**Retail Sales Analytics & Business Intelligence Solution**

**1. Introduction**

* Brief overview of retail data challenges.
* Importance of analytics and BI in decision-making.
* Tools & technologies used: **Python, SQL, Power BI, Tableau, Prompt Engineering**.

**2. Objective**

The objective of this project is to design an end-to-end data analytics and business intelligence solution for retail sales data. The solution leverages Python, SQL, Power BI, Tableau, and Prompt Engineering to transform raw transactional data into actionable insights. The focus is on understanding sales trends, customer behaviour, regional performance, and profitability to drive better business decisions.

**3. Problem Statement**

Retail businesses generate massive volumes of transactional data but struggle to extract meaningful insights for decision-making. This project addresses the challenge by:

* Cleaning and transforming raw data into structured formats.
* Identifying key business metrics such as sales growth, profitability, and customer segmentation.
* Building interactive dashboards and AI-driven summaries for executives.
* Delivering a scalable and automated pipeline that integrates multiple analytics tools.

**4. Scope of Work**

**4.1 Data Preparation & Storage**

* Collected and cleaned the **Global Superstore dataset**.
* Stored structured tables in **MySQL** for scalable querying.

**#Step 1: Create database**

CREATE DATABASE global\_superstore;

USE global\_superstore;

**#Step 2: Create tables for each CSV**

**#1. Orders.csv**

CREATE TABLE orders (

row\_id INT,

order\_id VARCHAR(50),

order\_date DATE,

ship\_date DATE,

ship\_mode VARCHAR(50),

customer\_id VARCHAR(50),

customer\_name VARCHAR(100),

segment VARCHAR(50),

country VARCHAR(50),

city VARCHAR(50),

state VARCHAR(50),

postal\_code VARCHAR(20),

region VARCHAR(50),

product\_id VARCHAR(50),

category VARCHAR(50),

sub\_category VARCHAR(50),

product\_name VARCHAR(200),

sales DECIMAL(12,2),

quantity INT,

discount DECIMAL(5,2),

profit DECIMAL(12,2)

);

**#2. Returns.csv**

CREATE TABLE returns (

order\_id VARCHAR(50),

returned VARCHAR(10) -- usually 'Yes' or NULL

);

**#3. People.csv**

CREATE TABLE people (

person VARCHAR(100),

region VARCHAR(50)

);

**4.2 SQL-Based Business Analysis**

* Designed SQL queries to extract customer insights, product profitability, and sales trends.
* Built regional and category-level comparisons for better decision-making.

#Step 2: SQL-Based Business Analysis

**#2.1 Customer Insights**

**#Top 10 customers by total sales**

SELECT Customer\_Name, SUM(Sales) AS Total\_Sales

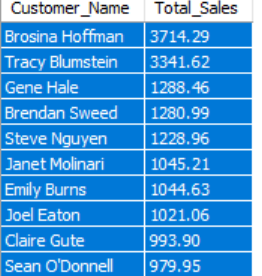
FROM Orders

GROUP BY Customer\_Name

ORDER BY Total\_Sales DESC

LIMIT 10;

**RESULT:**



**#Most profitable customers**

SELECT Customer\_Name, SUM(Profit) AS Total\_Profit

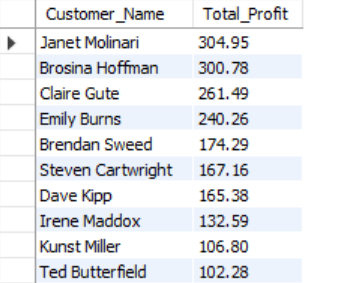
FROM Orders

GROUP BY Customer\_Name

ORDER BY Total\_Profit DESC

LIMIT 10;

**RESULT:**



**#Customers with highest order frequency**

SELECT Customer\_Name, COUNT(Order\_ID) AS Order\_Count

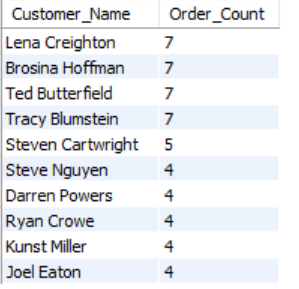
FROM Orders

GROUP BY Customer\_Name

ORDER BY Order\_Count DESC

LIMIT 10;

