Use northwind database for these exercises

1. **List Customers with Orders: Find all customers who have placed orders.**

SELECT CustomerID, ContactName

FROM Customers

WHERE CustomerID IN (

SELECT DISTINCT CustomerID

FROM Orders

);

1. **Total Sales per Category: Calculate the total sales for each product category.**

select

categoryname,

(

select

sum(od.unitprice \* od.quantity)

from order\_details as od join products as p using(productid)

where p.categoryid = c.categoryid

) as total\_sales

from categories as c;

1. **List Products with Low Stock: Find products with less than 10 units in stock.**

SELECT ProductName, UnitsInStock

FROM Products

WHERE UnitsInStock < 10;

1. **List Suppliers with No Products: Identify suppliers who do not supply any products.**

SELECT SupplierName

FROM Suppliers

WHERE SupplierID NOT IN (

SELECT DISTINCT SupplierID

FROM Products

);

1. **Customer Orders Count: List customers along with the number of orders they have placed.**

SELECT CustomerID, ContactName, (

SELECT COUNT(\*)

FROM Orders

WHERE Customers.CustomerID = Orders.CustomerID

) AS OrderCount

FROM Customers;

1. **Employee Sales Comparison: Find employees who have sold more than their average sales.**

SELECT E.EmployeeID, E.LastName, E.FirstName

FROM Employees E

WHERE E.EmployeeID IN (

SELECT O.EmployeeID

FROM Orders O

JOIN Order\_Details OD ON O.OrderID = OD.OrderID

GROUP BY O.EmployeeID

HAVING SUM(OD.UnitPrice \* OD.Quantity) > (

SELECT AVG(TotalSales)

FROM (

SELECT SUM(OD.UnitPrice \* OD.Quantity) AS TotalSales

FROM Orders O

JOIN Order\_Details OD ON O.OrderID = OD.OrderID

GROUP BY O.EmployeeID

) AS SalesData

)

);

1. **Product Sales Ranking: Find the top 5 products with the highest total sales.**

SELECT ProductName, TotalSales

FROM (

SELECT ProductName, SUM(Quantity \* UnitPrice) AS TotalSales

FROM Products

JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID

GROUP BY ProductName

ORDER BY TotalSales DESC

LIMIT 5

) AS TopProducts;

1. **Employee Sales Totals: List employees along with their total sales amount.**

SELECT EmployeeID, CONCAT(FirstName, ' ', LastName) AS EmployeeName, (

SELECT SUM(Quantity \* UnitPrice)

FROM Orders

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

WHERE Orders.EmployeeID = Employees.EmployeeID

) AS TotalSales

FROM Employees;

1. **Customers with High Orders: Find customers who have placed orders with a total value higher than $5,000.**

SELECT CustomerID, ContactName

FROM Customers

WHERE CustomerID IN (

SELECT CustomerID

FROM Orders

GROUP BY CustomerID

HAVING SUM(Quantity \* UnitPrice) > 5000

);

1. **Supplier with the Most Products: Identify the supplier who supplies the most products.**

SELECT SupplierName

FROM Suppliers

WHERE SupplierID = (

SELECT TOP 1 SupplierID

FROM Products

GROUP BY SupplierID

ORDER BY COUNT(\*) DESC

);

1. **Orders with Multiple Products: List orders that contain more than 5 different products.**

SELECT OrderID, COUNT(DISTINCT ProductID) AS ProductCount

FROM OrderDetails

GROUP BY OrderID

HAVING ProductCount > 5;

1. **Category with Highest Average Price: Find the category with the highest average product price.**

SELECT CategoryName

FROM Categories

WHERE CategoryID = (

SELECT CategoryID

FROM Products

GROUP BY CategoryID

HAVING AVG(UnitPrice) = (

SELECT MAX(avg\_price)

FROM (

SELECT CategoryID, AVG(UnitPrice) AS avg\_price

FROM Products

GROUP BY CategoryID

) AS AvgPrices

)

);

1. **Employees with No Orders: Identify employees who have not processed any orders.**

