



SQL Joins in PostgreSQL

Dr. Arrington

Overview

- An SQL join is one of the most popular types of query statements executed while handling relational databases
- There are five types of join operations:
 - Inner
 - Left
 - Right
 - Full Outer
 - Cross

Table Creation (1)

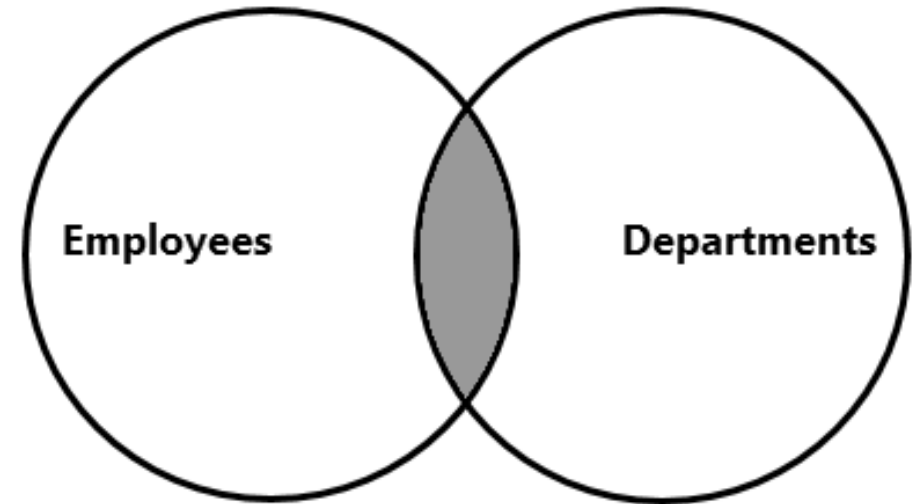
- To illustrate different types of joins, we will create ***Employees*** and ***Departments*** tables as following:
 - ***Employees*** table:
 - ***employee_id*** (int) / Primary Key
 - ***name*** (varchar)
 - ***dob*** (date)
 - ***department_id*** (int) (reference to departments table)
 - ***Departments*** table:
 - ***department_id*** (int) / Primary Key
 - ***department_name*** (varchar)





Table Creation (2)

- There are four departments defined within the ***Departments*** table:
 - Human resources (id: 1)
 - Development (id: 2)
 - Sales (id: 3)
 - Technical Support (id: 4)
- And there are five employees defined within the ***Employees*** table:
 - Alan Smith (Department: Human Resources)
 - Sultan Nader (Department: Human Resources)
 - Mohd Rasheed (Department: Development)
 - Brian Wallace (Department: Sales)
 - Peter Hilton (Not assigned to a department until now - *null*)

INNER JOIN

- Inner Joins select the rows where values are the mutual columns values are matched
- If we apply an inner join to the ***Employees-Departments*** example, it will only return the employees working within departments that have a row within the ***Departments*** table

