Advanced Sales Performance & Operational Insights on a Retail Superstore

By Satyam Sinha

Table of Content

Executive Summary

This project, titled **"Advanced Sales Performance & Operational Insights on a Retail Superstore,"** demonstrates how raw retail data can be transformed into business intelligence using **PostgreSQL and Power BI**.

The goal was to uncover why profits were dropping despite good sales, and what operational or product-level inefficiencies were responsible. Starting with a flat CSV dataset, we cleaned and normalized the data using SQL and split it into relational tables — orders, customers, products, and sales_facts.

We applied **advanced SQL techniques** like:

- Joins, subqueries, CTEs, and window functions
- CASE statements for bucketing discounts and classifying margins
- Statistical aggregation and hypothesis testing (e.g., "Do higher discounts increase profit?")

The insights were then visualized in a 2-page interactive Power BI dashboard, showing:

- High-selling but loss-making products (Tables, Bookcases)
- Profit drops beyond 20% discount
- Negative margins in certain cities
- Customer Lifetime Value (CLV) estimates
- Shipping delay impact analysis

Introduction

In a highly competitive retail environment, understanding where profits are coming from—and where they're leaking—is essential. This project focuses on analyzing the sales and operational data of a retail superstore to uncover patterns, losses, and opportunities for improvement. By closely examining product categories, discount strategies, customer segments, and shipping behavior, the aim is to derive insights that support smarter business decisions. The analysis goes beyond basic reporting, focusing on real performance drivers hidden within the data.

❗ Problem Statement

Despite generating high sales volumes, the retail superstore is experiencing inconsistent profit margins across regions, product categories, and customer segments. The business lacks visibility into which areas are driving profits and which are silently causing losses. Without clear insights, decisions related to discounting, shipping, and customer targeting are being made blindly — leading to operational inefficiencies and missed growth opportunities.

Objectives

- To identify which product categories and sub-categories are contributing the most to profits or losses.
- To analyze the impact of discounting on sales and profit margins.
- To examine shipping delays and their relation to profitability.
- To evaluate performance across regions, states, and cities to detect hidden loss areas.
- To estimate Customer Lifetime Value (CLV) using sales and profit data.
- To build an interactive dashboard for business users to explore key performance metrics and make data-driven decisions.

Methodology

The project followed a structured data analytics workflow to simulate how real-world business problems are solved using data. The steps were:

1. **Data Acquisition & Understanding**
   The dataset, containing customer orders, sales, product, and shipping details, was studied to identify key fields, data types, and relationships.
2. **Data Cleaning & Preparation**
   Using SQL, the data was cleaned by removing duplicates, fixing date inconsistencies, and handling null values. New columns such as shipping delays, year, and discount buckets were also created to support analysis.
3. **Data Modeling**
   The flat dataset was split into normalized relational tables — `orders`, `customers`, `products`, and `sales_facts` — to follow a star schema and ensure analytical efficiency.
4. **Exploratory Analysis & Hypothesis Testing**
   Key business hypotheses (e.g., discount vs. profit, city-wise losses) were tested using advanced SQL techniques like joins, window functions, subqueries, CTEs, and CASE statements.
5. **Visualization & Reporting**
   A two-page interactive Power BI dashboard was created to communicate the insights through KPIs, bar charts, maps, line charts, matrices, and treemaps. Filters and slicers were added to allow business users to explore data easily.

Data Cleaning & Preprocessing

Step 1: Checking for Missing or Null Values

To ensure the reliability of analysis and avoid errors during aggregations, the first step in preprocessing was to check for missing or null values in each column of the dataset. This is a foundational step in any data cleaning pipeline because nulls can distort metrics like sums, averages, and joins.

We performed a column-wise null value check using SQL's COUNT(*) FILTER (WHERE column IS NULL) syntax. This allowed us to identify which fields (if any) contained missing information. Special attention was given to critical columns like order_id, customer_id, product_id, sales, and profit, which are essential for relational integrity and business analysis.

Fortunately, the dataset was found to be structurally complete, with no null values in any of the major columns. This confirmed that the dataset was fit for further transformation and modeling without the need for imputation or row deletion.

