Beginner level

1. Retrieve Customer Details

Question:

Write an SQL query to retrieve the CustomerID, AccountNumber, and CustomerType for all customers who have a TerritoryID of 5.

SELECT CustomerID, AccountNumber, CustomerType

FROM customer

WHERE TerritoryID = 5;

2. Count Products in Inventory

Question:

Write a query to count the total number of distinct ProductID entries in the productinventory table.

SELECT COUNT(DISTINCT ProductID) AS TotalDistinctProducts

FROM productinventory;

3. Calculate Total Sales Amount

Question:

Write an SQL query to calculate the total sales amount (LineTotal) from the sales order detail table.

SELECT SUM(LineTotal) AS TotalSalesAmount

FROM salesorderdetail;

4. Find Expired Credit Cards

Question:

Write an SQL query to list all CreditCardID entries from the creditcard table where the ExpYear is less than the current year.

SELECT CreditCardID

FROM creditcard

WHERE ExpYear < YEAR(CURRENT_DATE);

5. Join Customers and Orders

Question:

"How would you write an SQL query to calculate the total sales (SubTotal) for each customer in the salesorderheader table, displaying the CustomerID alongside their TotalSubTotal?"

SELECT CustomerID, SUM(SubTotal) AS TotalSubTotal

FROM salesorderheader

GROUP BY CustomerID;

6. Filter High-Rated Products

Question:

Write an SQL query to fetch ProductID, Rating, and ReviewerName from the productreview table where the Rating is 4 or higher.

SELECT ProductID, Rating, ReviewerName

FROM productreview

WHERE	Rating	>=	4
-------	--------	----	---

7. Analyze Product Categories

Question:

Write an SQL query to list the Name and ProductCategoryID from the productcategory table, ordered alphabetically by Name.

SELECT Name, ProductCategoryID

FROM productcategory

ORDER BY Name ASC;

8. Identify Orders Pending Shipment

Question:

Write an SQL query to find all SalesOrderID entries in the salesorderheader table where the ShipDate is NULL.

SELECT SalesOrderID

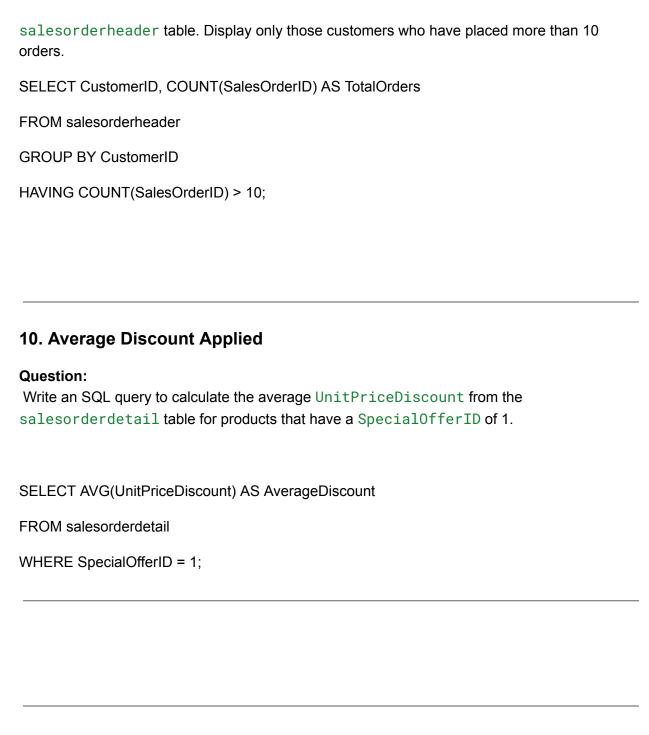
FROM salesorderheader

WHERE ShipDate IS NULL;

9. Find Frequent Customers

Question:

Write an SQL query to count the number of orders placed by each CustomerID in the



11. Retrieve customer details who placed orders in the last 6 months.

Use salesorderheader to filter orders by OrderDate.

SELECT DISTINCT CustomerID FROM salesorderheader WHERE OrderDate >= DATEADD(MONTH, -6, GETDATE());

12. Identify the top 5 products with the highest total sales revenue.

Aggregate UnitPrice * OrderQty from salesorderdetail.

SELECT TOP 5 ProductID, SUM(UnitPrice * OrderQty) AS TotalRevenue FROM salesorderdetail GROUP BY ProductID ORDER BY TotalRevenue DESC;

13. Find customers who have purchased products from more than 3 different categories.

Use productsubcateg and group purchases by CustomerID.

