Assuming we have a table named **employees** with the following structure:

**CREATE TABLE employees (**

**id INT PRIMARY KEY,**

**name VARCHAR(255),**

**manager\_id INT,**

**department VARCHAR(255),**

**hire\_date DATE,**

**salary DECIMAL(10, 2),**

**job\_title VARCHAR(255)**

**);**

INSERT INTO employees (id, name, manager\_id, department, hire\_date, salary, job\_title) VALUES

(1, 'John', NULL, 'Management', '2020-01-15', 80000.00, 'CEO'),

(2, 'Alice', 1, 'Management', '2021-05-20', 60000.00, 'COO'),

(3, 'Bob', 1, 'Management', '2021-03-10', 60000.00, 'CFO'),

(4, 'Charlie', 2, 'Engineering', '2019-07-05', 75000.00, 'Lead Engineer'),

(5, 'David', 2, 'Engineering', '2020-09-12', 70000.00, 'Software Engineer'),

(6, 'Eve', 3, 'Finance', '2022-02-28', 55000.00, 'Financial Analyst'),

(7, 'Frank', 3, 'Finance', '2022-01-10', 56000.00, 'Accountant');

1. **Retrieve a list of employees and their managers:**

select

e.name as employee,

m.name as manager

from employees as e left join employees as m

on e.manager\_id = m.id;

1. **List the employees who do not have a manager:**

SELECT

e1.name AS employee

FROM employees e1

LEFT JOIN employees e2 ON e1.id = e2.manager\_id

WHERE e2.id IS NULL;

1. **Display the hierarchy for a specific employee (e.g., 'Alice'):**

SELECT

e1.name AS employee,

e2.name AS manager

FROM employees e1

LEFT JOIN employees e2 ON e1.manager\_id = e2.id

WHERE e1.name = 'Alice';

1. **Find the highest-level manager (top-level) in the organization:**

SELECT

e1.name AS top\_manager

FROM employees e1

LEFT JOIN employees e2 ON e1.id = e2.manager\_id

WHERE e2.id IS NULL;

1. **List the employees who are managed by 'John':**

SELECT

e1.name AS employee

FROM employees e1

INNER JOIN employees e2 ON e1.manager\_id = e2.id

WHERE e2.name = 'John';

1. **Calculate the total number of subordinates for each manager:**

SELECT

e2.name AS manager,

COUNT(e1.id) AS num\_subordinates

FROM employees e1

INNER JOIN employees e2 ON e1.manager\_id = e2.id

GROUP BY e2.name;

1. **Find all employees who have at least one subordinate:**

SELECT

DISTINCT e2.name AS manager

FROM employees e1

INNER JOIN employees e2 ON e1.manager\_id = e2.id;

1. **Retrieve a list of employees and their immediate subordinates:**

SELECT

e2.name AS manager,

GROUP\_CONCAT(e1.name) AS subordinates

FROM employees e1

INNER JOIN employees e2 ON e1.manager\_id = e2.id

GROUP BY e2.name;

1. **Display the hierarchy for all employees in a tabular format with indentation:**

If you want to display the employee hierarchy in a tabular format without using WITH RECURSIVE, you would typically need to manually join the employees table multiple times to get the hierarchy levels, but it becomes impractical for large or deep hierarchies. Without recursion, you would need to join the employees table multiple times for each level of hierarchy.

Here's an approach that avoids using WITH RECURSIVE and instead uses a series of self-joins:

SELECT

CONCAT(

REPEAT(' ', 0), e1.name

) AS level\_0\_employee,

NULL AS manager\_name

FROM employees e1

WHERE e1.manager\_id IS NULL

UNION ALL

SELECT

CONCAT(

REPEAT(' ', 1), e2.name

) AS level\_1\_employee,

e1.name AS manager\_name

FROM employees e2

JOIN employees e1 ON e2.manager\_id = e1.id

WHERE e1.manager\_id IS NULL

UNION ALL

SELECT

CONCAT(

REPEAT(' ', 2), e3.name

) AS level\_2\_employee,

e2.name AS manager\_name

FROM employees e3

JOIN employees e2 ON e3.manager\_id = e2.id

JOIN employees e1 ON e2.manager\_id = e1.id

WHERE e1.manager\_id IS NULL

ORDER BY level\_0\_employee, level\_1\_employee, level\_2\_employee;

