July 12, 2022

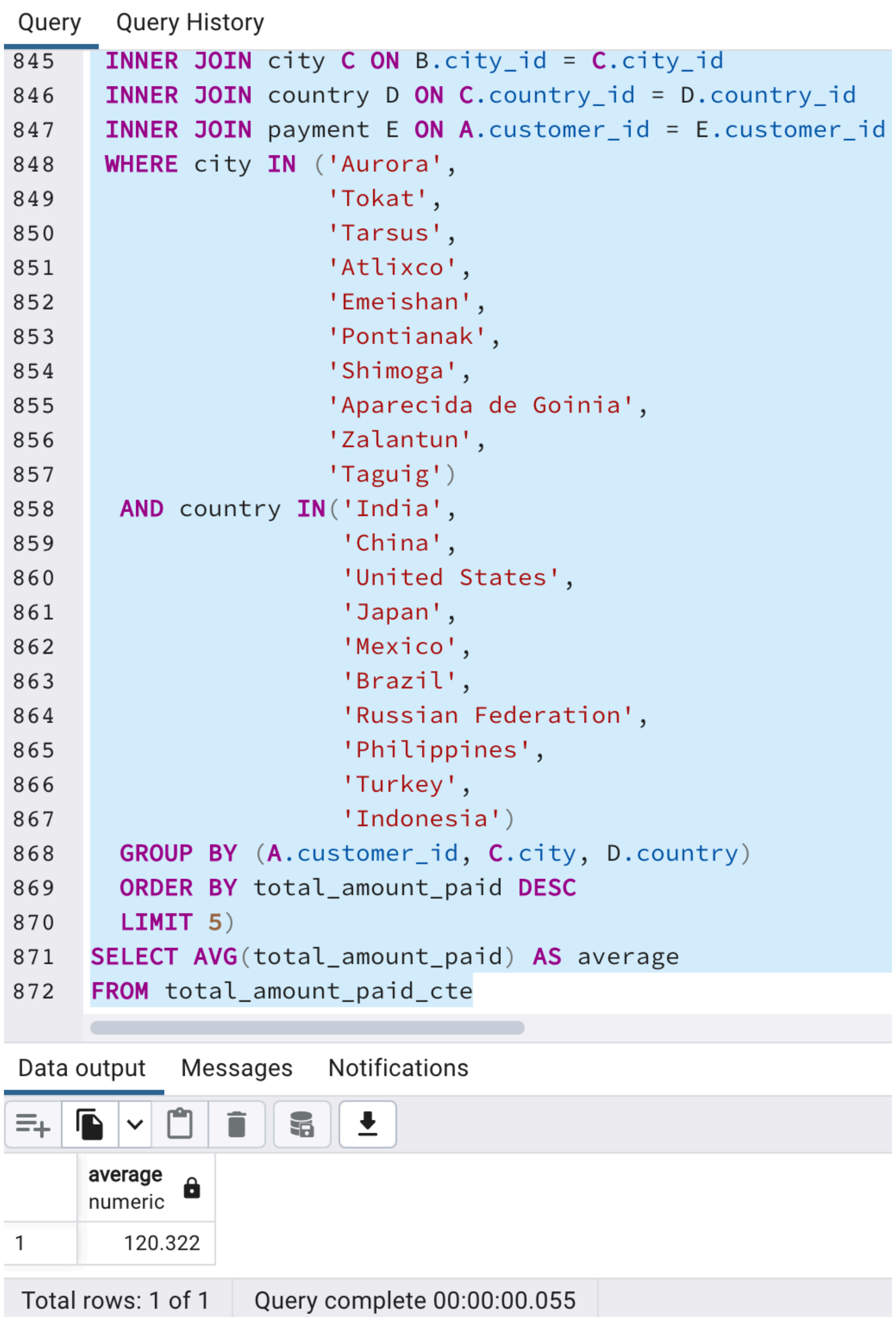
Data Immersion
Databases & SQL for Analysts
3.9 Common Table Expressions

1. **Rewrite the queries from steps 1 and 2 from Task 3.8 as CTE.**

CTE for average paid amount by top 5 customers:

