Data Analytics

–

CTEs

gamma tech school

1. ¿Qué es una CTE?

- Common Table Expression (CTE)
- Query temporal que se puede referenciar en la query principal.
- Simplifica consultas complejas.
- Hace el código más legible.
- Evita repetir subqueries.

```sql
WITH tabla_CTE_1 AS (
    SELECT
        col_1,
        col_2
    FROM tabla)
, tabla_CTE_2 AS (
    SELECT
        col_1,
        col_3
    FROM tabla)

SELECT *
FROM tabla_CTE_1 cte1
JOIN tabla_CTE_2 cte2
    ON cte1.col_1 = cte2.col_1;
```

2. ¿Por qué utilizar una subconsulta?

- Facilita la legibilidad y organización del código.
- Permiten reutilizar código en una misma query sin necesidad de repetir subconsultas.
- Facilitan la recursividad en datos jerárquicos.
- Mejoran el rendimiento.
- Permiten aislar subconsultas temporales sin tener que crear tablas temporales para ello.

```
WITH tabla_CTE_1 AS (
    SELECT
        col_1,
        col_2
    FROM tabla)
, tabla_CTE_2 AS (
    SELECT
        col_1,
        col_3
    FROM tabla)

SELECT *
FROM tabla_CTE_1 cte1
JOIN tabla_CTE_2 cte2
    ON cte1.col_1 = cte2.col_1;
```

3. Ejemplo con classicmodels

```sql
WITH ventasEmpleados AS (
    SELECT
        salesRepEmployeeNumber AS employeeNumber
        , SUM(quantityOrdered * priceEach) AS totalSales
    FROM
        orders o
        JOIN orderdetails od ON o.orderNumber = od.orderNumber
        JOIN customers c ON c.customerNumber = o.customerNumber
    GROUP BY
        salesRepEmployeeNumber
)

SELECT
    e.employeeNumber
    , CONCAT(e.firstName, ' ', e.lastName) AS employeeName
    , ROUND(ve.totalSales,0) as roundSales
FROM
    employees e
    LEFT JOIN ventasEmpleados ve ON e.employeeNumber = ve.employeeNumber
ORDER BY
    ve.totalSales DESC;
```

4. Ejemplo con sakila

Dinero generado por los ingresos de las películas en las que ha participado cada actor

```sql
WITH peliculas_actor AS (
    SELECT
        a.actor_id
        , a.first_name
        , a.last_name
        , COUNT(fa.film_id) AS total_peliculas
    FROM
        actor a
    JOIN
        film_actor fa ON a.actor_id = fa.actor_id
    GROUP BY
        a.actor_id, a.first_name, a.last_name
)
, peliculas_ingresos AS (
    SELECT
        fa.actor_id
        , fa.film_id
        , SUM(p.amount) AS total_ingresos
    FROM
        film_actor fa
        JOIN inventory i ON fa.film_id = i.film_id
        JOIN rental r ON i.inventory_id = r.inventory_id
        JOIN payment p ON r.rental_id = p.rental_id
    GROUP BY
        fa.actor_id, fa.film_id
)

SELECT
    pa.actor_id
    , pa.first_name
    , pa.last_name
    , pa.total_films
    , SUM(pi.total_ingresos) AS total_ingresos
FROM
    peliculas_actor pa
    JOIN peliculas_ingresos pi ON pa.actor_id = pi.actor_id
GROUP BY
    pa.actor_id, pa.first_name, pa.last_name, pa.total_films
```