JOIN

Merging of two or more tables horizontally is known as Joins
Q. Why do we need Join's?
➔ To retrieve the data from multiple tables we use join's.
When we have to retrieve the data from 2 tables then we perform join.

Note: - from clause is responsible to merge the table

Types of Join's

1. Cartesian join or cross join
2. Inner join or equi join
3. Outer join
    i. left outer join or left join
    ii. right outer join or right join
    iii. full outer join or full join
4. self join

1. **Cartesian join or cross joins: -**

If we join 2 tables, records from one table is merged with each and every records present in the other table is known as Cartesian join or cross join.

Ex. Let us consider 2 tables T1 and T2 with columns each and m, n as no. of rows respectively

T1

A1	**B1**
A	10
B	20
C	30

T2

A2	**B2**
B	200
C	300
D	400

If we perform a Cartesian join on table T1 and T2 the newly obtain table will have 4 columns and m X n no. of rows.

Note: -

1. Cartesian join has valid as well as invalid pairs.
2. Cartesian join as a universal set as it is having all the possible combinations.

T1 X T2 only when A2 = B

A1	B1	A2	B2
A	10	B	200
A	10	C	300
A	10	D	400
B	20	B	200
B	20	C	300
B	20	D	400
C	30	B	200
C	30	C	300
C	30	D	400

m X n = 3 X 3 = 9

Syntax for Cartesian join: -

1. ANSI syntax : -
   Select */column/expression
   From table1 cross join table2;

Eg. Select *

From T1 cross join T2;

2. Oracle syntax:-
   Select */column/expression
   From table1, table2,…..;

Eg. Select *

From T1, T2, T3,….;

For ex, let us consider the following query
Display employee name along with the department name

```
SQL> select A.ename, A.sal, B.dname
  2  from emp A, dept B ;
```

ENAME	SAL	DNAME
SMITH	800	ACCOUNTING
ALLEN	1600	ACCOUNTING
WARD	1250	ACCOUNTING
JONES	2975	ACCOUNTING
MARTIN	1250	ACCOUNTING
BLAKE	2850	ACCOUNTING
CLARK	2450	ACCOUNTING
SCOTT	3000	ACCOUNTING
KING	5000	ACCOUNTING
TURNER	1500	ACCOUNTING
ADAMS	1100	ACCOUNTING
JAMES	950	ACCOUNTING
FORD	3000	ACCOUNTING
MILLER	1300	ACCOUNTING
SMITH	800	RESEARCH
ALLEN	1600	RESEARCH
WARD	1250	RESEARCH
JONES	2975	RESEARCH
MARTIN	1250	RESEARCH
BLAKE	2850	RESEARCH
CLARK	2450	RESEARCH
SCOTT	3000	RESEARCH
KING	5000	RESEARCH
TURNER	1500	RESEARCH
ADAMS	1100	RESEARCH
JAMES	950	RESEARCH
FORD	3000	RESEARCH
MILLER	1300	RESEARCH
SMITH	800	SALES
ALLEN	1600	SALES
WARD	1250	SALES
JONES	2975	SALES
MARTIN	1250	SALES
BLAKE	2850	SALES

```
ENAME             SAL DNAME
---------- ---------- --------------
CLARK             2450 SALES
SCOTT             3000 SALES
KING              5000 SALES
TURNER            1500 SALES
ADAMS             1100 SALES
JAMES              950 SALES
FORD              3000 SALES
MILLER            1300 SALES
SMITH              800 OPERATIONS
ALLEN             1600 OPERATIONS
WARD              1250 OPERATIONS
JONES             2975 OPERATIONS
MARTIN            1250 OPERATIONS
BLAKE             2850 OPERATIONS
CLARK             2450 OPERATIONS
SCOTT             3000 OPERATIONS
KING              5000 OPERATIONS

SCOTT             3000 RESEARCH
KING              5000 RESEARCH
TURNER            1500 RESEARCH
ADAMS             1100 RESEARCH
JAMES              950 RESEARCH
FORD              3000 RESEARCH
MILLER            1300 RESEARCH
SMITH              800 SALES
ALLEN             1600 SALES
WARD              1250 SALES
JONES             2975 SALES
MARTIN            1250 SALES
BLAKE             2850 SALES

ENAME             SAL DNAME
---------- ---------- --------------
TURNER            1500 OPERATIONS
ADAMS             1100 OPERATIONS
JAMES              950 OPERATIONS
FORD              3000 OPERATIONS
MILLER            1300 OPERATIONS

56 rows selected.
```

From above – we can see that the above query returns 56 records – but we are expecting 14 records. This is because each and every record of employee table will be combined with each & every record of department table.

Thus, Cartesian join should not be used in real time scenarios.

The Cartesian join contains both correct and incorrect sets of data. We have to retain the correct ones & eliminate the incorrect ones by using the **inner join**.

3. **Inner join: -**

Inner join are also called as **equijoins**.
They return the matching records between the tables.
In the real time scenarios, this is the most frequently used Join.
We join two tables such that a record from one table is merged to a record from another table only when given condition is satisfied is known as inner join.

