

实验名称

一、 实验目的

- 熟悉通过 SQL 进行数据安全性控制的方法。

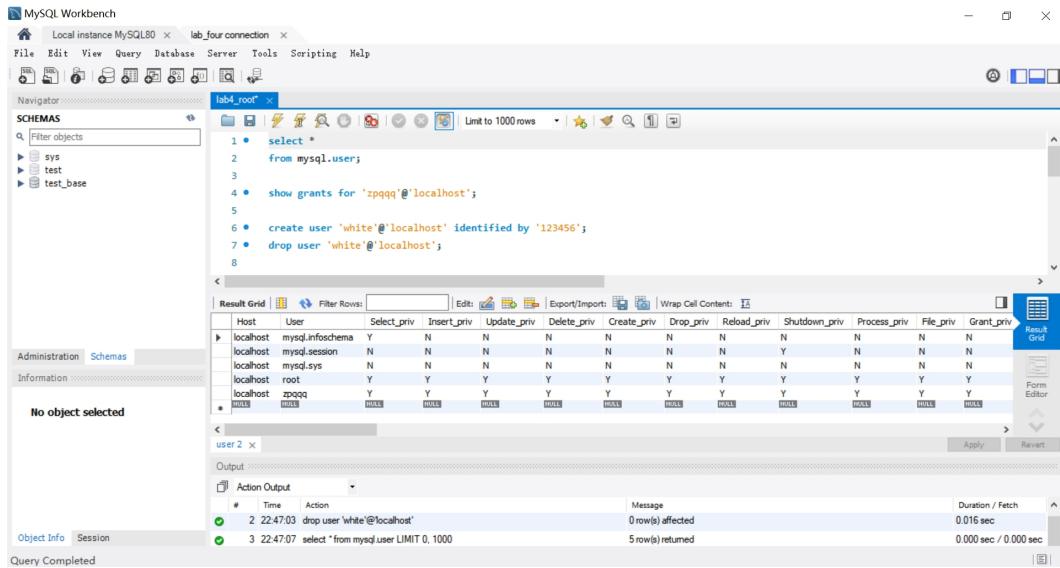
二、 实验环境

- 数据库管理系统：MySQL

三、 实验流程

1. 创建用户

登录 root 用户。如图 3.1.1 所示，查看用户，可以看见除了系统预设的用户外，没有普通用户。



The screenshot shows the MySQL Workbench interface. In the top navigation bar, 'File', 'Edit', 'View', 'Query', 'Database', 'Server', 'Tools', 'Scripting', and 'Help' are visible. Below the toolbar, there's a 'Navigator' pane showing 'SCHEMAS' with 'sys', 'test', and 'test_base'. The main area is titled 'lab4.root' and contains a SQL editor with the following code:

```
1 • select *
2   from mysql.user;
3
4 • show grants for 'zpqqq'@'localhost';
5
6 • create user 'white'@'localhost' identified by '123456';
7 • drop user 'white'@'localhost';
8
```

Below the SQL editor is a 'Result Grid' table with columns: Host, User, Select_priv, Insert_priv, Update_priv, Delete_priv, Create_priv, Drop_priv, Reload_priv, Shutdown_priv, Process_priv, File_priv, Grant_priv. The table lists several users, including 'mysql.infoschema', 'mysql.session', 'mysql.sys', 'root', and 'zpqqq'. The 'white' user is not yet present in the grid.

The bottom section shows the 'Output' tab with the following log entries:

Action	Time	Action	Message	Duration / Fetch
drop user	2 22:47:03	white'@'localhost'	0 row(s) affected	0.016 sec
select *	3 22:47:07	from mysql.user LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec

图 3.1.1 查看用户

然后通过语句创建一个名为 white 的普通用户。创建后查看用户，可以看见已经多出了一个名为 white 的普通用户，如图 3.1.2。

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the connection is set to 'Local instance MySQL80' and the schema is 'lab_four connection'. The main window displays a query editor titled 'lab4_root' containing the following SQL code:

```

1 • select *
  2   from mysql.user;
  3
  4 • show grants for 'zpqqq'@'localhost';
  5
  6 • create user 'white'@'localhost' identified by '123456';
  7 • drop user 'white'@'localhost';
  8

