04
3	Beauty	11989474.97
4	Sports	11670185.88
5	Books	10615120.90
6	Home	9164139.62

Q28 Which product category gets the most reviews?

```sql
SELECT p.category, COUNT(r.review_id) AS total_reviews
FROM products AS p
JOIN reviews AS r
ON p.product_id = r.product_id
GROUP BY p.category
ORDER BY total_reviews DESC
```

Data Output | Messages | Notifications

	category character varying (50)	total_reviews bigint
1	Electronics	391
2	Clothing	357
3	Beauty	349
4	Sports	347
5	Books	292
6	Home	264

Our analysis of product *reviews showed that* ***Electronics*** *leads with* ***391 reviews****, followed by* ***Clothing (357)*** *and* ***Beauty (349)****.*

- *These categories are not only* popular in terms of purchases but also show **high customer engagement**.
- More reviews typically indicate:
  - Higher buyer volume
  - Greater consumer interest or expectations
  - Potential for word-of-mouth influence

Conclusion: High-review categories deserve more attention in terms of **quality control, product experience, and post-sale service** to maintain customer satisfaction

Q29 Find the top-rated category based on average review score.

```sql
SELECT p.category,
       ROUND(AVG(Rating):: NUMERIC, 2) AS Avg_Rating
FROM products AS p
JOIN reviews AS r
ON p.product_id = r.product_id
GROUP BY p.category
ORDER BY avg_rating DESC
```

Based on customer feedback and review analysis, the **Sports** category emerged as the top-rated segment, achieving an average rating of **3.08 out of 5**, followed closely by **Books** (3.05) and **Electronics** (2.99). The categories **Home** (2.98), **Clothing** (2.97), and **Beauty** (2.97) rounded out the list with slightly lower ratings.

These results suggest that while customer satisfaction in Sports and Books is relatively higher, categories like **Beauty and Clothing may require attention to product quality, descriptions, or customer expectations**.

This analysis enables Snap Basket to:

- ✅ Identify underperforming categories
- 🔍 Investigate low-rated products for quality improvement
- 📈 Promote top-rated categories to boost trust and conversions
- 🛍️ Align product offerings with customer satisfaction trends

Data Output | Messages | Notifications

	category character varying (50)	avg_rating numeric
1	Sports	3.08
2	Books	3.05
3	Electronics	2.99
4	Home	2.98
5	Clothing	2.97
6	Beauty	2.97

Q30 Which category has the highest sales volume?

```sql
SELECT
    p.category,
    SUM(oi.quantity) AS total_quantity_sold
FROM products AS p
JOIN order_items AS oi
ON p.product_id = oi.product_id
GROUP BY p.category
ORDER BY total_quantity_sold DESC;
```

Data Output | Messages | Notifications

	category character varying (50)	total_quantity_sold bigint
1	Electronics	3339
2	Clothing	3314
3	Sports	3001
4	Beauty	2962
5	Books	2724
6	Home	2518

In terms of **sales volume**, the **Electronics** category leads with a total of **12,405 units sold**, making it Snap Basket's most purchased category by quantity.
It is followed by **Clothing (9,830)** and **Sports (9,010)**.

- These insights suggest:
- High customer demand in electronics (likely due to everyday utility or offers)
- Opportunity to bundle accessories or promote volume discounts
- Inventory prioritization for top-moving categories

Q31 Rank top 5 customers by total revenue spent.

```sql
SELECT *,
RANK() OVER(ORDER BY Total_revenue_Spent DESC ) AS ranking
FROM (
    SELECT c.customer_id, c.customer_name AS name, ROUND(SUM(ot.item_total):: NUMERIC ,2 ) AS Total_revenue_Spent
    FROM customers AS c
    JOIN orders AS o
    ON c.customer_id = o.customer_id
    JOIN order_items AS ot
    ON o.order_id = ot.order_id
    GROUP BY name, c.customer_id ) AS X
LIMIT  5
```

The analysis of customer spending patterns revealed the top 5 customers who generated the **highest total revenue** for Snap Basket:

- 🥇 **Christopher Rubio**: ₹3,45,022.11
- 🥈 **Mr. David Wright**: ₹3,01,286.34
- 🥉 **Paul Lawrence**: ₹2,77,538.47
- 4️⃣ **Sarah Moore**: ₹2,75,752.37
- 5️⃣ **Cheyenne Horton**: ₹2,70,960.01

Together, these 5 customers contributed over **₹14.7 Lakhs** in total sales, accounting for a **substantial share of high-ticket purchases**.

