本次实训的学习路径：

1.mysql系统库中的权限系统表(第12章)

user/db/table\_priv/columns\_priv表

2.访问权限的控制过程(第13章)

访问权限控制系统/权限类型/命名规则/账号和密码认证/权限检查

3.示例练习(17.1节，查看用户不同作用域的权限)

目录

[第12章　mysql系统库之权限系统表 1](#_Toc9624)

[12.1　user(包含用户账户、全局权限和其他非权限列表) 1](#_Toc18215)

[12.2　db(数据库级别的权限表) 2](#_Toc2826)

[12.3　tables\_priv(表级别的权限表) 3](#_Toc2786)

[12.4　columns\_priv(字段级别的权限表) 3](#_Toc6468)

[第13章　mysql系统库之访问权限控制系统 3](#_Toc13068)

[13.1　访问权限控制系统概述 3](#_Toc2783)

[13.2 MySQL提供了哪些权限 5](#_Toc30738)

[13.3 MySQL账号命名规则 6](#_Toc26079)

[13.4 MySQL账号访问控制两阶段 6](#_Toc503)

[13.4.1　第一阶段（账号和密码认证） 6](#_Toc11298)

[13.4.2　第二阶段（权限检查） 8](#_Toc25976)

[第17章　mysql系统库应用示例荟萃(部分) 11](#_Toc14434)

[17.1 查看用户不同作用域的权限 11](#_Toc3987)

[17.1.1　查看全局权限 11](#_Toc29733)

[17.1.2　查看库级别权限 12](#_Toc14893)

[17.1.3　查看表级别权限 13](#_Toc13905)

[17.1.4　查看字段级别权限 14](#_Toc6876)

**第12章　mysql系统库之权限系统表**

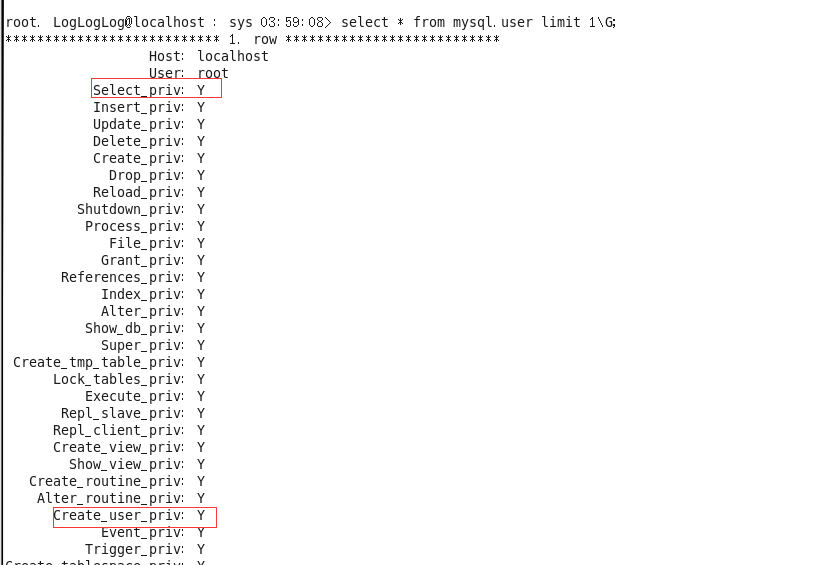
**mysql权限控制之于性能优化：就好比书中也安排了有关msql备份等内容，一个健壮的数据库系统是其性能调优的前提和基础。**

**注：对之前所学performance\_schema，information\_schema，sys以及这里的mysql系统库，可参考以下文章的简答梳理：**

<https://blog.csdn.net/cnds123/article/details/104562559>

**12.1　user(包含用户账户、全局权限和其他非权限列表)**

#user表提供查询全局权限信息。该表中的账号密码信息在认证1阶段决定着是否允许用户连接，对于通过账号密码认证阶段的连接，如果同时通过user表中的权限检查，那么就代表该用户拥有**全局权限**。

#从Select\_priv到Create\_tablespace\_priv之间的字段，官方称为权限字段。每一个字段对应一个具体的权限，值为Y代表有权限，值为N代表没有权限

mysql> select \* from mysql.user limit 1\G;

