Nested Queries and Subqueries

What is a Subquery?

If a sql statement contains another sql statement then the sql statement which is inside another sql statement is called Subquery. It is also known as nested query.

A Subquery or Inner query or Nested query is a query within another SQL query and embedded within the WHERE clause.

A subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved.

Subqueries can be used with the SELECT, INSERT, UPDATE, and DELETE statements along with the operators like =, , >=, <=, IN, BETWEEN etc.

Rules of Subqueries:

There are a few rules that subqueries must follow:

- Subqueries must be enclosed within parentheses.
- A subquery can have only one column in the SELECT clause, unless multiple columns are in the main query for the subquery to compare its selected columns.
- An ORDER BY cannot be used in a subquery, although the main query can use an ORDER BY. The GROUP BY can be used to perform the same function as the ORDER BY in a subquery.
- Subqueries that return more than one row can only be used with multiple value operators, such as the IN operator.
- The SELECT list cannot include any references to values that evaluate to a BLOB, ARRAY, CLOB, or NCLOB.
- A subquery cannot be immediately enclosed in a set function.
- The BETWEEN operator cannot be used with a subquery; however, the BETWEEN operator can be used within the subquery.

Let’s do some hands on now:

Suppose we have below job_role table:

123 id	ABC job_category	ABC country	123 salary
1	Data Scientist	India	80,000
2	ML Engineer	US	120,000
3	Developer	India	70,000
4	Data Analyst	India	65,000
5	Data Analyst	India	60,000
6	Developer	US	110,000
7	Data Scientist	US	150,000
8	Data Analyst	US	100,000
9	Data Scientist	UK	140,000
10	Data Scientist	UK	160,000
11	Data Analyst	UK	150,000
12	Data Scientist	US	200,000
13	Data Scientist	US	300,000
14	Developer	UK	151,000
15	Data Analyst	UK	101,000
16	Developer	UK	99,000
17	Developer	India	50,000
18	ML Engineer	India	55,000
19	ML Engineer	US	125,000
20	Developer	India	40,000

Subqueries with the SELECT Statement

Basic Syntax:

```
-- basic syntax

SELECT column_name [, column_name ]
FROM table1 [, table2 ]
WHERE column_name OPERATOR
 (SELECT column_name [, column_name ]
 FROM table1 [, table2 ]
 [WHERE]);
```

From above job_role table if we want to display records of having salary > 120000.

We can do it from below query:

```sql
SELECT *
 FROM sample_db.job_role
 WHERE id IN
     (SELECT id
         FROM sample_db.job_role
         WHERE salary > 120000);
```

ob_role 1

SELECT * FROM sample_db.job_role WHERE id IN (Enter a SQL expression t

	123 id	ABC job_category	ABC country	123 salary
1	7	Data Scientist	US	150,000
2	9	Data Scientist	UK	140,000
3	10	Data Scientist	UK	160,000
4	11	Data Analyst	UK	150,000
5	12	Data Scientist	US	200,000
6	13	Data Scientist	US	300,000
7	14	Developer	UK	151,000
8	19	ML Engineer	US	125,000

Here we have used subqueries with SELECT statement.

Subqueries with the INSERT Statement

Basic syntax:

```sql
-- basic syntax

INSERT INTO table_name [ (column1 [, column2 ]) ]
 SELECT [ *|column1 [, column2 ]
 FROM table1 [, table2 ]
 [ WHERE VALUE OPERATOR ]
```

* Consider a table job_role_copy with similar structure as job_role table. Now to copy complete job_role table into job_role_copy the query is as follows:

```
INSERT INTO sample_db.job_role_copy
     SELECT * FROM sample_db.job_role
     WHERE id IN (SELECT id
     FROM sample_db.job_role) ;
```

Statistics 1

INSERT INTO sample_db.job_role_copy SELECT * FF

Name	Value
Updated Rows	20
Query	INSERT INTO sample_db.job_role_copy
	SELECT * FROM sample_db.job_role
	WHERE id IN (SELECT id
	FROM sample_db.job_role)

Subqueries with the UPDATE Statement:

Basic Syntax:

```
    UPDATE table
SET column_name = new_value
[ WHERE OPERATOR [ VALUE ]
 (SELECT COLUMN_NAME
 FROM TABLE_NAME)
 [ WHERE) ]
```

