Instituto Tecnológico de Villahermosa
Ing. Sistemas Computacionales
6to Semestre

Taller de Base de Datos

Reporte Unidad 4 - Seguridad

Reynaldo Bernard de Dios de la Cruz

Sábado, 16 de Abril de 2016.

Taller de Base de Datos

Contenido

Taller de Base de Datos

Introducción

Cuando una persona está encargada de algunas tablas es un trabajo sencillo, se habla de una base de datos pequeña y relativamente flexible, pero ¿Qué ocurre cuando una organización o institución necesita que múltiples personas puedan manipular o mejor dicho consultar la base de datos. Existen algunas restricciones como lo son las operaciones que pueden hacer y además la base de datos, tablas y columnas, es decir, los diversos niveles de la base de datos.

Por tanto cada persona puede realizar las operaciones que desee con toda la libertad que se le otorgue, desde tener acceso a todas las tablas hasta solo seleccionar una columna. De esta manera cada persona tiene un espacio privado con el cual puede trabajar de manera aislada e independiente de los demás usuarios que se definan para la base de datos.

Un ejemplo claro puede ser un joven que está viendo las noticias en una página de internet, cuando entra a la página se van mostrando todas las noticias actuales a primera vista y las más viejas atrás de ellas. La persona que escribe cada acontecimiento como noticia tiene más autoridad como para intentar rellenar un formulario con Nombre, Fecha y Contenido de la noticia (por dar un ejemplo). Además de guardar el formulario en una base de datos y en todo caso si ha cometido un error poder modificar el Nombre, Fecha y Contenido de la noticia. Nuevamente el autor de la noticia puede guardar los cambios hechos en la base de datos. Entonces es aquí donde vemos la diferencia que existe entre un autor de noticia y un usuario de la noticia. El usuario en este caso podrá ver solo las noticias o ciertas partes que uno haya definido anteriormente. Por ejemplo, puede ver solo el nombre de la noticia y el cuerpo, pero no la fecha de publicación.

Para atender esta problemática se tiene la facilidad en MySQL de crear diversos usuarios con diferentes tipos de privilegios, estos usuarios pueden acceder desde la misma máquina o pueden ser consultores externos de la base de datos y realizar solo las transacciones que el administrador defina para cada uno de estos agentes.

4.1 - Tipos de usuario

Marco teórico

Como una buena práctica de seguridad para el servidor de base de datos es recomendable que no se proporcione la cuenta del root para que se autentique una persona o un sistema desarrollado en algún lenguaje de programación. La alternativa de solución es la creación de usuarios que tengan asignados ciertos privilegios sobre las bases de datos almacenadas en nuestro servidor. En otras palabras, con MySQL podemos crear usuarios limitados en las acciones que pueden realizar sobre el servidor de base de datos.

MySQL ofrece 5 niveles de privilegios que se le pueden asignar a los usuarios que se creen dentro del servidor de base de datos:

- **Globales**: es el nivel más alto de privilegios ya que se aplican al conjunto de todas las bases de datos del servidor.
- **Base de datos**: se aplican a una base de datos en particular y a todos los objetos que la componen.
- **Tabla**: se aplican a una tabla en particular y a todas las columnas que componen dicha tabla
- **Columna**: se aplica a una columna en una tabla en particular
- **Rutina**: se aplican sobre los procedimientos almacenados creados en una base de datos.

4.2 - Creación de Usuarios

Marco teórico

Comando GRANT

MySQL es un sistema de gestión de bases de datos claramente orientado a la web, y una de los síntomas en su arquitectura ha venido siendo que la creación de los usuarios se realiza en la misma sentencia que el permiso (grant) de acceso a una o varias bases de datos. La orientación de MySQL va cambiando con el tiempo y el uso que se le da a las bases de datos cada vez trasciende más el entorno web, la forma clásica, con la sentencia **GRANT**.

Utilizando la sentencia GRANT podemos crear un usuario a la par que otorgarle uno o varios privilegios sobre los objetos de una base de datos, o la base de datos completa. Al encontrarse una sentencia de tipo GRANT, el motor de MySQL revisa si el usuario existe previamente para el contexto que estamos asignándole permisos, y si dicho usuario no está presente en el sistema, lo crea.

Pongamos un ejemplo, queremos crear el usuario adolfo para la base de datos test:
Nos conectamos con un usuario que tenga privilegios, root, como propietario de la base de datos, los tiene.

