Database Management System Lab (CS-2094)

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

School of Computer Engineering

1 Credit

Dr. Jayanta Mondal

Lab Contents

Sr #	Major and Detailed Coverage Area	Lab#
1	DML ❑ Group by clause ❑ Having clause ❑ Group by with Order by ❑ Joins ❑ Set Operators	5

Grouping Data

GROUP BY clause

GROUP BY clause is used for grouping data. The column appearing in SELECT statement must also appear in GROUP BY clause. The syntax of this is:
SELECT column, groupfunction(column) FROM tablename GROUP BY column;

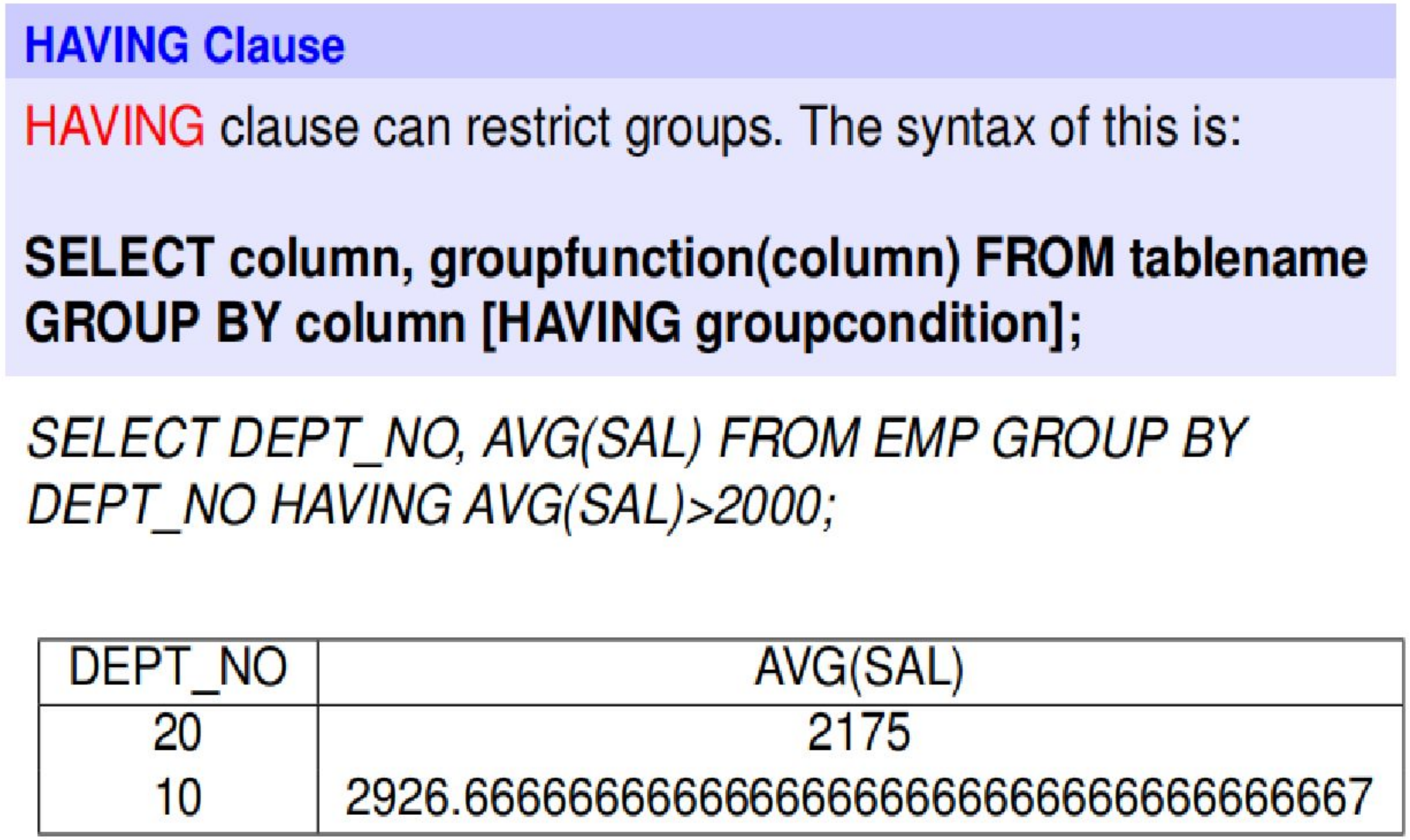
SELECT DEPT_NO FROM EMP GROUP BY DEPT_NO;

DEPT_NO
30
20
10

SELECT DEPT_NO, AVG(SAL) FROM EMP GROUP BY DEPT_NO;

