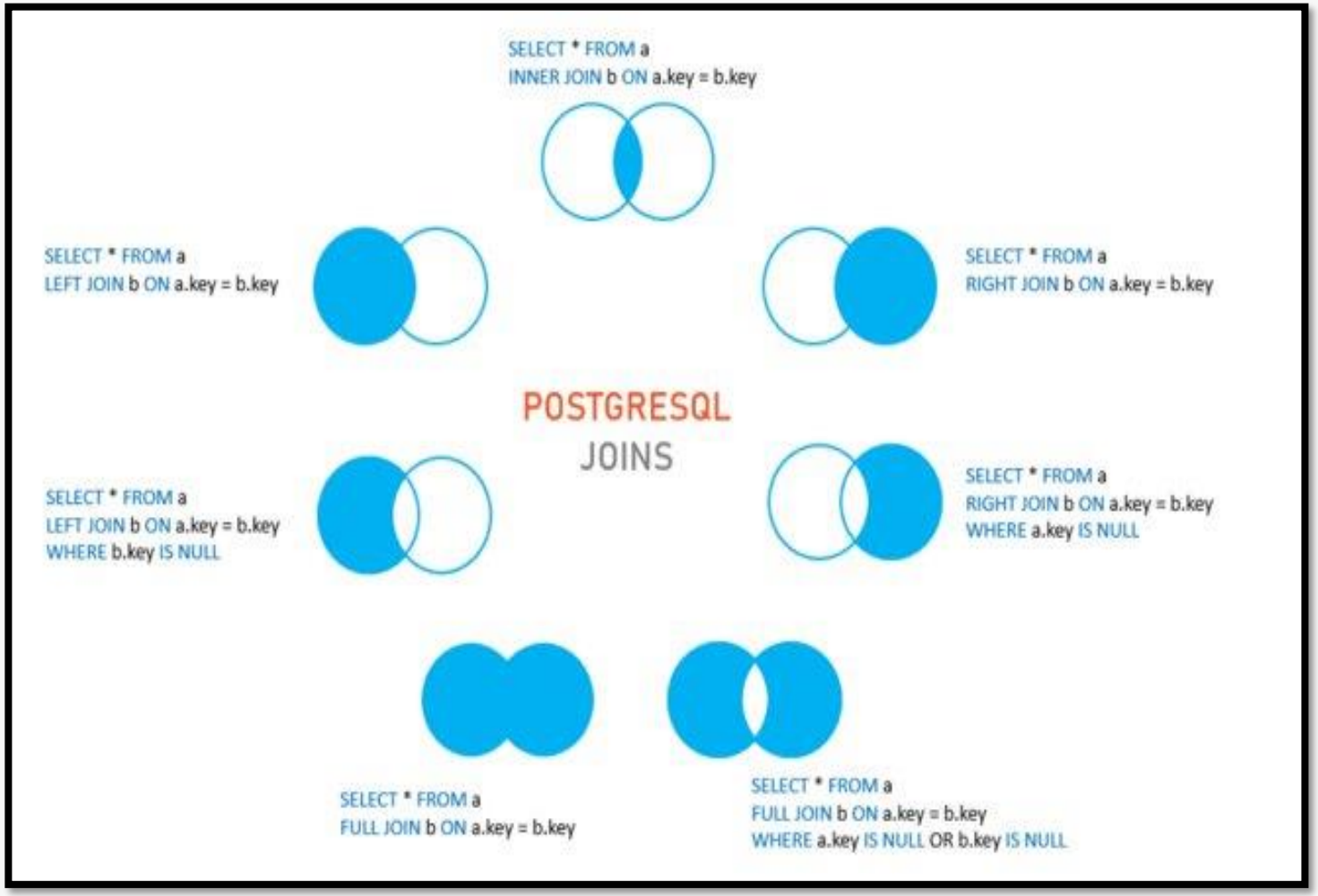
netlink
digital solutions

DATABASE

www.netlink.com

PostgreSQL Joins

- PostgreSQL join is used to combine columns from one (self-join) or more tables based on the values of the common columns between related tables.
- The common columns are typically the primary key columns of the first table and foreign key columns of the second table.
- PostgreSQL supports inner join, left join, right join, full outer join, cross join, natural join, and a special kind of join called self-join.