Query Used:

```
SELECT
    COUNT(*) FILTER (WHERE row_id IS NULL) AS null_row_id,
    COUNT(*) FILTER (WHERE order_id IS NULL) AS null_order_id,
    COUNT(*) FILTER (WHERE order_date IS NULL) AS null_order_date,
    COUNT(*) FILTER (WHERE ship_date IS NULL) AS null_ship_date,
    COUNT(*) FILTER (WHERE ship_mode IS NULL) AS null_ship_mode,
    COUNT(*) FILTER (WHERE customer_id IS NULL) AS null_customer_id,
    COUNT(*) FILTER (WHERE customer_name IS NULL) AS null_customer_name,
    COUNT(*) FILTER (WHERE segment IS NULL) AS null_segment,
    COUNT(*) FILTER (WHERE country IS NULL) AS null_country,
    COUNT(*) FILTER (WHERE city IS NULL) AS null_city,
    COUNT(*) FILTER (WHERE state IS NULL) AS null_state,
    COUNT(*) FILTER (WHERE postal_code IS NULL) AS null_postal_code,
    COUNT(*) FILTER (WHERE region IS NULL) AS null_region,
    COUNT(*) FILTER (WHERE product_id IS NULL) AS null_product_id,
    COUNT(*) FILTER (WHERE category IS NULL) AS null_category,
    COUNT(*) FILTER (WHERE sub_category IS NULL) AS null_sub_category,
    COUNT(*) FILTER (WHERE product_name IS NULL) AS null_product_name,
    COUNT(*) FILTER (WHERE sales IS NULL) AS null_sales,
    COUNT(*) FILTER (WHERE quantity IS NULL) AS null_quantity,
    COUNT(*) FILTER (WHERE discount IS NULL) AS null_discount,
    COUNT(*) FILTER (WHERE profit IS NULL) AS null_profit
FROM superstore;
```

Step 2: Checking for Duplicates

We checked for duplicate row_id values to ensure each record is unique, and also verified if any product was repeated within the same order (order_id, product_id). No duplicates were found, confirming that the data is clean and reliable for analysis.

Query Used:

```
---step 2: Check for Duplicates
SELECT row_id, COUNT(*)
FROM superstore
GROUP BY row_id
HAVING COUNT(*) > 1;

SELECT order_id, product_id, COUNT(*)
FROM superstore
GROUP BY order_id, product_id
HAVING COUNT(*) > 1;
```

Step 3: Checking for Negative Values

We checked whether any records had negative values for sales, quantity, or profit. These fields should logically be non-negative. The query confirmed that while sales and quantity were valid, some records had negative profit, which is acceptable in real business scenarios (e.g., due to high discounts or returns).

Query Used:

```
---Step 3: Check for Negative Values in sales, quantity, or profit
SELECT *
FROM superstore
WHERE sales < 0 OR quantity < 0 OR profit < 0;
```

Step 4: Validating Shipping Timeline

To ensure logical consistency, we checked that ship_date was not earlier than order_date. This condition validates that the product wasn't shipped before it was ordered. No violations were found, confirming clean temporal data.

QueryUsed:

```
---step 4:We have to  ensure ship_date >= order_date
SELECT *
FROM superstore
WHERE ship_date < order_date;
```

Step 5: Final Null Check (Full Table Scan)

As a final safeguard, we performed a full scan across all columns to ensure no NULL values were missed. The dataset passed this check, reaffirming that it was structurally complete and required no further null handling.

Query used:

```
---STEP 5: We will Handle NULL Values step wise
SELECT *
FROM superstore
WHERE row_id IS NULL
   OR order_id IS NULL
   OR order_date IS NULL
   OR ship_date IS NULL
   OR ship_mode IS NULL
   OR customer_id IS NULL
   OR customer_name IS NULL
   OR segment IS NULL
   OR country IS NULL
   OR city IS NULL
   OR state IS NULL
   OR postal_code IS NULL
   OR region IS NULL
   OR product_id IS NULL
   OR category IS NULL
   OR sub_category IS NULL
   OR product_name IS NULL
   OR sales IS NULL
   OR quantity IS NULL
   OR discount IS NULL
   OR profit IS NULL;
```

Step 6: Trimming Extra Spaces from Text Fields

To avoid issues during joins, filtering, and grouping, we removed leading and trailing spaces from all key text columns using the TRIM() function. This step ensures consistency in categorical fields like city, state, segment, and product_name, which are frequently used in analysis.

Query Used:

```
---STEP 6:We will trim Extra Spaces from Text Fields
UPDATE superstore
SET
    customer_id = TRIM(customer_id),
    customer_name = TRIM(customer_name),
    segment = TRIM(segment),
    country = TRIM(country),
    city = TRIM(city),
    state = TRIM(state),
    region = TRIM(region),
    product_id = TRIM(product_id),
    category = TRIM(category),
    sub_category = TRIM(sub_category),
    product_name = TRIM(product_name),
    ship_mode = TRIM(ship_mode);
```

Step 7: Data Validation – Outliers and Logical Errors

We performed logical validation checks to ensure data accuracy. First, we verified that neither order_date nor ship_date was set in the future. Next, we reconfirmed that ship_date was not earlier than order_date. Lastly, we checked for invalid negative values in numeric fields like sales, quantity, discount, and profit. All values were within acceptable ranges, with only profit having some negative entries, which is valid in retail scenarios involving losses or discounts.

