**[MySQL用户管理](http://www.cnblogs.com/kissdodog/p/4173337.html)**

　　MySQL分为普通用户与root用户。这两种用户的权限不一样。

**一、权限表**

　　安装MySQL时会自动安装一个名为mysql的数据库。mysql数据库里面存储的都是权限表。

**1、User表**

　　user表有39个字段。这些字段可以分为4类：

* 用户列；
* 权限列；
* 安全列；
* 资源控制列；

**1、用户列**

　　用户列包括Host、User、Password，分别表示主机名、用户名和密码。登录用的就是这个用户名与密码；

**2、权限列**

　　user表的权限列包括Select\_priv、Insert\_priv等以priv结尾的字段。

　　这些字段的值只有Y和N。Y表示该权限可以用到所有数据库上；N表示该权限不能用到所有数据库上；

　　通常，可以使用GRANT语句Wie用户赋予一些权限，也可以通过Update语句更新user表的方式来设置权限；

　　不过，修改user表之后，一定要执行一下FLUSH PRIVILEGES，否则可能会出现如下错误：

*Error (1133): Can’t find any matching row in the user table*

|  |  |
| --- | --- |
| 列 | 说明 |
| Select\_priv | 确定用户是否可以通过SELECT命令选择数据 |
| Insert\_priv | 确定用户是否可以通过INSERT命令插入数据 |
| Update\_priv | 确定用户是否可以通过UPDATE命令修改现有数据 |
| Delete\_priv | 确定用户是否可以通过DELETE命令删除现有数据 |
| Create\_priv | 确定用户是否可以创建新的数据库和表 |
| Drop\_priv | 确定用户是否可以删除现有数据库和表 |
| Reload\_priv | 确定用户是否可以执行刷新和重新加载MySQL所用各种内部缓存的特定命令，包括日志、权限、主机、查询和表 |
| Shutdown\_priv | 确定用户是否可以关闭MySQL服务器。在将此权限提供给root账户之外的任何用户时，都应当非常谨慎 |
| Process\_priv | 确定用户是否可以通过SHOW PROCESSLIST命令查看其他用户的进程 |
| File\_priv | 确定用户是否可以执行SELECT INTO OUTFILE和LOAD DATA INFILE命令 |
| Grant\_priv | 确定用户是否可以将已经授予给该用户自己的权限再授予其他用户 |
| References\_priv | 目前只是某些未来功能的占位符；现在没有作用 |
| Index\_priv | 确定用户是否可以创建和删除表索引 |
| Alter\_priv | 确定用户是否可以重命名和修改表结构 |
| Show\_db\_priv | 确定用户是否可以查看服务器上所有数据库的名字，包括用户拥有足够访问权限的数据库 |
| Super\_priv | 确定用户是否可以执行某些强大的管理功能，例如通过KILL命令删除用户进程，使用SET GLOBAL修改全局MySQL变量，执行关于复制和日志的各种命令 |
| Create\_tmp\_table\_priv | 确定用户是否可以创建临时表 |
| Lock\_tables\_priv | 确定用户是否可以使用LOCK TABLES命令阻止对表的访问/修改 |
| Execute\_priv | 确定用户是否可以执行存储过程 |
| Repl\_slave\_priv | 确定用户是否可以读取用于维护复制数据库环境的二进制日志文件。此用户位于主系统中，有利于主机和客户机之间的通信 |
| Repl\_client\_priv | 确定用户是否可以确定复制从服务器和主服务器的位置 |
| Create\_view\_priv | 确定用户是否可以创建视图 |
| Show\_view\_priv | 确定用户是否可以查看视图或了解视图如何执行 |
| Create\_routine\_priv | 确定用户是否可以更改或放弃存储过程和函数 |
| Alter\_routine\_priv | 确定用户是否可以修改或删除存储函数及函数 |
| Create\_user\_priv | 确定用户是否可以执行CREATE USER命令，这个命令用于创建新的MySQL账户 |
| Event\_priv | 确定用户能否创建、修改和删除事件 |
| Trigger\_priv | 确定用户能否创建和删除触发器 |

　　Update修改权限：

UPDATE user SET Select\_priv = 'N'

WHERE Host = '%'

