Recursive CTEs *in SQL*

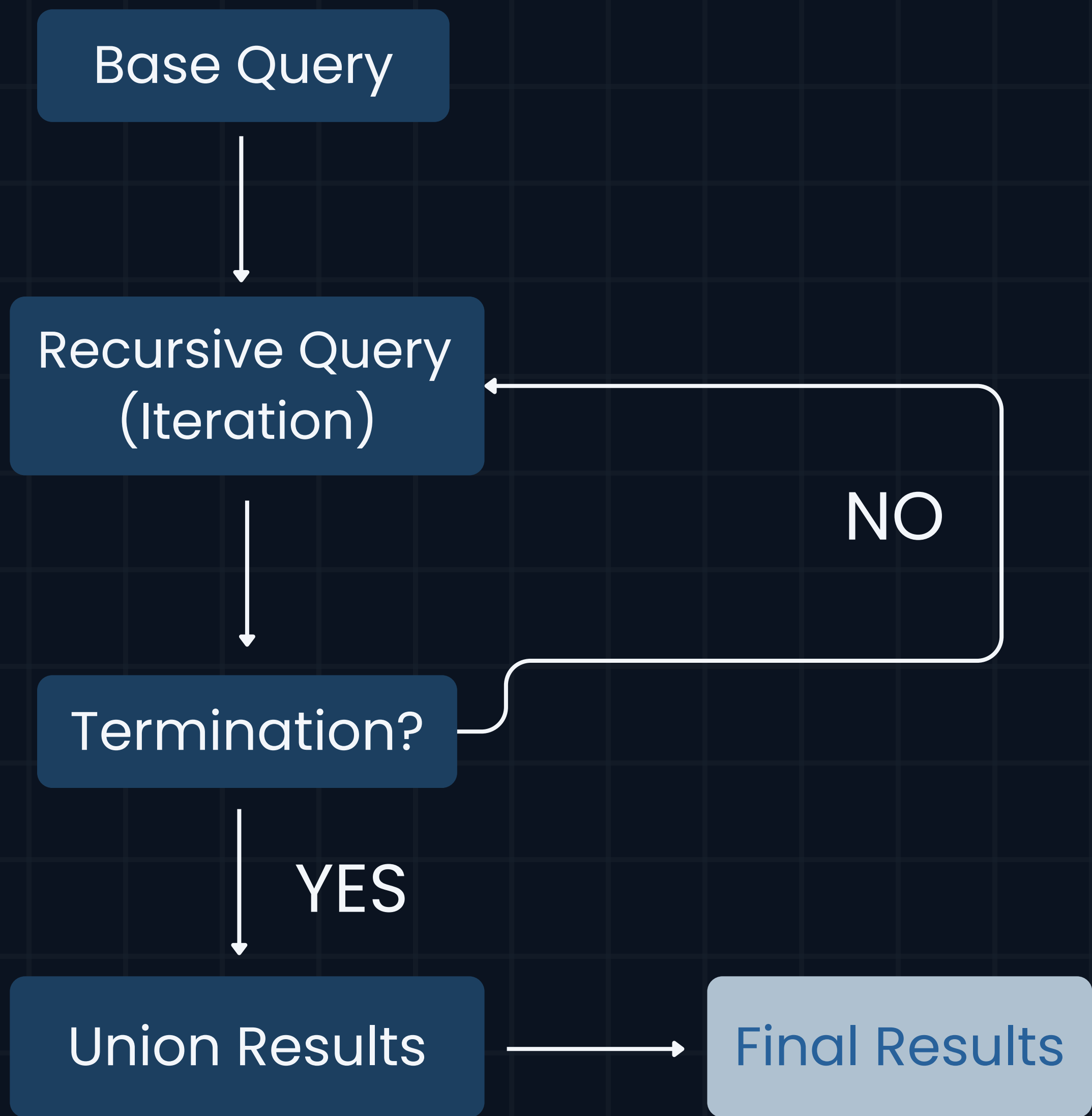
@cameron-css
Cameron Seamons - Data Analyst

What is a Recursive CTE?

A recursive CTE references itself during the execution. It returns a subset of the result, then it recursively references itself and terminates when it has returned all the results.

Basically a query you want to run again and again adding data until you tell it to stop.

Recursion Diagram



What is an Anchor Member? a.k.a Base Query

A recursive CTE starts it's execution with the anchor member(base query) which is a non recursive query.

It creates a table with the initial rows of data that serves as the starting point for the recursion.

Base Query Example

```
SELECT
  num,
  1 AS factorial
FROM numbers
WHERE num = 1
```

Result:

num	factorial
1	1

What is the Recursive Member? a.k.a The Iterations

The recursive member mainly consists of the SELECT statement that references the CTE itself.

Each iteration uses the results that were obtained in the previous iteration (or builds upon the initial base Query).

This process is repeated until a termination condition is met.

Iteration Example

```sql
SELECT
  n.num,
  n.num * c.factorial AS factorial
FROM numbers n
JOIN CTE c
ON n.num = c.num + 1
WHERE n.num <= 4
```

Result:

num	factorial
1	1
2	2
3	6
4	24

Example query Explained

Our base query gives us a base table with num 1 and factorial 1.

Our recursion then goes through, adds 1 to num making it 2.
And then multiplies the new num by the last factorial. 2 * 1, also giving us 2.

It contiunes looping back and repeating until it is terminated by the WHERE clause (WHERE n.num <= 4)

Resulting in a final table of 1 - 4 with their factorials.

Termination!!!

Make sure you build a terminator statement into your recursive query or else the query can get stuck in an infinite loop.

Most of the modern SQL engines have a built in safety net to prevent infinite loops. But its best to get into practice of including a terminator statement.
(i.e. WHERE n.num <= 4)

NOTE: PostgreSQL Does not have a default recursion limit.

UNION ALL

The UNION ALL operator combines the base query table with the iteration results

Full Query Example

```
WITH RECURSIVE cte AS (
SELECT
    num,
    1 AS factorial
FROM numbers
WHERE num = 1

UNION ALL

SELECT
    n.num,
    n.num * c.factorial AS factorial
FROM numbers n
JOIN CTE c
ON n.num = c.num + 1
WHERE n.num <= 4
)
SELECT * FROM cte;
```

This is the Base Query

This is the Recursive Query. a.k.a. Iteration

Final Note :

MySQL, PostgreSQL, and SQLite require you to explicitly state **RECURSIVE** in the cte WITH declaration clause.

SQL server does not.

MySQL, PostgreSQL, and SQLite:

WITH **RECURSIVE** query_example AS

Microsoft SQL Server:

WITH query_example AS

Example Usage

Recursive CTEs are a great tool to use in specific situations.

- Rolling Totals
- Hierarchical data
- Iterative calculations
- Sequence generation
- Finding ancestors/descendants

Have you ever Used a recursive CTE ??

Follow for more SQL tips

@cameron-css

Cameron Seamons - Data Analyst