Query Used:

```
---step 7: Data Validation - Outliers & Logical Errors
---7.1-> we will check for future dates (sales,shipping in the future would be invalid)
SELECT *
FROM superstore
WHERE order_date > CURRENT_DATE OR ship_date > CURRENT_DATE;

---7.2-> we will Check if ship_date is before order_date (it's illogical)
SELECT *
FROM superstore
WHERE ship_date < order_date;

---7.3->we will Check if their any negative values in sales, quantity, discount, profit
SELECT *
FROM superstore
WHERE sales < 0 OR quantity < 0 OR discount < 0 OR profit < 0;
```

Step 8: Data Transformation for Time-Based Analysis

Step 8.1: Extracting Date Features

To support time-based analysis and visual trends in the dashboard, we extracted new fields from order_date such as order_year, order_month, and order_weekday. These features help in analyzing seasonality, monthly performance, and weekday-wise sales patterns.

Query Used:

```
---8.1:we will extract Order Year, Order Month, Order Weekday
SELECT
    order_id,
    order_date,
    EXTRACT(YEAR FROM order_date) AS order_year,
    TO_CHAR(order_date, 'Month') AS order_month,
    TO_CHAR(order_date, 'Day') AS order_weekday
FROM superstore
LIMIT 5;
```

Step 8.2: Calculating Profit Margin %

We derived the profit_margin_percent to understand how much profit each sale generated relative to its value. This metric helps in identifying high-performing and loss-making transactions. The NULLIF function was used to safely handle cases where sales could be zero, preventing division errors.

Query Used:

```
---8.2:We will calculate Profit Margin
SELECT
    order_id,
    sales,
    profit,
    ROUND((profit / NULLIF(sales, 0)) * 100, 2) AS profit_margin_percent
FROM superstore
LIMIT 5;
```

Step 8.3: Creating Profit Category

To simplify profit analysis, we categorized each transaction into three groups: High Profit, Low Profit, and Loss. This classification helps in segmenting transactions based on profitability and enables quick identification of loss-making orders.

Query Used:

```
---8.3:We will create profit category here
SELECT
    order_id,
    profit,
    CASE
        WHEN profit > 100 THEN 'High Profit'
        WHEN profit BETWEEN 0 AND 100 THEN 'Low Profit'
        ELSE 'Loss'
    END AS profit_category
FROM superstore
LIMIT 5;
```

Step 9: Business Insight Generation through SQL Analytics

This covers all the upcoming analytical questions you'll answer using SQL, focused on deriving real-world business value from the data.

Step 9.1: Identifying Top 5 Customers by Total Profit

To understand which customers contribute most to the company's profit, we calculated the total profit generated by each customer and ranked them in descending order. This helps identify high-value customers who can be prioritized for loyalty programs or targeted marketing.

Query Used:

```
---9.1:Top 5 customer by total_profit
WITH customer_profit AS (
    SELECT
        customer_id,
        customer_name,
        SUM(profit) AS total_profit
    FROM superstore
    GROUP BY customer_id, customer_name
)
SELECT *
FROM customer_profit
ORDER BY total_profit DESC
LIMIT 5;
```

Step 9.2: Yearly Sales and Profit Trend

We analyzed the yearly trend of total sales and profit to observe business growth or decline over time. This helps in understanding long-term performance and planning future sales strategies accordingly.

Query Used:

```
---9.2:yearly sales trend
SELECT
    EXTRACT(YEAR FROM order_date) AS order_year,
    ROUND(SUM(sales), 2) AS total_sales,
    ROUND(SUM(profit), 2) AS total_profit
FROM superstore
GROUP BY order_year
ORDER BY order_year;
```

Step 9.3: Ranking Products by Sales Within Each Category

We used the RANK() window function to rank products based on their total sales within each product category. This helps identify top-performing products in every category, supporting inventory and marketing decisions.

Query Used:

```
---9.3:Rank products by sales in each category
SELECT
    category,
    sub_category,
    product_name,
    SUM(sales) AS total_sales,
    RANK() OVER (PARTITION BY category ORDER BY SUM(sales) DESC) AS sales_rank
FROM superstore
GROUP BY category, sub_category, product_name
ORDER BY category, sales_rank;
```

Step 9.4: Identifying Repeated Customers

We identified repeat customers by counting how many distinct orders each customer placed. Customers with more than 5 orders were considered frequent buyers, indicating higher loyalty and lifetime value.

