CAPÍTULO 5

Consultar varias tablas

En el Capítulo 2, demostré cómo los conceptos relacionados se dividen en partes separadas a través de un proceso conocido como normalización. El resultado final de este ejercicio fue dos tablas: persona y comida_favorita . Sin embargo, si desea generar un solo informe mostrando el nombre, la dirección y las comidas favoritas de una persona, necesitará un mecanismo para volver a juntar los datos de estas dos tablas; este mecanismo se conoce como una combinación, y este capítulo se concentra en la combinación más simple y común, la combinación interna unirse. El capítulo 10 muestra todos los diferentes tipos de combinación.

¿Qué es una unión?

Las consultas en una sola tabla ciertamente no son raras, pero encontrará que la mayoría de sus las consultas requerirán dos, tres o incluso más tablas. Para ilustrar, veamos la definición iniciones para las tablas de empleados y departamentos y luego defina una consulta que recupere datos de ambas tablas:

```
mysql> empleado DESC;
+ -------------------- + ---------------------- + ----- - + ----- + --------- +
| Campo                | Tipo                   | Nulo | Clave | Por defecto |
+ -------------------- + ---------------------- + ----- - + ----- + --------- +
| emp_id               | smallint (5) sin firmar | NO | PRI | NULL |
| fname                | varchar (20)           | NO | | NULL |
| lname                | varchar (20)           | NO | | NULL |
| fecha de inicio      | fecha                  | NO | | NULL |
| fecha final          | fecha                  | SI | | NULL |
| superior_emp_id | smallint (5) sin firmar | SI | MUL | NULL |
```

```
| dept_id | smallint (5) sin firmar | SI | MUL | NULL |
| titulo | varchar (20) | SI | | NULL |
| asignado_branch_id | smallint (5) sin firmar | SI | MUL | NULL |
+ ------------------- + --------------------- + ----- - + ----- + --------- +
9 filas en conjunto (0,11 seg)

  mysql> departamento DESC;
+ --------- + ---------------------- + ------ + ----- + --- ------ +
| Campo | Tipo | Nulo | Clave | Por defecto |
+ --------- + ---------------------- + ------ + ----- + --- ------ +
| dept_id | smallint (5) sin firmar | No | PRI | NULL |
```

81

re ldt W B kC

Página 2

```
| nombre | varchar (20) | No | | NULL |
+ --------- + ---------------------- + ------ + ----- + --- ------ +
2 filas en conjunto (0.03 seg)
```

Supongamos que desea recuperar el nombre y apellido de cada empleado junto con el nombre del departamento al que está asignado cada empleado. Por tanto, su consulta necesita recuperar las columnas employee.fname , employee.lname y department.name . Pero, ¿cómo puede recuperar datos de ambas tablas en la misma consulta? La respuesta esta en la columna employee.dept_id , que contiene el ID del departamento al que cada empleado es asignado (en términos más formales, la columna employee.dept_id es la clave de la tabla de departamentos). La consulta, que verá en breve, indica al servidor para utilizar la columna employee.dept_id como puente entre el empleado y tablas de departamento , permitiendo así que las columnas de ambas tablas se incluyan en la conjunto de resultados de la consulta. Este tipo de operación se conoce como combinación.

Producto cartesiano

La forma más fácil de comenzar es colocar las tablas de empleados y departamentos en la cláusula from de una consulta y ver qué pasa. Aquí hay una consulta que recupera la primera y apellidos junto con el nombre del departamento, con una cláusula from nombrando ambas tablas separados por la palabra clave de unión :

```
mysql> SELECT e.fname, e.lname, d.name
    -> DEL empleado e UNIRSE al departamento d;
+ ---------- + ----------- + ---------------- +
| fname | lname | nombre |
+ ---------- + ----------- + ---------------- +
```

Michael	Smith	Operaciones
Michael	Smith	Préstamos
Michael	Smith	Administración
Susan	Barker	Operaciones
Susan	Barker	Préstamos
Susan	Barker	Administración
Robert	Tyler	Operaciones
Robert	Tyler	Préstamos
Robert	Tyler	Administración
Susan	Hawthorne	Operaciones
Susan	Hawthorne	Préstamos
Susan	Hawthorne	Administración
John	Gooding	Operaciones
John	Gooding	Préstamos
John	Gooding	Administración
Helen	Fleming	Operaciones
Helen	Fleming	Préstamos
Helen	Fleming	Administración
Chris	Tucker	Operaciones
Chris	Tucker	Préstamos
Chris	Tucker	Administración
Sarah	Parker	Operaciones
Sarah	Parker	Préstamos
Sarah	Parker	Administración
Jane	Grossman	Operaciones

re ldt W B kC

Página 3

Jane	Grossman	Préstamos
Jane	Grossman	Administración
Paula	Roberts	Operaciones
Paula	Roberts	Préstamos
Paula	Roberts	Administración
Thomas	Ziegler	Operaciones
Thomas	Ziegler	Préstamos
Thomas	Ziegler	Administración
Samantha	Jameson	Operaciones
Samantha	Jameson	Préstamos
Samantha	Jameson	Administración
John	Blake	Operaciones
John	Blake	Préstamos
John	Blake	Administración
Cindy	Mason	Operaciones
Cindy	Mason	Préstamos
Cindy	Mason	Administración

```
| Frank | Portman | Operaciones |
| Frank | Portman | Préstamos                |
| Frank | Portman | Administración |
| Theresa | Markham | Operaciones |
| Theresa | Markham | Préstamos                |
| Theresa | Markham | Administración |
| Beth | Fowler | Operaciones |
| Beth | Fowler | Préstamos                |
| Beth | Fowler | Administración |
| Rick | Tulman | Operaciones |
| Rick | Tulman | Préstamos                |
| Rick | Tulman | Administración |
+ ---------- + ---------- + ---------------- +
54 filas en conjunto (0,23 seg)
```

Mmmm ... solo hay 18 empleados y 3 departamentos diferentes, entonces, ¿cómo fue el resultado? el conjunto termina con 54 filas? Mirando más de cerca, puede ver que el conjunto de 18 empleados se repite tres veces, con todos los datos idénticos excepto el nombre del departamento. Debido a que la consulta no especificó cómo se deben unir las dos tablas, la base de datos servidor generó el producto cartesiano, que es cada permutación de las dos tablas (18 empleados × 3 departamentos = 54 permutaciones). Este tipo de combinación se conoce como unión cruzada, y rara vez se usa (al menos a propósito). Las uniones cruzadas son una de las tipos que estudiamos en el Capítulo 10.

