

Bases de Datos I

22/09/2020

Resumen Manual SQL 5.7 22/09

IS-501

José Inestroza

Capítulo 03 Tutorial

3.1 Conexión y desconexión del servidor

Para conectarse al servidor, normalmente deberá proporcionar un nombre de usuario de MySQL cuando invoque mysql

y, muy probablemente, una contraseña. Si el servidor se ejecuta en una máquina distinta a aquella en la que inició sesión, también deberá especificar un nombre de host.

```
shell> mysql -h host -u user -p  
Enter password: *****
```

Host y usuario representan el nombre de host donde se ejecuta su servidor MySQL y el nombre de usuario de su Cuenta MySQL. Si está iniciando sesión en la misma máquina en la que se está ejecutando MySQL, puede omitir el host y simplemente utilice lo siguiente:

```
shell> mysql -u user -p
```

Si, cuando intenta iniciar sesión, recibe un mensaje de error como ERROR 2002 (HY000): No se puede conectarse al servidor MySQL local a través del socket '/tmp/mysql.sock' (2), significa que el daemon del servidor MySQL (Unix) o el servicio (Windows) no se está ejecutando. Algunas instalaciones de MySQL permiten a los usuarios conectarse como usuario anónimo (sin nombre) al servidor ejecutándose en el host local. Una vez que se haya conectado correctamente, puede desconectarse en cualquier momento escribiendo SALIR (cntrl + D \ quit) en mysql.

3.2 Introducción de consultas

Aquí hay una consulta simple que le pide al servidor que le diga su número de versión y la fecha actual:

```
mysql> SELECT VERSION(), CURRENT_DATE;  
+-----+-----+  
| VERSION() | CURRENT_DATE |  
+-----+-----+  
| 5.7.1-m4-log | 2012-12-25 |  
+-----+-----+  
1 row in set (0.01 sec)  
mysql>
```

Una consulta normalmente consta de una instrucción SQL seguida de un punto y coma. (Hay algunas excepciones donde se puede omitir un punto y coma. QUIT).

mysql muestra la salida de la consulta en forma tabular (filas y columnas). La primera fila contiene etiquetas para las columnas. Las siguientes filas son los resultados de la consulta. Normalmente, las etiquetas de columna son los nombres de columnas que obtiene de las tablas de la base de datos. Si está recuperando el valor de una expresión en lugar de un columna de la tabla (como en el ejemplo que se acaba de mostrar), mysql etiqueta la columna usando la expresión misma.

Las palabras clave se pueden introducir en cualquier tipo de letra. Las siguientes consultas son equivalentes:

```
mysql> SELECT VERSION(), CURRENT_DATE;
mysql> select version(), current_date;
mysql> SeLeCt vErSiOn(), current_DATE;
```

Aquí hay otra consulta. Demuestra que puede usar mysql como una calculadora simple:

```
mysql> SELECT SIN(PI()/4), (4+1)*5;
+-----+-----+
| SIN(PI()/4) | (4+1)*5 |
+-----+-----+
| 0.70710678118655 | 25 |
+-----+-----+
1 row in set (0.02 sec)
```

No es necesario que una consulta se proporcione en una sola línea, por lo que las consultas largas que requieren varias líneas no son un problema. MySQL determina dónde termina su declaración buscando el punto y coma final, no por buscando el final de la línea de entrada. (En otras palabras, mysql acepta entrada de formato libre: recopila líneas de entrada pero no los ejecuta hasta que ve el punto y coma). Aquí hay una declaración simple de varias líneas:

```
mysql> SELECT
    -> USER()
    -> ,
    -> CURRENT_DATE;
+-----+-----+
| USER() | CURRENT_DATE |
+-----+-----+
| jon@localhost | 2010-08-06 |
+-----+-----+
```

Así es como mysql indica que aún no ha visto una declaración completa y es esperando el resto. Si usas eso comentarios, siempre puede estar al tanto de lo que está esperando mysql. Si decide que no desea ejecutar una consulta que está en proceso de ingresar, cáncélela escribiendo \C:

```
mysql> SELECT
    -> USER()
    -> \c
mysql>
```

La siguiente tabla muestra cada una de las indicaciones que puede ver y resume lo que significan sobre el estado en el que se encuentra mysql.

