June 6, 2022

Exercise 3.9

In this task, you'll convert your subqueries from task 3.8 into CTEs to make your code easier to read.

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

```
WITH cte_average (customer_id, first_name, last_name, country, city, amount) AS
(SELECT A.customer_id,
A.first_name,
A.last_name,
D.country,
C.city,
SUM(E.amount) AS "total_amount_paid"
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
INNER JOIN payment E ON A.customer_id = E.customer_id
WHERE C.city IN ('Aurora','Acua','Citrus Heights','Iwaki','Ambattur','Shanwei','So
Leopoldo','Teboksary','Tianjin','Cianjur')
GROUP BY A.customer_id, D.country, C.city
ORDER BY "total_amount_paid" DESC
LIMIT 5)
SELECT AVG(amount)
FROM cte_average
```

```
105.5540000000000000
```

```
WITH cte_top_5_customers (customer_id, first_name, last_name, country, city) AS
(SELECT A.customer_id,
A.first_name,
A.last_name,
D.country,
C.city,
SUM(E.amount) AS "total_amount_paid"
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
INNER JOIN payment E ON A.customer_id = E.customer_id
WHERE C.city IN ('Aurora','Acua','Citrus Heights','Iwaki','Ambattur','Shanwei','So Leopoldo','Teboksary','Tianjin','Cianjur')
GROUP BY A.customer_id, D.country, C.city
ORDER BY "total_amount_paid" DESC
LIMIT 5)
SELECT DISTINCT(D.country),
COUNT(A.customer_id) AS "all_customer_count",
COUNT(DISTINCT D.country) 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_customers ON D.country = cte_top_5_customers.country
GROUP BY D.country, cte_top_5_customers.country
ORDER BY all_customer_count DESC
LIMIT 5
```

Rank	**Country**	**All_customer_count**	**Cte_top_5_customers**
1.	"India"	60	1
2.	"China"	53	1
3.	"United States"	36	1
4.	"Japan"	31	1
5.	"Mexico"	30	1

Steps performed.

First, I copied the subquery over as the CTE. Then I added the beginning, WITH…-…AS. Finally I added the last section of the query, SELECT AVG…-…cte_average.

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

I think that in this case, the two will perform quite similarly. When looking at the two queries (CTE vs. Subquery) side-by-side, they are not all that different. Maybe with more complex queries, it will change the results.

Ex. 3.8 Subquery – 1

```
"Aggregate  (cost=64.49..64.50 rows=1 width=32)"
"  ->  Limit  (cost=64.41..64.43 rows=5 width=270)"
"      ->  Sort  (cost=64.41..65.02 rows=244 width=270)"
"          Sort Key: (sum(e.amount)) DESC"
"          ->  HashAggregate  (cost=57.31..60.36 rows=244 width=270)"
"              Group Key: a.customer_id, d.country, c.city"
"              ->  Nested Loop  (cost=18.16..54.87 rows=244 width=28)"
"                  ->  Hash Join  (cost=17.88..37.14 rows=10 width=22)"
"                      Hash Cond: (c.country_id = d.country_id)"
"                      ->  Nested Loop  (cost=14.43..33.66 rows=10 width=15)"
"                          ->  Hash Join  (cost=14.15..29.77 rows=10 width=15)"
"                              Hash Cond: (b.city_id = c.city_id)"
"                              ->  Seq Scan on address b  (cost=0.00..14.03 rows=603 width=6)"
"                              ->  Hash  (cost=14.03..14.03 rows=10 width=15)"
"                                  ->  Seq Scan on city c  (cost=0.03..14.03 rows=10 width=15)"
"                                      Filter: ((city)::text = ANY ('{Aurora,Acua,""Citrus
Heights"",Iwaki,Ambattur,Shanwei,""So Leopoldo"",Teboksary,Tianjin,Cianjur}'::text[]))"
"                          ->  Index Scan using idx_fk_address_id on customer a  (cost=0.28..0.38 rows=1
width=6)"
"                              Index Cond: (address_id = b.address_id)"
"                      ->  Hash  (cost=2.09..2.09 rows=109 width=13)"
"                          ->  Seq Scan on country d  (cost=0.00..2.09 rows=109 width=13)"
"                  ->  Index Scan using idx_fk_customer_id on payment e  (cost=0.29..1.53 rows=24
width=8)"
"                      Index Cond: (customer_id = a.customer_id)"
```

