JOINS:

JOINS ARE USED TO RETRIEVE DATA FROM MULTIPLE TABLES AT A TIME. IN RELATIONAL DATABASES WE ARE STORING RELATED DATA IN MULTIPLE TABLES LIKE EMPLOYEE DETAILS, DEPARTMENT DETAILS, CUSTOMER DETILS, ORDERSDETAILS, PRODUCTS DETAILS, ……. ETC.

TO COMBINED DATA AND RETRIEVE DATA FROM THOSE MULTIPLE TABLES THEN WE NEED JOINS.JOINS ARE AGAIN CLASSIFIED INTO TWO WAYS

1) NON - ANSI FORMAT JOINS: (ORACLE 8I JOINS)

- **EQUI JOIN**
- **NON -EQUI JOIN**
- **SELF JOIN**

2) ANSI FORMAT JOINS: (ORACLE 9I JOINS)

- **INNER JOIN**
- **OUTER JOIN**
  - **LEFT OUTER JOIN**
  - **RIGHT OUTER JOIN**
  - **FULL OUTER JOIN**
- **CROSS JOIN (OR) CARTISEAN JOIN**
- **NATURAL JOIN**

- WHEN WE ARE RETRIEVING DATA FROM MULTIPLE TABLES BASED ON "WHERE"

CLAUSE CONDITION THEN WE CALLED AS NON-ANSI FORMAT JOIN.

- WHEN WE ARE RETRIEVING DATA FROM MULTIPLE TABLES WITH "ON" / "USING"

CLAUSE CONDITION THEN WE CALLED AS ANSI FORMAT JOIN.

SYNTAX FOR NON-ANSI JOINS:

SELECT * FROM TABLE NAME1, TABLE NAME2 WHERE <JOIN CONDITION>;

SYNTAX FOR ANSI JOINS:

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

JOINS TABLES:

EX:

STUDENT TABLE

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

COURSE TABLE

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

EQUI JOIN: RETRIEVING DATA FROM MULTIPLE TABLES BASED ON "EQUAL OPERATOR (=) "IS CALLED AS EQUI JOIN.

WHEN WE USE EQUI JOIN BETWEEN TWO (OR) MORE THAN TWO TABLES THERE MUST BE 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).EQUI JOIN ALWAYS RETRIEVING ONLY MATCHING DATA / MATCHNG ROWS.

SYNTAX:

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

(OR)

WHERE <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 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, COURSE WHERE
STUDENT.CID=COURSE.CID;
```

(OR)

```
SQL> SELECT * FROM STUDENT S, COURSE C WHERE 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,COURSE C WHERE S.CID=C.CID
AND C.CID=20;
```

EX3:

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

WHO ARE WORKING IN THE LOCATION IS 'CHICAGO'?

SOL:

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

EX7:

SQL> SELECT D. DEPTNO, DNAME, COUNT (*) FROM EMP E, DEPT D

WHERE E. DEPTNO=D.DEPTNO GROUP BY CUBE (D. DEPTNO, DNAME);

INNER JOIN:

- INNER JOIN IS SIMILAR TO EQUI JOIN.RETRIEVING DATA FROM MULTIPLE

TABLES WITH "ON" CLAUSE CONDITION.

SYNTAX:

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

(OR)

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

EX1:

WAQ TO RETRIEVE STUDENT,COURSE DETAILS FROM TABLES BY USING INNER JOIN?

SOL:

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

EX2:

WAQ TO DISPLAY EMPLOYEE FROM EMP,DEPT TABLES BY USING INNER 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';
```

(OR)

```
SQL> SELECT * FROM EMP E INNER JOIN DEPT D ON
E.DEPTNO=D.DEPTNO

WHERE LOC='CHICAGO';
```

<u>WHY ANSI JOINS:</u>

THESE JOINS ARE INTRODUCED IN ORACLE 9I.THE MAIN ADVANTAGE OF ANSI JOINS ARE PORTABILITY. IT MEANS THAT WE CAN MOVE JOIN STATEMENTS FROM ONE DATABASE TO ANOTHER DATABASE WITHOUT MAKING ANY CHANGES AS IT IS THE JOIN STATEMENTS ARE EXECUTED IN OTHER DATABASES.

<u>OUTER JOINS:</u>

- 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 MATCHING ROWS FROM RIGHT SIDE TABLE.

ANSI FORMAT:

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

(OR)

NON - ANSI FORMAT:

- WHEN WE WRITE OUTER JOINS IN NON-ANSI FORMAT THEN WE SHOULD USE

JOIN OPERATOR (+).

EX:

```
SQL> SELECT * FROM STUDENT S,COURSE C WHERE S.CID=C.CID(+);
```

RIGHT OUTER JOIN:

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

ANSI FORMAT:

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

(OR)

NON-ANSI FORMAT:

SQL> SELECT * FROM STUDENT S,COURSE C WHERE S.CID(+)=C.CID;

FULL OUTER JOIN:

- RETRIEVING MATCHING AND ALSO UN MATCHING ROWS FROM BOTH SIDES TABLES.

ANIS FORMAT:

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

(OR)

NON - ANSI FORMAT:

SELECT * FROM STUDENT S,COURSE C WHERE S.CID(+)=C.CID(+);

ERROR AT LINE 1:

ORA-01468: A PREDICATE MAY REFERENCE ONLY ONE OUTER-JOINED TABLE

NOTE: NON-ANSI FORMAT IS NOT SUPPORTING FULL OUTER JOIN MECHANISM. SO THAT WHEN WE WANT TO IMPLEMENT FULL OUTER JOIN IN NON -ANSI FORMAT THEN WE COMBINED THE RESULTS OF LEFT OUTER AND RIGHT OUTER JOINS BY USING "UNION" OPERATOR.