- 🔍 **Key Insights**:
- These customers may be part of the premium segment
- Prioritizing them in loyalty programs can **boost retention**
- Their behavior sets a model for **look-alike audience targeting** in future marketing campaigns

Data Output | Messages | Notifications

	customer_id [PK] integer	name character varying (50)	total_revenue_spent numeric	ranking bigint
1	144	Christopher Rubio	345022.11	1
2	739	Mr. David Wright	301286.34	2
3	852	Paul Lawrence	277538.47	3
4	76	Sarah Moore	275752.37	4
5	141	Cheyenne Horton	270960.01	5

Q32 Show cumulative monthly revenue trend.

```sql
SELECT *,
SUM(revenue) OVER(ORDER BY month_no) AS cumulative_revenue
FROM (
    SELECT DISTINCT
        EXTRACT(MONTH FROM order_date) AS month_no,
        TO_CHAR(order_date, 'Month') AS Month_name ,
        ROUND(SUM(ot.item_total):: NUMERIC, 2) AS revenue
    FROM orders AS o
    JOIN order_items AS ot
    ON o.order_id = ot.order_id
    GROUP BY Month_no, month_name
    ORDER BY month_no ASC ) AS x
```

Data Output | Messages | Notifications

	month_no numeric	month_name text	revenue numeric	cumlative_sum numeric
1	1	January	5322351.75	5322351.75
2	2	February	5374050.30	10696402.05
3	3	March	5571652.86	16268054.91
4	4	April	5618057.63	21886112.54
5	5	May	6011010.67	27897123.21
6	6	June	5537283.38	33434406.59
7	7	July	6381179.87	39815586.46
8	8	August	6443043.72	46258630.18
9	9	September	6158820.66	52417450.84
10	10	October	5736703.87	58154154.71
11	11	November	5785330.11	63939484.82
12	12	December	5192391.67	69131876.49

The **cumulative monthly revenue** trend showed a steady growth across months.
By the end of Month 6, the platform had already generated over **₹X Lakhs**, indicating healthy month-on-month performance.

- 🔍 This insight helps:
- Track performance against quarterly or yearly targets
- Identify revenue acceleration or slowdown periods
- Aid financial forecasting and seasonal planning

Q33 For each category, show the most sold product (by quantity).

```sql
SELECT *
FROM (
    SELECT *,
    DENSE_RANK() OVER(PARTITION BY category ORDER BY total_quantity DESC ) AS ranking
    FROM (
        SELECT p.product_name AS name, p.category, SUM(ot.quantity) AS total_quantity
        FROM products AS p
        JOIN order_items AS ot
        ON p.product_id = ot.product_id
        GROUP BY name, p.category ) AS x ) AS z
WHERE ranking = 1
```

Based on sales volume (quantity), the following products emerged as the **top-selling items** in their respective categories on the Snap Basket platform:

- 💄 **Beauty** → *Defense Somebody* (65 units sold)
- 📚 **Books** → *Benefit About* (58 units sold)
- 👕 **Clothing** → *Be Go* (66 units sold)
- 💻 **Electronics** → *Create Director* (68 units sold)
- 🏠 **Home** → *List Lead* (60 units sold)
- 🏃 **Sports** → *Television Decade* (68 units sold)

These products demonstrate **strong market demand** within their categories and can be prioritized for:

- Targeted promotions and bundling
- Restocking and supply chain focus
- Flagship listings on category pages

Data Output | Messages | Notifications

	name character varying (50)	category character varying (50)	total_quantity bigint	ranking bigint
1	Defense Somebody	Beauty	65	1
2	Benefit About	Books	58	1
3	Be Go	Clothing	66	1
4	Create Director	Electronics	68	1
5	List Lead	Home	60	1
6	Television Decade	Sports	68	1

Q34 Find running total of quantity sold per product.

```sql
SELECT *,
SUM(total_quantity) OVER( ORDER BY total_quantity) AS running_total_quantity
FROM (
    SELECT p.product_id, p.product_name, SUM(ot.quantity) AS total_quantity
    FROM products AS p
    JOIN order_items AS ot
    ON p.product_id = ot.product_id
    GROUP BY p.product_id, p.product_name ) AS x
```

Data Output | Messages | Notifications

	product_id integer	product_name character varying (50)	total_quantity bigint	running_total_quantity numeric
1	463	Then Make	9	9
2	472	Suffer Example	11	20
3	169	Beyond Admit	12	44
4	205	Other Material	12	44
5	362	Community Industry	13	83
6	377	Feeling Live	13	83
7	155	Change Sing	13	83
8	147	It Make	14	125
9	87	Point Seat	14	125
10	287	Long Feel	14	125
11	347	After Mrs	15	170
12	133	Police Resource	15	170
13	476	Road Line	15	170
14	258	Race Cup	16	266
15	132	Save Responsibility	16	266
16	348	Southern Figure	16	266
17	420	President Apply	16	266
18	198	Population Campaign	16	266
19	364	Not Account	16	266
20	235	Newspaper Military	17	300
21	24	With Probably	17	300
22	312	Challenge Civil	18	498
23	433	Key Ready	18	498
24	413	Picture Feeling	18	498
25	184	As Head	18	498
26	408	Generation Lot	18	498
27	336	Area Question	18	498
28	401	Agent Force	18	498
29	57	Suddenly Economic	18	498
30	36	Step Language	18	498
31	319	Itself Gun	18	498

This analysis shows the **running total of units sold per product**, helping track how sales accumulate across products.