Ex. 3.9 CTE – 1

```
"Aggregate  (cost=64.49..64.50 rows=1 width=32)"
"  ->  Limit  (cost=64.41..64.43 rows=5 width=270)"
"      ->  Sort  (cost=64.41..65.02 rows=244 width=270)"
"          Sort Key: (sum(e.amount)) DESC"
"          ->  HashAggregate  (cost=57.31..60.36 rows=244 width=270)"
"              Group Key: a.customer_id, d.country, c.city"
"              ->  Nested Loop  (cost=18.16..54.87 rows=244 width=28)"
"                  ->  Hash Join  (cost=17.88..37.14 rows=10 width=22)"
"                      Hash Cond: (c.country_id = d.country_id)"
"                      ->  Nested Loop  (cost=14.43..33.66 rows=10 width=15)"
"                          ->  Hash Join  (cost=14.15..29.77 rows=10 width=15)"
"                              Hash Cond: (b.city_id = c.city_id)"
"                              ->  Seq Scan on address b  (cost=0.00..14.03 rows=603 width=6)"
"                              ->  Hash  (cost=14.03..14.03 rows=10 width=15)"
"                                  ->  Seq Scan on city c  (cost=0.03..14.03 rows=10 width=15)"
"                                      Filter: ((city)::text = ANY ('{Aurora,Acua,""Citrus
Heights"",Iwaki,Ambattur,Shanwei,""So Leopoldo"",Teboksary,Tianjin,Cianjur}'::text[]))"
```

```
"                                      ->  Index Scan using idx_fk_address_id on customer a  (cost=0.28..0.38 rows=1 width=6)"
"                                            Index Cond: (address_id = b.address_id)"
"                                ->  Hash  (cost=2.09..2.09 rows=109 width=13)"
"                                      ->  Seq Scan on country d  (cost=0.00..2.09 rows=109 width=13)"
"                          ->  Index Scan using idx_fk_customer_id on payment e  (cost=0.29..1.53 rows=24 width=8)"
"                                Index Cond: (customer_id = a.customer_id)"
```