For ex, consider the query shown below,

```
Select A.ename, A.sal, B.dname
From emp A, dept B
Where A.deptno = B.deptno          - JOIN condition
And A.sal > 2000                   - FILTER condition
Order by A.sal ;
```

Let us see the output shown below,

```
SQL> Select A.ename, A.sal, B.dname
  2  From emp A, dept B
  3  Where A.deptno = B.deptno
  4  And A.sal > 2000
  5  Order by A.sal ;

ENAME             SAL DNAME
---------- ---------- --------------
CLARK            2450 ACCOUNTING
BLAKE            2850 SALES
JONES            2975 RESEARCH
FORD             3000 RESEARCH
SCOTT            3000 RESEARCH
KING             5000 ACCOUNTING

6 rows selected.
```

JOIN condition is mandatory for removing the Cartesian output.

Let us consider the following 2 scenarios shown below,

Scenario 1

A		
P	**Q**	**R**

B		
P	**S**	**T**

C		
P	**X**	**Y**

We want			
P	**Q**	**S**	**X**

The SQL query will be,

Select A.P, A.Q, B.S, C.X
From A, B, C
Where A.P = B.P
And A.P = C.P
} **Number of joins = 2**

Therefore, Number of JOINS = Number of tables - 1

Scenario 2

A		
P	**Q**	**R**

B			
P	**Q**	**S**	**T**

C		
P	**X**	**Y**

We want				
P	**Q**	**R**	**S**	**X**

The **SQL query is ,**

Select A.P, A.Q, A.R, B.S, C.X
From A, B, C
Where A.P = B.P
And A.Q = B.Q
And A.P = C.P ;

Number of Joins = 3

Therefore, Number of JOINS = Number of common columns

If there are no common columns, then reject it saying that the two tables can't be joined.

But there are some cases – where the 2 columns will be same but having different column names.
For ex – customerid & cid

ANSI Syntax:

```
Select *
From table1 inner join table 2
ON <join condition>
Where <filter condition>
```

Thus we, can see the changes ,

- In the 2nd line - ,(comma) has been replaced by the word 'join'
- In the 3rd line – 'where' has been replaced with 'on'

Note :

1. To perform inner join, join condition is mandatory
2. Join condition: - it is a condition which includes column from both the tables
3. Inner join is a sub set of Cartesian join or cross join
   Ex. Let us consider the table T1 And T2 we join T1 and T2 using the join condition
   T1.A1 = T2.A2
   The table obtained is as follows

A1	B1	A2	B2
B	20	B	200
C	30	C	300

Q. WAQTD Dept name, salary, comm of all the employees who are working in accounts or research dept. as a manager

ANSI
Select dname, sal, comm
From emp inner join dept
ON emp.deptno = dept.deptno
Where dname IN ('account','research') and job ='manager';

Select dname, sal, comm.
From emp, dept
Where emp.deptno=dept.deptno
And dname in ('account','research') and job ='manager';

Q. WAQTD dept name, ename, sal of all the employee whose name starts with A whose dept name ends with S and having the salary between 3000 and 5000

Dept Name, Ename, SAL
Condition
Ename starts with A and
Dname ends with S and
sal between 3000 and 5000

select detname, ename, sal
from emp, dept
where emp.deptno=dept.deptno and

Assignment

1) Display employee name and his department name for the employees whose name starts with 'S'

```
SQL> select A.ename, B.dname
  2  from emp A, dept B
  3  where A.deptno = B.deptno
  4  and A.ename not like 'S%' ;

ENAME      DNAME
---------- --------------
ALLEN      SALES
WARD       SALES
JONES      RESEARCH
MARTIN     SALES
BLAKE      SALES
CLARK      ACCOUNTING
KING       ACCOUNTING
TURNER     SALES
ADAMS      RESEARCH
JAMES      SALES
FORD       RESEARCH
MILLER     ACCOUNTING

12 rows selected.
```

Outer Join: -

It returns both matching and non-matching records

Outer join = inner join + non-matching records

Non-matching records means data present in one table, but absent in another table w.r.to common columns.

For ex, 40 is there in deptno of dept table, but not there in deptno of emp table.

Emp

Ename	Dept no.
1	10
2	20
3	10
4	10
5	

Dept

Dname	Dept No.
D1	10
D2	20
D3	30

Dname	Dept NO.	ename	Dept no.
D1	10	1	10
D1	10	3	10
D1	10	4	10
D2	20	2	20
D3	30		

Ename	Dept NO.	Dname	Dept no.
1	10	D1	10
2	20	D2	20
3	10	D1	10
4	10	D1	10
5			

Left outer join: - left outer join is used to obtain the unmatched of left table

Select *

From table1 left outer join table 2

On <join condition>

Where <filter conditioin>;

Note:

1. To get only unmatched records from the left table we should write a condition that is R_table_name.column_name IS null;
2. To get obly unmatched records from right table we should write the condition that is L_table_name.columnname IS null;

Q. WAQTD name of an employee who is not working in any department

```
Select ename
From emp left outer join dept
ON emp.deptno=dept.deptno
Where dept.deptno = null
```

2. Right outer join: - it is used to obtain the unmatched records go the right table

Display all the department names irrespective of any employee working in it or not. If an employee is working – display his name.

