11

第十一讲 数据库安全性管理

教学内容

- 11.1 用户管理
- 11.2 权限管理
- 11.3 角色管理
- 11.4 批量用户管理

用户

- 用户是数据库的使用者和管理者。
- **MySQL**通过用户的设置来控制数据库操作人员的访问与操作范围。
- 服务器中名为**mysql**的数据库，用于维护数据库的用户以及权限的控制和管理。
- **MySQL**中的所有用户信息都保存在**mysql.user**数据表中。

mysql.user表中常见的字段

字段名	数据类型	默认值
Host	char(60)	
User	char(32)	
Select_priv	enum('N', 'Y')	N
Insert_priv	enum('N', 'Y')	N
Update_priv	enum('N', 'Y')	N
Delete_priv	enum('N', 'Y')	N
Create_priv	enum('N', 'Y')	N
Drop_priv	enum('N', 'Y')	N
Reload_priv	enum('N', 'Y')	N
Shutdown_priv	enum('N', 'Y')	N
Process_priv	enum('N', 'Y')	N
File_priv	enum('N', 'Y')	N
Grant_priv	enum('N', 'Y')	N
ssl_type	enum('','ANY','X509','SPECIFIED')	

mysql.user表中常见的字段

字段名	数据类型	默认值
ssl_cipher	blob	NULL
x509_issuer	blob	NULL
x509_subject	blob	NULL
max_questions	int(11) unsigned	0
max_updates	int(11) unsigned	0
max_connections	int(11) unsigned	0
max_user_connections	int(11) unsigned	0
plugin	char(64)	mysql_native_password
authentication_string	text	NULL
password_expired	enum('N','Y')	N
password_last_changed	timestamp	NULL
password_lifetime	smallint(5) unsigned	NULL
account_locked	enum('N','Y')	N

根据mysql.user表字段的功能可将其分为6类

- 客户端访问服务器的账号字段
- 验证用户身份的字段
- 安全连接的字段
- 资源限制的字段
- 权限字段
- 账户是否锁定的字段

用户分类

- 超级管理员用户 **root**
  - 默认建立的，密码为空，拥有数据库中所有的权限。
  - 管理数据库用户，也就是增加用户，修改用户信息，和删除用户，并对用户授权等。
- 普通用户
  - 只拥有创建时赋予它的权限

user表

- 用户信息存放在数据库mysql 的user表中
- user 表中字段：
  - Host：可以登陆数据库的主机地址，“*” 表示所有客户端用户都可访问
  - User：用户登陆名
  - authentication_string ： 加过密的登陆密码
  - _priv 结尾的字段：表明用户的权限
  - 注：Host和User字段共同组成的复合主键用于区分MySQL中的账户
- 操作user表
  - 标准的SQL语句
  - MySQL特定的语句

查询user表中默认用户的Host和User值

- SELECT host, user FROM mysql.user;

host	user
localhost	mysql.infoschema
localhost	mysql.session
localhost	mysql.sys
localhost	root

- root：默认的超级用户。
- session：用于用户身份验证。
- sys：用于系统模式对象的定义，防止DBA（数据库管理员）重命名或删除root用户时发生错误。
- 默认情况，用户mysql.session和mysql.sys已被锁定

身份验证字段

- mysql.user表中已不再包含Password字段
- 使用plugin和 authentication_string字段保存用户身份验证的信息
- plugin
  - 用于指定用户的验证插件名称。
- authentication_string
  - 根据plugin指定的插件算法对账户明文密码（如123456）加密后的字符串。
- 示例
  - SELECT plugin, authentication_string FROM mysql.user
  - WHERE user='root';

创建用户

- 语法
  - **create user** 用户名@主机名 **[identified by [password]** ‘密码’**]**
  - **[,** 用户名@主机名**[identified by [password][**‘密码’**]] [,...]**
- 注意
  - 用户名区分大小写,主机名连接来自的主机；
  - 密码区分大小写；
  - 可以同时创建多个数据库用户，中间用逗号分隔。
  - 不指定主机地址、密码以及相关的用户选项，即对用户不限定客户端、不需要密码并且没有任何限制。
  - **host**值为“%”表示任何主机，**localhost**为本地主机，空字符串（’’）表示所有客户端。