Uniones internas

Para modificar la consulta anterior de modo que solo se incluyan 18 filas en el conjunto de resultados (una para cada empleado), debe describir cómo se relacionan las dos tablas. Antes, yo mostró que la columna employee.dept_id sirve como enlace entre las dos tablas, por lo que esta información debe agregarse a la subcláusula on de la cláusula from :

```
mysql> SELECT e.fname, e.lname, d.name
    -> DEL empleado e UNIRSE al departamento d
    -> ON e.dept_id = d.dept_id;
```


re ldt W B kC

Página 4

```
+ ---------- + ---------- + ---------------- +
| fname | lname | nombre                   |
```

```
+ ---------- + ----------- + ---------------- +
| Michael | Smith | Administración |
| Susan | Barker | Administración |
| Robert | Tyler | Administración |
| Susan | Hawthorne | Operaciones |
| John | Gooding | Préstamos                                    |
| Helen | Fleming | Operaciones |
| Chris | Tucker | Operaciones |
| Sarah | Parker | Operaciones |
| Jane | Grossman | Operaciones |
| Paula | Roberts | Operaciones |
| Thomas | Ziegler | Operaciones |
| Samantha | Jameson | Operaciones |
| John | Blake | Operaciones |
| Cindy | Mason | Operaciones |
| Frank | Portman | Operaciones |
| Theresa | Markham | Operaciones |
| Beth | Fowler | Operaciones |
| Rick | Tulman | Operaciones |
+ ---------- + ----------- + ---------------- +
18 filas en conjunto (0,00 seg)
```

En lugar de 54 filas, ahora tiene los esperados 18 filas debido a la adición de la de subcláusula, que indica al servidor que se una a las tablas de empleados y departamentos utilizando la columna dept_id para pasar de una tabla a otra. Por ejemplo, Susan Haw-
La fila de Thorne en la tabla de empleados contiene un valor de 1 en la columna dept_id (no mostrado en el ejemplo). El servidor usa este valor para buscar la fila en el departamento que tiene un valor de 1 en su columna dept_id y luego recupera el valor 'Operaciones' de la columna de nombre en esa fila.

Si existe un valor para la columna dept_id en una tabla pero no en la otra, la combinación falla para las filas que contienen ese valor y esas filas se excluyen del conjunto de resultados. Esta El tipo de combinación se conoce como combinación interna y es el tipo de combinación más utilizado. Para aclarar, si la tabla de departamentos contiene una cuarta fila para el departamento de marketing, pero no se han asignado empleados a ese departamento, entonces el departamento de marketing ment no se incluiría en el conjunto de resultados. Asimismo, si alguno de los empleados hubiera asignado al ID de departamento 99, que no existe en la tabla de departamentos , estos empleados quedarían fuera del conjunto de resultados. Si desea incluir todas las filas de una tabla u otra, independientemente de si existe una coincidencia, debe especificar una unirse, pero cubriremos esto más adelante en el libro.

En el ejemplo anterior, no especifiqué en la cláusula from qué tipo de combinación usar. Sin embargo, cuando desee unir dos tablas utilizando una combinación interna, debe explícitamente especifique esto en su cláusula from ; aquí está el mismo ejemplo, con la adición de la combinación tipo (tenga en cuenta la palabra clave INNER):

```
mysql> SELECT e.fname, e.lname, d.name
    -> DEL empleado e INNER JOIN departamento d
```


re ldt W B kC

Página 5

```
    -> ON e.dept_id = d.dept_id;
+ ---------- + ----------- + ---------------- +
| fname | lname | nombre                       |
+ ---------- + ----------- + ---------------- +
| Michael | Smith | Administración |
| Susan | Barker | Administración |
| Robert | Tyler | Administración |
| Susan | Hawthorne | Operaciones |
| John | Gooding | Préstamos                  |
| Helen | Fleming | Operaciones |
| Chris | Tucker | Operaciones |
| Sarah | Parker | Operaciones |
| Jane | Grossman | Operaciones |
| Paula | Roberts | Operaciones |
| Thomas | Ziegler | Operaciones |
| Samantha | Jameson | Operaciones |
| John | Blake | Operaciones |
| Cindy | Mason | Operaciones |
| Frank | Portman | Operaciones |
| Theresa | Markham | Operaciones |
| Beth | Fowler | Operaciones |
| Rick | Tulman | Operaciones |
+ ---------- + ----------- + ---------------- +
18 filas en conjunto (0,00 seg)
```

Si no especifica el tipo de combinación, el servidor realizará una combinación interna de forma predeterminada. Sin embargo, como verá más adelante en el libro, hay varios tipos de combinaciones, por lo que debe Adquiera el hábito de especificar el tipo exacto de unión que necesita.

Si los nombres de las columnas utilizadas para unir las dos tablas son idénticos, lo que es cierto en la consulta anterior, puede utilizar la subcláusula using en lugar de la subcláusula on , como en:

```
mysql> SELECT e.fname, e.lname, d.name
    -> DEL empleado e INNER JOIN departamento d
    -> USANDO (dept_id);
+ ---------- + ----------- + ---------------- +
| fname | lname | nombre                       |
+ ---------- + ----------- + ---------------- +
| Michael | Smith | Administración |
| Susan | Barker | Administración |
| Robert | Tyler | Administración |
| Susan | Hawthorne | Operaciones |
| John | Gooding | Préstamos                  |
| Helen | Fleming | Operaciones |
| Chris | Tucker | Operaciones |
| Sarah | Parker | Operaciones |
| Jane | Grossman | Operaciones |
| Paula | Roberts | Operaciones |
| Thomas | Ziegler | Operaciones |
| Samantha | Jameson | Operaciones |
| John | Blake | Operaciones |
| Cindy | Mason | Operaciones |
| Frank | Portman | Operaciones |
| Theresa | Markham | Operaciones |
| Beth | Fowler | Operaciones |
```


re ldt W B kC

Página 6

```
| Rick | Tulman | Operaciones |
+ ---------- + ----------- + ---------------- +
18 filas en conjunto (0.01 seg)
```

Dado que el uso es una notación abreviada que puede usar solo en una situación específica, prefiero utilizar siempre la subcláusula on para evitar confusiones.