**注：有许多字段，共分为三类：允许访问的客户端范围、权限字段、资源控制字段**

**注：1.4节创建了一个管理员program，所以这里可以看到它的权限：**

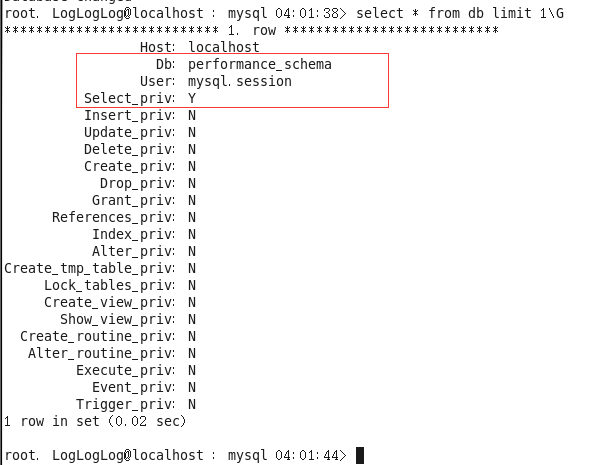
**表格

描述已自动生成**

**12.2　db(数据库级别的权限表)**

#db表中记录的权限信息代表用户是否可以使用这些权限来访问被授予访问的**数据库下的所有对象**

mysql> select \* from db limit 1\G

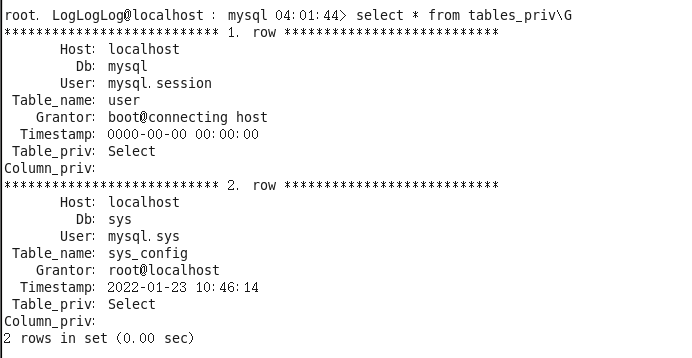


**注：以上结果显示，对于performance\_schema，mysql.session只有查询权限。**

**12.3　tables\_priv(表级别的权限表)**

# tables\_priv表中记录的权限信息代表用户是否可以使用这些权限访问**某个表中的所有字段**

mysql> select \* from tables\_priv\G



**注：留意第二行，root授予mysql.sys对sys库的查询权限。**

**12.4　columns\_priv(字段级别的权限表)**

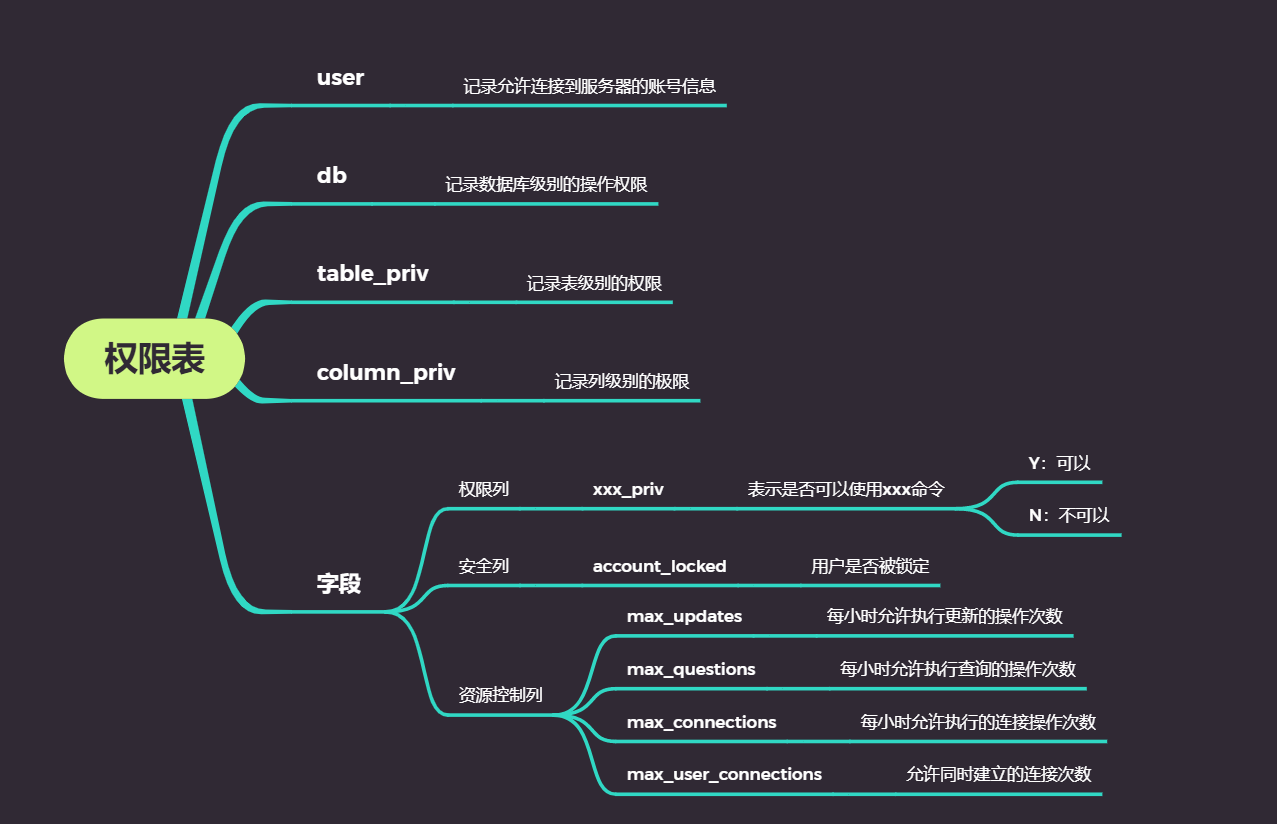
#columns\_priv表中记录的权限信息代表用户可以使用这些权限来访问某个表的指定字段

mysql> select \* from columns\_priv\G



**注：此处实验室环境大概会是空表。说明现在还没有针对字段级别的权限设置(一般来说也未必用得上如此细的权限设置)。17.1节有相关的示例讨论该表。**

**进行第13章练习前请补充：下表(思维导图)是对user、db和tables\_priv基本作用以及各字段含义的总结。**



**第13章　mysql系统库之访问权限控制系统**

**13.1　访问权限控制系统概述**

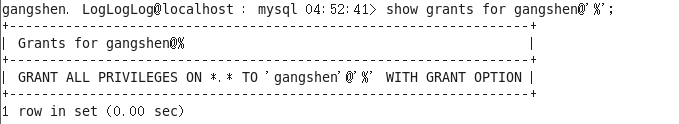
**重点：理解mysql如何控制用户访问权限的基本过程。**

#什么是**访问权限控制系统**？MySQL的mysql系统库提供了user、db、tables\_priv、columns\_priv、procs\_priv、proxies\_priv几个表，用于存放不同权限范围的用户账号相关数据，这些表共同组成了MySQL的访问权限控制系统。

#MySQL访问权限控制系统的主要功能是身份验证和校验权限。

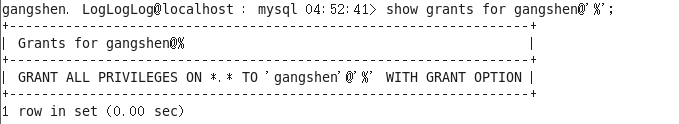
#当一个用户连接到MySQL服务器时，其认证身份由“请求连接的主机名和用户名”确定，MySQL使用主机名+用户名的方式来识别和区分“相同主机不同用户”和“不同主机相同用户”发出的请求：

mysql> show grants for gangshen@'localhost';



**注：以上是gangshen本地连接数据库时具有的权限。**

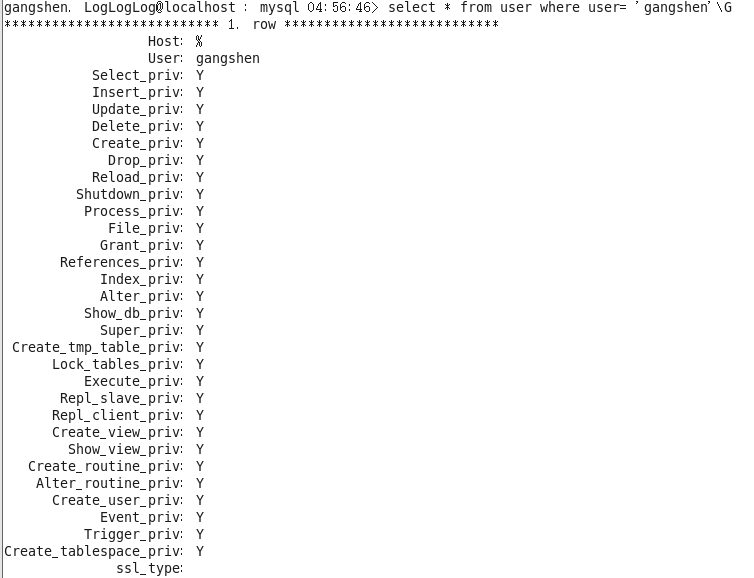
mysql> show grants for gangshen@'%';



**注：以上是gangshen从任意主机(显然也包含本地)连接数据库时具有的权限。**

**注：这里用之前在1.4节中的gangshen用户替代。这些信息在mysql系统库都已经记录了：**

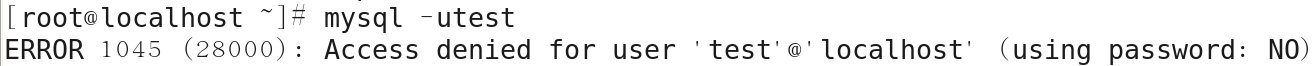
**select \* from user where user= 'gangshen'\G**