Lanzamos la sentencia GRANT, indicando los permisos que otorgamos, la base de datos y los objetos de la misma sobre los que estamos asignando privilegios, el nombre del usuario y el password:

```
mysql> GRANT SELECT, INSERT ON test.* TO 'adolfo'@'localhost' IDENTIFIED BY 'pass_adolfo';
```

En este ejemplo permitimos al usuario adolfo que seleccione (SELECT) e inserte (INSERT) en todos los objetos (*) de la base de datos test, además indicamos que el contexto sea la máquina local de la base de datos (localhost), lo que impedirá que el usuario se conecte desde otras máquinas, y finalmente asignamos un password mediante IDENTIFIED BY.

Si quisiéramos que el usuario no tuviera un password, deberemos omitir la cláusula IDENTIFIED BY.

En el caso de que el modo SQL del servidor estuviera en NO_AUTO_CREATE_USER, la creación de usuarios no estaría permitida a no ser que tuvieran asignado un password no vacío.

Una vez hecho esto, podremos conectarnos con nuestro usuario y realizar las acciones para las que hemos asignado permisos:

```
$ mysql -u adolfo -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 19
Server version: 5.0.67 Source distribution
mysql> use test;
Database changed
mysql> select * from frutas;
+-----------+----------+
| nombre    | color    |
+-----------+----------+
| fresa     | rojo     |
| manzana   | verde    |
| uva       | verde    |
+-----------+----------+
3 rows in set (0,03 sec)
```

Comando CREATE USER

A partir de la versión MySQL 5.0.2 existe la posibilidad de crear usuarios sin necesidad de asignarles privilegios, utilizando la sentencia CREATE USER.

Por ejemplo, para crear el usuario fernando:

```
$ mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 20
Server version: 5.0.67 Source distribution
mysql> CREATE USER 'fernando'@'localhost' IDENTIFIED BY 'fer_pass';
Query OK, 0 rows affected (0,00 sec)
```

Al igual que con la sentencia GRANT, el contexto 'localhost' define que el usuario solamente se puede conectar desde el servidor de MySQL, y el IDENTIFIED BY define el password del usuario, se puede omitir, para un usuario sin password, siempre que el modo SQL no sea NO_AUTO_CREATE_USER.

Conexión con el usuario, utilizando la opción -p:

```
$ mysql -u fernando -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 22
Server version: 5.0.67 Source distribution
```

Los privilegios necesarios para ejecutar la sentencia CREATE USER son CREATE USER o bien INSERT en la base de datos MySQL. El usuario recién creado no tiene privilegio alguno, por lo que deberemos asignarle permisos utilizando sentencias GRANT (esta vez sin la cláusula IDENTIFIED BY).

Comando INSERT

Este es un método que MySQL no recomienda demasiado, es un poco más complejo que los otros dos, pero va bien a la hora de resolver problemas, como que por ejemplo alguno de las formas anteriores esté dando algún problema extraño.

Para ello es necesario un usuario con privilegio INSERT en la base de datos MySQL. También debemos mencionar r que se ha de tener mucho cuidado con esta base de datos, ya que contiene toda la información de usuarios y permisos.

Ejemplo de creación del usuario mariano usando **INSERT** en nuestra base de datos. Nos conectamos con un usuario con privilegios, en este caso root, y seleccionamos la base de datos **MySql** mediante la sentencia **USE**.

```
$ mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 23
Server version: 5.0.67 Source distribution
mysql> use mysql
Database changed
```

Y después realizamos la sentencia de inserción para añadir nuestro usuario:

```
mysql> INSERT INTO user VALUES('localhost','mariano',PASSWORD('pass_mariano'),'Y','Y',
'N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N',
'N','N','N','','','','',0,0,0,0);
Query OK, 1 row affected (0,00 sec)
```

Es necesario llamar a la función PASSWORD() para almacenar el password codificado, en los otros casos, el IDENTIFIED BY se encarga de hacer la codificación.

