**1. Customer Segmentation By Gender**

SELECT c.Gender, COUNT(\*) as gender\_count

FROM customers c

GROUP BY c.Gender;

**2. Customer Segmentation By Age Group**

SELECT c.AgeRange, COUNT(CustomerKey)

FROM customers c

GROUP BY c.AgeRange;

**3. Customer Locations**

SELECT c.Cust\_State, COUNT(CustomerKey) as cust\_count

FROM customers c

GROUP BY c.Cust\_Country, c.Cust\_State;

**4. Customer by Order Frequency**

WITH CustomerOrders AS (

SELECT s.CustomerKey, COUNT(s.OrderNumber) AS OrderCount

FROM

Sales s

GROUP BY

s.CustomerKey

)

SELECT

c.CustomerKey,

COALESCE(co.OrderCount, 0) AS OrderCount,

CASE

WHEN COALESCE(co.OrderCount, 0) > 10 THEN 'Loyal Customer'

WHEN COALESCE(co.OrderCount, 0) BETWEEN 5 AND 10 THEN 'Periodic Buyer'

WHEN COALESCE(co.OrderCount, 0) BETWEEN 1 AND 4 THEN 'VIP Customer'

ELSE 'No Orders'

END AS CustomerOrderingType

FROM

Customers c

LEFT JOIN

CustomerOrders co ON c.CustomerKey = co.CustomerKey;

**5. Revenue and Profit for each Customer**

WITH CustomerRevenue AS (

SELECT

s.CustomerKey,

SUM(p.UnitPriceUSD \* s.Quantity) AS TotalRevenue,

SUM((p.UnitPriceUSD - p.UnitCostUSD)\* s.Quantity) As TotalProfit

FROM

Sales s

JOIN

Products p ON s.ProductKey = p.ProductKey

GROUP BY

s.CustomerKey

)

SELECT

c.CustomerKey,

COALESCE(cr.TotalRevenue, 0) AS TotalRevenue,

COALESCE(cr.TotalProfit, 0) AS TotalProfit

FROM

Customers c

LEFT JOIN

CustomerRevenue cr ON c.CustomerKey = cr.CustomerKey;

**6. Order Volume By Order Date**

SELECT s.OrderDate, SUM(s.Quantity)

FROM sales s

GROUP BY s.OrderDate;

**7. Stores with Zero Orders**

SELECT s.StoreKey

FROM stores s

LEFT JOIN Sales sa ON s.StoreKey = sa.StoreKey

WHERE sa.StoreKey IS NULL;

**8. Products with Zero Sales**

SELECT \*

FROM Products

WHERE ProductKey NOT IN (SELECT ProductKey FROM Sales);

**9. Sales Revenue from Stores**

WITH SalesData AS (

SELECT

s.StoreKey,

SUM(p.UnitPriceUSD \* s.Quantity) AS TotalRevenue

FROM

sales s

JOIN

products p ON s.ProductKey = p.ProductKey

GROUP BY

s.StoreKey

)

SELECT

s.StoreKey,

TotalRevenue

FROM

SalesData s

JOIN

stores st ON s.StoreKey = st.StoreKey

ORDER BY

TotalRevenue DESC;

**10. Top 10 Brands, Products**

SELECT p.ProductName, p.Brand, p.Category, c.Cust\_City, c.Cust\_State, c.AgeRange, c.Gender,

SUM(s.Quantity) AS Total\_Quantity, SUM(s.Quantity \* p.UnitPriceUSD) AS Total\_Revenue

FROM Sales s

JOIN Products p ON s.ProductKey = p.ProductKey

JOIN Customers c ON s.CustomerKey = c.CustomerKey

JOIN Stores st ON s.StoreKey = st.StoreKey

GROUP BY p.ProductName, p.Brand, p.Category, c.Cust\_City, c.Cust\_State, c.AgeRange, c.Gender

limit 10;

**11. Top 10 Customer Cities**

SELECT

Cust\_City,

COUNT(\*) AS Customer\_Count,

(SELECT COUNT(CustomerKey) FROM customers) AS All\_Customers,

(SELECT count(distinct(Cust\_City)) FROM Customers) AS All\_Cities,

(COUNT(\*) \* 100.0) / (SELECT COUNT(\*) FROM Customers) AS Percentage

FROM Customers

GROUP BY Cust\_City

ORDER BY Customer\_Count DESC

LIMIT 10;

**12.Customer CLV**

SELECT CustomerKey, SUM(Sales.TotalRevenue) / COUNT(DISTINCT Sales.OrderNumber) AS CLV

FROM Sales

GROUP BY CustomerKey;

**13.Customer Churn Rate**

WITH CustomerOrders AS (

SELECT CustomerKey, MAX(OrderDate) AS LastOrderDate

FROM Sales

GROUP BY CustomerKey

)

SELECT CustomerKey,

CASE WHEN LastOrderDate < DATEADD(year, -1, GETDATE()) THEN 'Churned' ELSE 'Active' END AS ChurnStatus

FROM CustomerOrders;

**14.Return Rate by Category**:

SELECT p.Category,

SUM(e.SumOfExchange) / SUM(s.Quantity) AS ReturnRate

FROM GESales s

JOIN GEExchange e ON s.ProductKey = e.ProductKey

JOIN GEProducts p ON s.ProductKey = p.ProductKey

GROUP BY p.Category;

**15.Total Revenue by Order Type (Online vs In-Store)**:

SELECT CASE

WHEN StoreMeters = 0 THEN 'Online'

ELSE 'In-Store'

END AS OrderType,

SUM(TotalRevenue) AS TotalRevenue

FROM Sales

JOIN Stores ON Sales.StoreKey = Stores.StoreKey

GROUP BY CASE

WHEN StoreMeters = 0 THEN 'Online'

ELSE 'In-Store'

END;

**16.AOV (Average Order Value)**:

SELECT OrderNumber, SUM(TotalRevenue) / COUNT(OrderNumber) AS AOV

FROM Sales

GROUP BY OrderNumber;