**DISPLAYING DATA FROM MULTIPLE TABLE**

**JOINS**:

When data from more than one table in the database is required, a join condition is used. Rows in one table can be joined to rows in another table according to common values existing in corresponding columns, that is, usually primary and foreign key columns.

To display data from two or more related tables, write a simple join condition in the WHERE clause.

• A Join is Query that combines rows from two or more tables, Views, or Materialized Views.

• A Join is performed whenever multiple table appear in the Queries FROM clause.

Sql>Select Empno,Ename,Dname,Loc From Emp,Dept ;

Sql>Select Empno,Ename,Sal,Grade From Emp,Salgrade;

Sql>Select Empno,Ename,Dname,Loc,Grade From Emp,Dept,Salgrade

Sql> Select Empno,Ename,Emp.Deptno,Dname,Loc From Emp,Dept

**JOIN Condition**:

* The Applied condition is called a JOIN CONDITION.
* To Execute a join…..
* Oracle combines pairs of rows, each containing one row from each table, for which the JOIN condition evaluates to TRUE.
* The column in the join condition need not be part of the SELECT list.
* The WHERE clause of join Query can also contain other conditions that refer to columns of only one table.
* If a join involves over two tables then oracle joins the first two based on the condition and then compares the result with the next table and so on.

Table 1 🡪 Join 01 🡨Table 2

↓

Result 01 🡪 Join 02 🡪 Table 3

↓

Result 02

* The oracle optimizer determines the order in which ORACLE should join the tables based on …..
* Given JOIN condition(s)
* INDEXES upon the tables.
* The LOB columns cannot be specified in the WHERE clause, when the WHERE clause contains any JOINS.

**Syntax:**

WHERE table1.column1 = table2.column2;

**Guidelines**:

* When writing a SELECT statement that joins tables, precede the column name with the table name for clarity and to enhance database access.
* If the same column name appears in more than one table, the column name must be prefixed with the table name.
* To join n tables together, you need a minimum of n-1 join conditions.
* For example, to join four tables, a minimum of three joins is required.

**Equi Join or Simple Joins or Inner Joins**:

* Based on equality condition tables are joined
* Only matching records are displayed
* Joining tables must have at least one common column with same data type and same values, (not column name same.)

**Qualifying Ambiguous column names**:

* The name of the column’s names should be qualified in the WHERE clause with the table name to avoid ambiguity.
* If there are no common column name between the two tables the qualification is not necessary but it is better.

Syntax:

>Select col1,col2,col3....

From <table1>, <table2>……….

where <table1>. <common col name>=<table2>. <common col name>

And……...

(This is joined condition)

Sql>Select

Emp.Empno Empno,

Emp .Ename Ename,

Emp.Deptno Deptno,

Dept.Deptno Deptno,

Dept.Dname Dname,

Dept.Loc Loc

From Emp,Dept

Where Emp.Deptno=Dept.deptno;

Sql> Select Empno ,Ename,Emp.Deptno Deptno,Dname

From Emp,Dept

Where Emp.Deptno = Dept.deptno and

Empno=&eno;

Sql> Select Empno ,Ename,Emp.Deptno Deptno,Dname

From Emp,Dept

Where Emp.Deptno = Dept.deptno and

Job=UPPER('clerk');

Sql> Select Empno ,Ename,Sal\*12 AnnualSal,Emp.Deptno,Loc

From Emp,Dept

Where Emp.Deptno = Dept.deptno

And Dname='SALES';

**Using Table Aliases**:

Qualifying column names with table names can be very time consuming, particularly if table names are lengthy. You can use table aliases instead of table names. Just as a column alias gives a column another name, a table alias gives a table another name. Table aliases help to keep SQL code smaller, therefore using less memory. The table name is specified in full, followed by a space and then table alias. The EMP table has been given an alias of 'e', and the DEPT table has an alias of 'd'.

**Guidelines**:

* Table aliases can be up to 30 characters in length, but the shorter they are the better.
* If a table alias is used for a particular table name in the FROM clause, then that table alias must be substituted for the table-throughout the SELECT statement.
* Table aliases should be meaningful.
* The table alias is valid only for the current SELECT statement.