Query Used:

```
---9.4:Repeated customers
SELECT
    customer_id,
    customer_name,
    COUNT(DISTINCT order_id) AS total_orders
FROM superstore
GROUP BY customer_id, customer_name
HAVING COUNT(DISTINCT order_id) > 5
ORDER BY total_orders DESC;
```

Step 10: Hypothesis-Driven Business Analysis

Hypothesis 1: High Discounts Lead to Lower Profits

To test whether higher discounts negatively impact profit, we grouped transactions by discount rate and calculated the average profit. The results help determine if excessive discounting is harming profitability, which is crucial for pricing strategy decisions.

Query Used:

```sql
--- Hypothesis 1:High discounts lead to lower profits
SELECT
    ROUND(discount, 2) AS discount_rate,
    ROUND(AVG(profit), 2) AS avg_profit,
    COUNT(*) AS transactions
FROM superstore
GROUP BY ROUND(discount, 2)
ORDER BY discount_rate;
```

Hypothesis 2: Furniture Has the Lowest Profit Margin Among All Categories

We calculated the overall profit margin for each product category to compare their profitability. This helped validate whether the Furniture category consistently generates lower returns compared to others, guiding decisions on pricing or product focus.

Code Used:

```sql
--- Hypothesis 2:Furniture has the lowest profit margin among all categories
SELECT
    category,
    ROUND(SUM(profit) / NULLIF(SUM(sales), 0) * 100, 2) AS profit_margin_percent
FROM superstore
GROUP BY category
ORDER BY profit_margin_percent;
```

Hypothesis 3: Orders from the West Region Are the Most Profitable

To test this hypothesis, we aggregated total profit by region and ranked them. This helped identify which region contributes the most to overall profitability, supporting regional strategy and resource allocation.

Code Used:

```sql
--- Hypothesis 3:Orders from the West region are the most profitable
SELECT
    region,
    ROUND(SUM(profit), 2) AS total_profit,
    COUNT(*) AS orders
FROM superstore
GROUP BY region
ORDER BY total_profit DESC;
```

Hypothesis 4: The Consumer Segment Brings in the Most Revenue

We evaluated total sales and profit across customer segments to determine which segment generates the highest revenue. This analysis supports customer segmentation strategy and helps focus marketing efforts on the most valuable group.

Code Used:

```
---Hypothesis 4:The Consumer segment brings in the most revenue
SELECT
    segment,
    ROUND(SUM(sales), 2) AS total_sales,
    ROUND(SUM(profit), 2) AS total_profit
FROM superstore
GROUP BY segment
ORDER BY total_sales DESC;
```

Hypothesis 5: Most Orders Are Shipped Using Standard Class

We analyzed the number of orders by shipping mode to identify the most commonly used method. The results help assess customer preferences and operational focus areas, especially in terms of shipping cost and delivery time.

Code used:

```
--- Hypothesis 5:Most orders are shipped using Standard Class
SELECT
    ship_mode,
    COUNT(*) AS total_orders,
    ROUND(SUM(sales), 2) AS total_sales
FROM superstore
GROUP BY ship_mode
ORDER BY total_orders DESC;
```

Hypothesis 6: The 'Tables' Sub-Category Causes the Most Loss

To identify the least profitable sub-category, we aggregated total profit by sub-category and sorted the results in ascending order. This helped confirm whether 'Tables' consistently results in losses, guiding decisions around pricing, discounting, or inventory control

Code Used:

```
---Hypothesis 6: Sub-category 'Tables' causes the most loss
SELECT
    sub_category,
    ROUND(SUM(profit), 2) AS total_profit
FROM superstore
GROUP BY sub_category
ORDER BY total_profit;
```

Final Business Problem Statements

Optimizing Profitability and Operational Efficiency for an E-commerce Retailer

Although the retail superstore generates healthy sales figures, a deeper analysis reveals that **some product categories are consistently incurring losses**, even with high sales volumes. This points to hidden inefficiencies in pricing, shipping, or product-level profitability.

Business Problem 1:

Certain product sub-categories are generating high sales but causing consistent losses.

Code Used:

```
SELECT
    p.sub_category,
    COUNT(sf.order_id) AS total_orders,
    SUM(sf.sales) AS total_sales,
    SUM(sf.profit) AS total_profit
FROM
    sales_facts sf
JOIN
    products p ON sf.product_id = p.product_id
GROUP BY
    p.sub_category
HAVING
    SUM(sf.profit) < 0
ORDER BY
    total_profit ASC;
```

Status:

Confirmed. Some sub-categories are causing losses despite high sales.

