

[www.devmedia.com.br](http://www.devmedia.com.br)

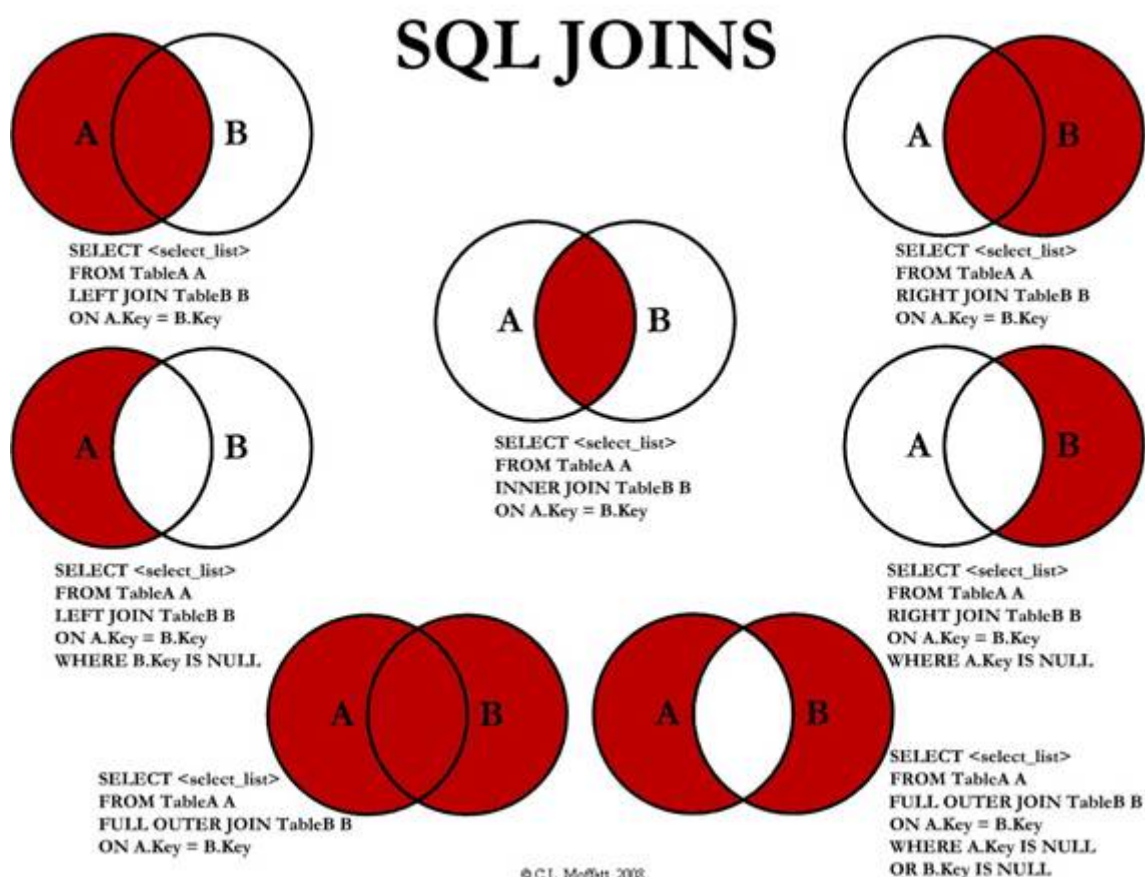
[versão para impressão]

Link original: <https://www.devmedia.com.br/sql-join-entenda-como-funciona-o-retorno-dos-dados/31006>

## SQL Join: Entenda como funciona o retorno dos dados

**Este artigo demonstrará de forma prática como funciona cada join do SQL e o que retornará de dados.**

Muitos desenvolvedores têm a dificuldade de saber qual resultado é retornado por cada join no SQL e, portanto, quando devem utilizar cada um. Para facilitar esse entendimento, a **Figura 1** traz uma representação gráfica, baseada na Teoria dos Conjuntos, muito conhecida na matemática. Nessa imagem, temos a representação de duas tabelas (A e B) e o resultado esperado por cada tipo de join (a área em vermelho representa os registros retornados pela consulta).



**Figura 1.** Representação gráfica dos joins

Nesse artigo analisaremos cada join individualmente, bom base em exemplos, e veremos seus resultados. Assim, podemos comparar seu funcionamento e decidir quando devemos usar cada um.

