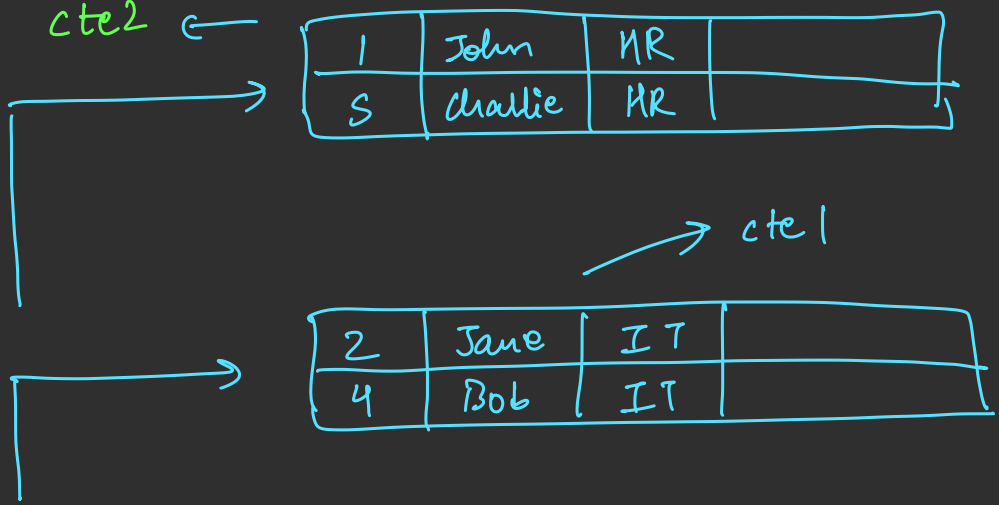


	employee_id [PK] integer	name character varying (100)	department character varying (100)	salary integer
1	1	John Doe	HR	50000
2	2	Jane Smith	IT	60000
3	3	Alice Johnson	Finance	55000
4	4	Bob Brown	IT	70000
5	5	Charlie White	HR	48000

WITH nameOfCte C

[Any query → temporary result of this → cte

)

Select * from nameOfCte]

```
WITH cte1 AS (
  Select employee_id, salary, 'HIGH' as category
  WHERE salary >= 70000
),
cte2 AS (
  Select employee_id, salary, 'Medium' as category
  WHERE salary >= 55000 AND salary < 70000
),
cte3 as (
  Select employee_id, salary, 'LOW' as category
  WHERE salary < 55000
)
Select * from cte1 ---> 1 row
UNION ALL
Select * from cte2 ---> 2 row
```

	employee_id [PK] integer	name character varying (100)	department character varying (100)	salary integer
1	1	John Doe	HR	50000
2	2	Jane Smith	IT	60000
3	3	Alice Johnson	Finance	55000
4	4	Bob Brown	IT	70000
5	5	Charlie White	HR	48000

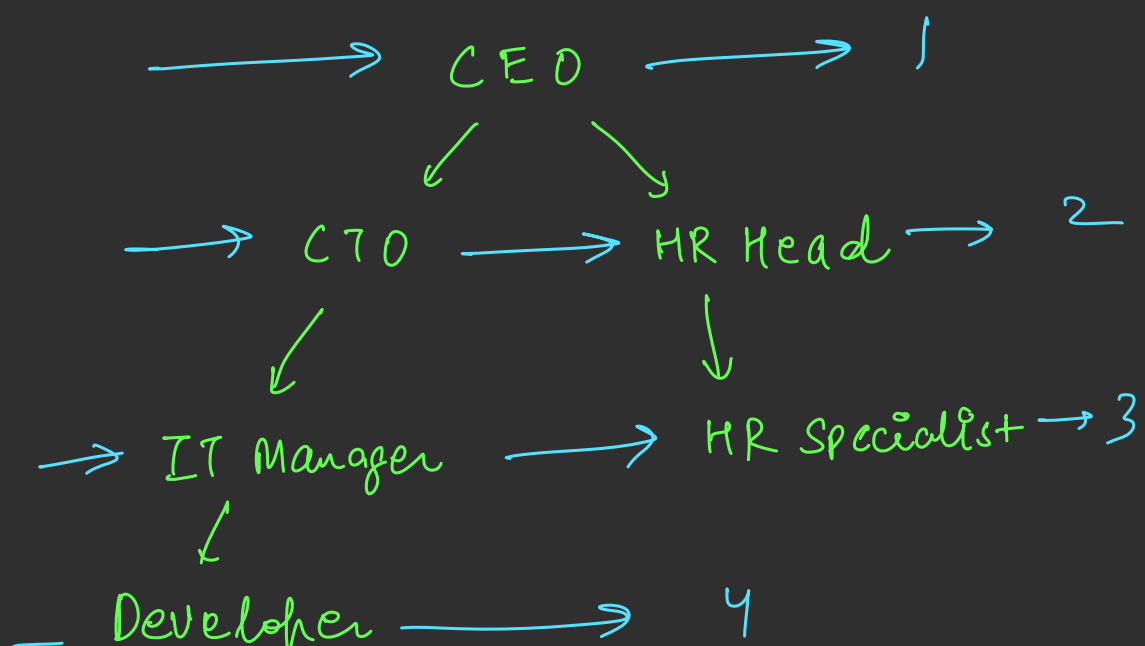
Multiple CTEs

```
WITH cte1 AS (
    Select * from employee
    WHERE department LIKE 'IT'
),
cte2 AS (
    Select * from employee
    WHERE department LIKE 'HR'
)
Select * from cte1
UNION ALL
Select * from cte2
```

```
56  SELECT * FROM employee_hierarchy
57
```