　　GRANT赋予权限

**3、安全列**

　　user表的安全列有4个字段：

* ssl\_type；
* ssl\_cipher；
* x509\_issuer；
* x509\_subject；

　　ssl用于加密；x509标准可以用来标识用户。普通的发行版都没有加密功能。可以使用SHOW VARIABLES LIKE 'have\_openssl'语句来查看是否具有ssl功能。如果取值为DISABLED，那么则没有ssl加密功能。

**4、资源控制列**

　　user表的4个资源控制列是：

* max\_questions：每小时可以允许执行多少次查询；
* max\_updates：每小时可以允许执行多少次更新；
* max\_connections：每小时可以建立多少连接；
* max\_user\_connections：单个用户可以同时具有的连接数。

　　默认值为0，表示无限制。

**2、db表和host表**

　　db表存储了某个用户对一个数据库的权限。

　　host表中存储了某个主机对数据库的操作权限，配合db表对给定主机上数据库级操作权限做更细致的控制；但是很少用，新版本已经取消了host表；

**1、用户列**

　　db表的用户列有3个字段：

* Host：主机名；
* Db：数据库名；
* User：用户名；

**2、权限列**

　　db表：

* Create\_routine\_priv：是否具有创建存储过程权限；
* Alter\_routine\_priv：是否具有修改存储过程权限；

**user表中的权限是针对所有数据库的**，如果user表中的Select\_priv字段取值为Y，那么该用户可以查询所有数据库中的表；

　　如果为某个用户值设置了查询test表的权限，那么user表的Select\_priv字段的取值为N。而这个SELECT权限则记录在db表中。db表中的Select\_priv字段的取值将会是Y。

**用户先根据user表的内容获取权限，然后再根据db表的内容获取权限。**

**3、tables\_priv表和columns\_priv表**

tables\_priv：可以对单个表进行权限设置：

* tables\_priv表包含8个字段：
* Host：主机名；
* DB：数据库名；
* User：用户名；
* Table\_name：表名
* Table\_priv：对表进行操作的权限(Select,Insert,Update,Delete,Create,Drop,Grant,References,Index,Alter)
* Column\_priv：对表中的数据列进行操作的权限(Select,Insert,Update,Rederences)；
* Timestamp：修改权限的事件
* Grantor：权限的设置者

　　columns\_priv：可以对单个数据列进行权限设置，有7个列，作用同上：

 　　Host、Db、User、Table\_name、Column\_name、Column\_priv、Timestamp。

**MySQL权限分配是按照user表-> db表 -> table\_priv表 -> columns\_priv表的顺序进行分配的。**

**在数据库系统中，先判断user表中的值是否为'Y'，如果user表中的值是'Y'，就不需要检查后面的表。如果user表为N，则一次检查后面的表。**

**4、procs\_priv表**

* procs\_priv表可以对存储过程和存储函数进行权限设置。
* procs\_priv表包含8个字段，分别是：
* Host：主机名；
* Db：数据库名；
* User：用户名；
* Routine\_name：存储过程或函数名称；
* Routine\_type：类型(取值有：FUNCTION或PROCEDURE)；
* Proc\_priv：拥有的权限(Execute：执行；Alter Routine：修改；Grant：权限赋予)；
* Timestamp：字段存储更新的时间；
* Grantor：字段设置者；

**二、账户管理**

　　账户管理是MySQL用户管理最基本的内容。包括登录，退出MySQL服务器、创建用户、删除用户、密码管理和权限管理等内容。

　　其中登录方式非常简单，在这个地址有：http://www.cnblogs.com/kissdodog/p/4154068.html

**1、新建普通用户**

　　在MySQL数据库中，建立用户有3种方式：

* 使用CREATE USER语句来创建新的用户；
* 直接在mysql.user表中INSERT用户；
* 使用GRANT语句来新建用户；

**1、CREATE USER**

　　使用CREATE USER语句创建用户，必须要拥有CREATE USER权限。其格式如下：

CREATE USER user[IDENTIFIED BY [PASSWORD] 'password'],  
[user[IDENTIFIED BY [PASSWORD] 'password']]...

