Como usar instruções JOIN de forma eficaz

Nessa leitura, você revisará como as instruções JOIN são usadas e conhecerá alguns recursos que podem ser usados para aprender mais sobre elas. A JOIN combina tabelas através de uma chave primária ou estrangeira para alinhar as informações de ambas as tabelas no processo de combinação. Ela usa essas chaves para identificar as relações e os valores correspondentes nas tabelas.

Precisa relembrar seu conhecimento sobre chaves primárias e estrangeiras? Consulte o glossário

do curso ou volte ao tópico
Bancos de dados em data analytics

.

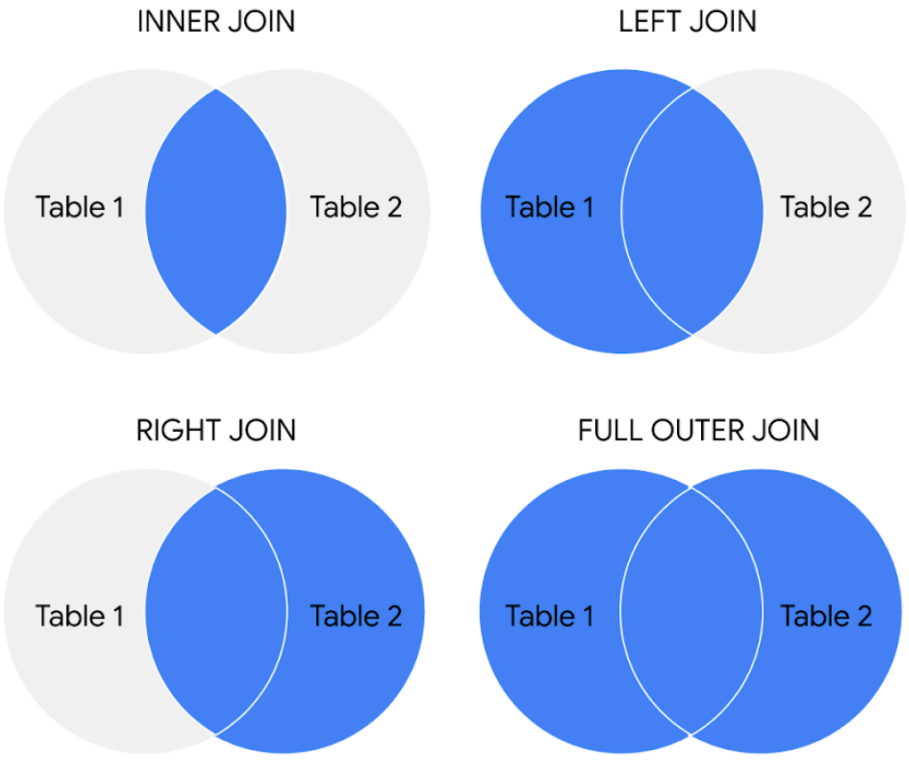
A sintaxe geral de JOIN

```
SELECT
   -- table columns from tables are inserted here
   table_name1.column_name
   table_name2.column_name
FROM
   table_name1
JOIN
   table_name2
ON table_name1.column_name = table_name2.column_name
```

Como é possível ver nesta sintaxe, a instrução JOIN é parte da cláusula FROM da consulta. No SQL, **JOIN** indica que você irá combinar os dados das duas tabelas. **ON**, no SQL,identifica a forma como as tabelas devem ser correspondidas com relação às informações corretas a serem combinadas a partir delas.

Tipo de instruções JOIN

Há quatro formas gerais de executar instruções JOIN em consultas do SQL: **INNER, LEFT, RIGHT e FULL OUTER.**



The circles represent left and right tables, and where they are joined is highlighted in blue
Os círculos representam as tabelas esquerda e direita, e o ponto em que estão unidas está destacado em azul

Veja o que essas diferentes consultas JOIN fazem.

INNER JOIN

INNER é *opcional* nessa consulta do SQL, pois é a operação JOIN padrão e a mais comumente usada. Ela pode aparecer simplesmente como JOIN. INNER JOIN retorna registros se os dados aparecem em ambas as tabelas. Se você usar, por exemplo, INNER JOIN para as tabelas "customers" e "orders" e corresponder os dados usando a chave "customer_id", você combinará os dados para cada "customer_id" que aparece nas tabelas. Se há uma "customer_id" na tabela "customers", mas não na "orders", os dados de "customer_id" não são combinados ou retornados pela consulta.

```
SELECT
   customers.customer_name,
   orders.product_id,
   orders.ship_date
FROM
   customers
INNER JOIN
   orders
ON customers.customer_id = orders.customer_id
```

Os resultados da consulta podem parecer com o abaixo, em que " customer_name" pertence à tabela "customers" e "product_id" e "ship_date" pertencem à tabela “orders":

customer_name	product_id	ship_date
Martin's Ice Cream	43998	23/02/2021
Beachside Treats	872012	25/02/2021
Mona's Natural Flavors	724956	28/02/2021
... etc.	... etc.	... etc.

Os dados das duas tabelas foram combinados ao corresponder a chave "customer_id" em comum em ambas. Observe que "customer_id" não aparece nos resultados da consulta; ele é simplesmente usado para definir a relação entre os dados das duas tabelas, para que os dados possam ser combinados e retornados.

