**PROJECT – SHOPFAST ECOMMERCE SQL ANALYSIS SCENARIO**

📅 – 02nd June 2025

DB CREATION

CREATE DATABASE ShopFast;

GO

USE ShopFast;

CSV File Import

A screenshot of a computer

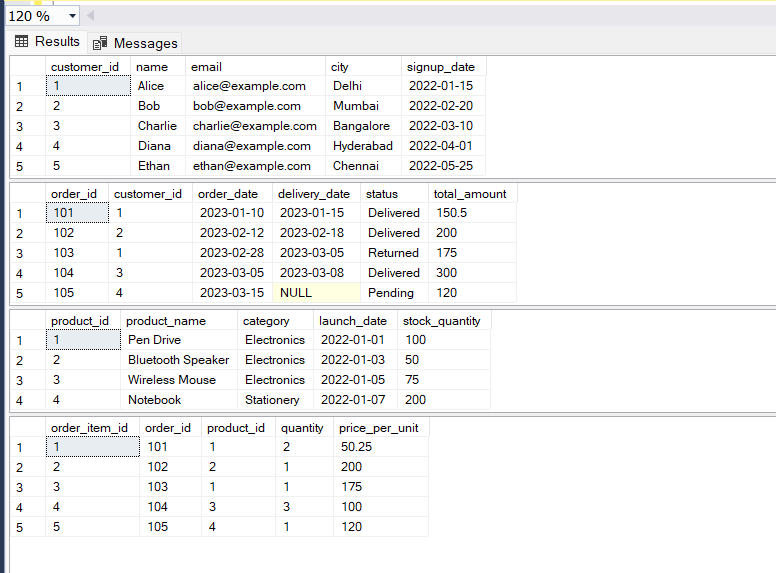
AI-generated content may be incorrect.

Imported tables in MSSQL Server

SELECT \* FROM CUSTOMERS;

SELECT \* FROM ORDERS;

SELECT \* FROM PRODUCTS;

SELECT \* FROM ORDER\_ITEMS;

A screenshot of a computer

AI-generated content may be incorrect.

SQL QUERIES FOR PROBLEM SET

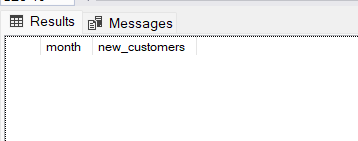
--1. Customer Sign-up Trend: New customers per month (last 12 months)

SELECT FORMAT(signup\_date, 'yyyy-MM') AS month, COUNT(\*) AS new\_customers

FROM CUSTOMERS

WHERE signup\_date >= DATEADD(MONTH, -12, year(2023)) --getdate()

GROUP BY FORMAT(signup\_date, 'yyyy-MM')

ORDER BY month;

A screenshot of a computer

AI-generated content may be incorrect.

Output 1 – With 2023 Year Output 2 – With GETDATE() Curretnt year 2025

--2. Top 5 Customers by Revenue

SELECT TOP 5 c.customer\_id, c.name, COUNT(o.order\_id) AS total\_orders,

SUM(o.total\_amount) AS total\_revenue,

AVG(o.total\_amount) AS avg\_order\_value

FROM CUSTOMERS c

JOIN ORDERS o ON c.customer\_id = o.customer\_id

GROUP BY c.customer\_id, c.name

ORDER BY total\_revenue DESC;

A screenshot of a computer

AI-generated content may be incorrect.Output -

--3. Order Status Distribution

SELECT status, COUNT(\*) AS order\_count

FROM ORDERS

GROUP BY status;

A screenshot of a computer

AI-generated content may be incorrect.Ouput –

--4. Revenue by Category

SELECT p.category, SUM(oi.quantity \* oi.price\_per\_unit) AS total\_revenue

FROM ORDER\_ITEMS oi

JOIN PRODUCTS p ON oi.product\_id = p.product\_id

GROUP BY p.category

ORDER BY total\_revenue DESC;

A screenshot of a computer

AI-generated content may be incorrect.Output –

--5. Best-Selling Products (Top 5 by quantity sold)

SELECT TOP 5 p.product\_name, SUM(oi.quantity) AS total\_sold

FROM ORDER\_ITEMS oi

JOIN PRODUCTS p ON oi.product\_id = p.product\_id

GROUP BY p.product\_name

ORDER BY total\_sold DESC;

Output –

A screenshot of a computer

AI-generated content may be incorrect.

