Multiple Row Subquery in Oracle

When a Subquery returns more than one value is called a Multiple Row Subquery in Oracle. In this Multiple Row Subquery, we can use the operators such as IN, ANY, ALL.

1. Employee details whose employee job is same as the job of the employee “SMITH”, and “CLARK”?

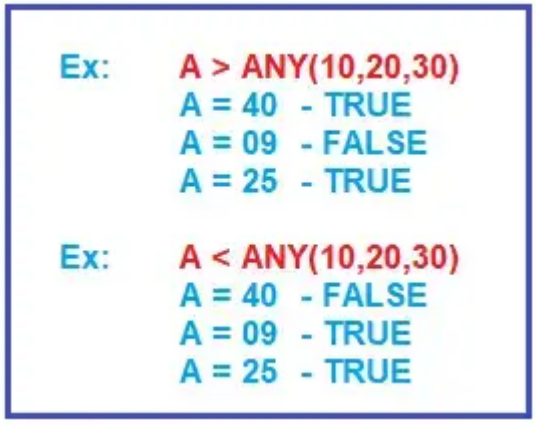
```
SELECT * FROM EMPLOYEE WHERE JOB IN(SELECT JOB FROM EMPLOYEE WHERE ENAME='SMITH' OR ENAME='CLARK');
```

2. Display employee details who are getting min, max salaries?

```
SELECT * FROM EMPLOYEE WHERE SAL IN(
SELECT MIN(SAL) FROM EMPLOYEE
UNION
SELECT MAX(SAL) FROM EMPLOYEE);
```

Working with “ANY” and “ALL” Operators in Oracle:

ANY: The ANY operator in Oracle is used to compare any of the values in the given list.



```
SELECT [column_name... | expression1 ]

FROM [table_name]

WHERE expression2 comparison_operator {ANY} (subquery)
```

Parameters:

1. **column_name**: Name of the column of the table.
2. **expression1**: Expression made up of a single constant, variable, scalar function, or
   column name and can also be the pieces of a SQL query that compare values against other values or perform arithmetic calculations.
3. **table_name**: Name of the table.
4. **WHERE expression2:** Compares a scalar expression until a match is found for ANY operator. One or
   more rows must match the expression to return a Boolean TRUE value for the ANY operator.
5. **comparison_operator:** Compares the expression to the subquery. The comparison must be a standard
   comparison operator (=, <>, !=, >, >=, <, or <=).

ALL: The ALL operator in Oracle is used to compare all of the values in the given list.

```
Ex:     A > ALL(10,20,30)
        A = 40  - TRUE
        A = 09  - FALSE
        A = 25  - FALSE

Ex:     A < ALL(10,20,30)
        A = 40  - FALSE
        A = 09  - TRUE
        A = 25  - FALSE
```

```
SELECT [column_name... | expression1 ]

FROM [table_name]

WHERE expression2 comparison_operator {ALL} (subquery)
```

Parameters:

1. **column_name**: Name of the column of the table.
2. **expression1**: Expression made up of a single constant, variable, scalar function, or
   column name and can also be the pieces of a SQL query that compare values against other values or perform arithmetic calculations.
3. **table_name**: Name of the table.
4. **WHERE expression2:** Compares a scalar expression, such as a column against every value in the
   subquery for ALL Operator. All rows must match the expression to return a Boolean TRUE value for the ALL operator.
5. **comparison_operator:** Compares the expression to the subquery. The comparison must be a standard
   comparison operator (=, <>, !=, >, >=, <, or <=).