Saiba mais sobre: [Cláusula JOIN no SQL](#)

## Preparando o ambiente de testes

Para demonstrar o funcionamento dos métodos de junção (joins), precisaremos criar duas tabelas entre as quais deve haver algum relacionamento para que possamos "cruzar" os dados. Como o objetivo aqui não é concentrar em boas práticas, modelagem ou normalização, criaremos apenas duas tabelas contendo uma coluna Nome, que será comum entre elas. O script da **Listagem 1** cria essa estrutura.

```
CREATE TABLE TabelaA(  
    Nome varchar(50) NULL  
)
```

GO

```
CREATE TABLE TabelaB(  
    Nome varchar(50) NULL  
)
```

### Listagem 1. Criando as tabelas para teste

Em seguida, precisaremos adicionar nas tabelas recém criadas alguns dados que nos permitam colocar à prova as junções. Sendo assim, vamos inserir, com o script da **Listagem 2** alguns registros de forma que haja nomes que estão presentes apenas em uma tabela, e também nomes que sejam comuns às duas.

```
INSERT INTO TabelaA VALUES('Fernanda')  
INSERT INTO TabelaA VALUES('Josefa')  
INSERT INTO TabelaA VALUES('Luiz')  
INSERT INTO TabelaA VALUES('Fernando')
```

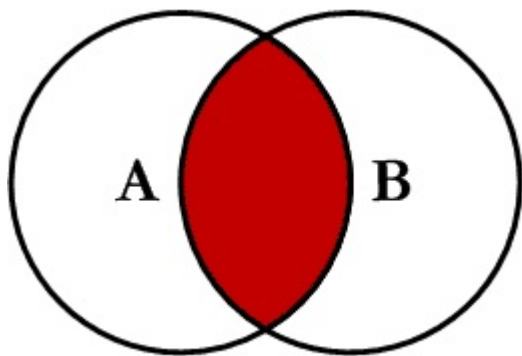
```
INSERT INTO TabelaB VALUES('Carlos')  
INSERT INTO TabelaB VALUES('Manoel')  
INSERT INTO TabelaB VALUES('Luiz')  
INSERT INTO TabelaB VALUES('Fernando')
```

### Listagem 2. Inserindo registros para testes

Saiba mais: [Como criar bancos e tabelas no SQL Server](https://www.devmedia.com.br/view/print.php?idcomp=31006)

## Inner Join

O Inner Join é o método de junção mais conhecido e, como ilustra a **Figura 2**, retorna os registros que são comuns às duas tabelas.

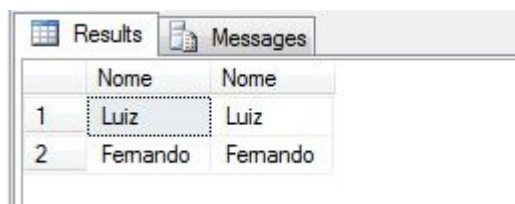


**Figura 2.** Representação do Inner Join

Na **Listagem 3** temos um exemplo de consulta com esse tipo de join, e na **Figura 3** podemos ver seu resultado.

```
SELECT a.Nome, b.Nome
FROM TabelaA as A
INNER JOIN TabelaB as B
        on a.Nome = b.Nome
```

**Listagem 3.** Usando o Inner Join

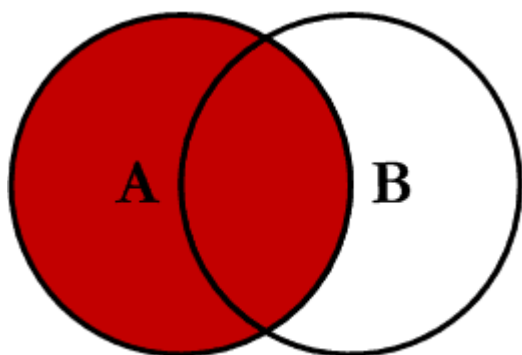


	Nome	Nome
1	Luiz	Luiz
2	Fernando	Fernando

**Figura 3.** Resultado do Inner Join

## Left Join

O Left Join, cujo funcionamento é ilustrado na **Figura 4**, tem como resultado todos os registros que estão na tabela A (mesmo que não estejam na tabela B) e os registros da tabela B que são comuns à tabela A.



**Figura 4.** Representação do Left Join

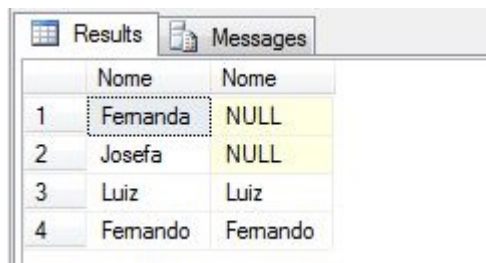
Para compreender melhor seu uso, temos um exemplo na **Listagem 4**, cujo resultado é apresentado em seguida na **Figura 5**.

```

SELECT a.Nome, b.Nome
FROM TabelaA as A
LEFT JOIN TabelaB as B
      on a.Nome = b.Nome

```

#### Listagem 4. Usando o Left Join



	Nome	Nome
1	Fernanda	NULL
2	Josefa	NULL
3	Luiz	Luiz
4	Fernando	Fernando

Figura 5. Resultado do Left Join

### Right Join

Usando o Right Join, conforme mostra a **Figura 6**, teremos como resultado todos os registros que estão na tabela B (mesmo que não estejam na tabela A) e os registros da tabela A que são comuns à tabela B.

