**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 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 –

Select \* into **EmpCopy** 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**);

**\*\*\*\*Co related sub queries in update statement**

alter table emp

add dname varchar(10);

update **Emp**

set dname = (select dname

from dept

where **Emp.deptno** = dept.deptno);

Alter Table Emp

Drop Column Dname; -- In SQL Server

**Exists with Correlated Sub Queries in Select statements:**

When you want to see all columns of only one table based on join condition of other table then instead of going with traditional join technique the correlated sub query along with Exists operator can be used.

To see records of emp1 who have matching deptno as per dept1.

**select** \* **from** emp1 e

**where** **Exists** (**select** **null**

**from** dept1 d

**where** e.deptno = d.deptno);

To see records in emp1 which are not matching as per deptno in dept1 table.

**select** \* **from** emp1 e

**where** **Not** **Exists** (**select** **null**

**from** dept1 d

**where** e.deptno =d.deptno);

**Exists with Correlated Sub Queries in DML statements:**

**drop** **table** All\_emp**;**

**drop** **table** Retired\_emp**;**

**Select \* into**  All\_Emp **from emp;**

**Select \* into**  Retired\_Emp **from emp where 1 = 2;**

**Insert** **into** Retired\_Emp

**select** \* **from** All\_Emp

**where** ename = 'JONES'**;**

**Insert** **into** Retired\_Emp

**select** \* **from** All\_Emp

**where** ename = 'MARTIN'**;**

**select** \* **from** All\_Emp**;**

**select** \* **from** Retired\_Emp**;**

Now we want to delete records from All\_Emp for matching deptno from Retired\_Emp.

**Delete** **from** All\_Emp

**where** Exists (**select** **null**

**from** Retired\_Emp R

**where** All\_Emp.empno = R.empno)**;**

**select** \* **from** All\_Emp**;**