La sintaxis de unión ANSI

La notación utilizada en este libro para unir tablas se introdujo en SQL92 versión del estándar ANSI SQL. Todas las principales bases de datos (Oracle Database, Microsoft SQL Server, MySQL, IBM DB2 Universal Database y Sybase Adaptive Server) han adoptado la sintaxis de combinación SQL92. Debido a que la mayoría de estos servidores desde antes del lanzamiento de la especificación SQL92, todos incluyen una unirse a la sintaxis también. Por ejemplo, todos estos servidores entenderían lo siguiente variación de la consulta anterior:

```
mysql> SELECT e.fname, e.lname, d.name
    -> DEL empleado e, departamento d
    -> DONDE e.dept_id = d.dept_id;
+ ---------- + ----------- + ---------------- +
| fname | lname | nombre |
+ ---------- + ----------- + ---------------- +
| Michael | Smith | Administración |
| Susan | Barker | Administración |
| Robert | Tyler | Administración |
| Susan | Hawthorne | Operaciones |
| John | Gooding | Préstamos |
| Helen | Fleming | Operaciones |
| Chris | Tucker | Operaciones |
| Sarah | Parker | Operaciones |
| Jane | Grossman | Operaciones |
| Paula | Roberts | Operaciones |
| Thomas | Ziegler | Operaciones |
| Samantha | Jameson | Operaciones |
| John | Blake | Operaciones |
| Cindy | Mason | Operaciones |
| Frank | Portman | Operaciones |
| Theresa | Markham | Operaciones |
| Beth | Fowler | Operaciones |
| Rick | Tulman | Operaciones |
+ ---------- + ----------- + ---------------- +
18 filas en conjunto (0.01 seg)
```

Este método más antiguo de especificar combinaciones no incluye la subcláusula on ; en cambio, tablas se nombran en la cláusula from separados por comas y se incluyen las condiciones de combinación

en la cláusula where . Si bien puede decidir ignorar la sintaxis SQL92 en favor de la sintaxis de unión anterior, la sintaxis de unión ANSI tiene las siguientes ventajas:

- Las condiciones de combinación y las condiciones de filtro se separan en dos cláusulas diferentes (la sobre la subcláusula y la cláusula where , respectivamente), lo que facilita la consulta entender.

re ldt W B kC

Página 7

- Las condiciones de combinación para cada par de tablas están contenidas en su propia cláusula on , lo que hace menos probable que parte de una combinación se omita por error.
- Las consultas que utilizan la sintaxis de combinación SQL92 son portables entre servidores de bases de datos, mientras que la sintaxis anterior es ligeramente diferente en los diferentes servidores.

Los beneficios de la sintaxis de combinación SQL92 son más fáciles de identificar para consultas complejas que incluir tanto condiciones de unión como de filtrado. Considere la siguiente consulta, que devuelve todos cuentas abiertas por cajeros experimentados (contratados antes de 2007) actualmente asignados a la Rama de Woburn:

```
mysql> SELECT a.account_id, a.cust_id, a.open_date, a.product_cd
    -> DE la cuenta a, sucursal b, empleado e
    -> DONDE a.open_emp_id = e.emp_id
    -> Y e.start_date <'2007-01-01'
    -> Y e.assigned_branch_id = b.branch_id
    -> Y (e.title = 'Cajero' O e.title = 'Cajero principal')
    -> Y b.name = 'Woburn Branch';
+ ------------ + --------- + ------------ + ------------ +
| account_id | cust_id | open_date | product_cd |
+ ------------ + --------- + ------------ + ------------ +
| 1 | 1 | 2000-01-15 | CHK |
| 2 | 1 | 2000-01-15 | SAV |
| 3 | 1 | 2004-06-30 | discos compactos |
| 4 | 2 | 2001-03-12 | CHK |
| 5 | 2 | 2001-03-12 | SAV |
| 17 | 7 | 2004-01-12 | discos compactos |
| 27 | 11 | 2004-03-22 | AUTOBÚS |
+ ------------ + --------- + ------------ + ------------ +
7 filas en conjunto (0,00 seg)
```

Con esta consulta, no es tan fácil determinar qué condiciones en la cláusula where son condiciones de unión y cuáles son condiciones de filtro. Tampoco es evidente qué

tipo de unión que se está utilizando (para identificar el tipo de unión, debe buscar de cerca en las condiciones de combinación en la cláusula where para ver si algún carácter especial se emplean), ni es fácil determinar si alguna condición de unión ha sido incorrecta
Tomado dejado fuera. Esta es la misma consulta que usa la sintaxis de combinación SQL92:

```
mysql> SELECT a.account_id, a.cust_id, a.open_date, a.product_cd
    -> DESDE la cuenta un empleado INNER JOIN e
    -> ON a.open_emp_id = e.emp_id
    -> INNER JOIN rama b
    -> ON e.assigned_branch_id = b.branch_id
    -> DONDE e.start_date <'2007-01-01'
    -> Y (e.title = 'Cajero' O e.title = 'Cajero principal')
    -> Y b.name = 'Woburn Branch';
+ ------------ + --------- + ------------ + ------------ +
| account_id | cust_id | open_date | product_cd |
+ ------------ + --------- + ------------ + ------------ +
|          1 | 1 | 2000-01-15 | CHK                    |
|          2 | 1 | 2000-01-15 | SAV                    |
|          3 | 1 | 2004-06-30 | discos compactos       |
|          4 | 2 | 2001-03-12 | CHK                    |
|          5 | 2 | 2001-03-12 | SAV                    |
```


re ldt W B kC

Página 8

```
|         17 | 7 | 2004-01-12 | discos compactos       |
|         27 | 11 | 2004-03-22 | AUTOBÚS                |
+ ------------ + --------- + ------------ + ------------ +
7 filas en conjunto (0.05 seg)
```

Con suerte, estará de acuerdo en que la versión que usa la sintaxis de unión SQL92 es más fácil de entender.