Ex. 3.8 Subquery – 2

```
"Limit  (cost=189.52..189.53 rows=5 width=34)"
"  ->  Sort  (cost=189.52..190.88 rows=545 width=34)"
"        Sort Key: (count(a.customer_id)) DESC"
"        ->  HashAggregate  (cost=175.02..180.47 rows=545 width=34)"
"              Group Key: count(a.customer_id), d.country, count(DISTINCT d.country)"
"              ->  GroupAggregate  (cost=157.99..170.93 rows=545 width=34)"
"                    Group Key: d.country, top_5_customers.country"
"                    ->  Sort  (cost=157.99..159.49 rows=599 width=22)"
"                          Sort Key: d.country, top_5_customers.country"
"                          ->  Hash Left Join  (cost=108.06..130.36 rows=599 width=22)"
"                                Hash Cond: ((d.country)::text = (top_5_customers.country)::text)"
"                                ->  Hash Join  (cost=43.52..63.30 rows=599 width=13)"
"                                      Hash Cond: (c.country_id = d.country_id)"
"                                      ->  Hash Join  (cost=40.07..58.22 rows=599 width=6)"
"                                            Hash Cond: (b.city_id = c.city_id)"
"                                            ->  Hash Join  (cost=21.57..38.14 rows=599 width=6)"
"                                                  Hash Cond: (a.address_id = b.address_id)"
"                                                  ->  Seq Scan on customer a  (cost=0.00..14.99 rows=599 width=6)"
"                                                  ->  Hash  (cost=14.03..14.03 rows=603 width=6)"
"                                                        ->  Seq Scan on address b  (cost=0.00..14.03 rows=603 width=6)"
"                                            ->  Hash  (cost=11.00..11.00 rows=600 width=6)"
"                                                  ->  Seq Scan on city c  (cost=0.00..11.00 rows=600 width=6)"
"                                      ->  Hash  (cost=2.09..2.09 rows=109 width=13)"
"                                            ->  Seq Scan on country d  (cost=0.00..2.09 rows=109 width=13)"
"                                ->  Hash  (cost=64.48..64.48 rows=5 width=9)"
"                                      ->  Subquery Scan on top_5_customers  (cost=64.41..64.48 rows=5 width=9)"
"                                            ->  Limit  (cost=64.41..64.43 rows=5 width=270)"
"                                                  ->  Sort  (cost=64.41..65.02 rows=244 width=270)"
"                                                        Sort Key: (sum(e.amount)) DESC"
"                                                        ->  HashAggregate  (cost=57.31..60.36 rows=244 width=270)"
"                                                              Group Key: a_1.customer_id, d_1.country, c_1.city"
"                                                              ->  Nested Loop  (cost=18.16..54.87 rows=244 width=28)"
"                                                                    ->  Hash Join  (cost=17.88..37.14 rows=10 width=22)"
"                                                                          Hash Cond: (c_1.country_id = d_1.country_id)"
"                                                                          ->  Nested Loop  (cost=14.43..33.66 rows=10 width=15)"
"                                                                                ->  Hash Join  (cost=14.15..29.77 rows=10 width=15)"
"                                                                                      Hash Cond: (b_1.city_id = c_1.city_id)"
"                                                                                      ->  Seq Scan on address b_1  (cost=0.00..14.03
rows=603 width=6)"
"                                                                                      ->  Hash  (cost=14.03..14.03 rows=10 width=15)"
"                                                                                            ->  Seq Scan on city c_1  (cost=0.03..14.03 rows=10
width=15)"
"                                                                                                  Filter: ((city)::text = ANY ('{Aurora,Acua,""Citrus
Heights"",Iwaki,Ambattur,Shanwei,""So Leopoldo"",Teboksary,Tianjin,Cianjur}'::text[]))"
"                                                                                ->  Index Scan using idx_fk_address_id on customer a_1
(cost=0.28..0.38 rows=1 width=6)"
"                                                                                      Index Cond: (address_id = b_1.address_id)"
"                                                                          ->  Hash  (cost=2.09..2.09 rows=109 width=13)"
"                                                                                ->  Seq Scan on country d_1  (cost=0.00..2.09 rows=109
width=13)"
"                                                                    ->  Index Scan using idx_fk_customer_id on payment e
(cost=0.29..1.53 rows=24 width=8)"
"                                                                          Index Cond: (customer_id = a_1.customer_id)"
```