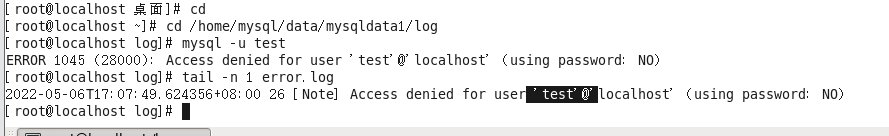
#当用户使用客户端程序连接到MySQL服务器时，**MySQL的访问控制分为如下两个阶段**。

阶段1. 服务器根据**身份标识**（主机名+用户名组成的账号名称）在MySQL的访问权限控制表中查询相关信息，以确定需要接受或拒绝该用户的连接（没有查询到就拒绝连接）。如果查询到了用户记录，则校验用户提供的账号密码是否正确，如果密码不正确则拒绝连接。

**注：比如我们所用系统中没有test用户：**

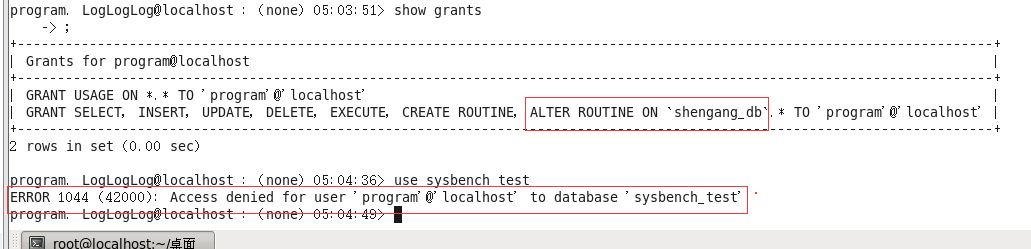


该错误被日志记录：

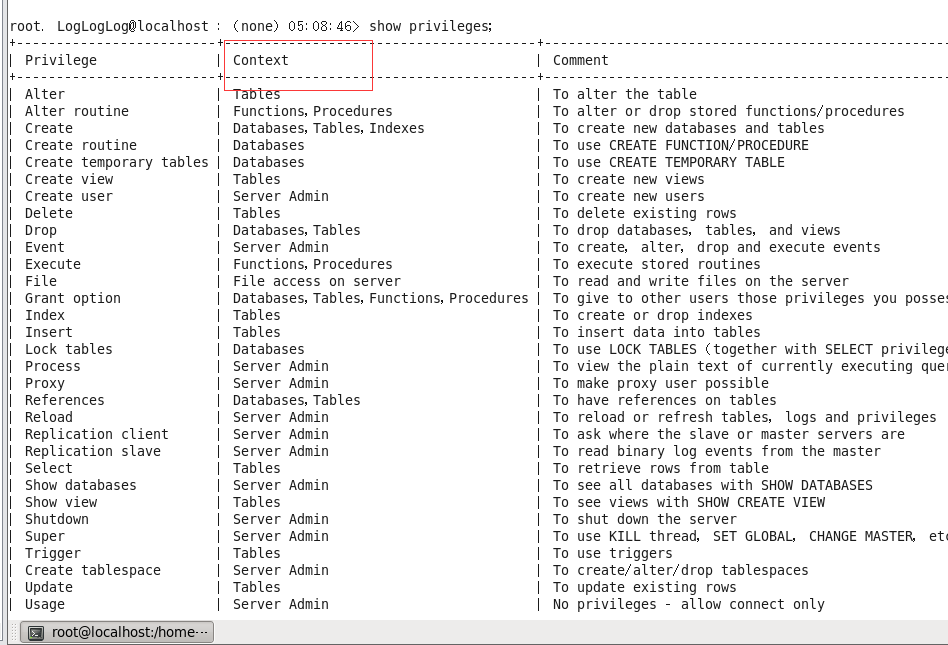


/home/mysql/data/mysqldata1/log

2. 阶段2：用户连接成功之后，服务器会检查用户访问请求中的每个声明，确定是否有足够的**权限**来执行。



**13.2 MySQL提供了哪些权限**



# Context字段显示了该权限的**使用环境**。根据Context不同，权限分为如下三类：

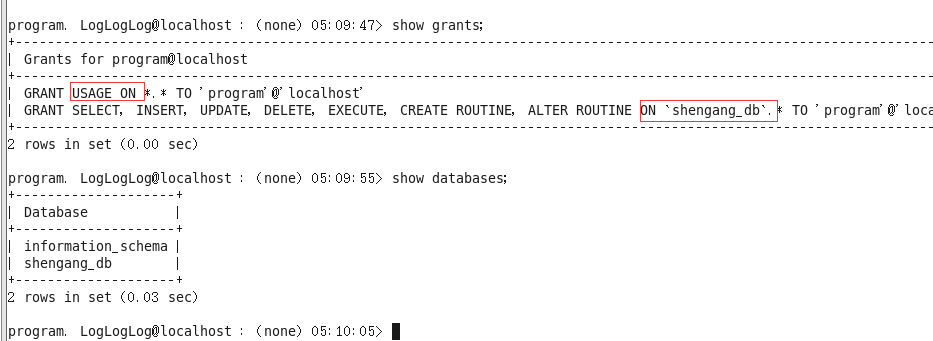
（1）管理权限(黄色)：用于管理MySQL服务器的操作。

（2）数据库级别权限(绿色)：授权范围可以是某数据库或某数据库中的所有对象，也可以是所有数据库（可以使用**\*.\*代表全局对象**；也可以使用db.\*代表某库下的所有对象）。

（3）数据库对象级别权限(蓝色)：授权范围可以为数据库中的特定对象、数据库内给定类型的对象，也可以是所有数据库（\*.\*代表全局对象，db.\*代表某库下的所有对象，db.tb代表某库下的某对象）。

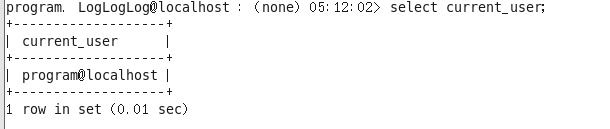
**注：参考教材了解表中的每个权限的解释。比如：**

# Usage：该权限代表用户“**无任何权限**”。全局级别权限，拥有该权限的用户可以登录到数据库服务器中，但在默认配置下除能够执行部分show命令之外，其他任何数据变更和数据库查询操作都无法执行。

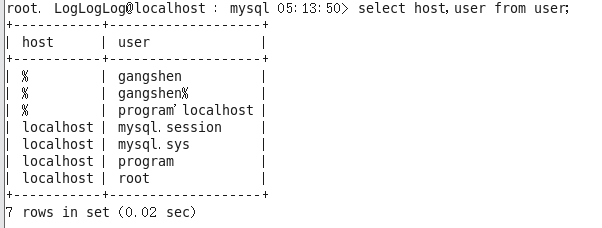


**13.3 MySQL账号命名规则**

#账号名称构成语法：**'user\_name'@'host\_name'(留意看很容易错)**，用户名严格区分大小写，但主机名不区分大小写。



# user\_name和host\_name**分开存储**在mysql库user表的User和Host两个字段中



**注：其余内容略。感兴趣者可自行阅读补充笔记。**

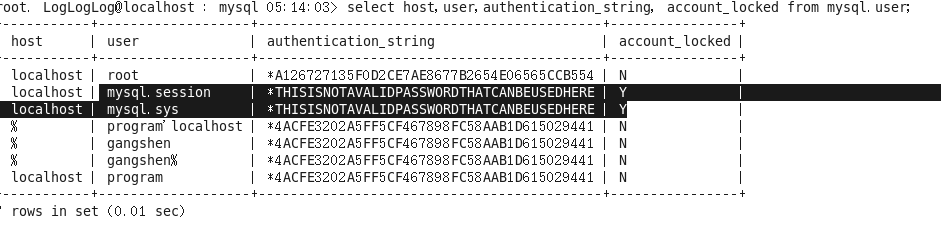
**13.4 MySQL账号访问控制两阶段**

**13.4.1　第一阶段（账号和密码认证）**

# MySQL服务器使用user表中的**Host、User、authentication\_string**三个字段存储的用户凭证信息来执行凭证检查。用户账号的锁定状态记录在user表的**account\_locked**字段中：

# user表中非空的authentication\_string字段值表示加密过的密码字符串（hash加密）

mysql> select host,user,authentication\_string, account\_locked from mysql.user;