Business Insights:

- The **Tables** sub-category generated over ₹2,00,000 in sales but incurred a total loss of ₹17,725.
- **Bookcases** and **Supplies** also showed negative profits, even with decent sales volume.
- These sub-categories are silently pulling down overall profitability.

Recommended Actions:

- Review pricing strategies and discount thresholds for these sub-categories.
- Investigate operational costs such as shipping and return rates.
- Consider phasing out or bundling these items with more profitable ones to minimize loss impact.

Business Problem 2:

Do discounts actually improve profit margins or reduce them?

Query Used:

```
SELECT
  CASE
    WHEN discount = 0 THEN '0%'
    WHEN discount > 0 AND discount <= 0.1 THEN '0.1 - 0.1'
    WHEN discount > 0.1 AND discount <= 0.2 THEN '0.1 - 0.2'
    WHEN discount > 0.2 AND discount <= 0.3 THEN '0.2 - 0.3'
    WHEN discount > 0.3 AND discount <= 0.4 THEN '0.3 - 0.4'
    ELSE '>0.4'
  END AS discount_range,
  COUNT(*) AS total_orders,
  ROUND(AVG(sales), 2) AS avg_sales,
  ROUND(AVG(profit), 2) AS avg_profit,
  ROUND(SUM(profit), 2) AS total_profit
FROM sales_facts
GROUP BY discount_range
ORDER BY discount_range;
```

Status:

Confirmed. Discounts above 20% consistently result in negative profit, despite higher sales volumes.

Business Insights:

- Orders with **0% to 20% discounts** showed healthy profit margins.
- Discounts in the **20% to 40% range** brought profit levels dangerously low.
- Discounts above **40%** led to **substantial losses**, indicating over-discounting backfires.

Recommended Actions:

- **Avoid giving discounts beyond 20%**, as they directly erode profit margins.
- Focus on **controlled, moderate discounts** (0–10%) which generate better balance between sales and profitability.
- Consider limiting high discounts to clearance or bundled deals only.

Business Problem 3:

Do high shipping delays reduce profitability?

Query Used:

```
SELECT
    CASE
        WHEN ship_date - order_date = 0 THEN '0 Days'
        WHEN ship_date - order_date BETWEEN 1 AND 2 THEN '1-2 Days'
        WHEN ship_date - order_date BETWEEN 3 AND 4 THEN '3-4 Days'
        WHEN ship_date - order_date BETWEEN 5 AND 6 THEN '5-6 Days'
        ELSE '7+ Days'
    END AS delay_bucket,
    COUNT(*) AS total_orders,
    ROUND(AVG(profit), 2) AS avg_profit,
    ROUND(SUM(profit), 2) AS total_profit
FROM orders
JOIN sales_facts USING(order_id)
GROUP BY delay_bucket
ORDER BY delay_bucket;
```

Status:

Rejected. There is no clear pattern showing that longer shipping delays negatively impact profit.

Business Insights:

- Orders with **5–6 days delay** had average profit of ₹27.29 — comparable to lower-delay buckets.
- Even **7+ days delayed orders** showed healthy profit margins.
- The lowest average profit was observed in the **3–4 day** range, but the difference was not significant

Recommended Actions:

- Shipping delays are **not a major factor** impacting profit in this dataset.
- Operational efforts should focus more on **discount control** and **product-level profitability** rather than aggressively shortening delivery windows.
- However, long delays may still impact **customer satisfaction**, which should be tracked separately.

Business Problem 4:

Does the impact of discounts on profit vary across sub-categories?

Query Used:

```sql
SELECT
    sub_category,
    CASE
        WHEN discount = 0 THEN '0%'
        WHEN discount > 0 AND discount <= 0.1 THEN '0-10%'
        WHEN discount > 0.1 AND discount <= 0.2 THEN '10-20%'
        WHEN discount > 0.2 AND discount <= 0.3 THEN '20-30%'
        WHEN discount > 0.3 AND discount <= 0.4 THEN '30-40%'
        WHEN discount > 0.4 THEN '>40%'
    END AS discount_range,
    COUNT(*) AS total_orders,
    ROUND(SUM(sales), 2) AS total_sales,
    ROUND(SUM(profit), 2) AS total_profit,
    ROUND(AVG(profit), 2) AS avg_profit_per_order
FROM sales_facts sf
JOIN products p ON sf.product_id = p.product_id
GROUP BY sub_category, discount_range
ORDER BY sub_category, discount_range;
```

Status:

Confirmed. The effect of discounting is not uniform — some sub-categories are severely hurt by high discounts, while others remain profitable even at 0% discount.

Business Insights:

Sub-categories severely hurt by high discounts:

- **Tables, Bookcases, Binders, Machines** show strong negative profit when discounts exceed 30%.
- High fixed costs or thin margins in these categories make discounting ineffective and damaging.

