Practicar con el JOIN múltiple

Una vez vistos los distintos tipos de JOIN veamos ahora cómo combinar más de dos tablas.

Antes de empezar conviene indicar que cuando me refiero a cualquier JOIN me refiero a todos excepto el CROSS JOIN (este lo llamaremos producto cartesiano).

Al igual que podemos combinar varias operaciones matemáticas para formar una expresión más compleja y utilizar los paréntesis para delimitar las operaciones, podemos combinar varios JOINs en una misma cláusula FROM utilizando o no paréntesis. Yo os aconsejo utilizarlos para, por lo menos, evitar errores sintácticos.

Sabemos que cada JOIN produce una tabla, pues a su vez esta tabla puede formar parte de otro JOIN, también tenemos que recordar que la sintaxis de cualquier JOIN es :

<tabla> <JOIN> <tabla> ON condición_combi

Donde <tabla> puede ser el nombre de una tabla pero también otro JOIN completo (con sus dos tablas y el ON), si este JOIN lo encierras entre paréntesis puede que lo veas mejor.
<JOIN> puede ser cualquier tipo de JOIN: {INNER|LEFT|RIGHT|FULL} JOIN
Condición_combi es la condición de combinación de las dos <tablas>

Si queremos combinar 3 tablas, lo mejor es combinar primero dos, ver el resultado como una nueva tabla y añadir a esta tabla las filas de la tercera tabla.

Por ejemplo, queremos obtener los pedidos junto con el nombre del cliente que realizó el pedido y el nombre del representante que lo tomó.

Aquí parece que “manda” el pedido, pues empezamos por esa tabla, la cogemos y le añadimos los datos del cliente (para poder sacar el nombre del cliente):

Pedidos INNER JOIN Clientes ON pedidos.clie=Clientes.numclie

Ya tenemos una tabla con los pedidos con su cliente correspondiente, ahora nos falta el nombre del representante, como lo tenemos en la tabla Empleados, el resultado del JOIN lo tengo que combinar con la tabla Empleados, por lo que lo pongo entre paréntesis:

(Pedidos INNER JOIN Clientes ON pedidos.clie=Clientes.numclie)

y esta tabla la combino con Empleados siguiendo el formato anterior
-<tabla><JOIN><tabla>ON condición_combi -:

(Pedidos INNER JOIN Clientes ON pedidos.clie=Clientes.numclie) INNER JOIN Empleados ON Pedidos.rep=Empleados.numemp

Si además queremos saber la descripción del producto que se ha vendido, tendremos que añadir la tabla Productos siguiendo el mismo proceso:

((Pedidos INNER JOIN Clientes ON pedidos.clie=Clientes.numclie) INNER JOIN Empleados ON Pedidos.rep=Empleados.numemp) INNER JOIN Productos ON pedidos.fab=productos.idfab AND pedidos.producto= Productos.idproducto

Una vez tenemos los JOINs montados nos podemos preguntar por el tipo de JOIN adecuado.

En este caso lo único que nos interesan son los pedidos con sus datos (no nos interesan ni los clientes que no tienen pedidos ni los empleados que no tienen pedidos ni los productos que no aparecen en ningún pedido) y como no puede haber pedidos sin cliente asignado, sin representante ni sin producto, dejamos el INNER en todos los JOINS.

1.- Modifica la tabla de Clientes para que la columna repclie no admita nulos.

2.- Ahora queremos saber de cada pedido el nombre del cliente y el nombre del representante asignado al cliente:

Primero juntamos los pedidos con sus clientes y luego añadimos los representantes (son empleados) asignados a cada cliente.

```
SELECT *
FROM (Pedidos INNER JOIN Clientes ON pedidos.clie=Clientes.numclie) INNER
JOIN Empleados ON Clientes.repclie=Empleados.numemp
```

Recuerda que quien me dice el representante asignado a un cliente es repclie, luego puede que el pedido lo haya tomado otro representante (el que aparece en rep).

No pueden haber pedidos sin cliente, no nos interesan los clientes que no tienen pedidos → dejamos INNER en el primer JOIN.

