Relational DB & SQL - C11

Common Table Expressions

To do: Go through the activity to the end

CTE vs. Subquery

Common Table Expressions and Subqueries are mostly used for the same purpose. But the main advantage of the Common Table Expression (when not using it for recursive queries) is encapsulation, instead of having to declare the sub-query in every place you wish to use it, you are able to define it once, but have multiple references to it.

Also CTEs are more common in practice, as they tend to be cleaner for someone (who didn't write the query) to follow the logic. In this context, we can say that CTEs are more readable.

The performance of CTEs and subqueries should, in theory, be the same since both provide the same information to the query optimizer. One difference is that a CTE used more than once could be easily identified and calculated once. The results could then be stored and read multiple times.

Let's compare two different queries with the same result, one written using Common Table Expression and the other using Subquery:

"departments" table:

```
id      name      dept_name         seniority    graduation   salary
------  --------  ----------------  --------     ----------   ------
10238   Eric      Economics         Experienced  MSc          72000
13378   Karl      Music             Candidate    BSc          42000
23493   Jason     Philosophy        Candidate    MSc          45000
36299   Jane      Computer Science  Senior       PhD          91000
30766   Jack      Economics         Experienced  BSc          68000
40284   Mary      Psychology        Experienced  MSc          78000
43087   Brian     Physics           Senior       PhD          93000
53695   Richard   Philosophy        Candidate    PhD          54000
58248   Joseph    Political Science Experienced  BSc          58000
63172   David     Art History       Experienced  BSc          65000
64378   Elvis     Physics           Senior       MSc          87000
96945   John      Computer Science  Experienced  MSc          80000
99231   Santosh   Computer Science  Experienced  BSc          74000
```

CTE:

```sql
WITH t1 AS
(
SELECT *
FROM departments
WHERE dept_name = 'Computer Science'
),
t2 as
(
SELECT *
FROM departments
WHERE dept_name = 'Physics'
)
SELECT d.name, t1.graduation AS graduation_CS, t2.graduation AS
    graduation_PHY
FROM departments as d
LEFT JOIN t1
ON d.id = t1.id
LEFT JOIN t2
ON d.id = t2.id
WHERE t1.graduation IS NOT NULL
OR    t2.graduation IS NOT NULL
ORDER BY 2 DESC, 3 DESC
```

output :

```
name     graduation_CS  graduation_PHY
-------  -------------  --------------
Jane     PhD            NULL
John     MSc            NULL
Santosh  BSc            NULL
Brian    NULL           PhD
Elvis    NULL           MSc
```

Subquery:

```sql
SELECT d.name, t1.graduation AS graduation_CS, t2.graduation AS
    graduation_PHY
FROM departments as d
LEFT JOIN
(
SELECT *
FROM departments
WHERE dept_name = 'Computer Science'
) AS t1
ON d.id = t1.id
LEFT JOIN
(
SELECT *
FROM departments
WHERE dept_name = 'Physics'
) AS t2
ON d.id = t2.id
WHERE t1.graduation IS NOT NULL
ON    t2.graduation IS NOT NULL
ORDER BY 2 DESC, 3 DESC
```

result:

```
name     graduation_CS  graduation_PHY
-------  -------------  --------------
Jane     PhD            NULL
John     MSc            NULL
Santosh  BSc            NULL
Brian    NULL           PhD
Elvis    NULL           MSc
```

In the CTE query, the compiler knows you're querying the same data set since it has saved it (albeit temporarily) as *temp_table*. In the second query, even though the SQL is the exact same, the compiler does not realize they're the same query until it runs them. Notice that we have to call the same query by the two distinct aliases: *t1* and *t2*. Not only does this query take more compute and contain redundancy, it also forces us to call the same query two different names. This is misleading.

As an advice would be to only use subqueries in Adhoc queries when you need results quickly. If the query is going to be read by others, run every day, or reused, try to use a CTE for readability and performance.

Previous

Next

You have completed 50% of the lesson

50%

Jump to...

Reset user tour on this page