**注：参考如下语句加锁/解锁。**

# 账号锁定状态可以通过ALTER USER语句进行更改

ALTER USER [IF EXISTS]

user [auth\_option] [, user [auth\_option]] ...

[REQUIRE {NONE | tls\_option [[AND] tls\_option] ...}]

[WITH resource\_option [resource\_option] ...]

[password\_option | lock\_option] ...

......

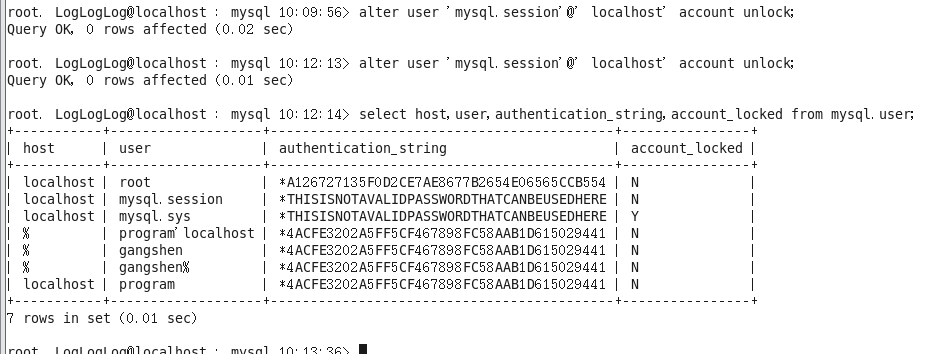
lock\_option: {

ACCOUNT LOCK

| ACCOUNT UNLOCK

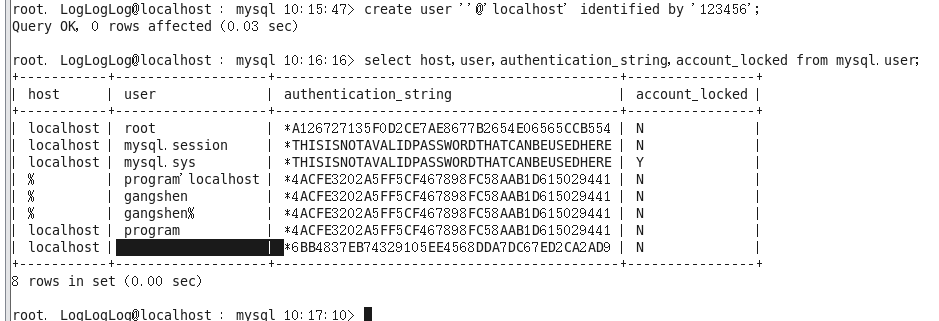
}

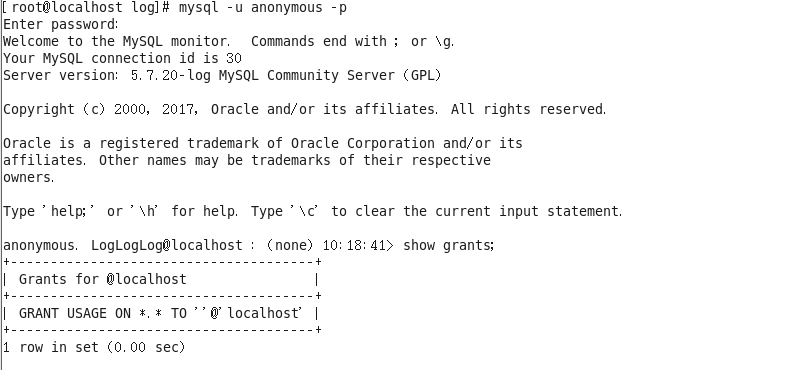
**注：例如用户mysql.session和mysql.sys被锁定了，以下操作解锁：**