Data Output | Messages | Notifications

	employee_id [PK] integer	name character varying (100)	manager_id integer
1	1	CEO	[null]
2	2	CTO	1
3	3	HR Head	1
4	4	IT Manager	2
5	5	HR Specialist	3
6	6	Developer	4



Two Case →

rCte

1	CEO	1

→ Base Case → starting ROW

```
Select employee_id, name, 1 as level
  from employee
  WHERE manger_id is NULL
```

→ Recursive Case → dependent row

```
Select employee_id, name, level + 1
from employee as eE
JOIN rcte ON eE.mangerid =
                rcte.employee_id
```

2	CEO	2
3	HR Head	2

employee_id,	name,	level
1	CEO	1
2	CTO	2
3	HR Head	2
4	IT Manager	3
5	HR specialist	3
6	Developer	4

Recursive CTE

SQL

1
2
3 → x
4 → x + 1
5

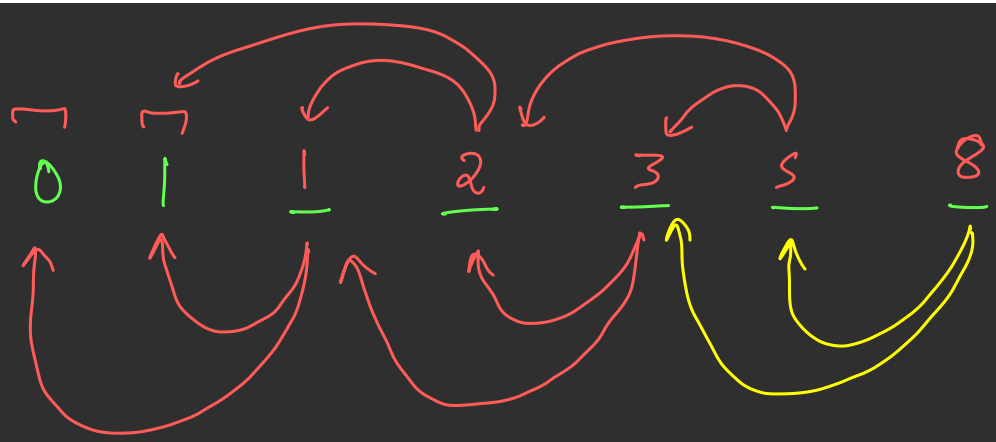
n

1 _ _ _ . n

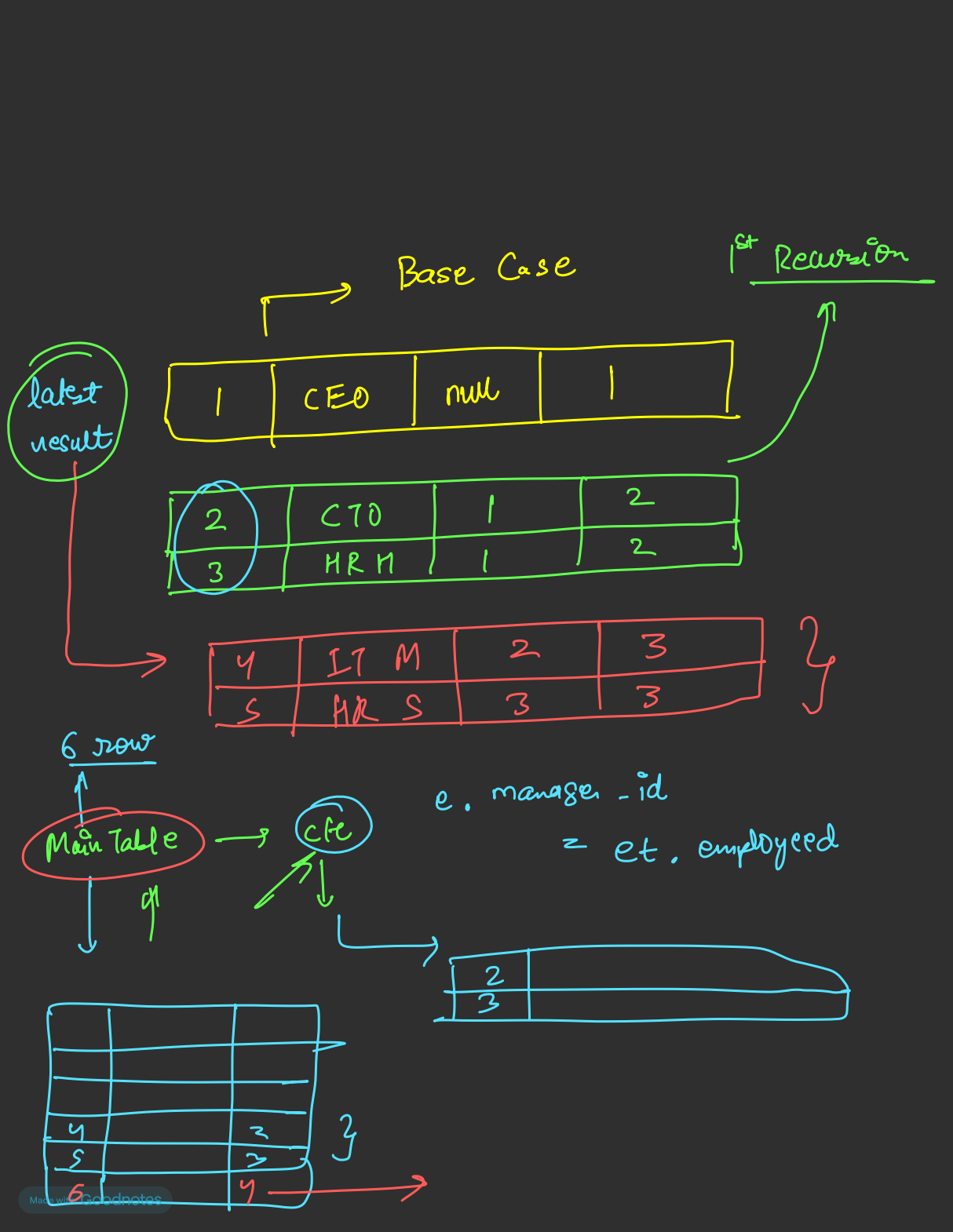
Most Imp

```
WITH RECURSIVE employee_tree AS (
    -- Base case: start with the CEO (the top level)
    SELECT employee_id, name, manager_id, 1 AS level
    FROM employee_hierarchy
    WHERE manager_id IS NULL

    UNION ALL

    -- Recursive case: find employees who report to the employees in the previous result