DEPT_NO	AVG(SAL)
30	1566.6666666666666666666666666666666667
20	2175
10	2926.6666666666666666666666666666666667

Grouping……



HAVING Clause

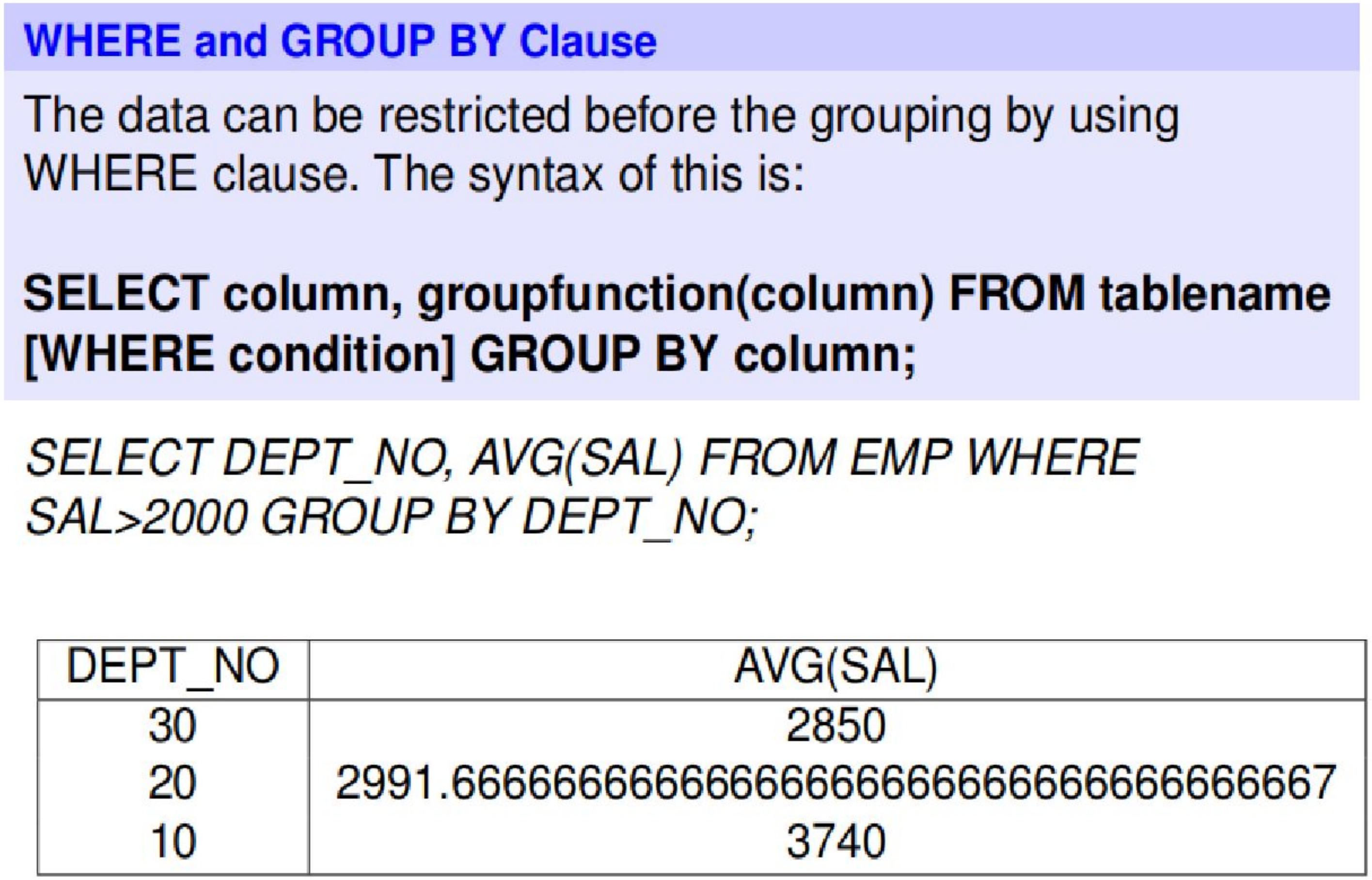
HAVING clause can restrict groups. The syntax of this is:

SELECT column, groupfunction(column) FROM tablename GROUP BY column [HAVING groupcondition];

SELECT DEPT_NO, AVG(SAL) FROM EMP GROUP BY DEPT_NO HAVING AVG(SAL)>2000;

