1. **Retrieve all customers:**

SELECT \* FROM Customers;

1. **Retrieve all orders:**

SELECT \* FROM Orders;

1. **Retrieve all products:**

SELECT \* FROM Products;

1. **Retrieve all employees:**

SELECT \* FROM Employees;

1. **Retrieve all suppliers:**

SELECT \* FROM Suppliers;

1. **Retrieve all categories:**

SELECT \* FROM Categories;

1. **Retrieve all shippers:**

SELECT \* FROM Shippers;

1. **Retrieve all customers from a specific country (e.g., USA):**

SELECT \* FROM Customers WHERE Country = 'USA';

9. **Retrieve all products with a specific unit price range (e.g., between $10 and $20):**

SELECT \* FROM Products WHERE UnitPrice BETWEEN 10 AND 20;

1. **Retrieve all orders for a specific customer (e.g., with CustomerID = 'ALFKI'):**

SELECT \* FROM Orders WHERE CustomerID = 'ALFKI';

1. **Retrieve all orders with a specific employee responsible (e.g., EmployeeID = 5):**

SELECT \* FROM Orders WHERE EmployeeID = 5;

1. **Retrieve the total number of orders in the database:**

SELECT COUNT(\*) AS TotalOrders FROM Orders;

1. **Retrieve the total number of products in a specific category (e.g., CategoryID = 2):**

SELECT COUNT(\*) AS TotalProducts FROM Products WHERE CategoryID = 2;

1. **Retrieve the highest unit price from the products table:**

SELECT MAX(UnitPrice) AS HighestPrice FROM Products;

1. **Retrieve the lowest unit price from the products table:**

SELECT MIN(UnitPrice) AS LowestPrice FROM Products;

1. **Retrieve the average unit price from the products table:**

SELECT AVG(UnitPrice) AS AveragePrice FROM Products;

1. **Retrieve the total revenue from all orders:**

SELECT SUM(UnitPrice \* Quantity) AS TotalRevenue FROM OrderDetails;

1. **Retrieve the top 10 customers who placed the most orders:**

SELECT CustomerID, COUNT(\*) AS OrderCount

FROM Orders

GROUP BY CustomerID

ORDER BY OrderCount DESC

LIMIT 10;

1. **Retrieve the names and contact information of all customers located in a specific city (e.g., London):**

SELECT ContactName, Phone FROM Customers WHERE City = 'London';

1. **Retrieve the names and order dates of all orders that were placed on a specific date (e.g., '1997-07-04'):**

SELECT CustomerID, OrderDate FROM Orders WHERE OrderDate = '1997-07-04';

1. **Retrieve the names of all products that have been discontinued:**

SELECT ProductName FROM Products WHERE Discontinued = 1;

1. **Retrieve the names and quantities of products that have a stock level less than a certain threshold (e.g., 10):**

SELECT ProductName, UnitsInStock FROM Products WHERE UnitsInStock < 10;

1. **Retrieve the names and total quantities of products sold in each order:**

SELECT OrderID, ProductName, SUM(Quantity) AS TotalQuantity

FROM OrderDetails

INNER JOIN Products ON OrderDetails.ProductID = Products.ProductID

GROUP BY OrderID, ProductName;

1. **Retrieve the names of employees who have a specific title (e.g., 'Sales Representative'):**

SELECT FirstName, LastName FROM Employees WHERE Title = 'Sales Representative';

1. **Retrieve the names of employees who were hired during a specific year (e.g., 1995):**

SELECT FirstName, LastName FROM Employees WHERE YEAR(HireDate) = 1995;

1. **Retrieve the names of employees who were born in a specific city (e.g., Seattle):**

SELECT FirstName, LastName FROM Employees WHERE BirthCity = 'Seattle';

1. **Retrieve the names of employees who have a specific manager (e.g., EmployeeID = 2):**

SELECT FirstName, LastName FROM Employees WHERE ReportsTo = 2;

1. **Retrieve the names and quantities of products that were ordered by a specific customer (e.g., CustomerID = 'ALFKI'):**

SELECT Products.ProductName, OrderDetails.Quantity

FROM OrderDetails

INNER JOIN Orders ON OrderDetails.OrderID = Orders.OrderID

INNER JOIN Products ON OrderDetails.ProductID = Products.ProductID

WHERE Orders.CustomerID = 'ALFKI';

1. **Retrieve the names and total sales of each product (sorted by total sales in descending order):**

