Subquery

- Query within another query is called Subquery/ nested query.
- Subquery is used to retrieve data from single or multiple tables by using more than one step.

Types of Sub Query

1. Non correlated Subquery
    - In non correlated subqueries child query is executed first then the parent query is executed.
2. Correlated Sub query
    - In correlated subqueries parent query is executed first then the child query is executed.

Non correlated Subquery

- Child Query
    - A query which provides value to another query is called child query/ inner query.
- Parent Query
    - A query which receives values from another query is called parent query / outer query.

Types of Non correlated Subquery

1. Single row sub query
2. Multiple row sub query
3. Multiple column sub query
4. Inline view sub query

Single row sub query

- In single row sub query, child query returns a single value (i.e. Single row having single column).
- We can use =, <, >, <=, >=, <>, between and operators.

Multiple row sub query

- In multiple row sub query, child query returns multiple value (i.e. multiple row having single column).
- In multiple row sub query we are using in operator.

Multiple column sub query

- In multiple column sub query, child query returns more than one column.

Inline view sub query

- In inline views we are using Subquery in place of table with in parent query.

Example#1

Write a query to display the employees who are getting more salary than the average salary from employee table.

ename	salary
Ram	30000.00
Raj	35000.00
Rahul	40000.00
Alok	60000.00
Sunil	35000.00
Tushar	50000.00

Requirement

ename	salary
Alok	60000.00
Tushar	50000.00

```
CREATE TABLE employee(
        ename varchar(20),
   salary decimal(8,2)
);

INSERT INTO employee VALUES
('Ram', 30000), ('Raj', 35000), ('Rahul', 40000),
('Alok', 60000), ('Sunil', 35000), ('Tushar', 50000);

SELECT * FROM employee;
SELECT avg(salary) FROM employee;
SELECT * FROM employee WHERE salary > (SELECT avg(salary) FROM employee);
```

Example#2

Write a query to display the employees who are working in IT department from employee, department tables by using sub query.

department_id	department_name
1	IT
2	HT
3	Finance

employee_id	employee_name	department_id
101	Raj	1
102	Alok	2
103	Sunil	1
104	Ram	3
105	Tushar	2

Requirements

employee_id	employee_name	department_id
101	Raj	1
103	Sunil	1

```
CREATE TABLE departments (
   department_id INT,
   department_name VARCHAR(50)
);
CREATE TABLE employees (
   employee_id INT,
   employee_name VARCHAR(100),
   department_id INT
);

INSERT INTO departments (department_id, department_name)
VALUES
   (1, 'IT'),
   (2, 'HT'),
   (3, 'Finance');

INSERT INTO employees (employee_id, employee_name, department_id)
VALUES
   (101, 'Raj', 1),
   (102, 'Alok', 2),
   (103, 'Sunil', 1),
```

```
    (104, 'Ram', 3),
    (105, 'Tushar', 2);
```

```
SELECT * FROM employees WHERE department_id =

(SELECT department_id FROM departments WHERE department_name='IT');
```

Example#3

Write a query to display the employees who are getting more salary than the height paid employee in '1' department.

employee_id	employee_name	salary	department_id
101	Raj	30000.00	1
102	Alok	40000.00	2
103	Sunil	35000.00	1
104	Ram	45000.00	3
105	Tushar	30000.00	2

Requirements

employee_id	employee_name	salary	department_id
102	Alok	40000.00	2
104	Ram	45000.00	3

```
CREATE TABLE departments (
    department_id INT,
    department_name VARCHAR(50)
);
CREATE TABLE employees (
    employee_id INT,
    employee_name VARCHAR(100),
    salary decimal(8,2),
    department_id INT
);

INSERT INTO departments (department_id, department_name)
VALUES
    (1, 'IT'),
```

```
  (2, 'HR'),
  (3, 'Finance');

INSERT INTO employees VALUES
  (101, 'Raj', 30000, 1),
  (102, 'Alok', 40000, 2),
  (103, 'Sunil', 35000, 1),
  (104, 'Ram', 45000, 3),
  (105, 'Tushar', 30000, 2);


SELECT max(salary) FROM employees WHERE department_id=1;

SELECT * FROM employees

WHERE salary > (SELECT max(salary) FROM employees WHERE department_id=1);
```

Example#4

Write a query to display the employees who are getting more salary than the height paid employee in 'IT' department.

```
SELECT * FROM employees WHERE salary >
(SELECT max(salary) FROM employees WHERE department_id=
(SELECT department_id FROM departments WHERE department_name='IT'));
```

