Tipos de JOIN

DigitalHouse >
Coding School

Certified Tech Developer
The Ultimate Degree

Índice

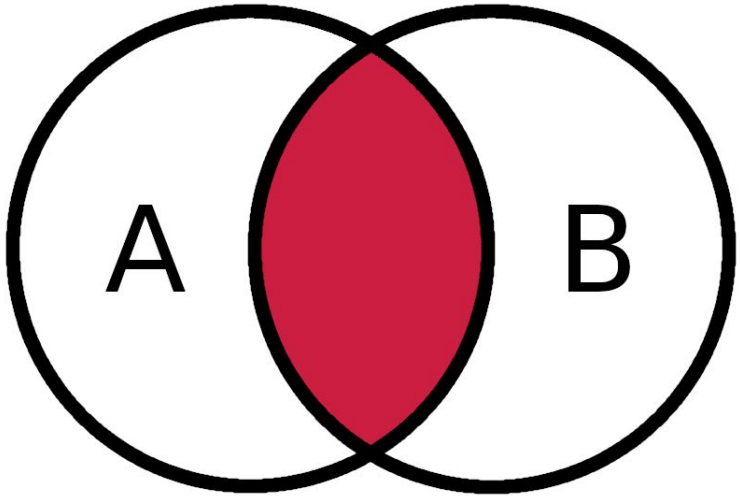
1 | INNER

INNER JOIN

El **INNER JOIN** entre dos tablas devuelve únicamente los registros que cumplen la condición indicada en la cláusula **ON**.

SQL

```sql
SELECT columna1, columna2, ...
FROM tabla A
INNER JOIN tabla B
ON condicion
```

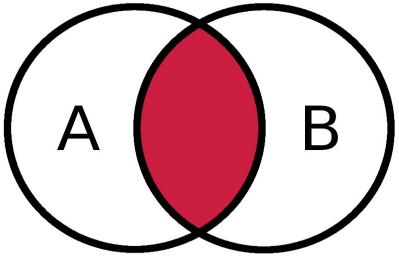


INNER JOIN

El **INNER JOIN** es la opción predeterminada y nos devuelve **todos los registros** donde **se cruzan dos o más tablas**. Por ejemplo, si tenemos una tabla cliente y otra factura, al cruzarlas con **INNER JOIN**, nos devuelve aquellos registros o filas donde haya un valor coincidente en ambas tablas.

cliente		
id	**nombre**	**apellido**
1	Juan	Perez
2	Clara	Sanchez
3	Marta	García

INNER JOIN



factura		
id	**cliente_id**	**fecha**
11	2	12/09/2019
12	**null**	20/09/2019
13	1	24/09/2019

INNER JOIN

El ejemplo anterior, se podría construir de la siguiente manera:

```sql
SELECT factura.id AS nro_factura, apellido, nombre, fecha
FROM cliente
INNER JOIN factura
ON cliente.id = factura.cliente_id;
```

A continuación, se muestran los datos obtenidos:

nro_factura	apellido	nombre	fecha
11	Sanchez	Clara	12/09/2019
13	Perez	Juan	24/09/2019

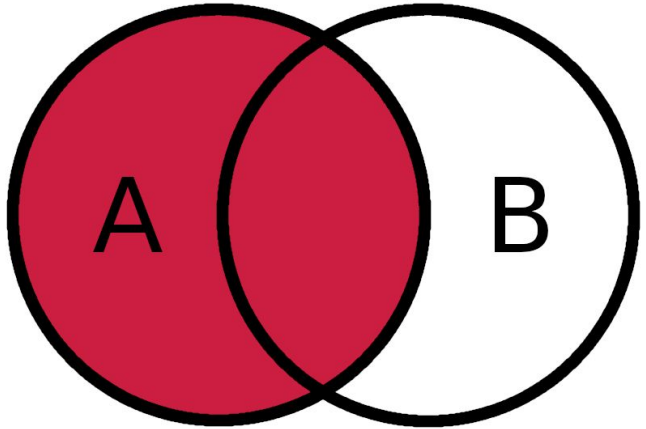
2 | LEFT

LEFT JOIN

El **LEFT JOIN** entre dos tablas devuelve todos los registros de la primera tabla (en este caso sería la tabla A), incluso cuando los registros no cumplan la condición indicada en la cláusula **ON**.

