Surekha

Ach.3 Task 3.8

Exercise 3.8

Step : 1

Query | Query History

```
1  select avg(total_amount_paid)
2  from
3  (select A.customer_id,
4  A.first_name,
5  A.last_name,
6  C.city,
7  D.country,
8  sum (E.amount) as total_amount_paid
9  from customer A
10 inner join address B on A.address_id = B.address_id
11 inner join city C on B.city_id = C.city_id
12 inner join country D on C.country_id = D.country_id
13 inner join payment E on A.customer_id = E.customer_id
14 where C.city in ('Aurora', 'Acua', 'Citrus Heights', 'Iwaki', 'Ambattur',
15                  'Shanwai', 'So leopoldo', 'Teboksary', 'Tianjin', 'Cianjur')
16 group by A.customer_id,
17 A.first_name,
18 A.last_name,
19 C.city,
20 D.country
21 order by total_amount_paid desc
22 limit 5) as average
```

Data Output | Messages | Notifications

	avg numeric
1	102.9660000000000000

Step : 2

Query | Query History

```sql
select D.country,
count (distinct A.customer_id) as all_customer_count,
count (top_5_customers) 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
(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 C.city in ('Aurora', 'Acua', 'Citrus Heights', 'Iwaki', 'Ambattur',
                 'Shanwai', 'So leopoldo', 'Teboksary', 'Tianjin', 'Cianjur')
group by A.customer_id,
A.first_name,
A.last_name,
C.city,
D.country
order by total_amount_paid desc
limit 5) top_5_customers on A.customer_id = top_5_customers.customer_id
group by D.country
order by all_customer_count desc
limit 5;
```

Data Output | Messages | Notifications

	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	India	60	1
2	China	53	0
3	United States	36	2
4	Japan	31	1
5	Mexico	30	1

Step : 3

I believe steps 1 and 2 could have been done without using subqueries with the right amount of knowledge about executing queries in SQL. The query we wrote seems a bit complex and redundant and there probably is a more optimal way of getting the results we need, possibly by just utilizing JOINS or CTEs like the reading referenced. I believe subqueries are useful when you are dealing with data that is changing all the time, when it is a complex query and you want to organize your steps, when you are performing multiple queries in one query, and when you want to make a table for the outer query to reference instead of writing a separate query.