DEPT_NO	AVG(SAL)
20	2175
10	2926.66666666666666666666666666666666667

Grouping……



WHERE and GROUP BY Clause

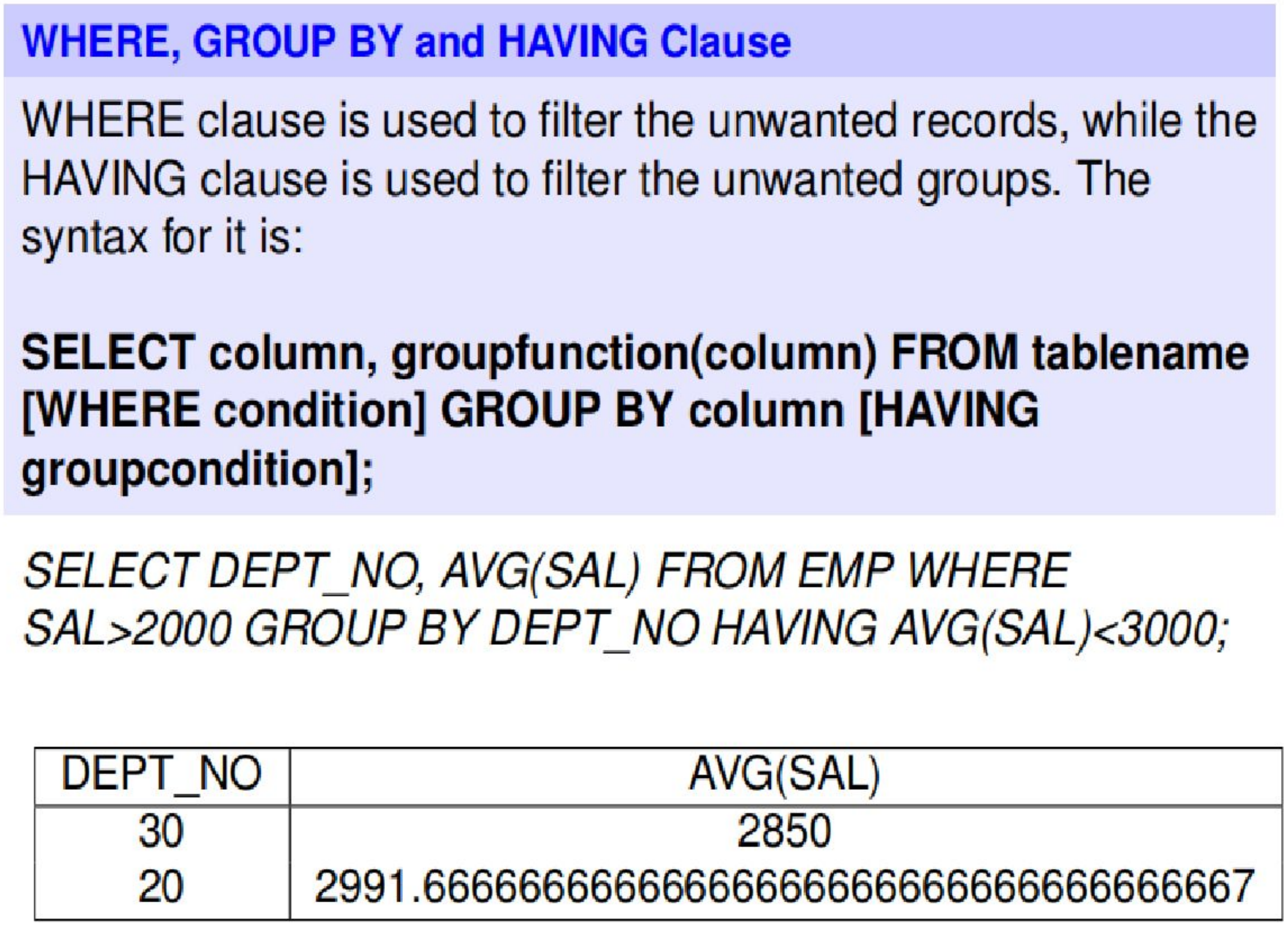
The data can be restricted before the grouping by using WHERE clause. The syntax of this is:

SELECT column, groupfunction(column) FROM tablename [WHERE condition] GROUP BY column;

SELECT DEPT_NO, AVG(SAL) FROM EMP WHERE SAL>2000 GROUP BY DEPT_NO;

DEPT_NO	AVG(SAL)
30	2850
20	2991.66666666666666666666666666666666667
10	3740

Grouping......



