Técnico en

DESARROLLO DE SOFTWARE

Bases de Datos II

(CC BY-NC-ND 4.0) Attribution-NonCommercial-NoDerivatives 4.0 International

Atribución

Usted debe reconocer el crédito de una obra de manera adecuada, proporcionar un enlace a la licencia, e indicar si se han realizado cambios. Puede hacerlo en cualquier forma razonable, pero no de forma tal que sugiera que tiene el apoyo del licenciante o lo recibe por el uso que hace.

No Comercial

Usted no puede hacer uso del material con fines comerciales.

Sin obra derivada

Si usted mezcla, transforma o crea un nuevo material a partir de esta obra, no puede distribuir el material modificado.

No hay restricciones adicionales - Usted no puede aplicar términos legales ni medidas tecnológicas que restrinjan legalmente a otros hacer cualquier uso permitido por la licencia.

http://creativecommons.org/licenses/by-nc-nd/4.0/

Bases de Datos II

Unidad I

Administración en MySQL

1. Gestión de usuarios

Al momento de instalar el motor de base de datos se crea el usuario 'root', que es el administrador, y que dispone de todos los privilegios disponibles en **MySQL**.

Sin embargo, normalmente no será una buena práctica dejar que todos los usuarios con acceso al servidor tengan todos los privilegios. Para conservar la integridad de los datos y de las estructuras será conveniente que sólo algunos usuarios puedan realizar determinadas tareas, y que otras, que requieren mayor conocimiento sobre las estructuras de bases de datos y tablas, sólo puedan realizarse por un número limitado y controlado de usuarios.

Los conceptos de usuarios y privilegios están íntimamente relacionados. No se pueden crear usuarios sin asignarle al mismo tiempo privilegios. De hecho, la necesidad de crear usuarios está ligada a la necesidad de limitar las acciones que tales usuarios pueden llevar a cabo.

MySQL permite definir diferentes usuarios y, además, asignar a cada uno determinados privilegios en distintos niveles o categorías de ellos.

Niveles de privilegios

En **MySQL** existen cinco niveles distintos de privilegios:

- **Globales:** se aplican al conjunto de todas las bases de datos en un servidor. Es el nivel más alto de privilegio, en el sentido de que su ámbito es el más general.
- **De base de datos:** se refieren a bases de datos individuales, y por extensión, a todos los objetos que contiene cada base de datos.
- **De tabla:** se aplican a tablas individuales, y por lo tanto, a todas las columnas de esas tabla.
- **De columna:** se aplican a una columna en una tabla concreta.
- **De rutina:** se aplican a los procedimientos almacenados. Aún no hemos visto nada sobre este tema, pero en **MySQL** se pueden almacenar procedimientos consistentes en varias consultas SQL.

Crear usuarios

Aunque en la versión 5.0.2 de **MySQL** existe una sentencia para crear usuarios, CREATE USER, en versiones anteriores se usa exclusivamente la sentencia GRANT para crearlos. En general es preferible usar GRANT, ya que si se crea un usuario mediante CREATE USER, posteriormente hay que usar una sentencia GRANT para concederle privilegios. Usando GRANT podemos crear un usuario y al mismo tiempo concederle también los privilegios que tendrá. La sintaxis simplificada que usaremos para GRANT, sin preocuparnos de temas de cifrados seguros que dejaremos ese tema para capítulos avanzados, es:

```
GRANT priv_type [(column_list)] [, priv_type [(column_list)]] ...
    ON
    TO user [IDENTIFIED BY [PASSWORD] 'password']
        [, user [IDENTIFIED BY [PASSWORD] 'password']] ...
```

La primera parte *priv_type [(column_list)]* permite definir el tipo de privilegio concedido para determinadas columnas. La segunda *ON {tbl_name | * | *.* | db_name.*}*, permite conceder privilegios en niveles globales, de base de datos o de tablas.

Para crear un usuario sin privilegios usaremos la sentencia:

```
mysql> GRANT USAGE ON *.* TO anonimo IDENTIFIED BY 'clave';
Query OK, 0 rows affected (0.02 sec)
```

Hay que tener en cuenta que la contraseña se debe introducir entre comillas de forma obligatoria.

