--1) List all sales with product names and sale amounts.

SELECT p.Product AS ProductName, s.Amount

FROM sales\_table s

JOIN products p ON s.ProductID = p.ProductID

where status = 'Sold';

A screenshot of a computer

AI-generated content may be incorrect.

--2) Show all unique customers who made purchases in January 2019.

SELECT DISTINCT Id

FROM Sales\_table

WHERE Date >= '2019-01-01' AND Date < '2019-02-01';

A screenshot of a computer

AI-generated content may be incorrect.

--3) Get the number of products sold per store.

SELECT StoreID, SUM(Unit) AS TotalProductsSold

FROM Sales\_table

GROUP BY StoreID;

A screenshot of a computer

AI-generated content may be incorrect.

--4) Retrieve a list of products with prices greater than $70.

select product, Price

from Products

where Price > 70;

A screenshot of a computer

AI-generated content may be incorrect.

--5) Count how many customers belong to each age group.

SELECT Age, COUNT(\*) AS CustomerCount

FROM customers

GROUP BY Age

order by CustomerCount desc;

A screenshot of a computer

AI-generated content may be incorrect.

--6) Calculate the total revenue per product category.

SELECT p.Category AS CategoryName, sum(s.Amount)

FROM sales\_table s

JOIN products p ON s.ProductID = p.ProductID

where status = 'Sold'

group by p.Category

A screenshot of a computer

AI-generated content may be incorrect.

--7) Find the top 5 products by number of units sold.

SELECT TOP 5 p.Product AS ProductName, SUM(s.Unit) AS Units\_Sold

FROM sales\_table s

JOIN products p ON s.ProductID = p.ProductID

WHERE s.Status = 'Sold'

group by p.Product

ORDER BY Units\_Sold DESC;

A screenshot of a computer

AI-generated content may be incorrect.

--8) Show the average sales amount per store.

select s.StoreID, AvG(Amount) AS AverageSalesAmount

FROM sales\_table s

WHERE s.Status = 'Sold'

group by s.StoreID

order by AverageSalesAmount;

A screenshot of a computer

AI-generated content may be incorrect.

--9) Identify which stores have sold more than 500 units in total.

select st.storeID,s.Store,sum(st.Unit) AS Units\_Sold

FROM sales\_table st

JOIN Stores s ON st.storeID = s.storeID

WHERE Status = 'Sold'

GROUP BY st.StoreID,s.Store

having sum(st.Unit) > 500

order by Units\_Sold desc

A screenshot of a computer

AI-generated content may be incorrect.

--10) List customers who made more than one purchase.

SELECT ID, COUNT(\*) AS PurchaseCount

FROM Sales\_table

GROUP BY ID

HAVING COUNT(\*) > 1;

A table of numbers

AI-generated content may be incorrect.

--11) Find stores that failed to meet their net sales goals.

SELECT st.StoreID, Net\_Sales\_Goal AS SalesGoal, SUM(Amount) AS NetSales

FROM Sales\_table st

JOIN storegoals g ON st.StoreID = g.StoreID

WHERE st.Status = 'Sold'

GROUP BY st.StoreID, g.Net\_Sales\_Goal

HAVING SUM(st.Amount) < g.Net\_Sales\_Goal;

--12) Identify the age groups that generated the highest total revenue.

SELECT c.Age, SUM(s.Amount) AS TotalRevenue

FROM sales\_table s

JOIN Customers c ON s.ID = c.ID

WHERE s.Status = 'Sold'

GROUP BY c.Age

ORDER BY TotalRevenue DESC;

A screenshot of a computer

AI-generated content may be incorrect.

--13) Show product segments with average prices above the overall average.

SELECT Segment, AVG(Price) AS AverageSegmentPrice

FROM products

GROUP BY Segment

HAVING AVG(Price) > (SELECT AVG(Price) FROM products);

A screenshot of a computer

AI-generated content may be incorrect.

--14) Find customers who bought products from at least 3 different categories.

SELECT s.ID, COUNT(DISTINCT p.Category) AS CategoryCount

FROM Sales\_table s

JOIN products p ON s.ProductID = p.ProductID

WHERE s.Status = 'Sold'

GROUP BY s.ID

HAVING COUNT(DISTINCT p.Category) >= 3;

A screenshot of a computer

AI-generated content may be incorrect.

--15) Calculate the sales achievement % per store region.

SELECT sg.Store\_Region AS StoreRegion, SUM(s.Amount) AS TotalSales, SUM(sg.Net\_Sales\_Goal) AS TotalGoal,

ROUND(SUM(s.Amount) \* 100.0 / NULLIF(SUM(sg.Net\_Sales\_Goal), 0), 2) AS SalesAchievementPercent

FROM Sales\_table s

JOIN storegoals sg ON s.StoreID = sg.StoreID

WHERE s.Status = 'Sold'

GROUP BY sg.Store\_Region;

A screenshot of a computer screen

AI-generated content may be incorrect.

--16) . Identify stores with declining weekly sales trends.

WITH WeeklySales AS (

SELECT StoreID, DATEPART(YEAR, Date) AS SalesYear, DATEPART(WEEK, Date) AS SalesWeek,

SUM(Amount) AS WeeklyAmount

FROM Sales\_table

WHERE Status = 'Sold'

GROUP BY StoreID, DATEPART(YEAR, Date), DATEPART(WEEK, Date)

)

, Trend AS (

SELECT StoreID, SalesYear, SalesWeek, WeeklyAmount,

LAG(WeeklyAmount) OVER (PARTITION BY StoreID ORDER BY SalesYear, SalesWeek) AS PrevWeekAmount

FROM WeeklySales

)

SELECT

StoreID, COUNT(\*) AS DeclineWeeks

FROM Trend

WHERE PrevWeekAmount IS NOT NULL AND WeeklyAmount < PrevWeekAmount

GROUP BY StoreID

HAVING COUNT(\*) >= 2;

A screenshot of a computer

AI-generated content may be incorrect.

--17) Determine which product segment contributes the most to total sales revenue.

SELECT p.Segment, SUM(s.Amount) AS TotalRevenue

FROM Sales\_table s

JOIN products p ON s.ProductID = p.ProductID

WHERE s.Status = 'Sold'

GROUP BY p.Segment

ORDER BY TotalRevenue DESC;

A screenshot of a computer

AI-generated content may be incorrect.

--18) List the top 3 performing stores in each region based on revenue.

SELECT s.StoreID, sg.Store\_Region, SUM(s.Amount) AS TotalRevenue

FROM Sales\_table s

JOIN storegoals sg ON s.StoreID = sg.StoreID

WHERE s.Status = 'Sold'

GROUP BY s.StoreID, sg.Store\_Region

order by sg.Store\_Region,TotalRevenue desc

A screenshot of a data

AI-generated content may be incorrect.

--19) Find the number of unique customers served per store.

SELECT StoreID, COUNT(DISTINCT ID) AS UniqueCustomers

FROM Sales\_table

WHERE Status = 'Sold'

GROUP BY StoreID

ORDER BY UniqueCustomers DESC;

A screenshot of a computer

AI-generated content may be incorrect.

--20) Retrieve sales data where the product belongs to a category starting with 'Office'.

SELECT s.\*

FROM Sales\_table s

JOIN products p ON s.ProductID = p.ProductID

WHERE p.Category LIKE 'Office%' AND s.Status = 'Sold';

A screenshot of a computer

AI-generated content may be incorrect.