WITH RECURSIVE EmployeeHierarchy AS (

SELECT

e1.id,

e1.name AS employee,

e1.manager\_id,

0 AS level

FROM employees e1

WHERE e1.manager\_id IS NULL

UNION ALL

SELECT

e2.id,

e2.name AS employee,

e2.manager\_id,

eh.level + 1

FROM EmployeeHierarchy eh

JOIN employees e2 ON eh.id = e2.manager\_id

)

SELECT

CONCAT(REPEAT(' ', eh.level), eh.employee) AS indented\_employee,

e.manager\_id,

e.name AS manager

FROM EmployeeHierarchy eh

LEFT JOIN employees e ON eh.manager\_id = e.id

ORDER BY eh.level, eh.employee;

**Consider the following table**

drop table employees;

**CREATE TABLE employees (**

**EmployeeID INT PRIMARY KEY,**

**LastName VARCHAR(255),**

**FirstName VARCHAR(255),**

**Title VARCHAR(255),**

**TitleOfCourtesy VARCHAR(255),**

**BirthDate DATE,**

**HireDate DATE,**

**Address VARCHAR(255),**

**City VARCHAR(255),**

**Region VARCHAR(255),**

**PostalCode VARCHAR(255),**

**Country VARCHAR(255),**

**HomePhone VARCHAR(255),**

**Extension VARCHAR(255),**

**Photo BLOB,**

**Notes TEXT,**

**ReportsTo INT,**

**PhotoPath VARCHAR(255)**

**);**

**Insert the following rows to the table**

INSERT INTO employees (EmployeeID, LastName, FirstName, Title, TitleOfCourtesy, BirthDate, HireDate, Address, City, Region, PostalCode, Country, HomePhone, Extension, Photo, Notes, ReportsTo, PhotoPath)

VALUES

(1, 'Doe', 'John', 'Manager', 'Mr.', '1980-05-15', '2005-07-10', '123 Main St', 'New York', 'NY', '10001', 'USA', '555-123-4567', '1234', NULL, 'Some notes about John', NULL, 'john.jpg'),

(2, 'Smith', 'Jane', 'Assistant Manager', 'Ms.', '1985-03-20', '2007-02-18', '456 Elm St', 'Los Angeles', 'CA', '90001', 'USA', '555-987-6543', '5678', NULL, 'Some notes about Jane', 1, 'jane.jpg'),

(3, 'Johnson', 'Robert', 'Sales Representative', 'Mr.', '1990-09-10', '2010-04-05', '789 Oak St', 'Chicago', 'IL', '60001', 'USA', '555-567-8901', '6789', NULL, 'Some notes about Robert', 1, 'robert.jpg'),

(4, 'Williams', 'Emily', 'Sales Representative', 'Ms.', '1988-11-25', '2009-08-15', '101 Pine St', 'San Francisco', 'CA', '94101', 'USA', '555-234-5678', '3456', NULL, 'Some notes about Emily', 3, 'emily.jpg'),

(5, 'Brown', 'Michael', 'Accountant', 'Mr.', '1987-07-12', '2012-01-22', '222 Cedar St', 'Boston', 'MA', '02101', 'USA', '555-345-6789', '4567', NULL, 'Some notes about Michael', 1, 'michael.jpg'),

(6, 'Davis', 'Jennifer', 'Sales Representative', 'Ms.', '1992-02-05', '2013-06-30', '333 Maple St', 'Dallas', 'TX', '75201', 'USA', '555-789-0123', '7890', NULL, 'Some notes about Jennifer', 3, 'jennifer.jpg'),

(7, 'Martinez', 'Daniel', 'Sales Representative', 'Mr.', '1989-04-30', '2011-11-11', '444 Oak St', 'Miami', 'FL', '33101', 'USA', '555-890-1234', '8901', NULL, 'Some notes about Daniel', 3, 'daniel.jpg'),

