3.9 Common Table Expressions

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

1. Rewrite your queries from steps 1 and 2 of task 3.8 as CTEs.
2. Copy-paste your CTEs and their outputs into your answers document

First query from 3.8 exercise

Used explain to get the cost.

Query | Query History

```
1  Explain Select avg(total_amount_paid.total_amount_paid) as average_amount_paid
2  from   (
3  Select A.customer_id,
4  A.first_name,
5  A.last_name,
6  D.city,
7  E.country,
8  SUM(B.amount) as total_amount_paid
9  From Customer A
10 Inner join payment B on A.customer_id = B.customer_id
11 Inner join address c on A.address_id = C.address_id
12 Inner join city D on C.city_id = D.city_id
13 Inner join country E on D.country_id = E.country_id
14 where D.city IN ('Aurora', 'Atlixco','Xintai', 'Adoni', 'Dhule (Dhulia)',
15                  'Kurashiki','Pingxang', 'Sivas', 'Celaya', 'So Leopoldo')
16 Group by A.customer_id, A.first_name, A.last_name, D.city, E.Country
17 Order by  total_amount_paid Desc
18 Limit 5) as total_amount_paid
```

Cost of first subquery 3.8

	QUERY PLAN text
1	Aggregate (cost=64.45..64.46 rows=1 width=32)
2	-> Limit (cost=64.37..64.39 rows=5 width=67)
3	-> Sort (cost=64.37..64.98 rows=243 width=67)
4	Sort Key: (sum(b.amount)) DESC
5	-> HashAggregate (cost=57.30..60.34 rows=243 width=67)
6	Group Key: a.customer_id, d.city, e.country
7	-> Nested Loop (cost=18.16..54.87 rows=243 width=41)
8	-> Hash Join (cost=17.88..37.14 rows=10 width=35)
9	Hash Cond: (d.country_id = e.country_id)
10	-> Nested Loop (cost=14.43..33.66 rows=10 width=28)
11	-> Hash Join (cost=14.15..29.77 rows=10 width=15)
12	Hash Cond: (c.city_id = d.city_id)
13	-> Seq Scan on address c (cost=0.00..14.03 rows=603 width=6)
14	-> Hash (cost=14.03..14.03 rows=10 width=15)
15	-> Seq Scan on city d (cost=0.03..14.03 rows=10 width=15)
16	Filter: ((city)::text = ANY ('{Aurora,Atlixco,Xintai,Adoni,"Dhule (Dhulia)",Kurashiki,Pingxang,Sivas,Celaya,"So Leopoldo"}'::tex...
17	-> Index Scan using idx_fk_address_id on customer a (cost=0.28..0.38 rows=1 width=19)
18	Index Cond: (address_id = c.address_id)
19	-> Hash (cost=2.09..2.09 rows=109 width=13)
20	-> Seq Scan on country e (cost=0.00..2.09 rows=109 width=13)

Common table expression based on the above query

Query | Query History

```
Explain WITH CTE_average_amount_paid (customer_id,
first_name,
last_name,
city,
total_amount_paid)
AS
(SELECT A.customer_id,
A.first_name,
A.last_name,
C.city,
SUM(E.amount) AS total_amount_paid
FROM payment E
INNER JOIN customer A ON E.customer_id=A.customer_id
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
WHERE C.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)',
                 'Kurashiki', 'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
GROUP BY A.customer_id, D.country, C.city
ORDER BY total_amount_paid DESC
LIMIT 5)
SELECT  AVG (total_amount_paid)
FROM CTE_average_amount_paid