Un usuario 'anónimo' podrá abrir una sesión **MySQL** mediante una orden:

```
C:\mysql -h localhost -u anonimo -p
```

Para conceder privilegios globales se usa *ON *.**, para indicar que los privilegios se conceden en todas las tablas de todas las bases de datos.

Para conceder privilegios en bases de datos se usa *ON nombre_db.**, indicando que los privilegios se conceden sobre todas las tablas de la base de datos 'nombre_db'.

Usando *ON nombre_db.nombre_tabla*, concedemos privilegios de nivel de tabla para la tabla y base de datos especificada.

En cuanto a los privilegios de columna, para concederlos se usa la sintaxis *tipo_privilegio (lista_de_columnas), [tipo_privilegio (lista_de_columnas)]*.

Otros privilegios que se pueden conceder son:

- ALL: para conceder todos los privilegios.
- CREATE: permite crear nuevas tablas.
- DELETE: permite usar la sentencia DELETE.
- DROP: permite borrar tablas.
- INSERT: permite insertar datos en tablas.
- UPDATE: permite usar la sentencia UPDATE.

Para ver una lista de todos los privilegios existentes consultar la sintaxis de la sentencia GRANT.

Se pueden conceder varios privilegios en una única sentencia. Por ejemplo:

```
mysql> GRANT SELECT, UPDATE ON prueba.gente TO anonimo IDENTIFIED BY 'clave';
Query OK, 0 rows affected (0.22 sec)

mysql>
```

Un detalle importante es que para crear usuarios se debe tener el privilegio GRANT OPTION, y que sólo se pueden conceder privilegios que se posean.

Revocar privilegios

Para revocar privilegios se usa la sentencia REVOKE.

```
REVOKE priv_type [(column_list)] [, priv_type [(column_list)]] ...
    ON
    FROM user [, user] ...
```

La sintaxis es similar a la de GRANT, por ejemplo, para revocar el privilegio SELECT de nuestro usuario 'anonimo', usaremos la sentencia:

```
mysql> REVOKE SELECT ON prueba.gente FROM anonimo;
Query OK, 0 rows affected (0.05 sec)
```

Mostrar los privilegios de un usuario

Podemos ver qué privilegios se han concedido a un usuario mediante la sentencia SHOW GRANTS. La salida de esta sentencia es una lista de sentencias GRANT que se deben ejecutar para conceder los privilegios que tiene el usuario. Por ejemplo:

```
mysql> SHOW GRANTS FOR anonimo;
+-----------------------------------------------------------------------+
| Grants for anonimo@%                                                  |
+-----------------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'anonimo'@'%' IDENTIFIED BY PASSWORD '*5...' |
| GRANT SELECT ON `prueba`.`gente` TO 'anonimo'@'%'                     |
+-----------------------------------------------------------------------+
2 rows in set (0.00 sec)

mysql>
```

Nombres de usuarios y contraseñas

Como podemos ver por la salida de la sentencia SHOW GRANTS, el nombre de usuario no se limita a un nombre simple, sino que tiene dos partes. La primera consiste en un nombre de usuario, en nuestro ejemplo 'anonimo'. La segunda parte, que aparece separada de la primera por el carácter '@' es un nombre de máquina (host). Este nombre

puede ser bien el de una máquina, por ejemplo, 'localhost' para referirse al ordenador local, o cualquier otro nombre, o bien una ip.

La parte de la máquina es opcional, y si como en nuestro caso, no se pone, el usuario podrá conectarse desde cualquier máquina. La salida de SHOW GRANTS lo indica usando el comodín '%' para el nombre de la máquina.

Si creamos un usuario para una máquina o conjunto de máquinas determinado, ese usuario no podrá conectar desde otras máquinas. Por ejemplo:

```
mysql> GRANT USAGE ON * TO anonimo@localhost IDENTIFIED BY 'clave';
Query OK, 0 rows affected (0.00 sec)
```

Un usuario que se identifique como 'anonimo' sólo podrá entrar desde el mismo ordenador donde se está ejecutando el servidor.