Sql> Select E.Empno ,E.Ename,D.Deptno,D.Dname

From Emp E,Dept D

Where E.Deptno = D.Deptno;

Sql> Select E.Empno,E.Ename,E.Job,D.Deptno,D.Dname

From Emp E,Dept D

Where E.Deptno = D.Deptno and

E.Job IN('MANAGER','SALESMAN');

Sql> Select E.Empno ,E.Ename,E.Job,D.Deptno,D.Dname

From Emp E,Dept D

Where E.Deptno = D.Deptno AND

D.Dname<>'SALES';

**Inner Join**:

Sql>Select e.Empno,e.Ename,e.Sal,e.Deptno,

d.Dname,d.Loc

From Emp e INNER JOIN Dept d

ON(e.Deptno=d.Deptno);

Sql>Select e.Ename "Employee Name",m.Ename "Manager"

From Emp e INNER JOIN Emp m

ON(e.mgr=m.empno);

**Non Equi Join**:

* A non equi-Join specifies the relationship between columns belonging to different tables by making use of the relational operations other than =.
* It is used to join table if value of one column of a table falls in the range of two column values of another table.
* It used between operator, it called as Between join.

Syntax:- Select col1,col2,………..

From<table A>,<table B> Where <tableA>.<col1>Between

<table B>.<Col1>and <table B>.<col2>;

Sql> Select E.empno,E.Ename,E.Sal,S.Grade,S.Losal,S.Hisal

From Emp E, Salgrade S

Where E.Sal Between S.Losal and S.Hisal;

Sql>Select E.empno,E.Ename,E.Sal,S.Grade,S.Losal,S.Hisal

From Emp E,Salgrade S

Where E.Empno=&Eno

And E.Sal Between S.Losal and S.Hisal;

Sql>Select E.empno,E.Ename,E.Sal,S.Grade

From Emp E,Salgrade S

Where (E.Sal>=S.Losal And E.Sal<=S.Hisal) And S.Grade=1;

**Self Joins**:

* It indicates joining the table to itself.
* The same table appears twice in the FROM clause and is followed by table aliases.
* The table aliases must qualify the column names in the join condition.
* It is used on same table the table must have at least 2 column with same (Data type, Value).
* To perform a self-join, oracle combines and returns rows of the table that satisfy the join condition.

Syntax:

Sql>Select Columns

From Table1 T1, Table1 T2

Where Tl.Column1=T2.Column2;

Example:

Sql>Select E.Ename "Emp Name",M.Ename "Manager Name"

From Emp E,Emp M

Where E.Mgr=M.Empno;

Sqf> Select E.Ename "Emp Name",M.Ename "Manager"

From Emp E,Emp M

Where E.Mgr=M.Empno

And E.Hiredate<M.Hiredate;

Sql> Select E.Ename| |'Emp Name'||M.Ename| | 'Manager Name'

From Emp E,Emp M

Where E.Mgr=M.Empno;

Sql> Select E.Ename ||'Works for'||

M.Ename "Employees and Managers”

From Emp E,Emp M

Where (E.Mgr=M.Empno);

Sql> Select E.Ename ||'''s Works for'||

M.Ename "Employees and Managers"

From Emp E,Emp M

Where (E.Mgr=M.Empno) And

E.Job='CLERK'

**Joining Data from More than Two Table**:

* JOINS can be established on more than two tables
* The Join is First executed upon the two most relevant tables and then the result is applied upon the Table.

Sql> Select Ename,e.Sal,Grade,D.Deptno,Dname

From Emp E JOIN Dept D

ON E.Deptno=D.Deptno

JOIN Salgrade

ON E.Sal BETWEEN Losal And Hisal;

Sql>Select

E.Ename Ename,E.Deptno,M.Ename Manager,M.deptno

From Emp E,Dept D,Emp M

Where E.Mgr=M.Empno And

E.deptno=D.Deptno;

Select E.Ename,M.Ename,Sal,Grade,D.Deptno,Dname

From Emp E INNER JOIN Dept D

ON E.Deptno=D.Deptno

INNER JOIN Emp M

ON E.MGR=M.Empno

INNER JOIN Salgrade S

ON E.Sal BETWEEN Losal And Hisal;