Interpretation:

These are likely heavy or costly-to-ship items. Applying high discounts shrinks margins or turns profit negative. These categories require stricter discount control or alternate sales strategies.

Sub-categories thriving even without discounts:

- **Copiers, Chairs, Phones** show strong profits at 0% discount.
- These high-margin items don't need discounting to sell.

Interpretation:

These are likely premium or essential items. Customers are willing to buy them at full price. The focus should be on value positioning and not on price cuts.

Sub-categories showing mixed signals:

- **Storage**: Profitable at 0% (₹25,528.17), but turns negative in the 10–20% discount band.
- **Supplies**: Enters loss even at moderate discount levels.
- **Accessories**: Profitable without discounts, but average profit per order drops steeply as discounts rise.

Interpretation:

These categories need a closer look — profitability is sensitive to even moderate discounting. One-size-fits-all strategies won't work here.

Business Problem 5:

Are some high-revenue cities or regions actually unprofitable?

Here we will identify high-revenue but low-margin or even loss-making geographies.

Query Used:

```
SELECT
    c.city,
    c.state,
    c.region,
    COUNT(o.order_id) AS total_orders,
    ROUND(SUM(sf.sales), 2) AS total_sales,
    ROUND(SUM(sf.profit), 2) AS total_profit,
    ROUND(SUM(sf.profit) / NULLIF(SUM(sf.sales), 0), 2) AS profit_margin
FROM
    sales_facts sf
JOIN
    orders o ON sf.order_id = o.order_id
JOIN
    customers c ON o.customer_id = c.customer_id
GROUP BY
    c.city, c.state, c.region
ORDER BY
    total_sales DESC
LIMIT 50;
```

Status:

Confirmed. Several cities generate high sales but deliver either low or negative profits.

Business Insights:

- Cities like **Monroe**, **Burlington**, **Santa Barbara**, and **Springfield** showed **significant sales** but resulted in **net losses**.
- Cities such as **Houston**, **Charlotte**, and **Dublin** reported **profit margins below 5%**, indicating inefficient profitability.

Interpretation:

These results suggest that some regions may be suffering from over-discounting, high operational costs, or poor product-market fit. Despite strong customer engagement (high order count), the profit per sale is too low or negative.

Recommended Actions:

For cities with negative profits:

- Audit discounting levels — especially those exceeding 30–40%.
- Check product return/refund rates and customer complaints.
- Evaluate last-mile delivery costs, warehousing inefficiencies, or regional taxes.

For cities with low profit margins (<5%):

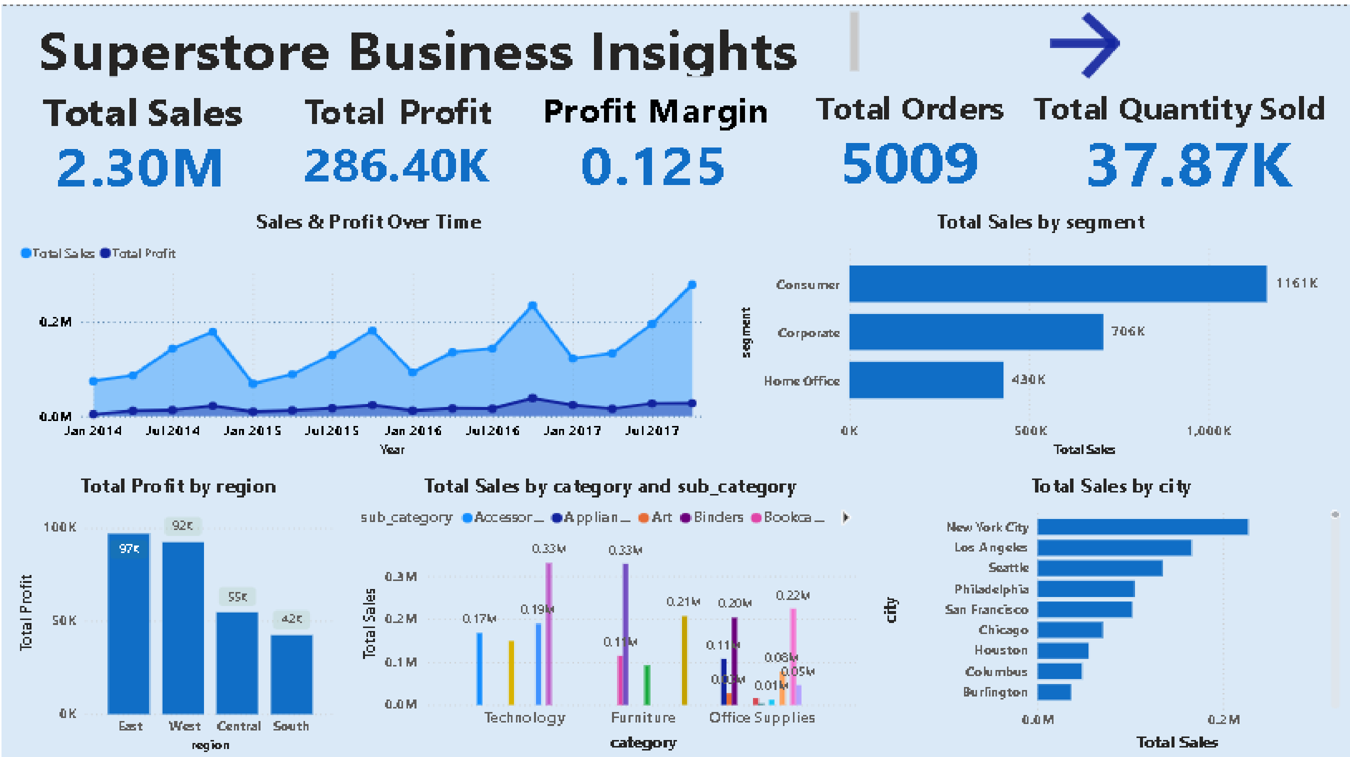
- Run targeted audits on high-selling SKUs.
- Use bundling to pair high-margin items with popular products.
- Consider strategic price revisions based on product performance in those regions.