示例

- 创建用户
  - create user STUD001@localhost identified by 'STUD001';
- 查看用户密码
  - SELECT plugin, authentication_string FROM mysql.user
  - WHERE user='STUD001';

修改密码

- #第1种语法（推荐）
  - ALTER USER 账户名 IDENTIFIED BY '明文密码';
- #第2种语法
  - SET PASSWORD [FOR账户名] = '明文密码'
- #第3种语法
  - SET PASSWORD [FOR账户名] = PASSWORD('明文密码)

mysqladmin修改用户密码

- 语法
  - **mysqladmin –u 用户名 [–h 主机地址] –p password 新密码**
- 参数
  - **-u：用于指定待要修改的用户名，如：root。**
  - **-h：用于指定对应的主机地址，默认为localhost。**
  - **-p：后面password为关键字，而不是待修改用户的原密码**
- 示例：
  - **mysqladmin -u STUD001 -h 10.100.3.17 -p password STUD000**
  - **alter user STUD001 identified with mysql_native_password by 'STUD000';**

root密码丢失找回

- 在MySQL的配置文件my.ini中添加skip-grant-tables选项。
- 重启MySQL服务
- 利用root用户登录，跳过密码的输入直接登录MySQL服务。
- 为root用户设置密码。

权限管理

- **MySQL权限信息根据其作用范围，分别存储在mysql数据库中的不同数据表中。**
- **启动MySQL，自动加载权限信息，并读取到内存中。**

数据表	描述
user	保存用户被授予的全局权限
db	保存用户被授予的数据库权限
tables_priv	保存用户被授予的表权限
columns_priv	保存用户被授予的列权限
procs_priv	保存用户被授予的存储过程权限
proxies_priv	保存用户被授予的代理权限

字段名	含义
Host	主机名
Db	数据库名
User	用户
Table_name	表名
Table_priv	表示对表进行操作的权限。包括 Select Insert Update Delete Create Drop Grant References Index 和Alter
Columns_priv	表示对列的操作的权限
Timestamp	表示修改权限的时间
Grantor	表示是谁设置的权限

权限分类

- **MySQL权限分为数据权限、结构权限以及管理权限。**

数据权限

分类	权限	权限级别	描述
数据权限	SELECT	全局、数据库、表、列	SELECT
	UPDATE	全局、数据库、表、列	UPDATE
	DELETE	全局、数据库、表	DELETE
	INSERT	全局、数据库、表、列	INSERT
	SHOW DATABASES	全局	SHOW DATABASES
	SHOW VIEW	全局、数据库、表	SHOW CREATE VIEW
	PROCESS	全局	SHOW PROCESSLIST

结构权限

分类	权限	权限级别	描述
结构权限	DROP	全局、数据库、表	允许删除数据库、表和视图
	CREATE	全局、数据库、表	创建数据库、表
	CREATE ROUTINE	全局、数据库	创建存储过程
	CREATE TABLESPACE	全局	允许创建、修改或删除表空间和日志文件组
	CREATE TEMPORARY TABLES	全局、数据库	CREATE TEMPORARY TABLE
	CREATE VIEW	全局、数据库、表	允许创建或修改视图
	ALTER	全局、数据库、表	ALTER TABLE
	ALTER ROUTINE	全局、数据库、存储过程	允许删除或修改存储过程
	INDEX	全局、数据库、表	允许创建或删除索引
	TRIGGER	全局、数据库、表	允许触发器的所有操作
	REFERENCES	全局、数据库、表、列	允许创建外键

