Superstore

These SQL questions were thoughtfully generated using ChatGPT and range from beginner to highly advanced levels. They are designed to support practical, real-world data exploration of a Superstore dataset. Each question focuses on uncovering insights related to customers, products, sales, and profit, with the goal of helping businesses make data-driven decisions to optimize performance and drive growth.

Dataset: SuperStore Time Series Dataset

📊 Sales & Profit Analysis

1. What is the total profit and sales per category and sub-category?
2. Which product has generated the highest total profit?
3. Which sub-category has the lowest average profit per order?
4. What is the profit margin (Profit/Sales) for each product, ordered by highest?
5. Which customer has placed the most orders and what's their total profit contribution?

⏱️ Time Series & Date Operations

6. How does monthly total sales trend over time?
7. What is the average shipping delay (days between Order_Date and Ship_Date) per region?
8. Which month and year had the highest total sales?
9. How many orders were shipped late (i.e., Ship_Date > Order_Date + 2 days)?
10. What is the average profit per order over the years?

📍 Geographic Analysis

11. Which state has the highest total discount given?
12. Which city generated the most sales per capita (average per order)?
13. Which region has the highest number of orders and total profit?

👥 Customer & Segment Insights

14. What is the average sales and profit per segment?
15. How many unique customers placed orders in each segment?
16. Who are the top 5 most profitable customers by total profit?

📦 Product-Level Exploration

17. Which products are sold the most by quantity, and what's their total sales?
18. What's the average discount by sub-category and its impact on average profit?
19. Are there any products that were sold at a loss (negative profit)? List them.
20. What are the top 10 best-selling products based on quantity sold?

🔄 Window Functions / Ranking

21. For each sub-category, rank the products by total sales. Then return the top-selling product per sub-category.

22. Calculate the running total of sales by month and segment.
23. For each customer, find their first and last purchase date, and calculate the total number of days they were active.
24. Identify products where the profit consistently decreased over time (yearly), using a window function to compare year-over-year profit.

📈 Complex Analysis / Correlated Subqueries

25. For each order, calculate the percentage contribution of each product’s sales to the total order sales.
26. List customers whose average profit per order is higher than the overall average profit per order.
27. Identify the sub-category that has the highest profit variance (standard deviation of profit).

🔍 CTEs / Recursive / Joins

28. Using a recursive CTE, simulate month-by-month cumulative sales growth for a given year.
29. Join the Customer_ID with itself to find customer pairs that ordered the same product in the same city but on different dates.

🧠 Advanced Insight & Optimization

30. Which product has the highest average profit per unit sold, but only among products that were sold in more than 10 different cities and had a discount applied at least once?

🔮 Deep Analytics / Multi-layered Logic

21. For each category, identify the product that had the highest number of returns (assume negative profit = return/loss).

    Then calculate the percentage that product contributed to total category losses.
22. Create a cohort analysis by grouping customers based on their first purchase year and then track their total profit contribution over the next 3 years.
23. Detect patterns where a customer made multiple purchases of the same product in the same month — return only those repeating product purchases and their profit impact.

🔮 Deep Analytics / Multi-layered Logic

31. For each category, identify the product that had the highest number of returns (assume negative profit = return/loss).

    Then calculate the percentage that product contributed to total category losses.
32. Create a cohort analysis by grouping customers based on their first purchase year and then track their total profit contribution over the next 3 years.
33. Detect patterns where a customer made multiple purchases of the same product in the same month — return only those repeating product purchases and their profit impact.

🧠 Dense Calculations with Subqueries / CTEs

34. Find the top 3 most profitable products within each region for the last full year of data.

35. Identify all products whose sales have increased every single quarter over time (monotonically increasing sales trend).

36. Determine which customer had the biggest percentage drop in profit between two consecutive years.

🔁 Recursive Logic & Hierarchical Joins

37. Assuming each order can lead to another (based on a Customer_ID placing another order within 7 days), build a recursive chain to calculate the longest sequence of dependent orders per customer.

📌 Outlier Detection / Statistical Analysis

38. Calculate the z-score of profit for each product, then return products with z-scores > 2 or < -2 (significant outliers in profitability).

🔀 Cross Analysis / Correlations

39. Correlate discount with profit by sub-category using a windowed correlation formula. Which sub-category shows the strongest negative correlation?

40. Create a product affinity table showing which products are frequently purchased together (same Order_ID), and count how often each pair occurs.

🧩 Business-Driven & Strategic Analysis Questions

41. Which customers are at risk of churn?
Identify customers who haven't placed any orders in the last 12 months and had a declining profit trend in their last 3 orders.

42. Which products are frequently sold with high discounts but generate little to no profit?
Analyze high-discount products and their corresponding profit margins to identify potential candidates for pricing strategy review.

43. What is the reorder rate for products by category?
Find the percentage of customers who ordered the same product more than once, grouped by category.

44. Which customer segments contribute the highest profit per marketing dollar?
Assume a marketing cost per order based on segment, and calculate profit minus marketing cost.

45. What is the lifetime value (LTV) of customers across different regions?
Aggregate profit per customer over time, grouped by region, to assess customer value distribution.

46. Which shipping modes are associated with higher return rates?
Assuming negative profit = return, analyze return rates by shipping mode to assess logistical impact.

47. Identify seasonal trends in product performance.
Highlight products that sell significantly more in specific quarters (seasonality) and their corresponding profit impact.

48. What are the top underperforming products by sales-to-inventory ratio?

Assume an inventory quantity field or use quantity sold as a proxy; identify products with low turnover.

49. Which cities show high demand but low profitability?

Find cities with high order volume but below-average profit per order — a red flag for operational inefficiencies or pricing issues.

50. Which sub-categories have rising customer acquisition but declining profitability?

Detect product sub-categories with an increasing number of new customers year over year but a decreasing profit trend.

Time Series Analysis

Time series data is a collection of variables whose values depend on time. Analyzing time-series data is trivial with Python, but with SQL, it becomes a pretty challenging task. Work on this project to understand what difficulties one might encounter using SQL for time series analysis.

Dataset: Use the SuperStore Time Series Dataset from Kaggle to work on this project. The dataset contains 20 columns, namely, Row ID, Order ID, Order Date, Ship Date, Ship Mode, Customer ID, Customer Name, Segment, Country, and City.

SQL Project Idea: Clean the data first using the data preprocessing method and make it SQL-ready. After that, complete the following tasks:

- Use the LEAD window function to create a new column sales_next that displays the sales of the next row in the dataset. This function will help you quickly compare a given row's values and values in the next row.
- Create a new column sales_previous to display the values of the row above a given row.
- Rank the data based on sales in descending order using the RANK function.
- Use common SQL commands and aggregate functions to show the monthly and daily sales averages.
- Analyze discounts on two consecutive days.
- Evaluate moving averages using the window functions.