SUBQUERY / NESTED QUERY:

> A QUERY INSIDE ANOTHER QUERY IS CALLED AS SUBQUERY OR NESTED QUERY.

> A SUBQUERY IS HAVING TWO MORE QUERIES THOSE ARE,

I) INNER / CHILD / SUB QUERY

II) OUTER / PARENT / MAIN QUERY

SYNTAX:

SELECT * FROM <TN> WHERE <CONDITION> (SELECT * FROM .....);

> AS PER THE EXECUTION PROCESS OF SUBQUERY IT AGAIN CLASSIFIED INTO TWO CATEGORISED.

1) NON - CORELETED SUBQUERIES:

- IN NON-CO RELATED SUBQUERY FIRST INNER QUERY IS EXECUTED AND RETURN A VALUE BASED ON RETURN VALUE OF INNER QUERY LATER OUTER QUERY WILL EXECUTE AND PRODUCE THE FINAL RESULT.

2) CORELATED SUBQUERIES:

- IN CO RELATED SUBQUERY FIRST OUTER QUERY IS EXECUTED AND RETURN VALUES BASED ON RETURN VALUES OF OUTER QUERY LATER INNER QUERY WILL EXECUTE AND PRODUCE THE FINAL RESULT.

TYPES OF NON - CORELETED SUBQUERIES:

> SINGLE ROW SUBQUERY

> MULTIPLE ROW SUBQUERY

> MULTIPLE COLUMN SUBQUERY

> INLINE VIEW

SINGLE ROW SUBQUERY:

WHEN A SQ RETURNS A SINGLE VALUE IS CALLED AS SRSQ.

IN SRSQ WE CAN USE OPERATORS ARE =, <, >, <=, >=, !=.

EX1:

WHO ARE GETTING THE FIRST HIGH SALARY FROM EMP TABLE?

SOL:

==========

|| SUBQUERY = OUTER + INNER ||

==========

STEP1:(INNER QUERY):

SELECT MAX(SAL) FROM EMP;

STEP2:(OUTER QUERY):

SELECT * FROM EMP WHERE SAL= (INNER QUERY);

STEP3: (SUBQUERY= OUTER+INNER):

SELECT * FROM EMP WHERE SAL= (SELECT MAX(SAL) FROM EMP);

EX2:

WHOSE EMPLOYEE JOB IS SAME AS THE JOB OF 'SMITH'?

SOL:

SQL> SELECT * FROM EMP WHERE JOB= (SELECT JOB FROM EMP WHERE ENAME='SMITH');

EX3:

WHOSE SALARY IS MORE THAN MAX.SALARY OF THE JOB IS "SALESMAN"?

SOL:

SQL> SELECT * FROM EMP WHERE SAL> (SELECT MAX(SAL) FROM EMP WHERE JOB='SALESMAN');

EX4:

WHOSE EMPLOYEE JOB IS SAME AS THE JOB OF "BLAKE" AND WHO ARE EARNING

SALARY MORE THAN "BLAKE" SALARY?

SOL:

SELECT * FROM EMP WHERE JOB= (SELECT JOB FROM EMP WHERE ENAME='BLAKE')

AND SAL> (SELECT SAL FROM EMP WHERE ENAME='BLAKE');

EX5:

DISPLAY SENIOR EMPLOYEE?

SOL:

```
SQL> SELECT * FROM EMP WHERE HIREDATE=(SELECT MIN(HIREDATE) FROM
EMP);
```

EX6:

TO FIND SECOND HIGH. SALARY FROM EMP TABLE ?

SOL:

```
SQL> SELECT MAX(SAL) FROM EMP WHERE SAL<(SELECT MAX(SAL) FROM
EMP);

  MAX(SAL)
----------
      3000
```

EX7:

WASQ DISPLAY EMPLOYEE DETAILS WHO ARE GETTING

SECOND HIGH. SALARY IN EMP TABLE?

