

Consultas con varias tablas

Sitio: [Agencia de Aprendizaje a lo largo de la Vida](#)
Curso: Administración de Base de Datos 1° G
Libro: Consultas con varias tablas

Imprimido por: Eduardo Moreno
Día: lunes, 16 de septiembre de 2024, 00:16

Tabla de contenidos

1. Uso de dos tablas	
1.1. Analizar las tablas	
1.2. Validar filas	
1.3. Salvar errores	
1.4. Evitar errores	
1.5. Resultado final	
1.6. Ejemplo de consulta	
2. Uso de más de dos tablas	

Uso de dos tablas



La función más común de las [consultas](#) es [recuperar datos](#) específicos de las tablas.

Los [datos que queremos ver generalmente](#) están distribuidos en varias tablas y las consultas permiten [verlos en una única hoja de datos](#). Por esta razón, [las consultas usan varias tablas del modelo](#) y hay determinados mecanismos para alcanzar el resultado esperado.



Recordá que hasta este momento vimos que con SQL podemos:

- [Crear una base de datos.](#)
- [Agregar contenido en cada atributo.](#)
- [Realizar algunas consultas basándote en la explicación de la semana.](#)

Pero esas consultas [involucran a una sola tabla](#). Para pasar al siguiente paso nos preguntamos:



consulta?

¿Cómo proceder cuando nos damos cuenta que [son dos o más tablas](#) las que necesita la



Consultas a tablas: ¿Qué necesitamos?

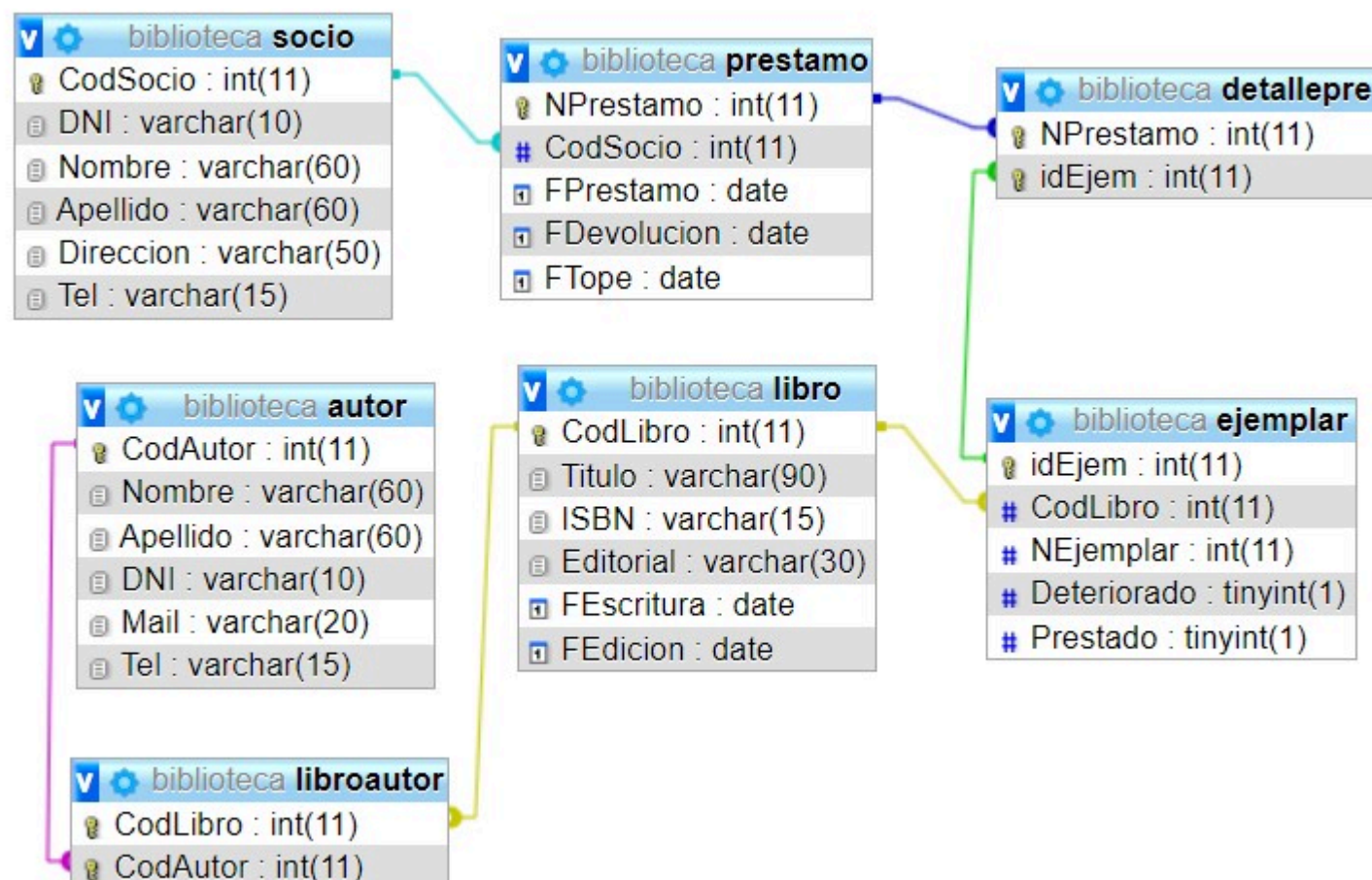
Lo primero es [determinar si las tablas tienen un vínculo](#), es decir, si están relacionadas entre sí.

