1. **List all products with their total sales, average ratings, and a flag indicating if the warehouse they are stored in are discontinued:**

SELECT

p.ProductName,

SUM(ODTotalValue) AS TotalSales,

AVG(Rating) AS AvgRating,

CASE WHEN w1.Discontinued = 1 THEN 'Discontinued'

ELSE 'Not discontinued' END AS DiscontinuedStatus

FROM PRODUCT p

JOIN ORDERDETAILS od ON p.ProductID = od.ProductID

JOIN ORDERS o ON od.OrderID = o.OrderID

JOIN WAREHOUSINGDETAILS w ON p.ProductID = w.ProductId

JOIN WAREHOUSE w1 ON w.warehouseid = w1.warehouseid

JOIN REVIEWDETAILS rd ON p.ProductID = rd.ProductID

GROUP BY p.ProductName, w1.discontinued, ODTotalValue, Rating;

1. **total sales value generated by each sales representative and identifies the top-performing representatives based on their sales performance:**

WITH SalesData AS (

SELECT

s.EmployeeID,

e.FirstName || ' ' || e.LastName AS SalesRepresentative,

SUM(o.OrderTotalValue) AS TotalSalesValue

FROM

OFFLINEORDER s

JOIN

Orders o ON s.OrderID = o.OrderID

JOIN

SALESREPRESENTATIVE sr ON s.EmployeeID = sr.EmployeeID

JOIN

EMPLOYEE e ON sr.EmployeeID = e.EmployeeID

GROUP BY

s.EmployeeID, e.FirstName, e.LastName

)

SELECT

EmployeeID,

SalesRepresentative,

TotalSalesValue

FROM

SalesData

ORDER BY

TotalSalesValue DESC;

1. **Find the details of the carrier, no. of shipments they are responsible for, date of next shipment, days between 2 shipments** :

WITH ShipmentCTE AS (

SELECT

carrierid,

TO\_DATE(ShipmentDate, 'DD-MON-YYYY') AS ShipmentDate,

LEAD(TO\_DATE(ShipmentDate, 'DD-MON-YYYY')) OVER (PARTITION BY carrierid ORDER BY TO\_DATE(ShipmentDate, 'DD-MON-YYYY')) AS NextShipmentDate

FROM

SHIPMENT

)

SELECT

c.CarrierName,

COUNT(s.ShipmentID) AS ShipmentCount,min(s.ShipmentDate) AS CURRENT\_SHIPMENT\_DATE,

TO\_CHAR(MAX(sc.NextShipmentDate), 'DD-MON-YYYY') AS Next\_Shipment\_Date,

COALESCE(TO\_CHAR((MAX(sc.NextShipmentDate) - MIN(TO\_DATE(s.ShipmentDate, 'DD-MON-YYYY'))), '99999'), '') AS Days\_Between\_Shipments

FROM

ShipmentCTE sc

JOIN

carrier c ON sc.carrierid = c.carrierid

LEFT JOIN

SHIPMENT s ON sc.carrierid = s.carrierid AND sc.NextShipmentDate = TO\_DATE(s.ShipmentDate, 'DD-MON-YYYY')

GROUP BY

c.CarrierName

ORDER BY

c.CarrierName;

1. **List the top 3 warehouses with the highest total shipment weights and their respective carriers:**

SELECT w.WarehouseId, w.WarehouseName, c.CarrierName, SUM(s.ShipmentWeight) AS TotalShipmentWeight

FROM WAREHOUSE w

JOIN SHIPMENT s ON w.WarehouseId = s.WarehouseID

JOIN CARRIER c ON s.CarrierID = c.CarrierID

GROUP BY w.WarehouseId, w.WarehouseName, c.CarrierName

ORDER BY TotalShipmentWeight DESC

FETCH FIRST 3 ROWS ONLY;

1. **Identify Customers with Unusual Order Frequency:**

SELECT

c.CustomerID,

c.CustomerName,

COUNT(o.OrderID) AS TotalOrders,

CASE

WHEN COUNT(o.OrderID) < AVG(COUNT(o.OrderID)) OVER () - STDDEV(COUNT(o.OrderID)) OVER () THEN 'Low Order Frequency'

WHEN COUNT(o.OrderID) > AVG(COUNT(o.OrderID)) OVER () + STDDEV(COUNT(o.OrderID)) OVER () THEN 'High Order Frequency'

ELSE 'Normal Order Frequency'

END AS OrderFrequencyStatus

FROM

CUSTOMER c

JOIN

ORDERS o ON c.CustomerID = o.CustomerID

GROUP BY

c.CustomerID, c.CustomerName

ORDER BY

TotalOrders DESC;