--6. Low-Stock Products (<10% stock using CASE)

SELECT product\_id, product\_name, stock\_quantity,

CASE

WHEN stock\_quantity < 10 THEN 'Critical Low'

WHEN stock\_quantity BETWEEN 10 AND 50 THEN 'Low'

ELSE 'Sufficient'

END AS stock\_status

FROM PRODUCTS;

A screenshot of a computer

AI-generated content may be incorrect.Output –

--7. Avg Delivery Time per Month

SELECT FORMAT(order\_date, 'yyyy-MM') AS order\_month,

AVG(DATEDIFF(DAY, order\_date, delivery\_date)) AS avg\_delivery\_days

FROM ORDERS

WHERE delivery\_date IS NOT NULL

GROUP BY FORMAT(order\_date, 'yyyy-MM')

ORDER BY order\_month;

A screenshot of a computer

AI-generated content may be incorrect.Output –

--8. Orders with Delivery >7 days

SELECT \*

FROM ORDERS

WHERE DATEDIFF(DAY, order\_date, delivery\_date) > 7;

A screenshot of a computer

AI-generated content may be incorrect.Output (No orders with 7 days date difference)-

--9. Repeat Customers

SELECT customer\_id, COUNT(\*) AS order\_count

FROM ORDERS

GROUP BY customer\_id

HAVING COUNT(\*) > 1;

A screenshot of a computer

AI-generated content may be incorrect.Output –

--10. Monthly Revenue Growth with LAG()

WITH MonthlyRevenue AS (

SELECT FORMAT(order\_date, 'yyyy-MM') AS order\_month,

SUM(total\_amount) AS revenue

FROM ORDERS

GROUP BY FORMAT(order\_date, 'yyyy-MM')

)

SELECT order\_month, revenue,

revenue - LAG(revenue) OVER (ORDER BY order\_month) AS revenue\_growth

FROM MonthlyRevenue;

A screenshot of a computer

AI-generated content may be incorrect.Output –

--11. Cohort Analysis using CTE (signup year)

WITH Cohorts AS (

SELECT customer\_id, YEAR(signup\_date) AS signup\_year

FROM CUSTOMERS

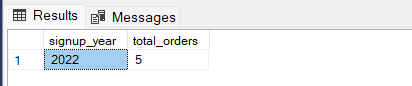
)

SELECT c.signup\_year, COUNT(o.order\_id) AS total\_orders

FROM Cohorts c

JOIN ORDERS o ON c.customer\_id = o.customer\_id

GROUP BY c.signup\_year;

Output –

--12. Cancelled/Returned Product Revenue Loss

SELECT status, SUM(total\_amount) AS revenue\_loss

FROM ORDERS

WHERE status IN ('Cancelled', 'Returned')

GROUP BY status;

A screenshot of a message

AI-generated content may be incorrect.Output –

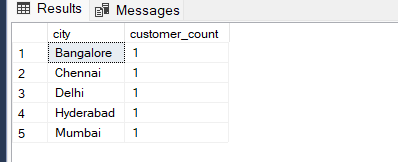
--13. Customer City Heatmap

SELECT city, COUNT(\*) AS customer\_count

FROM CUSTOMERS

GROUP BY city

ORDER BY customer\_count DESC;

Output –

--14. First & Last Order per Customer using ROW\_NUMBER()

WITH OrderedData AS (

SELECT customer\_id, order\_id, order\_date,

ROW\_NUMBER() OVER (PARTITION BY customer\_id ORDER BY order\_date ASC) AS rn\_asc,

ROW\_NUMBER() OVER (PARTITION BY customer\_id ORDER BY order\_date DESC) AS rn\_desc

FROM ORDERS

)

SELECT customer\_id, order\_id, order\_date, 'First Order' AS order\_type

FROM OrderedData WHERE rn\_asc = 1

UNION ALL

SELECT customer\_id, order\_id, order\_date, 'Last Order' AS order\_type

FROM OrderedData WHERE rn\_desc = 1

ORDER BY customer\_id, order\_type;

A screenshot of a computer

AI-generated content may be incorrect.Output –

-- 15. NULL Handling: Orders with missing delivery date or total amount

SELECT order\_id, customer\_id, order\_date, delivery\_date, total\_amount,

CASE WHEN delivery\_date IS NULL THEN 'Missing Delivery Date' ELSE '' END AS delivery\_issue,

CASE WHEN total\_amount IS NULL THEN 'Missing Amount' ELSE '' END AS amount\_issue

FROM ORDERS

WHERE delivery\_date IS NULL OR total\_amount IS NULL;

Output –

A screenshot of a computer

AI-generated content may be incorrect.