**RESULT:**



**##2.2 Product Profitability**

**#Top 10 products by profit**

SELECT Product\_Name, SUM(Profit) AS Total\_Profit

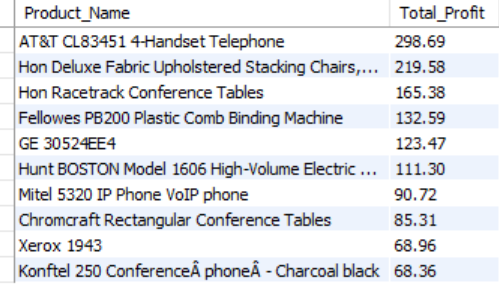
FROM Orders

GROUP BY Product\_Name

ORDER BY Total\_Profit DESC

LIMIT 10;

**RESULT:**



**#Products causing most losses (negative profit)**

SELECT Product\_Name, SUM(Profit) AS Total\_Profit

FROM Orders

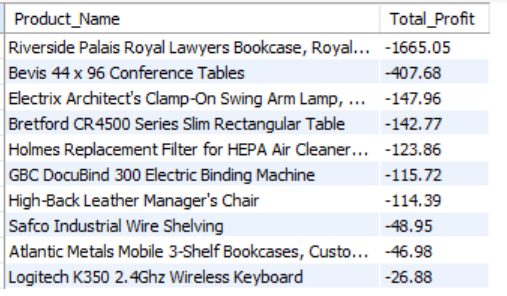
GROUP BY Product\_Name

HAVING SUM(Profit) < 0

ORDER BY Total\_Profit ASC

LIMIT 10;

**RESULT:**



**#Category-wise sales and profit**

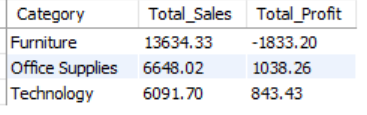
SELECT Category, SUM(Sales) AS Total\_Sales, SUM(Profit) AS Total\_Profit

FROM Orders

GROUP BY Category

ORDER BY Total\_Sales DESC;

**RESULT:**

****

**##2.3 Sales Trends**

**#Yearly sales & profit**

SELECT YEAR(Order\_Date) AS Year,

SUM(Sales) AS Total\_Sales,

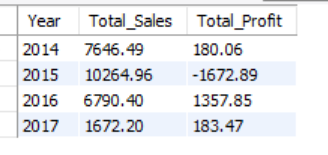
SUM(Profit) AS Total\_Profit

FROM Orders

GROUP BY YEAR(Order\_Date)

ORDER BY Year;

**RESULT:**



**#Monthly sales trend (across years)**

SELECT YEAR(Order\_Date) AS Year, MONTH(Order\_Date) AS Month,

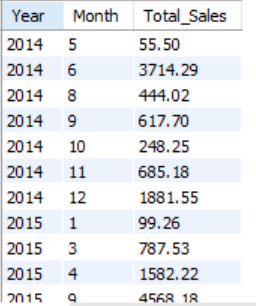
SUM(Sales) AS Total\_Sales

FROM Orders

GROUP BY YEAR(Order\_Date), MONTH(Order\_Date)

ORDER BY Year, Month;

**RESULT:**



**#2.4 Regional & Category-Level Comparisons**

**#Sales by Region & Category**

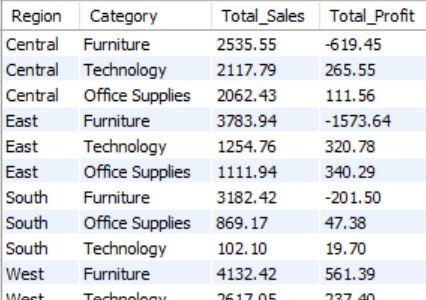
SELECT Region, Category, SUM(Sales) AS Total\_Sales, SUM(Profit) AS Total\_Profit

FROM Orders

GROUP BY Region, Category

ORDER BY Region, Total\_Sales DESC;

**RESULT:**



**4.3 Python Analysis & Visualization**

* Performed exploratory data analysis (EDA) using **pandas, matplotlib, seaborn**.
* Identified seasonal sales patterns, loss-making products, and regional performance variations.

