Subqueries in MySQL

A **subquery** is a query nested inside another query. Subqueries are useful for breaking down complex problems into smaller parts.

They can be used in:

- `SELECT` statements
- `WHERE` clauses
- `FROM` clauses

Example Scenario: Salary Comparison

Suppose we want to find all users who **earn more than the average salary** of all users.

Scalar Subquery Example

This subquery returns **a single value** — the average salary — and we compare each user's salary against it.

```
SELECT id, name, salary
FROM users
WHERE salary > (
    SELECT AVG(salary) FROM users
);
```

Explanation:

- The inner query: `SELECT AVG(salary) FROM users` returns the average salary.

- The outer query selects all users with a salary **greater than that average**.

Subquery with `IN`

Now let's say we want to find users who have been referred by someone who earns more than ₹75,000.

```
SELECT id, name, referred_by_id
FROM users
WHERE referred_by_id IN (
    SELECT id FROM users WHERE salary > 75000
);
```

Explanation:

- The inner query: `SELECT id FROM users WHERE salary > 75000` returns a list of user IDs (referrers) who earn more than ₹75,000.
- The outer query selects users whose `referred_by_id` is in that list.

Other Places Subqueries Are Used

You can also use subqueries:

- Inside `SELECT` columns (called scalar subqueries)
- In the `FROM` clause to create derived tables

Example in `SELECT`:

```
SELECT name, salary,
  (SELECT AVG(salary) FROM users) AS average_salary
FROM users;
```

This shows each user's salary along with the overall average.

Summary

Subquery Type	Use Case
Scalar Subquery	Returns one value (e.g. AVG, MAX)
Subquery with `IN`	Returns multiple values
Subquery in `SELECT`	Shows related calculated value
Subquery in `FROM`	Acts as a virtual table

Subqueries are powerful tools when filtering based on computed or dynamic conditions.