

3.9 Common Table Expressions

1. Step 1: Answer the business questions from step 1 and 2 of task 3.8 using CTEs

- Copy the Rewrite your queries from steps 1 and 2 of task 3.8 as CTEs.
- Copy-paste your CTEs and their outputs into your answers document.
- Write 2 to 3 sentences explaining how you approached this step, for example, what you did first, second, and so on.

```
WITH total_amount_paid (payment) AS
  (SELECT SUM(pay.amount) AS payment
   FROM customer as cu
   INNER JOIN address AS ad ON cu.address_id = ad.address_id
   INNER JOIN city AS ci ON ad.city_id = ci.city_id
   INNER JOIN country AS co ON ci.country_id = co.country_id
   INNER JOIN payment AS pay ON cu.customer_id = pay.customer_id
   WHERE city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule(Dhulia)', 'Kurashiki',
                  'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo'))

SELECT AVG(payment) AS average
FROM total_amount_paid;
```

Data output		Messages	Notifications
	average numeric		
1	107.3540000		

```

WITH top_5_customers (customer_id, first_name, last_name, city, country, total_amount_paid) AS
  (SELECT cu.customer_id,
    cu.first_name,
    cu.last_name,
    ci.city,
    co.country,
    SUM(pay.amount) AS total_amount_paid
  FROM customer AS cu
  INNER JOIN address AS ad ON cu.address_id = ad.address_id
  INNER JOIN city AS ci ON ad.city_id = ci.city_id
  INNER JOIN country AS co ON ci.country_id = co.country_id
  INNER JOIN payment AS pay ON cu.customer_id = pay.customer_id
  WHERE city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule(Dhulia)', 'Kurashiki',
    'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
  GROUP BY cu.customer_id, first_name, last_name, city, country
  ORDER BY total_amount_paid DESC
  LIMIT 5)

SELECT co.country,
  COUNT(DISTINCT cu.customer_id) AS all_customer_count,
  COUNT(DISTINCT co.country) AS top_customer_count
FROM country AS co
INNER JOIN city AS ci ON co.country_id = ci.country_id
INNER JOIN address AS ad ON ci.city_id = ad.city_id
INNER JOIN customer AS cu ON ad.address_id = cu.address_id
LEFT JOIN top_5_customers ON co.country = top_5_customers.country
GROUP BY co.country, top_5_customers
ORDER BY all_customer_count DESC;

```

Data output Messages Notifications			
<div> <div>≡+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> </div>			
	country character varying (50) 🔒	all_customer_count bigint 🔒	top_customer_count bigint 🔒
1	India	60	1
2	China	53	1
3	United States	36	1
4	Japan	31	1
5	Mexico	30	1
6	Mexico	30	1
7	Brazil	28	1
8	Russian Federation	28	1
9	Philippines	20	1
10	Turkey	15	1
Total rows: 109 of 109		Query complete 00:00:00.086	

As the tables are already located from the previous task, I proceed with only converting it to CTE. It is easier for me to think from inside to outside, meaning I define firstly the deepest query (inside WITH clause) and then close it with parenthesis. Afterwards, think about the outer layer which is the main statement.

2. Step 2: Compare the performance of your CTEs and subqueries.

- Which approach do you think will perform better and why?
CTE is indeed better in terms of readability and not cluttering which makes also easier for debugging. For me personally, I could think more systematically when using CTE compared to subquery.
- Compare the costs of all the queries by creating query plans for each one.

Question 1

CTE		Subquery	
Data output	Messages	Data output	Messages
<div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> </div>		<div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> </div>	
	QUERY PLAN text		QUERY PLAN text
1	Aggregate (cost=63.27..63.28 rows=1 width=32)	1	Aggregate (cost=63.27..63.28 rows=1 width=32)
2	-> Limit (cost=63.19..63.21 rows=5 width=36)	2	-> Limit (cost=63.19..63.21 rows=5 width=36)
3	-> Sort (cost=63.19..63.80 rows=244 width=36)	3	-> Sort (cost=63.19..63.80 rows=244 width=36)
4	Sort Key: (sum(pay.amount)) DESC	4	Sort Key: (sum(pay.amount)) DESC
5	-> HashAggregate (cost=56.09..59.14 rows=244 width=36)	5	-> HashAggregate (cost=56.09..59.14 rows=244 width=36)
6	Group Key: cu.customer_id	6	Group Key: cu.customer_id
7	-> Nested Loop (cost=18.16..54.87 rows=244 width=10)	7	-> Nested Loop (cost=18.16..54.87 rows=244 width=10)
8	-> Hash Join (cost=17.88..37.14 rows=10 width=4)	8	-> Hash Join (cost=17.88..37.14 rows=10 width=4)
9	Hash Cond: (ci.country_id = co.country_id)	9	Hash Cond: (ci.country_id = co.country_id)
10	-> Nested Loop (cost=14.43..33.66 rows=10 width=6)	10	-> Nested Loop (cost=14.43..33.66 rows=10 width=6)
11	-> Hash Join (cost=14.15..29.77 rows=10 width=6)		
Total rows: 22 of 22 Query complete 00:00:00.073		Total rows: 22 of 22 Query complete 00:00:00.063	

Question 2

CTE		Subquery	
Data output	Messages	Data output	Messages
<div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> </div>		<div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> <div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div> </div>	
	QUERY PLAN text		QUERY PLAN text
1	Sort (cost=195.70..197.06 rows=545 width=84)	1	Sort (cost=195.70..197.06 rows=545 width=84)
2	Sort Key: (count(DISTINCT cu.customer_id)) DESC	2	Sort Key: (count(DISTINCT cu.customer_id)) DESC
3	-> GroupAggregate (cost=157.99..170.93 rows=545 width=84)	3	-> GroupAggregate (cost=157.99..170.93 rows=545 width=84)
4	Group Key: co.country, top_5_customers.*	4	Group Key: co.country, top_5_customers.*
5	-> Sort (cost=157.99..159.49 rows=599 width=72)	5	-> Sort (cost=157.99..159.49 rows=599 width=72)
6	Sort Key: co.country, top_5_customers.*	6	Sort Key: co.country, top_5_customers.*
7	-> Hash Left Join (cost=108.06..130.36 rows=599 width=72)	7	-> Hash Left Join (cost=108.06..130.36 rows=599 width=72)
8	Hash Cond: ((co.country)::text = (top_5_customers.country)::text)	8	Hash Cond: ((co.country)::text = (top_5_customers.country)::text)
9	-> Hash Join (cost=43.52..63.30 rows=599 width=13)	9	-> Hash Join (cost=43.52..63.30 rows=599 width=13)
10	Hash Cond: (ci.country_id = co.country_id)	10	Hash Cond: (ci.country_id = co.country_id)
11	-> Hash Join (cost=40.07..58.22 rows=500 width=6)	11	-> Hash Join (cost=40.07..58.22 rows=500 width=6)
Total rows: 44 of 44 Query complete 00:00:00.077		Total rows: 44 of 44 Query complete 00:00:00.110	

- The EXPLAIN command gives you an estimated cost. To find out the actual speed of your queries, run them in pgAdmin 4. After each query has been run, a pop-up window will display its speed in milliseconds.
- Did the results surprise you? Write a few sentences to explain your answer.
Surprisingly, the cost for both CTE and subquery are the same. However, that is not always the case with the actual speed. Actual speed using subquery is faster for the question 1 and slower for the question 2 in comparison to the CTE. Not to mention, the first run takes always longer. As the queries cached, its actual time is reduced.