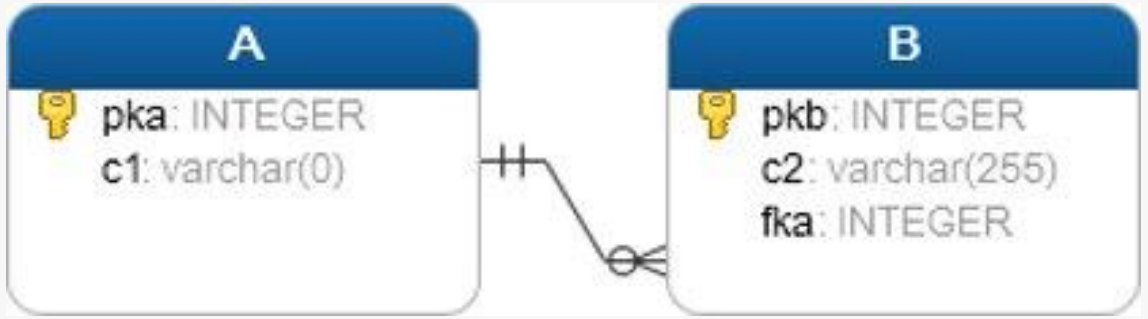


Inner Join

In a relation database, data is typically distributed in more than one table. To select complete data, you often need to query data from multiple tables.

To join table A with the table B, you follow these steps:

- First, specify columns from both tables that you want to select data in the SELECT clause.
- Second, specify the main table i.e., table A in the FROM clause.
- Third, specify the second table (table B) in the `INNER JOIN` clause and provide a join condition after the ON keyword.

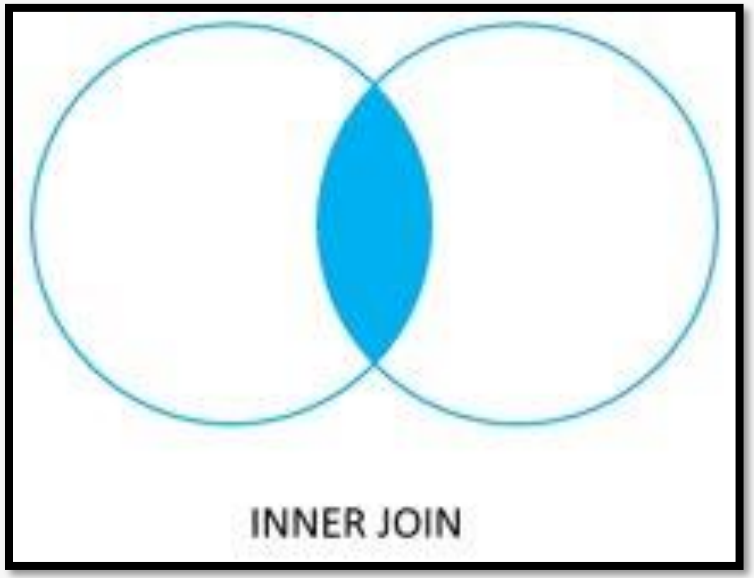


NOTE: ON is used with **JOIN in a join condition to identify which columns in each table to link** and can be used with all types of joins.

Inner Join

SYNTAX :

```
SELECT  pka,  c1,  pkb,  c2
FROM    A
INNER JOIN B ON pka = fka;
```



How the `INNER JOIN` works.