1. **Retrieve a list of products, their categories, and the number of orders placed for each product:**

SELECT

P.ProductID,

P.ProductName,

C.CategoryName,

S.SubCategoryName,

COUNT(OD.OrderID) AS NumberOfOrders

FROM

PRODUCT P

JOIN

SUBCATEGORY S ON P.SubCategoryID = S.SubCategoryID

JOIN

CATEGORY C ON S.CategoryID = C.CategoryID

LEFT JOIN

ORDERDETAILS OD ON P.ProductID = OD.ProductID

GROUP BY

P.ProductID, P.ProductName, C.CategoryName, S.SubCategoryName

ORDER BY

NumberOfOrders DESC;

1. **Calculate the percentage of units in stock compared to the capacity for each warehouse**:

SELECT w.WarehouseId, w.WarehouseName, round(((SUM(w.UnitsInStock) / w.Capacity) \* 100),2) AS StockPercentage

FROM WAREHOUSE w

GROUP BY w.WarehouseId, w.WarehouseName, w.Capacity;

1. **Find the customers who have exceeded their credit limit:**

SELECT c.CustomerID, c.CustomerName, CreditLimit, SUM(OrderTotalValue) AS TotalOrders

FROM CUSTOMER c

JOIN ORDERS o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, CustomerName, CreditLimit

HAVING SUM(OrderTotalValue) > CreditLimit;

1. **List the top 3 products with the highest average customer ratings:**

SELECT p.ProductID, p.ProductName, AVG(r.Rating) AS AvgRating

FROM PRODUCT p

JOIN REVIEWDETAILS r ON p.ProductID = r.ProductID

GROUP BY p.ProductID, p.ProductName

ORDER BY AvgRating DESC

FETCH FIRST 3 ROWS ONLY;

1. **Find the top 5 suppliers with the highest total supply order values:**

SELECT s.SupplierID, s.SupplierName, SUM(o.TotalValue) AS TotalOrderValue

FROM SUPPLIER s

JOIN SUPPLYORDER o ON s.SupplierID = o.SupplierID

GROUP BY s.SupplierID, s.SupplierName

ORDER BY TotalOrderValue DESC

FETCH FIRST 5 ROWS ONLY;

1. **Find Products with Fluctuating Demand**

WITH MonthlyProductSales AS (

SELECT

p.ProductID,

p.ProductName,

TO\_CHAR(od.OrderDate, 'YYYY-MM') AS OrderMonth,

SUM(od.OrderItems) AS MonthlySales

FROM

PRODUCT p

LEFT JOIN

ORDERDETAILS od ON p.ProductID = od.ProductID

GROUP BY

p.ProductID, p.ProductName, TO\_CHAR(od.OrderDate, 'YYYY-MM')

),

SalesFluctuationCTE AS (

SELECT

ProductID,

ProductName,

OrderMonth,

MonthlySales,

LAG(MonthlySales) OVER (PARTITION BY ProductID ORDER BY OrderMonth) AS PreviousMonthSales,

CASE

WHEN LAG(MonthlySales) OVER (PARTITION BY ProductID ORDER BY OrderMonth) IS NOT NULL

THEN MonthlySales - LAG(MonthlySales) OVER (PARTITION BY ProductID ORDER BY OrderMonth)

ELSE NULL

END AS SalesFluctuation

FROM

MonthlyProductSales

)

SELECT

ProductID,

ProductName,

OrderMonth,

MonthlySales,

PreviousMonthSales,

SalesFluctuation

FROM

SalesFluctuationCTE

WHERE

SalesFluctuation IS NOT NULL

ORDER BY

ProductID, OrderMonth;

1. **Retrieve a list of products and their total sales quantity, grouped by subcategory:**

WITH RankedProducts AS (  
 SELECT  
 p.ProductID,  
 p.ProductName,  
 c.CategoryName,  
 od.OrderItems,  
 RANK() OVER (PARTITION BY c.CategoryID ORDER BY od.OrderItems DESC) AS SalesRank  
 FROM  
 PRODUCT p  
 JOIN  
 SUBCATEGORY sc ON p.SubCategoryID = sc.SubCategoryID  
 JOIN  
 CATEGORY c ON sc.CategoryID = c.CategoryID  
 JOIN  
 ORDERDETAILS od ON p.ProductID = od.ProductID  
 )

SELECT  
 ProductID,  
 ProductName,  
 CategoryName,  
 OrderItems  
 FROM  
 RankedProducts  
 WHERE  
 SalesRank = 1;