```

Below the code, the 'Result Grid' tab is selected, showing a table of privileges for various hosts and users. A row for the newly created user 'white' is highlighted with a red border. The 'Action Output' tab at the bottom shows two log entries:

- # 1 Time Action Message Duration / Fetch
 - 4 22:47:43 create user 'white'@'localhost' identified by '123456' 0 row(s) affected 0.000 sec
 - 5 22:47:48 select * from mysql.user LIMIT 0, 1000 6 row(s) returned 0.000 sec / 0.000 sec

图 3.1.2 创建用户

然后登录该用户，只需要在主页点击加号，输入用户名与密码即可。

2. 考察普通用户权限

此处实验基本采用上一次实验创建的数据库 test_base，其中有数据表 employee 和 baclub。如图 3.2.1，在普通用户的连接内使用数据库，或者查看数据表，均提示操作被拒绝（denied）。

The screenshot shows the MySQL Workbench interface with a connection named 'lab_four connection'. The main window displays a query editor titled 'lab4_white' containing the following SQL code:

```

1 • use test_base;
2 • select * from test_base.employee;
3
4 • use test_base;
5 • create table authorization(
  6   X varchar(10),
  7   Y varchar(10) default 'N',
  8   Z varchar(10) default 'N',
  9   primary key (X)
10 );
11 • drop table authorization;
12 • drop table test_case_2.testone;
13 • drop database test_case_2;
14
15 • select *
  16   from id_name;
17
18 • update id_name

```

The first two lines of the code are highlighted with a red border. Below the code, the 'Action Output' tab shows two error messages:

- # 1 Time Action Message Duration / Fetch
 - 1 22:48:27 use test_base Error Code: 1044. Access denied for user 'white'@'localhost' to database 'test_base' 0.000 sec
 - 2 22:48:29 select * from test_base.employee LIMIT 0, 1000 Error Code: 1142. SELECT command denied to user 'white'@'localhost' for table 'e...' 0.000 sec

图 3.2.1 操作被拒绝

向 white 用户授予在数据库 test_base 上创建、删除的权限后，查看用户 white 的权限，可见其有了相应的权限记录，如图 3.2.2。

```

14 • create database test_2;
15 • show grants for 'white'@'localhost';
16 • select *
17   from mysql.tables_priv
18   where table_name='employee';
19   # grant select on test_base.employee to 'white'@'localhost';
20   # revoke select on test_base.employee from 'white'@'localhost';
21 • grant create, drop on test_base.* to 'white'@'localhost';

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | 
Grants for white@localhost
GRANT USAGE ON `*.*` TO 'white'@'localhost'
GRANT CREATE, DROP ON `test_base.*` TO ...

```

No object selected

Object Info Session

Query Completed

图 3.2.2 查看授权

此时查看表的权限记录，却发现向 white 用户授予在数据库 test_base 上创建、删除的权限后，用户对 test_base 内两个数据表的权限没有记录在此，如图 3.2.3。

```

14 • create database test_2;
15 • show grants for 'white'@'localhost';
16 • select *
17   from mysql.tables_priv
18   where table_name='employee';
19   # grant select on test_base.employee to 'white'@'localhost';
20   # revoke select on test_base.employee from 'white'@'localhost';
21 • grant create, drop on test_base.* to 'white'@'localhost';

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | 
Host Db User Table_name Grantor Timestamp Table_priv Column_priv

```

No object selected

Object Info Session

Query Completed

图 3.2.3 表的权限未记录

作为对比，如图 3.2.4，再向用户授予查询某一特定表的权利。这时再查看表的权限记录，可见 mysql.tables_priv 内记录了相应的信息。

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** The current schema is `lab4_root`.
- SQL Editor:** Contains the following SQL code:


```

17   from mysql.tables_priv
18   where table_name='employee';
19   # grant select on test_base.employee to 'white'@'localhost';
20   # revoke select on test_base.employee from 'white'@'localhost';
21   grant create, drop on test_base.* to 'white'@'localhost';
22   grant select on test_base.* to 'white'@'localhost';
23   grant select on test_base.employee to 'white'@'localhost';
24   grant create, drop on test_2.* to 'white'@'localhost';
      
```
- Result Grid:** Shows the execution results of the grant statements. One row is highlighted with a red border:

Host	Db	User	Table_name	Grantor	Timestamp	Table_priv	Column_priv
localhost	test_base	white	employee	root@localhost	0000-00-00 00:00:00	Select	
- Action Output:** Shows the log of executed actions:

#	Time	Action	Message	Duration / Fetch
12	22:55:49	grant select on test_base.employee to 'white'@'localhost'	0 row(s) affected	0.016 sec
13	22:55:53	select * from mysql.tables_priv where table_name='employee' LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

图 3.2.4 授予对表的权利

回到普通用户的连接内，再次执行一开始被拒绝的两条指令，可见两条指令都能被执行了，并且普通用户在被授权后成功查询到了相关数据，如图 3.2.5。

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** The current schema is `lab1_white`.
- SQL Editor:** Contains the following SQL code:


```

1 use test_base;
2 select * from test_base.employee;
      
```
- Result Grid:** Shows the results of the `select` query:

id	name	gender	age	grade
1000	xx	F	40	7
1001	yy	F	40	3
1002	zz	M	40	1
1003	ii	M	40	4
1004	jj	F	40	5
1007	dd	M	40	5
- Action Output:** Shows the log of executed actions:

#	Time	Action	Message	Duration / Fetch
3	22:56:19	use test_base	0 row(s) affected	0.000 sec
4	22:56:23	select * from test_base.employee LIMIT 0, 1000	6 row(s) returned	0.000 sec / 0.000 sec

图 3.2.5 授权后查询

回到 root 用户的连接内，收回授予用户 white 的对于数据库 test_base 的权限。此时查询数据表的权限，发现用户 white 对 test_base 内数据表 employee 查询的权限还在，如图 3.2.5。

The screenshot shows the MySQL Workbench interface with a query editor window titled 'lab4_root'. The code entered is:

```

22 • grant select on test_base.* to 'white'@'localhost';
23 • grant select on test_base.employee to 'white'@'localhost';
24 • grant create, drop on test_2.* to 'white'@'localhost';

26 • revoke create, drop, select on test_base.* from 'white'@'localhost';
27
28 • create table test_case_2.testone(
29   id char(5),

```

The result grid shows a single row of data:

Host	Db	User	Table_name	Grantor	Timestamp	Table_priv	Column_priv
localhost	test_base	white	employee	root@localhost	0000-00-00 00:00:00	Select	

The 'tables_priv' table in the 'Output' section shows two rows of data:

#	Time	Action	Message	Duration / Fetch
14	22:58:50	revoke create, drop, select on test_base.* from 'white'@'localhost'	0 row(s) affected	0.015 sec
15	22:58:54	select * from mysql.tables_priv where table_name='employee' LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

图 3.2.6 撤回对数据库的权限

回到普通用户的连接内，再次执行头两条指令，发现两条指令依然能被执行，如图 3.2.7。

The screenshot shows the MySQL Workbench interface with a query editor window titled 'lab1_white'. The code entered is:

```

1 • use test_base;
2 • select * from test_base.employee;
3
4 • use test_base;
5 • create table authorization(
6   X varchar(10),
7   Y varchar(10) default 'N',
8   Z varchar(10) default 'N',

```

The result grid shows the data from the 'employee' table:

id	namee	gender	age	grade
1000	xx	F	40	7
1001	yy	F	40	3
1002	zz	M	40	1
1003	ii	M	40	4
1004	jj	F	40	5
1007	dd	M	40	5

The 'Output' section shows the execution of the 'use' and 'select' statements:

#	Time	Action	Message	Duration / Fetch
5	22:59:15	use test_base	0 row(s) affected	0.000 sec
6	22:59:21	select * from test_base.employee LIMIT 0, 1000	6 row(s) returned	0.015 sec / 0.000 sec