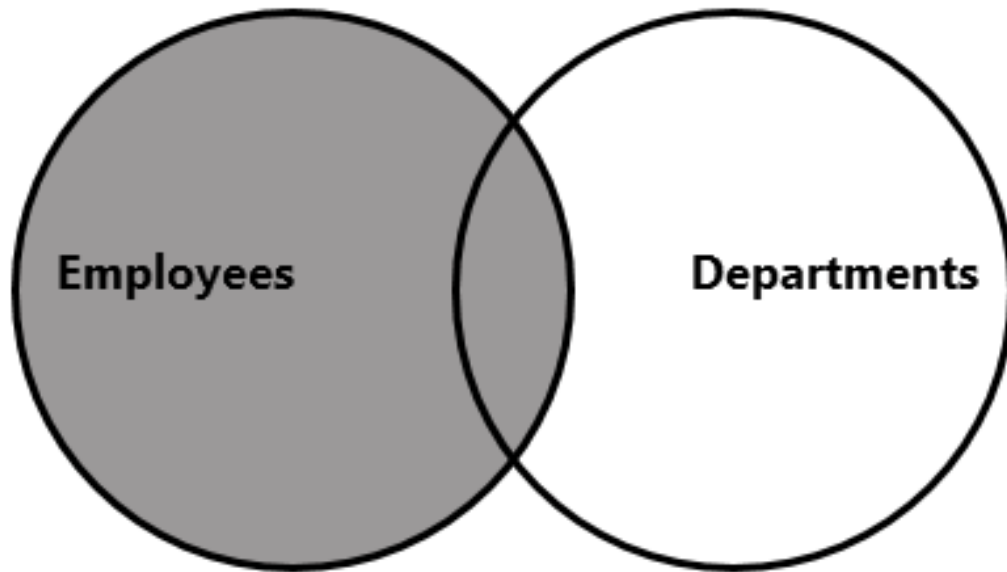


employee_id 	name 	dob 	department_name 
integer	character varying	date	character varying
1	Alan Smith	1989-01-01	Human Resources
2	Sultan Nader	1992-01-01	Human Resources
3	Mohd Rasheed	1999-01-01	Development
4	Brian Wallace	1979-01-01	Sales





INNER JOIN - Result

- From the screenshot, the fifth employee is not shown in the result since it is not assigned to any department
- Also, note that the **department_name** is retrieved instead of the **department_id**

LEFT JOIN



- LEFT JOINS are used to retrieve all rows from the first table mentioned in the FROM clause in addition to the matched rows from the second table
- In order to select all employees listed within the **Employees** table and to mention the department name if exists, we can use the following

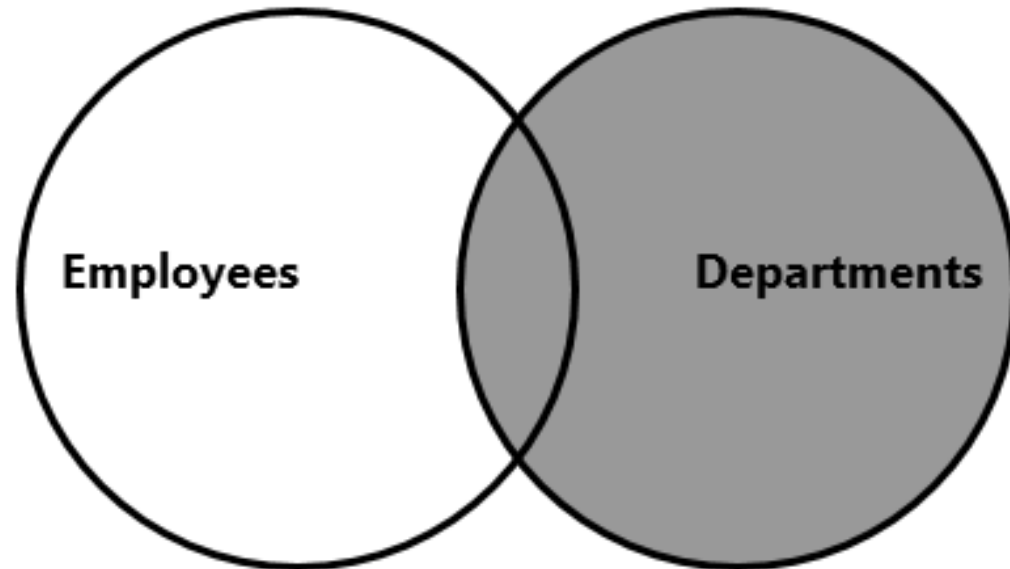
employee_id 	name 	dob 	department_name 
integer	character varying	date	character varying
1	Alan Smith	1989-01-01	Human Resources
2	Sultan Nader	1992-01-01	Human Resources
3	Mohd Rasheed	1999-01-01	Development
4	Brian Wallace	1979-01-01	Sales
5	Peter Hilton	1986-01-01	[null]

LEFT JOIN - Result

- The query result returns all five employees listed within the table with a NULL value in the **department_name** column in the fifth row since **Peter Hilton** is not assigned to any department yet

RIGHT JOIN

- RIGHT JOINS are very similar to LEFT JOINS since they return all rows from the table listed at the right of the JOIN operator with the matched values from the table listed at the left
- In order to select the employees that are working within departments, in addition to departments that does not have any employees, we can use the following