**alter user 'mysql.session'@' localhost' account unlock;**

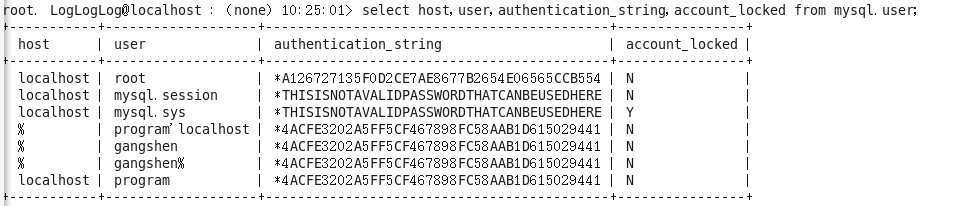
**注：下面的操作添加了匿名用户。**

**create user ''@'localhost' identified by '123456';**



**用另一个会话进行登录：**

**这种不安全的账号最后自然需要删掉：**



**留意原来匿名用户登录的会话中，用户不会立即下线，但因为内存中的记录已经删除，也就没有任何权限进行访问了：**

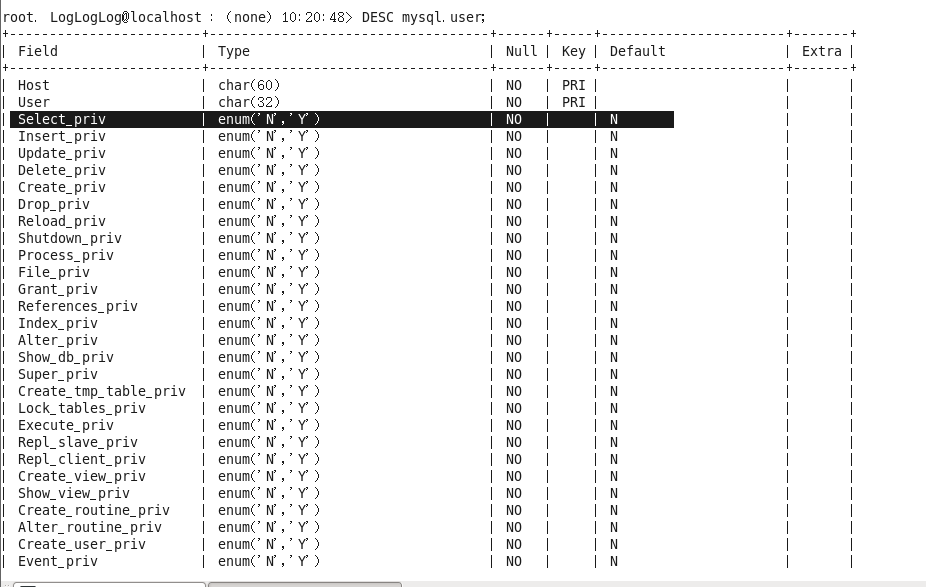


**注：书中有关于验证阶段的详细例子介绍，此处略去。**

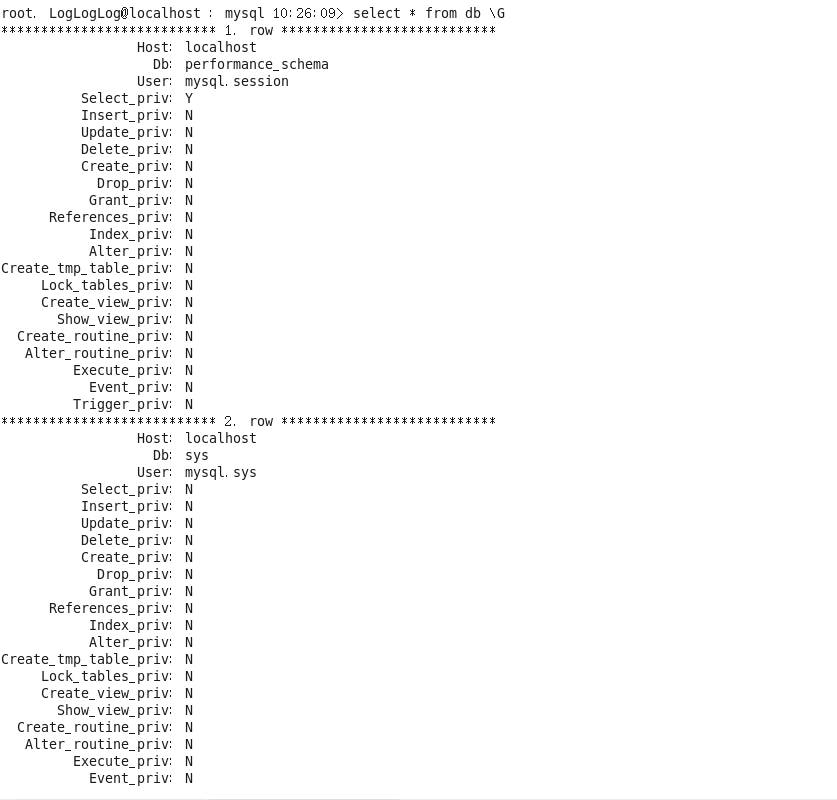
**13.4.2　第二阶段（权限检查）**

#在第二阶段中，客户端发送给服务器的每个请求，服务器都会检查请求操作的类型，然后检查是否有足够的访问权限来执行请求操作。检查工作依赖于mysql库下的user、db、**tables\_priv、columns\_priv、procs\_priv、proxies\_priv权限表**中存放的权限信息。

#user表中的权限**作用范围是全局性的**，当该表中的相应权限类型字段值为’Y’时，就表示对数据库实例中的**所有数据库表**都有该权限。

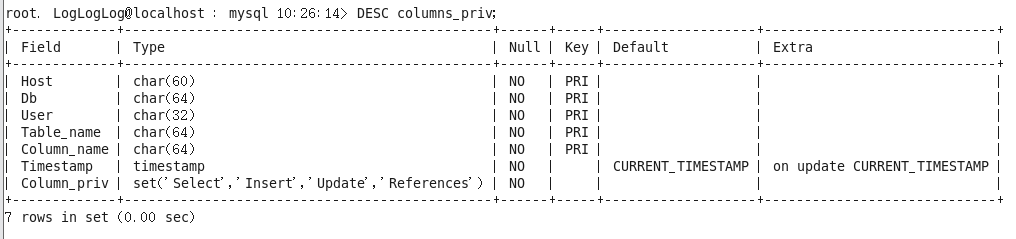


#db表中的权限作用范围是数据库级别的，对应数据库内的所有对象



**注：比如mysql.session对performance\_schema库中的表只有查询权限。**

#tables\_priv、columns\_priv和procs\_priv这三个表中记录着表级别权限、字段级别权限和存储程序权限。



**注：关于以上三种权限的关系以及如何判定用户是否有权限，书中通过如下布尔逻辑表达：**

global privileges

OR (database privileges AND host privileges)

OR table privileges

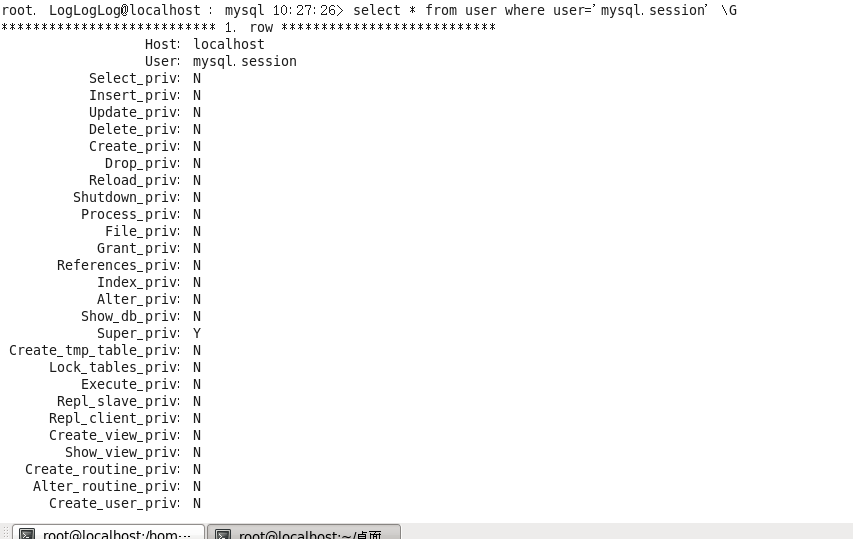
OR column privileges

OR routine privileges

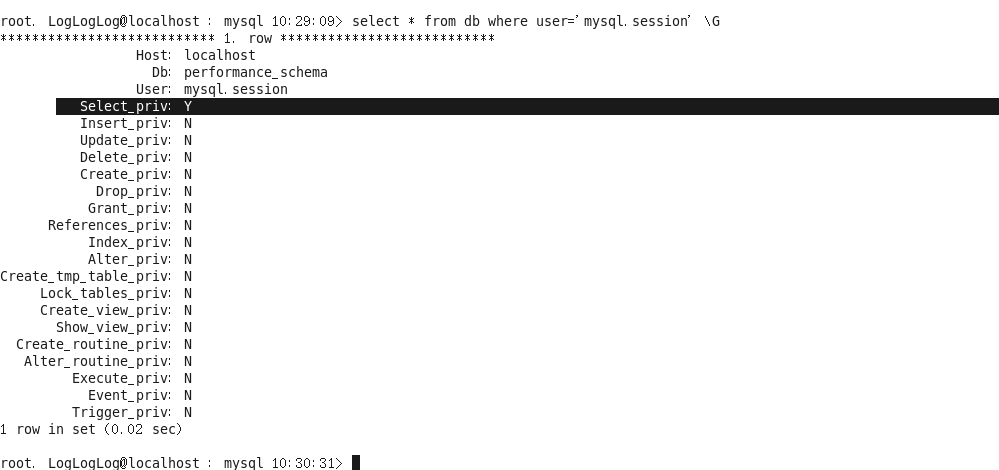
**就是说：**

#当一个客户端连接在进行第二阶段权限检查时，首先会检查user表，如果所检查权限是user表特有的（在其他权限表中没有该权限），且在user表中该权限记录为Yes，则服务器授予客户端访问权限，否则直接拒绝而不会继续检查其他权限表（因为在其他权限表中没有该权限，无须检查）；如果所检查权限除在user表中有之外，在其他权限表中也有（例如：DML操作权限），则即使在user表中不允许（毕竟user表中的权限是全局性的），也会继续往下检查db表，然后检查tables\_priv表，依此类推。

