

POSTGRESQL - JOINS

http://www.tutorialspoint.com/postgresql/postgresql_using_joins.htm

Copyright © tutorialspoint.com

The PostgreSQL **Joins** clause is used to combine records from two or more tables in a database. A JOIN is a means for combining fields from two tables by using values common to each.

Join Types in PostgreSQL are:

- The CROSS JOIN
- The INNER JOIN
- The LEFT OUTER JOIN
- The RIGHT OUTER JOIN
- The FULL OUTER JOIN

Before we proceed, let's consider two tables COMPANY and DEPARTMENT. We already have seen INSERT statements to populate COMPANY table. So just let's assume the list of records available in COMPANY table:

id	name	age	address	salary	join_date
1	Paul	32	California	20000	2001-07-13
3	Teddy	23	Norway	20000	
4	Mark	25	Rich-Mond	65000	2007-12-13
5	David	27	Texas	85000	2007-12-13
2	Allen	25	Texas		2007-12-13
8	Paul	24	Houston	20000	2005-07-13
9	James	44	Norway	5000	2005-07-13
10	James	45	Texas	5000	2005-07-13

Another table is DEPARTMENT, has the following definition:

```
CREATE TABLE DEPARTMENT(  
    ID INT PRIMARY KEY      NOT NULL,  
    DEPT CHAR(50) NOT NULL,  
    EMP_ID INT NOT NULL  
);
```

Here is the list of INSERT statements to populate DEPARTMENT table:

```
INSERT INTO DEPARTMENT (ID, DEPT, EMP_ID)  
VALUES (1, 'IT Billing', 1 );  
  
INSERT INTO DEPARTMENT (ID, DEPT, EMP_ID)  
VALUES (2, 'Engineering', 2 );  
  
INSERT INTO DEPARTMENT (ID, DEPT, EMP_ID)  
VALUES (3, 'Finance', 7 );
```

Finally, we have the following list of records available in DEPARTMENT table:

id	dept	emp_id
1	IT Billing	1
2	Engineering	2
3	Finance	7

The CROSS JOIN

A CROSS JOIN matches every row of the first table with every row of the second table. If the input tables have x and y columns, respectively, the resulting table will have x+y columns. Because CROSS JOINS have the potential to generate extremely large tables, care must be taken to only use them when appropriate.

Following is the syntax of CROSS JOIN:

```
SELECT ... FROM table1 CROSS JOIN table2 ...
```

Based on the above tables, we can write a CROSS JOIN as follows:

```
testdb=# SELECT EMP_ID, NAME, DEPT FROM COMPANY CROSS JOIN DEPARTMENT;
```

Above query will produce the following result:

emp_id	name	dept
1	Paul	IT Billing
1	Teddy	IT Billing
1	Mark	IT Billing
1	David	IT Billing
1	Allen	IT Billing
1	Paul	IT Billing
1	James	IT Billing
1	James	IT Billing
2	Paul	Engineering
2	Teddy	Engineering
2	Mark	Engineering
2	David	Engineering
2	Allen	Engineering
2	Paul	Engineering
2	James	Engineering
2	James	Engineering
7	Paul	Finance
7	Teddy	Finance
7	Mark	Finance
7	David	Finance
7	Allen	Finance
7	Paul	Finance
7	James	Finance
7	James	Finance

The INNER JOIN

A INNER JOIN creates a new result table by combining column values of two tables (table1 and table2) based upon the join-predicate. The query compares each row of table1 with each row of table2 to find all pairs of rows, which satisfy the join-predicate. When the join-predicate is satisfied, column values for each matched pair of rows of table1 and table2 are combined into a result row.

An INNER JOIN is the most common type of join and is the default type of join. You can use INNER keyword optionally.

Following is the syntax of INNER JOIN:

```
SELECT table1.column1, table2.column2...
FROM table1
INNER JOIN table2
ON table1.common_field = table2.common_field;
```

Based on the above tables, we can write an INNER JOIN as follows:

```
testdb=# SELECT EMP_ID, NAME, DEPT FROM COMPANY INNER JOIN DEPARTMENT
        ON COMPANY.ID = DEPARTMENT.EMP_ID;
```

Above query will produce the following result:

emp_id	name	dept
1	Paul	IT Billing
2	Allen	Engineering

The LEFT OUTER JOIN

The OUTER JOIN is an extension of the INNER JOIN. SQL standard defines three types of OUTER JOINS: LEFT, RIGHT, and FULL and PostgreSQL supports all of these.

In case of LEFT OUTER JOIN, an inner join is performed first. Then, for each row in table T1 that does not satisfy the join condition with any row in table T2, a joined row is added with null values in columns of T2. Thus, the joined table always has at least one row for each row in T1.

Following is the syntax of LEFT OUTER JOIN:

```
SELECT ... FROM table1 LEFT OUTER JOIN table2 ON conditional_expression ...
```

Based on the above tables, we can write a inner join as follows:

```
testdb=# SELECT EMP_ID, NAME, DEPT FROM COMPANY LEFT OUTER JOIN DEPARTMENT
         ON COMPANY.ID = DEPARTMENT.EMP_ID;
```

Above query will produce the following result:

emp_id	name	dept
1	Paul	IT Billing
2	Allen	Engineering
	James	
	David	
	Paul	
	Mark	
	Teddy	
	James	

The RIGHT OUTER JOIN

First, an inner join is performed. Then, for each row in table T2 that does not satisfy the join condition with any row in table T1, a joined row is added with null values in columns of T1. This is the converse of a left join; the result table will always have a row for each row in T2.

Following is the syntax of LEFT OUTER JOIN:

```
SELECT ... FROM table1 RIGHT OUTER JOIN table2 ON conditional_expression ...
```

Based on the above tables, we can write a inner join as follows:

```
testdb=# SELECT EMP_ID, NAME, DEPT FROM COMPANY RIGHT OUTER JOIN DEPARTMENT
         ON COMPANY.ID = DEPARTMENT.EMP_ID;
```

Above query will produce the following result:

emp_id	name	dept
1	Paul	IT Billing
2	Allen	Engineering
7		Finance

The FULL OUTER JOIN

First, an inner join is performed. Then, for each row in table T1 that does not satisfy the join condition with any row in table T2, a joined row is added with null values in columns of T2. Also, for each row of T2 that does not satisfy the join condition with any row in T1, a joined row with null values in the columns of T1 is added.

Following is the syntax of FULL OUTER JOIN:

```
SELECT ... FROM table1 FULL OUTER JOIN table2 ON conditional expression ...
```

Based on the above tables, we can write a inner join as follows:

```
testdb=# SELECT EMP_ID, NAME, DEPT FROM COMPANY FULL OUTER JOIN DEPARTMENT
        ON COMPANY.ID = DEPARTMENT.EMP_ID;
```

Above query will produce the following result:

emp_id	name	dept
1	Paul	IT Billing
2	Allen	Engineering
7		Finance
	James	
	David	
	Paul	
	Mark	
	Teddy	
	James	