SOL:

```
SQL> SELECT * FROM EMP WHERE SAL= (SELECT MAX(SAL) FROM EMP WHERE
SAL< (SELECT MAX(SAL) FROM EMP));
```

EX8:

DISPLAY EMPLOYEE DETAILS WHO ARE GETTING

3RD HIGH. SALARY IN EMP TABLE?

SOL:

```
SQL> SELECT * FROM EMP WHERE SAL=
        (SELECT MAX(SAL) FROM EMP WHERE SAL<
        (SELECT MAX(SAL) FROM EMP WHERE SAL<
        (SELECT MAX(SAL) FROM EMP)));
```

EX9:

TO DISPLAY NO. OF EMPLOYEE OF DEPARTMENT NUMBERS.IN WHICH DEPTNO NO. OF EMPLOYEE IS LESS THAN THE NO. OF EMPLOYEE OF DEPTNO IS 20?

SOL:

```
SQL> SELECT DEPTNO,COUNT(*) FROM EMP GROUP BY DEPTNO
      HAVING COUNT(*)<(SELECT COUNT(*) FROM EMP WHERE DEPTNO=20);
```

EX10:

SUM OF SALARY OF JOBS.IF SUM OF SALARY OF JOBS ARE

MORE THAN SUM OF SALARY OF THE JOB IS 'CLERK'?

SOL:

```
SQL> SELECT JOB,SUM(SAL) FROM EMP GROUP BY JOB
       HAVING SUM(SAL)>(SELECT SUM(SAL) FROM EMP WHERE JOB='CLERK');
```

SUBQUERY WITH "UPDATE":

EX11:

TO UPDATE EMPLOYEE SALARY WITH MAX.SALARY OF

EMP TABLE WHOSE EMPNO IS 7900?

SOL:

```
SQL> UPDATE EMP SET SAL=(SELECT MAX(SAL) FROM EMP) WHERE
EMPNO=7900;
```

SUBQUERY WITH "DELETE":

EX12:

WASQ TO DELETE EMPLOYEE DETAILS FROM EMP TABLE

WHOSE JOB IS SAME AS THE JOB OF 'SCOTT'?

SOL:

```
SQL> DELETE FROM EMP WHERE JOB=(SELECT JOB FROM EMP WHERE
ENAME='SCOTT');
```

MULTIPLE ROW SUBQUERY:

WHEN A SQ RETURNS MORE THAN ONE VALUE IS CALLED AS

MRSQ. IN THIS SUB QUERY WE CAN USE THE FOLLOWING OPERATORS

ARE IN, ANY, ALL.

EX1:

TO DISPLAY EMPLOYEE DETAILS WHOSE EMPLOYEE JOB IS SAME AS

THE JOB OF THE EMPLOYEE "SMITH","CLARK”?

SOL:

SQL> SELECT * FROM EMP WHERE JOB IN(SELECT JOB FROM EMP WHERE ENAME='SMITH' OR ENAME='CLARK');

EX2:

DISPLAY EMPLOYEE DETAILS WHO ARE GETTING MIN, MAX SALARIES?

SOL:

SELECT * FROM EMP WHERE SAL IN(

SELECT MIN(SAL) FROM EMP

UNION

SELECT MAX(SAL) FROM EMP);

EX3:

TO DISPLAY THE SENIOR MOST EMPLOYEES FROM EACH DEPTNO WISE?

SOL:

SQL> SELECT * FROM EMP WHERE HIREDATE IN(SELECT MIN(HIREDATE) FROM EMP GROUP BY DEPTNO);

EX4:

TO DISPLAY EMPLOYEES WHO ARE EARNING HIGHEST SALARY FROM EACH JOB WISE?

SQL> SELECT * FROM EMP WHERE SAL IN(SELECT MAX(SAL) FROM EMP GROUP BY JOB);

<u>WORKING WITH "ANY”, "ALL" OPERATORS:</u>

<u>ANY</u>: IS USED TO SATISFIED ANY OF THE VALUES IN THE GIVEN

LIST WITH USER DEFINED CONDITION.