SELECT Products.ProductName, SUM(OrderDetails.Quantity) AS TotalSales

FROM OrderDetails

INNER JOIN Products ON OrderDetails.ProductID = Products.ProductID

GROUP BY Products.ProductName

ORDER BY TotalSales DESC;

1. **Retrieve the names of customers along with the total amount they spent on orders (sorted by total amount in descending order):**

SELECT Customers.CustomerID, Customers.ContactName, SUM(OrderDetails.UnitPrice \* OrderDetails.Quantity) AS TotalAmount

FROM Customers

INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID

INNER JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

GROUP BY Customers.CustomerID, Customers.ContactName

ORDER BY TotalAmount DESC;

1. **Retrieve the names and contact information of customers who have not placed any orders:**

SELECT Customers.CustomerID, Customers.ContactName, Customers.Phone

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID

WHERE Orders.CustomerID IS NULL;

1. **Retrieve the names and contact information of customers who have placed at least one order:**

SELECT Customers.CustomerID, Customers.ContactName, Customers.Phone

FROM Customers

INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

1. **Retrieve the names and contact information of customers who have placed more than five orders:**

SELECT Customers.CustomerID, Customers.ContactName, Customers.Phone

FROM Customers

INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, Customers.ContactName, Customers.Phone

HAVING COUNT(Orders.OrderID) > 5;

1. **Retrieve the names of employees and the number of orders they have taken (sorted by the number of orders in descending order):**

SELECT Employees.EmployeeID, Employees.FirstName, Employees.LastName, COUNT(Orders.OrderID) AS OrderCount

FROM Employees

LEFT JOIN Orders ON Employees.EmployeeID = Orders.EmployeeID

GROUP BY Employees.EmployeeID, Employees.FirstName, Employees.LastName

ORDER BY OrderCount DESC;

1. **Retrieve the names and contact information of customers along with the latest order date for each customer:**

SELECT Customers.CustomerID, Customers.ContactName, Customers.Phone, MAX(Orders.OrderDate) AS LatestOrderDate

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, Customers.ContactName, Customers.Phone;

1. **Retrieve the names of products along with their category names:**

SELECT Products.ProductName, Categories.CategoryName

FROM Products

INNER JOIN Categories ON Products.CategoryID = Categories.CategoryID;

1. **Retrieve the names of products along with the names of their suppliers:**

SELECT Products.ProductName, Suppliers.CompanyName

FROM Products

INNER JOIN Suppliers ON Products.SupplierID = Suppliers.SupplierID;

1. **Retrieve the names of products along with their category names and supplier names:**

SELECT Products.ProductName, Categories.CategoryName, Suppliers.CompanyName

FROM Products

INNER JOIN Categories ON Products.CategoryID = Categories.CategoryID

INNER JOIN Suppliers ON Products.SupplierID = Suppliers.SupplierID;

1. **Retrieve the names of employees along with the names of their managers:**

SELECT E.FirstName AS EmployeeFirstName, E.LastName AS EmployeeLastName, M.FirstName AS ManagerFirstName, M.LastName AS ManagerLastName

FROM Employees E

LEFT JOIN Employees M ON E.ReportsTo = M.EmployeeID;

1. **Retrieve the names of employees along with the names of their territories:**

SELECT Employees.FirstName, Employees.LastName, Territories.TerritoryDescription

FROM Employees

INNER JOIN EmployeeTerritories ON Employees.EmployeeID = EmployeeTerritories.EmployeeID

INNER JOIN Territories ON EmployeeTerritories.TerritoryID = Territories.TerritoryID;

1. **Retrieve the names of employees along with the names of their territories and region names:**

SELECT Employees.FirstName, Employees.LastName, Territories.TerritoryDescription, Region.RegionDescription

FROM Employees

INNER JOIN EmployeeTerritories ON Employees.EmployeeID = EmployeeTerritories.EmployeeID

INNER JOIN Territories ON EmployeeTerritories.TerritoryID = Territories.TerritoryID

INNER JOIN Region ON Territories.RegionID = Region.RegionID;

1. **Retrieve the names of products along with the names of the employees responsible for their sales (using the 'ReportsTo' field):**

SELECT Products.ProductName, Employees.FirstName, Employees.LastName

FROM Products

INNER JOIN Employees ON Products.ReportsTo = Employees.EmployeeID;

1. **Retrieve the names of products along with the names of the employees responsible for their sales (using the 'EmployeeID' field):**

SELECT Products.ProductName, Employees.FirstName, Employees.LastName

FROM Products

INNER JOIN Employees ON Products.EmployeeID = Employees.EmployeeID;