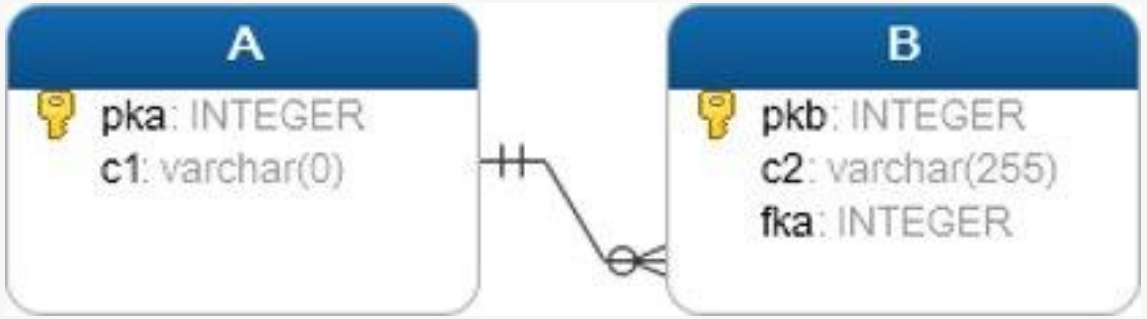
For each row in the table A, inner join compares the value in the pka column with the value in the fka column of every row in the table B:

- If these values are equal, the inner join creates a new row that contains all columns of both tables and adds it to the result set.
- In case these values are not equal, the inner join just ignores them and moves to the next row.

Left Join

- Each row in the table A may have zero or many corresponding rows in the table B while each row in the table B has one and only one corresponding row in the table A .

- To select data from the table A that may or may not have corresponding rows in the table B , you use the **LEFT JOIN** clause.



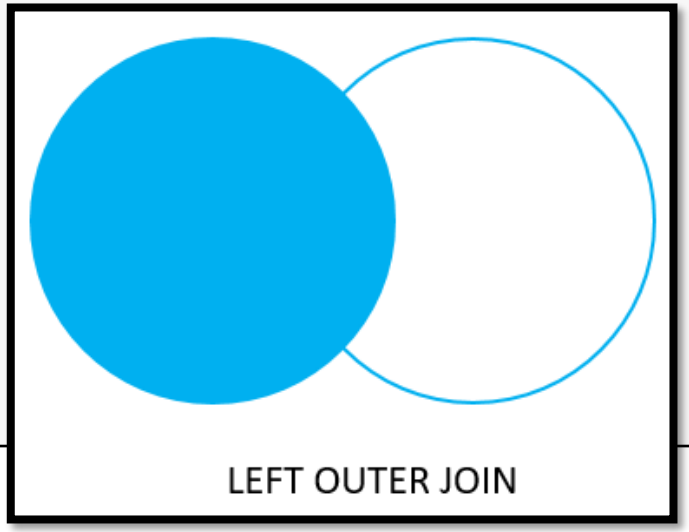
To join the table A with the table B table using a left join, you follow these steps:

- First, specify the columns in both tables from which you want to select data in the SELECT clause.
- Second, specify the left table (table A) in the FROM clause.
- Third, specify the right table (table B) in the LEFT JOIN clause and the join condition after the ON keyword.

Left Join

SYNTAX :

```
SELECT   pka,   c1,   pkb,   c2
FROM     A
Left JOIN B ON pka = fka;
```

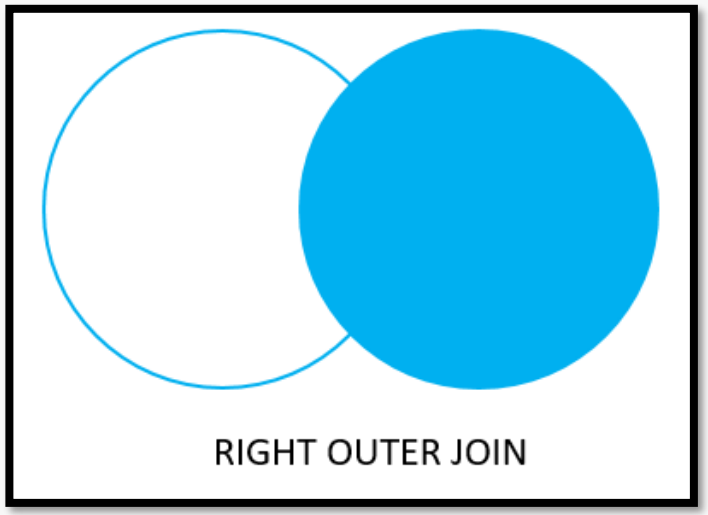


How the INNER JOIN works.