En este caso se le dan permisos globales de INSERT y SELECT, para saber a qué corresponde cada columna, se puede hacer un DESCRIBE user.

```
mysql> DESCRIBE user;
```

```
+-----------------------+-----------------------------------+------+-----+---------+-------+
| Field                 | Type                              | Null | Key | Default | Extra |
+-----------------------+-----------------------------------+------+-----+---------+-------+
| Host                  | char(60)                          | NO   | PRI |         |       |
| User                  | char(16)                          | NO   | PRI |         |       |
| Password              | char(41)                          | NO   |     |         |       |
| Select_priv           | enum('N','Y')                     | NO   |     | N       |       |
| Insert_priv           | enum('N','Y')                     | NO   |     | N       |       |
| Update_priv           | enum('N','Y')                     | NO   |     | N       |       |
| Delete_priv           | enum('N','Y')                     | NO   |     | N       |       |
| Create_priv           | enum('N','Y')                     | NO   |     | N       |       |
| Drop_priv             | enum('N','Y')                     | NO   |     | N       |       |
| Reload_priv           | enum('N','Y')                     | NO   |     | N       |       |
| Shutdown_priv         | enum('N','Y')                     | NO   |     | N       |       |
(...)
```

Para asignar privilegios a bases de datos específicas o tablas específicas, se debe usar GRANT.

Utilizando este método, tenemos que forzar que se refresquen las tablas de permisos usando **FLUSH PRIVILEGES**.

```
mysql> FLUSH PRIVILEGES;
```

Una vez hecho esto, ya nos podremos conectar:

```
$ mysql -u mariano -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 28
Server version: 5.0.67 Source distribution
```

Marco práctico

Retomando la base de datos previamente hecha, NAUTICA, se establecerán catorce usuarios, de esta manera se puede definir un poco de seguridad a la base de datos y además un control de lo que cada persona quiera hacer.

Primero que nada hay que pensar en que base de datos, tablas y columnas se van a establecer los privilegios, posteriormente los privilegios. Es decir, ¿Dónde y qué? privilegios se establecerán para cada usuario.

La siguiente tabla muestra los privilegios que cada usuario posee.

Taller de Base de Datos

Alcance	Usuario	Privilegios
Todas las bases de datos	Administrador	ALL
	Administrador_Cabecera	SELECT, CREATE, DELETE, DROP, INSERT
	Administrador_Auxilar	SELECT, CREATE, DROP, INSERT
	Moderador_Lider	SELECT, CREATE, DROP
	Moderador_Cabecera	SELECT, CREATE
	Moderador_Auxiliar	SELECT, DROP
	Moderador	SELECT
Únicamente la base de datos NAUTICA.	Colaborador_Veterano	ALL
	Colaborador	SELECT, DELETE, INSERT
	Asistente_Cabecera	SELECT, DELETE
	Asistente_Auxiliar	SELECT, INSERT
	Asistente	SELECT, DROP
	Auditor	SELECT
Unicamente la columna cuota_pago de la tabla barcos en la base de datos nautica. **Nautica.barcos.cuota_pago.**	Contador	SELECT

A continuación se muestran las sentencias para crear a los usuarios.

```
GRANT ALL ON *.* TO Administrador IDENTIFIED BY '12345';

GRANT SELECT, CREATE, DELETE, DROP, INSERT ON *.* TO Administrador_Cabecera IDENTIFIED BY '12345';

GRANT SELECT, CREATE, DROP, INSERT ON *.* TO Administrador_Auxiliar IDENTIFIED BY '12345';

GRANT SELECT, CREATE, DROP ON *.* TO Moderador_Lider IDENTIFIED BY '12345';

GRANT SELECT, CREATE ON *.* TO Moderador_Cabecera IDENTIFIED BY '12345';

GRANT SELECT, DROP ON *.* TO Moderador_Axuliar IDENTIFIED BY '12345';

GRANT SELECT ON *.* TO Moderador IDENTIFIED BY '12345';


GRANT ALL ON nautica.* TO Colaborador_Veterano IDENTIFIED BY '12345';

GRANT SELECT, DELETE, INSERT ON nautica.* TO Colaborador IDENTIFIED BY '12345';

GRANT SELECT, DELETE ON nautica.* TO Asistente_Cabecera IDENTIFIED BY '12345';

GRANT SELECT, INSERT ON nautica.* TO Asistente_Axuliar IDENTIFIED BY '12345';

GRANT SELECT ON nautica.* TO Asistente IDENTIFIED BY '12345';


GRANT SELECT ON nautica.* TO Auditor IDENTIFIED BY '12345';

GRANT SELECT(cuota_pago) ON nautica.barcos TO Contador IDENTIFIED BY '12345';
```

Taller de Base de Datos

Comprobación de privilegios

Para estar seguros que realmente el usuario tiene permisos para hacer solo lo que se indica, se probará al usuario **Contador** con privilegios **SELECT** solo en la base de datos **nautica**, tabla, **barcos** y la columna **cuota_pago**.