1. **Retrieve the names of employees along with the names of the customers they have handled orders for:**

SELECT DISTINCT Employees.FirstName, Employees.LastName, Customers.ContactName AS CustomerName

FROM Employees

INNER JOIN Orders ON Employees.EmployeeID = Orders.EmployeeID

INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;

1. **Retrieve the number of orders placed by each customer (sorted by the number of orders in descending order):**

SELECT Customers.CustomerID, Customers.ContactName, COUNT(Orders.OrderID) AS OrderCount

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, Customers.ContactName

ORDER BY OrderCount DESC;

1. **Retrieve the total revenue generated by each employee (sorted by total revenue in descending order):**

SELECT Employees.EmployeeID, Employees.FirstName, Employees.LastName, SUM(OrderDetails.UnitPrice \* OrderDetails.Quantity) AS TotalRevenue

FROM Employees

INNER JOIN Orders ON Employees.EmployeeID = Orders.EmployeeID

INNER JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

GROUP BY Employees.EmployeeID, Employees.FirstName, Employees.LastName

ORDER BY TotalRevenue DESC;

1. **Retrieve the number of orders and the total revenue generated for each employee:**

SELECT Employees.EmployeeID, Employees.FirstName, Employees.LastName, COUNT(Orders.OrderID) AS OrderCount, SUM(OrderDetails.UnitPrice \* OrderDetails.Quantity) AS TotalRevenue

FROM Employees

LEFT JOIN Orders ON Employees.EmployeeID = Orders.EmployeeID

LEFT JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

GROUP BY Employees.EmployeeID, Employees.FirstName, Employees.LastName;

1. **Retrieve the names of products along with the names of the suppliers and their respective cities:**

SELECT Products.ProductName, Suppliers.CompanyName, Suppliers.City

FROM Products

INNER JOIN Suppliers ON Products.SupplierID = Suppliers.SupplierID;

1. **Retrieve the names of products along with the names of the suppliers and their respective countries:**

SELECT Products.ProductName, Suppliers.CompanyName, Suppliers.Country

FROM Products

INNER JOIN Suppliers ON Products.SupplierID = Suppliers.SupplierID;

1. **Retrieve the names of products along with their category names and the total quantity of each product sold:**

SELECT Products.ProductName, Categories.CategoryName, SUM(OrderDetails.Quantity) AS TotalQuantity

FROM Products

INNER JOIN Categories ON Products.CategoryID = Categories.CategoryID

INNER JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID

GROUP BY Products.ProductName, Categories.CategoryName;

1. **Retrieve the names of products along with their category names and the total revenue generated by each product:**

SELECT Products.ProductName, Categories.CategoryName, SUM(OrderDetails.UnitPrice \* OrderDetails.Quantity) AS TotalRevenue

FROM Products

INNER JOIN Categories ON Products.CategoryID = Categories.CategoryID

INNER JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID

GROUP BY Products.ProductName, Categories.CategoryName;

1. **Retrieve the names of employees along with the names of their managers and the total revenue generated by each employee:**

SELECT E.EmployeeID, E.FirstName, E.LastName, M.FirstName AS ManagerFirstName, M.LastName AS ManagerLastName, SUM(OD.UnitPrice \* OD.Quantity) AS TotalRevenue

FROM Employees AS E

LEFT JOIN Employees AS M ON E.ReportsTo = M.EmployeeID

LEFT JOIN Orders AS O ON E.EmployeeID = O.EmployeeID

LEFT JOIN OrderDetails AS OD ON O.OrderID = OD.OrderID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, M.FirstName, M.LastName;

1. **Retrieve the names of employees along with the names of their managers and the total number of orders taken by each employee:**

SELECT E.EmployeeID, E.FirstName, E.LastName, M.FirstName AS ManagerFirstName, M.LastName AS ManagerLastName, COUNT(O.OrderID) AS OrderCount

FROM Employees AS E

LEFT JOIN Employees AS M ON E.ReportsTo = M.EmployeeID

LEFT JOIN Orders AS O ON E.EmployeeID = O.EmployeeID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, M.FirstName, M.LastName;

1. **Retrieve the names of employees along with the names of their territories and the total number of territories handled by each employee:**

SELECT E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription, COUNT(ET.TerritoryID) AS TerritoryCount

FROM Employees AS E

LEFT JOIN EmployeeTerritories AS ET ON E.EmployeeID = ET.EmployeeID

LEFT JOIN Territories AS T ON ET.TerritoryID = T.TerritoryID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription;