(8, 'Garcia', 'Sarah', 'Accounting Manager', 'Ms.', '1983-01-08', '2008-03-25', '555 Birch St', 'Seattle', 'WA', '98101', 'USA', '555-456-7890', '1235', NULL, 'Some notes about Sarah', 5, 'sarah.jpg'),

(9, 'Harris', 'Matthew', 'Sales Representative', 'Mr.', '1991-06-18', '2014-09-14', '666 Pine St', 'Atlanta', 'GA', '30301', 'USA', '555-123-9876', '2345', NULL, 'Some notes about Matthew', 3, 'matthew.jpg'),

(10, 'Wilson', 'Jessica', 'Sales Representative', 'Ms.', '1986-12-03', '2016-07-02', '777 Cedar St', 'Houston', 'TX', '77001', 'USA', '555-234-5678', '7891', NULL, 'Some notes about Jessica', 3, 'jessica.jpg'),

(11, 'Lee', 'David', 'IT Manager', 'Mr.', '1975-08-20', '2003-12-08', '888 Elm St', 'San Diego', 'CA', '92101', 'USA', '555-789-1234', '1236', NULL, 'Some notes about David', 8, 'david.jpg'),

(12, 'Nguyen', 'Linda', 'Database Administrator', 'Ms.', '1982-04-15', '2006-05-11', '999 Oak St', 'Phoenix', 'AZ', '85001', 'USA', '555-567-8901', '5678', NULL, 'Some notes about Linda', 11, 'linda.jpg'),

(13, 'Wong', 'William', 'Software Engineer', 'Mr.', '1995-03-10', '2019-02-28', '111 Maple St', 'Denver', 'CO', '80201', 'USA', '555-123-4567', '2345', NULL, 'Some notes about William', 11, 'william.jpg'),

(14, 'Chen', 'Susan', 'Software Engineer', 'Ms.', '1994-11-08', '2018-10-15', '222 Cedar St', 'Boston', 'MA', '02101', 'USA', '555-234-5678', '7890', NULL, 'Some notes about Susan', 11, 'susan.jpg'),

(15, 'Park', 'Kevin', 'Marketing Manager', 'Mr.', '1978-09-25', '2004-06-20', '333 Pine St', 'Chicago', 'IL', '60601', 'USA', '555-456-7890', '4567', NULL, 'Some notes about Kevin', 1, 'kevin.jpg'),

(16, 'Kim', 'Helen', 'Marketing Coordinator', 'Ms.', '1989-07-07', '2015-08-10', '444 Elm St', 'Los Angeles', 'CA', '90001', 'USA', '555-345-6789', '3456', NULL, 'Some notes about Helen', 15, 'helen.jpg'),

(17, 'Liu', 'James', 'Sales Representative', 'Mr.', '1993-12-12', '2017-12-05', '555 Oak St', 'San Francisco', 'CA', '94101', 'USA', '555-789-0123', '1234', NULL, 'Some notes about James', 3, 'james.jpg'),

(18, 'Brown', 'Karen', 'Sales Representative', 'Ms.', '1984-06-30', '2008-09-15', '666 Maple St', 'Seattle', 'WA', '98101', 'USA', '555-123-9876', '5678', NULL, 'Some notes about Karen', 3, 'karen.jpg'),

(19, 'Wilson', 'Brian', 'Customer Service Manager', 'Mr.', '1976-02-18', '2002-03-10', '777 Cedar St', 'Houston', 'TX', '77001', 'USA', '555-234-5678', '7890', NULL, 'Some notes about Brian', 1, 'brian.jpg'),

(20, 'Gonzalez', 'Maria', 'Customer Service Representative', 'Ms.', '1990-05-05', '2014-04-18', '888 Birch St', 'San Diego', 'CA', '92101', 'USA', '555-456-7890', '2345', NULL, 'Some notes about Maria', 19, 'maria.jpg');

1. **Retrieve a list of employees and their managers' names.**

SELECT

e1.FirstName AS EmployeeFirstName,

e1.LastName AS EmployeeLastName,

e2.FirstName AS ManagerFirstName,

e2.LastName AS ManagerLastName

FROM employees e1

LEFT JOIN employees e2 ON e1.ReportsTo = e2.EmployeeID;

