SESIÓN 21 - Consultas multitabla: JOINs y subconsultas

OBJETIVOS

- Entender qué hace SQL internamente al ejecutar un JOIN.
- Saber cuándo usar INNER, LEFT o RIGHT.
- Comprender el impacto de NULL.
- Diferenciar ON y WHERE.

Cómo piensa SQL cuando hace un JOIN

Cuando ejecutamos:

FROM A
JOIN B
ON condición

SQL hace esto mentalmente:

1. Toma una fila de A.
2. Busca filas en B que cumplan la condición.
3. Une las coincidencias.
4. Decide si conserva o descarta según el tipo de JOIN.

Dataset Base

CLIENTES

id_cliente	nombre	ciudad
1	Ana	Madrid
2	Luis	Sevilla
3	Marta	Valencia
4	Juan	Madrid

PEDIDOS

id_pedido	id_cliente	total	fecha
101	1	300	2025-01-10
102	1	150	2025-01-14

103	2	400	2025-01-20
104	5	900	2025-01-21

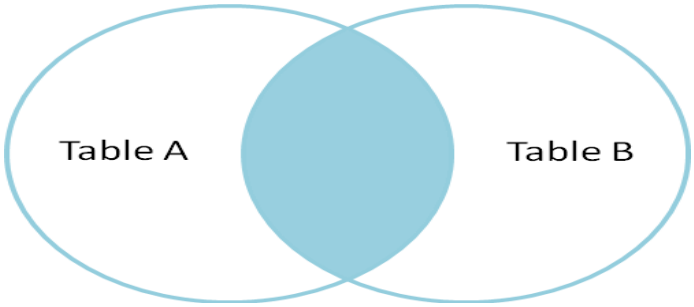
Importante:

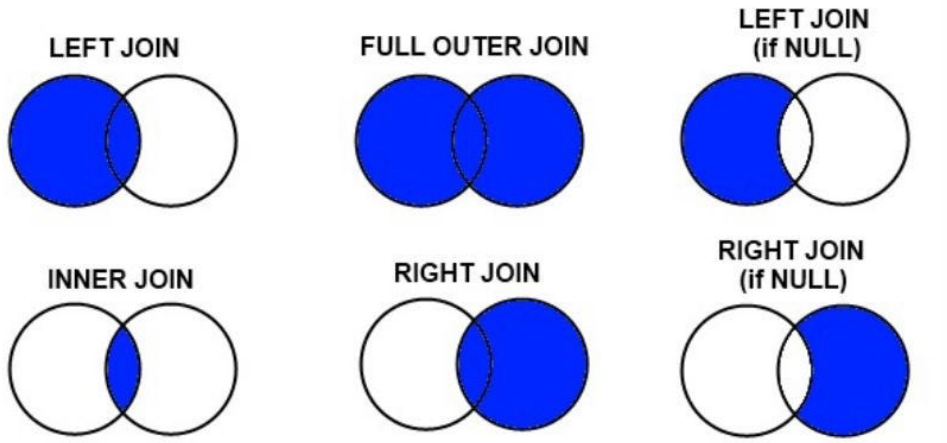
- Marta y Juan no tienen pedidos.
- Existe un pedido con cliente inexistente.

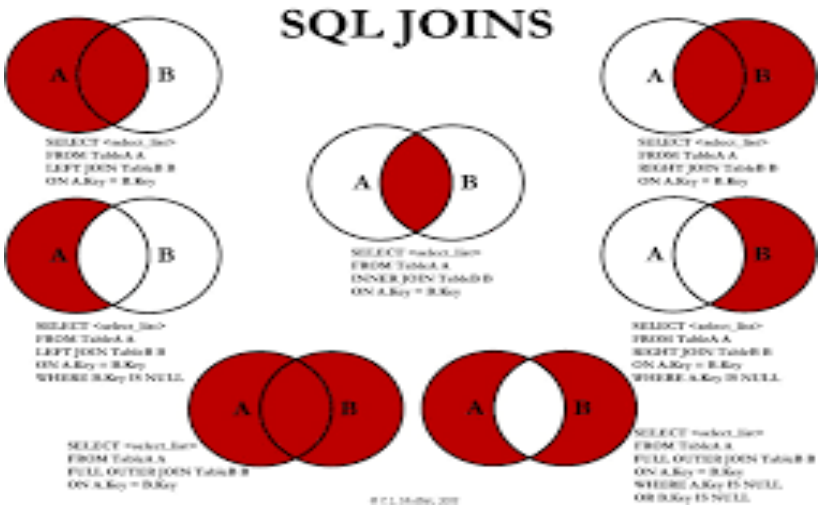
INNER JOIN

INNER JOIN devuelve únicamente filas que cumplen la condición en ambas tablas.

