**Exercise 1: List all customers and their orders (including customers with no orders).**

SELECT

c.ContactName as CustomerName,

o.OrderID

FROM customers c

LEFT JOIN orders o ON c.CustomerID = o.CustomerID;

**Exercise 2: Find all products and their suppliers.**

SELECT

p.ProductName,

s.CompanyName AS Supplier

FROM products p LEFT JOIN suppliers s ON p.SupplierID = s.SupplierID;

**Exercise 3: Retrieve all employees and the territories they are responsible for.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription

FROM employees e

LEFT JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

LEFT JOIN territories t ON et.TerritoryID = t.TerritoryID;

**Exercise 4: List all categories and their products.**

SELECT

c.CategoryName,

p.ProductName

FROM categories c

LEFT JOIN products p ON c.CategoryID = p.CategoryID;

**Exercise 5: Get a list of customers and their respective orders (excluding customers with no orders).**

SELECT

c.CustomerName,

o.OrderID

FROM customers c

INNER JOIN orders o ON c.CustomerID = o.CustomerID;

**Exercise 6: Retrieve products along with their suppliers and categories.**

SELECT

p.ProductName,

s.CompanyName AS Supplier,

c.CategoryName

FROM products p

INNER JOIN suppliers s ON p.SupplierID = s.SupplierID

INNER JOIN categories c ON p.CategoryID = c.CategoryID;

**Exercise 7: List all employees and the territories they are responsible for (including employees with no territories).**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription

FROM employees e

LEFT JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

LEFT JOIN territories t ON et.TerritoryID = t.TerritoryID;

**Exercise 8: Retrieve a list of customers and their orders (excluding customers with no orders) along with the shipper's company name.**

SELECT

c.CustomerName,

o.OrderID,

s.CompanyName AS Shipper

FROM customers c

INNER JOIN orders o ON c.CustomerID = o.CustomerID

INNER JOIN shippers s ON o.ShipperID = s.ShipperID;

**Exercise 9: Get a list of products and their suppliers along with the employees who processed the orders.**

SELECT

p.ProductName,

s.CompanyName AS Supplier,

e.FirstName AS Employee

FROM products p

INNER JOIN suppliers s ON p.SupplierID = s.SupplierID

INNER JOIN order\_details od ON p.ProductID = od.ProductID

INNER JOIN orders o ON od.OrderID = o.OrderID

INNER JOIN employees e ON o.EmployeeID = e.EmployeeID;

**Exercise 10: List products and their categories along with the total quantity ordered for each product.**

SELECT

p.ProductName,

c.CategoryName,

SUM(od.Quantity) AS TotalQuantityOrdered

FROM products p

INNER JOIN categories c ON p.CategoryID = c.CategoryID

LEFT JOIN order\_details od ON p.ProductID = od.ProductID

GROUP BY p.ProductName, c.CategoryName;

**Exercise 11: List customers who have placed orders in the past year along with their order counts.**

SELECT

c.CustomerName,

COUNT(o.OrderID) AS OrderCount

FROM customers c

LEFT JOIN orders o ON c.CustomerID = o.CustomerID

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

GROUP BY c.CustomerName;

**Exercise 12: Retrieve products and their suppliers along with the total revenue generated for each product.**

SELECT

p.ProductName,

s.CompanyName AS Supplier,

SUM(od.UnitPrice \* od.Quantity) AS TotalRevenue

FROM products p

INNER JOIN suppliers s ON p.SupplierID = s.SupplierID

INNER JOIN order\_details od ON p.ProductID = od.ProductID

GROUP BY p.ProductName, s.CompanyName;

**Exercise 13: Find employees who have not processed any orders.**

SELECT

e.EmployeeID,

e.FirstName,

e.LastName

FROM employees e

LEFT JOIN orders o ON e.EmployeeID = o.EmployeeID

WHERE o.EmployeeID IS NULL;