La [relación](#) de las tablas se [establece con la FK \(clave foránea\)](#) o incluso podemos estar frente a tablas que por alguna razón no se estableció formalmente el vínculo, pero tienen una [columna en común](#), que hacen referencia a la misma información, y esta situación las habilita a vincularse.



Ahora estarás pensando cómo darte cuenta de esta [relación](#). A continuación te explicamos.

1. Retomá la base de datos [Biblioteca](#) que fue creada en la [práctica 1](#) de la [semana 5](#):



2. Observá que las **líneas que se conectan entre las tablas** representan el vínculo entre las tablas. El primer atributo de cada tabla tiene una imagen de **llave** eso quiere decir que es la **PK**. En la tabla “**libroAutor**” y en “**detallePre**” hay dos llaves porque la clave es compuesta. La **PK** se conecta con el mismo atributo que se encuentra en la tabla en la que es **FK**.

3. Mirando la imagen es fácil establecer los vínculos y determinar el camino a seguir para obtener la información que estamos buscando.



Para seguir pensando

Observá nuevamente las tablas anteriores. Supongamos que queremos conocer el **nombre** y **apellido** del **socio** que tiene el préstamo **N° 6**.



Nos hacemos las siguientes preguntas :

- ¿Qué busco?
- ¿Dónde está ese dato?
- ¿La búsqueda tiene alguna condición?



En los siguientes capítulos iremos respondiendo paso a paso cada una de estas preguntas.

Paso 1 - Comencemos con el análisis



¿Qué tenemos que tener en cuenta?

Lo que debemos proyectar es el dato de la columna **nombre** y la columna **apellido**, esos datos se encuentran en la tabla **socio** y la condición que tiene la consulta es que el préstamo debe ser el **6** y ese número está en la tabla **préstamo**.

La proyección va en el **Select**, las tablas involucradas en el **From** y la condición en el **Where**, pero atención: son **relaciones** y se aplica la lógica del **álgebra relacional**.



Veamos un ejemplo

1. Ejecutamos las siguientes instrucciones:

```
Select * From socio;
```

```
Select * From préstamo;
```

El resultado es el siguiente:

```
MariaDB [biblioteca]> select * from socio;
```

CodSocio	DNI	Nombre	Apellido	Direccion	Tel
20145	11452452	Maria Josefuna	Luro	Av Nazca 21478 CABA	47857855
20154	19785452	Marcos	Nevarez	Trelles 1234 CABA	47852154
21474	22145986	Karina	Quirno	Bolivia 52345 CABA	47851414
21489	20145874	Juliana	Laprida	Bacacay 10789 Haedo	49061236
21523	20333564	Viviana	Martinez	Mendoza 123 Martinez	45038796