EX: A > ANY (10,20,30)

A = 40 - TRUE

A = 09 - FALSE

A = 25 - TRUE

A < ANY (10,20,30)

A = 40 - FALSE

A = 09 - TRUE

A = 25 - TRUE

<u>ALL:</u> IS USED TO SATISFIED ALL OF THE VALUES IN THE GIVEN LIST WITH USER DEFINED CONDITION.

EX: A > ALL (10,20,30)

A = 40 - TRUE

A = 09 - FALSE

A = 25 - FALSE

A < ALL (10,20,30)

A = 40 - FALSE

A = 09 - TRUE

A = 25 - FALSE

EX:

WASQ TO DISPLAY EMPLOYEES WHOSE SALARY IS MORE THAN ANY "SALESMAN"SALARY?

SOL:

SQL> SELECT * FROM EMP WHERE SAL>ANY (SELECT SAL FROM EMP WHERE JOB='SALESMAN');

EX:

WASQ TO DISPLAY EMPLOYEES WHOSE SALARY IS MORE THAN OF ALL "SALESMAN"SALARY?

SOL:

SQL> SELECT * FROM EMP WHERE SAL>ALL (SELECT SAL FROM EMP WHERE JOB='SALESMAN');

EX:

TO DISPLAY EMPLOYEES WHO ARE EARNING HIGHEST SALARY FROM EACH JOB WISE BY USING MULTIPLE ROW SUBQUERY?

```
SQL> UPDATE EMP SET SAL=1300 WHERE EMPNO=7902;

SQL> SELECT * FROM EMP WHERE SAL IN (SELECT MAX(SAL) FROM EMP
GROUP BY JOB);
```

OUTPUT:

JOB	SAL
---------	----------
ANALYST	1300
CLERK	1300
SALESMAN	1600
MANAGER	2975
ANALYST	3000
PRESIDENT	5000

NOTE: IN THE ABOVE EXAMPLE WHEN WE ARE COMPARING GROUP OF VALUES BY USING MULTIPLE ROW SUBQUERY THEN ORACLE RETURNS WRONG RESULT.TO OVERCOME THIS PROBLEM THEN WE USE MULTIPLE COLUMN SUBQUERY.

MULTIPLE COLUMN SUBQUERY:

MULTIPLE COLUMNS VALUES OF INNER QUERY COMPARING WITH MULTIPLE COLUMNS VALUES OF OUTER QUERY IS CALLED AS MCSQ.

SYNTAX:

```
SELECT * FROM <TN> WHERE(<COL1>,<COL2>,.....)
IN (SELECT <COL1>,<COL2>,..... FROM <TN>);
```

EX:

```
SQL> SELECT * FROM EMP WHERE (JOB, SAL) IN(SELECT JOB,MAX(SAL) FROM
EMP GROUP BY JOB);
```

OUTPUT:

JOB	SAL
---------	----------
CLERK	1300
SALESMAN	1600
MANAGER	2975
ANALYST	3000
PRESIDENT	5000

EX:

WAQ TO DISPLAY EMPLOYEE WHOSE JOB, MGR ARE SAME AS THE JOB, MGR OF THE EMPLOYEE "SCOTT”?

SOL:

```
SQL> SELECT * FROM EMP WHERE (JOB, MGR) IN (SELECT JOB,MGR FROM EMP
WHERE ENAME='SCOTT');
```

PSEUDO COLUMNS:

PSEUDO COLUMN IS JUST LIKE A TABLE COLUMN.

I) ROWID

II) ROWNUM

ROWID:

WHEN WE INSERT A NEW ROW INTO A TABLE INTERNALLY SYSTE IS CREATES A UNIQUE ID ADDRESS /NUMBER FOR EACH ROW WISE AUTOMATICALLY.

THESE ROWID’S ARE STORED IN DATABASE SO THAT THESE ARE PERMANENT ID’S.