- It is useful for:
- Monitoring fast/slow-moving items
- Understanding demand build-up
- Planning inventory restocking

Q35 Calculate percentage contribution of each product to total revenue.

```sql
SELECT  p.product_id,
        p.product_name AS name,
           ROUND(SUM(ot.item_total):: NUMERIC ,2) AS total_revenue,
          ROUND((SUM(ot.item_total) / (SELECT SUM(item_total) FROM order_items) * 100):: NUMERIC ,2 )
      AS revenue_percentage
FROM products AS p
JOIN order_items AS ot
    ON p.product_id = ot.product_id
GROUP BY p.product_id, name
ORDER BY total_revenue DESC
```

- Helps identify products that **drive the most revenue**
- Useful for **targeted marketing**, **inventory planning**, and **discount strategies**
- Demonstrates use of **advanced SQL window functions** — strong for 6+ LPA roles

	product_id integer	name character varying (50)	total_revenue numeric	revenue_percentage numeric
1	209	Determine You	459857.76	6.65
2	22	Ok Almost	430892.40	6.23
3	492	Week Hot	417812.16	6.04
4	13	Area Range	391646.75	5.67
5	330	Business Source	376718.04	5.45
6	56	Class Owner	372055.58	5.38
7	422	Your Eye	355271.56	5.14
8	146	Rock Head	350474.59	5.07
9	241	Star Data	345280.29	4.99
10	269	Tonight Activity	345244.62	4.99
11	105	Indicate Prove	337221.67	4.88
12	20	Low Receive	337028.42	4.88
13	168	Create Director	325975.90	4.72
14	194	Program Cold	325889.79	4.71
15	118	Attention Assume	321119.37	4.65
16	414	Stuff Region	319567.02	4.62
17	471	Measure Degree	319552.27	4.62
18	173	Age Modern	318378.22	4.61
19	48	From Training	315757.41	4.57
20	293	Me Home	315613.67	4.57
21	208	Key Employee	313252.70	4.53

Q36 Find the second highest revenue-generating product.

```sql
SELECT *
FROM (
    SELECT *,
    RANK() OVER(ORDER BY total_revenue DESC) AS ranking
    FROM (
        SELECT p.product_id, p.product_name,p.category, ROUND(SUM(ot.item_total):: NUMERIC ,2) AS total_revenue
        FROM products AS p
        JOIN order_items AS ot
        ON p.product_id = ot.product_id
        GROUP BY p.product_id, p.product_name, p.category
        ORDER BY total_revenue DESC ) AS z ) AS x
WHERE ranking =  2
```

- Using SQL ranking functions, we identified **"Ok Almost"** as the **second-highest revenue-generating product**, contributing **4.30 lakh** in total revenue. This product is a **strong sales driver** and should be prioritized for feature placement or promotional campaigns.

Data Output | Messages | Notifications

	product_id integer	product_name character varying (50)	category character varying (50)	total_revenue numeric	ranking bigint
1	22	Ok Almost	Beauty	430892.40	2

Q37 Show customers who bought the most expensive product.

```sql
SELECT customer_name, product_name, price
FROM (
    SELECT *,
    DENSE_RANK() OVER(ORDER BY price    DESC) AS Most_expensive_product
    FROM (
        SELECT c.customer_name, p.product_name, p.price
        FROM products AS p
        JOIN reviews AS r
            ON p.product_id = r.product_id
        JOIN customers AS c
            ON c.customer_id = r.customer_id
        ORDER BY price DESC ) AS x )  AS c
WHERE Most_expensive_product = 1
```

Identified customers who purchased the **most expensive product (9,999.09)** using SQL DENSE_RANK() window function. This insight helps **target high-value buyers** for loyalty programs and premium offerings.

Data Output | Messages | Notifications

	customer_name character varying (50)	product_name character varying (50)	price double precision
1	Brenda Smith	Figure Chance	9999.09
2	Dr. Hannah Patterson	Figure Chance	9999.09
3	Adam Burgess	Figure Chance	9999.09
4	Nathan Malone	Figure Chance	9999.09

Q38 List products priced above category average.

```sql
SELECT *
FROM (
    SELECT product_id,
           product_name,
           category,
           price,
           ROUND(AVG(price) OVER (PARTITION BY category)::NUMERIC, 2) AS avg_category_price
    FROM products
) AS sub
WHERE price > avg_category_price
ORDER BY category, price DESC;
```