　　其中，user参数表示新建用户的账户，user由用户名(User)和主机名(Host)构成；IDENTIFIED BY关键字用来设置用户的密码；password参数表示用户的密码；如果密码是一个普通的字符串，就不需要使用PASSWORD关键字。可以没有初始密码。

　　示例：

CREATE USER 'admin'@'%' IDENTIFIED BY 'admin'

　　执行之后user表会增加一行记录，但权限暂时全部为‘N’。

**2、用INSERT语句新建普通用户**

　　可以使用INSERT语句直接将用户的信息添加到mysql.user表。但必须拥有mysql.user表的INSERT权限。

　　另外，ssl\_cipher、x509\_issuer、x509\_subject没有值，必须要设置值，否则INSERT语句无法执行。

　　示例：

INSERT INTO mysql.user(Host,User,Password,ssl\_cipher,x509\_issuer,x509\_subject) VALUES('%','newuser1',PASSWORD('123456'),'','','')

　　执行INSERT之后，要使用命令：

FLUSH PRIVILEGES

　　命令来使用户生效。

**3、用GRANT语句来新建普通用户**

　　用GRANT来创建新的用户时，能够在创建用户时为用户授权。但需要拥有GRANT权限。

　　语法如下：

GRANT priv\_type ON database.table

TO user[IDENTIFIED BY [PASSWORD] 'password']

[,user [IDENTIFIED BY [PASSWORD] 'password']...]

* priv\_type：参数表示新yoghurt的权限；
* databse.table：参数表示新用户的权限范围；
* user：参数新用户的账户，由用户名和主机构成；
* IDENTIFIED BY关键字用来设置密码；
* password：新用户密码；

　　GRANT语句可以同时创建多个用户。

GRANT SELECT ON \*.\* TO netUser@'%'

　　\*.\*与db.\*的区别在于。\*.\*对所有数据库生效，所以user表的SELECT会变为Y。而db.\*user表为'N'，更改的是Db表。

**2、删除普通用户**

**1、DROP USER语句删除普通用户**

　　需要拥有DROP USER权限。语法如下：

DROP USER user[,user]...

　　user是需要删除的用户，由用户名(User)和主机名(Host)构成。

DROP USER 'newuser1'@'%'

**2、DELETE语句删除普通用户**

　　可以使用DELETE语句直接将用户的信息从mysql.user表中删除。但必须拥有对mysql.user表的DELETE权限。

DELETE FROM mysql.user WHERE Host = '%' AND User = 'admin'

　　删除完成后，一样要FLUSH PRIVILEGES才生效。

**3、修改密码**

**1、使用mysqladmin命令来修改root用户的密码**

　　语法：

mysqladmin -u -username -p password "new\_password"

　　新密码(new\_password)必须用括号括起来，单引号会报错。

　　示例：

　　（注：这个待验证，我怎么执行都不成功！）

**2、修改user表**

　　UPDATE user表的passwor字段的值，也可以达到修改密码的目的；

UPDATE user SET Password = PASSWORD('123') WHERE USER = 'myuser'

　　FLUSH PRIVILEGES后生效。

**3、使用SET语句来修改密码**

　　使用root用户登录到MySQL服务器后，可以使用SET语句来修改密码：

　　修改自己的密码，不需要用户名

SET PASSWORD = PASSWORD("**123**");

　　修改其他用户密码：

SET PASSWORD FOR 'myuser'@'%'=PASSWORD("**123456**")

　　FOR 用户名@主机名

**4、GRANT语句来修改普通用户的密码**

　　使用GRANT语句修改普通用户的密码，必须拥有GRANT权限。

GRANT priv\_type ON database.table TO user [IDENTIFIED BY [PASSWORD] 'password']

　　示例：

GRANT SELECT ON \*.\* TO 'test3'@'%' IDENTIFIED BY '123'