WHERE, GROUP BY and HAVING Clause

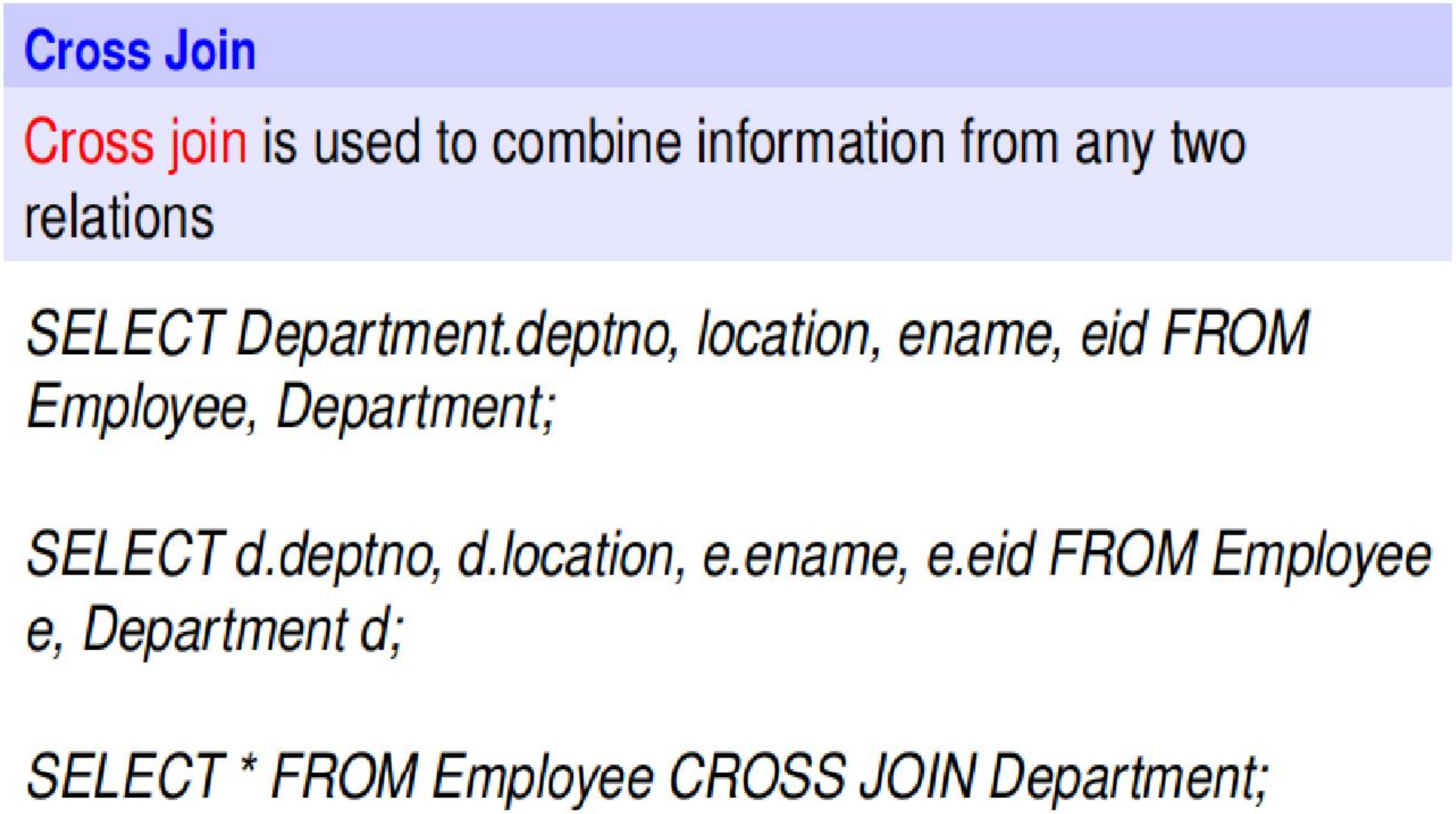
WHERE clause is used to filter the unwanted records, while the HAVING clause is used to filter the unwanted groups. The syntax for it is:

SELECT column, groupfunction(column) FROM tablename [WHERE condition] GROUP BY column [HAVING groupcondition];

SELECT DEPT_NO, AVG(SAL) FROM EMP WHERE SAL>2000 GROUP BY DEPT_NO HAVING AVG(SAL)<3000;