**Exercise 14: Retrieve a list of customers who have placed orders, the number of orders they've placed, and the total order amount for each customer.**

SELECT

c.CustomerName,

COUNT(o.OrderID) AS OrderCount,

SUM(od.UnitPrice \* od.Quantity) AS TotalOrderAmount

FROM customers c

LEFT JOIN orders o ON c.CustomerID = o.CustomerID

LEFT JOIN order\_details od ON o.OrderID = od.OrderID

GROUP BY c.CustomerName;

**Exercise 15: List products along with the categories they belong to and the number of times each product has been ordered.**

SELECT

p.ProductName,

c.CategoryName,

COUNT(od.OrderID) AS OrderCount

FROM products p

INNER JOIN categories c ON p.CategoryID = c.CategoryID

LEFT JOIN order\_details od ON p.ProductID = od.ProductID

GROUP BY p.ProductName, c.CategoryName;

**Exercise 16: Find employees and the number of orders they have processed (include employees with zero orders).**

SELECT

e.FirstName,

e.LastName,

COUNT(o.OrderID) AS OrderCount

FROM employees e

LEFT JOIN orders o ON e.EmployeeID = o.EmployeeID

GROUP BY e.FirstName, e.LastName;

**Exercise 17: Retrieve a list of customers and the products they've ordered along with the order date.**

SELECT

c.CustomerName,

p.ProductName,

o.OrderDate

FROM customers c

LEFT JOIN orders o ON c.CustomerID = o.CustomerID

LEFT JOIN order\_details od ON o.OrderID = od.OrderID

LEFT JOIN products p ON od.ProductID = p.ProductID;

**Exercise 18: List all products and their suppliers along with the total quantity ordered for each product in the past year.**

SELECT

p.ProductName,

s.CompanyName AS Supplier,

SUM(od.Quantity) AS TotalQuantityOrdered

FROM products p

INNER JOIN suppliers s ON p.SupplierID = s.SupplierID

LEFT JOIN order\_details od ON p.ProductID = od.ProductID

LEFT JOIN orders o ON od.OrderID = o.OrderID

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

GROUP BY p.ProductName, s.CompanyName;

**Exercise 19: Get a list of employees and the territories they are responsible for, sorted by employee last name and territory description.**

SELECT

e.LastName,

e.FirstName,

t.TerritoryDescription

FROM employees e

LEFT JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

LEFT JOIN territories t ON et.TerritoryID = t.TerritoryID

ORDER BY e.LastName, t.TerritoryDescription;

**Exercise 20: Retrieve products and their categories along with the total revenue generated for each category.**

SELECT

c.CategoryName,

SUM(od.UnitPrice \* od.Quantity) AS TotalCategoryRevenue

FROM categories c

LEFT JOIN products p ON c.CategoryID = p.CategoryID

LEFT JOIN order\_details od ON p.ProductID = od.ProductID

GROUP BY c.CategoryName;

**Exercise 21: Retrieve a list of customers, their orders, and the products in those orders.**

SELECT

c.CustomerName,

o.OrderID,

p.ProductName

FROM customers c

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID;

**Exercise 22: List employees, their territories, and the customers who have orders in those territories.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription,

c.CustomerName

FROM employees e

JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

JOIN territories t ON et.TerritoryID = t.TerritoryID

JOIN orders o ON o.employeeid = e.employeeid

JOIN customers c ON o.customerid= c.customerid;

**Exercise 23: Retrieve a list of products, their suppliers, and the customers who have ordered those products.**

SELECT

p.ProductName,

s.CompanyName AS Supplier,

c.CustomerName

FROM products p

JOIN suppliers s ON p.SupplierID = s.SupplierID

JOIN order\_details od ON p.ProductID = od.ProductID

JOIN orders o ON od.OrderID = o.OrderID

JOIN customers c ON o.CustomerID = c.CustomerID;

**Exercise 24: List customers, their orders, the employees who processed those orders, and the shippers who shipped them.**