Power BI Dashboard & Visual Reporting

After completing the SQL-based data analysis, the insights were transformed into an interactive and user-friendly **Power BI dashboard** to support business decision-making.

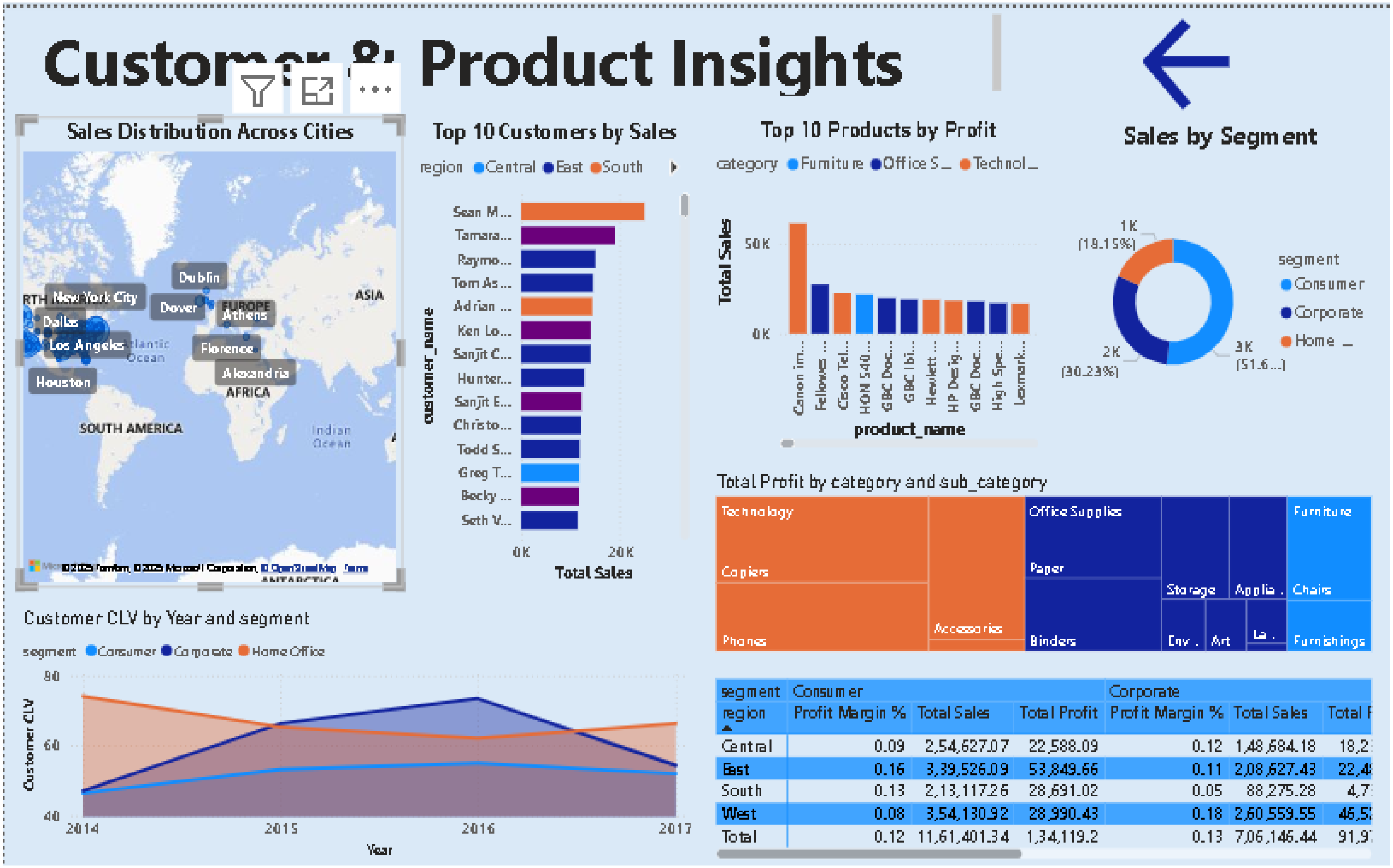
Page 1: Business Insights

The first page of the Power BI dashboard provides a comprehensive snapshot of the superstore's business performance across multiple dimensions. It begins with high-level KPIs, showcasing total sales, total profit, profit margin, total orders, and quantity sold — giving an immediate understanding of scale and efficiency. A time-series line chart visualizes monthly sales and profit trends, enabling seasonal pattern analysis. Further, the dashboard highlights regional profitability, segment-wise revenue contribution, city-level sales distribution, and product performance across categories and sub-categories. Together, these visuals allow stakeholders to identify top-performing regions, customer segments, and products while spotting underperforming areas. This page reflects strong analytical skills such as segmentation, trend detection, comparative analysis, and use of DAX for custom metrics, all presented through a clean, interactive layout suitable for strategic decision-making.



Page 2: Customer & Product Insights

The second page of the dashboard focuses on deeper analysis of customers and product performance. A geographic map shows the distribution of sales across global cities, helping visualize high-revenue zones and potential market clusters. Next to it, a bar chart ranks the **Top 10 customers by total sales**, color-coded by region, revealing high-value accounts that drive a significant portion of revenue. Another visual lists the **Top 10 most profitable products**, enabling the business to identify best-sellers that deliver both revenue and margins. A donut chart shows segment-wise sales contribution for quick comparison. The **Customer CLV chart** tracks Customer Lifetime Value trends over the years across segments, highlighting changes in long-term value from different customer types. A treemap breaks down total profit by category and sub-category, while a matrix gives a combined view of segment, region, sales, profit, and margins — allowing multi-dimensional analysis. This page demonstrates skills like customer segmentation, profitability ranking, CLV analysis, and spatial intelligence — all essential for customer targeting and product strategy planning.



