Oracle Sub Query Overview & its Functionality

In this document, let’s learn about the Oracle sub query that helps us to construct more readable queries and allows us to write queries without using the complex joins or unions.

A Sub Query may occur in:

- A SELECT clause
- A FROM clause
- A WHERE clause

Sub query which is nested within the FROM clause of the SELECT statement is called an inline view.
Sub query nested in the WHERE clause of the SELECT statement is called a nested sub query.
Sub query where some clauses refer to column expressions in the outer query is correlated sub query.

Type of Sub Queries:

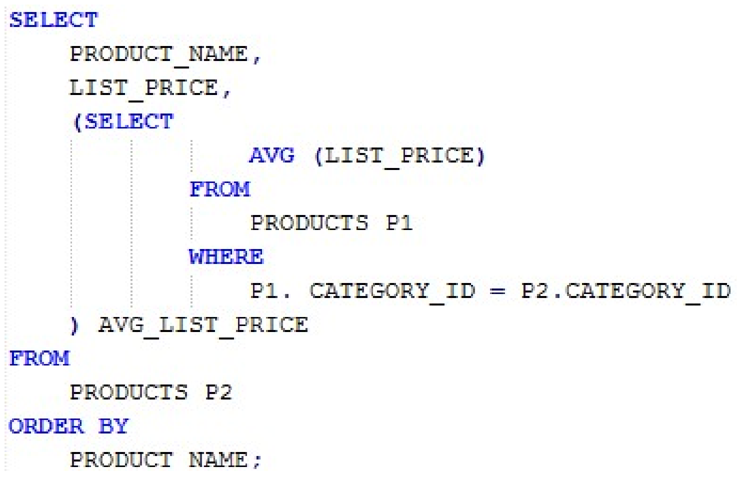
- Single Row: Returns zero or one row.
- Multiple row: Returns one or more rows.
- Multiple column: Returns one or more columns.
- Correlated Sub Query: A sub query because the query is related to the outer SQL statement.
- Nested Sub Queries: Are placed within another sub query.

Operators are being used:

- ANY and ALL keywords are used with a WHERE or HAVING clause.
- ANY and ALL operate on Sub Queries that return multiple values.
- ANY returns true if any of the Sub Query values meet the condition.
- ALL returns true if all of the Sub Query values meet the condition.
- WHERE Exists tests for the existence of any records in a Sub Query.
- EXISTS returns true if the Sub Query returns one or more records.
- EXISTS is commonly used with correlated Sub Queries.

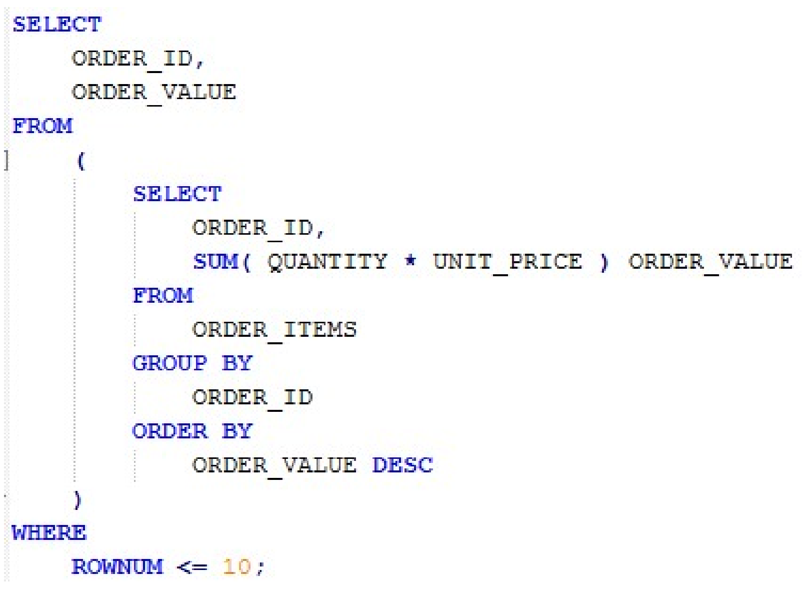
Few Examples as below:

A. **Sub query in SELECT Clause -** Statement returns the product name, list price, and the average list prices of products according to their categories:



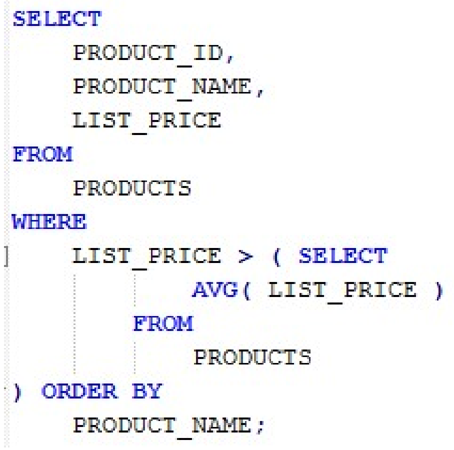
```
SELECT
    PRODUCT_NAME,
    LIST_PRICE,
    (SELECT
            AVG (LIST_PRICE)
        FROM
            PRODUCTS P1
        WHERE
            P1. CATEGORY_ID = P2.CATEGORY_ID
    ) AVG_LIST_PRICE
FROM
    PRODUCTS P2
ORDER BY
    PRODUCT_NAME;
```

B. **Sub query in the FROM Clause** - Statement returns the Top 10 orders with the highest values:



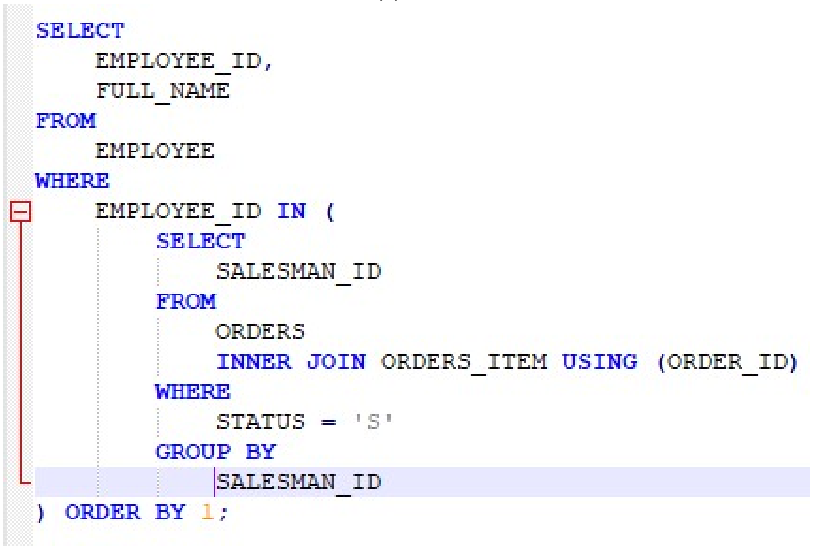
```
SELECT
    ORDER_ID,
    ORDER_VALUE
FROM
    (
        SELECT
            ORDER_ID,
            SUM( QUANTITY * UNIT_PRICE ) ORDER_VALUE
        FROM
            ORDER_ITEMS
        GROUP BY
            ORDER_ID
        ORDER BY
            ORDER_VALUE DESC
    )
WHERE
    ROWNUM <= 10;
```

C. **Subquery with Comparison Operators Example -** The following query finds products whose list price is greater than the average list price.



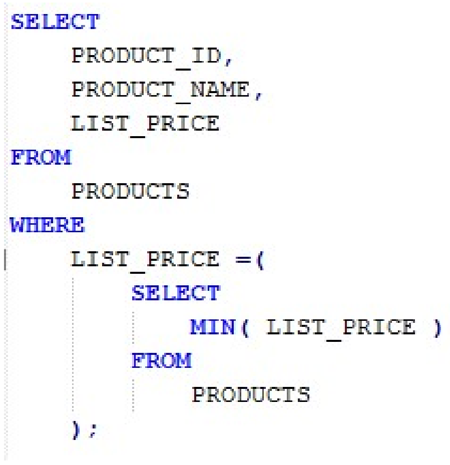
```
SELECT
    PRODUCT_ID,
    PRODUCT_NAME,
    LIST_PRICE
FROM
    PRODUCTS
WHERE
    LIST_PRICE > ( SELECT
            AVG( LIST_PRICE )
        FROM
            PRODUCTS
) ORDER BY
    PRODUCT_NAME;
```

