JOINS:

JOINS ARE USED TO RETRIEVE DATA FROM MULTIPLE TABLES AT A TIME. IN RELATIONAL DATABASES WE ARE STORING RELATED DATA IN MULTIPLE TABLESLIKE EMPLOYEE DETAILS , DEPARTMENT DETAILS ,CUSTOMER DETILS,ORDERS DETAILS,PRODUCTS DETAILS,……. etc.TO COMBINED DATA AND RETRIEVE DATA FROM THOSE MULTIPLE TABLES THEN WE NEED JOINS.

TYPES OF JOINS:

- INNER JOIN / EQUI JOIN
- OUTER JOIN
    - LEFT OUTER JOIN
    - RIGHT OUTER JOIN
    - FULL OUTER JOIN
- CROSS JOIN (OR) CARTISEAN JOIN
- NON EQUI JOIN
- SELF JOIN
- NATURAL JOIN

SYNTAX FOR JOINS:

```
SELECT * FROM <TABLE NAME1> <JOIN KEY> <TABLE NAME2 > ON <JOIN CONDITION>;
```

JOINS TABLES:

EX:

STUDENT

STID	SNAME	CID
-------	----------	----------
1021	SAI	10
1022	ADAMS	20
1023	JONES	30

COURSE

```
CID    CNAME      CFEE
------ ---------- ----------
 10    ORACLE     2500
 20    JAVA       6000
 40    PHP        4500
```

INNER JOIN / EQUI JOIN: RETRIEVING DATA FROM MULTIPLE TABLES BASED ON "EQUAL OPERATOR (=) " IS CALLED AS INNER JOIN / EQUI JOIN.WHEN WE USE INNER JOIN BETWEEN TWO (OR) MORE THAN TWO TABLES THERE MUST BE A COMMON COLUMN (OR) COMMON FIELD NAME IS NO NEED TO BE SAME NAME BUT RECOMMEND.COMMON COLUMN (OR) COMMON FIELD DATATYPE MUST BE MATCH.

WHEN WE PERFORM ANY JOIN OPERATION BETWEEN TABLES THERE IS NO NEED TO HAVE RELATIONSHIP(OPTIONAL).(i.e PRIMARY KEY & FOREIGN KEY RELATION)INNER JOIN / EQUI JOIN ALWAYS RETRIEVING ONLY MATCHING DATA / MATCHNG ROWS.

SYNTAX:

ON <TABLE NAME1>.<COMMON COLUMN> = <TABLE NAME2>.<COMMON COLUMN>;

(OR)

ON <TN1 ALIAS NAME>.<COMMON COLUMN> = <TN2 ALIAS NAME>.<COMMON COL>;

EX1:

WAQ TO RETRIEVE STUDENT AND THE CORRESPONDING COURSE DETAILS FROM

STUDENT,COURSE TABLES BY USING INNER JON / EQUI JOIN ?

SOL:

```
SQL> SELECT * FROM STUDENT,COURSE WHERE CID=CID;

ERROR at line 1:

ORA-00918: column ambiguously defined
```

NOTE: IN ABOVE EXAMPLE WE GET AN ERROR IS "column ambiguously defined".

TO OVER COME THIS ERROR THEN WE SHOULD USE A TABLE NAME AS AN IDENTITY

TO AMBIGUOUSE COLUMN CID LIKE BELOW,

SOL:

SQL> SELECT * FROM STUDENT INNER JOIN COURSE ON STUDENT.CID=COURSE.CID;

(OR)

SQL> SELECT * FROM STUDENT S INNER JOIN COURSE C ON S.CID=C.CID;

RULE OF JOIN: **A ROW IN A FIRST TABLE IS COMPARING THE GIVEN JOIN CONDITION WITH ALL ROWS OF SECOND TABLE.**

EX2:WAQ TO RETRIEVE STUDENT,COURSE DETAILS FROM TABLES IF CID IS 20 ?

SOL:

SQL> SELECT * FROM STUDENT S INNER JOIN COURSE C ON S.CID=C.CID AND C.CID=20;

EX3:

WAQ TO RETRIEVE LIST OF EMPLOYEE FROM EMP,DEPT TABLES BY USING EQUI JOIN

WHO ARE WORKING IN THE LOCATION IS 'CHICAGO' ?

SOL:

SQL> SELECT * FROM EMP E INNER JOIN DEPT D ON E.DEPTNO=D.DEPTNO AND LOC='CHICAGO';

EX4:

WAQ TO DISPLAY SUM OF SALARIES OF DEPARTMENTS FROM EMP,DEPT TABLES

BY USING EQUI JOIN ?