LEFT JOIN

Também pode aparecer como **LEFT OUTER JOIN**, mas a maioria dos usuários prefere LEFT JOIN. Todas as duas sintaxes estão corretas. A LEFT JOIN retorna todos os registros da tabela à esquerda e somente os registros correspondentes da tabela à direita. Use-a sempre que precisar dos dados da primeira tabela completa e dos valores da segunda tabela, se houver. Na consulta abaixo, por exemplo, LEFT JOIN retornará "customer_name" com o "sales_rep" correspondente, se disponível. Se houver um cliente que não interagiu com um representante de vendas, esse cliente ainda assim aparecerá nos resultados da consulta, porém com um valor **NULL** para “rep_vendas".

```
SELECT
   customers.customer_name,
   sales.sales_rep
FROM
   customers
LEFT JOIN
   sales
ON customers.customer_id = sales.customer_id
```

Os resultados da consulta podem parecer com o abaixo, em que "customer_name" pertence à tabela "customers" e "sales_rep" pertence à tabela "sales". Mais uma vez, os dados das duas tabelas foram combinados ao corresponder o "customer_id" em comum nas duas, mesmo que "customer_id" não tenha retornado nos resultados da consulta.

customer_name	rep_vendas
Martin's Ice Cream	Luis Reyes
Beachside Treats	NULL
Mona's Natural Flavors	Geri Hall
...etc.	...etc.

RIGHT JOIN

Pode aparecer como RIGHT OUTER JOIN ou RIGHT JOIN. A RIGHT JOIN retorna todos os registros da tabela à direita e os registros correspondentes da tabela à esquerda. Na prática, RIGHT JOIN é raramente usada. A maioria das pessoas simplesmente trocam as tabelas e continuam com a LEFT JOIN. Usando o exemplo anterior da LEFT JOIN, a consulta que usa RIGHT JOIN retornaria o seguinte:

```
SELECT
   sales.sales_rep,
   customers.customer_name
FROM
  sales
RIGHT JOIN
  customers
ON sales.customer_id = customers.customer_id
```

Os resultados da consulta são os mesmos do exemplo anterior.

customer_name	**rep_vendas**
Martin's Ice Cream	Luis Reyes
Beachside Treats	NULL
Mona's Natural Flavors	Geri Hall
...etc.	...etc.

FULL OUTER JOIN

Por vezes, aparece como FULL JOIN. A FULL OUTER JOIN retorna todos os registros das tabelas especificadas. Você pode combinar as tabelas dessa forma, mas não se esqueça de que isso pode levar, consequentemente, a uma grande extração de dados. A FULL OUTER JOIN retorna todos os registros das *duas* tabelas, mesmo que os dados não estejam preenchidos em uma delas. Na consulta abaixo, por exemplo, você obterá todos os clientes e as datas de envio dos produtos. Como está usando uma FULL OUTER JOIN, pode ser que clientes retornem sem as datas de envio correspondentes ou então que as datas de envio retornem sem os clientes correspondentes. Se não houver dados correspondentes em nenhuma tabela, será retornado um valor NULL (nulo).

```
SELECT
   customers.customer_name,
   orders.ship_date
FROM
   customers
FULL OUTER JOIN
   orders
ON customers.customer_id = orders.customer_id
```

Os resultados da consulta podem parecer com os mostrados a seguir.

customer_name	ship_date
Martin's Ice Cream	2021-02-23
Beachside Treats	2021-02-25
NULL	2021-02-25
The Daily Scoop	NULL
Mountain Ice Cream	NULL
Mona's Natural Flavors	2021-02-28
...etc.	...etc.

Para mais informações
As instruções JOIN serão úteis ao se trabalhar com bancos de dados relacionais e com o SQL, sem contar que você terá inúmeras oportunidades para praticá-las. Confira alguns outros recursos que trazem mais informações sobre as instruções JOIN e como usá-las:

- **JOIN no SQL**

  : É uma ótima explicação básica das instruções JOIN com exemplos. Precisa relembrar seu conhecimento sobre o que diferentes JOIN fazem? Esse é um excelente recurso para favoritar e consultar quando quiser.
- **Instruções JOIN de banco de dados - Introdução aos tipos e conceitos de JOIN**

  : É uma introdução bem completa sobre as instruções JOIN. Esse artigo não apenas explica o que são as instruções JOIN e como usá-las, como também aborda de forma detalhada as diferentes situações de quando e por que usar as instruções JOIN. É um ótimo recurso se tiver interesse em saber mais sobre a lógica por trás do uso da JOIN.
- **Tipos de JOIN SQL explicados em imagens**

  : Esse recurso, que traz uma representação visual das diferentes instruções JOIN, é uma forma muito útil de refletir sobre as instruções JOIN se você for do tipo que aprende com recursos visuais, além de ser uma forma bem bacana de se lembrar dos diferentes tipos.
- **JOIN no SQL: como reunir dados com uma JOIN por vez**

  : Além de contar com uma explicação detalhada sobre as instruções JOIN com exemplos, esse recurso traz exemplos de dados que você pode usar para acompanhar o guia passo a passo. É uma forma útil de praticar as instruções JOIN com dados concretos.
- **JOIN no SQL:**

  outro recurso que traz uma explicação clara sobre as instruções JOIN e usa exemplos para demonstrar como elas funcionam na prática. Esses exemplos também combinam as instruções JOIN com a atribuição de alias. É uma ótima oportunidade de ver como as instruções JOIN podem ser combinadas com outros conceitos do SQL que você já aprendeu durante o curso.