- INNER JOIN siempre devuelve coincidencias.
- Lo que cambia es cómo se construye o se interpreta la unión.







Es la forma correcta y profesional.

SELECT columnas
FROM tabla1
INNER JOIN tabla2
ON tabla1.campo = tabla2.campo;

Ejemplo:

```
SELECT C.nombre, P.total
FROM Clientes C
INNER JOIN Pedidos P
ON C.id_cliente = P.id_cliente;
```

INNER JOIN implícito (forma antigua)

```
SELECT C.nombre, P.total
FROM Clientes C, Pedidos P
WHERE C.id_cliente = P.id_cliente;
```

¿Qué diferencia hay?
Internamente hace lo mismo, pero:

- Es más fácil equivocarse.
- Puede generar producto cartesiano si olvidan la condición.
- No es recomendable en código profesional.

El problema que resuelve un JOIN
En bases de datos reales, la información está separada.

Tabla CLIENTES

id_cliente	nombre
1	Ana
2	Luis
3	Marta
4	Juan

Tabla PEDIDOS

id_pedido	id_cliente	total
101	1	300
102	1	150
103	2	400
104	5	900

Observa:

- Marta y Juan no tienen pedidos.
- Existe un pedido con cliente inexistente (id_cliente = 5).

¿Qué es un JOIN?

Un JOIN:
Une filas de dos tablas usando una condición.

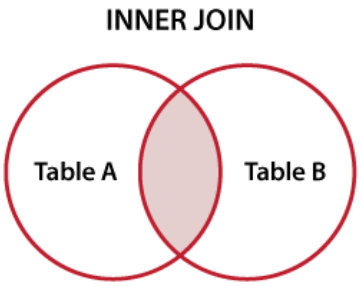
Siempre hay:

- Tabla izquierda (la que aparece en FROM).
- Tabla derecha (la que aparece después de JOIN).
- Condición (ON).

INNER JOIN

INNER JOIN devuelve:

- Solo las filas que coinciden en ambas tablas.



Solo la intersección.

```
SELECT C.nombre, P.total
FROM Clientes C
INNER JOIN Pedidos P
ON C.id_cliente = P.id_cliente;
```

Resultado

nombre	total
Ana	300
Ana	150
Luis	400

Qué NO aparece

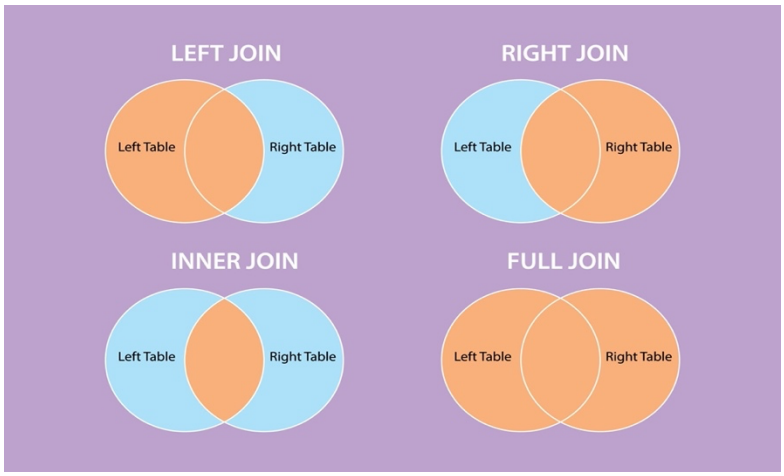
- Marta (no tiene pedido)
- Juan (no tiene pedido)
- Pedido 104 (no tiene cliente)

Si no hay coincidencia en ambas tablas : no aparece.

LEFT JOIN

LEFT JOIN devuelve:

- TODAS las filas de la tabla izquierda
- Y las coincidencias de la derecha
- Si no hay coincidencia: NULL



Todo el círculo izquierdo.