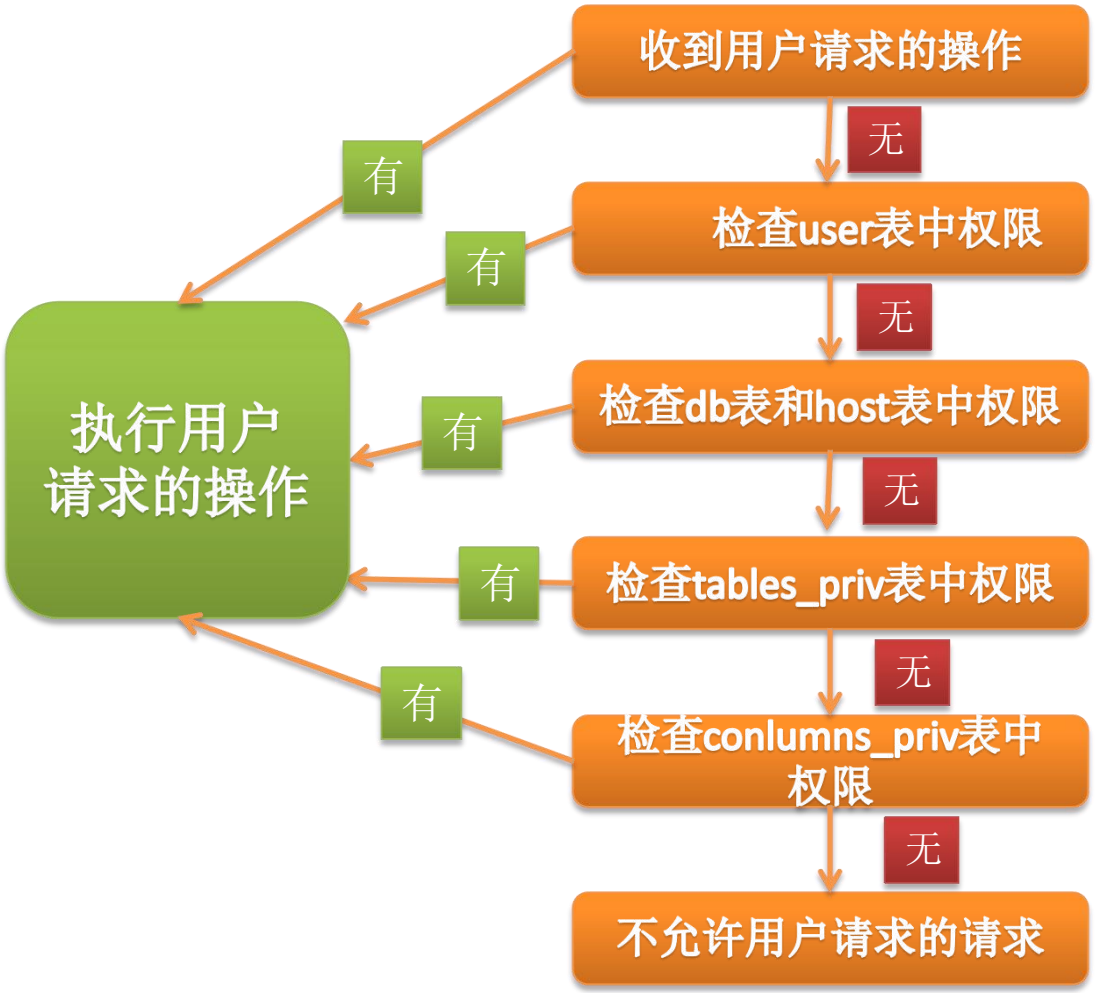
管理权限

分类	权限	权限级别	描述
管理权限	SUPER	全局	允许使用其他管理操作，如CHANGE MASTER TO等
	CREATE USER	全局	CREATE USER、DROP USER、RENAME USER 和REVOKEALL PRIVILEGES
	GRANT OPTION	全局、数据库、表、存储过程、代理	允许授予或删除用户权限
	RELOAD	全局	FLUSH操作
	PROXY		与代理的用户权限相同
	REPLICATION CLIENT	全局	允许用户访问主服务器或从服务器
	REPLICATION SLAVE	全局	允许复制从服务器读取的主服务器二进制日志事件
	SHUTDOWN	全局	允许使用mysqladmin shutdown
	LOCK TABLES	全局、数据库	允许在有SELECT表权限上使用LOCK TABLES

