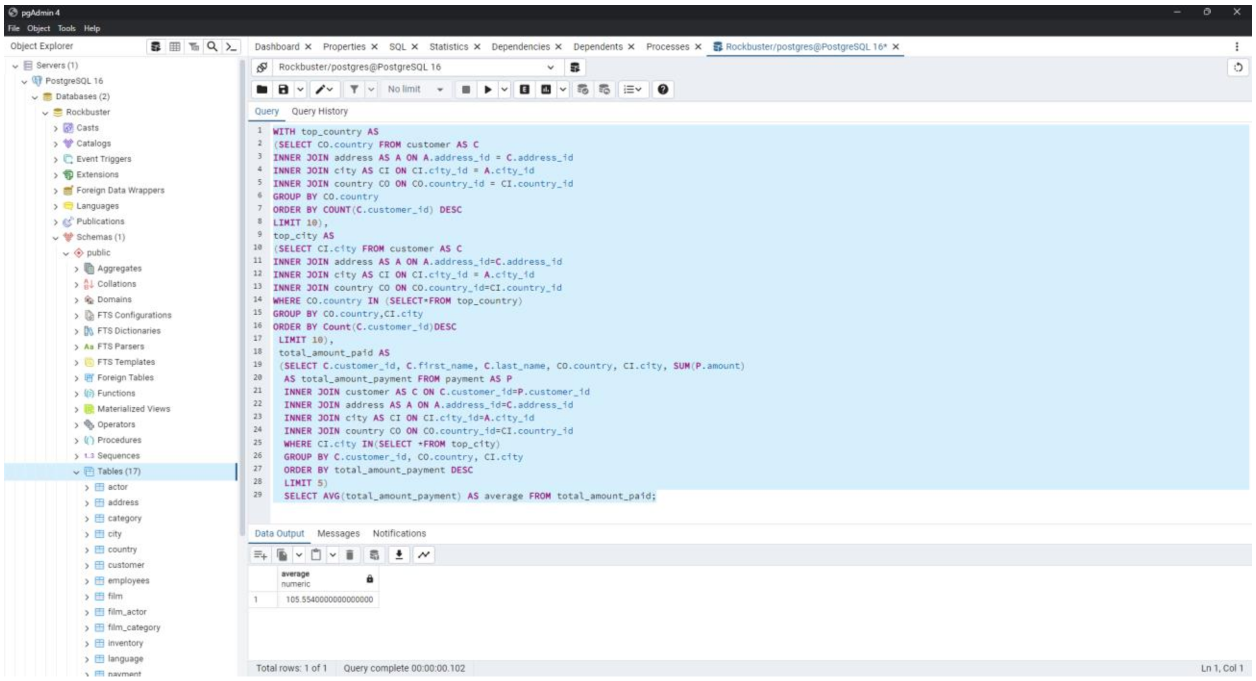
Mandy Yeung
Data Immersion Exercise 3.9

Common Table Expressions

Step 1:

Query1

```
WITH top_country AS
(SELECT CO.country FROM customer AS C
INNER JOIN address AS A ON A.address_id = C.address_id
INNER JOIN city AS CI ON CI.city_id = A.city_id
INNER JOIN country CO ON CO.country_id = CI.country_id
GROUP BY CO.country
ORDER BY COUNT(C.customer_id) DESC
LIMIT 10),
top_city AS
(SELECT CI.city FROM customer AS C
INNER JOIN address AS A ON A.address_id=C.address_id
INNER JOIN city AS CI ON CI.city_id = A.city_id
INNER JOIN country CO ON CO.country_id=CI.country_id
WHERE CO.country IN (SELECT*FROM top_country)
GROUP BY CO.country,CI.city
ORDER BY Count(C.customer_id)DESC
 LIMIT 10),
 total_amount_paid AS
 (SELECT C.customer_id, C.first_name, C.last_name, CO.country, CI.city, SUM(P.amount)
  AS total_amount_payment FROM payment AS P
  INNER JOIN customer AS C ON C.customer_id=P.customer_id
  INNER JOIN address AS A ON A.address_id=C.address_id
  INNER JOIN city AS CI ON CI.city_id=A.city_id
  INNER JOIN country CO ON CO.country_id=CI.country_id
  WHERE CI.city IN(SELECT *FROM top_city)
  GROUP BY C.customer_id, CO.country, CI.city
  ORDER BY total_amount_payment DESC
  LIMIT 5)
  SELECT AVG(total_amount_payment) AS average FROM total_amount_paid;
```



```
WITH top_country AS
(SELECT CO.country FROM customer AS C
INNER JOIN address AS A ON A.address_id = C.address_id
INNER JOIN city AS CI ON CI.city_id = A.city_id
INNER JOIN country CO ON CO.country_id = CI.country_id
GROUP BY CO.country
ORDER BY COUNT(C.customer_id) DESC
LIMIT 10),
top_city AS
(SELECT CI.city FROM customer AS C
INNER JOIN address AS A ON A.address_id=C.address_id
INNER JOIN city AS CI ON CI.city_id = A.city_id
INNER JOIN country CO ON CO.country_id=CI.country_id
WHERE CO.country IN (SELECT*FROM top_country)
GROUP BY CO.country,CI.city
ORDER BY Count(C.customer_id)DESC
 LIMIT 10),
 total_amount_paid AS
 (SELECT C.customer_id, C.first_name, C.last_name, CO.country, CI.city, SUM(P.amount)
  AS total_amount_payment FROM payment AS P
  INNER JOIN customer AS C ON C.customer_id=P.customer_id
  INNER JOIN address AS A ON A.address_id=C.address_id
  INNER JOIN city AS CI ON CI.city_id=A.city_id
  INNER JOIN country CO ON CO.country_id=CI.country_id
  WHERE CI.city IN(SELECT *FROM top_city)
  GROUP BY C.customer_id, CO.country, CI.city
  ORDER BY total_amount_payment DESC
  LIMIT 5)
  SELECT AVG(total_amount_payment) AS average FROM total_amount_paid;
```