Consulta

```
SELECT C.nombre, P.total
FROM Clientes C
LEFT JOIN Pedidos P
ON C.id_cliente = P.id_cliente;
```

Resultado

nombre	total
Ana	300
Ana	150
Luis	400
Marta	NULL
Juan	NULL

Concepto fundamental

NULL significa:

- No existe dato relacionado.

Uso típico

- Detectar clientes sin pedidos:

```
SELECT C.nombre
FROM Clientes C
LEFT JOIN Pedidos P
```

```
ON C.id_cliente = P.id_cliente
WHERE P.id_pedido IS NULL;
```

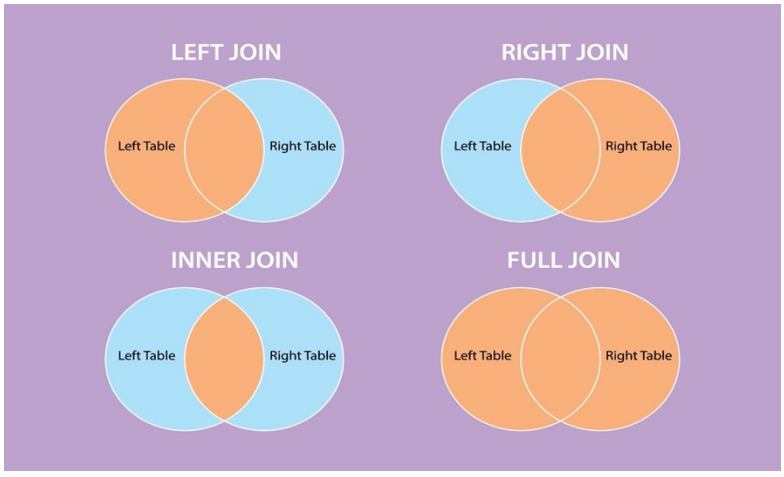
Resultado:

nombre
Marta
Juan

RIGHT JOIN

RIGHT JOIN devuelve:

- TODAS las filas de la tabla derecha
- Coincidencias de la izquierda
- Si no hay coincidencia: NULL



Todo el círculo derecho.

Consulta

```
SELECT C.nombre, P.id_pedido
FROM Clientes C
```

```
RIGHT JOIN Pedidos P
ON C.id_cliente = P.id_cliente;
```

Resultado

nombre	id_pedido
Ana	101
Ana	102
Luis	103
NULL	104

Uso típico

- Detectar pedidos huérfanos:

```
SELECT P.id_pedido
FROM Clientes C
RIGHT JOIN Pedidos P
ON C.id_cliente = P.id_cliente
WHERE C.id_cliente IS NULL;
```

Resultado:

id_pedido
104

COMPARACIÓN

Tipo	Conserva siempre	¿Puede mostrar NULL?	¿Qué elimina?
INNER	Solo coincidencias	No	Todo lo que no coincide
LEFT	Toda la izquierda	Sí (derecha)	Coincidencias inexistentes derecha se ponen NULL
RIGHT	Toda la derecha	Sí (izquierda)	Coincidencias inexistentes izquierda se ponen NULL

Comparación con nuestro ejemplo

Clientes:
Ana : Si
Luis: No
Marta: No
Juan: No

Pedidos:
101 : Si
102 : Si
103 : Si
104: No

Entonces:

- INNER: Ana y Luis
- LEFT: Ana, Luis, Marta, Juan
- RIGHT: 101,102,103,104

Error común: Confundir ON y WHERE

- ON: une tablas
- WHERE: filtra resultado final

INNER JOIN con múltiples condiciones

También existe INNER JOIN con más de una condición.

```
SELECT C.nombre, P.total
FROM Clientes C
INNER JOIN Pedidos P
ON C.id_cliente = P.id_cliente
AND P.total > 200;
```

Aquí el INNER JOIN solo considera pedidos mayores de 200.
Esto es importante porque:

- La condición está en el ON
- Afecta a la unión

INNER JOIN con múltiples tablas

INNER JOIN puede encadenarse.