D. **Oracle subquery with IN and NOT IN operators -** The following query finds the salesman who Order Status has been Shipped:

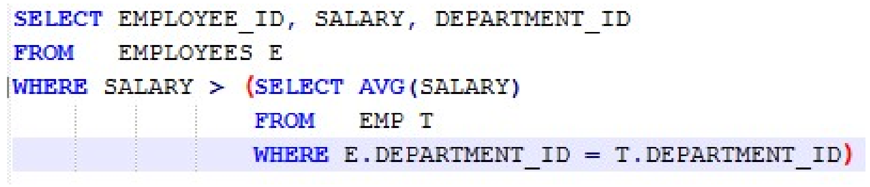


```
SELECT
    EMPLOYEE_ID,
    FULL_NAME
FROM
    EMPLOYEE
WHERE
    EMPLOYEE_ID IN (
        SELECT
            SALESMAN_ID
        FROM
            ORDERS
            INNER JOIN ORDERS_ITEM USING (ORDER_ID)
        WHERE
            STATUS = 'S'
        GROUP BY
            SALESMAN_ID
) ORDER BY 1;
```

E. **Oracle Correlated Subquery -** The following query returns the cheapest products from the products table using a subquery in the WHERE clause.



```
SELECT
    PRODUCT_ID,
    PRODUCT_NAME,
    LIST_PRICE
FROM
    PRODUCTS
WHERE
    LIST_PRICE =(
        SELECT
            MIN( LIST_PRICE )
        FROM
            PRODUCTS
    );
```



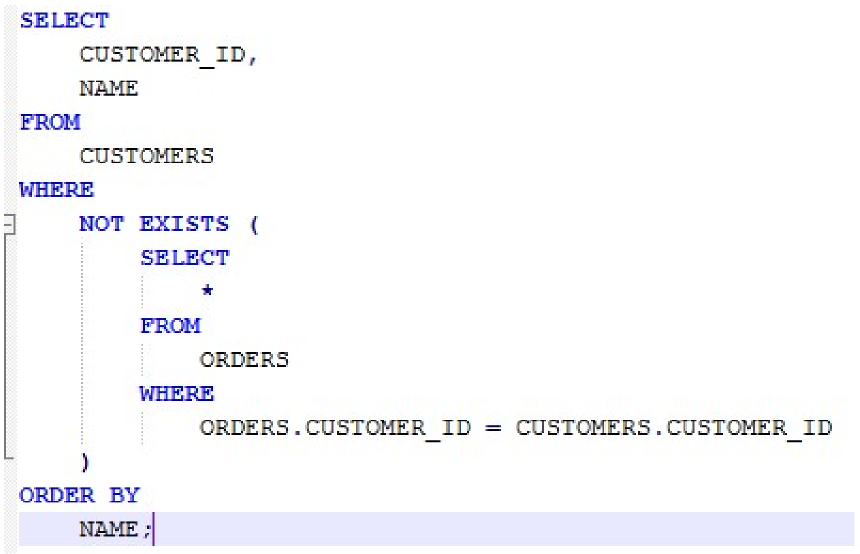
```
SELECT EMPLOYEE_ID, SALARY, DEPARTMENT_ID
FROM   EMPLOYEES E
WHERE SALARY > (SELECT AVG(SALARY)
                FROM   EMP T
                WHERE E.DEPARTMENT_ID = T.DEPARTMENT_ID)
```

First, we can execute the subquery independently.

Second, Oracle evaluates the subquery only once.