SELECT

c.CustomerName,

o.OrderID,

e.FirstName AS Employee,

s.ShipperName

FROM customers c

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN employees e ON o.EmployeeID = e.EmployeeID

JOIN shippers s ON o.ShipperID = s.ShipperID;

**Exercise 25: Retrieve a list of products, their categories, their suppliers, and the territories in which the suppliers are located.**

SELECT

p.ProductName,

c.CategoryName,

s.CompanyName AS Supplier,

t.TerritoryDescription

FROM products p

JOIN categories c ON p.CategoryID = c.CategoryID

JOIN suppliers s ON p.SupplierID = s.SupplierID

JOIN territories t ON s.TerritoryID = t.TerritoryID;

**Exercise 26: List customers, their orders, the employees who processed those orders, and the products in those orders.**

SELECT

c.CustomerName,

o.OrderID,

e.FirstName AS Employee,

p.ProductName

FROM customers c

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN employees e ON o.EmployeeID = e.EmployeeID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID;

**Exercise 27: Retrieve a list of employees, their territories, and the products in the orders that are associated with those territories.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription,

p.ProductName

FROM employees e

JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

JOIN territories t ON et.TerritoryID = t.TerritoryID

JOIN region r ON t.RegionID = r.RegionID

JOIN customers c ON r.RegionID = c.RegionID

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID;

**Exercise 28: List customers, their orders, the employees who processed those orders, the products in those orders, and the suppliers of those products.**

SELECT

c.CustomerName,

o.OrderID,

e.FirstName AS Employee,

p.ProductName,

s.CompanyName AS Supplier

FROM customers c

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN employees e ON o.EmployeeID = e.EmployeeID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID

JOIN suppliers s ON p.SupplierID = s.SupplierID;

**Exercise 29: Retrieve a list of products, their categories, their suppliers, and the customers who have ordered those products in the past year.**

SELECT

p.ProductName,

c.CategoryName,

s.CompanyName AS Supplier,

c1.CustomerName

FROM products p

JOIN categories c ON p.CategoryID = c.CategoryID

JOIN suppliers s ON p.SupplierID = s.SupplierID

JOIN order\_details od ON p.ProductID = od.ProductID

JOIN orders o ON od.OrderID = o.OrderID

JOIN customers c1 ON o.CustomerID = c1.CustomerID

WHERE YEAR(o.OrderDate) = YEAR(CURRENT\_DATE) - 1;

**Exercise 30: List employees, their territories, the customers who have orders in those territories, and the products in those orders.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription,

c.CustomerName,

p.ProductName

FROM employees e

JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

JOIN territories t ON et.TerritoryID = t.TerritoryID

JOIN region r ON t.RegionID = r.RegionID

JOIN customers c ON r.RegionID = c.RegionID

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID;

**Exercise 31: Retrieve a list of products and their suppliers where the supplier is located in the USA.**

SELECT

p.ProductName,

s.CompanyName AS Supplier

FROM products p

JOIN suppliers s ON p.SupplierID = s.SupplierID

WHERE s.Country = 'USA';

**Exercise 32: List customers who have placed orders in the past year along with their order counts.**

SELECT

c.CustomerName,

COUNT(o.OrderID) AS OrderCount

FROM customers c

LEFT JOIN orders o ON c.CustomerID = o.CustomerID

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

GROUP BY c.CustomerName;

**Exercise 33: Retrieve products and their categories where the category name contains 'Beverages'.**

SELECT

p.ProductName,

c.CategoryName

FROM products p

JOIN categories c ON p.CategoryID = c.CategoryID

WHERE c.CategoryName LIKE '%Beverages%';

**Exercise 34: List employees who have not processed any orders.**

SELECT

e.EmployeeID,

e.FirstName,

e.LastName

FROM employees e

LEFT JOIN orders o ON e.EmployeeID = o.EmployeeID

WHERE o.EmployeeID IS NULL;