DEPT_NO	AVG(SAL)
30	2850
20	2991.6666666666666666666666666666666667

Cross Join



Cross Join

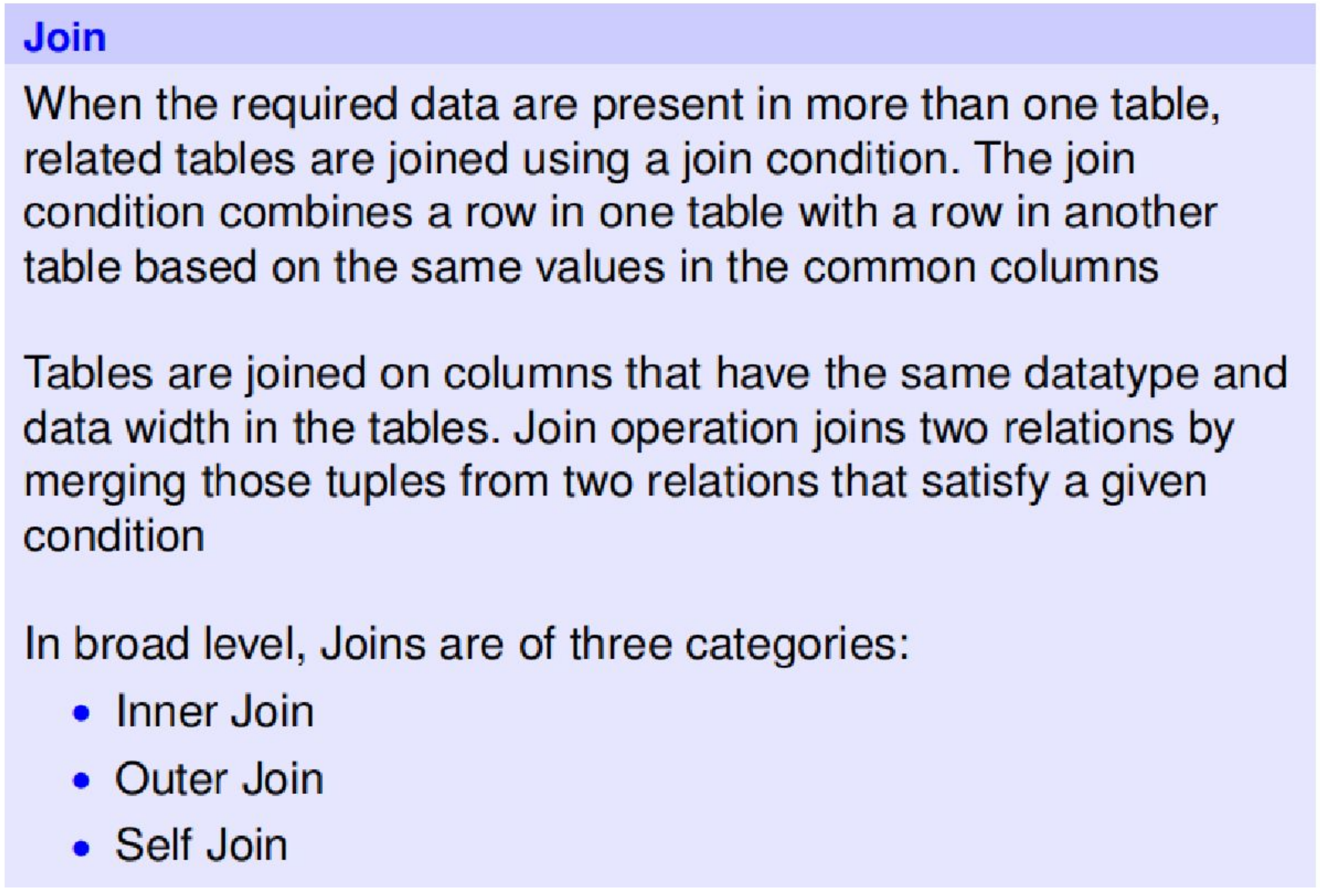
Cross join is used to combine information from any two relations

```
SELECT Department.deptno, location, ename, eid FROM
Employee, Department;

SELECT d.deptno, d.location, e.ename, e.eid FROM Employee
e, Department d;

SELECT * FROM Employee CROSS JOIN Department;
```