SOL:

SQL> SELECT DNAME,SUM(SAL) FROM EMP E INNER JOIN DEPT D ON E.DEPTNO=D.DEPTNO GROUP BY DNAME;

EX5:

WAQ TO DISPLAY SUM OF SALARIES OF DEPARTMENTS FROM EMP,DEPT TABLES

BY USING EQUI JOIN IF SUM OF SALARIES OF DEPARTMENTS ARE MORE THAN

10000?

SOL:

```
SQL> SELECT DNAME,SUM(SAL) FROM EMP E INNER JOIN DEPT D ON E.DEPTNO=D.DEPTNO
        GROUP BY DNAME HAVING SUM(SAL)>10000;
```

USING ROLLUP & CUBE CLAUSES IN JOINS:

EX6:

```
SQL> SELECT DNAME,COUNT(*) FROM EMP E INNER JOIN DEPT D ON E.DEPTNO=D.DEPTNO
        GROUP BY ROLLUP(DNAME);
```

EX7:

```
SQL> SELECT D.DEPTNO,DNAME,COUNT(*) FROM EMP E INNER JOIN DEPT D
        ON E.DEPTNO=D.DEPTNO GROUP BY CUBE(D.DEPTNO,DNAME);
```

OUTER JOINS:

- IN THE ABOVE EQUI / INNER JOIN WE ARE RETRIEVING ONLY MATCHING ROWS BUT NOT UN MATCHING ROWS FROM MULTIPLE TABLES.SO TO OVERCOME THIS PROBLEM THEN WE USE "OUTER JOINS" MECHANISM.THESE ARE AGAIN THREE TYPES:

1. LEFT OUTER JOIN
2. RIGHT OUTER JOIN
3. FULL OUTER JOIN

LEFT OUTER JOIN:

- RETRIEVING ALL ROWS(MATCHING & UN MATCHING) FROM LEFT SIDE TABLE BUT RETRIEVING MACTHING ROWS FROM RIGHT SIDE TABLE.

EX:

```
SQL> SELECT * FROM STUDENT S LEFT OUTER JOIN COURSE  C ON S.CID=C.CID;
```

RIGHT OUTER JOIN:

- RETRIEVING ALL ROWS(MATCHING & UN MATCHING) FROM RIGHT SIDE TABLE BUT RETRIEVING MACTHING ROWS FROM LEFT SIDE TABLE.

EX:

```
SQL> SELECT * FROM STUDENT S RIGHT OUTER JOIN COURSE C ON S.CID=C.CID;
```

FULL OUTER JOIN:

- RETRIEVING MACTHING AND ALSO UN MATCHING ROWS FROM BOTH SIDES

TABLES.

EX:

```
SELECT * FROM STUDENT S FULL OUTER JOIN COURSE  C ON S.CID=C.CID;
```

NON-EQUI JOIN:

- RETRIEVING DATA FROM MULTIPLE TABLES BASED ON ANY CONDITION EXCEPT EQUAL OPERATOR CONDITION IS CALLED AS NON-EQUI JOIN.IN THIE JOIN WE CAN USE THE FOLLOWING OPERATORS ARE <,>,<=,>=,AND,BETWEEN,.........etc.

EX1:

```
SQL> SELECT * FROM TEST1 T1 JOIN TEST2 T2 ON T1.SNO>T2.SNO;
```

EX2:

WAQ TO DISPLAY ENAME,SALARY,LOW SALARY,HIGH SALARY FROM EMP,SALGRADE

TABLES WHOSE SALARY BETWEEN LOW SALARY AND HIGH SALARY ?

SOL:

```
SQL> SELECT ENAME,SAL,LOSAL,HISAL FROM EMP JOIN SALGRADE

      ON SAL BETWEEN LOSAL AND HISAL;
```

(OR)

```
SQL> SELECT ENAME,SAL,LOSAL,HISAL FROM EMP JOIN SALGRADE

      ON (SAL>=LOSAL) AND (SAL<=HISAL);
```

CROSS JOIN / CARTISEAN JON:

- JOINING TWO (OR) MORE THAN TWO TABLES WITHOUT ANY CONDITION IS CALLED AS "CROSS / CARTISEAN JOIN".

- IN CROSS JOIN,EACH ROW OF THE FIRST TABLE WILL JOIN JOINS WITH EACH ROW OF THE SECOND TABLE.THAT MEANS A FIRST TABLE IS HAVING "m" NO.OF

ROWS AND A SECOND TABLE IS HAVING "n" NO.OF ROWS THEN THE RESULT IS mXn NO.OF ROWS.