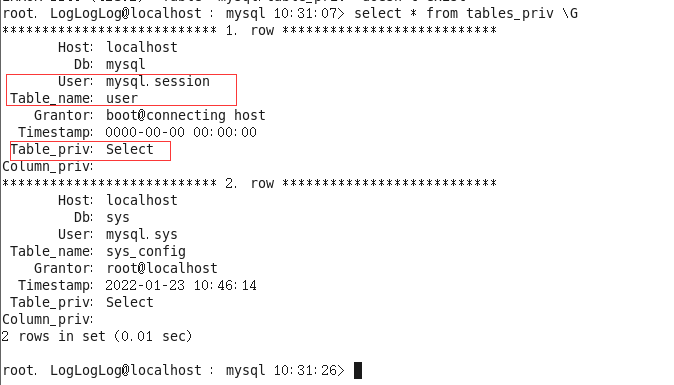
**比如mysql.session用户：“首先检查user表”**



**没有任何权限，但是Select\_priv权限字段在db也有，所以继续检查：**



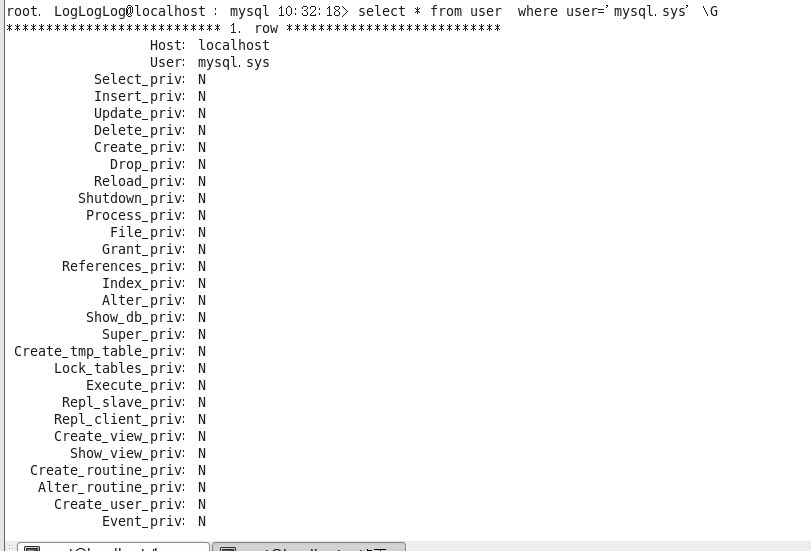
**发现原来可以查询performance\_schema（里面的所有表）。然后不要忘了Select\_priv权限字段在tables\_priv里面也有，所以继续查下去：**



**原来除了performance\_schema之外，mysql库里面的user表也是可以查询的。所以当此用户要查user表时，会经过重重检查并获得批准。**

**仿照以上例子，sys用户(可执行创建设置)的权限检查过程如下：**

**首先检查user表**



**没有任何权限，但是Select\_priv权限字段在db也有，所以继续检查**

## 

**Select\_priv权限字段在tables\_priv里面也有，最后在tables\_priv表中查找**



## 第17章　mysql系统库应用示例荟萃(部分)

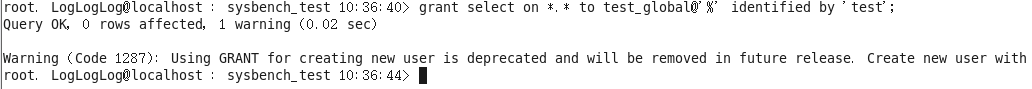
#介绍几个mysql系统库的日常应用示例，以便大家能够快速使用mysql系统库中的相关信息来处理数据库管理工作

**17.1 查看用户不同作用域的权限**

**17.1.1　查看全局权限**

#创建一个具有**全局权限**的用户test\_global，并授予Select权限。

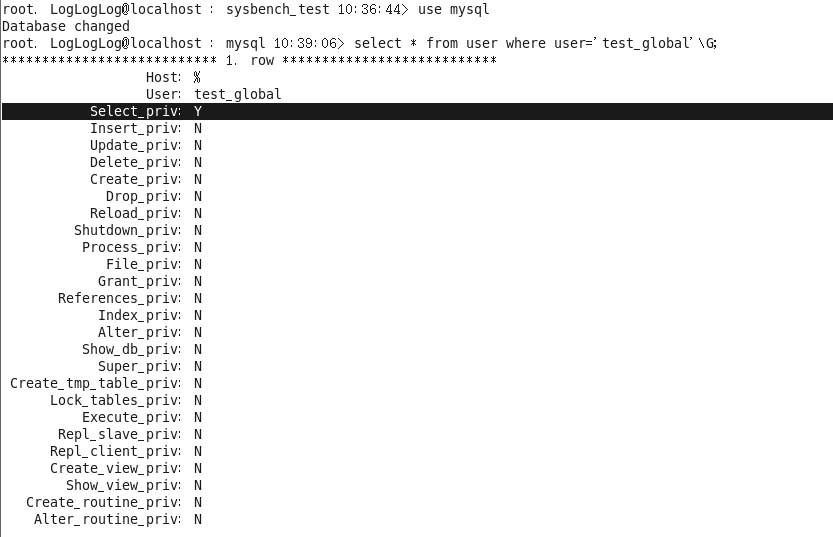
mysql> grant select on **\*.\*** to **test\_global@'%'** identified by 'test';



#查询全局权限表mysql.user来**验证Select\_priv字段值是否为“Y”**，其他字段值是否为“N”

mysql> use mysql

mysql> select \* from user where user='test\_global'\G

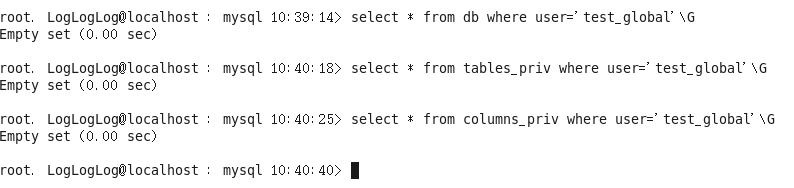


#查询库级别权限表db、表级别权限表tables\_priv和字段级别权限表columns\_priv，可发现没有权限。

mysql> select \* from db where user='test\_global'\G

mysql> select \* from tables\_priv where user='test\_global'\G

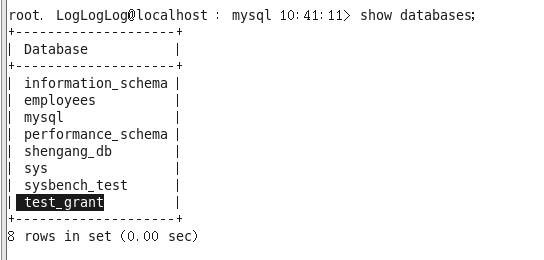
mysql> select \* from columns\_priv where user='test\_global'\G