**3.1 Setup & Data Import**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

orders = pd.read\_csv("Orders.csv")

returns = pd.read\_csv("Returns.csv")

people = pd.read\_csv("People.csv")

# Quick check

print(orders.shape)

print(orders.head())

**3.2 Exploratory Data Analysis (EDA)**

* Check missing values & data types

print(orders.info())

print(orders.isnull().sum())

* Summary statistics

print(orders.describe())

**3.3 Patterns to Explore**

🔹 **Seasonal Sales Trend**

orders['Order Date'] = pd.to\_datetime(orders['Order Date'])

orders['YearMonth'] = orders['Order Date'].dt.to\_period('M')

monthly\_sales = orders.groupby('YearMonth')['Sales'].sum()

plt.figure(figsize=(12,6))

monthly\_sales.plot()

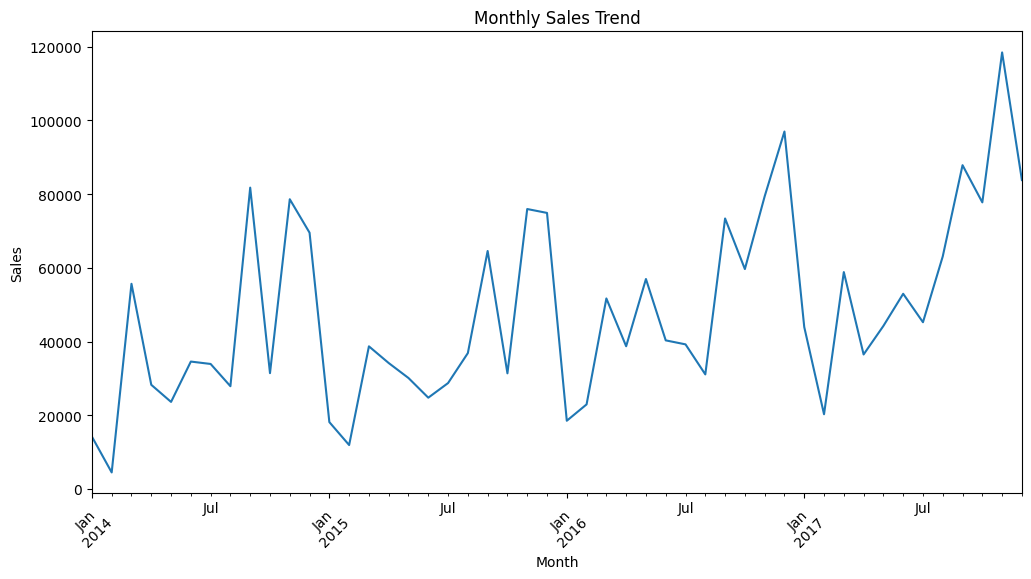
plt.title("Monthly Sales Trend")

plt.xlabel("Month")

plt.ylabel("Sales")

plt.xticks(rotation=45)

plt.show()



🔹 **Loss-Making Products** (negative profit)

loss\_products = (orders.groupby('ProductName')['Profit']

.sum()

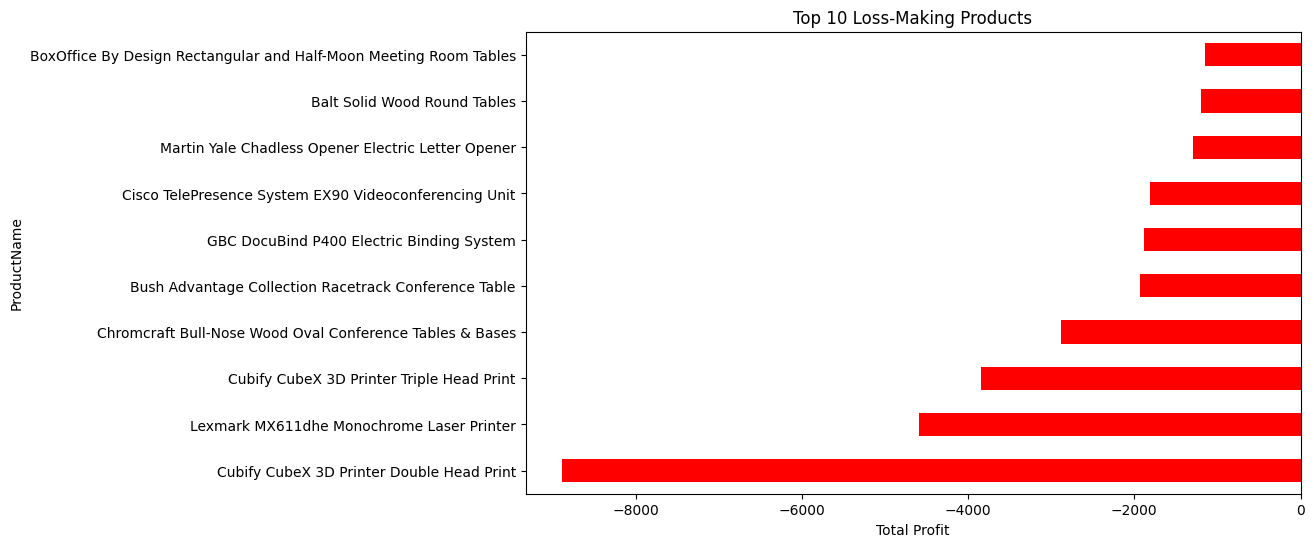
.sort\_values())