Unir tres o más tablas

Unir tres mesas es similar a unir dos mesas, pero con una ligera arruga. Con una combinación de dos tablas, hay dos tablas y un tipo de combinación en la cláusula from , y una única en la subcláusula para definir cómo se unen las tablas. Con una combinación de tres mesas, hay tres tablas y dos tipos de combinación en la cláusula from y dos en subcláusulas. Aquí está otro ejemplo de una consulta con una combinación de dos tablas:

```
mysql> SELECCIONAR a.account_id, c.fed_id
    -> DE la cuenta un cliente INNER JOIN c
```

```
        -> ON a.cust_id = c.cust_id
        -> DONDE c.cust_type_cd = 'B';
+ ------------ + ------------ +
| account_id | fed_id |
+ ------------ + ------------ +
|                 24 | 04-1111111 |
|                 25 | 04-1111111 |
|                 27 | 04-2222222 |
|                 28 | 04-3333333 |
|                 29 | 04-4444444 |
+ ------------ + ------------ +
5 filas en conjunto (0,15 seg)
```

Esta consulta, que devuelve el ID de cuenta y el número de impuestos federales para todas las recuentos, debería parecer bastante sencillo a estas alturas. Sin embargo, si agrega el empleado tabla a la consulta para recuperar también el nombre del cajero que abrió cada cuenta, tiene el siguiente aspecto:

```
mysql> SELECCIONAR a.account_id, c.fed_id, e.fname, e.lname
        -> DE la cuenta un cliente INNER JOIN c
        -> ON a.cust_id = c.cust_id
        -> INNER JOIN empleado e
        -> ON a.open_emp_id = e.emp_id
        -> DONDE c.cust_type_cd = 'B';
+ ------------ + ------------ + --------- + --------- +
| account_id | fed_id | fname | lname |
+ ------------ + ------------ + --------- + --------- +
|                 24 | 04-1111111 | Theresa | Markham |
|                 25 | 04-1111111 | Theresa | Markham |
|                 27 | 04-2222222 | Paula | Roberts |
|                 28 | 04-3333333 | Theresa | Markham |
|                 29 | 04-4444444 | John | Blake |
+ ------------ + ------------ + --------- + --------- +
5 filas en conjunto (0,00 seg)
```


re ldt W B kC

Página 9

Ahora tres mesas, dos tipos de combinación, y dos en los incisos se enumeran en la de la cláusula, así que las cosas se han puesto un poco más ocupadas. A primera vista, el orden en que las tablas se nombran puede hacer que piense que la tabla de empleados se está uniendo a la tabla de clientes , ya que la tabla de cuentas se nombra primero, seguida de la tabla de clientes , y luego la mesa de empleados . Si cambia el orden en el que aparecen las dos primeras tablas,

sin embargo, obtendrá exactamente los mismos resultados:

```
mysql> SELECCIONAR a.account_id, c.fed_id, e.fname, e.lname
    -> DESDE el cliente c INNER JOIN cuenta a
    -> ON a.cust_id = c.cust_id
    -> INNER JOIN empleado e
    -> ON a.open_emp_id = e.emp_id
    -> DONDE c.cust_type_cd = 'B';
+ ------------ + ------------ + --------- + --------- +
| account_id | fed_id | fname | lname |
+ ------------ + ------------ + --------- + --------- +
| 24 | 04-1111111 | Theresa | Markham |
| 25 | 04-1111111 | Theresa | Markham |
| 27 | 04-2222222 | Paula | Roberts |
| 28 | 04-3333333 | Theresa | Markham |
| 29 | 04-4444444 | John | Blake |
+ ------------ + ------------ + --------- + --------- +
5 filas en conjunto (0.09 seg)
```

La tabla de clientes ahora aparece primero, seguida de la tabla de cuentas y luego la mesa de empleados . Dado que las subcláusulas on no han cambiado, los resultados son los mismos. por en aras de la integridad, aquí está la misma consulta una última vez, pero con el orden de la tabla completamente invertido (empleado a cuenta a cliente):

```
mysql> SELECCIONAR a.account_id, c.fed_id, e.fname, e.lname
    -> DESDE el empleado e INNER JOIN cuenta a
    -> ON e.emp_id = a.open_emp_id
    -> INNER JOIN cliente c
    -> ON a.cust_id = c.cust_id
    -> DONDE c.cust_type_cd = 'B';
+ ------------ + ------------ + --------- + --------- +
| account_id | fed_id | fname | lname |
+ ------------ + ------------ + --------- + --------- +
| 24 | 04-1111111 | Theresa | Markham |
| 25 | 04-1111111 | Theresa | Markham |
| 27 | 04-2222222 | Paula | Roberts |
| 28 | 04-3333333 | Theresa | Markham |
| 29 | 04-4444444 | John | Blake |
+ ------------ + ------------ + --------- + --------- +
5 filas en conjunto (0,00 seg)
```


re ldt W B kC

Página 10

¿Importa la orden de unión?

Si usted está confundido acerca de por qué las tres versiones de la cuenta / empleado / cliente consulta producir los mismos resultados, tenga en cuenta que SQL es un lenguaje sin procedimientos, lo que significa que describa lo que desea recuperar y qué objetos de la base de datos deben ser involucrado, pero depende del servidor de la base de datos determinar la mejor manera de ejecutar su consulta. Usando estadísticas recopiladas de los objetos de su base de datos, el servidor debe elegir uno de tres tablas como punto de partida (la tabla elegida se conoce a partir de entonces como tabla), y luego decida en qué orden unirse a las tablas restantes. Por tanto, la orden en qué tablas aparecen en su cláusula from no es significativo.