- ✓ The LEFT JOIN clause starts selecting data from the left table. For each row in the left table, it compares the value in the pka column with the value of each row in the fka column in the right table.
- ✓ If these values are equal, the left join clause creates a new row that contains columns that appear in the SELECT clause and adds this row to the result set.
- ✓ In case these values are not equal, the left join clause also creates a new row that contains columns that appear in the SELECT clause. In addition, it fills the columns that come from the right table with NULL.

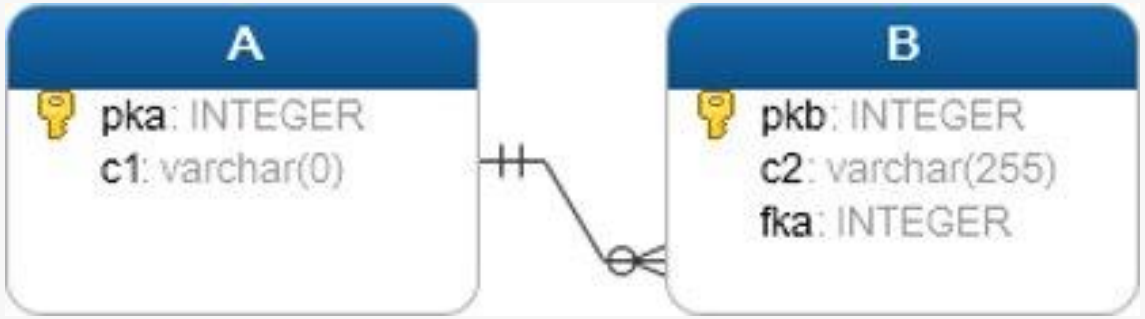
Right Join

To select data from the table B that may or may not have corresponding rows in the table A, you use the Right JOIN clause.



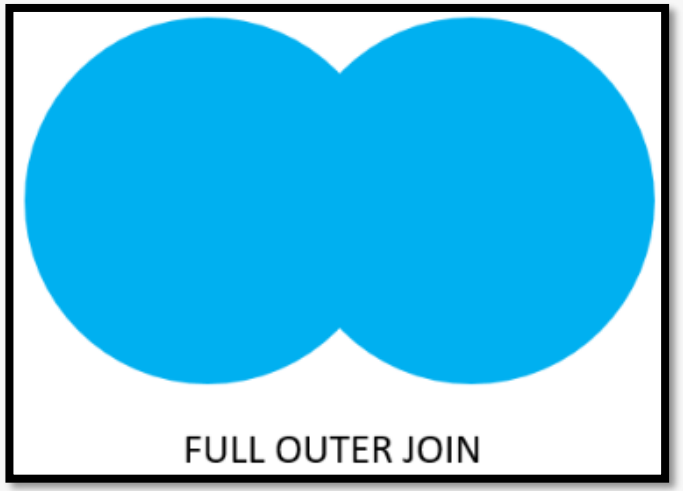
SYNTAX :

```
SELECT  pka,  c1,  pkb,  c2
FROM    A
RIGHTJOIN B ON pka = fka;
```



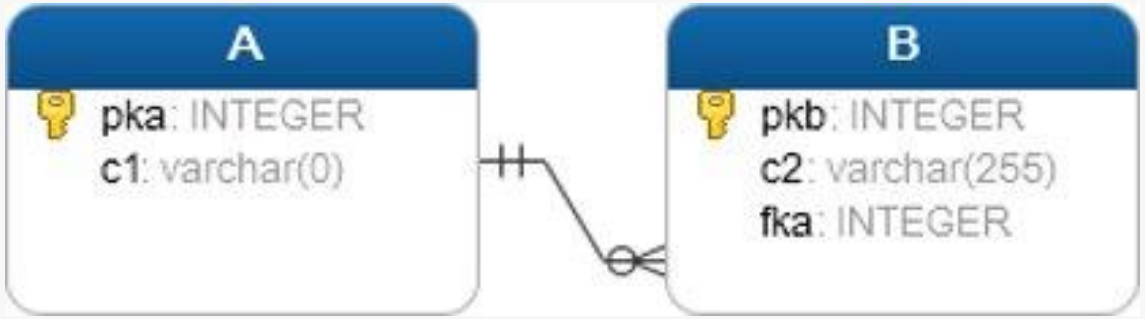
FULL OUTER JOIN

The full outer join combines the results of both left join and right join.