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName

FROM Customers AS C

INNER JOIN Orders AS O ON C.CustomerID = O.CustomerID

INNER JOIN Employees AS E ON O.EmployeeID = E.EmployeeID;

1. **Retrieve the names of products along with the names of the employees responsible for their sales (using the 'ReportsTo' field) and their respective territories:**

SELECT P.ProductName, E.FirstName, E.LastName, T.TerritoryDescription

FROM Products AS P

INNER JOIN Employees AS E ON P.ReportsTo = E.EmployeeID

INNER JOIN Territories AS T ON E.TerritoryID = T.TerritoryID;

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders and the territories they cover:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName, T.TerritoryDescription

FROM Customers AS C

INNER JOIN Orders AS O ON C.CustomerID = O.CustomerID

INNER JOIN Employees AS E ON O.EmployeeID = E.EmployeeID

INNER JOIN Territories AS T ON E.TerritoryID = T.TerritoryID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'ReportsTo' field):**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

INNER JOIN Products AS P ON E.ReportsTo = P.EmployeeID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'EmployeeID' field):**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

INNER JOIN Products AS P ON E.EmployeeID = P.EmployeeID;

1. **Retrieve the names of employees along with the names of their managers and the total revenue generated by each employee, including employees without any orders:**

SELECT E.EmployeeID, E.FirstName, E.LastName, M.FirstName AS ManagerFirstName, M.LastName AS ManagerLastName, COALESCE(SUM(OD.UnitPrice \* OD.Quantity), 0) AS TotalRevenue

FROM Employees AS E

LEFT JOIN Employees AS M ON E.ReportsTo = M.EmployeeID

LEFT JOIN Orders AS O ON E.EmployeeID = O.EmployeeID

LEFT JOIN OrderDetails AS OD ON O.OrderID = OD.OrderID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, M.FirstName, M.LastName;

1. **Retrieve the names of products along with their category names and the total quantity of each product sold, including products with no sales:**

SELECT P.ProductName, C.CategoryName, COALESCE(SUM(OD.Quantity), 0) AS TotalQuantity

FROM Products AS P

LEFT JOIN Categories AS C ON P.CategoryID = C.CategoryID

LEFT JOIN OrderDetails AS OD ON P.ProductID = OD.ProductID

GROUP BY P.ProductName, C.CategoryName;

1. **Retrieve the names of employees along with the names of their territories and the total number of territories handled by each employee, including employees without any territories:**

SELECT E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription, COALESCE(COUNT(ET.TerritoryID), 0) AS TerritoryCount

FROM Employees AS E

LEFT JOIN EmployeeTerritories AS ET ON E.EmployeeID = ET.EmployeeID

LEFT JOIN Territories AS T ON ET.TerritoryID = T.TerritoryID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription;

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders, including customers without any orders:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName

FROM Customers AS C

LEFT JOIN Orders AS O ON C.CustomerID = O.CustomerID

LEFT JOIN Employees AS E ON O.EmployeeID = E.EmployeeID;

1. **Retrieve the names of products along with the names of the employees responsible for their sales (using the 'ReportsTo' field) and their respective territories, including products without any responsible employees:**

SELECT P.ProductName, E.FirstName, E.LastName, T.TerritoryDescription

FROM Products AS P

LEFT JOIN Employees AS E ON P.ReportsTo = E.EmployeeID

LEFT JOIN Territories AS T ON E.TerritoryID = T.TerritoryID;

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders and the territories they cover, including customers without any orders:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName, T.TerritoryDescription

FROM Customers AS C

LEFT JOIN Orders AS O ON C.CustomerID = O.CustomerID

LEFT JOIN Employees AS E ON O.EmployeeID = E.EmployeeID

LEFT JOIN Territories AS T ON E.TerritoryID = T.TerritoryID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'ReportsTo' field), including employees without any products:**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

LEFT JOIN Products AS P ON E.ReportsTo = P.EmployeeID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'EmployeeID' field), including employees without any products:**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

LEFT JOIN Products AS P ON E.EmployeeID = P.EmployeeID;

1. **Retrieve the names of employees along with the names of their managers and the total revenue generated by each employee, including employees without any orders:**

SELECT E.EmployeeID, E.FirstName, E.LastName, M.FirstName AS ManagerFirstName, M.LastName AS ManagerLastName, COALESCE(SUM(OD.UnitPrice \* OD.Quantity), 0) AS TotalRevenue