**4、忘记密码的解决方案**

　　如果root用户密码丢失了，会给用户造成很大的麻烦。但是，可以通过某种特殊方法登录到root用户下，然后修改密码。

**1、使用--skip-grant-tables选项启动MySQL服务**

　　该选项将使MySQL服务器停止权限判断，任何用户都能够访问数据库。

　　在Windows操作系统下，使用mysqlid或mysqlid-nt来启动MySQL服务。

　　mysqld命令：

*mysqld --skip-grant-tables*

　　mysqld-nt命令：

*mysqld-nt --skip-grant-tables*

　　net start mysql命令：

*net start mysql --skip-grant-tables*

**2、登录root用户，并且设置新密码**

*mysql -u root*

　　必须使用UPDATE语句更新mysql数据库下的user表，而不能使用SET语句，因为--skip-grant-tables不能使用SET语句。

　　修改完密码以后，FLUSH PRIVILEGES语句加载权限表，新密码即生效。

**三、权限管理**

**1、MySQL的各种权限**

|  |  |  |
| --- | --- | --- |
| 权限名称 | 对应user表中的列 | 权限的范围 |
| CREATE | Create\_priv | 数据库、表或索引 |
| DROP | Drop\_priv | 数据库或表 |
| GRANT OPTION | Grant\_priv | 数据库、表、存储过程或函数 |
| REFERENCES | References\_priv | 数据库或表 |
| ALTER | Alter\_priv | 修改表 |
| DELETE | Delete\_priv | 删除表 |
| INDEX | Index\_priv | 用索引查询表 |
| INSERT | Insert\_priv | 插入表 |
| SELECT | Select\_priv | 查询表 |
| UPDATE | Update\_priv | 更新表 |
| CREATE VIEW | Create\_view\_priv | 创建视图 |
| SHOW VIEW | Show\_view\_priv | 查看视图 |
| ALTER ROUTINE | Alter\_routine | 修改存储过程或存储函数 |
| CREATE ROUTINE | Create\_routine\_priv | 创建存储过程或存储函数 |
| EXECUTE | Execute\_priv | 执行存储过程或存储函数 |
| FILE | File\_priv | 加载服务器主机上的文件 |
| CREATE TEMPORARY TABLES | Create\_temp\_table\_priv | 创建临时表 |
| LOCK TABLES | Lock\_tables\_priv | 锁定表 |
| CREATE USER | Create\_user\_priv | 创建用户 |
| PROCESS | Process\_priv | 服务器管理 |
| RELOAD | Reload\_priv | 重新加载权限表 |
| REPLICATION CLIENT | Repl\_client\_priv | 服务器管理 |
| REPLICATION SLAVE | Repl\_slave\_priv | 服务器管理 |
| SHOW DATABASES | Show\_db\_priv | 查看数据库 |
| SHUTDOWN | Shutdown\_priv | 关闭服务器 |
| SUPER | Super\_priv | 超级权限 |

**2、授权**

　　GRANT语法如下：

GRANT priv\_type [(column\_list)] ON database.table

TO user [IDENTIFIED BY [PASSWORD] 'password']

[,user [IDENTIFIED BY [PASSWORD] 'password']]...

WITH with\_option[with\_option]

* priv\_type参数表示权限类型；
* column\_list：参数表示权限作用于哪些列上，没设置则位于整个表上；
* user参数由用户名和主机名构成；形式是"'username'@'hostname'"；
* IDENTIFIED BY参数用于为用户设置密码；
* password：用户新密码；

　　WITH关键字后面带有一个或多个with\_option参数。有5个选项：

* GRANT OPTION：被授权的用户可以将这些权限赋予给别的用户；
* MAX\_QUERIES\_PER\_HOUR count：设置没消失可以允许执行count次查询；
* MAX\_UPDATES\_PER\_HOUR count：设置每个消失可以允许执行count次更新；
* MAX\_CONNECTIONS\_PER\_HOUR count：设置每小时可以建立count个连接；
* MAX\_USER\_CONNECTIONS count：设置单个用户可以同时具有的count个连接数；

　　示例：

GRANT SELECT,UPDATE ON \*.\*

TO 'myuser'@'%'

WITH GRANT OPTION;

**3、收回权限**

　　收回权限，就是取消某个用户的某些权限。MySQL中使用REVOKE关键字来为用户设置权限。

　　语法如下：

REVOKE priv\_type[(column\_list)]

ON database.table

FROM user[,user]

　　收回全部权限的REVOKE语句的基本语法如下：

REVOKE ALL PRIVILEGES,GRANT OPTION FROM user[,user]...

　　示例：回收用户myuser的SELECT权限

REVOKE SELECT ON \*.\*FROM 'myuser'@'%'

　　收回myuser的所有权限：

REVOKE ALL PRIVILEGES,GRANT OPTION FROM 'myuser'@'%'

**4、查看权限**

　　SHOW GRANTS语句用于查看权限。同时mysql数据库下的user表中存储着用户的基本权限。

SELECT \* FROM mysql.user

SHOW GRANTS

