

Title: Snapdeal Product and Regional Performance Analysis

Subtitle: An Analytical Study to Identify High-Performing Products, Categories, and Customer Segments for Strategic Decision-Making

by: Pratik Garud

Date: 30/10/2025

--Questions:

--1. Best-Performing Product Categories

--Snapdeal wants to focus on product categories that consistently exceed their sales targets to

--guide future marketing and inventory decisions. To analyse this, you will need to compare the total revenue

--generated by each category to the target for that category in the month of April 2018.

-- Expected Output Columns:

-- Category: The name of the product category.

-- Total Sales: The total revenue generated by that category in April 2018.

-- Sales Target: The target revenue for that category in April 2018.

```
SELECT d.category, sum(d.amount * d.quantity) as Total_Sales , sum(t.target) as
Sales_Target
FROM order_details d
LEFT JOIN sales_target t
ON d.category = t.category
WHERE t.month_of_order = 'Apr-18'
GROUP BY d.category
ORDER BY Total_Sales DESC;
```

	category character varying	total_sales numeric	sales_target numeric
1	Electronics	816583	2772000
2	Furniture	665765	2527200
3	Clothing	664522	11388000

Explanation:

- This query identifies which product **categories performed best in April 2018**.
- It joins the order_details and sales_target tables on the **category** column.
- Calculates **total revenue** for each category using amount × quantity.
- Compares this total with the **sales target** from the sales_target table.
- The result helps Snapdeal identify **categories exceeding their targets** and prioritize them for marketing and stocking.

Key Insight:

Shows which categories surpassed or fell short of their sales targets in April 2018.

-- 2. Customer Purchase Behavior by Location

-- Snapdeal wants to understand how customer purchasing behavior varies by location, particularly which states contribute

--the most orders. Write an SQL query to identify the top 3 states with the highest number of orders.

-- Expected Output Columns:

-- State: The state where the orders originated.

-- Total Orders: The total number of orders placed from that state.

```
SELECT state,count(order_id) AS Total_Orders FROM order_list
GROUP BY state
ORDER BY Total_Orders DESC
LIMIT 3;
```

	state character varying 🔒	total_orders bigint 🔒
1	Madhya Pradesh	101
2	Maharashtra	90
3	Rajasthan	32

Explanation:

- This query analyzes **customer purchasing behavior by location**.
- It counts the total number of **orders per state** using the order_list table.
- Then ranks states by total orders and selects the **top 3 states** with the most orders.

Key Insight:

Helps Snapdeal focus marketing efforts and logistics in **high-demand states**.

-- 3. High Revenue, Low Profit Products

-- Snapdeal is interested in identifying products that generate significant revenue but contribute little to overall profit.

-- These products might require pricing adjustments or cost reductions. To help with this, you will need to categorize

-- products based on their profit margins and identify those with high revenue but low profitability.

-- Write an SQL query to identify products that generate more than \$10,000 in revenue but have a

-- profit margin of less than 5% and classify products as having either a "Low Profit Margin" or a "High Profit Margin"
-- based on their total revenue and total profit.

-- Expected Output Columns:
-- Category: The product category.
-- Sub-Category: The product sub-category.
-- Total Revenue: The total revenue generated by that product.
-- Total Profit: The total profit generated by that product.
-- Profit Margin Category: Categorizes products as "Low Profit Margin" or "High Profit Margin" based on the profit margin.

```
WITH product_summary AS (  
    SELECT  
        category,  
        sub_category,  
        sum(amount * quantity) AS total_revenue,  
        SUM(profit) AS total_profit,  
        CASE  
            WHEN SUM(amount) = 0 THEN 0  
            ELSE SUM(profit) / SUM(amount)  
        END AS profit_margin  
    FROM order_details  
    GROUP BY category, sub_category  
)  
SELECT  
    category,  
    sub_category,  
    total_revenue,  
    total_profit,  
    ROUND(profit_margin * 100, 2) AS profit_margin_pct,  
    CASE  
        WHEN profit_margin < 0.05 THEN 'Low Profit Margin'  
        ELSE 'High Profit Margin'  
    END AS profit_margin_category  
FROM product_summary  
WHERE total_revenue > 10000  
ORDER BY total_revenue DESC;
```

	category character varying	sub_category character varying	total_revenue numeric	total_profit numeric	profit_margin_pct numeric	profit_margin_category text
1	Electronics	Printers	307963	5964	10.24	High Profit Margin
2	Furniture	Bookcases	295598	4888	8.60	High Profit Margin
3	Clothing	Saree	263523	352	0.66	Low Profit Margin
4	Furniture	Chairs	206479	577	1.69	Low Profit Margin
5	Electronics	Electronic Games	204850	-1236	-3.16	Low Profit Margin
6	Electronics	Phones	200893	2207	4.79	Low Profit Margin
7	Clothing	Trousers	124640	2847	9.48	High Profit Margin
8	Electronics	Accessories	102877	3559	16.38	High Profit Margin
9	Furniture	Tables	90706	-4011	-17.74	Low Profit Margin
10	Clothing	Stole	86155	2559	13.80	High Profit Margin
11	Clothing	Hankerchief	75518	2098	14.36	High Profit Margin
12	Furniture	Furnishings	72982	844	6.26	High Profit Margin
13	Clothing	T-shirt	41396	1500	20.32	High Profit Margin
14	Clothing	Shirt	39373	1131	14.97	High Profit Margin
15	Clothing	Kurti	14643	181	5.39	High Profit Margin
16	Clothing	Skirt	10213	235	12.08	High Profit Margin

Explanation:

- First, it calculates **total revenue**, **total profit**, and **profit margin** for each product (sub-category).
- Then filters out only those products where:
 - Revenue > \$10,000
 - Profit margin < 5% (classified as *Low Profit Margin*)
- Finally, it labels products as **Low** or **High Profit Margin**.