5 rows in set (0.00 sec)

```
MariaDB [biblioteca]> select * from prestamo;
```

NPrestamo	CodSocio	FPrestamo	FDevolucion	FTope
6	20154	2016-07-29	2016-08-08	2016-08-08
7	21474	2016-08-01	2016-08-08	2016-08-10
8	20154	2016-08-02	2015-08-11	2016-08-11

3 rows in set (0.00 sec)

Como podemos ver la tabla **socio** tiene **5 filas** y la **tabla préstamo** **3 filas**, si hacemos el producto cartesiano el resultado tendrá **5*3 = 15 filas**.

2. Ahora ejecutamos la siguiente instrucción:

```
Select * From socio, préstamo;
```

En pantalla veremos:

MariaDB [biblioteca]> select * from socio,prestamo;

CodSocio	DNI	Nombre	Apellido	Direccion	Tel	NPrestamo	CodSocio	FPrestamo	FDevolucion	FTope
20145	11452452	Maria Josefina	Luro	Av Nazca 21478 CABA	47857855	6	20154	2016-07-29	2016-08-08	2016-08-08
20145	11452452	Maria Josefina	Luro	Av Nazca 21478 CABA	47857855	7	21474	2016-08-01	2016-08-08	2016-08-10
20145	11452452	Maria Josefina	Luro	Av Nazca 21478 CABA	47857855	8	20154	2016-08-02	2015-08-11	2016-08-11
20154	19785452	Marcos	Nevarez	Trelles 1234 CABA	47852154	6	20154	2016-07-29	2016-08-08	2016-08-08
20154	19785452	Marcos	Nevarez	Trelles 1234 CABA	47852154	7	21474	2016-08-01	2016-08-08	2016-08-10
20154	19785452	Marcos	Nevarez	Trelles 1234 CABA	47852154	8	20154	2016-08-02	2015-08-11	2016-08-11
21474	22145986	Karina	Quirno	Bolivia 52345 CABA	47851414	6	20154	2016-07-29	2016-08-08	2016-08-08
21474	22145986	Karina	Quirno	Bolivia 52345 CABA	47851414	7	21474	2016-08-01	2016-08-08	2016-08-10
21474	22145986	Karina	Quirno	Bolivia 52345 CABA	47851414	8	20154	2016-08-02	2015-08-11	2016-08-11
21489	20145874	Juliana	Laprida	Bacacay 10789 Haedo	49061236	6	20154	2016-07-29	2016-08-08	2016-08-08
21489	20145874	Juliana	Laprida	Bacacay 10789 Haedo	49061236	7	21474	2016-08-01	2016-08-08	2016-08-10
21489	20145874	Juliana	Laprida	Bacacay 10789 Haedo	49061236	8	20154	2016-08-02	2015-08-11	2016-08-11
21523	20333564	Viviana	Martinez	Mendoza 123 Martinez	45038796	6	20154	2016-07-29	2016-08-08	2016-08-08
21523	20333564	Viviana	Martinez	Mendoza 123 Martinez	45038796	7	21474	2016-08-01	2016-08-08	2016-08-10
21523	20333564	Viviana	Martinez	Mendoza 123 Martinez	45038796	8	20154	2016-08-02	2015-08-11	2016-08-11

15 rows in set (0.00 sec)

Columnas de SOCIO

Columnas de PRESTAMO

La gráfica muestra las 6 columnas de **socio** y a continuación las 5 columnas de **préstamo**.

Vemos que la primera fila de socio se relacionó con **todas** las filas de préstamo, y así sucesivamente hasta la fila cinco de socio.

Con un círculo está marcado la columna de **relación**, el primer círculo es la **PK** de socio y el segundo círculo la **FK** de préstamo.

Paso 2 - ¿Son válidas las Filas?