权限等级

- 全局层级
  - 适用于一个给定服务器中所有的数据库。
- 数据库层级
  - 适用于一个给定数据库中的所有目标。
- 表层级
  - 适用于一个给定表中的所有列。
- 列层级
  - 适用于一个给定表中的单一列。
- 子程序层级
  - 适用于存储的子程序 ， 可以被授权为全局层级和数据库层级。

权限执行过程



授予权限Grant

- 语法
  - GRANT 权限类型 [字段列表]... ON [目标类型] 权限级别
  - TO 账户名 [用户身份验证选项] [, 账户名 [用户身份验证选项]] ...
  - [REQUIRE 连接方式]
  - [WITH {GRANT OPTION | 资源控制选项}]
- 事项
  - 权限类型：SELECT、DROP、CREATE等权限
  - 字段列表：列权限
  - 目标类型：TABLE、FUNCTION、PROCEDURE
  - 权限级别：全局权限、数据库权限和表权限
  - WITH GRANT OPTION：可以为其他账户进行授权

授权

- **GRANT语句须拥有GRANT OPTION权限；**
- **启用read_only系统变量时，必须要拥有SUPER权限。**

权限级别	实现语法
全局权限	GRANT权限列表 ON *.* TO 账户名[WITH GRANT OPTION];
数据库级权限	GRANT权限列表 ON 数据库名.* TO 账户名[WITH GRANT OPTION];
表级权限	GRANT权限列表 ON 数据库名.表名 TO 账户名[WITH GRANT OPTION];
列级权限	GRANT 权限类型 (字段列表) [,…]ON 数据库名.表名 TO 账户名[WITH GRANT OPTION];
存储过程权限	GRANT EXECUTE|ALTER ROUTINE|CREATE ROUTINE ON {[*.*|数据库名.*]|PROCEDURE 数据库名.存储过程} TO 账户名 [WITH GRANT OPTION];
代理权限	GRANT PROXY ON 账户名 TO 账户名1 [, 账户名2] ...[WITH GRANT OPTION]

用户授权

- **grant all privileges on *.* to 'STUD001'@'%' identified by 'STUD001' with grant option;**
- **all privileges**
  - 所有权限，也可具体权限，如：**SELECT、CREATE、DROP**等。
- **On**
  - 对哪些数据库和表生效，格式：数据库名.表名， “*” 所有数据库和表
- **To**
  - 授予哪个用户。格式：”用户名”@”登录IP或域名”。%表示没有限制，在任何主机都可以登录。比如：”STUD001”@”192.168.0.%”
- **identified by**
  - 登录密码
- **with grant option**
  - 允许用户将自己的权限授权给其它用户

授权及查看

- 授权
  - `update mysql.user set host='%' where user='STUD001';`
  - `GRANT SELECT(StudName, StudGender),update(StudName, StudGender) on studscore_db.studinfo To 'STUD001'@'%'`
  - `grant select,insert,update on studscore_db.* to 'STUD001'@'%' ;`
- 查看授权
  - `SHOW GRANTS FOR 'root'@'localhost';`
  - `SHOW GRANTS FOR 'STUD001'@'%';`