Key Insight:

Highlights **high-revenue but low-profit** products that may need **cost or pricing optimization**.

-- 4. Products with Highest Profit per Sale

-- Snapdeal is looking to identify the products with the highest profit per unit sold. This will help

-- inform decisions about which products to promote or prioritize in inventory. Write an SQL query to

-- calculate the profit per unit for each product.

-- Expected Output Columns:

-- Category: The product category.

-- Sub-Category: The product sub-category.

-- Total Profit: The total profit generated by that product.

-- Total Quantity: The total number of units sold.

-- Profit per Unit: The calculated profit per unit sold.

```
select category,sub_category, sum(profit) as total_profit,sum(quantity) as total_quantity
,
round(sum(profit) / sum(quantity) ,2) as profit_per_unit
```

FROM order_details

GROUP BY category, sub_category

	category character varying	sub_category character varying	total_profit numeric	total_quantity bigint	profit_per_unit numeric
1	Clothing	Trousers	2847	135	21.09
2	Electronics	Printers	5964	291	20.49
3	Furniture	Bookcases	4888	297	16.46
4	Electronics	Accessories	3559	262	13.58
5	Electronics	Phones	2207	304	7.26
6	Clothing	T-shirt	1500	305	4.92
7	Clothing	Shirt	1131	271	4.17
8	Clothing	Stole	2559	671	3.81
9	Clothing	Hankerchief	2098	754	2.78
10	Furniture	Furnishings	844	310	2.72
11	Furniture	Chairs	577	277	2.08
12	Clothing	Leggings	260	186	1.40
13	Clothing	Kurti	181	164	1.10
14	Clothing	Skirt	235	248	0.95
15	Clothing	Saree	352	782	0.45
16	Electronics	Electronic Games	-1236	297	-4.16
17	Furniture	Tables	-4011	61	-65.75

Explanation:

- Calculates **profit per unit sold** for each product (sub-category).
- It sums total profit and quantity, then divides them to find **profit per unit**.
- Helps identify which products generate the **most profit per sale**, not just total revenue.

Key Insight:

Guides Snapdeal on which products to **prioritize for promotions and inventory restock** due to high per-unit profitability.

-- 5. Set Operations for Category Comparison

-- Snapdeal wants to explore cross-sell opportunities by identifying customers who have purchased

-- from multiple product categories. This will help them understand which customers are more likely

-- to buy products from different categories. Write an SQL query to identify customers who purchased from multiple categories.

-- Expected Output Columns:

-- CustomerName: The name of the customer.

-- Unique Categories: The number of different categories the customer has purchased from.

```
SELECT ol.customer_name,count(distinct od.category) AS unique_category
```

```

FROM order_list ol
JOIN order_details od
ON od.order_id = ol.order_id
GROUP BY ol.customer_name
ORDER BY unique_category DESC;

```

	customer_name character varying	unique_category bigint
1	Vini	3
2	Oshin	3
3	Parishi	3
4	Parth	3
5	Pinky	3
6	Diwakar	3
7	Pooja	3
8	Ekta	3
9	Farah	3
10	Pranav	3

	customer_name character varying	unique_category bigint
323	Teena	1
324	Tejas	1
325	Turumella	1
326	Utkarsh	1
327	Utsav	1
328	Vaibhavi	1
329	Vineet	1
330	Vipul	1
331	Vivek	1
332	Yohann	1

Explanation:

- Identifies customers who purchase from **multiple product categories**.
- Uses a join between order_list and order_details to link customers with their purchases.
- Counts distinct categories per customer to find **cross-category buyers**.

Key Insight:

Reveals potential **cross-sell and upsell opportunities** among customers who buy from various categories.

-- 6. Product Revenue Comparison

-- Snapdeal wants to identify products that consistently outperform other products in the same category
 -- in terms of revenue. This will help guide future acquisition and promotional strategies.
 -- Write an SQL query to identify products whose total sales are higher than the average sales for other products in the same category.

-- Expected Output Columns:

-- Category: The product category.
 -- Sub-Category: The product sub-category.
 -- Total Revenue: The total revenue generated by that product.
 -- Q6 Corrected: Products (sub-categories) with revenue > average revenue in same category

```

WITH category_totals AS (
  SELECT
    category,
    sub_category,
    SUM(amount * quantity) AS total_revenue
  FROM order_details
  GROUP BY category, sub_category
),
category_avg AS (
  SELECT
    category,
    AVG(total_revenue) AS avg_revenue_in_category
  FROM category_totals
  GROUP BY category
)
SELECT
  ct.category AS Category,
  ct.sub_category AS "Sub-Category",
  ct.total_revenue AS "Total Revenue"
FROM category_totals ct
JOIN category_avg ca
  ON ct.category = ca.category
WHERE ct.total_revenue > ca.avg_revenue_in_category
ORDER BY ct.category, ct.total_revenue DESC;

```

	category character varying 🔒	Sub-Category character varying 🔒	Total Revenue numeric 🔒
1	Clothing	Saree	263523
2	Clothing	Trousers	124640
3	Clothing	Stole	86155
4	Clothing	Hankerchief	75518
5	Electronics	Printers	307963
6	Electronics	Electronic Games	204850
7	Furniture	Bookcases	295598
8	Furniture	Chairs	206479

Explanation:

- Calculates **total revenue per sub-category**, then finds the **average revenue per category**.
- Compares each product's revenue against the average in its category.
- Returns only those sub-categories that **outperform the category average**.

Key Insight:

Identifies **top-performing products within each category** — ideal for **promotions, partnerships, or restocking decisions**.