Ex. 3.9 CTE – 2

```
"Limit  (cost=189.52..189.53 rows=5 width=34)"
"  ->  Sort  (cost=189.52..190.88 rows=545 width=34)"
"        Sort Key: (count(a.customer_id)) DESC"
"        ->  HashAggregate  (cost=175.02..180.47 rows=545 width=34)"
"              Group Key: count(a.customer_id), d.country, count(DISTINCT d.country)"
"              ->  GroupAggregate  (cost=157.99..170.93 rows=545 width=34)"
"                    Group Key: d.country, cte_top_5_customers.country"
"                    ->  Sort  (cost=157.99..159.49 rows=599 width=22)"
"                          Sort Key: d.country, cte_top_5_customers.country"
"                          ->  Hash Left Join  (cost=108.06..130.36 rows=599 width=22)"
"                                Hash Cond: ((d.country)::text = (cte_top_5_customers.country)::text)"
"                                ->  Hash Join  (cost=43.52..63.30 rows=599 width=13)"
"                                      Hash Cond: (c.country_id = d.country_id)"
"                                      ->  Hash Join  (cost=40.07..58.22 rows=599 width=6)"
"                                            Hash Cond: (b.city_id = c.city_id)"
"                                            ->  Hash Join  (cost=21.57..38.14 rows=599 width=6)"
"                                                  Hash Cond: (a.address_id = b.address_id)"
"                                                  ->  Seq Scan on customer a  (cost=0.00..14.99 rows=599 width=6)"
"                                    ->  Hash  (cost=14.03..14.03 rows=603 width=6)"
"                                        ->  Seq Scan on address b  (cost=0.00..14.03 rows=603 width=6)"
"                                ->  Hash  (cost=11.00..11.00 rows=600 width=6)"
"                                    ->  Seq Scan on city c  (cost=0.00..11.00 rows=600 width=6)"
"                            ->  Hash  (cost=2.09..2.09 rows=109 width=13)"
"                                ->  Seq Scan on country d  (cost=0.00..2.09 rows=109 width=13)"
"                        ->  Hash  (cost=64.48..64.48 rows=5 width=9)"
"                            ->  Subquery Scan on cte_top_5_customers  (cost=64.41..64.48 rows=5 width=9)"
"                                ->  Limit  (cost=64.41..64.43 rows=5 width=270)"
"                                    ->  Sort  (cost=64.41..65.02 rows=244 width=270)"
"                                        Sort Key: (sum(e.amount)) DESC"
"                                        ->  HashAggregate  (cost=57.31..60.36 rows=244 width=270)"
"                                            Group Key: a_1.customer_id, d_1.country, c_1.city"
"                                            ->  Nested Loop  (cost=18.16..54.87 rows=244 width=28)"
"                                                ->  Hash Join  (cost=17.88..37.14 rows=10 width=22)"
"                                                    Hash Cond: (c_1.country_id = d_1.country_id)"
"                                                    ->  Nested Loop  (cost=14.43..33.66 rows=10 width=15)"
"                                                        ->  Hash Join  (cost=14.15..29.77 rows=10 width=15)"
"                                                            Hash Cond: (b_1.city_id = c_1.city_id)"
"                                                            ->  Seq Scan on address b_1  (cost=0.00..14.03
rows=603 width=6)"
"                                                            ->  Hash  (cost=14.03..14.03 rows=10 width=15)"
"                                                                ->  Seq Scan on city c_1  (cost=0.03..14.03 rows=10
width=15)"
"                                                                    Filter: ((city)::text = ANY ('{Aurora,Acua,""Citrus
Heights"",Iwaki,Ambattur,Shanwei,""So Leopoldo"",Teboksary,Tianjin,Cianjur}'::text[]))"
"                                                        ->  Index Scan using idx_fk_address_id on customer a_1
(cost=0.28..0.38 rows=1 width=6)"
"                                                            Index Cond: (address_id = b_1.address_id)"
"                                                    ->  Hash  (cost=2.09..2.09 rows=109 width=13)"
"                                                        ->  Seq Scan on country d_1  (cost=0.00..2.09 rows=109
width=13)"
"                                                ->  Index Scan using idx_fk_customer_id on payment e
(cost=0.29..1.53 rows=24 width=8)"
"                                                    Index Cond: (customer_id = a_1.customer_id)"
```

QUERY	COST
Exercise 3.8 – 1	Aggregate (cost=64.49..64.50 rows=1 width=32)
Exercise 3.9 – 1	Aggregate (cost=64.49..64.50 rows=1 width=32)
Exercise 3.8 – 2	Limit (cost=189.52..189.53 rows=5 width=34)
Exercise 3.9 – 2	Limit (cost=189.52..189.53 rows=5 width=34)

STEP 3 - Write 1 to 2 paragraphs on the challenges you faced when replacing your subqueries with CTEs.

There were a few challenges I faced when replacing subqueries with CTEs. I understood moving the subquery to the beginning, where I was confused was having to join the CTE and remembering to change the GROUP BY. Overall, the transformation wasn’t too complicated, after a few more examples, I should have the hang of it.