SYNTAX :

```
SELECT * FROM A
FULL [OUTER] JOIN B on A.id = B.id;
```



- ✓ If the rows in the joined table do not match, the full outer join sets NULL values for every column of the table that does not have the matching row.
- ✓ If a row from one table matches a row in another table, the result row will contain columns populated from columns of rows from both tables.

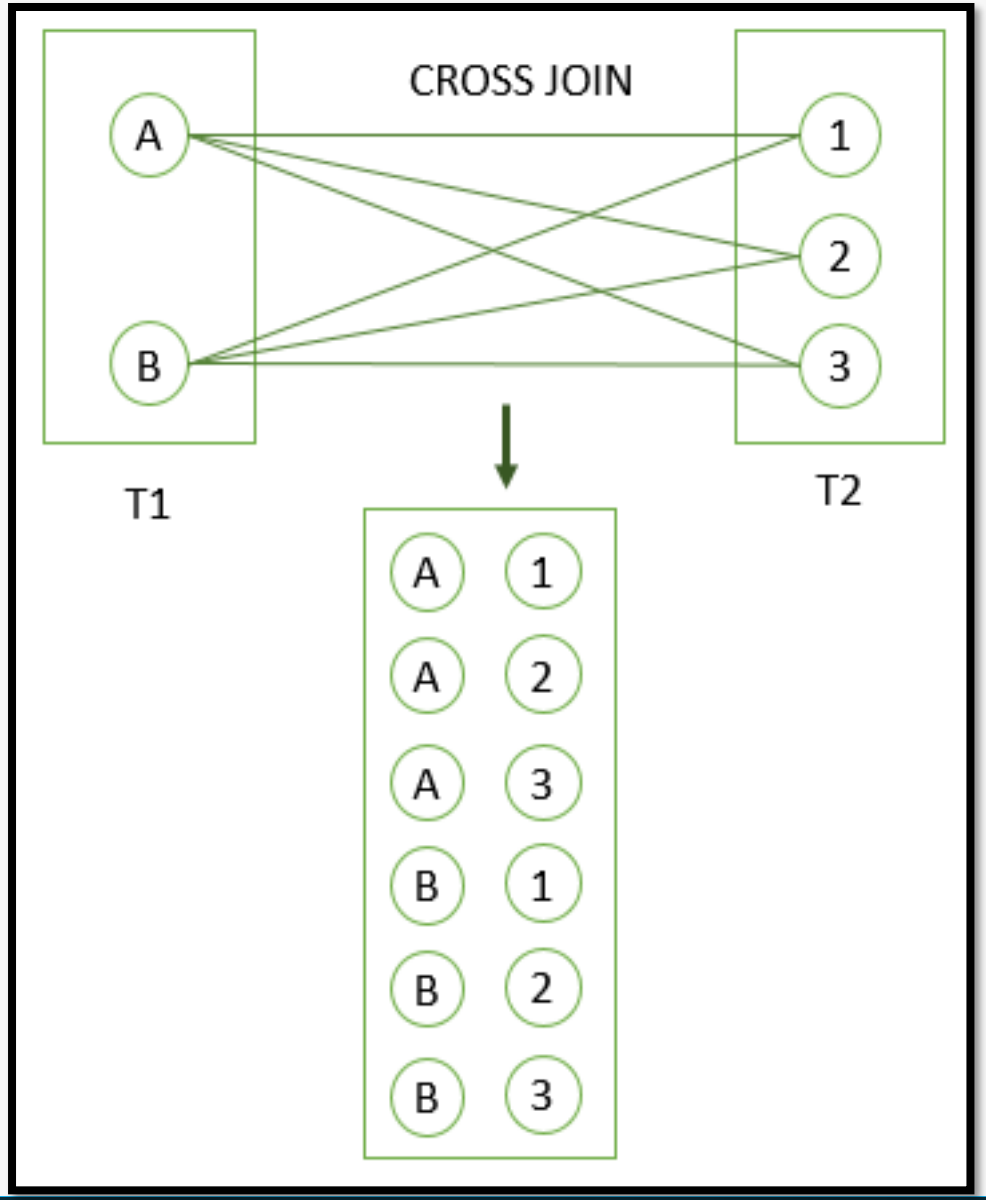
Cross JOIN

A CROSS JOIN clause allows you to produce a Cartesian Product of rows in two or more tables.