**17.1.2　查看库级别权限**

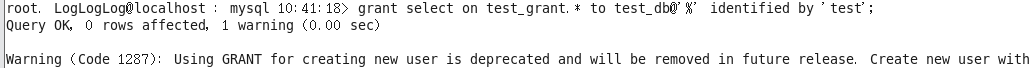
#创建一个测试数据库

mysql> create database test\_grant;



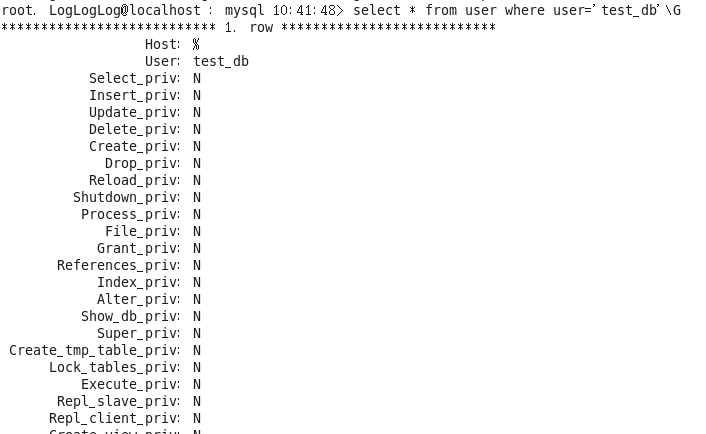
#创建一个具有**库级别权限**的用户test\_db并授予Select权限。库级别权限在操作上使用db\_name.\*形式表示。

mysql> grant select on **test\_grant.\*** to **test\_db@'%'** identified by 'test';

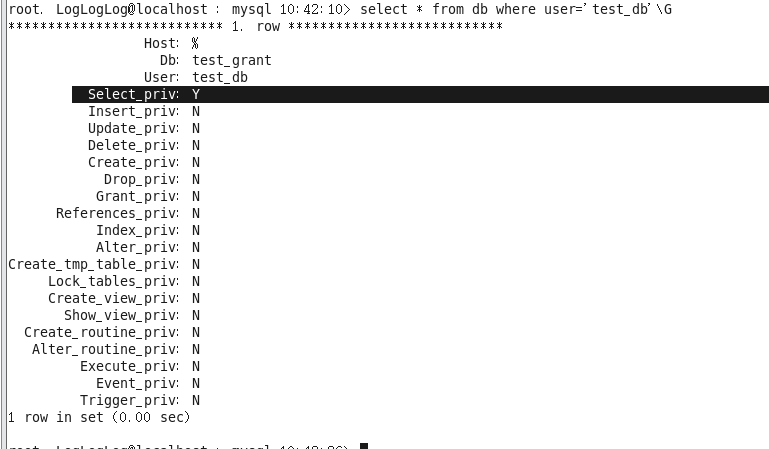


#查询全局权限表mysql.user，发现test\_db用户在该表中有权限记录，但是**所有权限标记字段值为“N”**。确认test\_db用户**并不具有全局权限**，但在该表中存在着test\_db用户的权限记录，是因为要在**这里保存密码加密字符串、密码过期时间、账号锁定状态等信息**。

mysql> select \* from user where user='test\_db'\G



mysql> select \* from db where user='test\_db'\G



#查询表级别权限表tables\_priv和字段级别权限表columns\_priv，同样发现没有权限。

mysql> select \* from tables\_priv where user='test\_db'\G

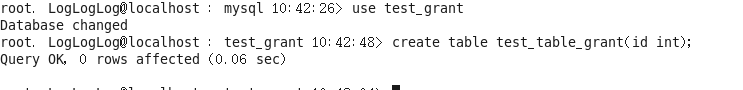
mysql> select \* from columns\_priv where user='test\_db'\G

**17.1.3　查看表级别权限**

#创建一个测试表

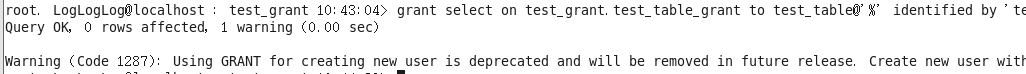
mysql> use test\_grant

mysql> create table test\_table\_grant(id int);



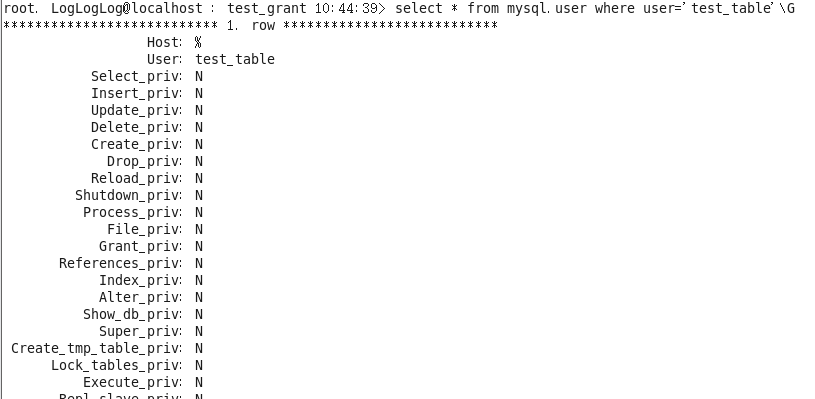
#然后创建一个具有表级别权限的用户test\_table，并授予Select权限。

mysql> grant select on **test\_grant.test\_table\_grant to test\_table@'%'** identified by 'test';



# test\_table用户并**不具有全局权限**，但该表中存在着test\_table用户的权限记录，是因为要在这里保存密码加密字符串、密码过期时间、账号锁定状态等信息。

mysql> select \* from mysql.user where user='test\_table'\G

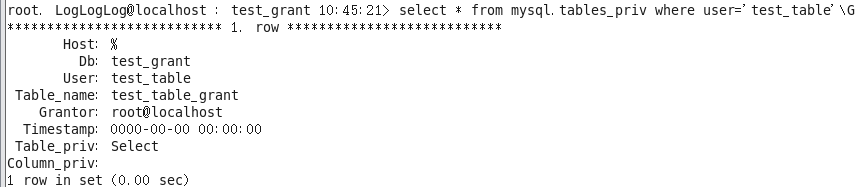


#**只有mysql.tables\_priv表**中存在test\_table用户的权限记录。

mysql> select \* from mysql.db where user='test\_table'\G



mysql> select \* from mysql.tables\_priv where user='test\_table'\G



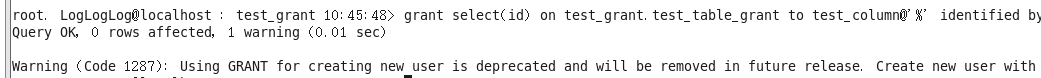
mysql> select \* from mysql.columns\_priv where user='test\_table'\G