First, I isolated the CTE by taking the subquery out of the main query and defined it by applying WITH at the beginning. Then, I changed the main statement/query by replacing the name of the column to extract, same as the name of the table from which will pull the data. The query ran with the CTE and provided same result

```
WITH total_amount_paid_cte(customer_id, first_name, last_name, city, country) AS
(SELECT A.customer_id,
        A.first_name,
        A.last_name,
        C.city,
        D.country,
       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 city IN ('Aurora',
                'Tokat',
                'Tarsus',
                'Atlixco',
                'Emeishan',
                'Pontianak',
                'Shimoga',
                'Aparecida de Goinia',
                'Zalantun',
                'Taguig')
  AND country IN('India',
                 'China',
                 'United States',
                 'Japan',
                 'Mexico',
                 'Brazil',
                 'Russian Federation',
                              'Philippines',
                              'Turkey',
                              'Indonesia')
 GROUP BY (A.customer_id, C.city, D.country)
 ORDER BY total_amount_paid DESC
 LIMIT 5)
SELECT AVG(total_amount_paid) AS average
FROM total_amount_paid_cte
```



```
 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 city IN ('Aurora',
               'Tokat',
               'Tarsus',
               'Atlixco',
               'Emeishan',
               'Pontianak',
               'Shimoga',
               'Aparecida de Goinia',
               'Zalantun',
               'Taguig')
  AND country IN('India',
                'China',
                'United States',
                'Japan',
                'Mexico',
                'Brazil',
                'Russian Federation',
                'Philippines',
                'Turkey',
                'Indonesia')
  GROUP BY (A.customer_id, C.city, D.country)
  ORDER BY total_amount_paid DESC
  LIMIT 5)
SELECT AVG(total_amount_paid) AS average
FROM total_amount_paid_cte
```