SELECT EmployeeID, CONCAT(FirstName, ' ', LastName) AS EmployeeName

FROM Employees

WHERE EmployeeID NOT IN (

SELECT DISTINCT EmployeeID

FROM Orders

);

1. **Customer with the Highest Order Value: Find the customer with the highest total order value.**

SELECT CustomerID, ContactName, MAX(TotalOrderValue) AS HighestOrderValue

FROM (

SELECT CustomerID, ContactName, SUM(Quantity \* UnitPrice) AS TotalOrderValue

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

GROUP BY CustomerID

) AS CustomerOrderTotals;

1. **Products Not Ordered: List products that have not been ordered by any customer.**

SELECT ProductName

FROM Products

WHERE ProductID NOT IN (

SELECT DISTINCT ProductID

FROM OrderDetails

);

1. **Employee Sales Growth: Find employees who have had an increase in total sales from the previous year.**

SELECT EmployeeID, CONCAT(FirstName, ' ', LastName) AS EmployeeName

FROM (

SELECT EmployeeID, SUM(Quantity \* UnitPrice) AS TotalSales

FROM Orders

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

WHERE YEAR(OrderDate) = YEAR(CURRENT\_DATE)

GROUP BY EmployeeID

) AS CurrentYearSales

WHERE TotalSales > (

SELECT SUM(Quantity \* UnitPrice) AS TotalSales

FROM Orders

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

WHERE YEAR(OrderDate) = YEAR(CURRENT\_DATE) - 1

GROUP BY EmployeeID

);

1. **Products with Prices Higher Than Average: List all products with a price higher than the average product price.**

SELECT ProductName

FROM Products

WHERE UnitPrice > (SELECT AVG(UnitPrice) FROM Products);

1. **Employees with Same Manager: Find employees who have the same manager**.

SELECT EmployeeID, LastName, FirstName

FROM Employees

WHERE ReportsTo IN (SELECT ReportsTo FROM Employees GROUP BY ReportsTo HAVING COUNT(\*) > 1);

1. **Customers with No Orders: List customers who have not placed any orders.**

SELECT CustomerName

FROM Customers

WHERE CustomerID NOT IN (SELECT CustomerID FROM Orders);

1. **Products in a Specific Category: Find products in a specific category (e.g., 'Beverages'**).

SELECT ProductName

FROM Products

WHERE CategoryID = (SELECT CategoryID FROM Categories WHERE CategoryName = 'Beverages');

1. **Orders with the Most Expensive Product: Retrieve orders with the most expensive product in each order.**

SELECT OrderID, ProductName, UnitPrice

FROM OrderDetails

WHERE (OrderID, UnitPrice) IN (SELECT OrderID, MAX(UnitPrice) FROM OrderDetails GROUP BY OrderID);

1. **Customers with Orders Over $1000: List customers who have placed orders with a total value exceeding $1000.**

SELECT CustomerName

FROM Customers

WHERE CustomerID IN (SELECT CustomerID FROM Orders GROUP BY CustomerID HAVING SUM(Total) > 1000);

1. **Suppliers with No Products: Identify suppliers who do not supply any products.**

SELECT SupplierName

FROM Suppliers

WHERE SupplierID NOT IN (SELECT DISTINCT SupplierID FROM Products);

1. **Employees with No Direct Reports: List employees who do not have any direct reports.**

SELECT EmployeeID, LastName, FirstName

FROM Employees

WHERE EmployeeID NOT IN (SELECT ReportsTo FROM Employees WHERE ReportsTo IS NOT NULL);

1. **Categories with More Than 5 Products: Find categories that have more than 5 products.**

SELECT CategoryName

FROM Categories

WHERE CategoryID IN (SELECT CategoryID FROM Products GROUP BY CategoryID HAVING COUNT(\*) > 5);

1. **Customers with Orders Placed After 2005: Retrieve customers who have placed orders after the year 2005.**

SELECT CustomerName

FROM Customers

WHERE CustomerID IN (SELECT CustomerID FROM Orders WHERE YEAR(OrderDate) > 2005);