loss\_products.head(10).plot(kind='barh', figsize=(10,6), color="red")

plt.title("Top 10 Loss-Making Products")

plt.xlabel("Total Profit")

plt.show()



🔹 **Regional Performance Variations**

region\_sales = orders.groupby('Region')[['Sales','Profit']].sum().reset\_index()

plt.figure(figsize=(8,6))

sns.barplot(x='Region', y='Sales', data=region\_sales)

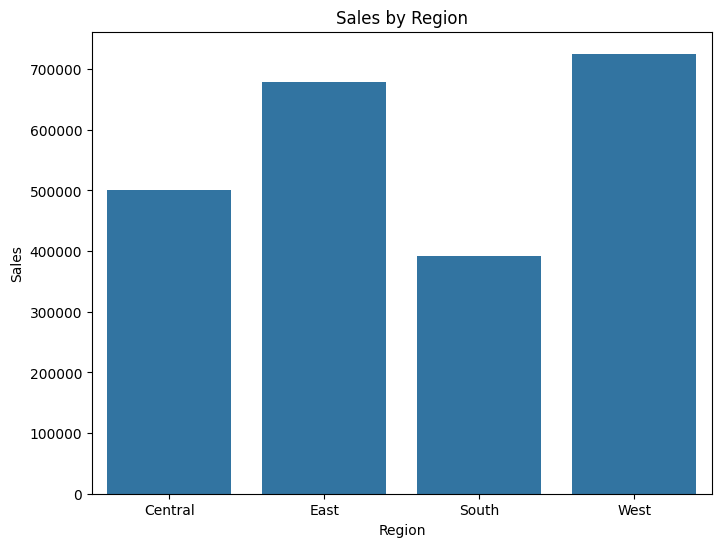
plt.title("Sales by Region")

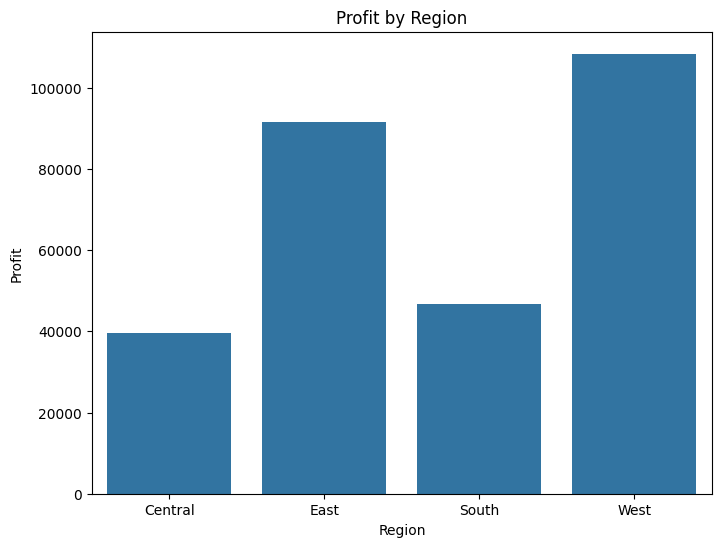
plt.show()

plt.figure(figsize=(8,6))

sns.barplot(x='Region', y='Profit', data=region\_sales)

plt.title("Profit by Region")

plt.show()



