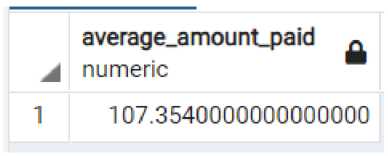
3.9: Common Table Expressions
Prepared By: Rob Rowland

Step 1:

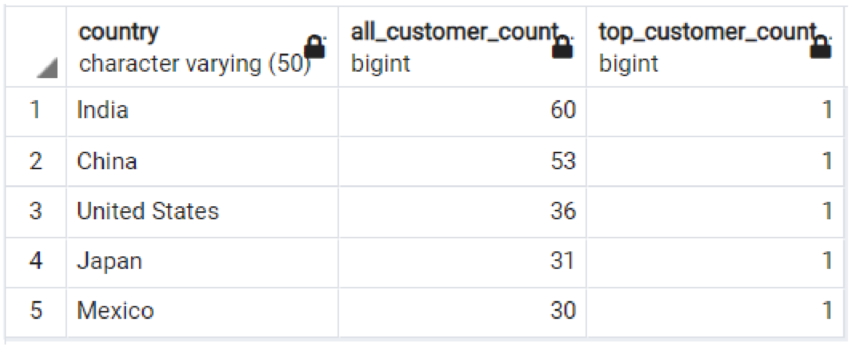
```
WITH avg_amount_paid_cte (amount) AS
(SELECT AVG(highest_amount_paid)
FROM
(SELECT A.customer_id, B.first_name, B.last_name, D.city, E.country,
SUM(amount) AS highest_amount_paid
FROM payment A
INNER JOIN customer B ON A.customer_id = B.customer_id
INNER JOIN address C ON B.address_id = C.address_id
INNER JOIN city D ON C.city_id = D.city_id
INNER JOIN country E ON D.country_id = E.country_id
WHERE country IN ('India',
                  'China',
                  'United States',
                  'Japan',
                  'Mexico',
                  'Brazil',
                  'Russian Federation',
                  'Philippines',
                  'Turkey',
                  'Indonesia')
AND city IN ('Aurora',
             'Atlixco',
              'Xintai',
              'Adoni',
              'Dhule (Dhulia)',
              'Kurashiki',
              'Pingxiang',
              'Sivas',
              'Celaya',
              'So Leopoldo')
GROUP BY A.customer_id, B.first_name, B.last_name, D.city, E.country
ORDER BY highest_amount_paid DESC
LIMIT 5) AS average)
SELECT AVG(amount) AS average_amount_paid
FROM avg_amount_paid_cte;
```



	average_amount_paid numeric
1	107.3540000000000000

First, I copied and pasted the original query from exercise 3.8. Then, I added the WITH statement, determined the name of my CTE (avg_amount_paid_cte), included amount as my only column, and added the AS statement to transition to the original query. I then added () around the original query. Lastly, I wrote the main statement at the end, using SELECT and AVG to select the average amount paid per customer from the CTE that I just created.

```
WITH top_countries_cte (country, all_customer_count, top_customer_count) AS
(SELECT country.country,
COUNT(DISTINCT customer.customer_id) AS all_customer_count,
COUNT(DISTINCT country.country) AS top_customer_count
FROM
(SELECT A.customer_id, B.first_name, B.last_name, D.city, E.country,
SUM(amount) AS highest_amount_paid
FROM payment A
INNER JOIN customer B ON A.customer_id = B.customer_id
INNER JOIN address C ON B.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 country IN ('India',
                                    'China',
                                    'United States',
                                    'Japan',
                                    'Mexico',
                                    'Brazil',
                                    'Russian Federation',
                                    'Philippines',
                                    'Turkey',
                                    'Indonesia')
AND city IN ('Aurora',
                              'Atlixco',
                               'Xintai',
                               'Adoni',
                               'Dhule (Dhulia)',
                               'Kurashiki',
                               'Pingxiang',
                               'Sivas',
                               'Celaya',
                               'So Leopoldo')
GROUP BY A.customer_id, B.first_name, B.last_name, D.city, E.country
ORDER BY highest_amount_paid DESC
LIMIT 5) AS top_5_customers
LEFT JOIN customer ON customer.customer_id = customer.customer_id
LEFT JOIN address ON customer.address_id = address.address_id
LEFT JOIN city ON address.city_id = city.city_id
LEFT JOIN country ON city.country_id = country.country_id
GROUP BY country.country
ORDER BY COUNT (country.country) DESC
LIMIT 5)
SELECT D.country,
COUNT(DISTINCT A.customer_id) AS all_customer_count,
COUNT(DISTINCT top_countries_cte) 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 top_countries_cte ON D.country =
top_countries_cte.country
GROUP BY D.country
ORDER BY top_customer_count DESC, all_customer_count DESC
LIMIT 5;
```