Sin embargo, si cree que las tablas de su consulta siempre deben unirse en un orden particular, puede colocar las tablas en el orden deseado y luego especificar el palabra clave STRAIGHT_JOIN en MySQL, solicite la opción FORCE ORDER en SQL Server, o utilice la sugerencia del optimizador ORDERED o LEADING en Oracle Database. Por ejemplo, para decirle al servidor MySQL que use la tabla de clientes como tabla de manejo y luego unirse las tablas de cuentas y empleados , puede hacer lo siguiente:

```
mysql> SELECT STRAIGHT_JOIN a.account_id, c.fed_id, e.fname, e.lname
    -> DESDE el cliente c INNER JOIN cuenta a
    -> ON a.cust_id = c.cust_id
    -> INNER JOIN empleado e
    -> ON a.open_emp_id = e.emp_id
    -> DONDE c.cust_type_cd = 'B';
```

Una forma de pensar en una consulta que usa tres o más tablas es como una bola de nieve rodando hacia abajo una colina. Las dos primeras mesas hacen rodar la pelota, y cada mesa subsiguiente se agrega a la bola de nieve mientras se dirige cuesta abajo. Puedes pensar en la bola de nieve como el intermedio conjunto de resultados, que está recogiendo más y más columnas a medida que se unen las tablas posteriores. Por lo tanto, la tabla de empleados no se une realmente a la tabla de cuentas , sino el conjunto de resultados intermedios creado cuando se unieron las tablas de clientes y cuentas . (En caso de que se esté preguntando por qué elegí una analogía de bola de nieve, escribí este capítulo en en medio de un invierno de Nueva Inglaterra: 110 pulgadas hasta ahora, y más mañana. Oh alegría.)

Usar subconsultas como tablas

Ya ha visto varios ejemplos de consultas que utilizan tres tablas, pero hay una variación que vale la pena mencionar: qué hacer si algunos de los conjuntos de datos son generados por consultas. Las subconsultas son el foco del Capítulo 9, pero ya presenté el concepto de una subconsulta en la cláusula from del capítulo anterior. Aquí hay otra versión de un consulta anterior (busque todas las cuentas abiertas por cajeros experimentados actualmente asignados al Woburn branch) que une la tabla de cuentas a las subconsultas de la rama y mesas de empleados :

```
1 SELECCIONE a.account_id, a.cust_id, a.open_date, a.product_cd
2 DESDE una cuenta INNER JOIN
3 (SELECT emp_id, asignado_branch_id
```


re ldt W B kC

Página 11

```
4 DE empleado
5 DONDE fecha_inicio <'2007-01-01'
6 Y (título = 'Cajero' O título = 'Cajero principal')) e
7 ON a.open_emp_id = e.emp_id
8 INNER JOIN
9 (SELECT branch_id
10 FROM rama
11 DONDE nombre = 'Rama Woburn') b
12 ON e.assigned_branch_id = b.branch_id;
```

La primera subconsulta, que comienza en la línea 3 y recibe el alias e , encuentra todos los cajeros. La segunda subconsulta, que comienza en la línea 9 y recibe el alias b , encuentra el Identificación de la sucursal de Woburn. Primero, la mesa de cuentas está unida al cajero experimentado. subconsulta usando el ID de empleado y luego la tabla resultante se une al Woburn subconsulta de rama usando el ID de rama. Los resultados son los mismos que los del anterior versión de la consulta (pruébelo y compruébelo usted mismo), pero las consultas se ven muy diferentes de unos y otros.

Realmente no hay nada impactante aquí, pero puede llevar un minuto descubrir qué es pasando. Observe, por ejemplo, la falta de una cláusula where en la consulta principal; puesto que todos las condiciones de filtro son contra las tablas de empleados y sucursales , las condiciones de filtro son todo dentro de las subconsultas, por lo que no hay necesidad de condiciones de filtro en la consulta principal. Una forma de visualizar lo que está sucediendo es ejecutar las subconsultas y observar el resultado. conjuntos. Estos son los resultados de la primera subconsulta en la tabla de empleados :

```
mysql> SELECT emp_id, asignado_branch_id
    -> DESDE empleado
    -> DONDE fecha_inicio <'2007-01-01'
    -> Y (título = 'Cajero' O título = 'Cajero principal');
+ -------- + -------------------- +
| emp_id | asignado_branch_id |
+ -------- + -------------------- +
| 8 | 1 |
| 9 | 1 |
| 10 | 2 |
| 11 | 2 |
| 13 | 3 |
| 14 | 3 |
| 16 | 4 |
| 17 | 4 |
| 18 | 4 |
+ -------- + -------------------- +
9 filas en conjunto (0.03 seg)
```

Por lo tanto, este conjunto de resultados consta de un conjunto de ID de empleado y su rama correspondiente Identificaciones. Cuando se unen a la tabla de la cuenta a través de la columna emp_id , ahora tiene un conjunto de resultados intermedios que consta de todas las filas de la tabla de cuentas con el columna nacional que contiene el ID de la sucursal del empleado que abrió cada cuenta. aquí son los resultados de la segunda subconsulta contra la tabla de ramas :

```
mysql> SELECT branch_id
    -> DESDE la rama
    -> DONDE nombre = 'Rama Woburn';
```


re ldt W B kC

Pagina 12

```
+ ----------- +
| branch_id |
+ ----------- +
| 2 |
+ ----------- +
1 fila en conjunto (0.02 seg)
```

Esta consulta devuelve una sola fila que contiene una sola columna: el ID del Woburn
rama. Esta tabla se une a la columna asignado_branch_id del resultado intermedio
configurada, lo que provoca que todas las cuentas abiertas por empleados que no sean de Woburn se filtren
el conjunto de resultados final.