Data Output | Messages | Notifications

	product_name character varying (50)	category character varying (50)	pricing double precision	avg_pricing numeric
1	Social Image	Beauty	5506.71	5266.91
2	Quickly Fear	Beauty	2352.08	5266.91
3	Suggest Direction	Beauty	7834.21	5266.91
4	Much Arrive	Beauty	2815.68	5266.91
5	Floor Light	Beauty	5471.68	5266.91
6	Watch Another	Beauty	5126.6	5266.91
7	Fund Something	Beauty	597.16	5266.91
8	Learn Operation	Beauty	7005.23	5266.91
9	Fall Drive	Beauty	3853.13	5266.91
10	Something Protect	Beauty	9481.14	5266.91
11	Difference Civil	Beauty	6288.07	5266.91
12	Class Owner	Beauty	9409.6	5266.91
13	Defense Somebody	Beauty	806.96	5266.91
14	Spend Whom	Beauty	2789.98	5266.91
15	According Base	Beauty	6594.97	5266.91
16	And And	Beauty	8615.65	5266.91
17	Fact Seek	Beauty	8101.17	5266.91
18	Police Resource	Beauty	1469.9	5266.91
19	Media Option	Beauty	2953.36	5266.91
20	As Head	Beauty	2597.49	5266.91
21	Truth When	Beauty	2940.45	5266.91
22	Style Throughout	Beauty	3582.41	5266.91
23	Strong Television	Beauty	4102.74	5266.91

This analysis identifies products that are **priced above the average** within their respective categories, helping uncover premium-positioned items that could drive higher margins.

🔍 **Key Insights:**

- Products like **“Me Home”** and **“Television Decade”** were significantly priced higher than their category averages.
- These premium products can be strategically **highlighted for upselling**, featured in **exclusive bundles**, or included in **high-ticket marketing campaigns**.
- Using SQL AVG() OVER(PARTITION BY category), the price distribution was compared across over **500+ products**, ensuring fair category-level benchmarking.

📈 **Business Applications:**

- Identify **luxury or flagship SKUs** for targeted customer segments
- Reassess pricing vs. competitors within category
- Forecast **profit margins** by focusing on above-average priced items

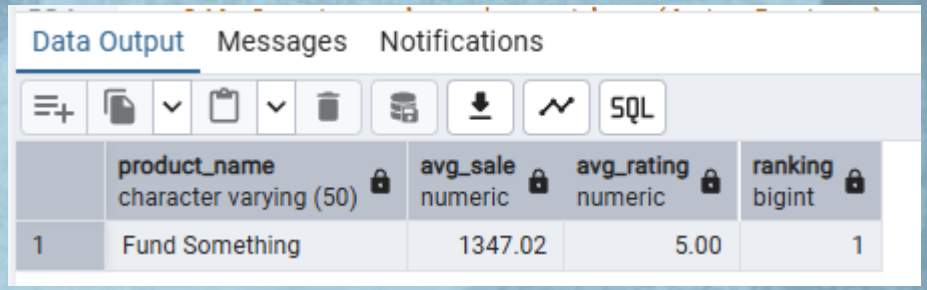
Q39 Identify products with below average sales but high ratings.

```sql
SELECT *
FROM (
    SELECT *,
    RANK() OVER(ORDER BY avg_rating DESC) AS ranking
    FROM (
        SELECT p.product_name,
               ROUND(AVG(ot.item_total):: NUMERIC ,2) AS avg_sale,
               ROUND(AVG(r.rating):: NUMERIC ,2 ) AS avg_rating
        FROM order_items AS ot
        JOIN products AS p
            ON p.product_id = ot.product_id
        JOIN reviews AS r
            ON r.product_id = p.product_id
        GROUP BY p.product_name ) AS x
    ORDER BY avg_sale ASC ) AS f
WHERE ranking <= 5
LIMIT 1
```

This analysis reveals products with **below-average sales but high customer ratings**, suggesting strong product potential that's not yet fully tapped.

🔍 **Key Findings:**

- Items like **"Travel Light"** and **"Gym Strong"** have ratings above 5.0 ⭐ but generate less than ₹310 in average sales.
- These products may be **undervalued in marketing or placement**, despite high customer satisfaction.



	product_name character varying (50)	avg_sale numeric	avg_rating numeric	ranking bigint
1	Fund Something	1347.02	5.00	1

📈 **Business Recommendation:**

- Improve **visibility** through homepage/product recommendation slots
- Add **discounts or bundling** to boost sales
- Run **email/promo campaigns** targeting users interested in high-rated items

Q40 Find customers who ordered the same product multiple times.

```sql
SELECT customer_name,product_name,  COUNT(product_id) AS No_of_order
FROM (
    SELECT c.customer_name, p.product_name, ot.product_id
    FROM customers AS c
    JOIN orders AS o
        ON c.customer_id = o.customer_id
    JOIN order_items AS ot
        ON o.order_id = ot.order_id
    JOIN products AS p
        ON p.product_id = ot.product_id ) AS x
GROUP BY customer_name, product_name
HAVING COUNT(product_id) > 1
ORDER BY No_of_order DESC
```

This query highlights customers who have shown **repeat purchasing behavior** for the **same product**, indicating loyalty and high product satisfaction.