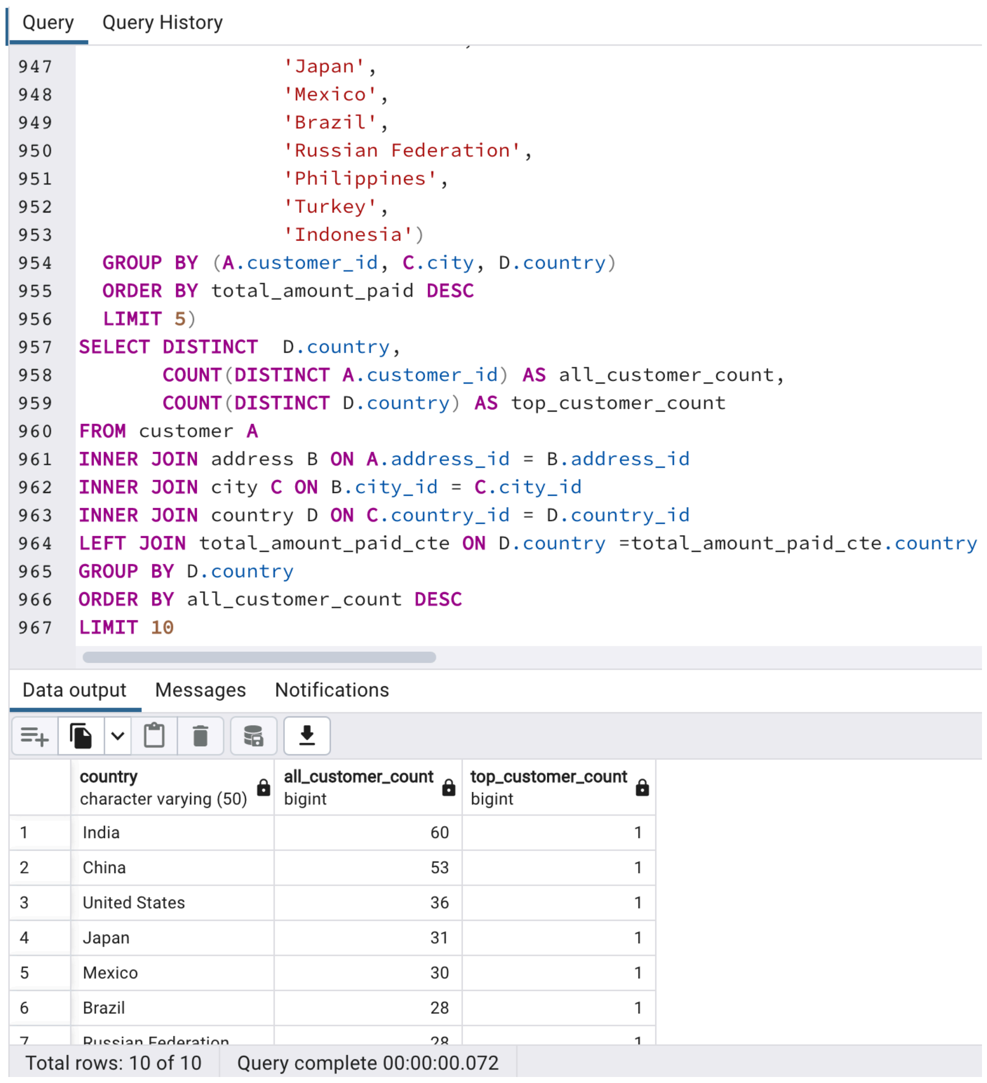
	average numeric
1	120.322

Total rows: 1 of 1   Query complete 00:00:00.055

CTE for top customers living within top countries:

First, I isolated the subquery and took it off the main statement, to then defining the CTE. After defining the CTE, I revised the main statement to see if it made sense; I selected the same columns and did the same joins, but in the last join (LEFT JOIN), I replaced the table name by the CTE name and assigned "country" as the key to connect CTE's table with the country table. The query with CTE ran well and gave the same results as if I was referencing a swubquery.

```
WITH total_amount_paid_cte(customer_id, first_name, last_name, city, country) AS
(SELECT A.customer_id,
            A.first_name,
            A.last_name,
            C.city,
            D.country,
            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 city IN ('Aurora',
                   'Tokat',
                   'Tarsus',
                   'Atlixco',
                   'Emeishan',
                   'Pontianak',
                   'Shimoga',
                   'Aparecida de Goinia',
                   'Zalantun',
                   'Taguig')
  AND country IN('India',
                    'China',
                    'United States',
                    'Japan',
                    'Mexico',
                    'Brazil',
                    'Russian Federation',
                    'Philippines',
                    'Turkey',
                    'Indonesia')
  GROUP BY (A.customer_id, C.city, D.country)
  ORDER BY total_amount_paid DESC
 LIMIT 5)
SELECT DISTINCT  D.country,
         COUNT(DISTINCT 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 total_amount_paid_cte ON D.country =total_amount_paid_cte.country
GROUP BY D.country
ORDER BY all_customer_count DESC
LIMIT 10
```



Query   Query History

```
                 'Japan',
                 'Mexico',
                 'Brazil',
                 'Russian Federation',
                 'Philippines',
                 'Turkey',
                 'Indonesia')
  GROUP BY (A.customer_id, C.city, D.country)
  ORDER BY total_amount_paid DESC
  LIMIT 5)
SELECT DISTINCT  D.country,
       COUNT(DISTINCT 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 total_amount_paid_cte ON D.country =total_amount_paid_cte.country
GROUP BY D.country
ORDER BY all_customer_count DESC
LIMIT 10
```

Data output   Messages   Notifications

	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	Brazil	28	1
7	Russian Federation	28	1

