

Otros tipos de *JOIN*

Cláusula *JOIN*

Recuerda que la base sintáctica de un *SELECT* es:

```
SELECT campos FROM tabla WHERE condición
```

En ella, es posible distinguir tres partes:

1. **Campos:** columnas a devolver.
2. **Tabla:** la o las tablas en las que se encuentran los datos.
3. **Condición:** condiciones que deben cumplirse al momento de efectuar la consulta.



Hasta ahora hemos utilizado solo una tabla en cada consulta. A continuación, veremos cómo se hace para **combinar datos que provienen de más de una tabla**.

Cuando se quieren consultar campos que se encuentran en **distintas tablas**, es necesario combinarlas a partir de la cláusula **JOIN** mediante un campo en común.

La **cantidad de joins** es igual a la cantidad de **tablas** que participan en la consulta -1.

Sintaxis alternativas

En la siguiente diapositiva, podrás ver dos maneras posibles de combinar tablas mediante la cláusula **JOIN**, para aquellos casos en los que el campo en común no tiene el **mismo nombre**.




Las sintaxis son las siguientes:

```
SELECT tabla1.campos, tabla2.campos  
FROM tabla1  
JOIN tabla2 ON tabla1.campo1=tabla2.campo1  
WHERE condición
```

```
SELECT tabla1.campos, tabla2.campos  
FROM tabla1, tabla2  
WHERE tabla1.campo1=tabla2.campo1  
AND condición
```



Notas:

- **Ambas sintaxis son válidas:** puedes utilizar el código con el cual te sientas más cómodo.
 - Hay que tener en cuenta que en ambos casos, si los campos por los cuales se combinan las tablas tienen el **mismo nombre**, hay que escribirlos en el **SELECT** como **“tabla.campo”** para especificar a qué tabla pertenece el campo.
 - En todos los tipos de **JOIN**, al referirse a la **tabla de la izquierda**, estamos hablando de la tabla especificada en el **FROM**, y la tabla de la derecha es la tabla definida a continuación del **JOIN**.
 - En las sintaxis 1 y 2, la **tabla de la izquierda** es **TABLA1** y la de la derecha, **TABLA2**.
- 

Otros tipos de *JOIN*

Los mismos pueden ser:

- ***LEFT [OUTER] JOIN***
- ***RIGHT [OUTER] JOIN***
- ***CROSS JOIN***

En las diapositivas siguientes, explicaremos cada ***JOIN*** con un ejemplo, a partir de las tablas que vemos a la derecha:

Tabla 1:

Codigo (int)	Nombre (varchar(15))
1	A
3	C
8	H

Tabla 2:

Codigo (int)	Nombre2 (varchar(15))
3	Tres
5	Cinco
8	Ocho

LEFT [OUTER] JOIN

Este **JOIN** devuelve todos los registros de la tabla de la izquierda y los registros que coinciden de la tabla de la derecha:

```
SELECT *  
FROM tabla1  
LEFT JOIN tabla2  
ON tabla1.codigo = tabla2.codigo;
```

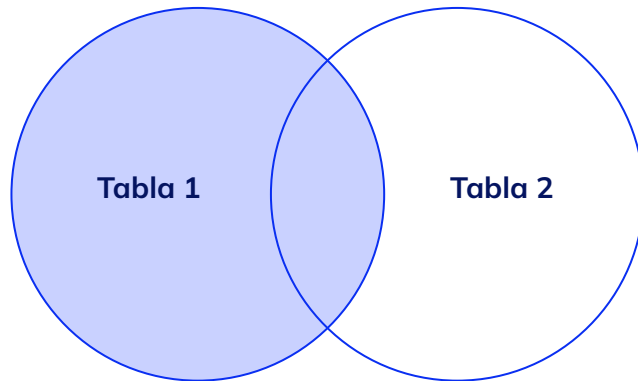
Se obtiene como resultado de la combinación de la **Tabla 1** y la **Tabla 2** los siguientes registros:

1	A	
3	C	Tres
8	H	Ocho

Esta consulta también puede escribirse de la siguiente manera:

```
SELECT * FROM tabla1  
LEFT OUTER JOIN tabla2  
ON tabla1.codigo = tabla2.codigo;
```

Basándonos nuevamente en los diagramas de Venn, podemos expresar el resultado del **LEFT [OUTER] JOIN** del siguiente modo:



RIGHT [OUTER] JOIN

Este **JOIN** devuelve todos los registros de la tabla de la derecha y los registros que coinciden de la tabla de la izquierda:

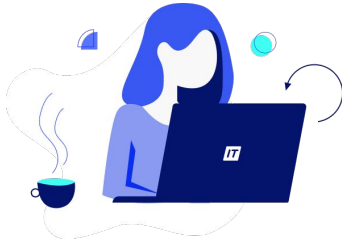
```
SELECT *  
FROM tabla1  
RIGHT JOIN tabla2  
ON tabla1.codigo = tabla2.codigo;
```

Se obtiene como resultado de la combinación de la **Tabla 1** y la **Tabla 2** los siguientes registros:

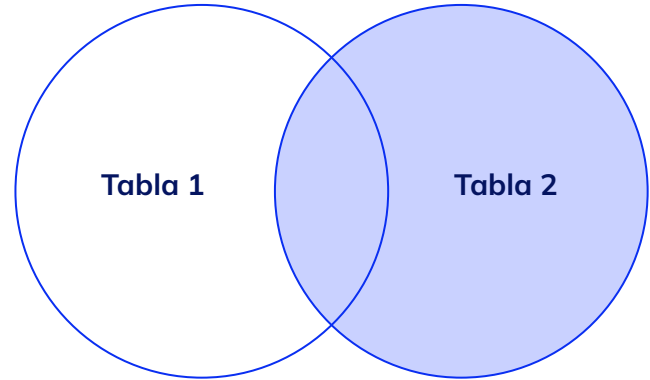
3	C	Tres
5		Cinco
8	H	Ocho

Esta consulta también puede escribirse de la siguiente manera:

```
SELECT * FROM tabla1  
RIGHT OUTER JOIN tabla2  
ON tabla1.codigo = tabla1.codigo;
```



Si seguimos con la analogía de los diagramas de Venn, podemos expresar el resultado del **RIGHT [OUTER] JOIN** del siguiente modo:



CROSS JOIN

Combina cada registro de la tabla de la izquierda con cada registro de la tabla de la derecha, **sin hacer coincidir un campo en particular**.

```
SELECT * FROM tabla1 CROSS JOIN tabla2;
```

Nota: este tipo de **JOIN** brinda la posibilidad de cruzar todos los registros con todos (*producto cartesiano*) y resulta **imposible de dibujar** con un diagrama de Venn. Se puede observar que representa un **JOIN muy poco eficiente** de ejecutar en tablas grandes.



Se obtiene como resultado de la combinación de la **Tabla 1** y la **Tabla 2** los siguientes registros:

1	A	3	Tres
1	A	5	Cinco
1	A	8	Ocho
3	C	3	Tres
3	C	5	Cinco
3	C	8	Ocho
8	H	3	Tres
8	H	5	Cinco
8	H	8	Ocho



JOINS ideados para la IA

Prompt 1:

Hacer una *query* para una base de datos en MySQL que muestre todos los productos que fueron pedidos por el cliente cuyo identificador sea igual a 11.

Prompt 2:

Hacer una *query* para una base de datos en MySQL donde muestre todos los clientes que hayan hecho una compra el último año.

Prompt 3:

Hacer una *query* para una base de datos en MySQL que muestre todos los productos que tienen ventas por más de \$5000 c/u.



**¡Sigamos
trabajando!**