Alcance	Usuario	Privilegios
Unicamente la columna cuota_pago de la tabla barcos en la base de datos nautica. **Nautica.barcos.cuota_pago.**	Contador	SELECT

Para ello se inicia sesión desde la consola utilizando **mysql -u USUARIO - p**, a continuación presionar ENTER, como se muestra a continuación.

```
Microsoft Windows [Versión 6.3.9600]

(c) 2013 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Rodolfo>mysql -u Contador -p

Enter password: *****

Welcome to the MySQL monitor.  Commands end with ; or \g.

Your MySQL connection id is 6

Server version: 5.7.11-log MySQL Community Server (GPL)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its

affiliates. Other names may be trademarks of their respective

owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Se muestran las bases de datos usando **SHOW DATABASES;**

```
mysql> show databases;
+--------------------------------+
| Database                       |
+--------------------------------+
| information_schema             |
| nautica                        |
+--------------------------------+
2 rows in set (0.00 sec)
mysql>
```

Taller de Base de Datos

Como solamente se la ha dado permiso a la base de datos NAUTICA entonces esa base de datos se mostrará solamente y las demás que se han creado no se podrán visualizar, debido a que no tiene permisos siquiera sobre las otras bases de datos.

Luego se selecciona la base de datos NAUTICA, a través de **USE NAUTICA**.

```
mysql> use nautica
Database changed
mysql>
```

Posteriormente se ejecuta la consulta para mostrar las tablas, a través de **SHOW TABLES;** .

```
mysql> show tables;
+------------------------------+
| Tables_in_nautica      |
+------------------------------+
| barcos                       |
+------------------------------+
1 row in set (0.00 sec)

mysql>
```

Aunque existan cien tablas diferentes; el usuario no podrá ver las otras tablas si es que no tiene los permisos. En este caso el usuario Contador solamente tiene permisos para consultar los registros en la tabla barcos. Es por ello que solamente llega a visualizar es única tabla.

Por último se describe la tabla para ver que campos puede consultar. Para ello se ejecuta la consulta **DESC BARCOS;**

```
mysql> desc barcos;
+-------------------+-------------------+----------+-------+--------------------+---------+
| Field             | Type              | Null     | Key   | Default            | Extra   |
+-------------------+-------------------+----------+-------+--------------------+---------+
| cuota_pago        | double            | NO       |       | NULL               |         |
+-------------------+-------------------+----------+-------+--------------------+---------+
1 row in set (0.06 sec)

mysql>
```

Como puede observarse la única columna que puede ver es **couta_pago.**

Ahora se pasará a comprobar sus privilegios, ya anteriormente se había definido solamente el privilegio **SELECT.** Por lo que se comprobará ejecutando un **DROP TABLE barcos;**

La consulta arrojaría esto:

```
mysql> drop table barcos;
ERROR 1142 (42000): DROP command denied to user 'Contador'@'localhost' for table 'barcos'
mysql>
```

Esto quiere decir que el comando DROP no está habilitado para el usuario Contador con el servidor localhost, para la tabla barcos.

Ahora hay que probar si efectivamente hace el SELECT. Para ello se ejecutará la siguiente consulta.

```
mysql> SELECT * FROM barcos;
ERROR 1143 (42000): SELECT command denied to user 'Contador'@'localhost' for column 'matricula' in table 'barcos'
mysql>
```

Una vez más el error es evidente, esto quiere decir que el usuario Contador no puede seleccionar todos los campos de la tabla barcos. Pero, si se es más específico se podrá apreciar que solamente se puede seleccionar el campo cuota_pago.

