1.

Rockbuster/postgres@PostgreSQL 13

Query Editor | Query History

```sql
SELECT "Customer_ID",
       "Customer_First_Name",
       "Customer_Last_Name",
       "City",
       "Country",
       "Total_Amount_Paid",
       AVG("Total_Amount_Paid") AS "Average_Amount_Paid"
FROM
    (SELECT B.customer_id AS "Customer_ID",
            B.first_name AS "Customer_First_Name",
            B.last_name AS "Customer_Last_Name",
            E.country AS "Country",
            D.city AS "City",
            SUM(amount) AS "Total_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')
    GROUP BY "Customer_ID", "Customer_First_Name", "Customer_Last_Name", "City", "Country"
    ORDER BY "Total_Amount_Paid" DESC
    LIMIT 5) AS "subquery"
GROUP BY "Customer_ID", "Customer_First_Name", "Customer_Last_Name", "City", "Country", "Total_Amount_Paid"
ORDER BY "Total_Amount_Paid" DESC
```

Data Output | Explain | Messages | Notifications

	Customer_ID integer	Customer_First_Name character varying (45)	Customer_Last_Name character varying (45)	City character varying (50)	Country character varying (50)	Total_Amount_Paid numeric	Average_Amount_Paid numeric
1	526	Karl	Seal	Cape Coral	United States	208.58	208.5800000000000000
2	178	Marion	Snyder	Santa Brbara dOeste	Brazil	194.61	194.6100000000000000
3	181	Ana	Bradley	Memphis	United States	167.67	167.6700000000000000
4	236	Marcia	Dean	Tanza	Philippines	166.61	166.6100000000000000
5	403	Mike	Way	Valparai	India	162.67	162.6700000000000000

2.

Rockbuster/postgres@PostgreSQL 13

Query Editor | Query History

```
SELECT D.country AS "Country",
       COUNT(DISTINCT A.customer_id) AS "all_customer_count",
       COUNT(DISTINCT "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 B.customer_id AS "Customer_ID",
            B.first_name AS "Customer_First_Name",
            B.last_name AS "Customer_Last_Name",
            E.country AS "Country",
            D.city AS "City",
            SUM(amount) AS "Total_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')
            GROUP BY "Customer_ID", "Customer_First_Name", "Customer_Last_Name", "City", "Country"
            ORDER BY "Total_Amount_Paid" DESC
            LIMIT 5) AS "top_5_customers"
       ON D.country = "top_5_customers"."Country"
GROUP BY D.country
ORDER BY "top_customer_count" DESC, "all_customer_count" DESC
```

Data Output | Explain | Messages | Notifications

	Country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	United States	36	2
2	India	60	1
3	Brazil	28	1
4	Philippines	20	1
5	China	53	0
6	Japan	31	0
7	Mexico	30	0
8	Russian Federation	28	0
9	Turkey	15	0

3. I think these could be done without subqueries because the inner and outer statements reference the same tables and data. It seems like the more straightforward approach would be to use common able expressions in place of subqueries. There are situations where subqueries can be useful. For example, if we needed to look at individual customer payments as well as the average payment of all customers, in order to find the customers who have below-average payments.