图 3.2.7 未撤回对表的权限

回到 root 用户的连接内，收回授予用户 white 查询数据表的权限，如图 3.2.8。

The screenshot shows the MySQL Workbench interface with a connection named 'lab_four connection'. In the 'Navigator' pane, under the 'SCHEMAS' section, the 'test_base' schema is selected. In the main query editor window, the following SQL code is shown:

```

22 • grant select on test_base.* to 'white'@'localhost';
23 • grant select on test_base.employee to 'white'@'localhost';
24 • grant create, drop on test_2.* to 'white'@'localhost';
25
26 • revoke create, drop, select on test_base.* from 'white'@'localhost';
27 • revoke select on test_base.employee from 'white'@'localhost';
28
29 • create table test_case_2.testone(

```

Below the code, the 'Result Grid' tab is active, showing a table with columns: Host, Db, User, Table_name, Grantor, Timestamp, Table_priv, and Column_priv. The 'Output' tab at the bottom shows the execution log:

Action	Time	Message	Duration / Fetch
revoke select on test_base.employee from 'white'@'localhost'	23:00:05	0 row(s) affected	0.000 sec
select * from mysql.tables_priv where table_name='employee' LIMIT 0, 1000	23:00:24	0 row(s) returned	0.000 sec / 0.000 sec

图 3.2.8 撤回查询数据表的权限

回到普通用户的连接内，再次执行头两条指令，发现两条指令被拒绝执行，如图 3.2.9。

The screenshot shows the MySQL Workbench interface with a connection named 'lab_four connection'. In the 'Navigator' pane, under the 'SCHEMAS' section, the 'test_base' schema is selected. In the main query editor window, the following SQL code is shown:

```

1 • use test_base;
2 • select * from test_base.employee;
3
4 • use test_base;
5 • create table authorization(
6     X varchar(10),
7     Y varchar(10) default 'N',
8     Z varchar(10) default 'N',
9     primary key (X)
10 );
11 • drop table authorization;
12 • drop table test_case_2.testone;
13 • drop database test_case_2;
14
15 • select *
16     from id_name;
17
18 • update id_name

```

Below the code, the 'Output' tab at the bottom shows the execution log:

Action	Time	Message	Duration / Fetch
use test_base	23:00:10	Error Code: 1044. Access denied for user 'white'@'localhost' to database 'test_b...' 0.000 sec	
select * from test_base.employee LIMIT 0, 1000	23:00:12	Error Code: 1142. SELECT command denied to user 'white'@'localhost' for table... 0.000 sec	

图 3.2.9 指令被拒绝执行

3. 考察表的生成者拥有该表的哪些权限

如图 3.3.1，首先用户 white 授权。

The screenshot shows the MySQL Workbench interface with a query editor titled 'lab4_root'. The code in the editor is:

```

19 # grant select on test_base.employee to 'white'@'localhost';
20 # revoke select on test_base.employee from 'white'@'localhost';
21 • grant create, drop on test_base.* to 'white'@'localhost';
22 • grant select on test_base.* to 'white'@'localhost';
23 • grant select on test_base.employee to 'white'@'localhost';
24 • grant create, drop on test_2.* to 'white'@'localhost';

25
26 • revoke create, drop, select on test_base.* from 'white'@'localhost';
27 • revoke select on test_base.employee from 'white'@'localhost';
28

29 • create table test_case_2.testone(
30   id char(5),
31   primary key(id));
32

33 • create view test_base.id_name as
34   select id, name
35   from test_base.employee;
36 • drop view test_base.id_name;

```

The output window shows two successful grants:

Action	Time	Message	Duration / Fetch
grant create, drop on test_base.* to 'white'@'localhost'	19 23:02:08	0 row(s) affected	0.000 sec
grant create, drop on test_2.* to 'white'@'localhost'	20 23:02:11	0 row(s) affected	0.000 sec