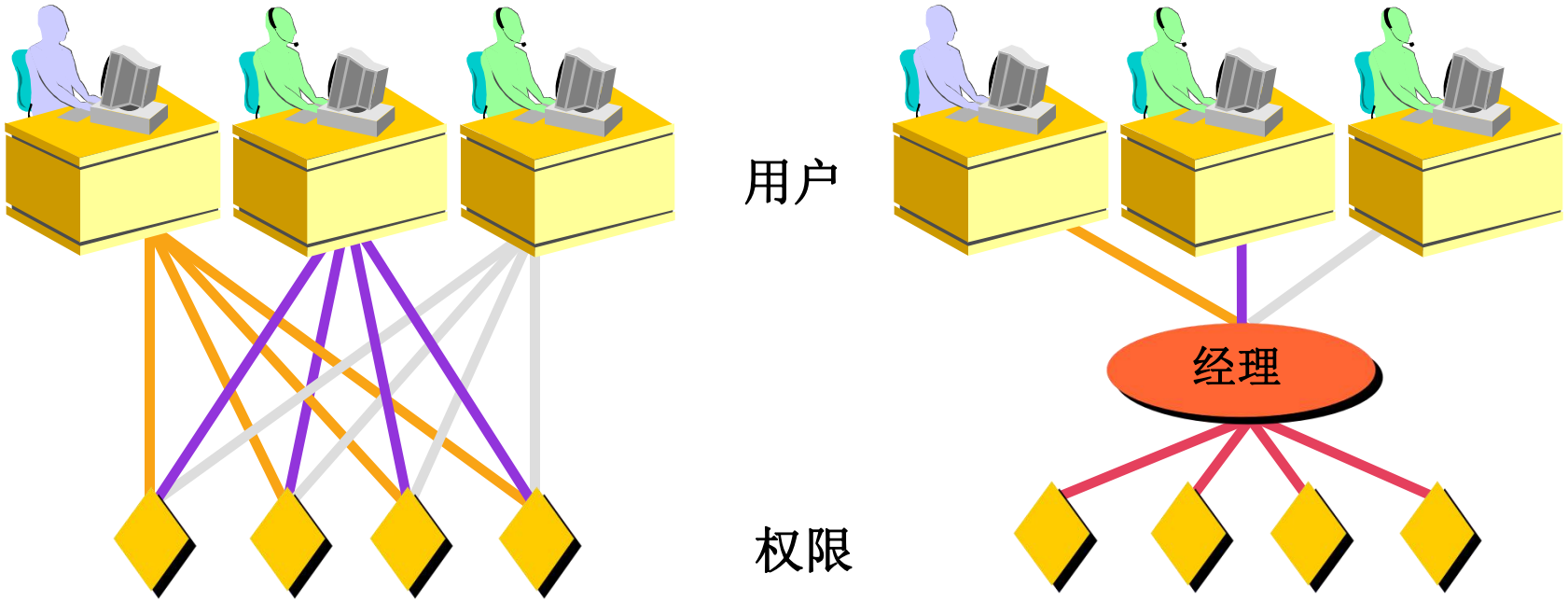
回收权限

- # ① 回收指定用户的指定权限
  - REVOKE 权限类型 [(字段列表)] [, 权限类型[(字段列表)]] …
- ON [目标类型] 权限级别 FROM 账户名 [, 账户名] …
- # ② 回收所有权限以及可为其他用户授权的权限
  - REVOKE ALL [PRIVILEGES], GRANT OPTION FROM 账户名 [, 账户名] …
- # ③ 回收用户的代理权限
  - REVOKE PROXY ON 账户名 FROM 账户名1 [, 账户名2] …
- 示例：
  - revoke all PRIVILEGES on studscore_db.* from 'STUD001'@'%' ;

修改用户和刷新权限

- 修改用户名：**rename**
  - **rename user 老用户 to 新用户 [, 老用户 to 新用户] ....**
  - **rename user cat@localhost to tom@localhost;**
- 刷新权限
  - **# 方式1**
  - **FLUSH PRIVILEGES;**
  - **# 方式2**
  - **mysqladmin -uroot -p reload**
  - **# 方式3**
  - **mysqladmin -uroot -p flush-privileges**

什么是角色?



不用角色分配权限　　使用角色分配权限

什么是角色？

- 角色是命名的可以授予用户的相关权限的组
- 角色使得授予、撤回和维护权限容易的多
- 一个用户可以使用几个角色
- 几个用户也可以被指定相同的角色
- 角色典型地为数据库应用程序创建

角色示例

- create role Teacher;
- grant select (classid,classname) on studscore_db.classinfo to Teacher;
- grant create, drop,alter ,create routine on *.* to Teacher;
- grant select on studscore_db.studinfo to Teacher;
- grant Teacher to Teacher001

- SET global activate_all_roles_on_login=ON;