EX:

SELECT * FROM STUDENT S,COURSE C WHERE S.CID=C.CID(+)

UNION

SELECT * FROM STUDENT S,COURSE C WHERE 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:

NON-ANSI:

SQL> SELECT * FROM TEST1 T1,TEST2 T2 WHERE T1.SNO>T2.SNO;

(OR)

ANSI:

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,SALGRADE
        WHERE SAL BETWEEN LOSAL AND HISAL;
```

(OR)

```
SQL> SELECT ENAME,SAL,LOSAL,HISAL FROM EMP,SALGRADE
        WHERE (SAL>=LOSAL) AND (SAL<=HISAL);
```

CROSS JOIN / CARTESIAN JON:

- JOINING TWO (OR) MORE THAN TWO TABLES WITHOUT ANY CONDITION IS CALLED AS "CROSS / CARTESIAN 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;---ANSI

(OR)

SQL> SELECT * FROM STUDENT,COURSE;----NON-ANSI

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
```

EX3:

```
SELECT * FROM COLORS CROSS JOIN SIZES;------ANSI
```

(OR)

```
SELECT * FROM COLORS,SIZES;-----NON - ANSI
```

NATURAL JOIN:

- NATURAL JOIN IS SIMILAR TO EQUI JOIN. WHEN WE USE NATURAL JOIN, WE SHOULD HAVE A COMMON COLUMN NAME. THIS COLUMN DATA TYPE MUST BE MATCH.

- WHENEVER WE ARE USING NATURAL JOIN THERE IS NO NEED TO WRITE A JOINING CONDITION BY EXPLICITLY BECAUSE INTERNALLY ORACLE SERVER IS PREPARING JOINING CONDITION BASED ON AN "EQUAL OPERATOR(=)" WITH COLUMN COLUMN NAME AUTOMATICALLY.

- BY USING NATURAL JOIN WE AVOID DUPLICATE COLUMNS WHILE RETRIEVING DATA FROM MULTIPLE TABLES.

EX:

SQL> SELECT * FROM STUDENT S NATURAL JOIN COURSE C;

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 ATABLE.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 NAME SHOULD BE DIFFERENT NAME.

- SELF JOIN CAN BE IMPLEMENTED AT TWO SITUATIONS:

1. COMPARING A SINGLE 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,TEST T2 WHERE
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,EMP E2
       WHERE E1.SAL=E2.SAL AND E2.ENAME='FORD';
```

EX. ON COMPARING TWO DIFF.COLUMNS TO EACH OTHER:

EX1:

WAQ TO DISPLAY MANAGERS AND THEIR EMPLOYEES FROM

EMP TABLE?

SSQL> SELECT M.ENAME MANAGER,E.ENAME EMPLOYEES FROM EMP E,EMP M

WHERE 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,EMP M WHERE M.EMPNO=E.MGR AND E.SAL>M.SAL;

EX3:

WAQ TO DISPLAY EMPLOYEE WHO ARE JOINED BEFORE THEIR MANAGER ?

SOL:

SQL> SELECT M.ENAME MANAGER,M.HIREDATE MDOJ,E.ENAME EMPLOYEE,E.HIREDATE EDOJ FROM EMP E,EMP M WHERE M.EMPNO=E.MGR AND E.HIREDATE<M.HIREDATE;

USING CLAUSE: IN ANSI FORMAT JOINS WHENEVER WE JOIN TWO OR MORE THAN TWO TABLES INSTEAD OF "ON" CLAUSE WE CAN USE "USING" CLAUSE ALSO. IT RETURNS COMMON COLUMN ONLY ONE TIME.

EX:

SQL> SELECT * FROM STUDENT S INNER JOIN COURSE C USING (S.CID);

ERROR AT LINE 2:

ORA-01748: ONLY SIMPLE COLUMN NAMES ALLOWED HERE

NOTE: WHEN WE USE "USING" CLAUSE WITH COMMON COLUMN NAME THERE IS NO NEED TO PREFIX WITH TABLE ALIAS NAME.

EX:

USING(S.CID); -------ERROR

USING(CID); ------ALLOWED

EX:

```
SQL> SELECT * FROM STUDENT S INNER JOIN COURSE C
USING(CID);
```

HOW TO JOIN MORE THAN TWO TABLES:

SYNTAX FOR NON-ANSI JOINS:

```
SELECT * FROM <TN1>,<TN2>,<TN3>,.....<TN N>

WHERE <CONDITION1> AND <CONDITION2> AND
<CONDITON3>.....;
```

EQUI JOIN:

```
SQL> SELECT * FROM STUDENT S,COURSE C,REGISTOR R

     WHERE S.CID=C.CID AND C.CID=R.CID;
```

SYNTAX FOR ANSI JOINS:

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

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

INNER JOIN:

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

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

(OR)

```
SQL> SELECT * FROM STUDENT S INNER JOIN COURSE C
USING(CID)

     INNER JOIN REGISTOR R USING(CID);
```

DIFFERENCES BETWEEN JOINS AND SET OPERATORS:

JOINS	SET OPERATORS
1. COMBINED DATA IN COLUMNS WISE.	- COMBINED DATA IN ROWS WISE.
2. COMBINED DATA HORIZONTAL	- COMBINED DATA VERTICALLY
3. DIFFERENT STRUCTURE TABLES TABLES ARE JOINED	- SIMILAR STRUCTURE ARE JOINED.