Total rows: 10 of 10   Query complete 00:00:00.072

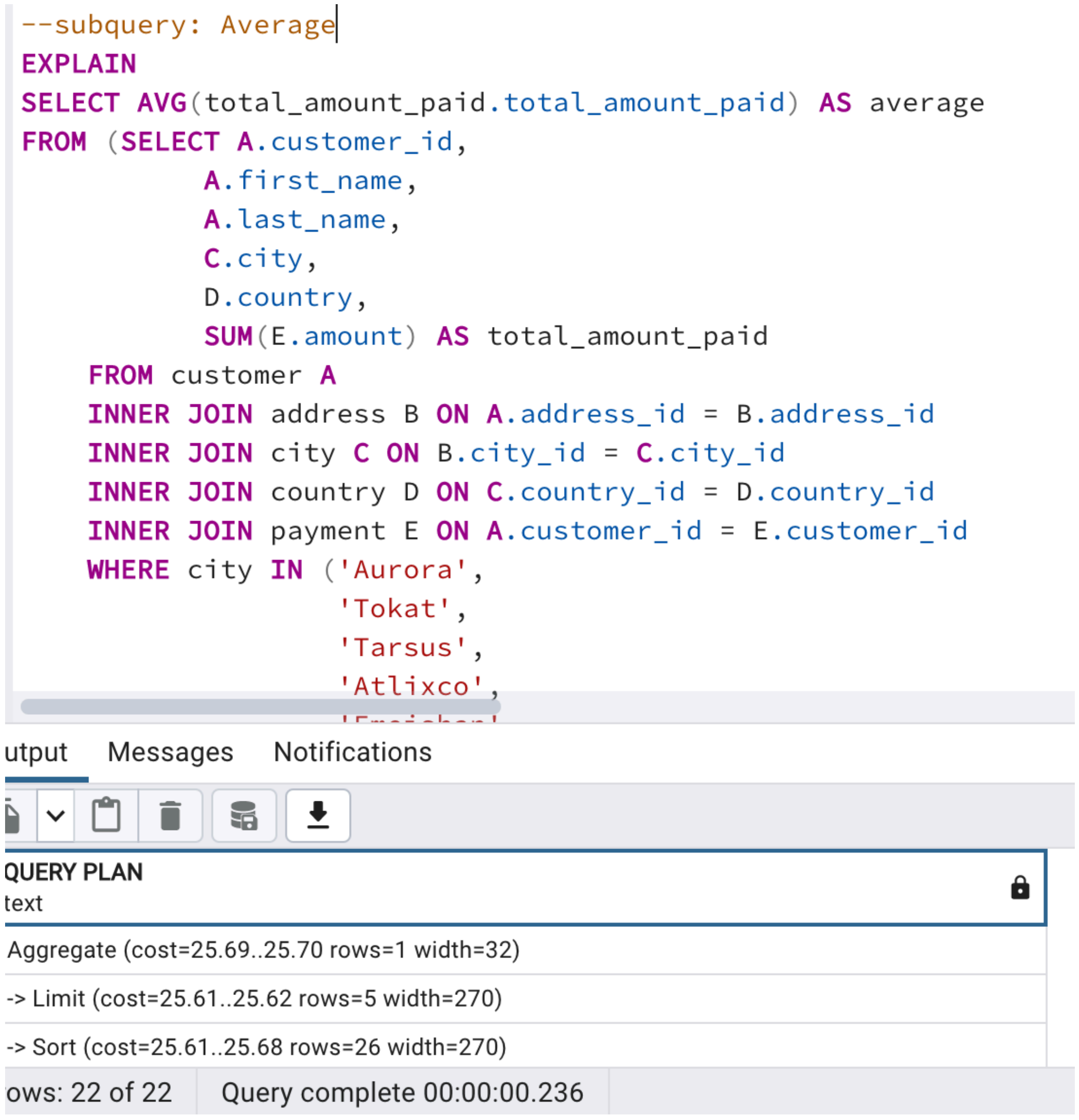
2. **Compare the performance of the CTEs against the subqueries**
    - **Which approach will you think will perform better and why?**

With little experience, I am keen to say that for many cases a CTE is better in many ways as it allows you to work even with user restrictions, and to define the CTE only once instead of referencing it as a subquery every time. It might not be less costly (or might be), but certainly it's more practical than using a subquery each time we need to reference to its results.

- **Compare the cost of each query with EXPLAIN, and create a plan for each one**

EXPLAIN Average subquery:

- Cost, 25.70
- Number of rows, 1
- Width of values in rows, 32

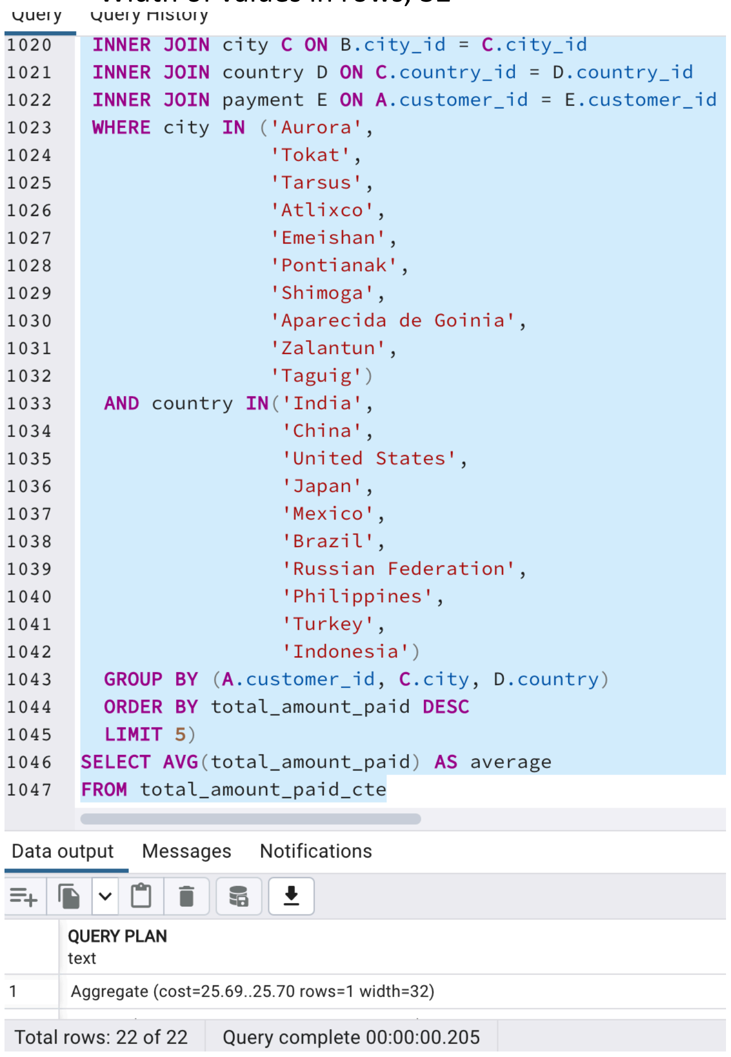


```
--subquery: Average
EXPLAIN
SELECT AVG(total_amount_paid.total_amount_paid) AS average
FROM (SELECT A.customer_id,
             A.first_name,
             A.last_name,
             C.city,
             D.country,
             SUM(E.amount) AS total_amount_paid
    FROM customer A
    INNER JOIN address 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 city IN ('Aurora',
                   'Tokat',
                   'Tarsus',
                   'Atlixco',
```

```
QUERY PLAN
text
Aggregate (cost=25.69..25.70 rows=1 width=32)
-> Limit (cost=25.61..25.62 rows=5 width=270)
-> Sort (cost=25.61..25.68 rows=26 width=270)
ows: 22 of 22    Query complete 00:00:00.236
```

EXPLAIN Average CTE:

- Cost, 25.70
- Number of rows, 1
- Width of values in rows, 32



```
 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 city IN ('Aurora',
               'Tokat',
               'Tarsus',
               'Atlixco',
               'Emeishan',
               'Pontianak',
               'Shimoga',
               'Aparecida de Goinia',
               'Zalantun',
               'Taguig')
  AND country IN('India',
                'China',
                'United States',
                'Japan',
                'Mexico',
                'Brazil',
                'Russian Federation',
                'Philippines',
                'Turkey',
                'Indonesia')
  GROUP BY (A.customer_id, C.city, D.country)
  ORDER BY total_amount_paid DESC
  LIMIT 5)
SELECT AVG(total_amount_paid) AS average
FROM total_amount_paid_cte
```

