Step One:

1.

```
1 WITH total_amount_paid_cte (customer_id, first_name, last_name, country, city, total_paid) AS
 2 (SELECT A.customer_id,
            B.first_name,
 3
 4
            B.last_name,
 5
            E.country,
 6
            D.city,
 7
            SUM(A.amount) AS total_amount_paid
 8 FROM payment A
 9
        INNER JOIN customer B ON A.customer_id = B.customer_id
10
        INNER JOIN address C ON B.address_id = C.address_id
        INNER JOIN city D ON C.city_id = D.city_id
11
12
        INNER JOIN country E ON D.country_id = E.country_id
    WHERE country IN ('India', 'China', 'United States', 'Japan', 'Mexica', 'Brazil',
13
                  'Russian Federation', 'Philippines', 'Turkey', 'Indonesia')
14
    AND city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulla)', 'Kurashiki',
15
                 'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
16
17
    GROUP BY A.customer_id,
18
            B.first_name,
19
            B.last_name,
20
            E.country,
21
            D.city
ORDER BY total_amount_paid DESC LIMIT 5)
23 SELECT AVG(total_paid)
24 From total_amount_paid_cte
Data Output Explain Messages Notifications
   avg
numeric A
      97.772
```

2.

```
1 WITH top_5_cte (customer_id, first_name, last_name, country, city, total_paid) AS
2 (SELECT A.customer_id,
           B.first_name,
3
4
          B.last_name,
          E.country,
           D.city,
7
           SUM(A.amount) AS total_amount_paid
 8 FROM payment A
      INNER JOIN customer B ON A.customer_id = B.customer_id
9
       INNER JOIN address C ON B.address_id = C.address_id
10
11
      INNER JOIN city D ON C.city_id = D.city_id
12
      INNER JOIN country E ON D.country_id = E.country_id
13 WHERE country IN ('India', 'China', 'United States', 'Japan', 'Mexica', 'Brazil',
                 'Russian Federation', 'Philippines', 'Turkey', 'Indonesia')
14
15 AND city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulla)', 'Kurashiki',
                'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
```

17	GROUP BY A.customer_id,
18	B.first_name,
19	B.last_name,
20	E.country,
21	D.city
22	ORDER BY total_amount_paid DESC LIMIT 5)
23	SELECT country.country,
24	COUNT (DISTINCT customer.customer_id) AS all_customer_count,
25	COUNT (DISTINCT country.country) AS top_customer_count
26	From top_5_cte
27	LEFT JOIN customer ON customer.customer_id = customer.customer_id
28	LEFT JOIN address ON customer.address_id = address.address_id
29	LEFT JOIN city ON address.city_id = city.city_id
30	LEFT JOIN country ON city.country_id = country_id
31	GROUP BY country.country
32	ORDER BY 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

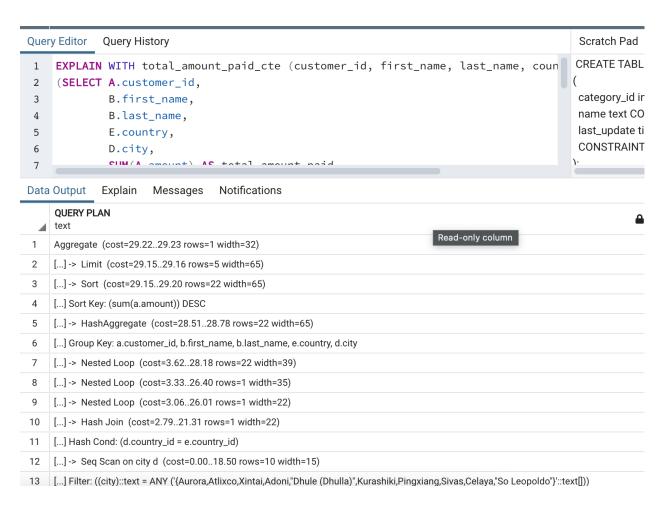
3. I started the CTE with the 'WITH' clause and named it . Then listed the columns that will be in the CTE definition using the 'AS' clause. I copied the query from the previous exercise starting with the 'SELECT' clause to get the answer.

Step two:

4.