🔹 **Category & Sub-Category Insights**

plt.figure(figsize=(10,6))

sns.barplot(x='Category', y='Sales', data=orders, estimator=sum, ci=None)

plt.title("Sales by Category")

plt.show()

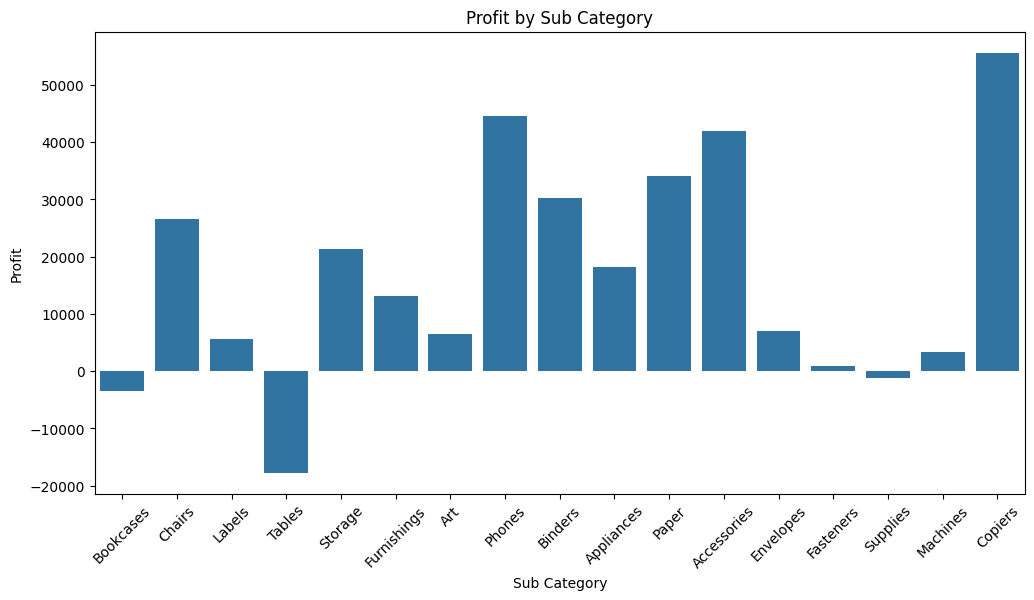
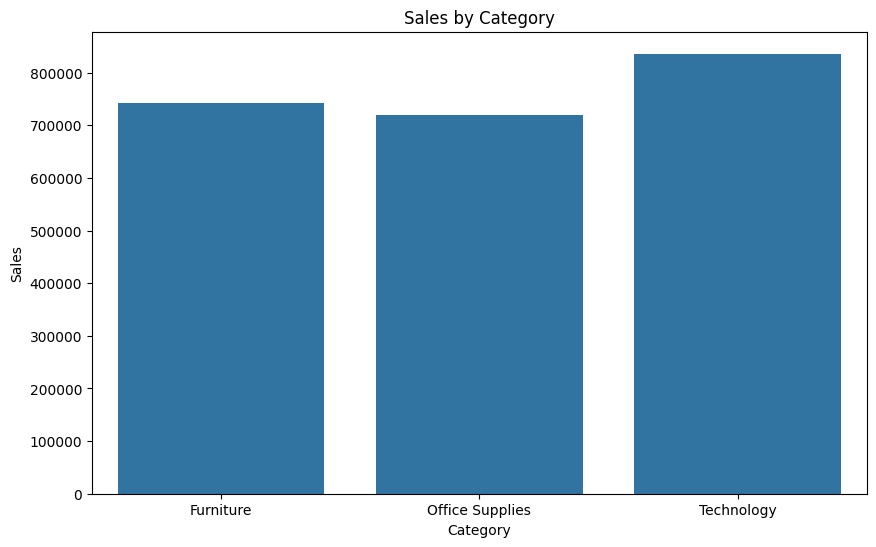
plt.figure(figsize=(12,6))

sns.barplot(x='Sub Category', y='Profit', data=orders, estimator=sum, ci=None)

plt.title("Profit by Sub Category")

plt.xticks(rotation=45)

plt.show()



**4.4 Business Intelligence Dashboards**

* **Power BI** dashboards: Sales KPIs, regional breakdowns, and customer segmentation.
* **Tableau storyboards**: Visual storytelling of sales & profitability trends.