SELECT CustomerID
FROM salesorderdetail sd
JOIN product p ON sd.ProductID = p.ProductID
JOIN productsubcategory ps ON p.ProductSubcategoryID = ps.ProductSubcategoryID
GROUP BY CustomerID
HAVING COUNT(DISTINCT ps.ProductSubcategoryID) > 3;

14. List customers who have used more than one payment method.

Join salesorderheader and creditcard to identify customers with multiple CreditCardID entries.

SELECT CustomerID

FROM salesorderheader

GROUP BY CustomerID

HAVING COUNT(DISTINCT CreditCardID) > 1;

15. Retrieve the names of products that were not sold in the current year.

Use product and exclude ProductID present in salesorderdetail.

SELECT p.Name

FROM product p

WHERE p.ProductID NOT IN

16. Calculate the average order value for each customer.

Use salesorderheader and group by CustomerID.

SELECT CustomerID, AVG(TotalDue) AS AverageOrderValue FROM salesorderheader GROUP BY CustomerID;

17. Find the most commonly used shipping method for online orders.

Hint: Query ShipMethodID from salesorderheader where
OnlineOrderFlag = 1.

SELECT ShipMethodID, COUNT(*) AS UsageCount FROM salesorderheader WHERE OnlineOrderFlag = 1 GROUP BY ShipMethodID ORDER BY UsageCount DESC LIMIT 1;

18. Identify customers who placed an order but canceled it later.

Use salesorderheader and look for specific Status codes indicating cancellations.

SELECT CustomerID, SalesOrderID, Status
FROM salesorderheader
WHERE Status = 'Canceled'; -- Replace 'Canceled' with the exact code for canceled status.

19. List employees who processed orders with a total value above \$10,000.

Join salesorderheader and employee and filter by aggregated TotalDue.

SELECT e.EmployeeID, e.FirstName, e.LastName, soh.SalesOrderID, SUM(soh.TotalDue) AS TotalOrderValue
FROM salesorderheader soh
JOIN employee e ON soh.SalesPersonID = e.EmployeeID
GROUP BY e.EmployeeID, e.FirstName, e.LastName, soh.SalesOrderID
HAVING SUM(soh.TotalDue) > 10000;

20. Find the product with the highest reorder point and identify its suppliers.

Hint: Use product, productvendor, and VendorID to determine the relationship.

```
SELECT p.ProductID, p.Name AS ProductName, p.ReorderPoint, v.VendorID, v.Name AS VendorName
FROM product p
JOIN productvendor pv ON p.ProductID = pv.ProductID
JOIN vendor v ON pv.VendorID = v.VendorID
WHERE p.ReorderPoint = (
SELECT MAX(ReorderPoint) FROM product
);
```

21. Identify Sales Trends Over Time

SELECT

YEAR(OrderDate) AS Year,
MONTH(OrderDate) AS Month,
SUM(LineTotal) AS TotalSales

FROM salesorderheader soh

JOIN salesorderdetail sod ON soh.SalesOrderID = sod.SalesOrderID

WHERE YEAR(OrderDate) = 2023

GROUP BY YEAR(OrderDate), MONTH(OrderDate)

ORDER BY Year, Month;

22. Product Sales Analysis

```
p.ProductID,
p.Name AS ProductName,
SUM(sod.UnitPrice * sod.OrderQty) AS TotalSales
FROM product p

JOIN salesorderdetail sod ON p.ProductID = sod.ProductID

GROUP BY p.ProductID, p.Name

ORDER BY TotalSales DESC

LIMIT 5;
```

23. Identify Customer Purchase Behavior

```
SELECT
```

c.CustomerID,

COUNT(soh.SalesOrderID) AS PurchaseCount

FROM customer c

JOIN salesorderheader soh ON c.CustomerID = soh.CustomerID

WHERE YEAR(soh.OrderDate) = YEAR(GETDATE()) - 1

GROUP BY c.CustomerID

HAVING PurchaseCount > 10;

24. Average Order Value by Territory

```
SELECT
```

TerritoryID,

AVG(SubTotal) AS AverageOrderValue

FROM salesorderheader

GROUP BY TerritoryID;

25. High-Spending Customers

SELECT

soh.CustomerID,

SUM(sod.LineTotal) AS TotalPurchases

FROM salesorderheader soh

JOIN salesorderdetail sod ON soh.SalesOrderID = sod.SalesOrderID

GROUP BY soh.CustomerID

HAVING TotalPurchases > 10000;

26. Discount Analysis

SELECT

ProductID,

SUM(UnitPriceDiscount * OrderQty) AS TotalDiscount

FROM salesorderdetail

GROUP BY ProductID;