```

Cost of CTE 3.9

Step

	QUERY PLAN text
1	Aggregate (cost=64.45..64.46 rows=1 width=32)
2	-> Limit (cost=64.37..64.39 rows=5 width=270)
3	-> Sort (cost=64.37..64.98 rows=243 width=270)
4	Sort Key: (sum(e.amount)) DESC
5	-> HashAggregate (cost=57.30..60.34 rows=243 width=270)
6	Group Key: a.customer_id, d.country, c.city
7	-> Nested Loop (cost=18.16..54.87 rows=243 width=28)
8	-> Hash Join (cost=17.88..37.14 rows=10 width=22)
9	Hash Cond: (c.country_id = d.country_id)
10	-> Nested Loop (cost=14.43..33.66 rows=10 width=15)
11	-> Hash Join (cost=14.15..29.77 rows=10 width=15)
12	Hash Cond: (b.city_id = c.city_id)
13	-> Seq Scan on address b (cost=0.00..14.03 rows=603 width=6)
14	-> Hash (cost=14.03..14.03 rows=10 width=15)
15	-> Seq Scan on city c (cost=0.03..14.03 rows=10 width=15)
16	Filter: ((city)::text = ANY ('{Aurora,Atlixco,Xintai,Adoni,"Dhule (Dhulia)",Kurashiki,Pingxiang,Sivas,Cel...
17	-> Index Scan using idx_fk_address_id on customer a (cost=0.28..0.38 rows=1 width=6)
18	Index Cond: (address_id = b.address_id)
19	-> Hash (cost=2.09..2.09 rows=109 width=13)
20	-> Seq Scan on country d (cost=0.00..2.09 rows=109 width=13)
21	-> Index Scan using idx_fk_customer_id on payment e (cost=0.29..1.53 rows=24 width=8)
22	Index Cond: (customer_id = a.customer_id)

Step 2

Second subquery from 3.8

Query | Query History

```
Explain select DISTINCT (A.country),
count(distinct D.customer_id) as all_customer_count,
count (distinct A.country)as top_customer_count
from country A
inner join city B on A.country_id=B.country_id
inner join address C on B.city_id = C.city_id
inner join customer D on C.address_id = D.address_id
Left join(
Select A.customer_id,
A.first_name,
A.last_name,
D.city,
E.country,
SUM(B.amount) as total_amount_paid
From Customer A
Inner join payment B on A.customer_id = B.customer_id
Inner join address c on A.address_id = C.address_id
Inner join city D on C.city_id = D.city_id
Inner join country E on D.country_id = E.country_id
where D.city IN ('Aurora', 'Atlixco','Xintai', 'Adoni', 'Dhule (Dhulia)',
                    'Kurashiki','Pingxang', 'Sivas', 'Celaya', 'So Leopoldo')

AND E.country in ('India','China', 'United states', 'Japan','Mexico','Brazil',
                  'Russian Fedration','Philippines', 'Turkey', 'Indonesia')
Group by A.customer_id, A.first_name, A.last_name, D.city, E.Country
Order by  total_amount_paid Desc
Limit 5 )as top_5_customers on A.country = top_5_customers.country
Group by A.country, top_5_customers
order by all_customer_count desc
Limit 5;
```

Cost of subquery

	QUERY PLAN text
1	Limit (cost=150.46..150.47 rows=5 width=84)
2	-> Sort (cost=150.46..151.82 rows=545 width=84)
3	Sort Key: (count(DISTINCT d.customer_id)) DESC
4	-> HashAggregate (cost=135.96..141.41 rows=545 width=84)
5	Group Key: count(DISTINCT d.customer_id), a.country, count(DISTINCT a.country)
6	-> GroupAggregate (cost=118.93..131.87 rows=545 width=84)
7	Group Key: a.country, top_5_customers.*
8	-> Sort (cost=118.93..120.43 rows=599 width=72)
9	Sort Key: a.country, top_5_customers.*
10	-> Hash Left Join (cost=69.00..91.30 rows=599 width=72)
11	Hash Cond: ((a.country)::text = (top_5_customers.country)::text)
12	-> Hash Join (cost=43.52..63.30 rows=599 width=13)
13	Hash Cond: (b.country_id = a.country_id)
14	-> Hash Join (cost=40.07..58.22 rows=599 width=6)
15	Hash Cond: (c.city_id = b.city_id)
16	-> Hash Join (cost=21.57..38.14 rows=599 width=6)
17	Hash Cond: (d.address_id = c.address_id)
18	-> Seq Scan on customer d (cost=0.00..14.99 rows=599 width=6)
19	-> Hash (cost=14.03..14.03 rows=603 width=6)
20	-> Seq Scan on address c (cost=0.00..14.03 rows=603 width=6)
21	-> Hash (cost=11.00..11.00 rows=600 width=6)
22	-> Seq Scan on city b (cost=0.00..11.00 rows=600 width=6)

Dashboard Properties SQL Statistics Dependencies Dependents Rockbuster/postgres@PostgreSQL 14*