Si añadimos tabla PRODUCTOS:

```
SELECT C.nombre, PR.nombre_producto, P.total
FROM Clientes C
INNER JOIN Pedidos P
ON C.id_cliente = P.id_cliente
INNER JOIN Productos PR
ON P.id_producto = PR.id_producto;
```

Aquí hay dos INNER JOIN consecutivos.
SQL va uniendo paso a paso:
Clientes + Pedidos → resultado intermedio
Resultado intermedio + Productos → resultado final

INNER JOIN con alias obligatorios (buenas prácticas)

Cuando se usan varias tablas, es obligatorio usar alias:

```
SELECT C.nombre, P.total
FROM Clientes AS C
INNER JOIN Pedidos AS P
ON C.id_cliente = P.id_cliente;
```

Evita ambigüedades si ambas tablas tienen columna con el mismo nombre.

INNER JOIN vs LEFT JOIN con condición en WHERE

```
SELECT C.nombre, P.total
FROM Clientes C
LEFT JOIN Pedidos P
ON C.id_cliente = P.id_cliente
WHERE P.total > 200;
```

Esto se comporta como un INNER JOIN.

¿Por qué?
Porque el WHERE elimina los NULL.

Ejemplo 1: Nombre y total

```
SELECT C.nombre, P.total
FROM Clientes C
INNER JOIN Pedidos P
ON C.id_cliente = P.id_cliente;
```

Resultado:

nombre	total
Ana	300
Ana	150
Luis	400

¿Por qué no aparece Marta?
¿Por qué no aparece el pedido 104?
Si responden correctamente, lo han entendido.

Ejemplo 2: INNER JOIN con filtro

```
SELECT C.nombre, P.total
FROM Clientes C
INNER JOIN Pedidos P
ON C.id_cliente = P.id_cliente
WHERE P.total > 200;
```

Resultado:

nombre	total
Ana	300
Luis	400

Aquí introducimos:

- ON une
- WHERE filtra después

Ejemplo 3: INNER JOIN con varias condiciones

```
SELECT C.nombre, P.total
FROM Clientes C
INNER JOIN Pedidos P
ON C.id_cliente = P.id_cliente
AND P.total >= 300;
```

Diferencia sutil pero importante.

LEFT JOIN

LEFT JOIN devuelve:

- Todas las filas de la izquierda.
- Coincidencias de la derecha.
- Si no hay coincidencia → NULL.

Ejemplo 1 : Ver todos los clientes

```
SELECT C.nombre, P.total
FROM Clientes C
LEFT JOIN Pedidos P
ON C.id_cliente = P.id_cliente;
```

Resultado:

nombre	total
Ana	300
Ana	150
Luis	400
Marta	NULL
Juan	NULL

Entender NULL

NULL significa:

- No existe dato relacionado.
- No es cero.
- No es cadena vacía.

Ejemplo 2: Detectar clientes sin pedidos

```
SELECT C.nombre
FROM Clientes C
LEFT JOIN Pedidos P
ON C.id_cliente = P.id_cliente
WHERE P.id_pedido IS NULL;
```

Resultado:

nombre
Marta
Juan

Diferencia: ON vs WHERE

Caso A

```
SELECT C.nombre, P.total
FROM Clientes C
LEFT JOIN Pedidos P
ON C.id_cliente = P.id_cliente
WHERE P.total > 200;
```

Aquí Marta y Juan desaparecen.
¿Por qué?

- Porque WHERE elimina los NULL.

Caso B

```
SELECT C.nombre, P.total
FROM Clientes C
LEFT JOIN Pedidos P
```

```
ON C.id_cliente = P.id_cliente
AND P.total > 200;
```

Aquí Marta y Juan siguen apareciendo.

- Esto es una diferencia avanzada y fundamental.

RIGHT JOIN

Devuelve todas las filas de la tabla derecha.

Ejemplo – Ver todos los pedidos

```
SELECT C.nombre, P.id_pedido
FROM Clientes C
RIGHT JOIN Pedidos P
ON C.id_cliente = P.id_cliente;
```

Resultado:

nombre	id_pedido
Ana	101
Ana	102
Luis	103
NULL	104

Detectar pedidos huérfanos

```
SELECT P.id_pedido
FROM Clientes C
RIGHT JOIN Pedidos P
ON C.id_cliente = P.id_cliente
WHERE C.id_cliente IS NULL;
```

Resultado:

id_pedido
104

TABLA COMPARATIVA

JOIN	Conserva siempre	Puede mostrar NULL	Uso típico
INNER	Solo coincidencias	No	Consultas normales
LEFT	Toda tabla izquierda	Sí (derecha)	Detectar ausencia
RIGHT	Toda tabla derecha	Sí (izquierda)	Detectar registros huérfanos