Join



Join

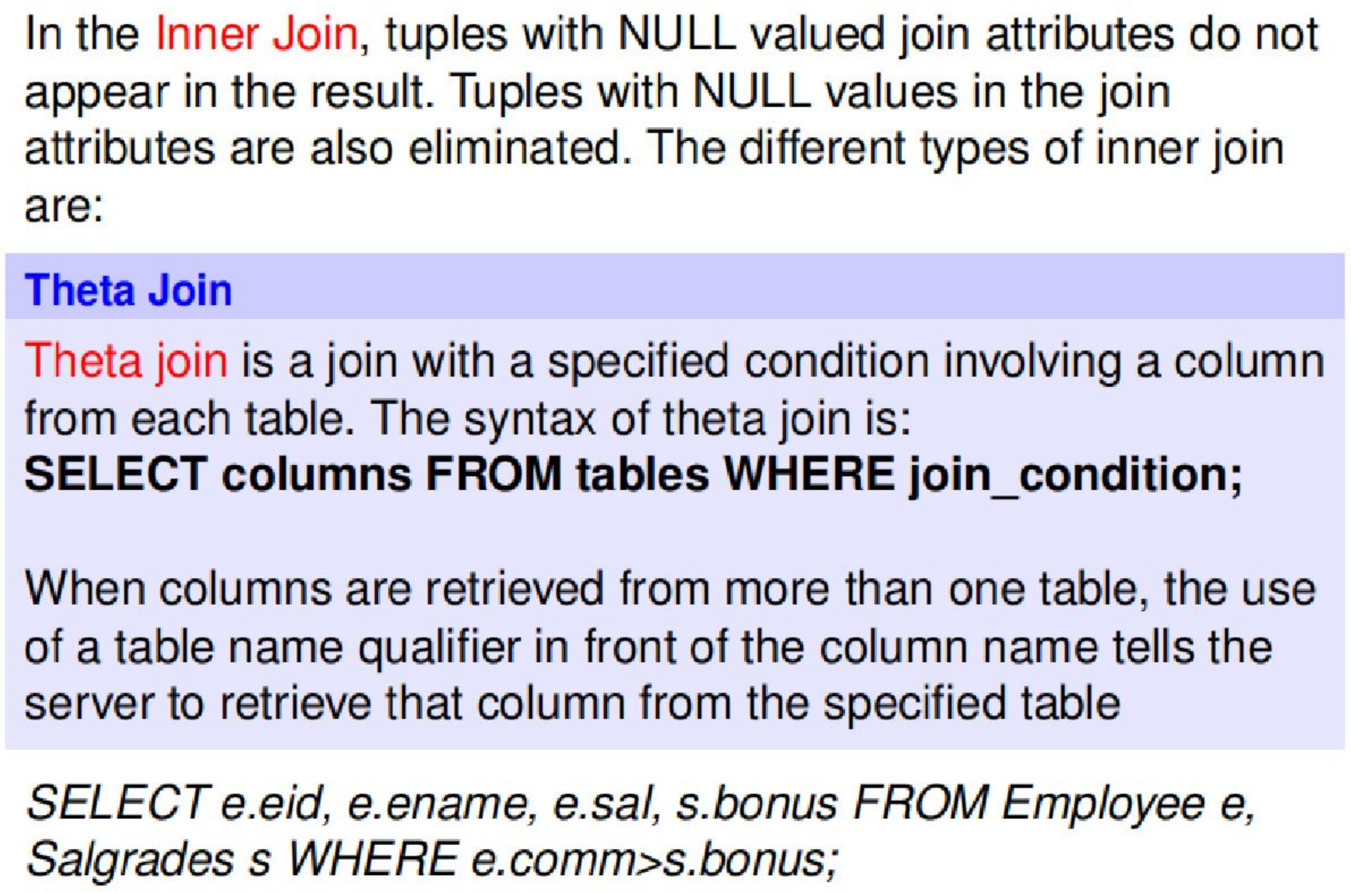
When the required data are present in more than one table, related tables are joined using a join condition. The join condition combines a row in one table with a row in another table based on the same values in the common columns

Tables are joined on columns that have the same datatype and data width in the tables. Join operation joins two relations by merging those tuples from two relations that satisfy a given condition

In broad level, Joins are of three categories:

- Inner Join
- Outer Join
- Self Join

Inner Join...



In the Inner Join, tuples with NULL valued join attributes do not appear in the result. Tuples with NULL values in the join attributes are also eliminated. The different types of inner join are:

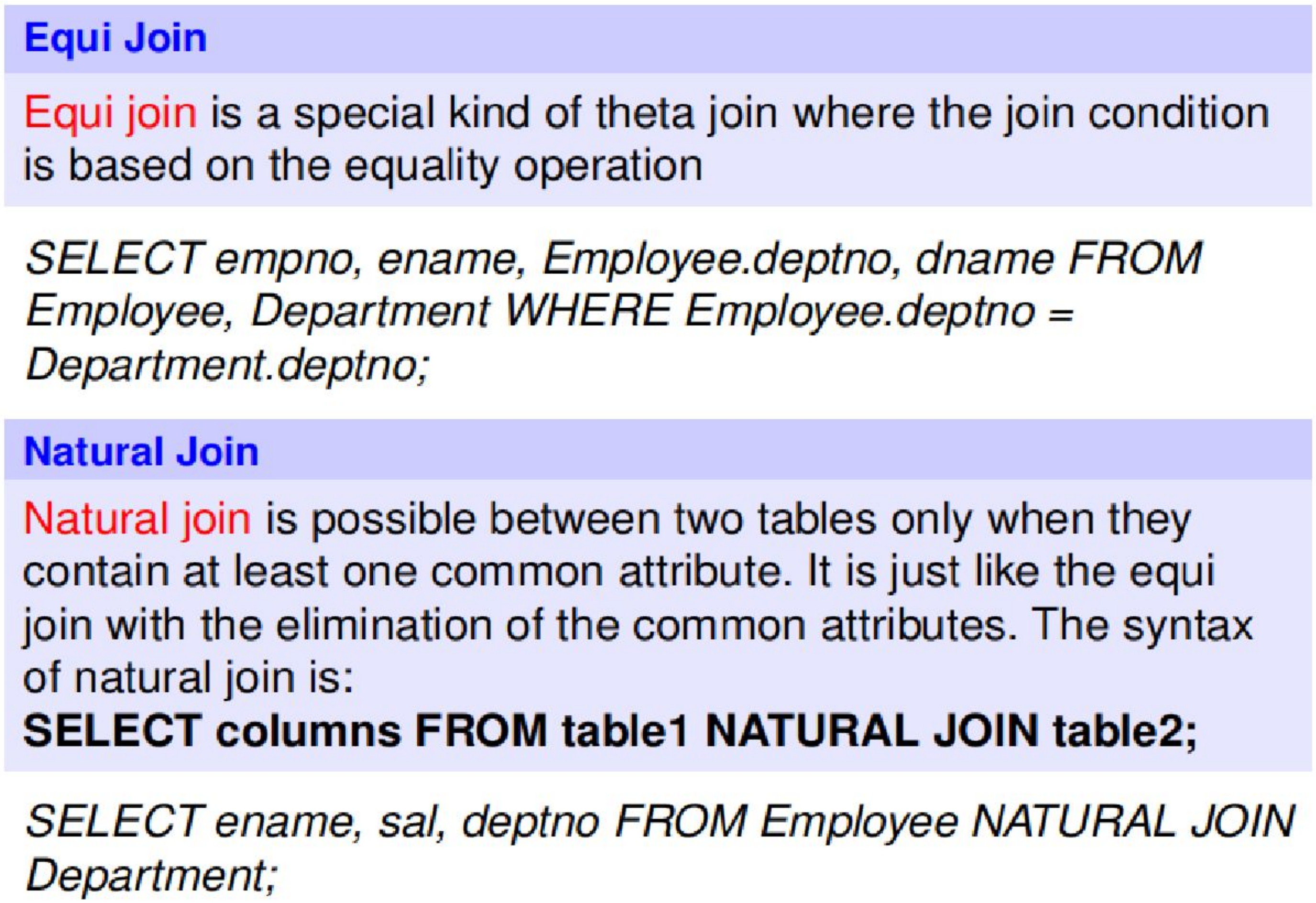
Theta Join

Theta join is a join with a specified condition involving a column from each table. The syntax of theta join is:
SELECT columns FROM tables WHERE join_condition;

When columns are retrieved from more than one table, the use of a table name qualifier in front of the column name tells the server to retrieve that column from the specified table

SELECT e.eid, e.ename, e.sal, s.bonus FROM Employee e, Salgrades s WHERE e.comm>s.bonus;

Inner Join....



Equi Join

Equi join is a special kind of theta join where the join condition is based on the equality operation

SELECT empno, ename, Employee.deptno, dname FROM Employee, Department WHERE Employee.deptno = Department.deptno;

Natural Join

Natural join is possible between two tables only when they contain at least one common attribute. It is just like the equi join with the elimination of the common attributes. The syntax of natural join is:
SELECT columns FROM table1 NATURAL JOIN table2;

SELECT ename, sal, deptno FROM Employee NATURAL JOIN Department;