- Different from other join clauses such as LEFT JOIN or INNER JOIN, the `CROSS JOIN` clause does not have a join predicate.
- Suppose you have to perform a `CROSS JOIN` of two tables T1 and T2.
- If T1 has n rows and T2 has `m` rows, the result set will have `nxm` rows.
- For example, the T1 has `1,000` rows and T2 has `1,000` rows, the result set will have `1,000 x 1,000 = 1,000,000` rows.

SYNTAX :

```
SELECT select_list FROM T1
CROSS JOIN T2;
```



Natural JOIN

A natural join is a join that creates an implicit join based on the same column names in the joined tables.

- **A natural join can be an inner join, left join, or right join. If you do not specify a join explicitly e.g., INNER JOIN, LEFT JOIN, RIGHT JOIN,**
- **PostgreSQL will use the INNER JOIN by default.**

SYNTAX :

```
SELECT select_list
FROM T1
NATURAL [INNER, LEFT, RIGHT] JOIN T2;
```

If you use the asterisk (*) in the select list, the result will contain the following columns:

- ✓ **All the common columns, which are the columns from both tables that have the same name.**
- ✓ **Every column from both tables, which is not a common column.**

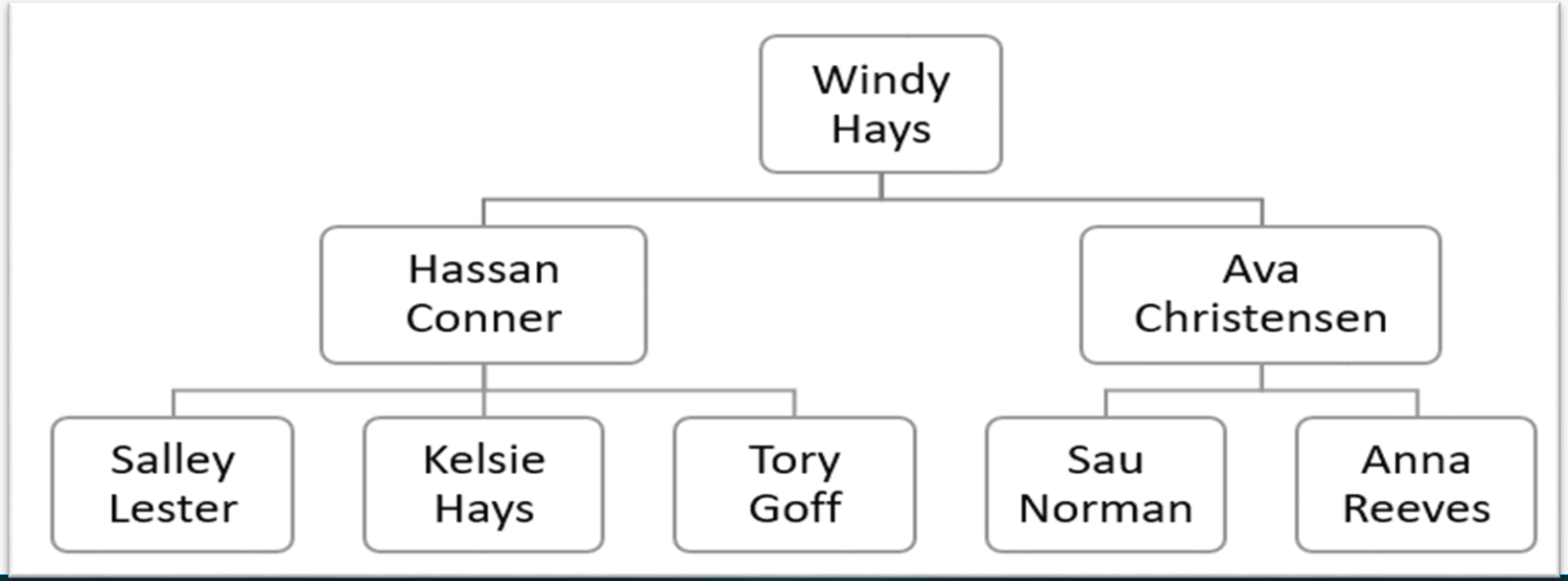
NOTE: If there are no attributes in common between two relations and you perform a natural join, it will return the cartesian product of the two relations.