EX:

```
SQL> SELECT ROWID, ENAME FROM EMP;

ROWID                  ENAME
------------------     ----------
AAAPLMAAEAAAAADAAA     SMITH
```

```
SQL> SELECT ROWID, ENAME, DEPTNO FROM EMP WHERE DEPTNO=10;

ROWID                  ENAME       DEPTNO
------------------     ----------  ----------
AAAPLMAAEAAAAADAAN MILLER          10
```

EX:

```
SQL> SELECT MIN(ROWID) FROM EMP;

MIN(ROWID)
------------------
AAAPLMAAEAAAAADAAA
```

SQL> SELECT MAX(ROWID) FROM EMP;

MAX(ROWID)

AAAPLMAAEAAAAADAAN

HOW TO DELETE MULTIPLE DUPLICATE ROWS EXCEPT ONE DUPLICATE ROW FROM A TABLE:

WHENEVER WE WANT TO DELETE MULTIPLE DUPLICATE ROWS EXCEPT ONE DUPLICATE ROW FROM A TABLE THEN WE USE "ROWID" PSEUDO COLUMN.

EX:

SQL> SELECT * FROM TEST;

SNO	NAME
10	A
10	A
10	A
20	B
20	B
30	C
30	C
30	C
40	D
40	D
50	E
50	E
50	E

SOL:

SQL> DELETE FROM TEST WHERE ROWID NOT IN (SELECT MAX(ROWID) FROM TEST GROUP BY SNO);

OUTPUT:

SQL> SELECT * FROM TEST;

SNO	NAME
10	A
20	B
30	C
40	D
50	E

ROWNUM:

TO GENERATE ROW NUMBERS TO EACH ROW WISE / GROUP OF ROWS WISE AUTOMATICALLY.THESE ROWNUMBERS ARE NOT SAVED IN DB.SO THAT THESE ARE TEMPORARY NUMBERS.

EX:

SQL> SELECT ROWNUM, ENAME FROM EMP;

ROWNUM	ENAME
1	SMITH
2	ALLEN

SQL> SELECT ROWNUM, ENAME, DEPTNO FROM EMP WHERE DEPTNO=10;

ROWNUM	ENAME	DEPTNO
1	MILLER	10
2	CLARK	10
3	KING	10

EX:

WAQ TO FETCH THE FIRST ROW EMPLOYEE DETAILS FROM EMP

TABLE BY USING ROWNUM?

SOL:

SQL> SELECT * FROM EMP WHERE ROWNUM=1;

EX:

WAQ TO FETCH THE SECOND ROW EMPLOYEE DETAILS FROM EMP

TABLE BY USING ROWNUM?

SOL:

SQL> SELECT * FROM EMP WHERE ROWNUM=2;

NO ROWS SELECTED

NOTE: GENERALLY, ROWNUM IS ALWAYS START WITH 1 FROM EVERYSELECTED ROW IN A TABLE.SO TO AVOID THIS PROBLEM THEN WE USE <, <= OPERATORS.

EX:

SELECT * FROM EMP WHERE ROWNUM<=2

MINUS

SELECT * FROM EMP WHERE ROWNUM=1;

EX:

WAQ TO FETCH THE FIRST FIVE ROWS FROM EMP TABLE BY USING ROWNUM?

SOL:

SQL> SELECT * FROM EMP WHERE ROWNUM<=5;

EX:

WAQ TO FETCH THE FIFTH ROW EMPLOYEE DETAILS FROM EMP TABLE BY

USING ROWNUM?

SOL:

SELECT * FROM EMP WHERE ROWNUM<=5

MINUS

SELECT * FROM EMP WHERE ROWNUM<=4;

EX:

WAQ TO FETCH FROM 3RD TO 9TH ROW FROM EMP TABLE BY USING ROWNUM?

SOL:

```
SELECT * FROM EMP WHERE ROWNUM<=9
MINUS
SELECT * FROM EMP WHERE ROWNUM<3;
```

EX:

WAQ TO FETCH THE LAST TWO ROWS FROM EMP TABLE BY ROWNUM?

