# Database System Lab 4

## 1.实验目的

1.熟悉通过SQL进行用户权限控制的方法。

2.探究附加权限控制后SQL的允许与限制。

## 2.实验内容

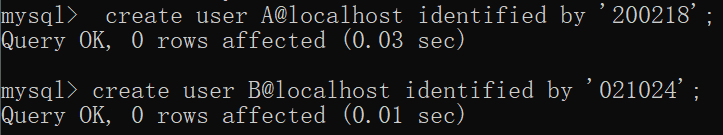
1.建立表，考查表的生成者拥有该表的权限。

2.使用SQL的grant和revoke命令对其他用户进行授权和权力回收，考查相应的作用与限制及with grant option 语句的功能。

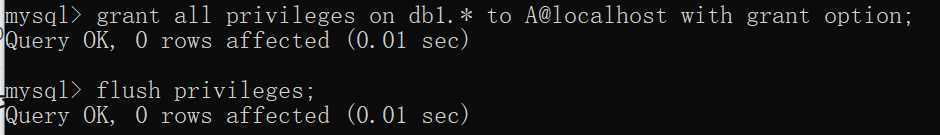
3.建立视图，并把该视图的查询权限授予其他用户，考察通过数据进行权限控制的作用。

## 3.实验步骤

3.1 使用create user语句创立新的角色，代码实现及结果见下：

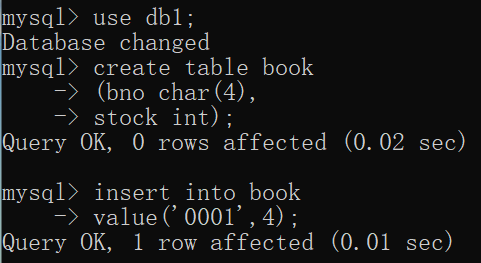


3.2 在root账户的权限下，建立数据库 db1，并用grant 语句向用户A赋予使用db1数据库中所有表的所有权限，并通过使用with grant option语句，使得用户A可以将这些权限级联分发，代码及实现结果如下：

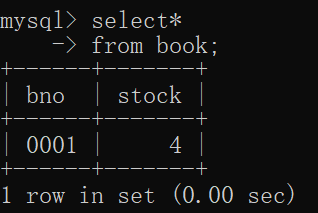


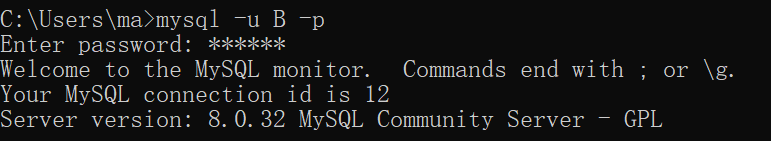
## 33.3 使用A用户登录数据库db1，并创建表book(bno char(4), stock int)，并插入一条记录(‘0001’，4)，观察数据库管理器的反映，代码及运行结果如下：

1. 成功使用用户A登录数据库



由上述结果可以看出，用户A成功获得了使用数据库db的所有权限，因而可以建立book表，并向其中插入数据，我们可以调用select语句，查看数据是否正确的插入了book表中，结果如下（数据已被正确插入）。



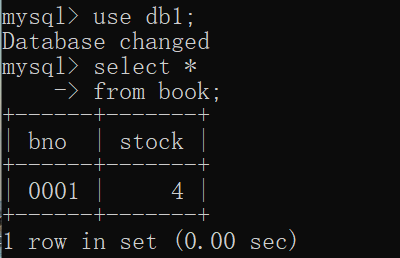
3.4 使用用户B登入，发现B没有权限访问数据库db1，代码及结果如下：

7

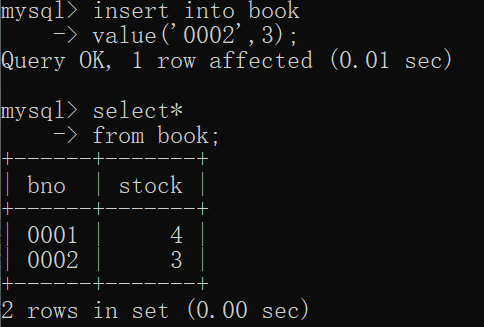
3.5 因为我们在给A分配权限时，使用了with grant option语句，因而我们可以通过A来为B分配权限；因此，我们可以使用grant语句为B分配insert和select曲线，代码如下：

