**Day – 6**

🔑 Step-by-Step Learning Roadmap (SQL Pro Journey)

✅ Phase 1: Core Foundations (You already covered!)

* SELECT, WHERE, ORDER BY, LIMIT
* Logical operators (AND, OR, NOT, BETWEEN, LIKE, IN)
* Aggregations: COUNT, SUM, AVG, MIN, MAX
* Grouping: GROUP BY, HAVING, offset. Distinct.

👉 You’re already strong here. These are 70% of real interview queries**. Day – 6 Sabquery(Update and alter function), 7 – Joins, Day 8- Case when, Windows function – Day – 9** | Day 10 – Dobut session, Part – 30 Q&A, Part- 2 (50 Q&A), part – 3 (100-Q&A)

**🌟 What is a Subquery?, Kyon Subquery ka name Rakha gaya hai. Kya zarurat Hai.**

👉 A subquery is simply a query inside another query. (Select \* from bank\_trasanctions (select Avg amount from table\_name)  
Think of it like a “query within parent**heses” that gives a result, which the main query uses.**

**Subquery = a query inside another query, used when you need a calculated or filtered result before applying the main condition.**

**In Detail – Step we will be discussed**

**🔎 What does “query inside parentheses” mean?**

**You already know this query:**

**SELECT customer\_name, AVG(amount) as avgamount FROM bank\_transactions;**

**👉 This query gives one number → the average balance of all customers.  
Let’s say result = ₹50,000.**

**Bank\_Transactions (example data)**

| **customer\_name** | **branch\_name** | **transaction\_type** | **amount** |
| --- | --- | --- | --- |
| Ram | Delhi | deposit | 50,000 |
| Sita | Mumbai | deposit | 40,000 |
| Mohan | Delhi | deposit | 70,000 |
| Radha | Kolkata | withdrawal | 30,000 |
| Gita | Mumbai | deposit | 90,000 |

**Question:**

**Find customers whose total transactions as total\_amount , avg(amount) as avg\_amount\_customerwise,**

**are greater than the average transaction amount of all customers.**

Step 1: First calculate the average (without subquery)

SELECT AVG(amount) FROM bank\_transactions;

Suppose the result is: 56,000

**Step 2: Use it manually in WHERE**

SELECT customer\_name, SUM(amount) AS total\_amount

FROM bank\_transactions

GROUP BY customer\_name

HAVING SUM(amount) > 56000;

✅ Works, but problem:

* If data changes, average changes → you must edit query manually. ❌

Step 1 → For each customer, calculate the total amount of all their transactions (SUM(amount)).

select customer\_name,

sum(amount) as percusotmerwise\_totalTransaction\_amount

from bank\_transactions

group by customer\_name;

------

Step 2 → Calculate the average transaction amount across all customers (based on totals, not individual transactions).

select avg(amount)

from bank\_transactions;

------

Step 3 → Compare each customer’s total with that overall average.

------

Step 4 → Return those customers whose total > average of all customers.

**Step 3: Use Subquery**

Now instead of writing 56000, we insert the query inside:

SELECT customer\_name, SUM(amount) AS total\_amount

FROM bank\_transactions

GROUP BY customer\_name

HAVING SUM(amount) > (SELECT AVG(amount) FROM bank\_transactions);

✔ Here:

* (SELECT AVG(amount) FROM bank\_transactions) → calculates average on the fly.
* Outer query checks each customer’s SUM(amount) against that average.
* Always correct, even if new data is added.

⚡ That’s the **power of subquery** → it removes hardcoding.  
Think: *Whenever I see “compare against overall average / overall max / overall min” → I need a subquery.*

SELECT customer\_name,

SUM(amount) AS total\_amount

FROM bank\_transactions

GROUP BY customer\_name

HAVING SUM(amount) > (

SELECT AVG(total\_amount)

FROM (

SELECT customer\_name, SUM(amount) AS total\_amount

FROM bank\_transactions

GROUP BY customer\_name

) AS customer\_totals

)

ORDER BY total\_amount DESC;

**✅ Easy Subquery Questions**

1. **Find customers who made at least one transaction higher than the average transaction amount.**

SELECT customer\_name, amount

FROM bank\_transactions

WHERE amount > (SELECT AVG(amount) FROM bank\_transactions);

List all customers who have made transactions in the same branch as ‘Paresh’.

SELECT DISTINCT customer\_name

FROM bank\_transactions

WHERE branch\_name IN (

SELECT branch\_name

FROM bank\_transactions

WHERE customer\_name = 'Paresh'

);

Find the branches where the total transactions are above the overall average of all branches.

SELECT branch\_name, SUM(amount) AS total\_branch\_amount

FROM bank\_transactions

GROUP BY branch\_name

HAVING SUM(amount) > (

SELECT AVG(total\_amount)

FROM (

SELECT branch\_name, SUM(amount) AS total\_amount

FROM bank\_transactions

GROUP BY branch\_name

) AS branch\_totals

);

Show customers who made a transaction equal to the maximum transaction amount.

SELECT customer\_name, amount

FROM bank\_transactions

WHERE amount = (SELECT MAX(amount) FROM bank\_transactions);

Find customers who never made a withdrawal.

SELECT DISTINCT customer\_name

FROM bank\_transactions

WHERE customer\_name NOT IN (

SELECT DISTINCT customer\_name

FROM bank\_transactions

WHERE transaction\_type = 'withdrawal'

);

Find customers who made at least one deposit in the same branch where ‘Ravi’ has transacted.

SELECT DISTINCT customer\_name

FROM bank\_transactions

WHERE branch\_name IN (

SELECT DISTINCT branch\_name

FROM bank\_transactions

WHERE customer\_name = 'Ravi'

);

List customers whose total amount is greater than the average total amount of all customers.

SELECT customer\_name, SUM(amount) AS total\_amount

FROM bank\_transactions

GROUP BY customer\_name

HAVING SUM(amount) > (

SELECT AVG(total\_amount)

FROM (

SELECT customer\_name, SUM(amount) AS total\_amount

FROM bank\_transactions

GROUP BY customer\_name

) AS customer\_totals

);

Show branches where at least one customer’s total is greater than ₹1,00,000.

SELECT DISTINCT branch\_name

FROM bank\_transactions

WHERE customer\_name IN (

SELECT customer\_name

FROM bank\_transactions

GROUP BY customer\_name

HAVING SUM(amount) > 100000

);

Find the customer(s) with the highest total transaction amount.

SELECT customer\_name, SUM(amount) AS total\_amount

FROM bank\_transactions

GROUP BY customer\_name

HAVING SUM(amount) = (

SELECT MAX(total\_amount)

FROM (

SELECT customer\_name, SUM(amount) AS total\_amount

FROM bank\_transactions

GROUP BY customer\_name

) AS totals

);

Show all transactions that are above the average transaction amount of their own branch. (correlated subquery → a bit advanced, but still easy to read)

SELECT customer\_name, branch\_name, amount

FROM bank\_transactions bt

WHERE amount > (

SELECT AVG(amount)

FROM bank\_transactions

WHERE branch\_name = bt.branch\_name

);