FROM Employees AS E

LEFT JOIN Employees AS M ON E.ReportsTo = M.EmployeeID

LEFT JOIN Orders AS O ON E.EmployeeID = O.EmployeeID

LEFT JOIN OrderDetails AS OD ON O.OrderID = OD.OrderID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, M.FirstName, M.LastName;

1. **Retrieve the names of products along with their category names and the total quantity of each product sold, including products with no sales:**

SELECT P.ProductName, C.CategoryName, COALESCE(SUM(OD.Quantity), 0) AS TotalQuantity

FROM Products AS P

LEFT JOIN Categories AS C ON P.CategoryID = C.CategoryID

LEFT JOIN OrderDetails AS OD ON P.ProductID = OD.ProductID

GROUP BY P.ProductName, C.CategoryName;

1. **Retrieve the names of employees along with the names of their territories and the total number of territories handled by each employee, including employees without any territories:**

SELECT E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription, COALESCE(COUNT(ET.TerritoryID), 0) AS TerritoryCount

FROM Employees AS E

LEFT JOIN EmployeeTerritories AS ET ON E.EmployeeID = ET.EmployeeID

LEFT JOIN Territories AS T ON ET.TerritoryID = T.TerritoryID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription;

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders, including customers without any orders:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName

FROM Customers AS C

LEFT JOIN Orders AS O ON C.Customer

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders and the territories they cover, including customers without any orders:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName, T.TerritoryDescription

FROM Customers AS C

LEFT JOIN Orders AS O ON C.CustomerID = O.CustomerID

LEFT JOIN Employees AS E ON O.EmployeeID = E.EmployeeID

LEFT JOIN Territories AS T ON E.TerritoryID = T.TerritoryID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'ReportsTo' field), including employees without any products:**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

LEFT JOIN Products AS P ON E.ReportsTo = P.EmployeeID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'EmployeeID' field), including employees without any products:**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

LEFT JOIN Products AS P ON E.EmployeeID = P.EmployeeID;

1. **Retrieve the names of employees along with the names of their managers and the total revenue generated by each employee, including employees without any orders:**

SELECT E.EmployeeID, E.FirstName, E.LastName, M.FirstName AS ManagerFirstName, M.LastName AS ManagerLastName, COALESCE(SUM(OD.UnitPrice \* OD.Quantity), 0) AS TotalRevenue

FROM Employees AS E

LEFT JOIN Employees AS M ON E.ReportsTo = M.EmployeeID

LEFT JOIN Orders AS O ON E.EmployeeID = O.EmployeeID

LEFT JOIN OrderDetails AS OD ON O.OrderID = OD.OrderID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, M.FirstName, M.LastName;

1. **Retrieve the names of products along with their category names and the total quantity of each product sold, including products with no sales:**

SELECT P.ProductName, C.CategoryName, COALESCE(SUM(OD.Quantity), 0) AS TotalQuantity

FROM Products AS P

LEFT JOIN Categories AS C ON P.CategoryID = C.CategoryID

LEFT JOIN OrderDetails AS OD ON P.ProductID = OD.ProductID

GROUP BY P.ProductName, C.CategoryName;

1. **Retrieve the names of employees along with the names of their territories and the total number of territories handled by each employee, including employees without any territories:**

SELECT E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription, COALESCE(COUNT(ET.TerritoryID), 0) AS TerritoryCount

FROM Employees AS E

LEFT JOIN EmployeeTerritories AS ET ON E.EmployeeID = ET.EmployeeID

LEFT JOIN Territories AS T ON ET.TerritoryID = T.TerritoryID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription;

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders, including customers without any orders:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName

FROM Customers AS C

LEFT JOIN Orders AS O ON C.CustomerID = O.CustomerID

LEFT JOIN Employees AS E ON O.EmployeeID = E.EmployeeID;

1. **Retrieve the names of products along with the names of the employees responsible for their sales (using the 'ReportsTo' field) and their respective territories, including products without any responsible employees:**

SELECT P.ProductName, E.FirstName, E.LastName, T.TerritoryDescription

FROM Products AS P

LEFT JOIN Employees AS E ON P.ReportsTo = E.EmployeeID

LEFT JOIN Territories AS T ON E.TerritoryID = T.TerritoryID;

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders and the territories they cover, including customers without any orders:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName, T.TerritoryDescription

FROM Customers AS C

LEFT JOIN Orders AS O ON C.CustomerID = O.CustomerID

LEFT JOIN Employees AS E ON O.EmployeeID = E.EmployeeID

LEFT JOIN Territories AS T ON E.TerritoryID = T.TerritoryID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'ReportsTo' field), including employees without any products:**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