**Part 1: Power BI Dashboards (Sales KPIs, Regional Breakdown, Customer Segmentation)**

**Step 1: Create a Unique Orders Table**

1. Go to Modeling → New Table in Power BI.
2. Enter this DAX formula:

UniqueOrders = DISTINCT(Orders[Order ID])

* This creates a table with one row per Order ID (no duplicates).

**Step 2: Link the Tables**

1. Go to Model View.
2. Drag UniqueOrders[Order ID] → Returns[Order ID].
3. Set Cardinality → One-to-Many (UniqueOrders = One, Returns = Many).
4. Cross filter → Single.

**Step 3: KPIs**

**1. Total Sales**

Total Sales = SUM(Orders[Sales])

**2. Total Profit**

Total Profit = SUM(Orders[Profit])

**3. Profit Margin %**

Profit Margin % = DIVIDE([Total Profit], [Total Sales], 0)

**4. Total Quantity**

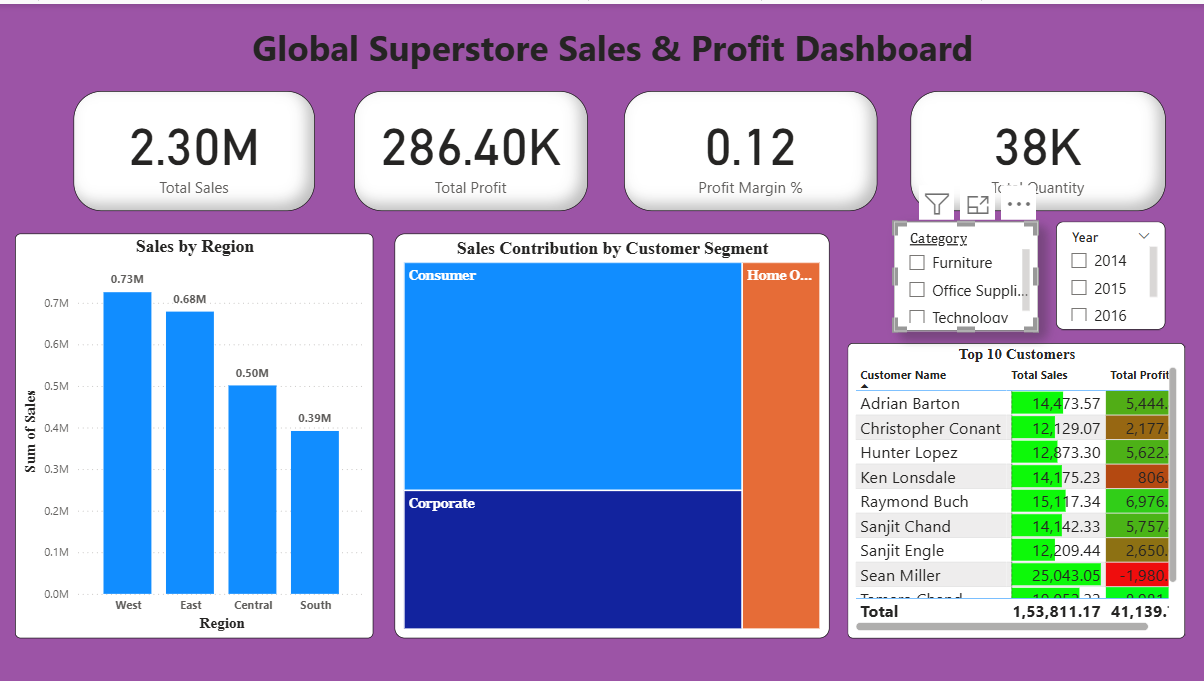
Total Quantity = SUM(Orders[Quantity])