No pueden haber clientes sin representante asignado (acabamos de poner el campo a no nulo) ni los empleados que no están asignados a ningún cliente, pues dejamos INNER.

3.- Obtener una lista de las oficinas con todos los empleados asignados a cada oficina y los pedidos realizados por estos empleados. Las oficinas que no tienen empleados no interesan.

Empezamos por las oficinas con sus empleados.

```
SELECT *
FROM Oficinas INNER JOIN empleados ON Oficinas.oficina = empleados.oficina;
```

Añadimos los pedidos de los empleados:

```
SELECT *
FROM (Oficinas INNER JOIN empleados ON Oficinas.oficina = empleados.oficina)
          INNER JOIN pedidos ON empleados.numemp = pedidos.rep;
```

No interesan las oficinas que no tienen empleados ni los empleados sin oficina luego dejamos el primer INNER, pero pueden haber empleados sin pedidos y nos interesa que salgan (queremos TODOS los empleados de la oficina) luego el segundo JOIN es un LEFT, no pueden haber pedidos sin representante, dejamos el LEFT.

```
SELECT *
FROM (Oficinas INNER JOIN empleados ON Oficinas.oficina = empleados.oficina)
          LEFT JOIN pedidos ON empleados.numemp = pedidos.rep;
```

Si pruebas la instrucción sin LEFT verás que sale una fila menos (la de la oficina 12 y empleado 104, este empleado no tiene pedidos).

El orden de las tablas se puede cambiar siempre que pongamos los JOIN adecuados.

```
SELECT *
FROM (empleados LEFT JOIN pedidos ON empleados.numemp = pedidos.rep)
           INNER JOIN oficinas ON Oficinas.oficina = empleados.oficina
```

O bien

```
SELECT *
FROM (pedidos RIGHT JOIN empleados ON pedidos.rep = empleados.numemp)
           INNER JOIN oficinas ON Oficinas.oficina = empleados.oficina
```

La única diferencia está en el orden de las columnas del resultado (si utilizamos *).

4.- Listar todos los clientes con el nombre del representante asignado (lo llamaremos representante para aclararnos) y el nombre de los empleados que dirige (los llamaremos dirigidos, son los empleados que tienen como jefe el representante) y la ciudad de su oficina (la del empleado asignado al jefe).

Empezamos por clientes con su representante:

```
SELECT *
FROM Clientes INNER JOIN Empleados ON Clientes.repclie=Empleados.numemp
```

Añadimos los empleados que dirige el representante:

```
SELECT *
FROM (Clientes INNER JOIN Empleados ON Clientes.repclie=Empleados.numemp)
         INNER JOIN empleados AS dirigidos ON empleados.numemp=dirigidos.jefe
```

Aquí debemos definir un alias de tabla porque es la segunda vez que aparece la tabla empleados.

Añadimos la ciudad del representante asignado al cliente:

```
SELECT *
FROM ((Clientes INNER JOIN Empleados ON Clientes.repclie=Empleados.numemp)
        INNER JOIN empleados AS dirigidos ON empleados.numemp=dirigidos.jefe)
         INNER JOIN oficinas ON empleados.oficina=oficinas.oficina
```

Estudiamos los tipos de JOIN:

Para el primer JOIN:
No pueden haber clientes sin empleado y empleados no asignados no interesan→ INNER
Para el segundo JOIN:
Representantes que no sean jefes puede ser e interesa que salgan → LEFT
Empleados que no tengan jefe no interesan → dejamos LEFT
Para el tercer JOIN:
Representantes que no tengan oficina puede haber e interesa que salgan → LEFT
Pero no interesan las oficinas sin empleados → dejamos LEFT

La instrucción final es:

```
SELECT *
FROM ((Clientes INNER JOIN Empleados ON Clientes.repclie=Empleados.numemp)
        LEFT JOIN empleados AS dirigidos ON empleados.numemp=dirigidos.jefe)
         LEFT JOIN oficinas ON empleados.oficina=oficinas.oficina
```