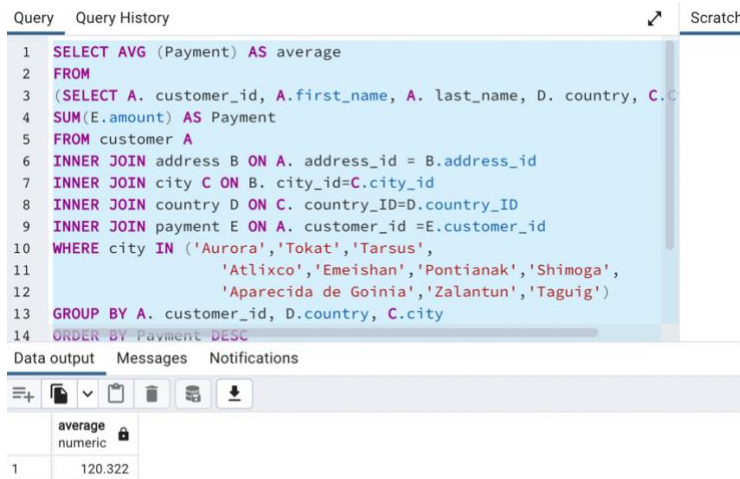


## 3.8 Answers- Subqueries

1.

```
SELECT AVG (Payment) AS average
FROM
(SELECT A. customer_id, A.first_name, A. last_name, D. country, C.City,
SUM(E.amount) AS Payment
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')
GROUP BY A. customer_id, D.country, C.city
ORDER BY Payment DESC
LIMIT 5) AS Total_amount_paid
```



The screenshot shows a SQL query editor with a query history tab. The query is the same as the one provided for question 1. Below the query editor, there is a 'Data output' tab showing the results of the query. The results are displayed in a table with two columns: 'average' and 'numeric'. The first row shows the value 120.322.

	average	numeric
1	120.322	

2.

```
SELECT DISTINCT (A.COUNTRY),
COUNT (DISTINCT D. customer_id) AS all_customer_count,
COUNT (DISTINCT A. country) as top_customer_count
FROM country A
INNER JOIN city B ON A. country_id = B.country_id
INNER JOIN address C ON B. city_id = B.city_id
INNER JOIN customer D ON C. address_id = D.address_id
LEFT JOIN(SELECT A. customer_id, A.first_name, A. last_name, D. country, C.City,
SUM(E.amount) AS Payment
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')
GROUP BY A. customer_id, D.country, C.city
ORDER BY Payment DESC
LIMIT 5) AS top_5_customers
ON A. country=top_5_customers.COUNTRY
GROUP BY A. country, top_5_customers
ORDER BY all_customer_count DESC
LIMIT 5

```

Rockbuster 1.5/postgres@PostgreSQL 14

Query Query History

```

1 SELECT DISTINCT (A.COUNTRY),
2 COUNT (DISTINCT D. customer_id) AS all_customer_count,
3 COUNT (DISTINCT A. country) as top_customer_count
4 FROM country A
5 INNER JOIN city B ON A. country_id = B.country_id
6 INNER JOIN address C ON B. city_id = B.city_id
7 INNER JOIN customer D ON C. address_id = D.address_id
8 LEFT JOIN(SELECT A. customer_id, A.first_name, A. last_name, D. country, C.City,
9 SUM(E.amount) AS Payment
10 FROM customer A
11 INNER JOIN address B ON A. address_id = B.address_id
12 INNER JOIN city C ON B. city_id=C.city_id
13 INNER JOIN country D ON C. country_ID=D.country_ID
14 INNER JOIN payment E ON A. customer_id =E.customer_id
15 WHERE city IN ('Aurora','Tokat','Tarsus',
16               'Atlixco','Emeishan','Pontianak','Shimoga',
17               'Aparecida de Goinia','Zalantun','Taguig')
18 GROUP BY A. customer_id, D.country, C.city

```

Data output Messages Notifications

	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	Afghanistan	599	1
2	Algeria	599	1
3	American Samoa	599	1
4	Angola	599	1
5	Anguilla	599	1

```

SELECT DISTINCT (A.COUNTRY),
COUNT (DISTINCT D. customer_id) AS all_customer_count,
COUNT (DISTINCT A. country) as top_customer_count
FROM country A
INNER JOIN city B ON A. country_id = B.country_id
INNER JOIN address C ON B. city_id = B.city_id
INNER JOIN customer D ON C. address_id = D.address_id
LEFT JOIN(SELECT A. customer_id, A.first_name, A. last_name, D. country, C.City,
SUM(E.amount) AS Payment
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')
GROUP BY A. customer_id, D.country, C.city
ORDER BY Payment DESC
LIMIT 5) AS top_5_customers

```

```
ON A. country=top_5_customers.COUNTRY  
GROUP BY A. country, top_5_customers  
ORDER BY all_customer_count DESC  
LIMIT 5
```

3 For query number 1, this could've been made without subqueries and using HAVE clause and aggregated functions instead, but Query number 2 needed a subquery, because this was pulling from different data tables. Subqueries are useful when there is data that is constantly changing and instead of writing the query or pulling information that is obsolete, then this