Observá detenidamente la siguiente imagen:

CodSocio	DNI	Nombre	Apellido	Direccion	Tel	NPrestamo	CodSocio	FPrestamo	FDevolucion	FTope
20145	11452452	Maria Josefina	Luro	Av Nazca 21478 CABA	47857855	6	20154	2016-07-29	2016-08-08	2016-08-08
20145	11452452	Maria Josefina	Luro	Av Nazca 21478 CABA	47857855	7	21474	2016-08-01	2016-08-08	2016-08-10
20145	11452452	Maria Josefina	Luro	Av Nazca 21478 CABA	47857855	8	20154	2016-08-02	2015-08-11	2016-08-11

¿Son válidas todas las filas? La respuesta es **no**, analicemos por qué.

Volvamos a ver la imagen

Observá con atención la primera fila de socio.

María Josefina Luro tiene el código de socio **20145** y cuando analizamos con el código de socio que figura en la tabla **préstamo** vemos que ninguno coincide, ya que son **20154, 21474 y 20154**, esto significa que María Josefina no llevó libros prestados.

Otro ejemplo

Ahora analicemos la segunda fila de socio que es Marcos Nevarez.

20154	19785452	Marcos	Nevarez	Trelles 1234 CABA	47852154	6	20154	2016-07-29	2016-08-08	2016-08-08
20154	19785452	Marcos	Nevarez	Trelles 1234 CABA	47852154	7	21474	2016-08-01	2016-08-08	2016-08-10
20154	19785452	Marcos	Nevarez	Trelles 1234 CABA	47852154	8	20154	2016-08-02	2015-08-11	2016-08-11

Marcos Nevarez tiene el código de socio **20154** y cuando analizamos con el código de socio que figura en la tabla **préstamo** hay dos coincidencias, esto quiere decir que Marcos realizo dos prestamos

Podemos seguir con el resto de los socios, y resulta que **Karina Quirno** que tiene el código de socio **21474** solicitó un préstamo, que **Juliana Laprida** con código de socio **21489** no llevó libros y que **Viviana Martinez** con código de socio **21523** tampoco llevó libros.

Vemos entonces que hay filas que presentan errores ya que **no coinciden los dominios de las columnas en común**.

Paso 3 - ¿Cómo salvamos los errores?

Para tener resultados correctos los dominios de las columnas que están involucradas en la relación deben ser **iguales**, entonces la sintaxis correcta es la siguiente:

```
select *  
  from socio, prestamo  
 where socio.codsocio = prestamo.codsocio;
```

Vemos que la coma entre socio y préstamo indica el **producto cartesiano** y la **condición de igualdad** de atributos está en el **where**, esta notación simula el algebra relacional.

Contamos con otro formato, el que usaremos durante el curso, donde la coma se reemplaza por la sentencia “**inner join**” y la igualdad de los atributos se coloca a continuación.

```
select *  
  from socio inner join prestamo on socio.codsocio = prestamo.codsocio;
```

La palabra clave **inner join** selecciona todas las filas de ambas tablas siempre que haya una coincidencia entre las columnas, la columna se indica después de la palabra reservada **on**, en este caso es el **codsocio**.

Paso 4 - ¿Cómo evitar el error de “ambiguo”?

Volvamos a la imagen del capítulo anterior:

```
select *
  from socio inner join prestamo on socio.codsocio = prestamo.codsocio;
```



¿Observaste que en la sintaxis de la consulta hay una notación diferente? ¿Notaste que el nombre del atributo lleva delante el nombre de la tabla?

Como el nombre de la columna es el mismo en la tabla socio y en la tabla préstamo, se debe indicar la procedencia y es por eso que se antepone separado por un punto el nombre de la tabla; si no se coloca el nombre, no se ejecuta y produce un error indicando que el nombre es ambiguo (no sabe a qué tabla ir).