SOL:

```
SELECT * FROM EMP WHERE ROWNUM<=14
MINUS
SELECT * FROM EMP WHERE ROWNUM<=12;
```

(OR)

```
SELECT * FROM EMP
MINUS
SELECT * FROM EMP WHERE ROWNUM<=(SELECT COUNT(*)-2 FROM EMP);
```

INLINE VIEW:

IT IS SPECIAL TYPE OF SUBQUERY.PROVIDING A SELECT QUERY IN PLACE OF TABLE NAME IN SELECT STATEMENT.

IN INLINE VIEW SUBQUERY,THE RESULT OF INNER QUERY WILL ACT AS A TABLE FOR THE OUTER QUERY.

SYNTAX:

```
SELECT * FROM <TABLE NAME>; --------SQL SELECT QUERY
SELECT * FROM (<SELECT QUERY>); -----INLINE VIEW
```

NOTE:

1. GENERALLY SUBQUERY IS NOT ALLOWED TO USE "ORDER BY" CLAUSE.SO THAT WE USE "INLINE VIEW".

2. GENERALLY COLUMN ALIAS NAMES ARE NOT ALLOWED TO USE IN "WHERE" CLAUSE CONDITION.SO THAT WE USE "INLINE VIEW ".

USING COLUMN ALIAS NAMES IN WHERE CLAUSE CONDITION:

EX:

WAQ TO DISPLAY EMPLOYEE WHOSE EMPLOYEE ANNUAL SALARY IS MORE THAN 25000?

SOL:

```
SQL> SELECT * FROM (SELECT ENAME,SAL,SAL*12 ANNUSAL FROM EMP)
WHERE ANNUSAL>25000;
```

USING "ORDER BY" CLAUSE IN SUBQUERY:

EX:

WAQ TO DISPLAY FIRST FIVE HIGHEST SALARIES OF EMPLOYEE FROM EMP

TABLE BY USING ROWNUM ALONG WITH INLINE VIEW?

SOL:

```
SQL> SELECT * FROM (SELECT * FROM EMP ORDER BY SAL DESC) WHERE
ROWNUM<=5;
```

EX:

WAQ TO DISPLAY 5TH HIGHEST SALARY OF EMPLOYEE FROM EMP

TABLE BY USING ROWNUM ALONG WITH INLINE VIEW?

SOL:

```
SELECT * FROM (SELECT * FROM EMP ORDER BY SAL DESC) WHERE
ROWNUM<=5
MINUS
SELECT * FROM (SELECT * FROM EMP ORDER BY SAL DESC) WHERE
ROWNUM<=4;
```

USING ROWNUM ALIAS NAME:

EX:

WAQ TO DISPLAY 3RD POSITION ROW FROM EMP TABLE BY USING ROWNUM ALIAS NAME ALONG WITH INLNE VIEW?

SOL:

```
SQL> SELECT * FROM (SELECT ROWNUM R,ENAME,JOB,SAL FROM EMP) WHERE
R=3;
```

(OR)

```
SQL> SELECT * FROM (SELECT ROWNUM R,EMP.* FROM EMP) WHERE R=3;
```

EX:

WAQ TO DISPLAY 1ST,3RD,5TH,7TH,9TH ROWS FROM EMP TABLE BY USING ROWNUM ALIAS NAME ALONG WITH INLINE VIEW?

SOL:

SQL> SELECT * FROM (SELECT ROWNUM R,EMP.* FROM EMP) WHERE R IN(1,3,5,7,9);

EX:

WAQ TO DISPLAY EVEN POSITION ROWS FROM EMP TABLE BY USING ROWNUM ALIAS NAME ALONG WITH INLINE VIEW?

SOL:

SQL> SELECT * FROM (SELECT ROWNUM R,EMP.* FROM EMP) WHERE MOD(R,2)=0;

EX:

WAQ TO DISPLAY FIRST ROW AND LAST ROW FROM EMP TABLE BY USING ROWNUM ALIAS NAME ALONG WITH INLINE VIEW?