图 3.3.1 授权

此后，在普通用户的连接内创建数据表，如图 3.3.2。

The screenshot shows the MySQL Workbench interface with a query editor titled 'lab1_white'. The code in the editor is:

```

1 • use test_base;
2 • select * from test_base.employee;
3
4 • use test_base;
5 • create table authorization(
6   X varchar(10),
7   Y varchar(10) default 'N',
8   Z varchar(10) default 'N',
9   primary key (X)
10 );
11 • drop table authorization;
12 • drop table test_case_2.testone;
13 • drop database test_case_2;
14
15 • select *
16   from id_name;
17
18 • update id_name

```

The output window shows the creation of the 'authorization' table:

Action	Time	Message	Duration / Fetch
use test_base	9 23:02:35	0 row(s) affected	0.000 sec
create table authorization(X varchar(10), Y varchar(10) default 'N', Z varchar(10) default 'N', primary key (X));	10 23:02:38	0 row(s) affected	0.047 sec

图 3.3.2 创建数据表

回到 root 用户的连接内，刷新后可在左侧看到新创建的数据表，如图 3.3.3。

```

MySQL Workbench
File Edit View Query Database Server Tools Scripting Help
lab4_root >
Schemas
Filter objects
sys
test
test_2
test_base
Tables
authorization
badlub
employee
Views
Stored Procedures
Functions
Administration Schemas
Information
Table: authorization
Columns:
X varchar(10) PK
Y varchar(10)
Z varchar(10)
Object Info Session
Ready

```

```

19 # grant select on test_base.employee to 'white'@'localhost';
20 # revoke select on test_base.employee from 'white'@'localhost';
21 • grant create, drop on test_base.* to 'white'@'localhost';
22 • grant select on test_base.* to 'white'@'localhost';
23 • grant select on test_base.employee to 'white'@'localhost';
24 • grant create, drop on test_2.* to 'white'@'localhost';
25
26 • revoke create, drop, select on test_base.* from 'white'@'localhost';
27 • revoke select on test_base.employee from 'white'@'localhost';
28
29 • create table test_case_2.testone(
30   id char(5),
31   primary key(id));
32
33 • create view test_base.id_name as
34   select id, name
35   from test_base.employee;
36 • drop view test_base.id_name;

```

Action Output

#	Time	Action	Message	Duration / Fetch
19	23:02:08	grant create, drop on test_base.* to 'white'@'localhost'	0 row(s) affected	0.000 sec
20	23:02:11	grant create, drop on test_2.* to 'white'@'localhost'	0 row(s) affected	0.000 sec

图 3.3.3 查看数据表

此时，虽然用户 white 是表的创建者，但是其没有查询表的权利，如图 3.3.4，系统提示查询被拒绝。事实上，由于授权，用户 white 只有创建和删除的权限，其余权限均由 root 用户保留。这一点也可以通过查询数据表的授权以及用户的权限来确认。

```

MySQL Workbench
File Edit View Query Database Server Tools Scripting Help
lab1_white >
Schemas
Filter objects
test_2
test_base
Tables could not be fetched
Views could not be fetched
Stored Procedures could not be fetched
Functions could not be fetched
Administration Schemas
Information
No object selected
Object Info Session
Query interrupted

```

```

1 • use test_base;
2 • select * from test_base.employee;
3
4 • use test_base;
5 • create table authorization(
6   X varchar(10),
7   Y varchar(10) default 'N',
8   Z varchar(10) default 'N',
9   primary key (X)
10 );
11 • select * from authorization;
12 • drop table authorization;
13 • drop table test_case_2.testone;
14 • drop database test_case_2;
15
16
17 • select *
18   from id_name;

```

Action Output

#	Time	Action	Message	Duration / Fetch
12	23:04:34	select * from mysql.tables_priv where table_name='authorization' LIMIT 0,1000	Error Code: 1142 SELECT command denied to user 'white'@'localhost' for table...	0.000 sec
13	23:11:20	select * from authorization LIMIT 0,1000	Error Code: 1142 SELECT command denied to user 'white'@'localhost' for table...	0.000 sec