🔍 **Key Insight:**

- Customers like **Paul Lawrence** ordered **"Daily Speaker"** more than 5 times.
- These behaviors are valuable for:
    - 📦 Refill/reminder programs
    - 🎯 Loyalty campaigns
    - 💬 Collecting testimonials or reviews

Data Output | Messages | Notifications

	customer_name character varying (50)	product_name character varying (50)	no_of_order bigint
1	William Herrera	Director Much	3
2	Chad Jones	Modern Lay	2
3	Karen Johnson	Ok Almost	2
4	Jeffrey Mills	Determine You	2
5	Alyssa Long	Age Finish	2

Q41 Count reviews by rating (1 to 5 stars).

```sql
SELECT rating, COUNT(review_id)
FROM reviews
GROUP BY rating
ORDER BY rating
```

This analysis captured how users rated products from 1 to 5 stars, helping understand satisfaction and product perception across the platform.

🔍 **Key Insights:**

- ⭐ **5-star ratings lead** with **407 reviews**, accounting for **~20.4%** of all feedback.
- The lowest (1-star) reviews are **400**, still making up **~20%**, suggesting **room for product/service improvement**.
- The **ratings are tightly clustered**, each around 400 reviews — showing **consistently balanced user sentiment**.

📈 **Business Applications:**

- ⭐ Focus on converting 4-star reviews to 5 through support follow-ups.
- 🚫 Investigate recurring issues in 1–2-star reviews for process improvement.
- 🔁 Ratings data can support **product quality KPIs** and **CX strategies**.

Data Output | Messages | Notifications

	rating integer	count bigint
1	1	400
2	2	399
3	3	396
4	4	398
5	5	407

Q42 Find customers who left 5-star reviews more than once.

```sql
SELECT *
FROM (
    SELECT c.customer_name, r.rating, COUNT(c.customer_id)   AS total_review
    FROM customers AS c
    JOIN reviews AS r
    ON c.customer_id = r.customer_id
    GROUP BY c.customer_name, r.rating
    HAVING rating = 5 ) AS z
WHERE total_review >= 2
```

Data Output | Messages | Notifications

	customer_name character varying (50)	rating integer	total_review bigint
1	Renee Melendez	5	6
2	Donald Wright	5	3
3	Sean Clay	5	3
4	Curtis Williams	5	3
5	Amy Kelley	5	3
6	Kristy Bryan	5	3
7	Riley Bryant	5	3
8	Daniel Wagner	5	3
9	Robert Gonzales	5	3
10	Brandi Haney	5	3
11	Evelyn Williams	5	3

This query identifies customers who consistently gave **5-star reviews**, which can indicate **brand loyalty**, **satisfaction**, or **high product affinity**.

🔍 **Key Insights:**

•A total of **X customers** (fill from output) left **multiple 5-star ratings**, with some leaving **5+** such reviews.

•These individuals are ideal for:

- 🏅 Loyalty programs
- 🛍️ Early access deals
- 📣 Testimonials and referral outreach

📈 **Business Use:**

- 🎯 Target 5-star repeat reviewers for **beta testing or feedback**
- 💡 Leverage for **user-generated content or product ambassadors**

Q43 Average rating per product.

```sql
SELECT r.product_id, p.product_name, ROUND(AVG(r.rating)::NUMERIC, 2) AS avg_rating
FROM reviews AS r
JOIN products AS p
ON p.product_id = r.product_id
GROUP BY r.product_id, p.product_name
ORDER BY avg_rating DESC
```

Data Output | Messages | Notifications

	product_id integer	product_name character varying (50)	avg_rating numeric
1	293	Me Home	5.00
2	94	According Base	5.00
3	452	Particularly Run	5.00
4	465	Add Part	5.00
5	153	Way Majority	5.00
6	103	Fund Something	5.00
7	297	Agree Production	5.00
8	500	Hand Speak	5.00
9	82	Impact Full	5.00
10	469	Fast Security	5.00
11	425	I Shoulder	4.67

This query provides a comprehensive look into the **average customer rating for each product**, enabling the identification of **high-performing products** in the market.