SQL> SELECT * FROM (SELECT ROWNUM R,EMP.* FROM EMP) WHERE R=1 OR R=(SELECT COUNT(*) FROM EMP);

ANALYTICAL FUNCTIONS:

- ORACLE SUPPORTING THE FOLLOWING THREE TYPES OF ANALYTICAL FUNCTIONS THOSE ARE,

1. **ROW_NUMBER ()**
2. **RANK ()**
3. **DENSE_RANK ()**

THESE ANALYTICAL FUNCTIONS ARE AUTOMATICALLY GENERATE RANKING NUMBERS TO EACH ROW WISE (OR) GROUP OF ROWS WISE EXCEPT ROW_NUMBER (). THIS FUNCTION WILL GENERATE ROW NUMBERS FOR EACH ROW WISE / GROUP OF ROWS WISE.

EX:

ENAME	SALARY	ROW_NUMBER ()	RANK ()	DENSE_RANK ()
A	85000	1	1	1
B	72000	2	2	2
C	72000	3	2	2
D	68000	4	4	3
E	55000	5	5	4

RANK (), DENSE_RANK () WILL ASSIGN SAME RANK NUMBER TO SAME VALUE BUT RANK () WILL SKIP THE NEXT RANK NUMBER IN THE ORDER WHERE AS DENSE_RANK () WILL NOT SKIP THE NEXT RANK NUMBER IN THE ORDER.

SYNTAX:

ANALYTICAL FUNCTION NAME () OVER () [PARTITION BY <COLUMN NAME>] ORDER BY <COLUMN NAME> [ASC / DESC])

HERE,

PARTITION BY CLAUSE IS OPTIONAL.

ORDER BY CLAUSE IS MANDATORY.

WITHOUT PARTITION BY CLAUSE:

EX:

```
SQL> SELECT ENAME,SAL,ROW_NUMBER()OVER(ORDER BY SAL DESC) RANKS
FROM EMP;

SQL> SELECT ENAME,SAL,RANK()OVER(ORDER BY SAL DESC) RANKS FROM
EMP;

SQL> SELECT ENAME,SAL,DENSE_RANK()OVER(ORDER BY SAL DESC) RANKS
FROM EMP;
```

WITH PARTITION BY CLAUSE:

EX:

```
SELECT ENAME,SAL,DEPTNO,DENSE_RANK()OVER(PARTITION BY DEPTNO
ORDER BY SAL DESC) RANKS FROM EMP;
```

OUTPUT:

ENAME	SAL	DEPTNO	RANKS
KING	5000	10	1
CLARK	2450	10	2
MILLER	1300	10	3

EX1:

WAQ TO DISPLAY 3RD HIGHEST SALARY EMPLOYEE DETAILS FROM EMP TABLE IN EACH DEPTNO WISE BY USING DENSE_RANK () ALONG WITH INLINE VIEW?

SOL:

SELECT * FROM (SELECT ENAME,SAL,DEPTNO,DENSE_RANK()OVER(PARTITION BY DEPTNO ORDER BY SAL DESC) R FROM EMP) WHERE R=3;

EX2:

WAQ TO DISPLAY THE 4TH SENIOR MOST EMPLOYEE FROM EACH JOB WISE?

SOL:

SELECT * FROM (SELECT ENAME, JOB, HIREDATE, DENSE_RANK () OVER (PARTITION BY JOB ORDER BY HIREDATE) R FROM EMP) WHERE R=4;

2. CO-RELATED SUBQUERY:

- IN CO-RELATED SUBQUERY FIRST OUTER QUERY IS EXECUTED AND LATER INNER QUERY WILL EXECUTE.

SYNTAX TO FIND "NTH" HIGH / LOW SALARY:

SELECT * FROM <TN> <TABLE ALIAS NAME1> WHERE N-1= (SELECT COUNT (DISTINCT <COLUMN NAME>) FROM <TN> <TABLE ALIAS NAME2> WHERE <TABLE ALIAS NAME2>.<COLUMN NAME> (< / >) <TABLE ALIAS NAME1>.<COLUMN NAME>);

EX1: TO FIND OUT EMPLOYEE WHO ARE GETTING FIRST HIGHEST SALARY FROM EMPLOYEE TABE?