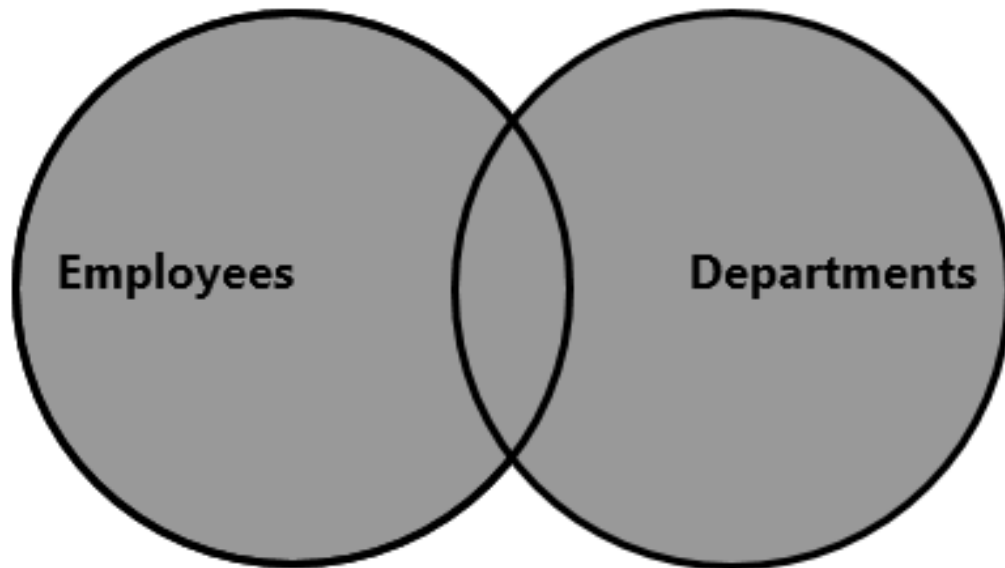


RIGHT JOIN - Result





- As shown in the screenshot, the query returned the same rows of the INNER JOIN query in addition to the **Technical Support** department that doesn't have any employees

employee_id integer	name character varying	dob date	department_name character varying
1	Alan Smith	1989-01-01	Human Resources
2	Sultan Nader	1992-01-01	Human Resources
3	Mohd Rasheed	1999-01-01	Development
4	Brian Wallace	1979-01-01	Sales
[null]	[null]	[null]	Technical Support

FULL OUTER JOIN



- FULL OUTER JOINs return all matched rows between both tables specified in the JOIN operation in addition to all unmatched rows from the first and second tables
- The Full Outer join query will return all employees working within departments plus all employees that are not assigned and all departments that doesn't contain any employee

employee_id 	name 	dob 	department_name 
integer	character varying	date	character varying
1	Alan Smith	1989-01-01	Human Resources
2	Sultan Nader	1992-01-01	Human Resources
3	Mohd Rasheed	1999-01-01	Development
4	Brian Wallace	1979-01-01	Sales
5	Peter Hilton	1986-01-01	[null]
[null]	[null]	[null]	Technical Support

FULL OUTER JOIN -
Result

- The screenshot displays **Peter Hilton** has no value in the **department_name** field, and **Technical support** department is shown with no employee information

CROSS JOIN

- CROSS JOINS are a bit different from the other Join operations. They are used to create a combination of two different sets without have mutual columns
- As an example, if we need to create a combination of all departments with all employees

employee_id integer	name character varying	dob date	department_name character varying
2	Sultan Nader	1992-01-01	Human Resources
3	Mohd Rasheed	1999-01-01	Human Resources
4	Brian Wallace	1979-01-01	Human Resources
5	Peter Hilton	1986-01-01	Human Resources
1	Alan Smith	1989-01-01	Development
2	Sultan Nader	1992-01-01	Development
3	Mohd Rasheed	1999-01-01	Development
4	Brian Wallace	1979-01-01	Development
5	Peter Hilton	1986-01-01	Development
1	Alan Smith	1989-01-01	Sales
2	Sultan Nader	1992-01-01	Sales
3	Mohd Rasheed	1999-01-01	Sales
4	Brian Wallace	1979-01-01	Sales
5	Peter Hilton	1986-01-01	Sales
1	Alan Smith	1989-01-01	Technical Support
2	Sultan Nader	1992-01-01	Technical Support
3	Mohd Rasheed	1999-01-01	Technical Support
4	Brian Wallace	1979-01-01	Technical Support
5	Peter Hilton	1986-01-01	Technical Support



Joins Review

- **Five Primary Types of Joins used with PostgreSQL:**
 - `INNER JOIN` returns records that have matching values in both tables.
 - `LEFT JOIN` returns all records from the left table and the matched records from the right table.
 - `RIGHT JOIN` returns all records from the right table and the matched records from the left table.
 - `CROSS JOIN` returns records that match every row of the left table with every row of the right table. This type of join has the potential to make very large tables.
 - `FULL OUTER JOIN` places null values within the columns that do not match between the two tables, after an inner join is performed.