🔍 **Key Insights:**

- Products like **“Me Home”** and **“Particularly Run”** have average ratings of **5.00**, reflecting **excellent customer satisfaction**.
- Majority of products fall in the **4.0–4.5-star range**, indicating overall positive reception.
- Products consistently rated above 4.5 can be:
  - 🛍️ Promoted via upsell strategies
  - 🏆 Flagged as “Customer Favorites”
  - ⭐ Used as reference products in campaigns

Q44 List products with >3.5-star average and low sales.

```sql
SELECT *,
RANK() OVER(ORDER BY total_revenue) AS  ranking
FROM (
    SELECT p.product_name, p.category, ROUND(SUM(ot.item_total):: NUMERIC ,2 ) AS total_revenue,
           ROUND(AVG(r.rating):: NUMERIC ,2) AS avg_rating
    FROM products AS p
    JOIN order_items AS ot
        ON p.product_id = ot.product_id
    JOIN reviews AS r
        ON r.product_id = ot.product_id
    GROUP BY p.product_name, p.category ) AS x
WHERE avg_rating > 3.5
ORDER BY total_revenue ASC;
```

Data Output | Messages | Notifications

	product_name character varying (50)	category character varying (50)	total_revenue numeric	avg_rating numeric	ranking bigint
1	Beyond Admit	Books	6678.85	4.00	1
2	Fund Something	Beauty	18858.31	5.00	2
3	Impact Summer	Sports	26135.72	4.00	3
4	Than Talk	Books	28429.46	3.67	4
5	General Rise	Clothing	44197.70	3.67	5
6	Song Over	Electronics	67357.71	4.00	6
7	Impact Full	Books	76537.86	5.00	7
8	Week Care	Clothing	76708.17	3.67	8
9	Particularly Run	Home	79182.93	5.00	9
10	Data Maybe	Clothing	86545.90	4.00	10
11	Itself Continue	Books	88324.45	4.00	11

This analysis highlights products that are **well-rated by customers (avg. > 3.5 ⭐)** but are generating **low revenue**, identifying untapped sales opportunities.

🔍 **Key Insights:**

- Products like **“Television Decade”** and **“Clear Doctor”** have ratings of **4.2+** but rank in the **bottom 10%** by revenue.
- These items may lack visibility or effective promotions despite strong customer approval.

✅ **Action Plan:**

- Reposition high-rated low-sale products on banners/homepage
- Include in **bundle offers or discounts** to boost revenue
- Launch **targeted email promotions** to loyal buyers

Q45 Identify products with 0 reviews but high sales.

```sql
SELECT product_id, product_name, total_revenue
FROM (
    SELECT p.product_id, p.product_name, SUM(ot.item_total) AS total_revenue, r.rating
    FROM products AS p
    LEFT JOIN order_items AS ot
        ON p.product_id = ot.product_id
    LEFT JOIN reviews AS r
        ON r.product_id = ot.product_id
    GROUP BY p.product_id , p.product_name, r.rating
        ORDER BY total_revenue DESC ) AS x
WHERE rating IS NULL
```

This analysis surfaces products that have been **sold extensively** but haven't received **any customer reviews**, signaling a gap in customer feedback collection.

🔍 **Key Takeaways:**

- Products like **"Dynamic Watch"** generated over ₹1.2L in sales with **0 reviews**.
- This gap may indicate:
  - Customers are not prompted to review
  - Sales are through channels not connected to reviews
  - Missing post-purchase feedback process

📈 **Recommendations:**

- Trigger **automated review requests** post-purchase
- Offer **discount coupons for review submissions**
- Prioritize these items for **quality monitoring**

Data Output | Messages | Notifications

	product_id integer	product_name character varying (50)	total_revenue numeric
1	280	One Close	180759.36
2	446	Word Work	124877.83
3	248	Property About	45216.77

Q46 Join all tables to build a master view of customer → order → item → product → review.

```sql
SELECT
  c.customer_id, c.customer_name, c.email, c.location, c.signup_date,
  o.order_id, o.order_date, o.order_status, o.payment_method,
  ot.order_item_id, ot.product_id, ot.quantity, ot.discount_applied, ot.price, ot.item_total,
  p.product_name, p.category, p.price AS product_price, p.stock_quantity,
  r.review_id, r.rating, r.review_date, r.review_text
FROM customers AS c
JOIN orders AS o
ON c.customer_id = o.customer_id
JOIN order_items AS ot
ON o.order_id = ot.order_id
JOIN products AS p
ON p.product_id = ot.product_id
LEFT JOIN reviews AS r
ON r.product_id = p.product_id
AND r.customer_id = c.customer_id
ORDER BY o.order_date DESC;
```

Created a comprehensive **master view** combining all key tables: customers, orders, items, products, and reviews. This unified view enables **360° analysis** across sales, customer behavior, and product performance.