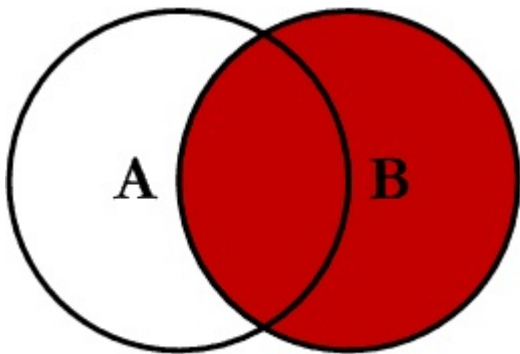


Figura 6. Representação do Right Join

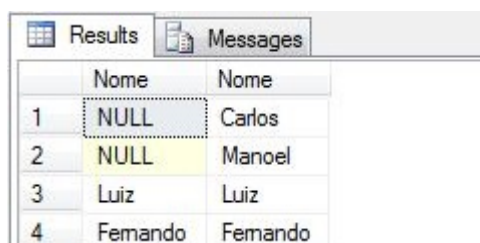
Na **Listagem 5** temos um exemplo desse tipo de consulta, e na **Figura 7** vemos seu resultado.

```

SELECT a.Nome, b.Nome
FROM TabelaA as A
RIGHT JOIN TabelaB as B
      on a.Nome = b.Nome

```

#### Listagem 5. Usando o Right Join

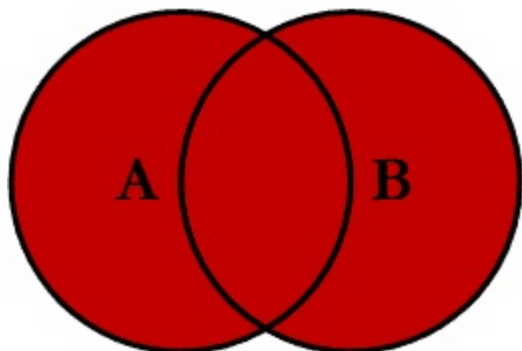


	Nome	Nome
1	NULL	Carlos
2	NULL	Manoel
3	Luiz	Luiz
4	Fernando	Fernando

Figura 7. Resultado do Right Join

## Outer Join

O Outer Join (também conhecido por Full Outer Join ou Full Join), conforme mostra a **Figura 8**, tem como resultado todos os registros que estão na tabela A e todos os registros da tabela B.



**Figura 8.** Representação do Outer Join

Para obter esse resultado, devemos executar a consulta seguindo a estrutura que é demonstrada na **Listagem 6**, e cujo retorno é apresentado na **Figura 9**.

```
SELECT a.Nome, b.Nome
FROM TabelaA as A
FULL OUTER JOIN TabelaB as B
on a.Nome = b.Nome
```

**Listagem 6.** Usando o Outer Join

	Nome	Nome
1	Femanda	NULL
2	Josefa	NULL
3	Luiz	Luiz
4	Fernando	Fernando
5	NULL	Carlos
6	NULL	Manoel

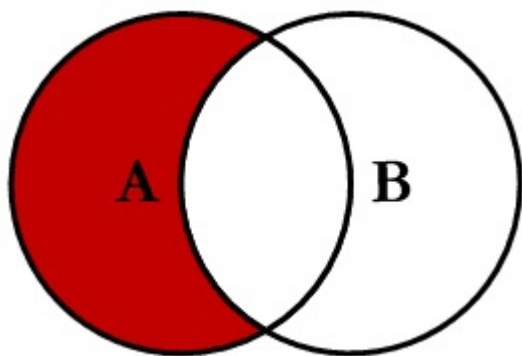
**Figura 9.** Resultado do Outer Join

Nesse código, a palavra reservada OUTER é opcional. Portanto, se a removermos, deixando apenas a expressão FULL JOIN, o resultado será o mesmo.

Saiba mais: [Como utilizar o Outer Join no Oracle](#)

## Left Excluding Join

Na **Figura 10** temos a representação gráfica do Left Excluding Join, que retorna como resultado todos os registros que estão na tabela A e que não estejam na tabela B.