**Recommended Visuals:**

* **Clustered Column Chart** (best for 4 regions)
  1. X-axis → Region
  2. Y-axis → Sales (or Profit)
  3. Optional: Add **Profit as color** (Conditional Formatting)
  4. Turn on **Data Labels** → shows numbers on bars
* **Tree Map**
  1. Group → Segment
  2. Values → Sales
  3. Shows proportion of each segment visually
* **Table Visual (Top Customers)**
  1. Columns → Customer Name, Sales, Profit
  2. Sort by Sales descending → shows top 10 customers
  3. Optional: Add conditional formatting → highlight high/low profit

**Dashboard Interactivity**

* Add **Slicers**:
  + Year (Order Date)
  + Category / Sub-Category



**Part 2: Tableau Storyboards (Sales & Profitability Trends)**

**Step 1: Import Data**

* Load your Orders, Returns, People CSVs into Tableau.

**Step 2: Create Dashboards**

**1️⃣ Sales Trend Dashboard**

**Visuals & Titles:**

* **Line Chart:** Monthly Sales Trend
  + X-axis → Order Date (Month/Year)
  + Y-axis → Sales
  + Title: *“Monthly Sales Trend”*
* **Bar Chart:** Sales by Category
  + X-axis → Category
  + Y-axis → Sales
  + Title: *“Sales by Category”*

**2️⃣ Profitability Dashboard**

**Visuals & Titles:**

* **Bar Chart:** Profit by Sub-Category
  + X-axis → Sub-Category
  + Y-axis → Profit
  + Title: *“Profit by Sub-Category”*

**3️⃣ Regional Performance Dashboard**

**Visuals & Titles:**

* **Filled Map:** Sales & Profit by Region
  + Color → Profit or Sales intensity
  + Tooltip → show Sales & Profit numbers
  + Title: *“Regional Sales & Profit Map”*
* **Stacked Bar Chart:** Sales by Region & Category
  + X-axis → Region
  + Y-axis → Sales
  + Color → Category
  + Title: *“Sales by Region & Category”*

**Step 3: Build Storyboard**

* Go to **Story** tab → add each dashboard as a **story point/slide**.
* Suggested story sequence:
  1. **Executive Summary** → Overall Sales & Profit
  2. **Sales Trend Analysis** → Line + Bar charts
  3. **Profitability Insights** → Profit by Sub-Category
  4. **Regional Performance** → Map + Stacked Bar

**4.5 AI & Prompt Engineering Integration**

* Applied **prompt engineering** to generate executive summaries and recommendations.
* Developed chatbot-style **Q&A system** for interactive data exploration.

Act as a helpful data analyst. Your task is to answer questions about a company's sales performance based on three CSV files: **Orders.csv**, **People.csv**, and **Returns.csv**.

**The Data:**

* **Orders.csv** → Contains detailed information for each order.  
  *Columns:* Row ID, Order ID, Order Date, Ship Date, Ship Mode, Customer ID, Customer Name, Segment, Country, City, State, Postal Code, Region, Product ID, Category, Sub-Category, Product Name, Sales, Quantity, Discount, Profit.
* **People.csv** → Connects regional managers to their regions.  
  *Columns:* Person, Region.
* **Returns.csv** → Lists all the returned orders.  
  *Columns:* Returned, Order ID.

**Your Task:**

* Act as a chatbot and answer questions from the user.
* Use only the data from the provided CSV files.
* Your answers should be clear, accurate, and easy to understand.
* You may need to combine information from the Orders.csv, People.csv, and Returns.csv files.
* If a question cannot be answered with the data in these files, simply state that the information is not available.

**Start by introducing yourself as the data analysis chatbot and wait for the first question.**

**5. Key Outcomes**

* Improved understanding of **sales trends & profitability drivers**.
* Automated pipeline for **data cleaning, analysis, and visualization**.
* Enhanced **executive decision-making** with AI-powered summaries.
* Scalable and multi-tool integration for **cross-platform analytics**.

**6. Tools & Technologies**

* **Data Storage**: MySQL
* **Programming**: Python (pandas, matplotlib, seaborn)
* **Business Intelligence**: Power BI, Tableau
* **AI Integration**: Prompt Engineering

**7. Conclusion**

This project demonstrates how raw retail sales data can be transformed into actionable insights using a combination of data engineering, analytics, and business intelligence tools. By integrating **SQL, Python, BI dashboards, and AI techniques**, the solution empowers businesses to make **data-driven strategic decisions**.