Conclusion

This project demonstrates the complete journey of a data analyst — from cleaning raw retail data in PostgreSQL to delivering actionable insights through an interactive Power BI dashboard. We began with detailed data cleaning and transformation, ensuring accuracy in fields like dates, profit margins, and discount ranges. Using advanced SQL techniques like joins, subqueries, CTEs, and window functions, we explored key business questions around profitability, customer behavior, and operational inefficiencies.

Our analysis revealed high-sales sub-categories that are actually loss-making, the negative impact of over-discounting beyond 20%, and geographic regions where strong sales don't guarantee strong profits. These insights were summarized in a two-page Power BI dashboard designed for decision-makers, with filters, visuals, and KPIs for clear storytelling.

More than just tool usage, this project reflects strong business understanding, analytical thinking, and clear communication — the core of what real-world data analytics is all about.

Future Scope

While this project delivered actionable insights using structured SQL analysis and Power BI visualization, it also opens the door for several future enhancements:

1. **Real-Time Data Integration**
   Connecting the dashboard to a live database or API would enable dynamic reporting and up-to-date insights without manual refreshes.
2. **Return Rate & Refund Analysis**
   Adding return/refund data can help identify products or regions with high reverse logistics costs — improving profit estimation accuracy.
3. **Customer Segmentation Using RFM or ML**
   Advanced segmentation using Recency, Frequency, and Monetary value — or clustering via machine learning — can help build personalized marketing strategies.
4. **Forecasting & Predictive Modeling**
   Incorporating time-series forecasting (using Prophet or ARIMA) for future sales and demand planning would bring proactive decision-making into the business workflow.
5. **Operational Dashboards for Specific Teams**
   Separate dashboards can be built for sales, supply chain, or marketing teams — each focused on their KPIs, backed by the same data model.
6. **Integration with Financial KPIs**
   Adding overheads, fixed costs, and advertising data would turn the dashboard into a full profit-and-loss tool, giving leaders a clearer financial picture.