Sql> Select

E.Ename Ename,

M.Ename Manager,

D.Dname Dname,

E.Sal Esal,

SE.Grade Egrade,

M.Sal Msal,

SM.Grade Mgrade

From Emp E,Dept D,Emp M,salgrade SE,Salgrade SM

Where

(E.Deptno=D.Deptno) And

(E.Mgr=M.Empno) And

(E.Sal BETWEEN SE.Losal And SE.Hisal) And

(M.Sal BETWEEN SM.Losal And SM.Hisal);

**Cartesian Products**:

* A Cartesian product results in all combinations of rows displayed. This is done by either omitting the WHERE clause or specifying the CROSS-JOIN clause.
* During Cartesian product oracle combines each row of one table with each row of the other.
* It tends to generate a large number of rows and the result is rarely useful.
* A Cartesian product is formed when:
* A join condition is omitted.
* A join condition is invalid.

Sql>Select Ename,Job,Dname

From Emp,Dept;

Sql>Select Ename,Job,Dname

From Emp,Dept

Where Job='MANAGER';

**Outer Joins**:

Used to retrieve all the rows from one table but matching rows from other table.

An Outer join extends the result of a simple or inner join.

An OUTER join returns all rows that satisfy the join condition and also those rows from one table for which no rows from the other TABLE satisfy the join condition.

It is use an operator ( + ),it called join operator.

(+) used with the table which missing the data.

To perform an outer join of table 'A' and 'B' and returns all rows from 'A', apply the outer join operator '(+)' to all columns of table'B'.

For all rows in 'A' that have no matching rows in 'B' oracle returns null for any select list expressions containing columns of 'B'.

Sql>Select

Table1.Column,Table2.Column

From Tablel,Table2

Where Table1.Column(+)=Table2.Column;

Sql>Select

Table l.Column,Table2. Column

From Tablel,Table2

Where Table1.Column=Table2.Column(+);

**Rules And Restrictions**:

* The (+) operator can appear only in the WHERE clause.
* If 'A' and 'B' are joined by multiple join conditions, we must use the (+) operator in all of these conditions.
* (+) used only with one table.
* A condition cannot use the IN comparison operator compare a column marked with the( + ) operator with, expression.
* A condition cannot compare any column marked with the (+) operator with a sub Query.

Sql> Select E.Ename,D.Deptno,D.Dname

From Emp E,Dept D

Where E.deptno(+) = D.deptno

ORDER BY E.Deptno;

Sql> Select E.Ename,D.Deptno,D.Dname

From Emp E,Dept D

Where E.deptno=D.Deptno(+) and

E.deptno=10

ORDER BY E.Deptno;

Sql> Select E.Ename "Employe",

NVL(M.Ename,' Supreme Authority') "Manager"

From Emp E,Emp M

Where E.MGR=M.Empno( + );

**Cross join**:

• CROSS JOIN Returns a Cartesian product from the two tables.

Sql>SelectEname,Dept.Deptno,Dname,Loc

From Emp CROSS JOIN Dept

Where Emp.Deptno=Dept.Deptno;

**Natural Join**:

Sql>Select Empno,Ename,Sal,Deptno,Dname,loc

From Emp Natural Join Dept;

Note:-Natural not accept an alias names.

**Creating Joins with the USING Clause**:

* Clause can be modified with the USING clause to specify the columns that should be used for an equi join.
* Use the USING clause to match only one column when more than one column matches.
* Do not use a table name or alias in the referenced columns.
* The NATURAL JOIN and USING clauses are mutually exclusive.

Sql> SELECT e.ename, d.dname

FROM Emp e JOIN dept d USING (deptno)

WHERE deptno =10;

**Left Outer Join**:

Sql> select a.empno,a.ename,a.sal,a.deptno,

b.dname,b.loc

from emp a left outer join dept b

on(a.deptno=b.deptno)

**Rigth Ourter Join**:

Sql>select a.empno,a.ename,a.sal,b.deptno,

b.dname,b.loc

from emp a right outer join dept b

on(a.deptno=b.deptno);

**Full outer Join**:

Sql>select a.empno,a.ename,a.sal,b.deptno,

b.dname,b.loc

from emp a full outer join dept b

on(a.deptno=b.deptno);

**Note**:-

Full outer join=left outer join+Right outer join.

**Some Special Example**:

Sql>Select e.Ename "Employee Name",m.Ename "Manager"

From Emp e LEFT OUTER JOIN Emp m

ON(e.mgr=m.empno)

Order by 2;