**Figura 10.** Representação do Left Excluding Join

Os comandos desse join podem ser vistos na **Listagem 7**, e seu resultado é apresentado na **Figura 11**.

```
SELECT a.Nome, b.Nome
FROM TabelaA as A
LEFT JOIN TabelaB as B
      on a.Nome = b.Nome
WHERE b.Nome is null
```

**Listagem 7.** Usando Left Excluding Join

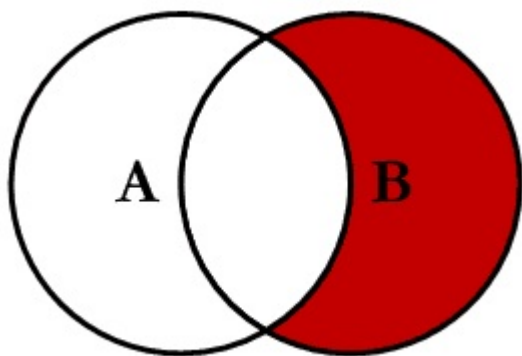
A screenshot of a database query result window. It has two tabs: 'Results' and 'Messages'. The 'Results' tab is active, showing a table with two columns: 'Nome' and 'Nome'. There are two rows of data. The first row has '1' in the first column and 'Femanda' in the second column. The second row has '2' in the first column and 'Josefa' in the second column. The 'Nome' column values are highlighted in yellow.

	Nome	Nome
1	Femanda	NULL
2	Josefa	NULL

**Figura 11.** Resultado do Left Excluding Join

## Right Excluding Join

O Right Excluding Join, como ilustra a **Figura 12**, retorna como resultado todos os registros que estão na tabela B e que não estejam na tabela A. Para vermos isso na prática, podemos executar os comandos da **Listagem 8**. Como resultado, teremos os mesmos registros apresentados na **Figura 13**.

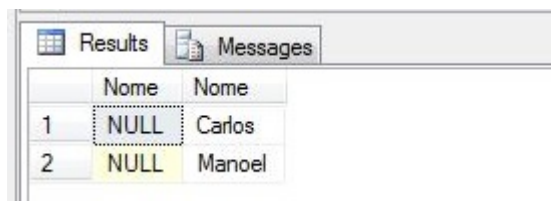


**Figura 12.** Representação do Right Excluding Join

```
SELECT a.Nome, b.Nome
FROM TabelaA as A
```

```
RIGHT JOIN TabelaB as B
    on a.Nome = b.Nome
WHERE a.Nome is null
```

### Listagem 8. Utilizando Right Excluding Join



	Nome	Nome
1	NULL	Carlos
2	NULL	Manoel

Figura 13. titulo

### Outer Excluding Join

Usando o Outer Excluding Join, conforme mostra a **Figura 14**, teremos como resultado todos os registros que estão na tabela B, mas que não estejam na tabela A, e todos os registros que estão na tabela A, mas que não estejam na tabela B.

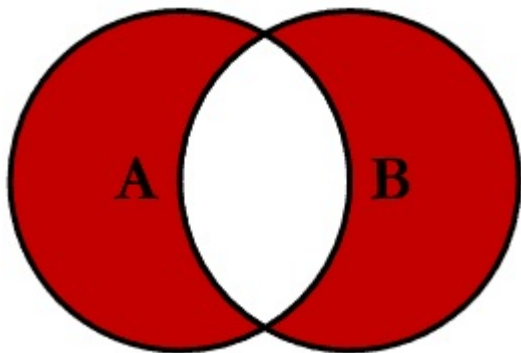


Figura 14. Representação do Outer Excluding Join

O código da **Listagem 9** mostra a sintaxe necessária para executar esse tipo de consulta. Nesse caso o termo OUTER é optativo e, se removido, deixando apenas FULL JOIN, o resultado será o mesmo.

```
SELECT a.Nome, b.Nome
FROM TabelaA as A
FULL OUTER JOIN TabelaB as B
    on a.Nome = b.Nome
WHERE a.Nome is null or b.Nome is null
```

### Listagem 9. Usando Outer Excluding Join

O resultado dessa consulta pode ser visto na **Figura 15**.



	Nome	Nome
1	Fernanda	NULL
2	Josefa	NULL
3	NULL	Carlos
4	NULL	Manoel

**Figura 15.** Resultado do Outer Excluding Join

As cláusulas JOIN na [linguagem SQL](#) são extremamente úteis e utilizadas com muita frequência, portanto, seu conhecimento é fundamental para quem trabalha com bancos de dados relacionais com o [SQL Server](#), o [MySQL](#), o [Firebird](#), etc.

#### Links:

- Saiba mais sobre o [uso de joins no Firebird](#).
- [Conceitos Fundamentais de banco de dados](#)
- [Administração de Banco de Dados com SQL Server](#)
- Curso relacionado: [Oracle](#)