8

3.6 在分配权限之后，我们再测试B用户是否可以进行相应的操作，代码及运行结果如下：



## 查询结果正常

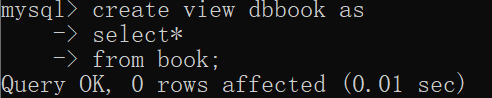


（2）插入后查询同样有效

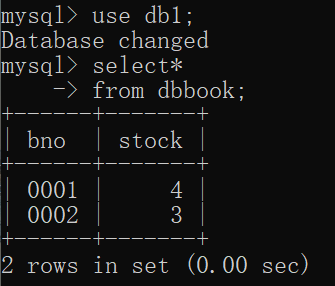
## 143.7使用A用户登录，用revoke命令回收B用户对book的插入权限，再次用B用户登录测试权限，测试代码及结果见下：

由上述结果我们可以看到，在收回B的插入权限后，当我们使用用户B登录，并再次尝试向book表中插入数据时，系统产生报错。

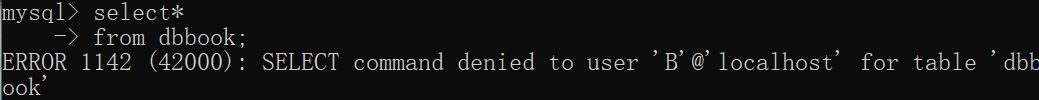
3.8 使用root用户登录，在数据库db1中建立book表的一个视图，并分别测试用户A和用户B对视图的访问权限，测试结果如下：



（1）建立视图



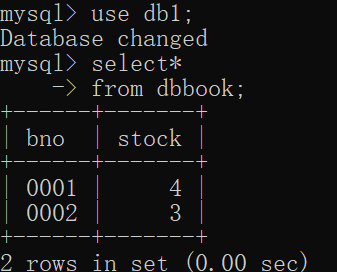
（2）使用用户A登录后对视图的访问



（3）使用用户B登录后对视图的访问

由上述结果可以看出，当我们使用用户A登录时，可以正常访问该视图，因为我们赋予A的是全部的权限；而使用用户B登录时，我们则无法正常的访问视图，因为用户B的权限仅限于book表的插入。

3.9使用root用户登录，将视图的select权限赋予用户B，再次测试用户B对该视图的访问权限，代码运行及结果如下：

18

由上述结果我们可以看出，随着用户root将相应的权限赋予用户B，B可以使用select操作访问新建的视图，这也说明权限赋予的正确性。

## 实验感想

较为简短的一个实验，确乎很好的体会了数据库中权限的发放与收回问题，就可能在角色（非用户）权限发放及级联收回方面的实践理解有所欠缺；当然，我在实验过程中也遇到了一些问题，算是一些细微之处的实践心得。

首先，想要操作一个数据库的权限收回或发放，我们必须使用use语句，切换到目标数据库；其次，在赋予权限操作时，仅使用grant整个数据库权限的操作似乎不被允许，即语句运行成功了，但是实际权限并没有发放（特别是create）权限，因而我在all后面附加了privileges对操作加以明确，并在数据库名后添加了.\*来表示数据库中所有的表格，这样的语句在运行后，才真正的将捐献分配给了目标用户；另外，还有上述privilege的使用，好像只有在all的范围限定下，才可以使用privilege，若在另外实际操作词后（如select，insert）添加privilege，系统会产生报错；最后，是用户名的问题，在权限分配的操作中，如果设置了@localhost 仅支持本地登录，则所有权限操作语句中，都必须将这一点表现出来，即在用户名称后跟上@localhost，不然系统会默认认为其为@all登录地限制，因而产生错误。