Example#5

Write a query to display the employees who are getting more salary than 'Sunil' and works in the department where 'Tushar' is works.

```
SELECT * FROM employees
WHERE salary > (SELECT salary FROM employees WHERE employee_name='Sunil')
AND department_id = (SELECT department_id FROM employees WHERE
employee_name='Tushar');
```

Example#6

Write a query to display the second highest salary from employee table.

```
SELECT max(salary) FROM employees
WHERE salary < (SELECT max(salary) FROM employees);
```

Example#7

Write a query to display the second highest salary employee details from employee table.

```
SELECT * FROM employees where salary = (
SELECT max(salary) FROM employees
WHERE salary < (SELECT max(salary) FROM employees));
```

Example#8

Write a query to display the nth highest salary employee details from employee table.

```
SELECT * FROM employees e1 where 3 = (
SELECT count(distinct(salary)) FROM employees e2 WHERE e2.salary >= e1.salary);
```

Multiple row sub query

Example#1

Write a query to display employee details those are getting maximum salary in each departments.

```
SELECT * FROM employees WHERE salary =
(SELECT  max(salary) FROM employees GROUP BY department_id);
```

```
SELECT * FROM employees WHERE salary in
(SELECT  max(salary) FROM employees GROUP BY department_id);
```

Example#2

Write a query to display employees who are working in either IT or HR departments.

```
SELECT * FROM employees
WHERE department_id in (SELECT department_id FROM departments
WHERE department_name in ('IT','HR'));
```

Example#3

Write a query to list the department names which have no employee.

```
SELECT * FROM departments
WHERE department_id not in (SELECT department_id FROM employees);
```

Multiple Column Sub Query

Example#1

Write a query to find the employees whose department_id and salary are same as 'Alok'.

```
SELECT * FROM employees WHERE (department_id, salary) IN (SELECT department_id, salary
FROM employees WHERE employee_name='Alok');
```

Inline Views/ Derived Table

- In inline views we are using Subquery in place of table with in parent query.
- Syntax:
  ```
  select * from (subquery)
  ```

Example#1

Write a query to display employee those are getting more than 420000 annual salaries.

```
SELECT * FROM (
SELECT employee_name, salary, salary*12 as total_sal FROM employees ) AS emp
WHERE total_sal > 420000;
```

Example#2

Write a query to display employee those are getting minimum annual salaries.

```
SELECT * FROM (SELECT *, salary*12 as total_sal FROM employees ) AS emp
WHERE total_sal = (SELECT min(salary*12) FROM employees );
```

Example#3

Write a query to display lowest average salary based on department from employee table.

```
SELECT min(salary) FROM (SELECT avg(salary) as salary FROM employees GROUP BY
department_id) as table1;
```

Example#4

Write a query to display departments having lowest average salary from employee table.

```
SELECT department_id, avg(salary)
FROM employees group by department_id
HAVING avg(salary) = (SELECT min(sal) FROM (SELECT department_id, avg(salary) as sal
FROM employees group by department_id) as table1);
```

Example#5

Write a query to display those departments where average salary more than the average salary of department id 1.

```
SELECT department_id, avg(salary)
FROM employees group by department_id
HAVING avg(salary) > (SELECT avg(salary)
FROM employees WHERE department_id=1);
```

Example#6

Write a query to display department having maximum number of employees.

```
SELECT department_id, count(*) FROM employees
GROUP BY department_id
HAVING count(*) = (SELECT max(emp_count) FROM
( SELECT count(*) as emp_count FROM employees group by department_id) as table1);
```

Co-related Subquery

- In co-related Subquery, parent query is executed first and then child query is executed.
- In co-related Subquery, child query is executed for each row for the parent query table.
- In co-related Subquery, we must create alias name in the parent query table and then we will use that alias name in child query WHERE condition.

Syntax

```
SELECT * FROM tablename aliasname
WHERE columnname= (SELECT * FROM tablename
WHERE columnname = alianame.columnname)
```

Example#1

- Write a query to display the employees who are getting more salary than average salary of their department using correlated sub query.

```
SELECT * FROM employees emp
WHERE salary > (select avg(salary) FROM employees WHERE department_id =
emp.department_id);
```

Example#2

Write a query to find 1st highest salary in employee table.