27. Pending Orders

SELECT

SalesOrderID,

Status,

OrderDate

FROM salesorderheader

WHERE Status = 'Pending';

28. Product Category Sales

SELECT

pc.Name AS CategoryName,

SUM(sod.UnitPrice * sod.OrderQty) AS TotalSales

FROM productcategory pc

JOIN productsubcategory psc ON pc.ProductCategoryID = psc.ProductCategoryID

JOIN product p ON p.ProductSubcategoryID = psc.ProductSubcategoryID

JOIN salesorderdetail sod ON p.ProductID = sod.ProductID

GROUP BY pc.Name;

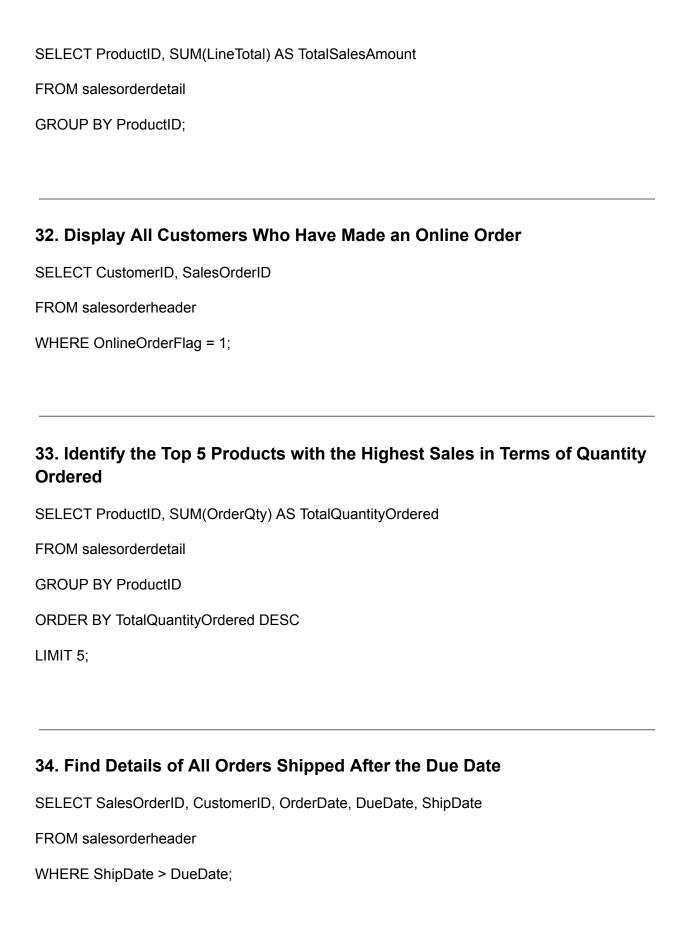
29. Top Customers by Sales Order Count

SELECT

c.CustomerID,

COUNT(soh.SalesOrderID) AS OrderCount
FROM customer c
JOIN salesorderheader soh ON c.CustomerID = soh.CustomerID
GROUP BY c.CustomerID
ORDER BY OrderCount DESC
LIMIT 5;
30. Sales Performance by Salesperson
SELECT
SalesPersonID,
SUM(SubTotal) AS TotalSales
FROM salesorderheader
GROUP BY SalesPersonID;

31. Retrieve the Total Sales Amount for Each Product



35. Retrieve Customers and Their Corresponding Total Spending

SELECT soh.CustomerID, SUM(sod.LineTotal) AS TotalSpending

FROM salesorderheader soh

JOIN salesorderdetail sod ON soh.SalesOrderID = sod.SalesOrderID

GROUP BY soh.CustomerID;

36. Identify All Orders Where the Discount Was Greater Than 10%

SELECT SalesOrderID, ProductID, OrderQty, UnitPrice, UnitPriceDiscount

FROM salesorderdetail

WHERE UnitPriceDiscount > 0.1;

37. Retrieve a List of All Addresses and the Number of Customers Associated with Each Address

SELECT a.AddressID, a.AddressLine1, a.City, COUNT(ca.CustomerID) AS NumberOfCustomers

FROM address a

JOIN customeraddress ca ON a.AddressID = ca.AddressID

GROUP BY a.AddressID, a.AddressLine1, a.City;