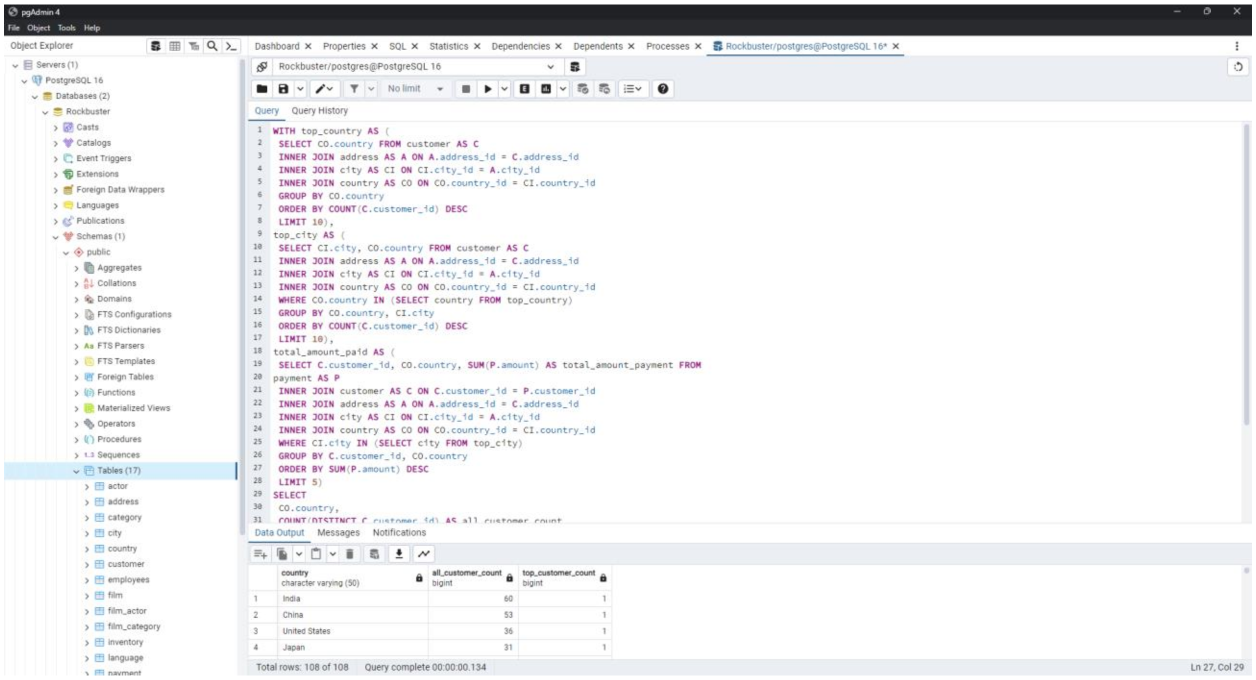
	average numeric
1	105.5540000000000000

Total rows: 1 of 1   Query complete 00:00:00.102

Query 2:

WITH top_country AS (
SELECT CO.country FROM customer AS C
INNER JOIN address AS A ON A.address_id = C.address_id
INNER JOIN city AS CI ON CI.city_id = A.city_id
INNER JOIN country AS CO ON CO.country_id = CI.country_id
GROUP BY CO.country
ORDER BY COUNT(C.customer_id) DESC
LIMIT 10),
top_city AS (
SELECT CI.city, CO.country FROM customer AS C
INNER JOIN address AS A ON A.address_id = C.address_id
INNER JOIN city AS CI ON CI.city_id = A.city_id
INNER JOIN country AS CO ON CO.country_id = CI.country_id
WHERE CO.country IN (SELECT country FROM top_country)
GROUP BY CO.country, CI.city
ORDER BY COUNT(C.customer_id) DESC
LIMIT 10),
total_amount_paid AS (
SELECT C.customer_id, CO.country, SUM(P.amount) AS total_amount_payment FROM payment AS P
INNER JOIN customer AS C ON C.customer_id = P.customer_id
INNER JOIN address AS A ON A.address_id = C.address_id
INNER JOIN city AS CI ON CI.city_id = A.city_id
INNER JOIN country AS CO ON CO.country_id = CI.country_id
WHERE CI.city IN (SELECT city FROM top_city)
GROUP BY C.customer_id, CO.country

```
 ORDER BY SUM(P.amount) DESC
 LIMIT 5)
SELECT
 CO.country,
 COUNT(DISTINCT C.customer_id) AS all_customer_count,
 COUNT(DISTINCT T5.customer_id) AS top_customer_count
FROM customer AS C
INNER JOIN address AS A ON A.address_id = C.address_id
INNER JOIN city AS CI ON CI.city_id = A.city_id
INNER JOIN country AS CO ON CO.country_id = CI.country_id
LEFT JOIN total_amount_paid AS T5 ON T5.customer_id = C.customer_id AND T5.country =
CO.country
GROUP BY CO.country
ORDER BY all_customer_count DESC;
```



```
WITH top_country AS (
 SELECT CO.country FROM customer AS C
 INNER JOIN address AS A ON A.address_id = C.address_id
 INNER JOIN city AS CI ON CI.city_id = A.city_id
 INNER JOIN country AS CO ON CO.country_id = CI.country_id
 GROUP BY CO.country
 ORDER BY COUNT(C.customer_id) DESC
 LIMIT 10),
top_city AS (
 SELECT CI.city, CO.country FROM customer AS C
 INNER JOIN address AS A ON A.address_id = C.address_id
 INNER JOIN city AS CI ON CI.city_id = A.city_id
 INNER JOIN country AS CO ON CO.country_id = CI.country_id
 WHERE CO.country IN (SELECT country FROM top_country)
 GROUP BY CO.country, CI.city
 ORDER BY COUNT(C.customer_id) DESC
 LIMIT 10),
total_amount_paid AS (
 SELECT C.customer_id, CO.country, SUM(P.amount) AS total_amount_payment FROM
payment AS P
 INNER JOIN customer AS C ON C.customer_id = P.customer_id
 INNER JOIN address AS A ON A.address_id = C.address_id
 INNER JOIN city AS CI ON CI.city_id = A.city_id
 INNER JOIN country AS CO ON CO.country_id = CI.country_id
 WHERE CI.city IN (SELECT city FROM top_city)
 GROUP BY C.customer_id, CO.country
 ORDER BY SUM(P.amount) DESC
 LIMIT 5)
SELECT
 CO.country,
 COUNT(DISTINCT C.customer_id) AS all_customer_count
```

	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

Total rows: 108 of 108   Query complete 00:00:00.134   Ln 27, Col 29

First, I copied the query from my query history into the query tool. Then I rearrange the query into it's correct format for CTE. Starting from the innermost layer, rewrite each subqueries into CTEs.

<u>Step 2</u>

1. I believe CTE will perform better than subquery because of it's readability and the structure of the queries, making it easier to understand and maintain.
2. Costs:

Query 1 task 3.8	EXPLAIN: Aggregate (cost=166.06..166.07 rows=1 width =32
	Actual Execution: 85 msec
Query 1 task 3.9	EXPLAIN: Aggregate (cost=256.43..266.55 rows=109 width=25)
	Actual Execution: 125 msec

Query 2 – task 3.8	EXPLAIN: Aggregate (270.33..270.60 rows=109 width=25) Actual Execution: 89 msec
Query 2 – task 3.9	EXPLAIN: Aggregate (cost=356.43..266.55 rows=109 width=25) Actual Execution: 113 msec

I am somewhat surprised as I would have thought that CTEs would be less costly and faster but it's the contrary.

Step 3

Changing the subqueries into CTEs is a bit challenging for me as you would have to rearrange the statements around to the correct format.