Usar la misma tabla dos veces

Si se une a varias mesas, es posible que necesite unirse a la misma mesa
mas de una vez. En la base de datos de muestra, por ejemplo, hay claves foráneas para el
tabla de sucursales de la tabla de cuentas (la sucursal en la que se abrió la cuenta)
y la mesa de empleados (la sucursal en la que trabaja el empleado). Si quieres incluir
ambas ramas de su resultado, puede incluir la rama mesa dos veces en el de
cláusula, unida una vez a la tabla de empleados y una vez a la tabla de cuentas . Para que esto funcione,
tendrá que dar a cada instancia de la tabla de rama un alias diferente para que el servidor
sabe a cuál te refieres en las distintas cláusulas, como en:

```
mysql> SELECT a.account_id, e.emp_id,
    -> b_a.name open_branch, b_e.name emp_branch
    -> DE la cuenta una rama INNER JOIN b_a
    -> ON a.open_branch_id = b_a.branch_id
    -> INNER JOIN empleado e
    -> ON a.open_emp_id = e.emp_id
    -> INNER JOIN rama b_e
    -> ON e.assigned_branch_id = b_e.branch_id
    -> DONDE a.product_cd = 'CHK';
+ ------------ + -------- + ---------------- + ------------ ---- +
| account_id | emp_id | open_branch | emp_branch |
+ ------------ + -------- + ---------------- + ------------ ---- +
```

```
|                 10 | 1 | Sede | Sede |
|                 14 | 1 | Sede | Sede |
|                 21 | 1 | Sede | Sede |
|                  1 | 10 | Sucursal Woburn | Sucursal Woburn |
|                  4 | 10 | Sucursal Woburn | Sucursal Woburn |
|                  7 | 13 | Sucursal de Quincy | Sucursal de Quincy |
|                 13 | 16 | Entonces. Sucursal NH | Entonces. Sucursal NH |
|                 18 | 16 | Entonces. Sucursal NH | Entonces. Sucursal NH |
|                 24 | 16 | Entonces. Sucursal NH | Entonces. Sucursal NH |
|                 28 | 16 | Entonces. Sucursal NH | Entonces. Sucursal NH |
+ ------------ + -------- + ---------------- + ------------ ---- +
10 filas en conjunto (0,16 seg)
```

Esta consulta muestra quién abrió cada cuenta corriente, en qué sucursal se abrió, ya qué sucursal está asignado actualmente el empleado que abrió la cuenta. los La tabla de ramas se incluye dos veces, con los alias b_a y b_e . Asignando diferentes alias

92 | Capítulo 5: Consultar varias tablas

re ldt W B kC

Página 13

a cada instancia de la tabla de sucursales , el servidor puede comprender qué instancia se refieren a: el que se une a la tabla de la cuenta , o el que se une al empleado mesa. Por lo tanto, este es un ejemplo de una consulta que requiere el uso de alias de tabla.

Autouniones

No solo puede incluir la misma tabla más de una vez en la misma consulta, sino que puede realmente unir una mesa a sí mismo. Esto puede parecer extraño al principio, pero hay razones válidas para hacerlo. La tabla de empleados , por ejemplo, incluye un haciendo referencia a la clave externa, lo que significa que incluye una columna (superior_emp_id) que apunta a la clave principal dentro de la misma tabla. Esta columna apunta al empleado gerente (a menos que el empleado sea el jefe, en cuyo caso la columna es nula).
Al utilizar una autounión, puede escribir una consulta que enumere el nombre de cada empleado junto con el nombre de su gerente:

```
mysql> SELECT e.fname, e.lname,
    -> e_mgr.fname mgr_fname, e_mgr.lname mgr_lname
    -> DESDE empleado e INNER JOIN empleado e_mgr
    -> ON e.superior_emp_id = e_mgr.emp_id;
+ ---------- + ----------- + ----------- + ----------- +
```

```
| fname | lname | mgr_fname | mgr_lname |
+ ---------- + ---------- + ---------- + ---------- +
| Susan | Barker | Michael | Smith |
| Robert | Tyler | Michael | Smith |
| Susan | Hawthorne | Robert | Tyler |
| John | Gooding | Susan | Hawthorne |
| Helen | Fleming | Susan | Hawthorne |
| Chris | Tucker | Helen | Fleming |
| Sarah | Parker | Helen | Fleming |
| Jane | Grossman | Helen | Fleming |
| Paula | Roberts | Susan | Hawthorne |
| Thomas | Ziegler | Paula | Roberts |
| Samantha | Jameson | Paula | Roberts |
| John | Blake | Susan | Hawthorne |
| Cindy | Mason | John | Blake |
| Frank | Portman | John | Blake |
| Theresa | Markham | Susan | Hawthorne |
| Beth | Fowler | Theresa | Markham |
| Rick | Tulman | Theresa | Markham |
+ ---------- + ---------- + ---------- + ---------- +
17 filas en conjunto (0,00 seg)
```

Esta consulta incluye dos instancias de la tabla de empleados : una para proporcionar nombres de empleados (con el alias de la tabla e), y el otro para proporcionar los nombres del administrador (con el alias de la tabla e_mgr). La subcláusula on utiliza estos alias para unir la tabla de empleados a sí misma a través de la superior_emp_id clave externa. Este es otro ejemplo de una consulta para la que los alias de tabla son requeridos; de lo contrario, el servidor no sabría si se refiere a un em-
ployee o el gerente de un empleado.

re ldt W B kC

Página 14

Si bien hay 18 filas en la tabla de empleados , la consulta arrojó solo 17 filas; los
El presidente del banco, Michael Smith, no tiene superior (su columna superior_emp_id es null), por lo que la combinación falló para su fila. Para incluir a Michael Smith en el conjunto de resultados, necesitaría usar una combinación externa, que cubrimos en el Capítulo 10.

Equi-Joins versus no Equi-Joins

Todas las consultas de múltiples tablas mostradas hasta ahora han empleado equi-joins, lo que significa que los valores de las dos tablas deben coincidir para que la combinación sea correcta. Un equi-join siempre emplea un signo igual, como en:

```
ON e.assigned_branch_id = b.branch_id
```

Si bien la mayoría de sus consultas emplearán equi-joins, también puede unir sus tablas a través de rangos de valores, que se conocen como no equi-uniones. He aquí un ejemplo de consulta que se une por un rango de valores:

```
SELECCIONE e.emp_id, e.fname, e.lname, e.start_date
DESDE el empleado e INNER JOIN producto p
    ON e.start_date> = p.date_offered
    Y e.start_date <= p.date_retired
DONDE p.name = 'cheque sin cargo';
```

Esta consulta une dos tablas que no tienen relaciones de clave externa. La intención es encontrar todos los empleados que comenzaron a trabajar para el banco mientras el producto de cheques sin cargo se estaba ofreciendo. Por lo tanto, la fecha de inicio de un empleado debe estar entre la fecha en que el producto ofrecido y la fecha en que se retiró el producto.

