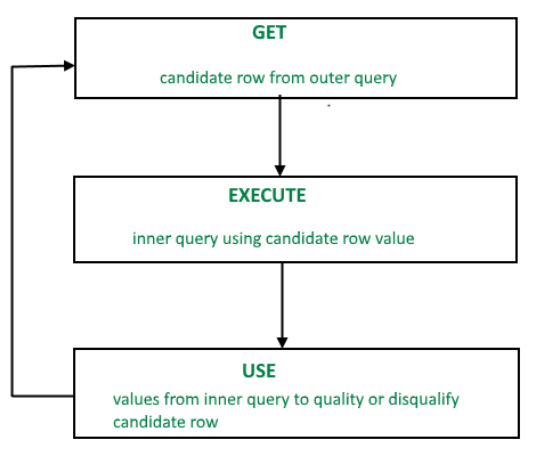
**Database Management System**

**Practical No : 8**

**Aim :** Understand the concept of Co-related subqueries.

Correlated subqueries are used for row-by-row processing. Each subquery is executed once for every row of the outer query.



A correlated subquery is evaluated once for each row processed by the parent statement. The parent statement can be a SELECT, UPDATE, or DELETE statement.

SELECT column1, column2, ....

FROM table1 outer WHERE column1 operator SELECT column1, column2 FROM table2 WHERE expr1 = outer.expr2);

A correlated subquery is one way of reading every row in a table and comparing values in each row against related data. It is used whenever a subquery must return a different result or set of results for each candidate row considered by the main query. In other words, you can use a correlated subquery to answer a multipart question whose answer depends on the value in each row processed by the parent statement.

**Nested Subqueries Versus Correlated Subqueries :**

**Nested subquery:** The inner SELECT query runs first and executes once.

**Correlated subqueries**: Executes once for each candidate row considered by the outer query.

**Nested subquery Example:**

Select ename, city, from emp where salary =(select max(salary) from emp)

The inner query executes once.

**Correlated subqueries Example:**

SELECT last\_name, salary FROM employees outer WHERE salary >(SELECT AVG(salary)FROM employees WHERE department\_id = outer.department\_id);

The inner query executes for every row because outer.department\_id is different for all.

**CORRELATED UPDATE :**

UPDATE table1 alias1

SET column = (SELECT expression

FROM table2 alias2

WHERE alias1.column =

alias2.column);

Use a correlated subquery to update rows in one table based on rows from another table.

**CORRELATED DELETE :**

DELETE FROM table1 alias1

WHERE column1 operator

(SELECT expression

FROM table2 alias2

WHERE alias1.column = alias2.column);

Use a correlated subquery to delete rows in one table based on the rows from another table.

**Examples:**

select p.last\_name, p.department\_id from employees p where p.salary < (select avg(s.salary) from employees s where s.department\_id=p.department\_id)

The flow of execution is as follows:

1. Start at the first row of the EMPLOYEES table.
2. Read the DEPARTMENT\_ID and SALARY of the current row.
3. Run the subquery using the DEPARTMENT\_ID from step 2
4. Compare the result of step 3 with the SALARY from step 2, and return the row if the SALARY is less than the result.
5. Advance to the next row in the EMPLOYEES table.
6. Repeat from step 2.

**Exercise**

1. Write a query to determine who earns more than Mr. Lex.

2. Write a query find the job with the highest average salary.

3. Write a query to print employee id along with their manager id.

4. Write a query to print employees name who earns more than that of their managers.

5. Write a query to print the name of employee who earns more than that of their department’s average.

6. Write a query to print the name and id of those department whose total no employee is greater than at least one of the departments.

7. Write a query to print the name of employee who draw second lowest salary in the company.

8. Write a query to print the name of employee who draw second lowest salary in their respective departments.

What should be the output of the following query:

9. select employee\_id from employees where salary < all (select salary from employees where department\_id=30);

10. select employee\_id from employees where salary < (select min(salary) from employees where department\_id=60);

11. select last\_name from employees where department\_id in(select department\_id from departments where department\_name=’Sales’);