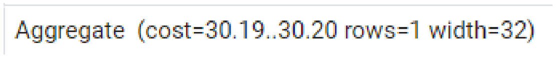
	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	India	60	1
2	China	53	1
3	United States	36	1
4	Japan	31	1
5	Mexico	30	1

I used that same process as the first CTE above. However, the main statement was a little more intensive. In fact, I feel like I may have made it hard than I needed to. For the main statement, I selected the three columns I wanted, two of which required the COUNT statement. I then used INNER JOIN and LEFT JOIN statements in order to connect the customer table to the new CTE. I then grouped by country, and ordered by top_customer_count, then _all_customer_count, and limited the output to 5. Again, I got to the desired result but feel like I made it too complicated.

Step 2:

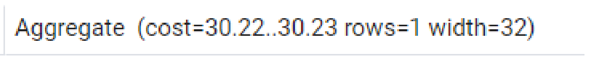
1. In these cases, I think the subquery approach will perform better because the SQL statements are shorter. It seems that using CTEs is most beneficial when writing longer queries, to decrease the number of lines by referencing the CTE frequently.

Avg Amount Paid Subquery:



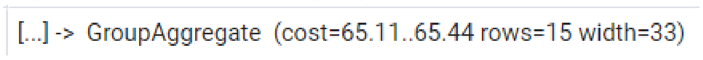
```
Aggregate  (cost=30.19..30.20 rows=1 width=32)
```

Avg Amount Paid CTE:



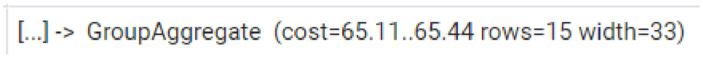
```
Aggregate  (cost=30.22..30.23 rows=1 width=32)
```

Top Customer Count Subquery:



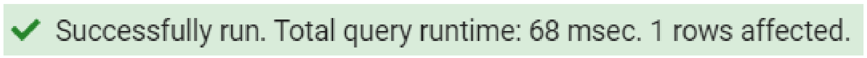
```
[...] -> GroupAggregate  (cost=65.11..65.44 rows=15 width=33)
```

Top Customer Count CTE:



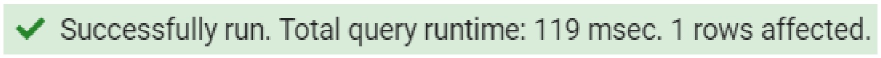
```
[...] -> GroupAggregate  (cost=65.11..65.44 rows=15 width=33)
```

Avg Amount Paid Subquery:



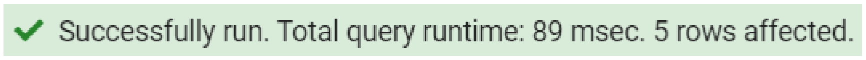
```
✓ Successfully run. Total query runtime: 68 msec. 1 rows affected.
```

Avg Amount Paid CTE:



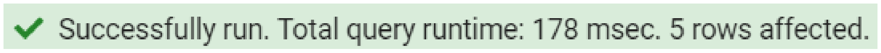
```
✓ Successfully run. Total query runtime: 119 msec. 1 rows affected.
```

Top Customer Count Subquery:



```
✓ Successfully run. Total query runtime: 89 msec. 5 rows affected.
```

Top Customer Count CTE:



```
✓ Successfully run. Total query runtime: 178 msec. 5 rows affected.
```

The results did not surprise me. It appears that I was correct in my thinking that, in this case, the subquery option was the most efficient.

Step 3:

I did not find the concept of CTEs difficult to understand. However, especially with the second CTE I wrote, I did feel like I was being repetitive when writing the main statement. I felt like I needed to connect the correct tables to the CTE within the main statement, but it felt inefficient. Perhaps there's a quicker way to do this. I was able to accomplish the desired result, but I am curious if I was doing it in the best way possible.