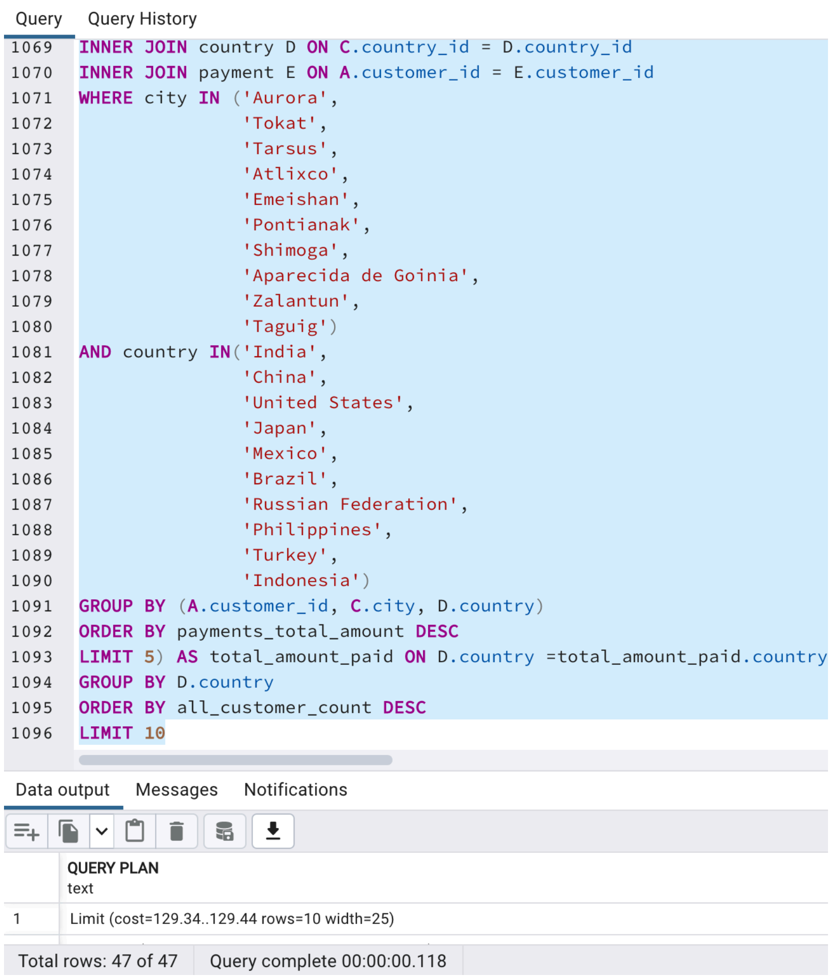
```
   QUERY PLAN
   text
1  Aggregate (cost=25.69..25.70 rows=1 width=32)

Total rows: 22 of 22    Query complete 00:00:00.205
```

In both cases, the cost is the same. We can use one or the other, we just need to consider if the subquery needs to be referenced more than once so we can use a CTE instead, but since this is a one-time request, we can go with whichever option.

EXPLAIN Subquery customers living in countries with biggest base of customers:

- Cost, 129.44
- Number of rows, 10
- Width of values in rows, 25


Query Query History

```
INNER JOIN country D ON C.country_id = D.country_id
INNER JOIN payment E ON A.customer_id = E.customer_id
WHERE city IN ('Aurora',
               'Tokat',
               'Tarsus',
               'Atlixco',
               'Emeishan',
               'Pontianak',
               'Shimoga',
               'Aparecida de Goinia',
               'Zalantun',
               'Taguig')
AND country IN('India',
               'China',
               'United States',
               'Japan',
               'Mexico',
               'Brazil',
               'Russian Federation',
               'Philippines',
               'Turkey',
               'Indonesia')
GROUP BY (A.customer_id, C.city, D.country)
ORDER BY payments_total_amount DESC
LIMIT 5) AS total_amount_paid ON D.country =total_amount_paid.country
GROUP BY D.country
ORDER BY all_customer_count DESC
LIMIT 10
```