3. Display all employees, whose salary is more than any “SALESMAN” salary?

```
SELECT * FROM EMPLOYEE WHERE SAL>ANY(SELECT SAL FROM EMPLOYEE WHERE JOB='SALESMAN');
```

5. fetch all the records from the PermanentEmployee table where the Age is at least greater than one value from the Age column of the ContractEmployee table.

```
SELECT * FROM PermanentEmployee WHERE Age > (SELECT MIN(Age) FROM ContractEmployee);
```

```
SELECT * FROM PermanentEmployee WHERE Age > ANY (SELECT Age FROM ContractEmployee);
```

6. Display all employees, whose salary is more than all “SALESMAN” salaries?

```
SELECT * FROM EMPLOYEE WHERE SAL>ALL(SELECT SAL FROM EMPLOYEE WHERE JOB='SALESMAN');
```

7. fetch all the records from the PermanentEmployee table where the Age is greater than the maximum value of the Age column in the ContractEmployee table.

```
SELECT * FROM PermanentEmployee WHERE Age > (SELECT MAX(AGE) FROM ContractEmployee);
```

```
SELECT * FROM PermanentEmployee WHERE Age > ALL (SELECT AGE FROM ContractEmployee);
```

What is SOME Operator in Oracle?

The SOME Operator in Oracle is used to compare a value to each value in a list of results from a query and evaluate to true if the result of an inner query contains at least one row. The SOME Operator must match at least one row in the subquery and must be preceded by comparison
operators. Suppose using greater than (>) with SOME Operator means greater than at least one value.

Parameters:

1. **column_name**: Name of the column of the table.
2. **expression1**: Expression made up of a single constant, variable, scalar function, or
   column name and can also be the pieces of a SQL query that compare values against other values or perform arithmetic calculations.
3. **table_name**: Name of the table.
4. **WHERE expression2:** Compares a scalar expression until a match is found for SOME operator. One or
   more rows must match the expression to return a Boolean TRUE value for the SOME operator.

5. **comparison_operator:** Compares the expression to the subquery. The comparison must be a standard
   comparison operator (=, <>, !=, >, >=, <, or <=).

fetch all the records from the PermanentEmployee table where the Age is at least greater than one value from the Age column of the ContractEmployee table.

```
SELECT * FROM PermanentEmployee WHERE Age > (SELECT MIN(Age) FROM ContractEmployee);
```

```
SELECT * FROM PermanentEmployee WHERE Age > SOME (SELECT Age FROM ContractEmployee);
```

What is EXISTS Operator in Oracle?

The EXISTS operator in Oracle is used in combination with a subquery and is considered to be met if the subquery returns at least one row. That means the EXISTS operator is used to check the existence of a result of a subquery. It can be used in a SELECT, INSERT, UPDATE, or DELETE
statement.

The EXISTS condition is very similar to IN condition. In the IN condition, we directly define constant values to evaluate with data rows, while in EXISTS condition we use a subquery. If the subquery returns TRUE the main query is evaluated. If the subquery returns FALSE the oracle server won't return the data rows.

```
SELECT * FROM Emp
WHERE EXISTS (SELECT * FROM Proj WHERE Emp.Id = Proj.EmployeeID);
```

```
SELECT * FROM Emp
WHERE NOT EXISTS (SELECT * FROM Proj WHERE Emp.Id = Proj.EmployeeID);
```

EXISTS Operator in Oracle with Update Statements

```
UPDATE Emp SET City = "Bangalore"
WHERE EXISTS (SELECT 1 FROM Proj WHERE ClientId = 3 AND Proj.EmployeeId = Employee.I
d);
```

EXISTS Operator in Oracle with Delete Statement:

```
DELETE FROM Emp WHERE EXISTS(
  SELECT 1 FROM Proj WHERE Proj.EmployeeId = Emp.Id AND Emp.city = 'Bangalore');
```

EXISTS Operator in Oracle with INSERT Statement:

`CREATE TABLE Employee_Archive AS SELECT * FROM Employee WHERE 1=0;`

```
INSERT INTO Employee_Archive
SELECT * FROM Emp
WHERE NOT EXISTS(
   SELECT 1 FROM Proj
   WHERE Proj.EmployeeId = Emp.Id
);

SELECT * FROM Employee_Archive;
```