Rockbuster/postgres@PostgreSQL 14

No limit

Query Query History

```sql
WITH CTE_top_5_customer_count AS (SELECT A.customer_id,
A.first_name,
A.last_name,
C.city,
D. country_id,
SUM(E.amount) AS total_amount
FROM payment E
INNER JOIN customer A ON E.customer_id=A.customer_id
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
WHERE C.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)', 'Kurashiki',
                 'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
GROUP BY A.customer_id, D.country_id, C.city
ORDER BY total_amount DESC
LIMIT 5),
all_customer_count_cte AS(SELECT  D.country,
COUNT (DISTINCT A.customer_id) AS all_customer_count
FROM customer A
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
GROUP BY D.country)

SELECT  D.country,
COUNT (DISTINCT A.customer_id) AS all_customer_count,
COUNT (DISTINCT CTE_top_5_customer_count.customer_id) AS top_customer_count
FROM customer A
INNER JOIN address B ON A.address_id = B.address_id
INNER JOIN city C ON B.city_id = C.city_id
INNER JOIN country D ON C.country_id = D.country_id
LEFT JOIN CTE_top_5_customer_count ON D.country_id = CTE_top_5_customer_count.country_id
GROUP BY D.country
ORDER BY top_customer_count DESC
LIMIT 5;
```

Cost of CTE

Data output Messages Notifications

	QUERY PLAN text
1	Limit (cost=166.84..166.86 rows=5 width=25)
2	-> Sort (cost=166.84..167.12 rows=109 width=25)
3	Sort Key: (count(DISTINCT cte_top_5_customer_count.customer_id)) DESC
4	-> GroupAggregate (cost=157.95..165.03 rows=109 width=25)
5	Group Key: d.country
6	-> Sort (cost=157.95..159.45 rows=599 width=17)
7	Sort Key: d.country
8	-> Hash Left Join (cost=108.02..130.32 rows=599 width=17)
9	Hash Cond: (d.country_id = cte_top_5_customer_count.country_id)
10	-> Hash Join (cost=43.52..63.30 rows=599 width=17)
11	Hash Cond: (c.country_id = d.country_id)
12	-> Hash Join (cost=40.07..58.22 rows=599 width=6)
13	Hash Cond: (b.city_id = c.city_id)
14	-> Hash Join (cost=21.57..38.14 rows=599 width=6)
15	Hash Cond: (a.address_id = b.address_id)
16	-> Seq Scan on customer a (cost=0.00..14.99 rows=599 width=6)
17	-> Hash (cost=14.03..14.03 rows=603 width=6)
18	-> Seq Scan on address b (cost=0.00..14.03 rows=603 width=6)
19	-> Hash (cost=11.00..11.00 rows=600 width=6)

3.Write 2 to 3 sentences explaining how you approached this step, for example, what you did first, second, and so on.

-First, I define the CTE using the WITH keyword.
-I assigned a name to CTE for example CTE_top_5_customer_count.
-I wrote the subquery in the CTE body. For example, in the CTE body I selected the customer id, first name etc.
-Then I used the CTE in the main query by referencing it's assigned name.

Step 2: Compare the performance of your CTEs and subqueries.
1.Which approach do you think will perform better and why?

I assume the CTE is faster and will perform better. The CTE makes queries easier to follow and understand.

2.Compare the costs of all the queries by creating query plans for each one.

3.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.

	Query	**Cost**	**Runtime**
1st query – extract the average amount paid by top 5 customers	Subquery 3.8	Limit (cost=64.37..64.39 rows=5 width=67)"	Total query runtime: 112 msec
	CTE 3.9	Limit (cost=64.37..64.39 rows=5 width=270)"	Total query runtime: 94 msec
2nd query – Top 5 customers based in each country	Subquery 3.8	Limit (cost=150.46..150.47 rows=5 width=84)"	Total query runtime: 130 msec
	CTE 3.9	Limit (cost=166.84..166.86 rows=5 width=25)	Total query runtime: 95 msec

I used the explain syntax to extract the cost and the runtime.

The cost of the step 1 query for both subquery and CTE are the same and for the step 2 query the CTE is slightly higher (cost = 166.84 vs 150.46).

The runtimes are slightly higher for the subqueries compared to CTEs. Step 1 & 2 CTEs, runtime for CTEs is 94 msec and 95 msec. Whilst the step 1 & 2, runtime for subqueries is 112 msec and 130msec respectively.

4.Did the results surprise you? Write a few sentences to explain your answer.

The results are surprising because CTEs are easier to understand and follow, so I thought the cost of CTEs will be less than subquery.

4.Did the results surprise you? Write a few sentences to explain your answer.

It was quiet challenging putting together the CTEs query in step 2 because it had multiple CTEs and combining the CTEs was a bit hard.