1. **Display the hierarchy for a specific employee (e.g., 'Jane Smith').**

SELECT

e1.FirstName AS EmployeeFirstName,

e1.LastName AS EmployeeLastName,

e2.FirstName AS ManagerFirstName,

e2.LastName AS ManagerLastName

FROM employees e1

LEFT JOIN employees e2 ON e1.ReportsTo = e2.EmployeeID

WHERE e1.FirstName = 'Jane' AND e1.LastName = 'Smith';

1. **Find the highest-level manager (top-level) in the organization.**

SELECT

e1.FirstName AS TopManagerFirstName,

e1.LastName AS TopManagerLastName

FROM employees e1

LEFT JOIN employees e2 ON e1.EmployeeID = e2.ReportsTo

WHERE e2.EmployeeID IS NULL;

1. **List the employees who are managed by 'John Doe'.**

SELECT

e1.FirstName AS EmployeeFirstName,

e1.LastName AS EmployeeLastName

FROM employees e1

INNER JOIN employees e2 ON e1.ReportsTo = e2.EmployeeID

WHERE e2.FirstName = 'John' AND e2.LastName = 'Doe';

1. **Calculate the total number of subordinates for each manager.**

SELECT

m.FirstName AS ManagerFirstName,

m.LastName AS ManagerLastName,

COUNT(e.EmployeeID) AS NumberOfSubordinates

FROM employees m

LEFT JOIN employees e ON m.EmployeeID = e.ReportsTo

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

1. **Find all employees who have at least one subordinate.**

SELECT

DISTINCT e1.FirstName AS ManagerFirstName,

e1.LastName AS ManagerLastName

FROM employees e1

INNER JOIN employees e2 ON e1.EmployeeID = e2.ReportsTo;

1. **Retrieve a list of employees and their immediate subordinates.**

SELECT

m.FirstName AS ManagerFirstName,

m.LastName AS ManagerLastName,

s.FirstName AS SubordinateFirstName,

s.LastName AS SubordinateLastName

FROM employees m

LEFT JOIN employees s ON m.EmployeeID = s.ReportsTo;

1. **Display the hierarchy for all employees in a tabular format with indentation**.

WITH RECURSIVE EmployeeHierarchy AS (

SELECT EmployeeID, FirstName, LastName, ReportsTo, 1 AS Level

FROM employees

WHERE ReportsTo IS NULL

UNION ALL

SELECT e.EmployeeID, e.FirstName, e.LastName, e.ReportsTo, eh.Level + 1

FROM employees e

INNER JOIN EmployeeHierarchy eh ON e.ReportsTo = eh.EmployeeID

)

SELECT CONCAT(REPEAT(' ', Level - 1), FirstName) AS EmployeeName,

CONCAT(REPEAT(' ', Level - 2), ManagerFirstName) AS ManagerName

FROM EmployeeHierarchy;

1. **Find all employees who have no subordinates.**

SELECT

e1.FirstName AS EmployeeFirstName,

e1.LastName AS EmployeeLastName

FROM employees e1

LEFT JOIN employees e2 ON e1.EmployeeID = e2.ReportsTo

WHERE e2.EmployeeID IS NULL;

1. **Find the employees with the same manager.**

SELECT

e1.FirstName AS Employee1FirstName,

e1.LastName AS Employee1LastName,

e2.FirstName AS Employee2FirstName,

e2.LastName AS Employee2LastName,

m.FirstName AS ManagerFirstName,

m.LastName AS ManagerLastName

FROM employees e1

INNER JOIN employees e2 ON e1.ReportsTo = e2.ReportsTo

INNER JOIN employees m ON e1.ReportsTo = m.EmployeeID

WHERE e1.EmployeeID <> e2.EmployeeID

ORDER BY m.FirstName, m.LastName;

1. **Retrieve a list of employees and their immediate managers who are not at the top level.**

SELECT

e1.FirstName AS EmployeeFirstName,

e1.LastName AS EmployeeLastName,

e2.FirstName AS ManagerFirstName,

e2.LastName AS ManagerLastName

FROM employees e1

INNER JOIN employees e2 ON e1.ReportsTo = e2.EmployeeID

WHERE e2.ReportsTo IS NOT NULL;