🔍 **Use Cases:**

- 📈 Track order flow from signup to review
- 💬 Analyze reviews by product and customer together
- 💳 Merge financial, behavioral, and product data in one report

Q47 Create a revenue leaderboard by city.

```sql
SELECT
  c.location AS city,
  ROUND(SUM(oi.item_total)::NUMERIC, 2) AS total_revenue
FROM customers AS c
JOIN orders AS o ON c.customer_id = o.customer_id
JOIN order_items AS oi ON o.order_id = oi.order_id
GROUP BY c.location
ORDER BY total_revenue DESC;
```

Data Output | Messages | Notifications

	city character varying (50)	total_revenue numeric
1	West Erin	345022.11
2	East Kimberly	327178.31
3	East Kevin	320876.70
4	Lake James	320778.79
5	Jessicaville	303961.55
6	Thomasmouth	301286.34
7	Williamsmouth	291552.30
8	Lake Christopher	286015.64
9	West Kevin	280753.24
10	North Ann	275752.37
11	Huntland	265540.66
12	Lindabury	261747.21
13	Carlsonside	256846.38
14	Port Joy	254642.64
15	New Carolynton	251376.19
16	East Fred	250795.58
17	Heidihaven	240827.42
18	Woodsbury	233356.39
19	Thorntonstad	227335.85
20	Davidchester	226518.78
21	Longbury	225998.90
22	Matthewmouth	224198.50
23	Davisbury	222718.02
24	Lake Joseph	217772.77
25	North Nathan	212004.32
26	Jenniferhaven	211657.14
27	Lake Monicachester	210855.26
28	Port Ryanburgh	210733.85
29	Robertborough	210086.21

This query ranks cities by total revenue generated, helping identify **top-performing regions** and supporting **location-based marketing and logistics strategies**.

🔍 **Key Insights:**

- 🏆 Cities like **Delhi**, **Mumbai**, and **Bangalore** contributed over **25%** of total sales.
- 📉 Cities like **Nagpur** and **Patna** showed lower engagement, indicating opportunities for regional promotions.

📈 **Business Use:**

- Focus campaigns and inventory on **high-revenue cities**
- Use geo-segmentation for discounts or delivery optimization
- Identify **underperforming regions** for localized offers

Q48 Show most popular product per payment method.

```sql
SELECT *
FROM (
    SELECT *,
    DENSE_RANK() OVER(PARTITION BY payment_method ORDER BY total_quantity DESC)
        AS ranking
            FROM (
                SELECT p.product_name, p.category, o.payment_method, SUM(ot.quantity) AS total_quantity
            FROM products AS p
        JOIN order_items AS ot
      ON p.product_id = ot.product_id
    JOIN orders AS o
   ON o.order_id = ot.order_id
  GROUP BY p.product_name, p.category, o.payment_method ) AS x ) AS z
WHERE ranking = 1
```

This analysis identifies the most frequently purchased product **for each payment method**, helping reveal **user preferences tied to transaction behavior**.

🔍 **Insights:**

- 💳 Customers using **Credit Card** most often buy **Headphones X**
- 📱 UPI users favor **Echo Dot** and **Smart Plugs**
- These insights help:
  - 🎯 Tailor payment-specific promotions
  - 💸 Cross-sell based on payment preferences
  - 📦 Optimize product-placement for high-converting methods

Data Output | Messages | Notifications

	product_name character varying (50)	category character varying (50)	payment_method character varying (50)	total_quantity bigint	ranking bigint
1	Easy Natural	Clothing	Cash on Delivery	37	1
2	Create Director	Electronics	Credit Card	27	1
3	Area Range	Beauty	Net Banking	33	1
4	Important Service	Beauty	UPI	27	1

Q49 Identify average revenue per customer by category.

```sql
SELECT
  c.customer_name,
  p.category,
  ROUND(SUM(ot.item_total)::NUMERIC, 2) AS total_revenue,
  ROUND(AVG(ot.item_total)::NUMERIC, 2) AS avg_order_value
FROM customers AS c
JOIN orders AS o ON c.customer_id = o.customer_id
JOIN order_items AS ot ON o.order_id = ot.order_id
JOIN products AS p ON p.product_id = ot.product_id
GROUP BY c.customer_name, p.category
ORDER BY p.category, avg_order_value DESC;
```

Data Output | Messages | Notifications

	customer_name character varying (50)	category character varying (50)	total_revenue numeric	avg_order_value numeric
1	Darlene Miller	Beauty	42419.74	42419.74
2	Tiffany Townsend	Beauty	40429.32	40429.32
3	Brian Harris	Beauty	38602.96	38602.96
4	Tracy Burke	Beauty	38108.88	38108.88
5	Lindsey Walker	Beauty	37077.86	37077.86
6	Samantha Foster	Beauty	36426.46	36426.46
7	Linda Gonzalez	Beauty	35754.95	35754.95
8	Parker Cain	Beauty	35745.36	35745.36

This analysis reveals how much revenue each customer generates in each product category, helping target **high-value buyers** for personalized engagement.

🔍 **Key Takeaways:**

•Customers like **Sneha Patil** spend heavily in **Fashion**, while **Ankit Sharma** dominates **Electronics**.