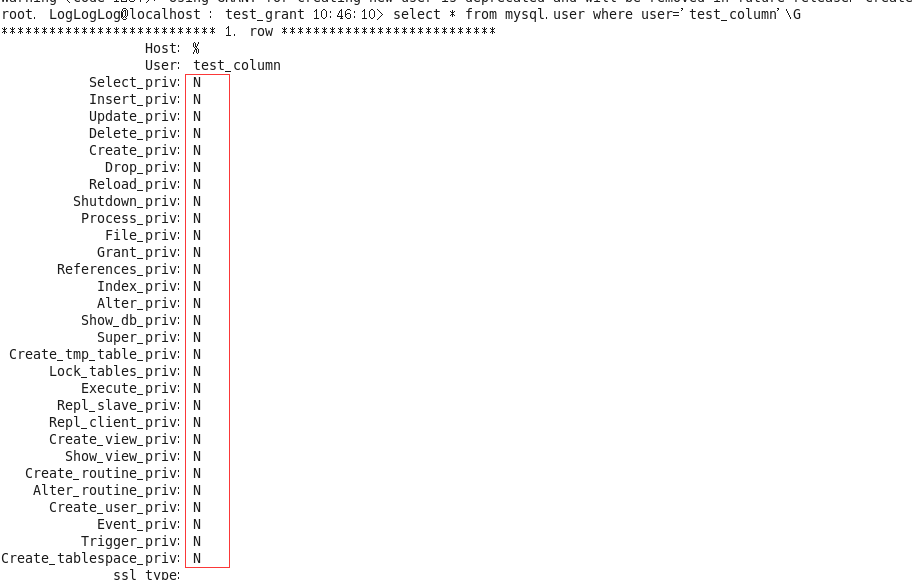
**17.1.4　查看字段级别权限**

#创建一个对**test\_table\_grant表**具有字段级别权限的**用户test\_column**，并授予id字段的Select权限。

mysql> grant **select(id)** on test\_grant.**test\_table\_grant** to test\_column@'%' identified by 'test';



mysql> select \* from mysql.**user** where user='test\_column'\G

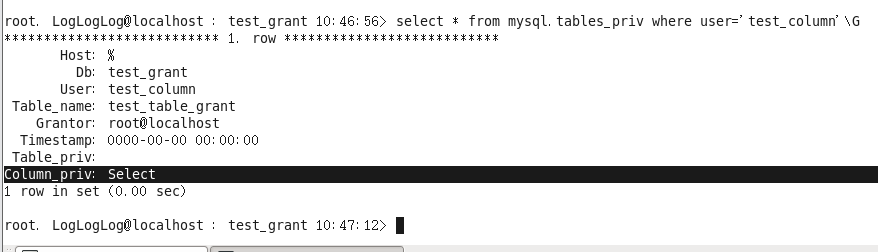


mysql> select \* from mysql.**db** where user='test\_column'\G



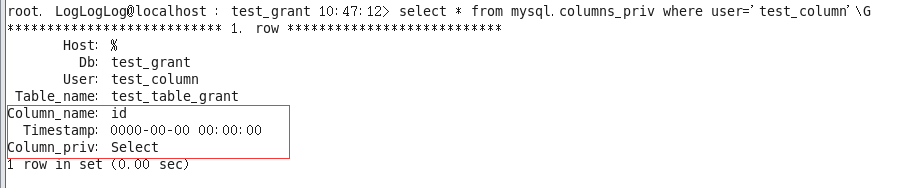
# 通过对mysql.tables\_priv表的查询发现，**test\_column用户对test\_table\_grant表具有字段级别的Select权限**。

mysql> select \* from mysql.tables\_priv where user='test\_column'\G



#由于字段级别权限是针对具体的字段授予的，这些信息需要通过**mysql.columns\_priv表**来查询，如下所示

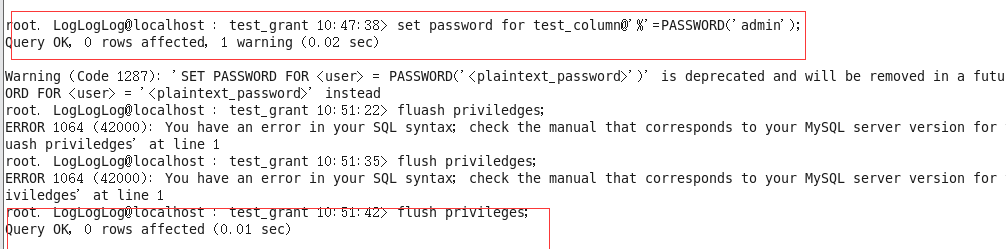
mysql> select \* from mysql.**columns\_priv** where user='test\_column'\G



#提示：以上示例中未涉及的权限管理请读者自行尝试，这里不再赘述。

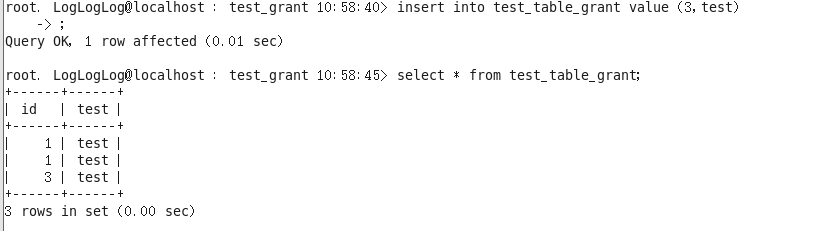
**补充：测试列访问权限的设置。首先初始化test\_column的密码：**

**set password for test\_column@'%'=PASSWORD('admin');**

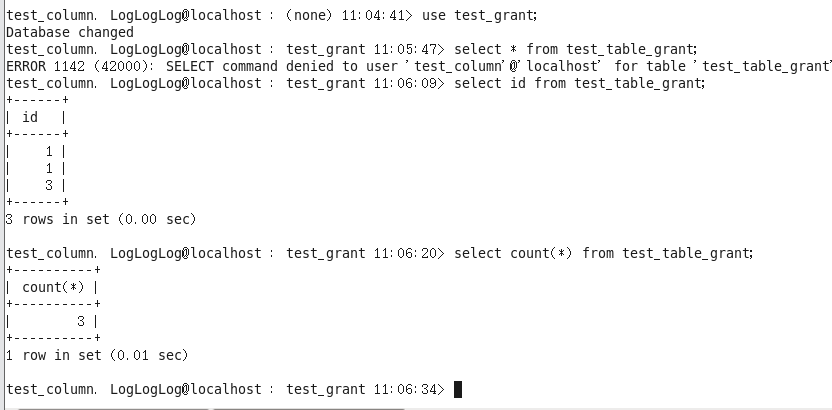


**并且为表test\_table\_grant多增加一列(自行插入一些数据)：**

**alter table test\_table\_grant add test varchar(10) default 'test';**



**然后用test\_column登录mysql测试权限设置：**



**总结：再次理解权限访问控制的的基本次序原理：**

