|  |  |
| --- | --- |
| Mirar creación de usuarios y establecimiento de privilegios en workbench | |
| 1. **Sistema de permisos en MYSQL🡪 quién puede conectar al servidor y qué puede hacer cada usuario conectado. 🡪 base de datos mysql.**   **Acceso al servidor: Fase1. Se comprueba identidad**  **Fase 2. Se comprueban permisos.** | |
| **Tablas de permisos** | User y db |
| **Primera fase de conexión** | Tabla *user* |
| **Segunda fase de conexión** | Tablas db, tables\_priv y columns\_priv |
| **Tabla user** | Permisos globales 🡪 para todas las BD  Las columnas **host** y **user** son columnas de alcance 🡪 quién va a conectar y desde dónde |
| **Tabla db** | Si en user está a N pasamos a esta tabla |
| **Tabla tables\_priv** | Si en db está a N pasamos a esta tabla |
| **Tabla columns\_priv** | Si en tables\_priv está a N pasamos a esta tabla |
| **Cambiar un permiso (Ej.)** | update user set select\_priv='N' where host=’localhost’ |
| **Probar a crear un usuario y luego una conexión nueva para ese usuario** | Dar todos los permisos al usuario |
| **Leer de nuevo el contenido de las tablas de permisos** | FLUSH PRIVILEGES; |
| **Consultar permisos de una cuenta dada** | show grants for 'root'@'localhost' |
| **Privilegios en MYSQL** | |
| **Información sobre privilegios en las tablas** | User, db, tables\_priv, columns\_priv |
| **Control de acceso detallado** | |
| **Nivel 1: comprobación de la conexión** | |
| **Nombre de máquina o cliente (o IP) y nombre de usuario** | Tabla user🡪 columnas host, user y password |
| **Todos los privilegios para un usuario** | grant all privileges on db.\* to root@'localhost' |
| **Si concuerdan nombre de cliente y de usuario en dos filas** | El servidor ordena los registros y coge la primera concordancia (nombres más específicos al principio) |
| **Para ver la cuenta de usuario con la que nos hemos conectado** | Select current\_user(); |
| **EJERCICIO:** CONECTAR CON USUARIOS DISTINTOS Y COMPROBAR (SI HACE FALTA CREAR ALGUNOS NUEVOS) | |
| **Nivel 2: Verificación de permisos** | |
| **Comodines en los valores** | % y \_ |
| **Privilegios de administrador (como SHUTDOWN o RELOAD)** | Sólo se comprueba el registro de la tabla USER |
| **Para peticiones sobre bases de datos (INSERT, UPDATE…)** | Tabla User primero y luego Db |
| Ejercicio: denegar acceso a comando UPDATE en USER pero darle acceso al usuario root en la tabla DB y/o HOST | |
| **Mostrar posibles privilegios** | Show privileges |
| **Los cambios de privilegios tienen efecto** | 1. Si usamos GRANT, REVOKE o SET PASSWORD, inmediatamente 2. Si usamos sentencias como INSERT, UPDATE o DELETE hay que reiniciar el servidor o ejecutar FLUSH PRIVILEGES |
| **Probar comando: create user 'pepe'@'localhost'** | ¿Qué permisos se dan por defecto?  ¿Qué tablas se modifican? |
| ***Ejercicios creación de usuarios 1*** | |
| **Gestión de recursos 🡪 cómo gestionar cuentas y permisos** | |
| **Podemos crear cuentas** | 1. Usando GRANT (recomendado) 🡪 como root. 2. Manipulando tablas de permisos. 3. Con MySQLYog o similares. |
| **Ejemplos con GRANT:**  grant all privileges on \*.\* to 'antonio'@'localhost'  identified by 'sierra' with grant option  grant all privileges on \*.\* to 'antonio'@'%'  identified by 'sierra' with grant option  GRANT SELECT (código\_película, título) ON cine.peliculas  TO 'ana'@'localhost'  Son dos cuentas de superusuario  grantselecton \*.\* to 'admin'@'%' (Da el permiso select al usuario admin🡪 probar a crear primero este usuario, ejecutar el commando y comprobar en modo gráfico el efecto) | |
| **Ejemplos con INSERT:**  INSERT INTO user  VALUES (‘localhost’, ‘antonio’, PASSWORD(‘sierra’), ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’);  INSERT INTO user SET Host=’localhost’, User=’admin’, select;\_priv=’Y’  FLUSH PRIVILEGES; | |
| **PASSWORD** | Encripta contraseñas (GRANT no lo necesita) |
