DAY 7 – SQL CASE STUDIES & BUSINESS ANALYSIS

Dataset: Superstore Sales Dataset
Tool Used: MySQL 8.0
Analysis Type: SQL Case Studies

1.Objective of Day 7

The objective of Day 7 was to solve real-world business problems using SQL by applying aggregation, joins, window functions, and conditional logic. These case studies simulate SQL interview questions and real analytics tasks.

2.Case Study Summary

Case Study 1: Top Profit-Generating Products

- Identified top sub-categories contributing to profit
- Used window functions for ranking

```
WITH profit_data AS (
   SELECT
     category,
     sub_category,
     ROUND(SUM(profit),2) AS total_profit
   FROM orders
   GROUP BY category, sub_category
)
SELECT
  category,
  sub_category,
  total_profit,
  DENSE_RANK() OVER (
     PARTITION BY category
     ORDER BY total_profit DESC
  ) AS rank_in_category
FROM profit_data
WHERE total_profit > 0
```

	category	sub_category	total_profit	rank_in_category
▶	Furniture	Chairs	26590.15	1
	Furniture	Furnishings	13059.25	2
	Office Supplies	Paper	34053.34	1
	Office Supplies	Binders	30221.64	2
	Office Supplies	Storage	21279.05	3
	Office Supplies	Appliances	18138.07	4
	Office Supplies	Envelopes	6964.10	5
	Office Supplies	Art	6527.96	6
	Office Supplies	Labels	5546.18	7
	Office Supplies	Fasteners	949.53	8
	Technology	Copiers	55617.90	1
	Technology	Phones	44516.25	2
	Technology	Accessories	41936.78	3
	Technology	Machines	3384.73	4

```
ORDER BY category, rank_in_category;
```

Case Study 2: Loss-Making Categories

- Detected categories with negative profit
- Highlighted need for cost control

```
SELECT
  category,
  ROUND(SUM(sales),2) AS total_sales,
  ROUND(SUM(profit),2) AS total_profit
FROM orders
GROUP BY category
HAVING SUM(profit) < 0;
```

	category	total_sales	total_profit

Case Study 3: Year-Over-Year Growth Analysis

- Measured annual business growth
- Identified decline periods

	year	total_sales	yoy_growth
▶	2014	484247.56	NULL
	2015	470532.46	-13715.10
	2016	609205.86	138673.40
	2017	733215.19	124009.33

```
WITH yearly_sales AS (
    SELECT
      YEAR(order_date) AS year,
      ROUND(SUM(sales),2) AS total_sales
    FROM orders
    GROUP BY YEAR(order_date)
)
SELECT
  year,
  total_sales,
  total_sales - LAG(total_sales) OVER (ORDER BY year) AS yoy_growth
FROM yearly_sales;
```

Case Study 4: Product Segmentation

- Segmented products using NTILE
- Identified top and bottom performance tiers

	sub_category	total_profit	profit_quartile
▶	Copiers	55617.90	1
	Phones	44516.25	1
	Accessories	41936.78	1
	Paper	34053.34	1
	Binders	30221.64	1
	Chairs	26590.15	2
	Storage	21279.05	2
	Appliances	18138.07	2
	Furnishings	13059.25	2
	Envelopes	6964.10	3
	Art	6527.96	3
	Labels	5546.18	3
	Machines	3384.73	3
	Fasteners	949.53	4
	Supplies	-1188.99	4
	Bookcases	-3472.56	4
	Tables	-17725.59	4

```
SELECT
  sub_category,
  ROUND(SUM(profit),2) AS total_profit,
  NTILE(4) OVER (ORDER BY SUM(profit) DESC) AS profit_quartile
FROM orders
GROUP BY sub_category;
```

Case Study 5: Order Profitability Classification

- Classified orders using CASE
- Measured operational efficiency

	order_type	total_orders
▶	Profitable	8058
	Loss	1936

```
SELECT
  CASE
    WHEN profit > 0 THEN 'Profitable'
    ELSE 'Loss'
  END AS order_type,
  COUNT(*) AS total_orders
FROM orders
GROUP BY order_type;
```

Key Insights from Day 7

- A small number of products drive most profits
- Sales growth must be evaluated alongside profitability
- Window functions simplify complex analysis
- SQL can directly answer business questions

Business Recommendations

- Focus on high-profit product segments
- Review pricing strategies for loss-making categories
- Track year-over-year performance regularly
- Use segmentation for targeted decision-making

Key Learnings from Day 7

- Translating business problems into SQL queries
- Applying advanced SQL for analytics
- Writing interview-ready SQL solutions
- Thinking like a professional data analyst