**Exercise 35: Retrieve a list of customers and their respective orders where the order total is greater than $1,000.**

SELECT

c.CustomerName,

o.OrderID

FROM customers c

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN

(

SELECT

OrderID,

SUM(UnitPrice \* Quantity) AS TotalAmount

FROM order\_details

GROUP BY OrderID

HAVING TotalAmount > 1000

) t ON o.OrderID = t.OrderID;

**Exercise 36: List products and their suppliers where the supplier's company name starts with 'A'.**

SELECT

p.ProductName,

s.CompanyName AS Supplier

FROM products p

JOIN suppliers s ON p.SupplierID = s.SupplierID

WHERE s.CompanyName LIKE 'A%';

**Exercise 37: Retrieve a list of employees and their territories where the territory description contains 'West'.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription

FROM employees e

JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

JOIN territories t ON et.TerritoryID = t.TerritoryID

WHERE t.TerritoryDescription LIKE '%West%';

**Exercise 38: List customers and their orders where the order was shipped via Speedy Express.**

SELECT

c.CustomerName,

o.OrderID

FROM customers c

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN shippers s ON o.ShipperID = s.ShipperID

WHERE s.ShipperName = 'Speedy Express';

**Exercise 39: Retrieve a list of employees and the products in the orders they have processed where the employee ID is less than 5.**

SELECT

e.FirstName,

e.LastName,

p.ProductName

FROM employees e

JOIN orders o ON e.EmployeeID = o.EmployeeID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID

WHERE e.EmployeeID < 5;

**Exercise 40: List products and their categories where the product price is greater than $50.**

SELECT

p.ProductName,

c.CategoryName

FROM products p

JOIN categories c ON p.CategoryID = c.CategoryID

WHERE p.UnitPrice > 50.00;

**Exercise 41: Retrieve a list of customers, their orders, products in those orders, employees who processed those orders, and shippers who shipped them.**

SELECT

c.CustomerName,

o.OrderID,

p.ProductName,

e.FirstName AS Employee,

s.ShipperName

FROM customers c

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID

JOIN employees e ON o.EmployeeID = e.EmployeeID

JOIN shippers s ON o.ShipperID = s.ShipperID;

**Exercise 42: List employees, their territories, customers in those territories, orders from those customers, and products in those orders.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription,

c.CustomerName,

o.OrderID,

p.ProductName

FROM employees e

JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

JOIN territories t ON et.TerritoryID = t.TerritoryID

JOIN region r ON t.RegionID = r.RegionID

JOIN customers c ON r.RegionID = c.RegionID

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID;

**Exercise 43: Retrieve a list of products, their categories, suppliers of those products, employees who handled orders containing those products, and customers who placed orders with those products.**

SELECT

p.ProductName,

c.CategoryName,

s.CompanyName AS Supplier,

e.FirstName AS Employee,

cu.CustomerName

FROM products p

JOIN categories c ON p.CategoryID = c.CategoryID

JOIN suppliers s ON p.SupplierID = s.SupplierID

JOIN order\_details od ON p.ProductID = od.ProductID

JOIN orders o ON od.OrderID = o.OrderID

JOIN employees e ON o.EmployeeID = e.EmployeeID

JOIN customers cu ON o.CustomerID = cu.CustomerID;

**Exercise 44: List employees, their territories, regions where they are assigned, customers in those regions, orders from those customers, and products in those orders.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription,

r.RegionDescription,

c.CustomerName,

o.OrderID,

p.ProductName

FROM employees e

JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

JOIN territories t ON et.TerritoryID = t.TerritoryID

JOIN region r ON t.RegionID = r.RegionID

JOIN customers c ON r.RegionID = c.RegionID

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID;