En este otro ejemplo:

```
mysql> GRANT USAGE ON * TO anonimo@10.28.56.15 IDENTIFIED BY 'clave';
Query OK, 0 rows affected (0.00 sec)
```

El usuario 'anonimo' sólo puede conectarse desde un ordenador cuyo IP sea '10.28.56.15'. Aunque asignar una contraseña es opcional, por motivos de seguridad es recomendable asignar siempre una. La contraseña se puede escribir entre comillas simples cuando se crea un usuario, o se puede usar la salida de la función PASSWORD() de forma literal, para evitar enviar la clave en texto legible. Si al añadir privilegios se usa

una clave diferente en la cláusula *IDENTIFIED BY*, sencillamente se sustituye la contraseña por la nueva.

Borrar Usuarios

Para eliminar usuarios se usa la sentencia DROP USER.

No se puede eliminar un usuario que tenga privilegios, por ejemplo:

```
mysql> DROP USER anonimo;
ERROR 1268 (HY000): Can't drop one or more of the requested users
mysql>
```

Para eliminar el usuario primero hay que revocar todos sus privilegios:

```
mysql> SHOW GRANTS FOR anonimo;
+-----------------------------------------------------------------------+
| Grants for anonimo@%                                                  |
+-----------------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'anonimo'@'%' IDENTIFIED BY PASSWORD '*5...' |
| GRANT SELECT ON `prueba`.`gente` TO 'anonimo'@'%'                     |
+-----------------------------------------------------------------------+
2 rows in set (0.00 sec)

mysql> REVOKE SELECT ON prueba.gente FROM anonimo;
Query OK, 0 rows affected (0.00 sec)

mysql> DROP USER anonimo;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

2. Back up

Las copias de seguridad en base de datos en muy importante por lo cual vamos a documentar los tipos de copias que se pueden realizar.

Se pueden clasificar en 2 tipos:

- Físicos
    - Online
    - Offline
- Lógicos
    - Online

Backups físicos

Los backups físicos son aquellos que copian físicamente los ficheros de BD. Existen dos opciones: en frío y en caliente. Se dice que el backup es en frio cuando los ficheros se copian cuando la BD está parada. En caliente es cuando se copian los ficheros con la BD abierta y funcionando.

Backup en Frío

El primer paso es parar la BD en un estado consistente, es decir, una parada normal del servidor. Después se copian los ficheros de datos.

Una buena idea es automatizar todo este proceso con los scripts correspondientes, de modo que no nos olvidemos de copiar ningún fichero.

Como este tipo de backup es una copia de los ficheros de la BD, si estos contienen algún tipo de corrupción, la traspasaremos a la copia de seguridad sin detectarla. Por esto es importante comprobar las copias de seguridad.

Ventajas:

- Más rápido que los backups lógicos
- Menos probabilidades de inconsistencias

Desventajas:

- El servidor no estará disponible temporalmente

Backup en Caliente

Si la BD requiere disponibilidad 24×7, no se pueden realizar backups en frío. Para efectuar un backup en caliente debemos trabajar con la BD en modo online (funcionamiento normal). El procedimiento de backup en caliente es bastante parecido al frío. Pero para evitar inconsistencias en la base de datos es recomendable bloquear las actualizaciones e inserciones.

Así como el backup en frio permitía realizar una copia de toda la BD sin parar el servidor.

Ventajas:

- Más rápido que los backups lógicos
- La base de datos está disponible

Desventajas:

- Alta probabilidad de inconsistencias si no se controla

Backups lógicos

Este tipo de backups se realizan con el servidor en funcionamiento, copian el contenido de la BD pero sin almacenar la posición física de los datos. Se realizan con alguna herramienta que copia los datos y la definición de la BD en un fichero.

Durante el proceso no se deberán realizar actualizaciones ni inserciones puesto que daría lugar a inconsistencias en la base de datos.

Ventajas:

- La base de datos está disponible
- Los backups son portables

Desventajas:

- Lentitud
- Alta probabilidad de inconsistencias si no se controla

En esta parte vamos a mostrar varias formas de hacer una copia de seguridad de una o varias bases de datos en MySQL

Backup en Caliente

Usaremos el comando “mysqldump” que se creó al instalar MySQL, por lo que si se instaló correctamente, deberíamos tenerlo.

Además, aprovechando que tenemos una herramienta gráfica para gestionar MySQL (MySQL Workbench) realizaremos copias de seguridad con ella.

Backup en Frío

Usaremos comandos comunes de Linux como “cp”, “rm”, “mkdir”

Copias lógicas en caliente

- MySQL Enterprise Backup.
- MySQLhotcopy.
- MySQLdump.
- MySQL workbench

3. Logs

En **MySQL** podemos habilitar un **log de queries ejecutadas en el servidor** llamado “**general log**“. Aunque no es muy recomendable tenerlo habilitado normalmente, puede resultar útil en momentos puntuales. Vamos a ver su funcionamiento:

Las variables que controlan el **general log** son las siguientes:

- **general_log**: Indicamos si queremos habilitar o no dicho log
- **general_log_file**: Indicamos el path del fichero dónde queremos el log

Mediante **SHOW VARIABLES** podemos verlos:

```
mysql> show variables like 'general_log%';
+------------------+---------------------------+
| Variable_name    | Value                     |
+------------------+---------------------------+
| general_log      | OFF                       |
| general_log_file | /var/log/mysql/query.log |
+------------------+---------------------------+
2 rows in set (0.00 sec)
```

También a partir de MySQL 5.1 tenemos que tener en cuenta el valor de "**log_output**", ya que podemos indicar que en lugar de un fichero se guarde en una tabla:

```
mysql> show variables like 'log_out%';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| log_output    | FILE  |
+---------------+-------+
1 row in set (0.00 sec)
```

Mediante **SET GLOBAL** podemos **habilitar el general log**:

```
mysql> set global general_log=1;
Query OK, 0 rows affected (0.10 sec)
```

En el fichero veremos cómo **se guardan tanto las queries como las conexiones**:

```
100925 19:19:53 46223 Query     select version()
100925 19:19:57 46223 Quit
100925 19:20:14 46224 Connect   root@localhost on
                46224 Query     select @@version_comment limit 1
100925 19:20:16 46224 Query     select version()
100925 19:20:17 46224 Query     select version()
                46224 Query     select version()
100925 19:20:18 46224 Quit
```

Por otro lado, tenemos también la variable de **session sql_log_off**, la cual tenemos por defecto a **OFF** indicando que no deshabilita por defecto el log. Podemos ver su funcionamiento con la siguiente **sesión de ejemplo**:

```
mysql> select "sql_log_off en off";
+--------------------+
| sql_log_off en off |
+--------------------+
| sql_log_off en off |
+--------------------+
1 row in set (0.00 sec)

mysql> set session sql_log_off=1;
Query OK, 0 rows affected (0.00 sec)

mysql> select "sql_log_off en on";
+-------------------+
| sql_log_off en on |
+-------------------+
| sql_log_off en on |
+-------------------+
1 row in set (0.00 sec)
```

En el **log** quedaría:

```
100925 19:21:41 46226 Query     select "sql_log_off en off"
100925 19:21:54 46226 Query     set session sql_log_off=1
```

Por lo que **veríamos que se deshabilita el log**, pero no las queries que se ejecuten a posteriori. No resulta muy problemática, ya que esta variable únicamente se puede modificar si el usuario tiene el privilegio **SUPER**.

Descargo de responsabilidad

La información contenida en este documento descargable en formato PDF o PPT es un reflejo del material virtual presentado en la versión online del curso. Por lo tanto, su contenido, gráficos, links de consulta, acotaciones y comentarios son responsabilidad exclusiva de su(s) respectivo(s) autor(es) por lo que su contenido no compromete al área de e-Learning del Departamento GES o al programa académico al que pertenece.

El área de e-Learning no asume ninguna responsabilidad por la actualidad, exactitud, obligaciones de derechos de autor, integridad o calidad de los contenidos proporcionados y se aclara que la utilización de este descargable se encuentra limitada de manera expresa para los propósitos educacionales del curso.

Galileo UNIVERSIDAD La Revolución en La Educación

GES Galileo Educational System