LEFT JOIN Products AS P ON E.ReportsTo = P.EmployeeID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'EmployeeID' field), including employees without any products:**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

LEFT JOIN Products AS P ON E.EmployeeID = P.EmployeeID;

1. **Retrieve the names of employees along with the names of their managers and the total revenue generated by each employee, including employees without any orders:**

SELECT E.EmployeeID, E.FirstName, E.LastName, M.FirstName AS ManagerFirstName, M.LastName AS ManagerLastName, COALESCE(SUM(OD.UnitPrice \* OD.Quantity), 0) AS TotalRevenue

FROM Employees AS E

LEFT JOIN Employees AS M ON E.ReportsTo = M.EmployeeID

LEFT JOIN Orders AS O ON E.EmployeeID = O.EmployeeID

LEFT JOIN OrderDetails AS OD ON O.OrderID = OD.OrderID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, M.FirstName, M.LastName;

1. **Retrieve the names of products along with their category names and the total quantity of each product sold, including products with no sales:**

SELECT P.ProductName, C.CategoryName, COALESCE(SUM(OD.Quantity), 0) AS TotalQuantity

FROM Products AS P

LEFT JOIN Categories AS C ON P.CategoryID = C.CategoryID

LEFT JOIN OrderDetails AS OD ON P.ProductID = OD.ProductID

GROUP BY P.ProductName, C.CategoryName;

1. **Retrieve the names of employees along with the names of their territories and the total number of territories handled by each employee, including employees without any territories:**

SELECT E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription, COALESCE(COUNT(ET.TerritoryID), 0) AS TerritoryCount

FROM Employees AS E

LEFT JOIN EmployeeTerritories AS ET ON E.EmployeeID = ET.EmployeeID

LEFT JOIN Territories AS T ON ET.TerritoryID = T.TerritoryID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, T.TerritoryDescription;

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders, including customers without any orders:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName

FROM Customers AS C

LEFT JOIN Orders AS O ON C.CustomerID = O.CustomerID

LEFT JOIN Employees AS E ON O.EmployeeID = E.EmployeeID;

1. **Retrieve the names of products along with the names of the employees responsible for their sales (using the 'ReportsTo' field) and their respective territories, including products without any responsible employees:**

SELECT P.ProductName, E.FirstName, E.LastName, T.TerritoryDescription

FROM Products AS P

LEFT JOIN Employees AS E ON P.ReportsTo = E.EmployeeID

LEFT JOIN Territories AS T ON E.TerritoryID = T.TerritoryID;

1. **Retrieve the names of customers along with the names of the employees responsible for handling their orders and the territories they cover, including customers without any orders:**

SELECT C.CustomerID, C.ContactName, E.FirstName, E.LastName, T.TerritoryDescription

FROM Customers AS C

LEFT JOIN Orders AS O ON C.CustomerID = O.CustomerID

LEFT JOIN Employees AS E ON O.EmployeeID = E.EmployeeID

LEFT JOIN Territories AS T ON E.TerritoryID = T.TerritoryID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'ReportsTo' field), including employees without any products:**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

LEFT JOIN Products AS P ON E.ReportsTo = P.EmployeeID;

1. **Retrieve the names of employees along with the names of the products they are responsible for (using the 'EmployeeID' field), including employees without any products:**

SELECT E.FirstName, E.LastName, P.ProductName

FROM Employees AS E

LEFT JOIN Products AS P ON E.EmployeeID = P.EmployeeID;

1. **Retrieve the names of employees along with the names of their managers and the total revenue generated by each employee, including employees without any orders:**

SELECT E.EmployeeID, E.FirstName, E.LastName, M.FirstName AS ManagerFirstName, M.LastName AS ManagerLastName, COALESCE(SUM(OD.UnitPrice \* OD.Quantity), 0) AS TotalRevenue

FROM Employees AS E

LEFT JOIN Employees AS M ON E.ReportsTo = M.EmployeeID

LEFT JOIN Orders AS O ON E.EmployeeID = O.EmployeeID

LEFT JOIN OrderDetails AS OD ON O.OrderID = OD.OrderID

GROUP BY E.EmployeeID, E.FirstName, E.LastName, M.FirstName, M.LastName;

These queries should give you a good overview of the types of queries you can perform on the Northwind database and the data you can retrieve from it. You can use these queries as a starting point for your own exploration and analysis of the database. Remember to replace the table and column names with your actual table and column names if you're using a different database schema.