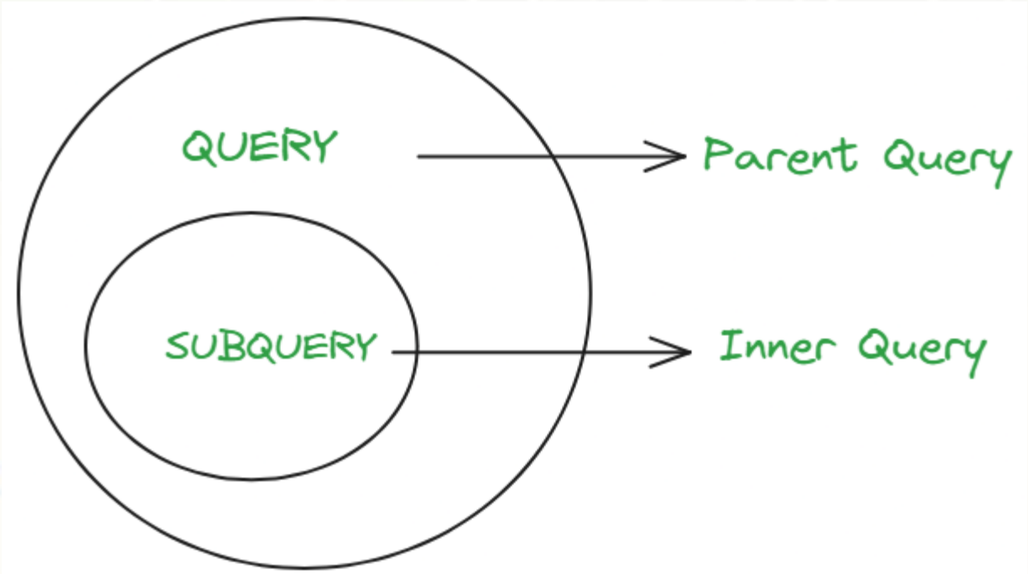
Introduction to SQL Subqueries

Subqueries are a fundamental part of SQL. They allow us to embed queries within other queries, enabling more complex data manipulation and analysis. Subqueries can be used to filter data, retrieve specific information, and perform calculations based on the results of other queries.



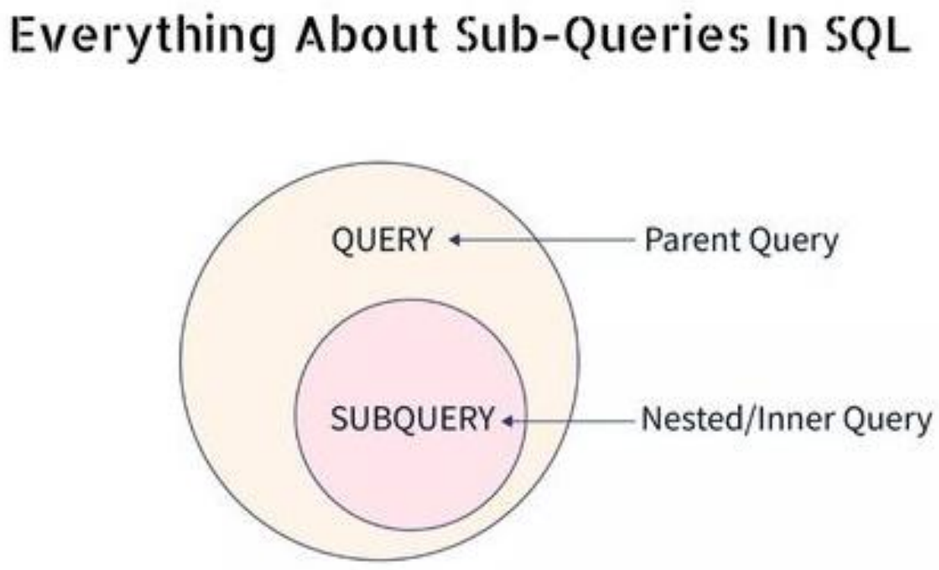
What is a Subquery?

A subquery is a query embedded within another query. The inner subquery is executed first, and its results are then used by the outer, main query.

1

Inner Subquery

The nested query that retrieves the initial data set.



2

Outer Query

The primary query that utilizes the results from the inner subquery to perform further data manipulation or analysis.