EX1:

```
SQL> SELECT * FROM STUDENT CROSS JOIN COURSE;
```

EX2:

```
SQL> SELECT I1.INAME,I1.PRICE,I2.INAME,I2.PRICE,

       I1.PRICE+I2.PRICE TOTAL_AMOUNT FROM

      ITEMS1 I1 CROSS JOIN ITEMS2 I2;
```

OUTPUT:

```
--------------

INAME                PRICE        INAME                PRICE        TOTAL_AMOUNT

--------------------  ----------   --------------------  ----------   ------------

PIZZA                250          COCACOLA                 25       275
```

NATURAL JOIN:

- SQLSERVER DOESNOT SUPPORTING NATURAL JOIN.

SELF JOIN: JOINING A TABLE BY ITSELF IS CALLED AS SELF JOIN.IN SELF JOIN A ROW IN ONE TABLE JOINED WITH THE ROW OF SAME TABLE.WHEN WE USE SELF JOIN MECHANISM THEN WE SHOULD CREATE ALIAS NAMES ON A TABLE.ONCE WE CREATE ALIAS NAME ON A TABLE INTERNALLY ORACLE SERVER IS CREATING VIRTUAL TABLE(COPY) ON EACH ALIAS NAME.

WE CAN CREATE ANY NO.OF ALIAS NAMES ON A SINGLE TABLE BY EACH ALIAS NAMESHOULD BE DIFFERENT NAME.SELF JOIN CAN BE IMPLEMENTED AT TWO SITUATIONS:

1. COMPARING A SINLGE COLUMN VALUES BY ITSELF IN THE TABLE.

2. COMPARING TWO DIFFERENT COLUMNS VALUES TO EACH OTHER IN THE TABLE.

EX.ON COMPARING A SINGLE COLUMN VALUES BY ITSELF:

Q: WAQ TO DISPLAY EMPLOYEE WHO ARE WORKING IN THE SAME LOCATION OF THE EMPLOYEE IS "SCOTT" ?

SOL:

```
SQL> SELECT T1.ENAME,T1.LOC FROM TEST T1 JOIN TEST T2 ON T1.LOC=T2.LOC

       AND T2.ENAME='SCOTT';
```

Q: WAQ TO DISPLAY EMPLOYEE WHOSE SALARY IS SAME AS THE SALARY OF THE EMPLOYEE FORD?

SOL:

```
SQL> SELECT E1.ENAME,E1.SAL FROM EMP E1 JOIN EMP E2

        ON E1.SAL=E2.SAL AND E2.ENAME='FORD';
```

<u>EX.ON COMPARING TWO DIFF.COLUMNS TO EACH OTHER:</u>

EX1:

WAQ TO DISPLAY MANAGERS AND THEIR EMPLOYEES FROM

EMP TABLE?

```
SQL> SELECT M.ENAME MANAGER,E.ENAME EMPLOYEES FROM EMP E JOIN EMP M

         ON M.EMPNO=E.MGR;
```

EX2:

WAQ TO DISPLAY EMPLOYEE WHO ARE GETTING MORE THAN THEIR MANAGER SALARY?

SOL:

```
SQL> SELECT M.ENAME MANAGER,M.SAL MSALARY,E.ENAME EMPLOYEE,E.SAL ESALARY FROM EMP E
JOIN EMP M ON M.EMPNO=E.MGR AND E.SAL>M.SAL;
```

EX3:WAQ TO DISPLAY EMPLOYEE WHO ARE JOINED BEFORE THEIR MANAGER ?

SOL: SELECT M.ENAME MANAGER,M.HIREDATE MDOJ,E.ENAME EMPLOYEE,E.HIREDATE EDOJ FROM EMP E JOIN EMP M ON M.EMPNO=E.MGR AND E.HIREDATE<M.HIREDATE;

<u>HOW TO JOIN MORE THAN TWO TABLES:</u>

<u>SYNTAX FOR JOINS:</u>

```
SELECT * FROM <TN1> <JOIN KEY> <TN2> ON <CONDITION1>

<JOIN KEY> <TN3> ON <CONDITION2> <JOIN KEY> <TN4> ............;
```

<u>INNER JOIN / EQUI JOIN</u>:

```
SQL> SELECT * FROM STUDENT S INNER JOIN COURSE C ON S.CID=C.CID

        INNER JOIN REGISTOR R ON C.CID=R.CID;
```