Following example updates SALARY by 1.2 times in job_role table for all the employees whose job_category is data scientist:

```
UPDATE sample_db.job_role_copy SET salary = salary * 1.2
WHERE job_category IN (SELECT job_category FROM sample_db.job_role
WHERE job_category = 'Data Scientist' );

select * from sample_db.job_role_copy where job_category='Data Scientist';
```

job_role_copy 1

select * from sample_db.job_role_copy where job_ | Enter a SQL expression to filter results (use Ctrl+Spa

	123 id	ABC job_category	ABC country	123 salary
1	1	Data Scientist	India	96,000
2	7	Data Scientist	US	180,000
3	9	Data Scientist	UK	168,000
4	10	Data Scientist	UK	192,000
5	12	Data Scientist	US	240,000
6	13	Data Scientist	US	360,000

Subqueries with the DELETE Statement:

Basic Syntax:

```
-- basis syntax
DELETE FROM TABLE_NAME
[ WHERE OPERATOR [ VALUE ]
(SELECT COLUMN_NAME
FROM TABLE_NAME)
[ WHERE) ]
```

Following example deletes records from job_role_copy table for all the customers whose job_role is developer

```
DELETE FROM sample_db.job_role_copy
WHERE job_category  IN (SELECT job_category FROM sample_db.job_role_copy
WHERE job_category = 'Developer' );



-- some more nested queries
```

Statistics 1

DELETE FROM sample_db.job_role_copy WHERE jo | Enter a SQL expression to filter results (use Ctrl+Spa

Name	Value
Updated Rows	6
Query	DELETE FROM sample_db.job_role_copy
	WHERE job_category IN (SELECT job_category FROM sample_db.job_role_copy
	WHERE job_category = 'Developer')

We can clearly see from the below query all the records related to developer are deleted from the job_role_copy table:

```
select count(*) from sample_db.job_role_copy
 WHERE job_category = 'Developer';
```

ults 1

ct count(*) from sample_db.job_role_copy WH | Enter a SQL exp

123 count
0

Types of Nested Queries

There are mainly two types of nested queries:

Independent Nested Queries:

In independent nested queries, query execution starts from innermost query to outermost queries. The execution of inner query is independent of outer query, but the result of inner query is used in execution of outer query.

Various operators like IN, NOT IN, ANY, ALL etc are used in writing independent nested queries.

Suppose we have a table student as follows:

123 roll_no	ABC name	ABC address	ABC phone	123 age
1	HARSH	DELHI	XXXX	18
2	PRATIK	BIHAR	XXXX	19
3	VIKASH	SHIMLA	XXXX	20
4	DEEPA	KOLKATA	XXXX	18
5	DHEERAJ	BHOPAL	XXXX	19
6	BHANU	BIHAR	XXXX	20
7	ROHIT	UP	XXXX	18
8	VINAY	GURUGRAM	XXXX	19

And other table course as follows:

123 course_id	123 roll_no
1	1
2	2
2	3
3	4
1	5
4	9
5	10
4	11

If we have to find out roll_no who are enrolled in course_id 1 or 2.

We can write it with the help of independent nested query and IN operator as follows:

```
Select roll_no from course
where course_id IN
    (SELECT course_id from course
        where course_id in (1,2));
```

course 1

Select roll_no from course where course_id IN (SEL

	123 roll_no
1	1
2	2
3	3
4	5

Here, the inner query will return a set with course_id 1 and 2 and outer query will return those roll_no for which course_id is equal to any member of set (1 and 2 in this case).

Co-related Nested Queries: In co-related nested queries, the output of inner query depends on the row which is being currently executed in outer query.

For finding NAME of STUDENTs who are enrolled in course_id 1 we can write nested queries as follows:

```
Select name from student where roll_no in
 ( select roll_no from course where student.roll_no = course.roll_no and course.course_id= 1);
```

student 1

Select name from student where roll_no in (select | Enter a SQL expression to filter results (use Ctrl+Space)

	name
1	HARSH
2	DHEERAJ

Here we have two students HARSH and DHEERAJ who have enrolled in course having course_id = 1.

Similarly, we can write and practice subqueries in SQL.