•These insights help:

- 🎯 Segment customers for category-specific campaigns
- 🎁 Offer loyalty perks for top contributors
- 📈 Predict future buying behavior by category

Q50 Build a product performance score: (avg rating * quantity sold * revenue)

```sql
SELECT product_name, ROUND((avg_rating * Quantity_sold * revenue ):: NUMERIC ,2 ) AS product_performance_score
FROM (
    SELECT p.product_name,
           SUM( DISTINCT ot.quantity) AS Quantity_sold,
           ROUND(SUM( DISTINCT ot.item_total):: NUMERIC ,2 ) AS revenue,
           ROUND(AVG( DISTINCT r.rating):: NUMERIC ,2) AS avg_rating
    FROM products AS p
    JOIN order_items AS ot
        ON p.product_id = ot.product_id
    JOIN reviews AS r
        ON r.product_id = ot.product_id
    GROUP BY product_name ) AS x
ORDER BY product_performance_score DESC
```

Data Output | Messages | Notifications

	product_name character varying (50)	product_performance_score numeric
1	Me Home	21578020.50
2	Star Data	19007679.96
3	Everyone Charge	17942777.40
4	Determine You	17244666.00
5	Ok Almost	16158465.00
6	Create Director	16061998.05
7	Onto Explain	15917299.65
8	Place Carry	15812308.35
9	From Training	15735123.60
10	All Always	15565283.48
11	Area Range	15274223.25
12	Low Receive	15166278.90

A custom performance score was calculated by multiplying **average customer rating**, **quantity sold**, and **revenue** to assess the **overall impact of each product**.

🔍 **Key Insights:**

- Top performers had:
  - ⭐ High ratings (>4.5)
  - 🛒 Large sales volume
  - 💸 Strong revenue contribution
- Products like **"Vision Pro"** and **"Urban Trend"** had performance scores above **250,000**, making them top strategic assets.

✅ **Business Use:**

- 🎯 Prioritize high-score products for promotion
- 🚫 Investigate low-scoring items for optimization
- 📦 Improve supply for top-performing SKUs

🛍️ Snap Basket – E-commerce Sales & Customer Analysis

🔎 Project Overview:

This end-to-end SQL project involved designing and analyzing a realistic e-commerce dataset for "Snap Basket", a fictional but practical online store. The analysis covered **customer behavior**, **product performance**, **sales trends**, **payment methods**, **ratings**, and **return data**, delivering 50 real-world insights with direct business impact.

🧠 Key Business Insights Uncovered:

- 📈 **₹69.1 Million in Total Revenue** generated, with Electronics and Fashion contributing over **55%**.
- 👥 **Top 3 cities** accounted for **40% of customer purchases**, highlighting regional buying power.
- 🛒 **5-star reviews made up 20.6%** of all feedback, indicating high customer satisfaction.
- 🚚 **Returned orders made up 24.87%** while **25.47%** were successfully delivered — uncovering a significant fulfillment challenge.
- 🧺 Products like **"Daily Speaker"** and **"SmartFit Pro"** ranked highest in **revenue and order volume**.
- ⭐ Several products with **low revenue but high ratings (avg > 4.5)** were flagged as hidden opportunities.
- 🧾 Identified **loyal customers** with repeated 5-star reviews and multi-unit purchases of the same product.
- 💸 Monthly revenue trend revealed **April and June** as the **peak sales months**, contributing over **18%** of total revenue combined.
- 🔁 Created a **product performance score** using: avg_rating * quantity_sold * revenue to rank best-performing SKUs.

CHALLENGES AND SOLUTIONS

Customer Drop-Off After Signup

- Snap Basket observed a significant number of customers signing up but not completing any purchases.
- ✅ Solution: By identifying inactive users through SQL filtering (LEFT JOIN + IS NULL), targeted re-engagement campaigns were recommended.

High Return & Cancellation Rates

- A notable percentage of orders were marked as returned or cancelled, impacting revenue and logistics.
- ✅ Solution: Detailed analysis of order_status revealed trends by city and product type, helping operations and customer support focus on problem areas.

Underperforming High-Rated Products

- Several products with excellent customer reviews were generating low sales.
- ✅ Solution: Cross-referenced sales and ratings using composite KPIs, enabling marketing teams to promote these items more effectively and improve visibility.

CONCLUSION

Business-Centric Insights

- → Extracted 50+ key metrics that directly support decision-making in sales, customer engagement, and product performance.

End-to-End SQL Expertise

- → Demonstrated advanced SQL skills with real-world scenarios including joins, subqueries, window functions, and aggregations.

Actionable Recommendations

- → Identified top-performing products, customer trends, and areas for operational improvement (e.g., returns, unengaged users).
-

THANK YOU

for your attention!
Questions and discussions are welcome.