CTE step one

Data Output Explain Messages Notifications



CTE step 2

```
Query Editor Query History
    EXPLAIN WITH top_5_cte (customer_id, first_name, last_name, country, city, total_paid) AS
     (SELECT A.customer_id,
 3
                B.first_name,
 4
                B.last_name,
                E.country,
Data Output Explain Messages Notifications
     QUERY PLAN

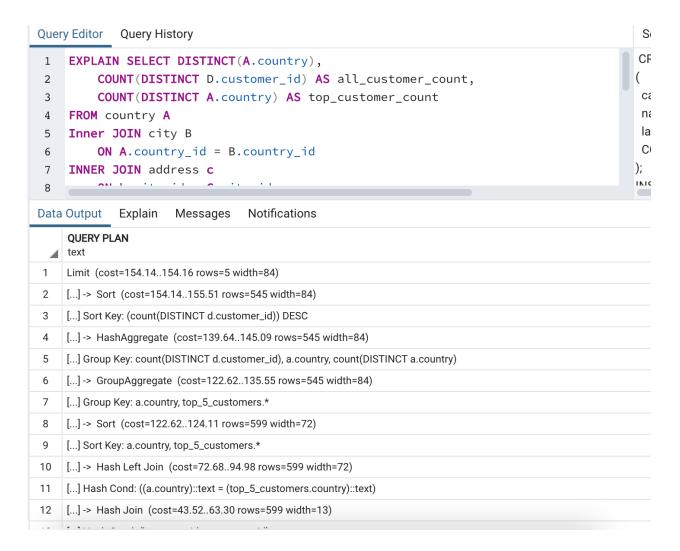
    text

 1 Limit (cost=337.24..337.25 rows=5 width=25)
 2 [...] -> Sort (cost=337.24..337.51 rows=109 width=25)
 3 [...] Sort Key: (count(DISTINCT customer.customer_id)) DESC
 4 [...] -> GroupAggregate (cost=304.38..335.42 rows=109 width=25)
 5 [...] Group Key: country.country
 6 [...] -> Sort (cost=304.38..311.87 rows=2995 width=13)
 7 [...] Sort Key: country.country
 8 [...] -> Nested Loop Left Join (cost=72.67..131.45 rows=2995 width=13)
 9 [...] -> Limit (cost=29.15..29.16 rows=5 width=65)
10 [...] -> Sort (cost=29.15..29.20 rows=22 width=65)
11 [...] Sort Key: (sum(a.amount)) DESC
12 [...] -> HashAggregate (cost=28.51..28.78 rows=22 width=65)
13 [...] Group Key: a.customer_id, b.first_name, b.last_name, e.country, d.city
14 [...] -> Nested Loop (cost=3.62..28.18 rows=22 width=39)
15 [...] -> Nested Loop (cost=3.33..26.40 rows=1 width=35)
```

Subquery step one

```
Query Editor
                                                                                                 Scra
             Query History
                                                                                                CRE
     EXPLAIN SELECT Round(AVG(total_paid),2) AS average
     FROM(SELECT A.customer_id, A.first_name, A.last_name, E.country, B.city, SUM(C
 3
          From customer A
                                                                                                 cat€
    Inner Join payment C ON A.customer_id = C.customer_id
                                                                                                 nam
 4
                                                                                                 last.
    Inner Join address D on A.address_id = D.address_id
                                                                                                 CON
 6
    Inner Join city B on D.city_id = B.city_id
    Inner JOIN country E on B.country_id = E.country_id
 7
    Where city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulla)', 'Kura INSE
 8
                    'Pingxiang', 'Sivas', 'Celaya', 'So Leopoldo')
                                                                                                VAIU
 9
10
    GROUP BY A.customer_id, E.country, B.city
11 ORDER BY total_paid DESC
12
    LIMIT 5) AS total_amount_paid;
Data Output Explain Messages Notifications
     QUERY PLAN
    Aggregate (cost=64.41..64.43 rows=1 width=32)
   [...] -> Limit (cost=64.34..64.35 rows=5 width=270)
3
   [...] -> Sort (cost=64.34..64.94 rows=242 width=270)
   [...] Sort Key: (sum(c.amount)) DESC
   [...] -> HashAggregate (cost=57.29..60.32 rows=242 width=270)
   [...] Group Key: a.customer_id, e.country, b.city
    [...] -> Nested Loop (cost=18.16..54.87 rows=242 width=28)
    [...] -> Hash Join (cost=17.88..37.14 rows=10 width=22)
```

Subquery step two



I think using the CTE is best due to the fact that if you need to reuse the query, you can just state the name you named it. There's less coding with using a CTE as well. I rewrote my subqueries so the cost results I got may not reflect which one costs more or less. I believe they should be the same cost though.