También puede encontrar la necesidad de una auto-no-combinación, lo que significa que una tabla está unida a sí misma utilizando un non-equi-join. Por ejemplo, digamos que el gerente de operaciones ha decidido tener un torneo de ajedrez para todos los cajeros bancarios. Se le ha pedido que cree una lista de todos los maridajes. Puede intentar unir la mesa de empleados a sí misma para todos los cajeros (título = 'Teller') y devuelve todas las filas donde los emp_id s no coinciden (ya que una persona no puede jugar ajedrez contra sí mismo):

```
mysql> SELECCIONE e1.fname, e1.lname, 'VS' vs, e2.fname, e2.lname
    -> DESDE empleado e1 INNER JOIN empleado e2
    -> ON e1.emp_id! = E2.emp_id
    -> DONDE e1.title = 'Cajero' Y e2.title = 'Cajero';
+ ---------- + ---------- + ---- + ---------- + ---------- +
| fname | lname | vs | fname | lname |
+ ---------- + ---------- + ---- + ---------- + ---------- +
| Sarah | Parker | VS | Chris | Tucker |
| Jane | Grossman | VS | Chris | Tucker |
| Thomas | Ziegler | VS | Chris | Tucker |
| Samantha | Jameson | VS | Chris | Tucker |
| Cindy | Mason | VS | Chris | Tucker |
| Frank | Portman | VS | Chris | Tucker |
| Beth | Fowler | VS | Chris | Tucker |
| Rick | Tulman | VS | Chris | Tucker |
| Chris | Tucker | VS | Sarah | Parker |
```


re ldt W B kC

Página 15

```
| Jane | Grossman | VS | Sarah | Parker |
| Thomas | Ziegler | VS | Sarah | Parker |
| Samantha | Jameson | VS | Sarah | Parker |
| Cindy | Mason | VS | Sarah | Parker |
| Frank | Portman | VS | Sarah | Parker |
| Beth | Fowler | VS | Sarah | Parker |
| Rick | Tulman | VS | Sarah | Parker |
...
| Chris | Tucker | VS | Rick | Tulman |
| Sarah | Parker | VS | Rick | Tulman |
| Jane | Grossman | VS | Rick | Tulman |
| Thomas | Ziegler | VS | Rick | Tulman |
| Samantha | Jameson | VS | Rick | Tulman |
| Cindy | Mason | VS | Rick | Tulman |
| Frank | Portman | VS | Rick | Tulman |
| Beth | Fowler | VS | Rick | Tulman |
+ ---------- + ---------- + ---- + ---------- + ---------- +
72 filas en conjunto (0.01 seg)
```

Estás en el camino correcto, pero el problema aquí es que para cada emparejamiento (p. Ej., Sarah Parker versus Chris Tucker), también hay un emparejamiento inverso (por ejemplo, Chris Tucker versus Sarah Parker). Una forma de lograr los resultados deseados es utilizar la condición de unión e1.emp_id <e2.emp_id para que cada cajero esté emparejado solo con aquellos cajeros que tengan un mayor ID de empleado (también puede usar e1.emp_id> e2.emp_id si lo desea):

```
mysql> SELECCIONE e1.fname, e1.lname, 'VS' vs, e2.fname, e2.lname
    -> DESDE empleado e1 INNER JOIN empleado e2
    -> ACTIVADO e1.emp_id <e2.emp_id
    -> DONDE e1.title = 'Cajero' Y e2.title = 'Cajero';
+ ---------- + ---------- + ---- + ---------- + ---------- +
| fname | lname | vs | fname | lname |
+ ---------- + ---------- + ---- + ---------- + ---------- +
| Chris | Tucker | VS | Sarah | Parker |
| Chris | Tucker | VS | Jane | Grossman |
| Sarah | Parker | VS | Jane | Grossman |
| Chris | Tucker | VS | Thomas | Ziegler |
| Sarah | Parker | VS | Thomas | Ziegler |
| Jane | Grossman | VS | Thomas | Ziegler |
| Chris | Tucker | VS | Samantha | Jameson |
| Sarah | Parker | VS | Samantha | Jameson |
| Jane | Grossman | VS | Samantha | Jameson |
| Thomas | Ziegler | VS | Samantha | Jameson |
| Chris | Tucker | VS | Cindy | Mason |
| Sarah | Parker | VS | Cindy | Mason |
| Jane | Grossman | VS | Cindy | Mason |
| Thomas | Ziegler | VS | Cindy | Mason |
| Samantha | Jameson | VS | Cindy | Mason |
| Chris | Tucker | VS | Frank | Portman |
| Sarah | Parker | VS | Frank | Portman |
| Jane | Grossman | VS | Frank | Portman |
| Thomas | Ziegler | VS | Frank | Portman |
| Samantha | Jameson | VS | Frank | Portman |
| Cindy | Mason | VS | Frank | Portman |
| Chris | Tucker | VS | Beth | Fowler |
| Sarah | Parker | VS | Beth | Fowler |
```


re ldt W B kC

Página 16

```
| Jane | Grossman | VS | Beth | Fowler |
| Thomas | Ziegler | VS | Beth | Fowler |
| Samantha | Jameson | VS | Beth | Fowler |
| Cindy | Mason | VS | Beth | Fowler |
| Frank | Portman | VS | Beth | Fowler |
| Chris | Tucker | VS | Rick | Tulman |
| Sarah | Parker | VS | Rick | Tulman |
| Jane | Grossman | VS | Rick | Tulman |
| Thomas | Ziegler | VS | Rick | Tulman |
| Samantha | Jameson | VS | Rick | Tulman |
| Cindy | Mason | VS | Rick | Tulman |
| Frank | Portman | VS | Rick | Tulman |
| Beth | Fowler | VS | Rick | Tulman |
+ ---------- + ---------- + ---- + ---------- + ---------- +
36 filas en conjunto (0,00 seg)
```

Ahora tiene una lista de 36 emparejamientos, que es el número correcto al elegir pares de 9 cosas distintas.