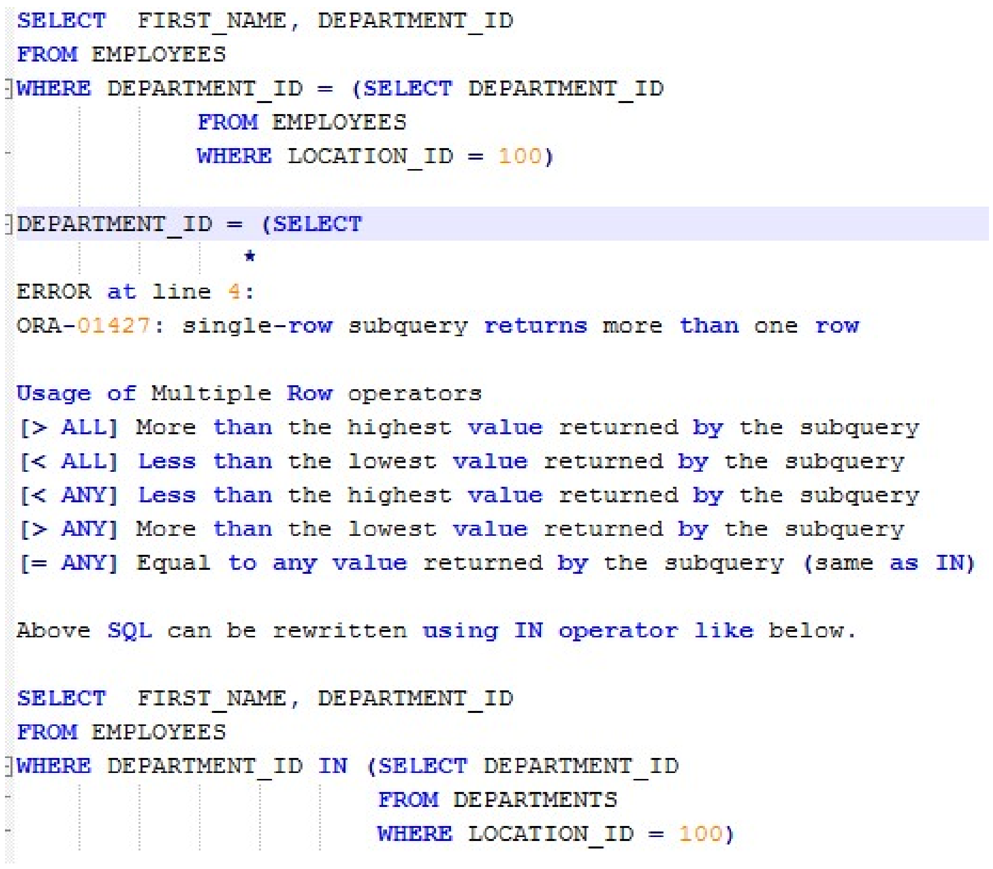
Third, after the subquery returns a result set, the outer query makes use of them. In other words, the outer query depends on the subquery. However, the subquery is isolated and not dependent on the values of the outer query.

F. **Oracle Correlated Sub query with the EXISTS operator example -** We usually use a correlated subquery with the EXISTS operator. For example, the following statement returns all customers who have no orders:



```
SELECT
    CUSTOMER_ID,
    NAME
FROM
    CUSTOMERS
WHERE
    NOT EXISTS (
        SELECT
            *
        FROM
            ORDERS
        WHERE
            ORDERS.CUSTOMER_ID = CUSTOMERS.CUSTOMER_ID
    )
ORDER BY
    NAME;
```

Another Example -



```
SELECT  FIRST_NAME, DEPARTMENT_ID
FROM EMPLOYEES
WHERE DEPARTMENT_ID = (SELECT DEPARTMENT_ID
            FROM EMPLOYEES
            WHERE LOCATION_ID = 100)

DEPARTMENT_ID = (SELECT
               *
ERROR at line 4:
ORA-01427: single-row subquery returns more than one row

Usage of Multiple Row operators
[> ALL] More than the highest value returned by the subquery
[< ALL] Less than the lowest value returned by the subquery
[< ANY] Less than the highest value returned by the subquery
[> ANY] More than the lowest value returned by the subquery
[= ANY] Equal to any value returned by the subquery (same as IN)

Above SQL can be rewritten using IN operator like below.

SELECT  FIRST_NAME, DEPARTMENT_ID
FROM EMPLOYEES
WHERE DEPARTMENT_ID IN (SELECT DEPARTMENT_ID
                        FROM DEPARTMENTS
                        WHERE LOCATION_ID = 100)
```