```
mysql> SELECT cuota_pago FROM barcos;
+------------+
| cuota_pago |
+------------+
|    2474.74 |
|    7172.32 |
|    3329.98 |
|     6250.3 |
|    2224.57 |
|    4251.52 |
|     5172.4 |
|     3616.8 |
|     4676.2 |
|    4519.07 |
|    4074.95 |
|    4077.79 |
|    7885.85 |
|    7781.45 |
|    4467.25 |
|    2438.25 |
|     4112.8 |
|    2820.72 |
|     5419.7 |
|    5693.04 |
|    2840.57 |
|    2525.93 |
|     882.44 |
|    3356.65 |
|    1901.07 |
|     6657.9 |
|     915.24 |
|    2440.39 |
|     731.96 |
|     2736.6 |
+------------+
30 rows in set (0.04 sec)
mysql>
```

De esta forma se puede comprobar que realmente hace todo lo que debe hacer y además afecta la base de datos, tabla o columna que se haya definido.

4.3 - Privilegios a usuarios

Aquí está una pequeña lista del resto de los posibles permisos que los usuarios pueden gozar.

- ALL: como mencionamos previamente esto permite a un usuario de MySQL acceder a todas las bases de datos asignadas en el sistema.
- CREATE: permite crear nuevas tablas o bases de datos.
- DROP: permite eliminar tablas o bases de datos.
- DELETE: permite eliminar registros de tablas.
- INSERT: permite insertar registros en tablas.
- SELECT: permite leer registros en las tablas.
- UPDATE: permite actualizar registros seleccionados en tablas.
- GRANT OPTION: permite remover privilegios de usuarios.

Anexo

Link de repositorio el GIT del script escrito en python para generar los datos, la base de datos en un archivo txt y además el reporte en formato PDF.

https://github.com/Reynald0/taller_bd

Sencilla página web que da un resumen del trabajo realizado y su seguimiento.

http://reynald0.github.io/taller_bd/

Taller de Base de Datos

Conclusion

En MySQL, toda la información referente a usuarios, bases de datos y permisos (o privilegios, en jerga MySQL) se encuentra almacenada en diferentes tablas pertenecientes a la base de datos de sistema: mysql.

La tabla user almacena toda la información referente a los usuarios, junto con sus privilegios globales. Los columnas de esta tabla se utilizan para determinar cuándo rechazar o permitir las conexiones entrantes, para cada usuario. Para las conexiones aceptadas, todos los privilegios otorgados a través de esta tabla (columnas que finalizan con la cadena "_priv") se denominan globales y aplican a todas las bases de datos en el servidor.

También es posible seleccionar el campo "Host", el cual indica desde qué host(s) se le permite el acceso a cada usuario.

Internamente el servidor almacena la información de privilegios en diferentes tablas de la base de datos mysql, llamadas grant tables. El servidor MySQL lee el contenido de estas tablas en memoria cuando inicia y toma las decisiones de control de acceso basándose en estas copias de dichas tablas en memoria. Por esta razón es que es necesario ejecutar el comando FLUSH PRIVILEGES (para que el servidor recargue estas tablas nuevamente en memoria) cada vez que se modifican permisos utilizando sentencias INSERT, UPDATE o DELETE.

La tabla db almacena los privilegios a nivel bases de datos. A través de la misma se determina qué usuarios pueden acceder a qué bases de datos desde qué hosts. Las columnas de privilegios determinan las operaciones permitidas. Un privilegio otorgado a nivel base de datos aplica a la base de datos y a todos los objetos en la misma, tales como tablas y procedimientos.

Luego es posible determinar qué tipo de acceso tiene un usuario en particular examinando el resto de las columnas de la tabla db.

De igual forma, es posible examinar qué usuarios tienen privilegios a nivel tablas (examinando la tabla tables_priv) y privilegios a nivel columnas (examinando la tabla columns_priv). Estas tablas son similares a la tabla db, sólo que implementan una mayor granularidad. Un privilegio a nivel tabla aplica a la tabla y todas sus columnas. Un privilegio a nivel columna aplica sólo a la columna indicada. Por otro lado, la tabla procs_priv aplica a los procedimientos almacenados (stored procedures).

Bibliografía

Aula Cic. (2012). El DDL, Lenguaje de Definición de Datos (I). Febrero 18, 2016, de Aula Clic Sitio web: http://www.aulaclic.es/sqlserver/t_8_1.htm

MySQL. (2016). MySQL 5.7 Reference Manual. Febrero 15, 2016, de MySQL Sitio web: http://dev.mysql.com/doc/refman/5.7/en/http://www.desarrolloweb.com/articulos/intro-indices-mysql.html

Conchoi. (2004). Integridad referencial en MySQL . Febrero 15, 2006, de Programacion.net Sitio web: http://programacion.net/articulo/integridad_referencial_en_mysql_263/4