SQL

```sql
SELECT columna1, columna2, ...
FROM tabla A
LEFT JOIN tabla B
ON condicion
```

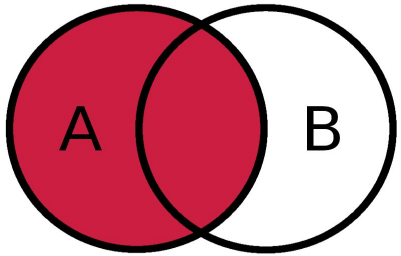


LEFT JOIN

Entonces, **LEFT JOIN** nos devuelve **todos** los registros donde se **cruzan dos o más tablas**. Incluso los registros de una primera tabla (A) que **no cumplan** con la condición indicada en la cláusula **ON**. Por ejemplo, si tenemos una tabla cliente y otra factura, al cruzarlas, nos devuelve aquellos registros donde haya un valor coincidente entre ambas, más los registros de aquellos clientes que no tengan una factura asignada.

cliente		
id	**nombre**	**apellido**
1	Juan	Perez
2	Clara	Sanchez
3	Marta	García

LEFT JOIN



factura		
id	**cliente_id**	**fecha**
11	2	12/09/2019
12	**null**	20/09/2019
13	1	24/09/2019

LEFT JOIN

El ejemplo anterior, se podría construir de la siguiente manera:

```sql
SELECT factura.id AS nro_factura, apellido, nombre, fecha
FROM cliente
LEFT JOIN factura
ON cliente.id = factura.cliente_id;
```

A continuación, se muestran los datos obtenidos:

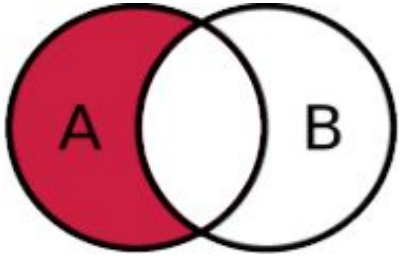
nro_factura	apellido	nombre	fecha
11	Sanchez	Clara	12/09/2019
13	Perez	Juan	24/09/2019
null	García	Marta	**null**

LEFT Excluding JOIN

Este tipo de **LEFT JOIN** nos devuelve únicamente los **registros** de una primera tabla (A), excluyendo los registros que cumplan con la condición indicada en la cláusula **ON**. Por ejemplo, si tenemos una tabla cliente y otra factura, al cruzarlas, nos devuelve solo aquellos registros de clientes que no tengan una factura asignada.

cliente		
id	**nombre**	**apellido**
1	Juan	Perez
2	Clara	Sanchez
3	Marta	García

LEFT Excluding JOIN



factura		
id	**cliente_id**	**fecha**
11	2	12/09/2019
12	**null**	20/09/2019
13	1	24/09/2019

LEFT Excluding JOIN

Continuando con el ejemplo, se podría construir de la siguiente manera:

```SQL
SELECT factura.id AS nro_factura, apellido, nombre, fecha
FROM cliente
LEFT JOIN factura
ON cliente.id = factura.cliente_id
WHERE ISNULL(factura.id);
```

A continuación, se muestran los datos obtenidos:

nro_factura	apellido	nombre	fecha
null	García	Marta	**null**

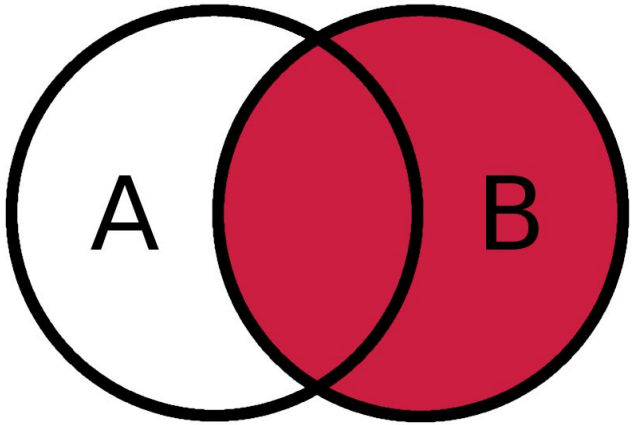
3 | RIGHT

RIGHT JOIN

El **RIGHT JOIN** entre dos tablas devuelve todos los registros de la segunda tabla, incluso cuando los registros no cumplan la condición indicada en la cláusula **ON**.