Approach1:

```
SELECT * FROM employees e1
WHERE 1 =  (SELECT count(distinct(salary)) FROM employees e2
WHERE e2.salary >= e1.salary);
```

Execution Process

Step1:

- Get candidate row (first row)

Step2:

- execute child query
- candidate row value will be substituted
- return result

step3:

- execute parent query
- when condition is false candidate row does not return into result

Repeat process for all rows in parent query table

Approach2

```
SELECT columnlist FROM table1 t1
WHERE (N-1) = (SELECT COUNT(DISTINCT(salary)) FROM table1 t2
        WHERE t1.salary < t2.salary)
```

```
SELECT * FROM employees e1
WHERE (1-1) =  (SELECT count(distinct(salary)) FROM employees e2
WHERE e1.salary < e2.salary);
```

Exists Operator

- Exists operator returns Boolean value.
- It is used in where condition of the parent query.
- We are not allowed to use column name along with exists operator.
- If we want to test one table column values are available in another related table column value then we can use co-related Subquery exists operator.

Syntax

```
SELECT * FROM tablename aliasname
WHERE exists (SELECT * FROM tablename
WHERE columnname = aliasname.columnname)
```

Example#1

Write a query to display those departments having employees using co-related Subquery.

```
SELECT * FROM departments dept
WHERE  exists (SELECT * FROM employees
WHERE department_id = dept.department_id)
```

Example#2

Write a query to display those departments without having employees using co-related Subquery.

```
SELECT * FROM departments dept
WHERE  not exists (SELECT * FROM employees
WHERE department_id = dept.department_id)
```

Example#3

Write a query to display those departments having employees in employee table using non co-related Subquery.

```
SELECT * FROM departments dept
WHERE  department_id in (SELECT department_id FROM employees)
```

Example#4

Write a query to display those departments without having employees in employee table using non co-related Subquery.

```
SELECT * FROM departments dept
WHERE  department_id not in (SELECT department_id FROM employees)
```

Note

- Not in operator will not work with null values. Not exists operator will work with null values.

Handling NULL values in NOT IN Operator

```
SELECT * FROM departments

WHERE department_id  NOT IN ( SELECT department_id FROM employees

WHERE department_id is not null);
```

```
SELECT * FROM departments

WHERE department_id  NOT IN ( SELECT coalesce(department_id,0) FROM employees

WHERE department_id);
```

```
SELECT * FROM departments

WHERE department_id  NOT IN ( SELECT ifnull(department_id,0) FROM employees

WHERE department_id);
```

Example#5

Write a query to display the employees who are getting same salary as ‘Alok’ by using correlated Subquery.

```
SELECT * FROM employees emp1
WHERE exists (SELECT * FROM employees
WHERE employee_name='Alok' and salary = emp1.salary)
```

Example#6

```
SELECT * FROM employees
WHERE exists (SELECT * FROM employees
WHERE employee_name='Alok');
```

Example#7

```
SELECT * FROM employees
WHERE exists (SELECT * FROM employees
WHERE employee_name='Alok');
```

Limit

- It is used to restrict rows in a table.
- Syntax
  SELECT * FROM tablename limit row_indes,number_of_row;

Example#1

Write a query to display 1st row from employee table.

```
SELECT * FROM employees limit 0,1;
```

Example#2

Write a query to display the second highest salary employee details from employee table.

```
SELECT * FROM employees ORDER BY salary DESC limit 1,1;
```

Example#3

```
SELECT * FROM employees LIMIT 1;

SELECT * FROM employees LIMIT 1,2;

SELECT * FROM employees LIMIT 0,3;


# Display 101,102, 104, 106

(SELECT * FROM employees LIMIT 0,2)

UNION

(SELECT * FROM employees LIMIT 3,1)

UNION

(SELECT * FROM employees LIMIT 5,1)
```

Example#4

Write a query to display last 2 records from a table using the LIMIT.

```
SELECT * FROM employees ORDER BY employee_id DESC LIMIT 2;
```

Display in Original Order

```
SELECT * FROM
```

(SELECT * FROM employees ORDER BY employee_id DESC LIMIT 2) AS emp ORDER BY employee_id ASC;

Questions

1. What is sub query?
2. What is the difference between non correlated and correlated sub query?
3. What is single row sub query?
4. What is multiple row sub query?
5. What is multiple column sub query?
6. What is inline view sub query?
7. What is correlated sub query?
8. Write query to find the Nth highest salary?
9. Write a query to find the highest salary in each department?
10. What is the difference between sub query and join?