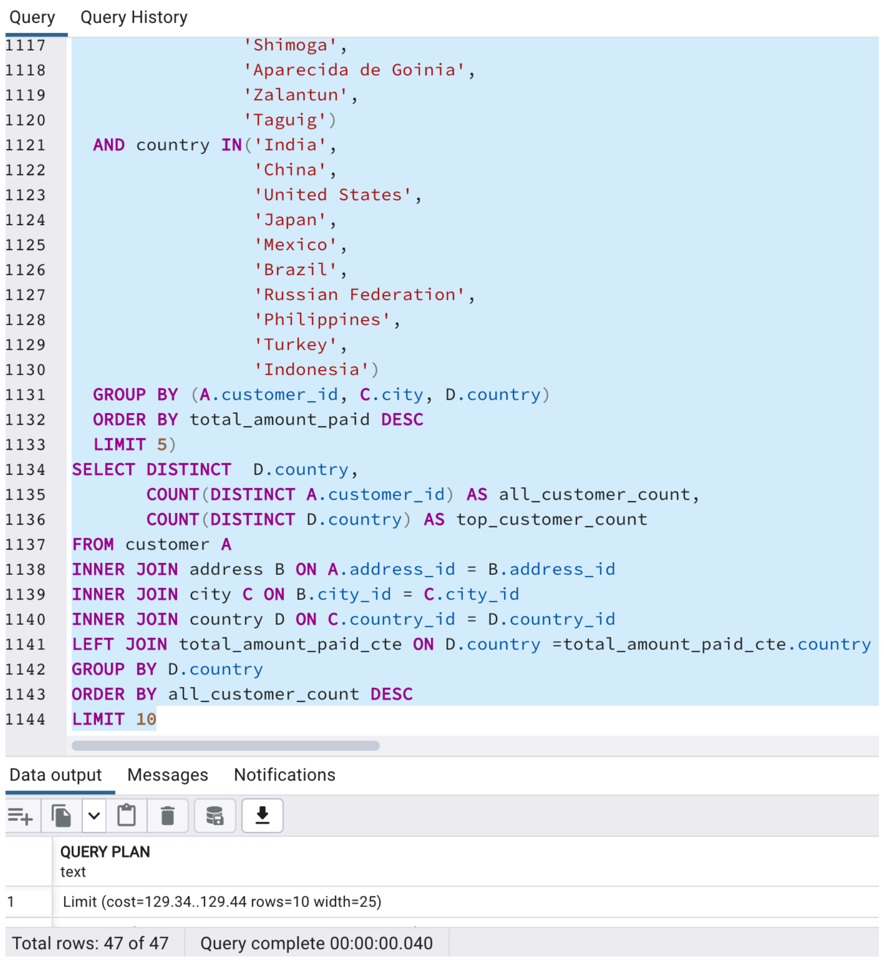
Data output Messages Notifications

```
   QUERY PLAN
   text
1  Limit (cost=129.34..129.44 rows=10 width=25)
```

Total rows: 47 of 47 Query complete 00:00:00.118

EXPLAIN CTE customers living in countries with biggest base of customers:

- Cost, 129.44
- Number of rows, 10
- Width of values in rows, 25



```
                'Shimoga',
                'Aparecida de Goinia',
                'Zalantun',
                'Taguig')
  AND country IN('India',
                 'China',
                 'United States',
                 'Japan',
                 'Mexico',
                 'Brazil',
                 'Russian Federation',
                 'Philippines',
                 'Turkey',
                 'Indonesia')
  GROUP BY (A.customer_id, C.city, D.country)
  ORDER BY total_amount_paid DESC
  LIMIT 5)
SELECT DISTINCT  D.country,
       COUNT(DISTINCT 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 total_amount_paid_cte ON D.country =total_amount_paid_cte.country
GROUP BY D.country
ORDER BY all_customer_count DESC
LIMIT 10
```

```
   QUERY PLAN
   text
1  Limit (cost=129.34..129.44 rows=10 width=25)

Total rows: 47 of 47    Query complete 00:00:00.040
```

In this case, we also have the same cost both for the subquery and the CTE ways. It seems that the reduction of costs is based on the amount of references that we have to do with inner statement, meaning that if we will use it more than once along the query, then it's worth to work with a CTE to reduce costs instead of referencing the query every time.

3. What are the challenges faced when replacing the subqueries by CTEs?

Following the instructions given during the lesson, I think it was quite straightforward to replacing queries by CTEs. I think once you understand the concept of each one and that (two of) the main differences are how many times we will reference the subquery or CTE along the main statement and how much it costs to query one or the other, it's easy to decide whereas a subquery will be used (in case is referenced once and is less costly) or a CTE instead (in case we need to make more than one reference and is less costly).

The easy part is to examine the cost and decide how many times we will reference one or the other, the hard part is to write the queries themselves.