Prompt	Meaning
mysql>	Ready for new query
->	Waiting for next line of multiple-line query
'>	Waiting for next line, waiting for completion of a string that began with a single quote (')
">	Waiting for next line, waiting for completion of a string that began with a double quote (")
`>	Waiting for next line, waiting for completion of an identifier that began with a backtick (`)
/*>	Waiting for next line, waiting for completion of a comment that began with /*

Las indicaciones '>' y '>' ocurren durante la recopilación de cadenas (otra forma de decir que MySQL está esperando la finalización de una cadena). En MySQL, puede escribir cadenas rodeadas por' o" caracteres (por ejemplo, 'hola' o " adiós "), y mysql le permite ingresar cadenas que abarcan varias líneas. Cuando ve un mensaje '> o ">', significa que ha ingresado una línea que contiene una cadena que comienza con un carácter de comilla' o", pero aún no ha ingresado la comilla coincidente que termina la cadena. Esto a menudo indica que ha omitido inadvertidamente un carácter de cita. Por ejemplo:

```
mysql> SELECT * FROM my_table WHERE name = 'Smith AND age < 30;  
      '>
```

3.3 Crear y usar una base de datos

Una vez que sepa cómo ingresar declaraciones SQL, estará listo para acceder a una base de datos. Suponga que tiene varias mascotas en su casa (su colección de animales) y le gustaría realizar un seguimiento de varios tipos de información sobre ellos. Puede hacerlo creando tablas para almacenar sus datos y cargar ellos con la información deseada. Luego, puede responder diferentes tipos de preguntas sobre sus animales recuperando datos de las tablas. Esta sección le muestra cómo realizar las siguientes operaciones:

- Crea una base de datos
- Crea una tabla
- Cargar datos en la tabla
- Recuperar datos de la tabla de varias formas
- Utilice varias tablas

Utilice la instrucción SHOW para averiguar qué bases de datos existen actualmente en el servidor:

```
mysql> SHOW DATABASES;  
+-----+  
| Database |  
+-----+  
| mysql   |  
| test    |  
| tmp     |  
+-----+
```

Si la base de datos de prueba existe, intente acceder a ella:

```
mysql> USE test  
Database changed
```

Puede usar la base de datos de prueba (si tiene acceso a ella) para los ejemplos que siguen, pero cualquier otra persona que tenga acceso a ella puede eliminar cualquier cosa que cree en esa base de datos. Por esta razón, probablemente debería pedirle permiso a su administrador de MySQL para usar una base de datos propia. Suponga que quiere llamar a su casa de fieras. El administrador debe ejecutar una declaración como esta:

```
mysql> GRANT ALL ON menagerie.* TO 'your_mysql_name'@'your_client_host';
```

3.3.1 Crear y seleccionar una base de datos

Si el administrador crea su base de datos por usted cuando configura sus permisos, puede comenzar a usar eso. De lo contrario, debe crearlo usted mismo:

```
mysql> CREATE DATABASE menagerie;
```

La creación de una base de datos no la selecciona para su uso; debes hacerlo explícitamente. Para hacer de la colección de animales la corriente base de datos, use esta declaración:

```
mysql> USE menagerie  
Database changed
```

3.3.2 Crear una tabla

Crear la base de datos es la parte fácil, pero en este punto está vacía, como le dice SHOW TABLES:

```
mysql> SHOW TABLES;  
Empty set (0.00 sec)
```

Use una declaración CREATE TABLE para especificar el diseño de su tabla:

```
mysql> CREATE TABLE pet (name VARCHAR(20), owner VARCHAR(20),  
species VARCHAR(20), sex CHAR(1), birth DATE, death DATE);
```

Para verificar que su tabla se creó de la manera que esperaba, use una declaración DESCRIBE:

```
mysql> DESCRIBE pet;  
+-----+-----+-----+-----+-----+  
| Field | Type | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+  
| name | varchar(20) | YES | | NULL | |  
| owner | varchar(20) | YES | | NULL | |  
| species | varchar(20) | YES | | NULL | |  
| sex | char(1) | YES | | NULL | |  
| birth | date | YES | | NULL | |  
| death | date | YES | | NULL | |  
+-----+-----+-----+-----+-----+
```

3.3.3 Carga de datos en una tabla

Después de crear su tabla, debe completarla. Las sentencias LOAD DATA e INSERT son útiles para esto. Suponga que los registros de su mascota se pueden describir como se muestra aquí. (Observe que MySQL espera fechas en formato 'AAAA-MM-DD'; esto puede diferir de lo que está acostumbrado).

name	owner	species	sex	birth	death
Fluffy	Harold	cat	f	1993-02-04	
Claws	Gwen	cat	m	1994-03-17	
Buffy	Harold	dog	f	1989-05-13	
Fang	Benny	dog	m	1990-08-27	
Bowser	Diane	dog	m	1979-08-31	1995-07-29
Chirpy	Gwen	bird	f	1998-09-11	
Whistler	Gwen	bird		1997-12-09	
Slim	Benny	snake	m	1996-04-29	