Types of Subqueries

Subqueries can be classified based on their relationship with the outer query. They can be correlated, uncorrelated, or nested.

Correlated

The inner query depends on the outer query. Each iteration of the outer query runs the inner query with the current row's value.

Uncorrelated

The inner query is independent of the outer query. It executes once, returning a set of results used by the outer query.

Nested

Multiple subqueries are nested within each other. The innermost query executes first, and the results are used by the next query, and so on.

Correlated Subqueries

Correlated subqueries rely on the outer query's data. They are executed for each row processed by the outer query, using values from that row.

Outer Query	Inner Query
SELECT * FROM customers	WHERE EXISTS (SELECT * FROM orders WHERE orders.customer_id = customers.customer_id)

Select * from customer c, order o where c. c_id = o.c_id;

Uncorrelated Subqueries

Uncorrelated subqueries are independent of the outer query. They execute once and return a set of results used by the outer query, regardless of the outer query's data.

Outer Query	Inner Query
SELECT * FROM employees	WHERE salary > (SELECT AVG(salary) FROM employees)

Subqueries in the SELECT clause

Subqueries in the SELECT clause return data that is used to create a new column in the result set. The subquery's result is used as an expression within the SELECT statement.

Outer Query	Inner Query
SELECT employee_name, (SELECT AVG(salary) FROM employees) AS average_salary	FROM employees

Subqueries in the WHERE clause

Subqueries in the WHERE clause filter the rows of the main query. The subquery returns a set of values that are used in the WHERE clause to compare against the main query's data.

Outer Query	Inner Query
SELECT * FROM customers	WHERE customer_id IN (SELECT customer_id FROM orders WHERE order_date = '2023-08-01')

Subqueries in the FROM clause

Subqueries in the FROM clause act as a derived table, creating a temporary table from the subquery's results. This temporary table is then used in the main query.

Outer Query	Inner Query
SELECT * FROM (SELECT employee_id, employee_name FROM employees) AS employee_data	WHERE salary > (SELECT AVG(salary) FROM employees)

Subqueries in the HAVING clause

Subqueries in the HAVING clause filter groups of rows based on the results of aggregate functions. They are executed after the GROUP BY clause, and the subquery's results are used to compare against the aggregated values.

Outer Query	Inner Query
SELECT department, COUNT(*) AS total_employees FROM employees GROUP BY department HAVING COUNT(*) > (SELECT AVG(COUNT(*)) FROM employees GROUP BY department)	

Nested Subqueries

Nested subqueries involve one or more subqueries nested inside other subqueries. The innermost subquery executes first, and its results are used by the next outer subquery, and so on.

Outer Query	Inner Query 1	Inner Query 2
SELECT * FROM employees WHERE salary > (SELECT MAX(salary) FROM (SELECT salary FROM employees WHERE department = 'Sales') AS top_sales_salaries)	SELECT salary FROM employees WHERE department = 'Sales'	

Types of Subqueries

1.Single-row Subquery: Returns only one row as the result.

Example: Find the employee with the highest salary.

```
SELECT Name FROM Employees WHERE Salary = (SELECT MAX(Salary) FROM Employees);
```

2.Multi-row Subquery: Returns multiple rows as the result.

Example: Find employees whose salaries match those in another department.

```
SELECT Name FROM Employees WHERE Salary IN (SELECT Salary FROM Employees WHERE DepartmentID = 102);
```

3.Scalar Subquery: Returns a single value (one row and one column).

Example: Find the average salary of employees and display it alongside their names.

```
SELECT Name, Salary, (SELECT AVG(Salary) FROM Employees) AS AvgSalary FROM Employees;
```

4.Correlated Subquery: Refers to a column in the outer query, making the subquery execute once for each row of the outer query.

Example: Find employees who earn more than the average salary of their department.

```
SELECT Name FROM Employees e1 WHERE Salary > (SELECT AVG(Salary) FROM Employees e2 WHERE e1.DepartmentID = e2.DepartmentID);
```

5.Nested Subquery: A subquery within another subquery.

Example: Find employees in the department with the highest average salary.

```
SELECT Name FROM Employees WHERE DepartmentID = ( SELECT DepartmentID FROM Departments WHERE AvgSalary = ( SELECT MAX(AvgSalary) FROM (SELECT AVG(Salary) AS AvgSalary, DepartmentID FROM Employees GROUP BY DepartmentID) AS Temp ) );
```