Self JOIN

- ✓ A self-join is a regular join that joins a table to itself.
- ✓ In practice, you typically use a self-join to query hierarchical data or to compare rows within the same table.
- ✓ To form a self-join, you specify the same table twice with different table aliases and provide the join predicate after the ON keyword

```
SELECT select_list
FROM table_name t1
INNER JOIN table_name t2 ON join_predicate;
```

NOTE : you can use the INNER , LEFT JOIN or RIGHT JOIN clause to join table to itself like this:



Sub Query

- ✓ The query inside the brackets is called a subquery or an inner query.
- ✓ The query that contains the subquery is known as an outer query.

PostgreSQL executes the query that contains a subquery in the following sequence:

1. First, executes the subquery.
2. Second, gets the result and passes it to the outer query.
3. Third, executes the outer query.

- ➢ Without inner query, Outer query and not execute hence Outer query dependent on inner query
- ➢ Inner Query Execute not execute with another query hence this is not dependent in Outer query

NOTE : To construct a subquery, we put the second query in brackets and use it in the WHERE clause as an expression:

When & Why Sub-Query

Case 1 :- Whenever we come across unknown value we need to go for Subquery

Condition to Write :-

- From the inner Query we can select only one column
- The column name selected in inner Query and the column name

Dname of C

```
Select Dname
From Dept
Where Dno =(Select Dno
                From Emp
                where ename='Cleark');
```

When & Why Sub-Query

Case 2 : whenever the displayed from one table and the condition is executed From another table then we need to go for Sub query

Name	Dept no
Avi	20
Blake	10
Cleark	30
Demain	20

Dept no	Dname	Loc
10	IT	Bhopal
20	Operation	Indore
30	Developer	Bangalore
40	Full stack	Maysure

Type of Sub-Query

Single Row Sub Query :

- ✓ If the sub-Query returns single-row, then it is called single row Subquery.
- ✓ For the Single row sub-query, we can use normal Operator

Example : (= , != , <, > , <= ,>=)

Multi-Row Sub Query :

- ✓ If the subquery return multiple rows, then it is called Multiple row sub-query
- ✓ For Multi-Row Sub-Query, we cannot use normal Operator, but we can use special Operators

Example : IN , NOT IN , EXIST ,NOT EXIST , ANY ,ALL

Query

Question: Dname of employee if they were working as salesman and getting salary more then smith

```
Select  Dname
From dept
Where Deptno =(Select Deptno
                    from emp
                    where sal > (Select sal
                                  from emp
                                 where ename='smith'));
```

Question: Details of emp if they were working in sales dept and getting salary less then king

```
Select *
From emp
Where deptno In (select deptno
                  from dept
                   where dname='sales' And Sal< ( select sal
                                                  from emp
                                                  where ename= 'King'));
```

Query

Question: Dname of an employee in which atlist there two sales man are working

```
Select Dname
From dept
Where deptno in (select deptno
                 from emp
                  Where  job ='salesman'
                 group by dept no
                 having count (*)>2 );
```

Question: Location of an employee in that location atlist 5 emp were working

```
Select loc
From dept
Where dept no  in (select dept no
                   from emp
                   group by deptno
                   Having count (*)>4);
```

Query

Question: Details of an employee if the employee getting salary less then Adams

```
Select *
From emp
Where sal< (select sal
            from emp
            where ename ='adams');
```

Question: Details of an employee if the employee getting salary more then any of the salesman salary

```
Select *
From emp
Where sal Any (Select  max(sal)
               from emp
               where job ='salesman')
```


netlink
digital solutions
Thank you
www.netlink.com