| **Creamos una cuenta y le damos acceso a una base de datos específica:** | GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, drop  on biblioteca.\*  to 'usuario'@'localhost'  IDENTIFIED BY 'despintado';  (comprobar cambios en tablas user y db) |
| **Igual que arriba pero utilizando INSERT** | INSERT INTO user (Host, User, Password)  VALUES (‘localhost’, ‘usuario’, PASSWORD(‘despintado’)  INSERT INTO db  (Host, db, user, select\_priv,insert\_priv, update\_priv, delete\_priv, create\_priv, drop\_priv)  VALUES(‘localhost’, ‘biblioteca’,’usuario’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’) |
| **FLUSH PRIVILEGES** | Hace que el servidor vuelva a leer las tablas |
| **REVOKE** | Elimina permisos |
| **Eliminar permisos de un usuario sobre una base de datos** | REVOKE ALL PRIVILEGES on biblioteca.\* FROM usuario@localhost |
| **Eliminar una cuenta** | DROP USER 'usuario'@'localhost' |
| **Limitar recursos de cuentas (tabla USER)** | |
| **Nº de consultas por hora** | Max\_questions |
| **Nº de actualizaciones por hora** | Max\_updates |
| **Nº de conexiones por hora** | Max\_conections |
| **Para todas las cuentas (global)** | Max\_user\_conections |
| **Limitar recursos para cuenta nueva** | GRANT ALL ON biblioteca.\* TO 'admin'@'localhost'  IDENTIFIED BY 'contraseña'  WITH MAX\_QUERIES\_PER\_HOUR 20  MAX\_UPDATES\_PER\_HOUR 10  MAX\_CONNECTIONS\_PER\_HOUR 5  MAX\_USER\_CONNECTIONS 2; |
| **Limitar recursos para cuenta existente** | GRANT USAGE ON \*.\* TO 'admin'@'localhost'  WITH MAX\_QUERIES\_PER\_HOUR 100; |
| **Para eliminar un límite se pone a 0** | GRANT USAGE ON \*.\* TO 'admin'@'localhost'  WITH MAX\_QUERIES\_PER\_HOUR 0; |
| **Reiniciar contadores a 0 para todas las cuentas** | FLUSH USER\_RESOURCES |
| **ASIGNAR CONTRASEÑAS A CUENTAS** | |
| **Como root** | SET PASSWORD FOR 'admin'@'localhost' =PASSWORD('contraseña') |
| **Como usuario normal** | SET PASSWORD=PASSWORD('contraseña') |
| **Otra forma (sin afectar a los permisos actuales de la cuenta)** | GRANT USAGE ON \*.\* TO 'admin'@'localhost' IDENTIFIED BY 'contraseña' |
| **Otra forma (modificando tabla para nuevo usuario)** | INSERT INTO user (Host, user, Password)  VALUES ('%', 'juan', PASSWORD('contraseña'));  FLUSH PRIVILEGES; |
| **Otra forma (modificando tabla para usuario existente)** | UPDATE user SET Password=PASSWORD ('contraseña')  WHERE Host='%' AND User='juan';  FLUSH PRIVILEGES; |
| |  | | --- | | **ACTIVIDADES 3.2.** | | 1. Crea un usuario local nuevo llamado admin con CREATE USER, asígnale la contraseña *admin* encriptándola con la función PASSWORD y dale permisos globales sobre todas las bases de datos con GRANT. Haz cada operación con un comando distinto.      1. Elimina el permiso SUPER y GRANT del usuario *admin* anterior. Usa las dos formas vistas: con REVOKE y de manera manual.      1. Haz lo necesario para que el usuario *admin* pueda conectarse a la base de datos *biblioteca* desde una única IP [www.guara.com](http://www.guara.com/).      1. ¿Qué ocurre si intentamos crear un usuario con contraseña con el comando CREATE USER sin usar la función PASSWORD?, ¿podemos conocer la contraseña original de un usuario?     He creado el usuario “admin2” con el comando CREATE USER sin usar la función PASSWORD y aun así no podemos ver la contraseña.   1. ¿Con qué cinco permisos deberíamos ser más cautelosos a la hora de concederlos y por qué?   Cinco permisos con los que debemos tener cuidado son: create, drop, delete, update y grant option. Estos permisos es importante tenerlos en cuenta ya que si cualquier usuario los tuviese, nos podría malograr el proyecto en cuestión, incluso si tiene privilegios en todas las BBDD, podría estropearnos todas las BBDD.   1. Averigua para qué sirven las funciones *encode, md5 y password* de MySQL.   Estas funciones sirven para hashear las contraseñas, es decir, encriptarlas usando un algoritmo que no es reversible para así poder guardarlas con seguridad sin que nadie las vea en texto plano. | | |