Puede crear un archivo de texto pet.txt que contenga un registro por línea, con valores separados por tabulaciones y en el orden en que se enumeran las columnas en la declaración CREATE TABLE. Para valores perdidos (como sexos desconocidos o fechas de muerte para animales que aún viven), puede usar valores NULL. Para representarlos en su archivo de texto, use \N (barra invertida, N mayúscula). Por ejemplo, el registro de Whistler the bird se vería así (donde el espacio en blanco entre los valores es un carácter de tabulación única):

```
Whistler      Gwen      bird      \N      1997-12-09      \N
```

Para cargar el archivo de texto pet.txt en la tabla de mascotas, use esta declaración:

```
mysql> LOAD DATA LOCAL INFILE '/path/pet.txt' INTO TABLE pet;
```

Cuando desee agregar nuevos registros uno a la vez, la instrucción INSERT es útil. En su forma más simple, usted proporciona valores para cada columna, en el orden en que se enumeraron las columnas en la instrucción CREATE TABLE. Supongamos que Diane tiene un nuevo hámster llamado "Puffball". Puede agregar un nuevo registro usando una instrucción INSERT como esta:

```
mysql> INSERT INTO pet
    VALUES ('Puffball', 'Diane', 'hamster', 'f', '1999-03-30', NULL);
```

3.3.4 Recuperar información de una tabla

La instrucción SELECT se usa para extraer información de una tabla. La forma general de la declaración es:

```
SELECT what_to_select
FROM which_table
WHERE conditions_to_satisfy;
```

3.3.4.1 Seleccionar todos los datos

La forma más simple de SELECT recupera todo de una tabla:

```
mysql> SELECT * FROM pet;
+-----+-----+-----+-----+-----+
| name | owner | species | sex | birth      | death     |
+-----+-----+-----+-----+-----+
| Fluffy | Harold | cat   | f   | 1993-02-04 | NULL      |
| Claws  | Gwen   | cat   | m   | 1994-03-17 | NULL      |
| Buffy  | Harold | dog   | f   | 1989-05-13 | NULL      |
| Fang   | Benny  | dog   | m   | 1990-08-27 | NULL      |
| Bowser | Diane  | dog   | m   | 1979-08-31 | 1995-07-29 |
| Chirpy | Gwen   | bird  | f   | 1998-09-11 | NULL      |
| Whistler | Gwen | bird  | NULL | 1997-12-09 | NULL      |
| Slim   | Benny  | snake | m   | 1996-04-29 | NULL      |
| Puffball | Diane | hamster | f   | 1999-03-30 | NULL      |
+-----+-----+-----+-----+-----+
```

Edite el archivo pet.txt para corregir el error, luego vacíe la tabla y vuelva a cargarlo usando DELETE y LOAD DATA:

```
mysql> DELETE FROM pet;
mysql> LOAD DATA LOCAL INFILE '/path/pet.txt' INTO TABLE pet;
```

Corrija solo el registro erróneo con una instrucción UPDATE:

```
mysql> UPDATE pet SET birth = '1989-08-31' WHERE name = 'Bowser';
```

3.3.4.2 Seleccionar filas particulares

Como se muestra en la sección anterior, es fácil recuperar una tabla completa. Simplemente omita la cláusula WHERE de la instrucción SELECT. Pero, por lo general, no desea ver la tabla completa, especialmente cuando se vuelve grande. En cambio, generalmente está más interesado en responder una pregunta en particular, en cuyo caso especifica algunas restricciones sobre la información que desea. Veamos algunas consultas de selección en términos de preguntas sobre sus mascotas que responden. Puede seleccionar solo filas particulares de su tabla. Por ejemplo, si desea verificar el cambio que realizó en la fecha de nacimiento de Bowser, seleccione el registro de Bowser de esta manera:

```
mysql> SELECT * FROM pet WHERE name = 'Bowser';
+-----+-----+-----+-----+-----+
| name | owner | species | sex | birth       | death      |
+-----+-----+-----+-----+-----+
| Bowser | Diane | dog     | m   | 1989-08-31 | 1995-07-29 |
+-----+-----+-----+-----+-----+
```

Puede especificar condiciones en cualquier columna, no solo en el nombre. Por ejemplo, si quiere saber qué animales nacieron durante o después de 1998, pruebe la columna de nacimiento:

```
mysql> SELECT * FROM pet WHERE birth >= '1998-1-1';
+-----+-----+-----+-----+-----+
| name | owner | species | sex | birth       | death      |
+-----+-----+-----+-----+-----+
| Chirpy | Gwen | bird    | f   | 1998-09-11 | NULL      |
| Puffball | Diane | hamster | f   | 1999-03-30 | NULL      |
+-----+-----+-----+-----+-----+
```

Puede combinar condiciones, por ejemplo, para localizar perras:

```
mysql> SELECT * FROM pet WHERE species = 'dog' AND sex = 'f';
+-----+-----+-----+-----+-----+
| name | owner | species | sex | birth       | death      |
+-----+-----+-----+-----+-----+
| Buffy | Harold | dog     | f   | 1989-05-13 | NULL      |
+-----+-----+-----+-----+-----+
```