批量创建用户

```
delimiter $$
drop procedure if exists Proc_Batch_CreateUser;
create procedure Proc_Batch_CreateUser()
  begin
    declare i int default 1001;
    declare StrUser varchar(20);
    declare StrPWD varchar(20) default 'STUD2021';
    while i<=1010 do
            SET StrUser=concat('STUD2019115',i);
             SET @CreationOfUser = CONCAT("CREATE USER '",StrUser,"'@'%' IDENTIFIED BY
'",StrPWD,"';");
      set @GrantUser=CONCAT("grant SELECT, INSERT, UPDATE, DELETE, CREATE,
DROP,ALTER,references,create routine on *.* to '", StrUser, "'@'%';");
             select @CreationOfUser,@GrantUser;
             PREPARE CreateUser FROM @CreationOfUser;
             EXECUTE CreateUser;
             PREPARE GrantUser FROM @GrantUser;
             EXECUTE GrantUser;
       set i=i+1;
    end while;
  end $$
delimiter;
call Proc_Batch_CreateUser;
```

批量删除用户

```
delimiter $$
drop procedure if exists Proc_Batch_DropUser;
create procedure Proc_Batch_DropUser()
  begin
    declare i int default 1001;
    declare StrUser varchar(20);
    while i<=1010 do
       SET StrUser=concat('STUD2019115',i);
            Set @DropUser = concat("drop user '" ,struser, "'@'%';");
            PREPARE DropUser FROM @DropUser;
            EXECUTE DropUser;
       set i=i+1;
    end while;
  end $$
delimiter;
call Proc_Batch_DropUser;
```

游标批量创建用户

```
create procedure Proc_cur_CreateUser()
begin
  declare SID varchar(30) ;
  declare have int default 1;
  declare cur_STUDinfo cursor for
      select STUDno from STUDinfo;
  declare exit handler for NOT FOUND set have:= 0;
    open cur_STUDinfo;
    repeat
      fetch cur_STUDinfo into SID;
            SET @CreationOfUser = CONCAT("CREATE USER 'STUD",SID,"'@'%' IDENTIFIED BY 'STUD",SID,"';");
                set @GrantUser=CONCAT("grant SELECT, INSERT, UPDATE, DELETE, CREATE, DROP,ALTER,references,create
routine on *.* to 'STUD", SID, "'@'%';");
                  PREPARE CreateUser FROM @CreationOfUser;
                  EXECUTE CreateUser;
                  PREPARE GrantUser FROM @GrantUser;
                  EXECUTE GrantUser;
    until have = 0 end repeat;
  close cur_STUDinfo;
end;
call Proc_cur_CreateUser;
```

游标批量删除用户

```
create procedure Proc_cur_DropUser()
begin
  declare SID varchar(30) ;
  declare SName varchar(30);
   declare have int default 1;
  declare cur_STUDinfo cursor for
     select STUDno,STUDname from STUDinfo;
  declare exit handler for NOT FOUND set have:= 0;
   open cur_STUDinfo;
   repeat
    fetch cur_STUDinfo into SID,SName;
            Set @DropUser = concat("drop user 'STUD" ,SID, "'@'%';");
             PREPARE DropUser FROM @DropUser;
             EXECUTE DropUser;
   until have = 0 end repeat;
  close cur_STUDinfo;
end;
call Proc_cur_DropUser;
```

批量创建用户

- select Concat("Create User 'STUD",STUDNo,"'@'%' Identified by ","'STUD2021';") struser from STUDinfo
- into outfile 'd:/createuser.sql' LINES TERMINATED BY '\r\n';
- select CONCAT("grant SELECT, INSERT, UPDATE, DELETE, CREATE, DROP,ALTER,references,create routine on *.* to 'STUD", STUDno, "'@'%';")
- from STUDinfo
- into outfile 'd:/grantuser.sql' LINES TERMINATED BY '\r\n';

下次课内容

- 事务
  - 事务ACID属性
  - 事务类型
  - 隔离等级
- 锁
  - 锁概念
  - 并发性
- 备份与恢复
  - 备份
  - 恢复