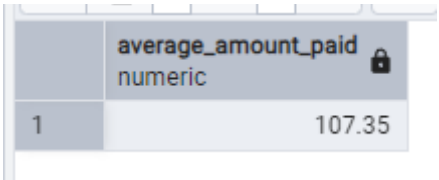
Answers 3.8

Step 1: Find the average amount paid by the top 5 customers.

```
SELECT
ROUND(AVG(total_payment), 2) AS average_amount_paid
FROM
(SELECT
c.customer_id,
CONCAT(c.first_name,' ',c.last_name) AS full_name,
co.country AS country,
ci.city AS city,
SUM(p.amount) AS total_payment
FROM customer AS c
INNER JOIN address AS a ON a.address_id = c.address_id
INNER JOIN city AS ci ON ci.city_id = a.city_id
INNER JOIN country AS co ON co.country_id = ci.country_id
INNER JOIN payment AS p ON p.customer_id = c.customer_id
WHERE ci.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)','Kurashiki', 'Pingxiang',
'Sivas', 'Celaya', 'So Leopoldo')
GROUP BY 1,2,3,4--,5,6
ORDER BY total_payment DESC
LIMIT 5
) AS total_amount_paid
```



	average_amount_paid numeric
1	107.35

Step 2: Find out how many of the top 5 customers you identified in step 1 are based within each country.

```
SELECT
co.country AS country,
COUNT(DISTINCT c.customer_id) AS all_customer_count,
COALESCE(MAX(abcd.top_customer_count),0) AS top_customer_count
FROM customer AS c
INNER JOIN address AS a ON a.address_id = c.address_id
INNER JOIN city AS ci ON ci.city_id = a.city_id
INNER JOIN country AS co ON co.country_id = ci.country_id
LEFT JOIN (SELECT
abc.country AS abccountry,
COUNT(DISTINCT abc.customer_id) AS top_customer_count
```

```
FROM
(SELECT
c.customer_id,
CONCAT(c.first_name,' ',c.last_name) AS full_name,
co.country AS country,
ci.city AS city,
SUM(p.amount) AS total_payment
FROM customer AS c
INNER JOIN address AS a ON a.address_id = c.address_id
INNER JOIN city AS ci ON ci.city_id = a.city_id
INNER JOIN country AS co ON co.country_id = ci.country_id
INNER JOIN payment AS p ON p.customer_id = c.customer_id
WHERE ci.city IN ('Aurora', 'Atlixco', 'Xintai', 'Adoni', 'Dhule (Dhulia)','Kurashiki', 'Pingxiang',
'Sivas', 'Celaya', 'So Leopoldo')
GROUP BY 1,2,3,4--,5,6
ORDER BY total_payment DESC
LIMIT 5) AS abc
GROUP BY abc.country) AS abcd ON abcd.abccountry = co.country
GROUP BY 1
ORDER BY all_customer_count DESC
```

	country character varying (50)	all_customer_count bigint	top_customer_count bigint
1	India	60	1
2	China	53	0
3	United States	36	1
4	Japan	31	0
5	Mexico	30	2
6	Brazil	28	0
7	Russian Federation	28	0
8	Philippines	20	0
9	Turkey	15	1
10	Indonesia	14	0

Step 3

Steps 1 and 2 could be done without subqueries by restructuring the query with `JOIN` statements and aggregates, but this would make it harder to read and maintain. Subqueries simplify complex logic by breaking it into manageable parts, especially when intermediate results like the top 5 customers are needed.

Subqueries are most useful for modular query design, temporary filtering, or aggregation before joining other tables. They improve readability and make handling layered conditions easier.