Outer Join

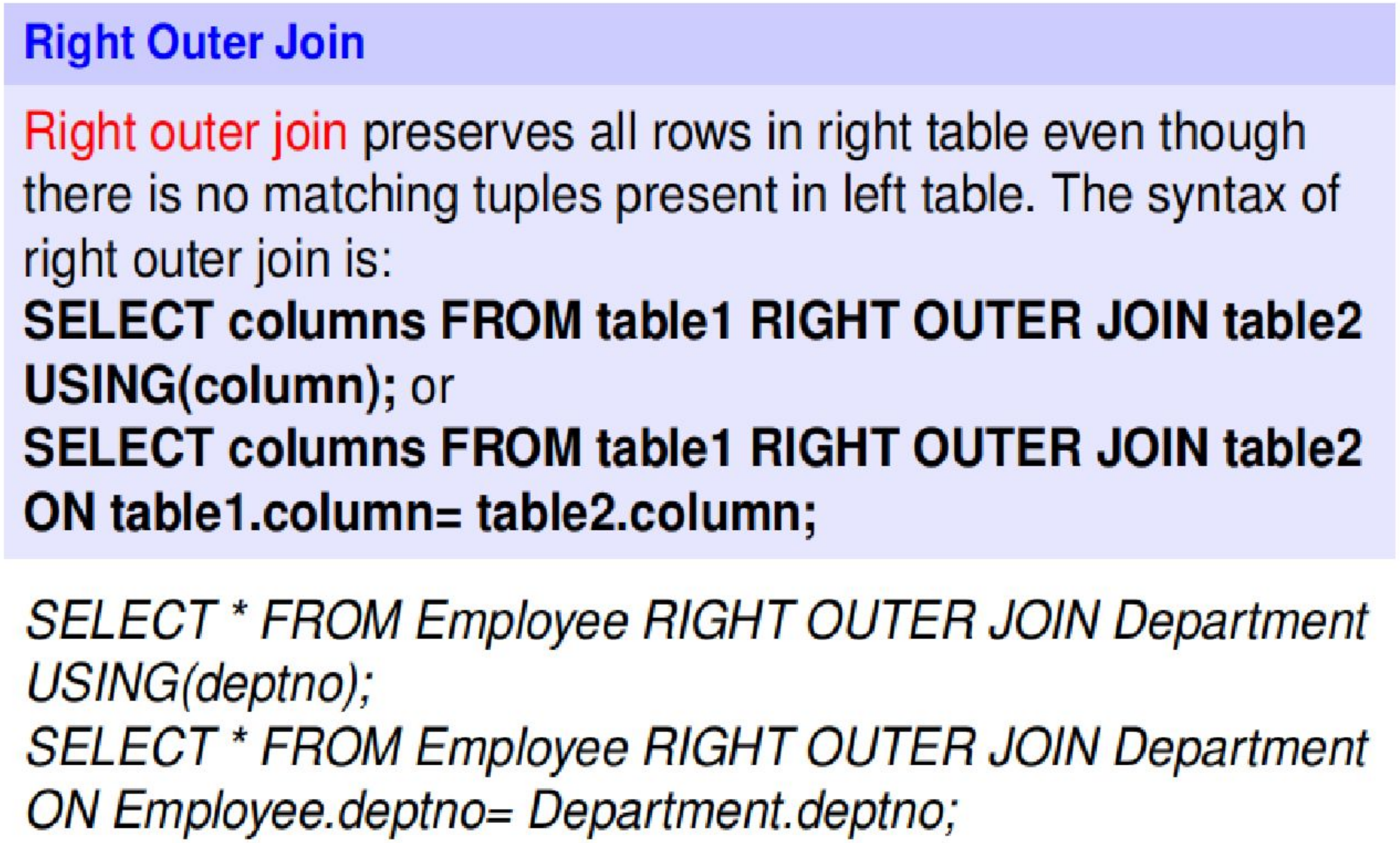
Outer join is an extension of the natural join operation to deal with the missing information. The different types of outer join are:

Left Outer Join

Left outer join preserves all rows in left table even though there is no matching tuples present in the right table. The syntax of left outer join is:
SELECT columns FROM table1 LEFT OUTER JOIN table2 USING(column); or
SELECT columns FROM table1 LEFT OUTER JOIN table2 ON table1.column= table2.column;

*SELECT * FROM Employee LEFT OUTER JOIN Department USING(deptno);*
*SELECT * FROM Employee LEFT OUTER JOIN Department ON Employee.deptno= Department.deptno;*

Outer Join....



Right Outer Join

Right outer join preserves all rows in right table even though there is no matching tuples present in left table. The syntax of right outer join is:

SELECT columns FROM table1 RIGHT OUTER JOIN table2 USING(column); or

SELECT columns FROM table1 RIGHT OUTER JOIN table2 ON table1.column= table2.column;

*SELECT * FROM Employee RIGHT OUTER JOIN Department USING(deptno);*

*SELECT * FROM Employee RIGHT OUTER JOIN Department ON Employee.deptno= Department.deptno;*

Outer Join....

Full Outer Join

Full outer join preserves all records in both tables. The syntax of full outer join is:

```
SELECT columns FROM table1 FULL OUTER JOIN table2
USING(column); or

SELECT columns FROM table1 FULL OUTER JOIN table2
ON table1.column= table2.column;
```

```
SELECT * FROM Employee FULL OUTER JOIN Department
USING(deptno);

SELECT * FROM Employee FULL OUTER JOIN Department
ON Employee.deptno= Department.deptno;
```

Self Join

Self Join

Self join is similar to the theta join. It joins a relation to itself by a condition. When a table is joined to itself, two copies of the same table are used. The syntax for this is:

SELECT columns FROM table T1, table T2 WHERE T1.column operator T2.column;

SELECT e.ename AS ëmployee", m.ename AS managerFROM Employee m, Employee e WHERE e.mgr=m.empno;

Set Operators

To perform the set operations such as UNION, DIFFERENCE and INTERSECTION, the relations need to be union compatible for the result to be a valid relation. The different set operators are:

UNION

Union is used to combine data from two relations

SELECT name, job FROM Employee WHERE dept=20 UNION SELECT name, job FROM Employee WHERE dept=30;

Set Operators....

DIFFERENCE

Difference is used to identify the rows that are in one relation and not in another

```
SELECT name, job FROM Employee WHERE dept=20 MINUS
SELECT name, job FROM Employee WHERE dept=30;
```

INTERSECTION

Intersection is used to identify the rows that are common to two relations

```
SELECT name, job FROM Employee WHERE dept=20
INTERSECT SELECT name, job FROM Employee WHERE
dept=30;
```