    SELECT e.employee_id, e.name, e.manager_id, et.level + 1
    FROM employee_hierarchy e
    INNER JOIN employee_tree et ON e.manager_id = et.employee_id
)
SELECT employee_id, name, level
FROM employee_tree
ORDER BY level;
```

ignored by compiler



$$F(n) = F(n-1) + F(n-2)$$


Base Case
1st Recursion
latest result
1 | CEO | null | 1
2 | CTO | 1 | 2
3 | HRM | 1 | 2
4 | IT M | 2 | 3
5 | HR S | 3 | 3
6 row
Main Table
cte
e. manager_id = et. employeed
2
3
4 3
5 3
6 4

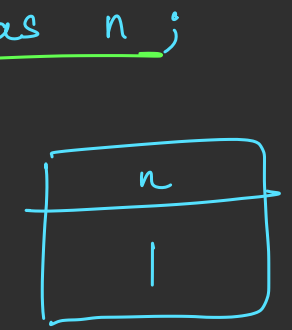
WITH RECURSIVE rcte AS (

Select 1 as n ;

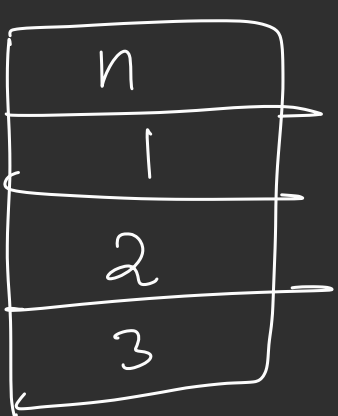
UNION ALL

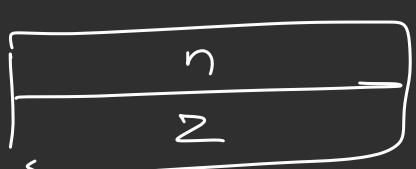
Select n + 1 from rcte ;

)



1
2
3
4
5
6
7
8
9
10





```
WITH RECURSIVE tens AS (
    SELECT 1 as n
 UNION ALL
    SELECT n+1 FROM tens
)
SELECT n FROM tens limit 10;
```

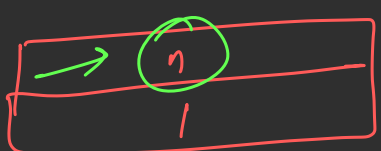
Base Case

Recursive Case

WHERE n < 10;

STOP

HARD



discarel

latest row generakd

100	1	1	1	0
200	2	2	2	0.16
200	3	2	2	0.16
300	4	4	3	0.5
500	5	5	4	
500	6	5	4	
700	7	7	5	

RANK

Dense_Rank