38. Determine the Average Shipping Time (in Days) for All Orders

FROM salesorderheader	
WHERE ShipDate IS NOT NULL;	
39. Find the Top 3 Customers Based on Total Number of Transactions	1
SELECT CustomerID, COUNT(SalesOrderID) AS TotalTransactions	
FROM salesorderheader	
GROUP BY CustomerID	
ORDER BY TotalTransactions DESC	
LIMIT 3;	
40. List All Employees Who Have Managed More Than 10 Orders	
SELECT SalesPersonID, COUNT(SalesOrderID) AS ManagedOrders	
FROM salesorderheader	
GROUP BY SalesPersonID	
HAVING COUNT(SalesOrderID) > 10;	

SELECT c.CustomerID, c.FirstName, c.LastName, a.AddressLine1, a.City, a.PostalCode

FROM customer c

JOIN customeraddress ca ON c.CustomerID = ca.CustomerID

JOIN address a ON ca.AddressID = a.AddressID;

42. Find Total Sales Amount for Each Product Category

SELECT pc.ProductCategoryID, pc.Name AS ProductCategoryName, SUM(sod.LineTotal) AS TotalSales

FROM salesorderdetail sod

JOIN product p ON sod.ProductID = p.ProductID

JOIN productcategory pc ON p.ProductCategoryID = pc.ProductCategoryID

GROUP BY pc.ProductCategoryID, pc.Name;

43. Display Customers Who Made Purchases in the Last 30 Days

SELECT c.CustomerID, c.FirstName, c.LastName, soh.OrderDate

FROM customer c

JOIN salesorderheader soh ON c.CustomerID = soh.CustomerID

WHERE soh.OrderDate >= DATEADD(DAY, -30, GETDATE());

44. Identify the Top 5 Products with the Highest Total Revenue

SELECT p.ProductID, p.Name AS ProductName, SUM(sod.UnitPrice * sod.OrderQty) AS TotalRevenue

FROM salesorderdetail sod

JOIN product p ON sod.ProductID = p.ProductID

GROUP BY p.ProductID, p.Name
ORDER BY TotalRevenue DESC
LIMIT 5;

45. Generate a Summary Report of Orders by Year and Month

 ${\tt SELECT\ YEAR} (soh. Order Date)\ AS\ Year,\ MONTH (soh. Order Date)\ AS\ Month,$

COUNT(soh.SalesOrderID) AS TotalOrders, SUM(sod.LineTotal) AS TotalSalesAmount

FROM salesorderheader soh

JOIN salesorderdetail sod ON soh.SalesOrderID = sod.SalesOrderID

GROUP BY YEAR(soh.OrderDate), MONTH(soh.OrderDate)

ORDER BY Year, Month;

46. Find Customers Who Have Not Placed Any Orders

SELECT c.CustomerID, c.FirstName, c.LastName

FROM customer c

LEFT JOIN salesorderheader soh ON c.CustomerID = soh.CustomerID

WHERE soh.SalesOrderID IS NULL;

47. Display the Average Discount Applied for Each Product

SELECT p.ProductID, p.Name AS ProductName, AVG(sod.UnitPriceDiscount) AS AverageDiscount

FROM salesorderdetail sod

JOIN product p ON sod.ProductID = p.ProductID

GROUP BY p.ProductID, p.Name;

48. Total Number of Orders Handled by Each Salesperson

SELECT e.SalesPersonID, e.FirstName, e.LastName, COUNT(soh.SalesOrderID) AS TotalOrders

FROM employee e

JOIN salesorderheader soh ON e.EmployeeID = soh.SalesPersonID

GROUP BY e.SalesPersonID, e.FirstName, e.LastName;

49. Most Frequently Purchased Product for Each Customer

```
WITH RankedProducts AS (
```

SELECT c.CustomerID, c.FirstName, c.LastName, sod.ProductID, p.Name AS ProductName,

ROW_NUMBER() OVER (PARTITION BY c.CustomerID ORDER BY SUM(sod.OrderQty) DESC) AS Rank

FROM customer c

JOIN salesorderheader soh ON c.CustomerID = soh.CustomerID

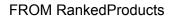
JOIN salesorderdetail sod ON soh.SalesOrderID = sod.SalesOrderID

JOIN product p ON sod.ProductID = p.ProductID

GROUP BY c.CustomerID, c.FirstName, c.LastName, sod.ProductID, p.Name

)

SELECT CustomerID, FirstName, LastName, ProductID, ProductName



WHERE Rank = 1;

50. Products with Stock Levels Below Safety Stock Level

SELECT p.ProductID, p.Name AS ProductName, pi.Quantity, p.SafetyStockLevel

FROM productinventory pi

JOIN product p ON pi.ProductID = p.ProductID

WHERE pi.Quantity < p.SafetyStockLevel;