Veamos qué resultado arrojó esa sentencia:

```
MariaDB [biblioteca]> select *
-> from socio inner join prestamo on socio.codsocio = prestamo.codsocio;
```

CodSocio	DNI	Nombre	Apellido	Direccion	Tel	NPrestamo	CodSocio	FPrestamo	FDevolucion	FTope
20154	19785452	Marcos	Nevarez	Trelles 1234 CABA	47852154	6	20154	2016-07-29	2016-08-08	2016-08-08
20154	19785452	Marcos	Nevarez	Trelles 1234 CABA	47852154	8	20154	2016-08-02	2015-08-11	2016-08-11
21474	22145986	Karina	Quirno	Bolivia 52345 CABA	47851414	7	21474	2016-08-01	2016-08-08	2016-08-10

3 rows in set (0.00 sec)

Ahora vemos que el codsocio de la tabla socio coincide con el codsocio de la tabla préstamo para cada fila del resultado de la consulta.



Ya aprendimos como utilizar dos tablas en una consulta ¿pero es el resultado que pide la consigna inicial? La respuesta es no.

Paso 5 - Resultado Final

Observemos la siguiente imagen:

```
select nombre, apellido
  from socio inner join prestamo on socio.codsocio = prestamo.codsocio
 where nprestamo = 6;
```

La consigna pide **nombre y apellido del socio que tiene el préstamo N° 6**.

Debemos considerar el filtro de búsqueda. Para ello usamos el **where** y debemos proyectar solo las columnas que nos piden.

Ahora sí el resultado que muestra la consulta es el correcto. Y así lo podemos ver en la pantalla.

```
mariaDB [biblioteca]> select nombre, apellido
-> from socio inner join prestamo on socio.codsocio = prestamo.codsocio
-> where nprestamo = 6;
-----+-----+
nombre | apellido |
-----+-----+
Marcos | Nevarez  |
-----+-----+
1 row in set (0.00 sec)
```

Buscar códigos de libros prestados



Veamos esta consulta:

Mostrar los códigos de los libros de cada préstamo .

Tablas involucradas ---->> [detallePre](#) y [Ejemplar](#)

Atributo en común ----->> [idejem](#)

Condiciones ---->> [ninguna](#)

```
select NPrestamo, codlibro
  from detallepre inner join ejemplar on detallepre.idejem = ejemplar.idejem;
```



Importante. Si utilizamos más de una tabla, estas deben tener un atributo en común y para que los datos sean consistentes las columnas en común se deben igualar.

Uso de más de dos tablas



Veamos ahora qué sucede si la consulta tiene más de dos tablas, para ello consideremos la última consulta con alguna modificación.

La consulta propuesta en el anterior capítulo decía *Mostrar los códigos de los libros de cada préstamo*; la modificamos de la siguiente manera: *Mostrar los títulos de los libros de cada préstamo ordenados por préstamo*.

Tablas involucradas --->> `detallePre`, `Ejemplar` y `Libro` ¿Por qué estas y no otras? Porque `detallePre` además del número de préstamo muestra cual es el ejemplar del libro que se presta, en `ejemplar` está el código del libro que le corresponde al ejemplar prestado y en `libro` esta el título que es el dato que debemos mostrar.

Atributo en común --->> entre `detallePre` y `Ejemplar` es `idEjem` y entre `Ejemplar` y `Libro` es `codLibro`. ¿Cómo sabemos esto? Por el MER.

Condiciones -->> ninguna

Sintaxis de la consulta



Veamos cómo es la sintaxis de la consulta que señalamos anteriormente.

```
select NPrestamo , titulo
from detallepre as d inner join ejemplar as e on d.idejem = e.idejem
inner join libro as l on e.codlibro = l.codlibro
order by NPrestamo;
```

Atributos a mostrar

Primer producto cartesiano

Segundo producto cartesiano

Ordenado por el atributo que pide la consigna

Cuando en la consulta se tienen más de dos tablas en juego, el procedimiento de resolución es el siguiente: el gestor toma las dos primeras realiza el producto cartesiano y el resultado lo considera como un único elemento, luego lo vincula con la tabla siguiente procediendo nuevamente con el producto cartesiano y así sucesivamente hasta agotar las tablas.



Nota. Si querés observar cómo trabaja el gestor, ejecutá la instrucción anterior y en lugar de proyectar los atributos solicitados colocá el ***** (asterisco) y verás que están todos los atributos de las tablas intervinientes.