SQL

```sql
SELECT columna1, columna2, ...
FROM tabla A
RIGHT JOIN tabla B
ON condicion
```

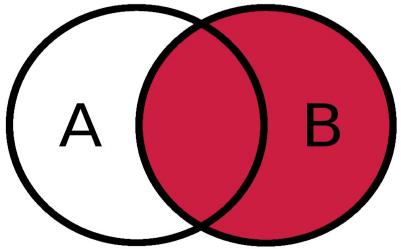


RIGHT JOIN

Entonces, **RIGHT JOIN** nos devuelve **todos** los registros donde se **cruzan dos o más tablas**. Incluso los registros de una segunda tabla (B) que **no cumplan** con la condición indicada en la cláusula **ON**. Por ejemplo, si tenemos una tabla cliente y otra factura, al cruzarlas, nos devuelve aquellos registros donde haya un valor coincidente entre ambas, más los registros de aquellas facturas que no tengan un cliente asignado.

cliente		
id	**nombre**	**apellido**
1	Juan	Perez
2	Clara	Sanchez
3	Marta	García

RIGHT JOIN



factura		
id	**cliente_id**	**fecha**
11	2	12/09/2019
12	**null**	20/09/2019
13	1	24/09/2019

RIGHT JOIN

El ejemplo anterior, se podría construir de la siguiente manera:

```sql
SELECT factura.id AS nro_factura, apellido, nombre, fecha
FROM cliente
RIGHT JOIN factura
ON cliente.id = factura.cliente_id;
```

A continuación, se muestran los datos obtenidos:

nro_factura	apellido	nombre	fecha
11	Sanchez	Clara	12/09/2019
12	**null**	**null**	20/09/2019
13	Perez	Juan	24/09/2019

RIGHT Excluding JOIN

Este tipo de **RIGTH JOIN** nos devuelve únicamente los registros de una segunda tabla (B), **excluyendo** los registros que cumplan con la condición indicada en la cláusula **ON**. Por ejemplo, si tenemos una tabla cliente y otra factura, al cruzarlas, nos devuelve solo aquellos registros de facturas que no tengan asignado un cliente.

cliente		
id	**nombre**	**apellido**
1	Juan	Perez
2	Clara	Sanchez
3	Marta	García

RIGHT Excluding JOIN

factura		
id	**cliente_id**	**fecha**
11	2	12/09/2019
12	**null**	20/09/2019
13	1	24/09/2019

RIGHT Excluding JOIN

Continuando con el ejemplo, se podría construir de la siguiente manera:

```SQL
SELECT factura.id AS nro_factura, apellido, nombre, fecha
FROM cliente
RIGHT JOIN factura
ON cliente.id = factura.cliente_id
WHERE ISNULL(cliente.id);
```

A continuación, se muestran los datos obtenidos:

nro_factura	apellido	nombre	fecha
12	**null**	**null**	20/09/2019

4 Experimentando con LEFT y RIGHT

¿Qué pasa si **intercambiamos** de lugar las tablas o si cambiamos LEFT por **RIGHT**?

LEFT JOIN -> RIGHT JOIN

Experimentemos con el último ejemplo que vimos.

```sql
SELECT factura.id AS factura, apellido
FROM cliente
LEFT JOIN factura
ON factura.cliente_id = cliente.id;
```

```sql
SELECT factura.id AS factura, apellido
FROM cliente
RIGHT JOIN factura
ON factura.cliente_id = cliente.id;
```

Comparemos los resultados:

factura	apellido
11	Sanchez
13	Perez
null	García

factura	apellido
11	Sanchez
12	**null**
13	Perez

Intercambiando tablas

Ahora, intercambiamos el lugar de las tablas implicadas.

```sql
SELECT factura.id AS factura, apellido
FROM factura
LEFT JOIN cliente
ON factura.cliente_id = cliente.id;
```

```sql
SELECT factura.id AS factura, apellido
FROM factura
RIGHT JOIN cliente
ON factura.cliente_id = cliente.id;
```

Comparemos los resultados:

factura	apellido
11	Sanchez
12	**null**
13	Perez

factura	apellido
11	Sanchez
13	Perez
null	García


DigitalHouse>
Coding School