图 3.3.4 查询被拒绝

此外，授权时的“.*”意味着对整个数据库的权限。这一点可以通过删除数据库来确认，如图 3.3.5，用户 white 可以直接删除数据库 test_2。

The screenshot shows the MySQL Workbench interface with a connection named 'lab_four connection'. In the 'Navigator' pane, under 'SCHEMAS', the 'test_base' schema is selected. In the 'Query Editor' pane, the following SQL code is visible:

```

4 • use test_base;
5 • ② create table authorization(
6   X varchar(10),
7   Y varchar(10) default 'N',
8   Z varchar(10) default 'N',
9   primary key (X)
10  );
11 • select * from authorization;
12 • drop table authorization;
13 • drop database test_2; highlighted
14
15
16 • select *
17   from id_name;
18
19 • update id_name
20   set id = id+100
21   where id < 0;

```

In the 'Output' pane, two actions are listed:

#	Time	Action	Message	Duration / Fetch
14	23:12:29	drop database test_case_2	Error Code: 1044. Access denied for user 'white'@'localhost' to database 'test_c...' 0 row(s) affected 0.000 sec	
15	23:12:37	drop database test_2	0 row(s) affected 0.016 sec	

At the bottom, the status bar says 'Query Completed'.

图 3.3.5 删除数据库

4. 视图权限

如图 3.4.1, 在数据表 employee 上创建一个关于 id 与姓名的视图。并把查询与更新的权限授予用户 white。

The screenshot shows the MySQL Workbench interface with a connection named 'lab_four connection'. In the 'Navigator' pane, under 'SCHEMAS', the 'test_base' schema is selected. In the 'Query Editor' pane, the following SQL code is visible:

```

27 • revoke select on test_base.employee from 'white'@'localhost';
28
29 • ② create table test_case_2.testone(
30   id char(5),
31   primary key(id));
32
33 • create view test_base.id_name as
34   select id, namee
35   from test_base.employee;
36 • drop view test_base.id_name;
37 • grant select, update on test_base.id_name to 'white'@'localhost';
38
39 • select * from authorization;
40
41 • select *
42   from mysql.user
43   where user='white';

```

In the 'Output' pane, two actions are listed:

#	Time	Action	Message	Duration / Fetch
32	23:13:32	create view test_base.id_name as select id, namee from test_base.employee	0 row(s) affected 0.016 sec	
33	23:13:45	grant select, update on test_base.id_name to 'white'@'localhost'	0 row(s) affected 0.000 sec	

At the bottom, the status bar says 'Query Completed'.

图 3.4.1 创建视图

如图 3.4.2, 在普通用户的连接内利用视图进行查询, 可见查询能正常查询到结果。

```

12 • drop table authorizations;
13 • drop database test_2;
14
15
16 • select *
from id_name;
17
18
19 • update id_name

```

id	name
1000	xx
1001	yy
1002	zz
1003	ii
1004	jj
1007	dd

Action Output

#	Time	Action	Message	Duration / Fetch
15	23:12:37	drop database test_2	0 row(s) affected	0.016 sec
16	23:14:12	select * from id_name LIMIT 0, 1000	6 row(s) returned	0.000 sec / 0.000 sec

图 3.4.2 视图查询

如图 3.4.3，在普通用户的连接内利用视图进行数据更新，可见数据表的数据被正常更新了，id 为 1003 的行，id 变成了 1103。

```

15
16 • select *
from id_name;
17
18
19 • update id_name
set id = id+100
where id = 1003;
20
21
22

```

id	name
1000	xx
1001	yy
1002	zz
1004	jj
1007	dd
1103	ii

Action Output

#	Time	Action	Message	Duration / Fetch
6	23:16:44	update id_name set id = id+100 where id = 1003	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
7	23:16:47	select * from id_name LIMIT 0, 1000	6 row(s) returned	0.000 sec / 0.000 sec

图 3.4.3 视图更新

四、遇到的问题及解决方法

1. 关于数据库与数据表的授权

通过上述实验，可以看出 MySQL 应该是将对表的授权以及对数据库的授权分开记录。当我撤回了对数据库的权限后，普通用户依然可以对数据表进行查询。

五、总结

此次实验让我了解了数据库权限的设计，这种设计极大地方便了数据库的管理。