La consulta anterior utiliza el operador lógico AND. También hay un operador OR:

```
mysql> SELECT * FROM pet WHERE species = 'snake' OR species = 'bird';
+-----+-----+-----+-----+-----+
| name | owner | species | sex | birth       | death      |
+-----+-----+-----+-----+-----+
| Chirpy | Gwen | bird    | f   | 1998-09-11 | NULL      |
| Whistler | Gwen | bird    | NULL | 1997-12-09 | NULL      |
| Slim | Benny | snake   | m   | 1996-04-29 | NULL      |
+-----+-----+-----+-----+-----+
```

AND y OR pueden estar entremezclados, aunque AND tiene mayor precedencia que OR. Si usa ambos operadores, es una buena idea usar paréntesis para indicar explícitamente cómo deben agruparse las condiciones:

```
mysql> SELECT * FROM pet WHERE (species = 'cat' AND sex = 'm')
          OR (species = 'dog' AND sex = 'f');
+-----+-----+-----+-----+-----+
| name | owner | species | sex | birth       | death      |
+-----+-----+-----+-----+-----+
| Claws | Gwen | cat     | m   | 1994-03-17 | NULL      |
| Buffy | Harold | dog     | f   | 1989-05-13 | NULL      |
+-----+-----+-----+-----+-----+
```

3.3.4.3 Seleccionar columnas particulares

Si no desea ver filas enteras de su tabla, simplemente nombre las columnas en las que está interesado, separadas por comas. Por ejemplo, si desea saber cuándo nacieron sus animales, seleccione las columnas de nombre y nacimiento:

```
mysql> SELECT name, birth FROM pet;
```

Para minimizar la salida, recupere cada registro de salida único solo una vez agregando la palabra clave DISTINCT:

```
mysql> SELECT DISTINCT owner FROM pet;
```

3.3.4.4 Ordenar filas

A menudo es más fácil examinar el resultado de la consulta cuando las filas se ordenan de una manera significativa. Para ordenar un resultado, use una cláusula ORDER BY. Aquí están los cumpleaños de los animales, ordenados por fecha:

```
mysql> SELECT name, birth FROM pet ORDER BY birth;
```

El orden de clasificación predeterminado es ascendente, con los valores más pequeños primero. Para ordenar en orden inverso (descendente), agregue la palabra clave DESC al nombre de la columna por la que está ordenando:

```
mysql> SELECT name, birth FROM pet ORDER BY birth DESC;
```

3.3.4.5 Cálculos de fecha

MySQL proporciona varias funciones que puede utilizar para realizar cálculos en fechas, por ejemplo, para calcular edades o extraer partes de fechas. Para determinar cuántos años tiene cada una de sus mascotas, use la función TIMESTAMPDIFF(). Sus argumentos son la unidad en la que desea que se exprese el resultado y las dos fechas para las que tomar la diferencia. La siguiente consulta muestra, para cada mascota, la fecha de nacimiento, la fecha actual y la edad en años. Se utiliza un alias (edad) para que la etiqueta de la columna de salida final sea más significativa.

```
mysql> SELECT name, birth, CURDATE(),
    TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age
    FROM pet;
```

La consulta funciona, pero el resultado podría escanearse más fácilmente si las filas se presentaran en algún orden. Esto se puede hacer agregando una cláusula ORDER BY name para ordenar la salida por nombre:

```
mysql> SELECT name, birth, CURDATE(),
    TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age
    FROM pet ORDER BY name;
```

Para ordenar la salida por edad en lugar de por nombre, simplemente use una cláusula ORDER BY diferente:

```
mysql> SELECT name, birth, CURDATE(),
    TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age
    FROM pet ORDER BY age;
```

¿Qué pasa si quieres saber qué animales cumplen años el próximo mes? Para este tipo de cálculo, el año y el día son irrelevantes; simplemente desea extraer la parte del mes de la columna de nacimiento. MySQL proporciona varias funciones para extraer partes de fechas, como YEAR(), MONTH() y DAYOFMONTH(). MES() es la función apropiada aquí. Para ver cómo funciona, ejecute una consulta simple que muestre el valor de nacimiento y MES(nacimiento):

```
mysql> SELECT name, birth, MONTH(birth) FROM pet;
+-----+-----+-----+
```

Encontrar animales con cumpleaños en el próximo mes también es sencillo. Suponga que el mes actual es abril. Entonces, el valor del mes es 4 y puede buscar animales nacidos en mayo (mes 5) así:

```
mysql> SELECT name, birth FROM pet WHERE MONTH(birth) = 5;
```