SOL:

SELECT * FROM EMPLOYEE E1 WHERE 0= (SELECT COUNT (DISTINCT SAL) FROM EMPLOYEE E2 WHERE E2.SAL>E1.SAL);

EX2: TO FIND OUT EMPLOYEE WHO ARE GETTING FOURTH HIGHEST SALARY FROM EMPLOYEE TABLE?

SOL:

SQL> SELECT * FROM EMPLOYEE E1 WHERE=(SELECT COUNT(DISTINCT SAL) FROM EMPLOYEE E2 WHERE E2.SAL>E1.SAL);

SYNTAX TO DISPLAY "TOP N" HIGH / LOW SALARIES:

SELECT * FROM <TN> <TABLE ALIAS NAME1> WHERE N>(SELECT COUNT(DISTINCT <COLUMN NAME>) FROM <TN> <TABLE ALIAS NAME2> WHERE <TABLE ALIAS NAME2>.<COLUMN NAME> (< / >) <TABLE ALIAS NAME1>.<COLUMN NAME>);

EX1:

WAQ TO DISPLAY TOP 3 HIGHEST SALARIES EMPLOYEE DETAILS FROM EMPLOYEE TABLE?

SOL:

```
SQL> SELECT * FROM EMPLOYEE E1 WHERE 3>(SELECT COUNT(DISTINCT SAL)
FROM EMPLOYEE E2 WHERE E2.SAL>E1.SAL);
```

NOTE:

1. TO FIND OUT "NTH" HIGH / LOW SALARY ----------> N-1

2. TO DISPLAY "TOP N" HIGH / LOW SALARIES -----> N>

EXISTS OPERATOR:

- IT A SPECIAL OPERATOR WHICH IS USED IN CO-RELATED SUBQUERY ONLY.THIS OPERATOR IS USED TO CHECK WETHER ROW / ROWS EXISTS IN THE TABLE OR NOT.

- IT RETRUNS EITHER TRUE (OR) FALSE.IF SUBQUERY RETRUNS AT LEAST ONE ROW THEN RETRUNS TRUE OR ELSE IF SUBQUERY NOT RETRUNS ANY ROW THEN RETURN FALSE.

SYNTAX:

```
WHERE EXISTS (<SELECT STATEMENT>)
```

EX1:

WAQ TO DISPLAY DEPARTMENT DETAILS IN WHICH DEPARTMENT EMPLOYEE ARE WORKING?

SOL:

```
SQL> SELECT * FROM DEPT D WHERE EXISTS (SELECT DEPTNO FROM EMP E
WHERE E. DEPTNO=D.DEPTNO);
```

EX2:

WAQ TO DISPLAY DEPARTMENT DETAILS IN WHICH DEPARTMENT EMPLOYEE ARE NOT WORKING?

SOL:

```
SQL> SELECT * FROM DEPT D WHERE NOT EXISTS (SELECT DEPTNO FROM EMP
E WHERE E. DEPTNO=D.DEPTNO);
```

SCALAR SUBQUERY:

- SUBQUERIES IN SELECT CLAUSE IS CALLED AS SCALAR SUBQUERY.EVERY SUBQUERY OUTPUT WILL ACT AS A COLUMN.

SYNTAX:

```
----------
SELECT (SUBQUERY1), (SUBQUERY2), .....FROM <TABLE NAME> [ WHERE
<COND>];
```

EX:

```
SQL> SELECT (SELECT COUNT (*) FROM EMP) AS EMPTOTAL, (SELECT COUNT
(*) FROM DEPT) AS DEPTTOTAL FROM DUAL;

 EMPTOTAL        DEPTTOTAL
----------------  ----------
      14              4
```

EX:

```
SQL> SELECT (SELECT SUM(SAL) FROM EMP WHERE DEPTNO=10) AS "10",
          (SELECT SUM(SAL) FROM EMP WHERE DEPTNO=20) AS "20",
          (SELECT SUM(SAL) FROM EMP WHERE DEPTNO=30) AS "30"
          FROM DUAL;

   10              20          30
------------      ----------  ----------
  8750              10875       9400
```