Using right join

```
SQL> select A.ename, A.job, B.dname, B.loc
  2  from emp A right join dept B
  3  on A.deptno = B.deptno ;

ENAME      JOB       DNAME          LOC
---------- --------- -------------- -------------
CLARK      MANAGER   ACCOUNTING     NEW YORK
KING       PRESIDENT ACCOUNTING     NEW YORK
MILLER     CLERK     ACCOUNTING     NEW YORK
JONES      MANAGER   RESEARCH       DALLAS
FORD       ANALYST   RESEARCH       DALLAS
ADAMS      CLERK     RESEARCH       DALLAS
SMITH      CLERK     RESEARCH       DALLAS
SCOTT      ANALYST   RESEARCH       DALLAS
WARD       SALESMAN  SALES          CHICAGO
TURNER     SALESMAN  SALES          CHICAGO
ALLEN      SALESMAN  SALES          CHICAGO
JAMES      CLERK     SALES          CHICAGO
BLAKE      MANAGER   SALES          CHICAGO
MARTIN     SALESMAN  SALES          CHICAGO
                     OPERATIONS     BOSTON

15 rows selected.
```

Using left join

```
SQL> select A.ename, A.job, B.dname, B.loc
  2  from dept B left join emp A
  3  on A.deptno = B.deptno ;
```

Using full join

```
SQL> select A.ename, A.job, B.dname, B.loc
  2  from dept B full join emp A
  3  on A.deptno = B.deptno ;
```

Assignment

1) Display employee name and his department name for the employees whose name starts with 'S'

```
SQL> select A.ename, B.deptno
  2  from emp A, dept B
  3  where A.deptno = B.deptno
  4  and A.ename like 'S%' ;

ENAME          DEPTNO
---------- ----------
SMITH              20
SCOTT              20
```

2) Display employee name and his department name who is earning 1st maximum salary

```
SQL> select A.ename, B.dname
  2  from emp A, dept B
  3  where A.deptno = B.deptno
  4  and A.sal = (select max(sal) from emp) ;

ENAME      DNAME
---------- --------------
KING       ACCOUNTING
```

SELF JOIN

Self join used to obtain the data to be selected in the same record or row
Joining a table to itself is called self join

The **FROM** clause looks like this,

FROM emp A, emp B

Or

FROM emp A join emp B - *ANSI style*

For ex, - **Display employee name along with their manager name**

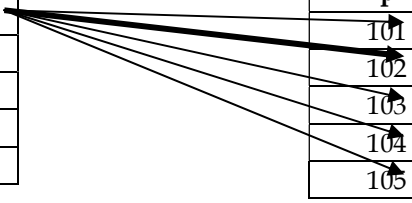
```
SQL> select A.ename "EMP",
  2         B.ename "MANAGER"
  3  from emp A, emp B
  4  where A.mgr = B.empno ;

EMP        MANAGER
---------- ----------
SMITH      FORD
ALLEN      BLAKE
WARD       BLAKE
JONES      KING
MARTIN     BLAKE
BLAKE      KING
CLARK      KING
SCOTT      JONES
TURNER     BLAKE
ADAMS      SCOTT
JAMES      BLAKE
FORD       JONES
MILLER     CLARK

13 rows selected.
```

Now, let us see how this i.e the logic (the above query) works,

Emp (A)		
EmpNo	**Ename**	**Mgr**
101	Scott	102
102	Blake	103
103	King	-
104	Smith	103
105	Jones	104



Emp (B)		
EmpNo	**Ename**	**Mgr**
101	Scott	102
102	Blake	103
103	King	-
104	Smith	103
105	Jones	104

Now, when we give the above query – in Oracle – it starts matching the '**mgr**' column of **emp A** with the '**empno**' of **emp b** – we get two tables because in **self join** – a duplicate of the table required is created.

Now let us consider the **first employee Scott** – it starts the **mgrid** of **Scott** with the **empno** of all the records in **emp B** – when two **ids** match, then the **empno** in **emp B** becomes the **mgr** of the **empno** in **emp A**. Thus, we can see that – **mgr id** 102 is matching with **empno** 102 **Blake** in **emp B**. Therefore, Blake is the manager of Scott.
Similarly we do the same for all the other records of **emp A** and thus find the employees and their respective managers.

Display the employees who are getting the same salary

Select a.ename, a.sal
From emp a, emp b
Where a.sal=b.sal

```
SQL> select A.ename, A.sal
  2  from emp A join emp B
  3  on A.sal = B.sal
  4  and A.empno <> B.empno ;

ENAME             SAL
---------- ----------
MARTIN            1250
WARD              1250
FORD              3000
SCOTT             3000
```