**Correlated Sub Queries**

**Correlated sub queries in where clause**

To see employees who earn salary less than the average salary **of their own job.**

Without the correlated sub query technique also it was possible using the set operator Union All by explicitly combining select statements.

The script could have been like this –

SELECT \* FROM EMP

WHERE SAL < (SELECT AVG(SAL) FROM EMP WHERE JOB = 'CLERK')

AND JOB = 'CLERK'

**UNION ALL**

SELECT \* FROM EMP

WHERE SAL < (SELECT AVG(SAL) FROM EMP WHERE JOB = 'MANAGER')

AND JOB = 'MANAGER'

**UNION ALL**

SELECT \* FROM EMP

WHERE SAL < (SELECT AVG(SAL) FROM EMP WHERE JOB = 'ANALYST')

AND JOB = 'ANALYST'

**UNION ALL**

SELECT \* FROM EMP

WHERE SAL < (SELECT AVG(SAL) FROM EMP WHERE JOB = 'SALESMAN')

AND JOB = 'SALESMAN'

But this technique is not effective when the jobs types are more or if job types are not known.

So the co related sub query technique is used to make the query more compact.

**Parent query takes each row and submits it to child query.**

**Child query gets executed for each row**

select \* from emp **E**

where sal < **(select avg(sal)**

**from emp**

**where job = E.job);**

The parent query table and the child query table is same so the table alias (E) is must to identify the row transferred by the parent query to the child query.

**If the tables were different then alias was not mandatory.**

For Example –

create table empcopy

as

select \* from emp;

The emp table is going to check the average of salaries from the EmpCopy table. In this case the correlated sub query does not require a alias.

**select \* from Emp**

**where sal** **<** (**select avg(sal)**

**from EmpCopy**

**where job** = **Emp.job**);

**Generally, when correlated sub query is used with "Select" statement then it can be "slow" in processing if volume of rows is high. So, it is less used with "Select" statement.**

**With "Select" Inline Views or Derived Tables is the "BEST" technique.**

**But, Correlated Sub Query is "BEST" when it gets used with "Update" and / or "Delete" statements. Actually, it is widely used in "Projects"!!!!**

**Corelated sub queries in update statement**

alter table emp

add dname varchar(10);

Select \* from Emp; **-- Right now Dnames are NULL**

update emp

set dname = (select dname

from dept

where emp.deptno = dept.deptno);

Select \* from Emp; **-- Now Dnames have got correct values!!!**

Alter Table Emp

Drop Column Dname;

**Exists and Not Exists Operators can be used with:**

**1) Simple Sub Query**

**2) Correlated Sub Query (Widely used in Projects!!)**

**Exists Operator With Corelated Sub Queries:**

-- To display ename and sal from emp table who have their deptno matching as per dept table.

**Change Scott’s deptno**

Update Emp

set deptno = 49

where ename = 'SCOTT';

**select ename, sal**

**from emp**

**where** **EXISTS** (select null

from dept

where **emp.deptno** = dept.deptno); -- Will not show Scott’s record

select ename, sal

from **emp**

where **NOT EXISTS** (select null

from dept

where **emp.deptno** = dept.deptno); -- Will show **Only** Scott’s record

Revert Scott’s Deptno to

**When we want to show records of only one table (columns of only one table) based on join condition of other table then following 3 ways can be used:**

**1) Using normal Inner Joins (Slower)**

select ename, sal

from emp join dept

on emp.deptno = dept.deptno;

**--------------------------------------------------**

**2) Using Simple Sub Query, i.e., Multi Row Sub Query (Better)**

select ename, sal

from emp

where deptno in (select deptno from dept);

**--------------------------------------------------**

**3) Using Correlated Sub Query using Exists (Super-Fast)**

select ename, sal

from emp

where EXISTS (select null

from dept

where emp.deptno = dept.deptno);