Condiciones de unión versus condiciones de filtro

Ahora está familiarizado con el concepto de que las condiciones de unión pertenecen a la subcláusula on , mientras que las condiciones de filtro pertenecen a la cláusula where . Sin embargo, SQL es flexible en cuanto a dónde usted coloca sus condiciones, por lo que deberá tener cuidado al construir sus consultas.
Por ejemplo, la siguiente consulta une dos tablas con una sola condición de combinación y también incluye una condición de filtro único en la cláusula where :

```
mysql> SELECCIONAR a.account_id, a.product_cd, c.fed_id
    -> DE la cuenta un cliente INNER JOIN c
    -> ON a.cust_id = c.cust_id
    -> DONDE c.cust_type_cd = 'B';
+ ------------ + ------------ + ------------ +
| account_id | product_cd | fed_id |
+ ------------ + ------------ + ------------ +
|                 24 | CHK                 | 04-1111111 |
|                 25 | AUTOBÚS             | 04-1111111 |
|                 27 | AUTOBÚS             | 04-2222222 |
|                 28 | CHK                 | 04-3333333 |
|                 29 | SBL                 | 04-4444444 |
+ ------------ + ------------ + ------------ +
5 filas en conjunto (0.01 seg)
```

Eso fue bastante sencillo, pero ¿qué sucede si pones el filtro por error condición en la subcláusula on en lugar de en la cláusula where ?

```
mysql> SELECCIONAR a.account_id, a.product_cd, c.fed_id
    -> DE la cuenta un cliente INNER JOIN c
    -> ON a.cust_id = c.cust_id
    -> Y c.cust_type_cd = 'B';
+ ------------ + ------------ + ------------ +
| account_id | product_cd | fed_id |
+ ------------ + ------------ + ------------ +
|                 24 | CHK                 | 04-1111111 |
```

re ldt W B kC

Página 17

```
|          25 | AUTOBÚS          | 04-1111111 |
|          27 | AUTOBÚS          | 04-2222222 |
|          28 | CHK              | 04-3333333 |
|          29 | SBL              | 04-4444444 |
+ ------------ + ------------ + ------------ +
5 filas en conjunto (0.01 seg)
```

Como puede ver, la segunda versión, que tiene ambas condiciones en la subcláusula `on` y no tiene cláusula `where` , genera los mismos resultados. ¿Qué pasa si ambas condiciones se colocan en la cláusula `where` pero la cláusula `from` todavía usa la sintaxis de unión ANSI?

```
mysql> SELECCIONAR a.account_id, a.product_cd, c.fed_id
    -> DE la cuenta un cliente INNER JOIN c
    -> DONDE a.cust_id = c.cust_id
    -> Y c.cust_type_cd = 'B';
+ ------------ + ------------ + ------------ +
| account_id | product_cd | fed_id |
+ ------------ + ------------ + ------------ +
|          24 | CHK              | 04-1111111 |
|          25 | AUTOBÚS          | 04-1111111 |
|          27 | AUTOBÚS          | 04-2222222 |
|          28 | CHK              | 04-3333333 |
|          29 | SBL              | 04-4444444 |
+ ------------ + ------------ + ------------ +
5 filas en conjunto (0.01 seg)
```

Una vez más, el servidor MySQL ha generado el mismo conjunto de resultados. Depende de ti poner sus condiciones en el lugar adecuado para que sus consultas sean fáciles de entender y mantener.

Prueba tus conocimientos

Los siguientes ejercicios están diseñados para probar su comprensión de las uniones internas. Por favor consulte el Apéndice C para conocer las soluciones a estos ejercicios.

Ejercicio 5-1

Complete los espacios en blanco (indicados por < # >) para la siguiente consulta para obtener los resultados que seguir:

```
mysql> SELECT e.emp_id, e.fname, e.lname, b.name
    -> DESDE el empleado e INNER JOIN <1> b
    -> ON e.assigned_branch_id = b. <2>;
+ -------- + ---------- + ----------- + --------------- +
| emp_id | fname | lname | nombre                                          |
+ -------- + ---------- + ----------- + --------------- +
| 1 | Michael | Smith | Sede |
| 2 | Susan | Barker | Sede |
| 3 | Robert | Tyler | Sede |
| 4 | Susan | Hawthorne | Sede |
```


re ldt W B kC

Página 18

```
| 5 | John | Gooding | Sede |
| 6 | Helen | Fleming | Sede |
| 7 | Chris | Tucker | Sede |
| 8 | Sarah | Parker | Sede |
| 9 | Jane | Grossman | Sede |
| 10 | Paula | Roberts | Sucursal Woburn |
| 11 | Thomas | Ziegler | Sucursal Woburn |
| 12 | Samantha | Jameson | Sucursal Woburn |
| 13 | John | Blake | Sucursal de Quincy |
| 14 | Cindy | Mason | Sucursal de Quincy |
| 15 | Frank | Portman | Sucursal de Quincy |
| 16 | Theresa | Markham | Entonces. Sucursal NH |
| 17 | Beth | Fowler | Entonces. Sucursal NH |
| 18 | Rick | Tulman | Entonces. Sucursal NH |
+ -------- + ---------- + ----------- + --------------- +
18 filas en conjunto (0.03 seg)
```

Ejercicio 5-2

Escriba una consulta que devuelva el ID de cuenta para cada cliente no comercial (customer.cust_type_cd = 'I') con la identificación federal del cliente (customer.fed_id) y el nombre del producto en el que se basa la cuenta (product.name).

Ejercicio 5-3

Construya una consulta que encuentre a todos los empleados cuyo supervisor esté asignado a un Departamento. Recupere la identificación, el nombre y el apellido de los empleados.

re ldt W B kC