**Exercise 45: Retrieve a list of employees, their territories, customers in those territories, products in orders from those customers, and suppliers of those products.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription,

c.CustomerName,

p.ProductName,

s.CompanyName AS Supplier

FROM employees e

JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

JOIN territories t ON et.TerritoryID = t.TerritoryID

JOIN customers c ON t.TerritoryID = c.TerritoryID

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID

JOIN suppliers s ON p.SupplierID = s.SupplierID;

**Exercise 46: List customers, their orders, employees who processed those orders, products in those orders, and suppliers of those products.**

SELECT

c.CustomerName,

o.OrderID,

e.FirstName AS Employee,

p.ProductName,

s.CompanyName AS Supplier

FROM customers c

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN employees e ON o.EmployeeID = e.EmployeeID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID

JOIN suppliers s ON p.SupplierID = s.SupplierID;

**Exercise 47: Retrieve a list of employees, their territories, customers in those territories, orders from those customers, products in those orders, and suppliers of those products.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription,

c.CustomerName,

o.OrderID,

p.ProductName,

s.CompanyName AS Supplier

FROM employees e

JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

JOIN territories t ON et.TerritoryID = t.TerritoryID

JOIN customers c ON t.TerritoryID = c.TerritoryID

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID

JOIN suppliers s ON p.SupplierID = s.SupplierID;

**Exercise 48: List products, their categories, suppliers of those products, employees who processed orders containing those products, shippers of those orders, and customers who placed orders with those products.**

SELECT

p.ProductName,

c.CategoryName,

s.CompanyName AS Supplier,

e.FirstName AS Employee,

sh.ShipperName,

cu.CustomerName

FROM products p

JOIN categories c ON p.CategoryID = c.CategoryID

JOIN suppliers s ON p.SupplierID = s.SupplierID

JOIN order\_details od ON p.ProductID = od.ProductID

JOIN orders o ON od.OrderID = o.OrderID

JOIN employees e ON o.EmployeeID = e.EmployeeID

JOIN shippers sh ON o.ShipperID = sh.ShipperID

JOIN customers cu ON o.CustomerID = cu.CustomerID;

**Exercise 49: Retrieve a list of employees, their territories, customers in those territories, products in orders from those customers, suppliers of those products, and shippers of those orders.**

SELECT

e.FirstName,

e.LastName,

t.TerritoryDescription,

c.CustomerName,

p.ProductName,

s.CompanyName AS Supplier,

sh.ShipperName

FROM employees e

JOIN employee\_territories et ON e.EmployeeID = et.EmployeeID

JOIN territories t ON et.TerritoryID = t.TerritoryID

JOIN customers c ON t.TerritoryID = c.TerritoryID

JOIN orders o ON c.CustomerID = o.CustomerID

JOIN order\_details od ON o.OrderID = od.OrderID

JOIN products p ON od.ProductID = p.ProductID

JOIN suppliers s ON p.SupplierID = s.SupplierID

JOIN shippers sh ON o.ShipperID = sh.ShipperID;

**Exercise 50: List products, their categories, suppliers of those products, employees who processed orders containing those products, shippers of those orders, and customers who placed orders with those products where the total order amount is greater than $1,000.**

SELECT

p.ProductName,

c.CategoryName,

s.CompanyName AS Supplier,

e.FirstName AS Employee,

sh.ShipperName,

cu.CustomerName,

SUM(od.UnitPrice \* od.Quantity) AS TotalOrderAmount

FROM products p

JOIN categories c ON p.CategoryID = c.CategoryID

JOIN suppliers s ON p.SupplierID = s.SupplierID

JOIN order\_details od ON p.ProductID = od.ProductID

JOIN orders o ON od.OrderID = o.OrderID

JOIN employees e ON o.EmployeeID = e.EmployeeID

JOIN shippers sh ON o.ShipperID = sh.ShipperID

JOIN customers cu ON o.CustomerID = cu.CustomerID

GROUP BY

p.ProductName,

c.CategoryName,

s.CompanyName,

e.FirstName,

sh.ShipperName,

cu.CustomerName

HAVING TotalOrderAmount > 1000;