

数据库原理及安全实验报告

姓 名	肖轩淦	学号	2016301500327	班级	信安 4 班
实验名称	WebApp			日期	2019.1.11

【实验内容及要求】

数据库大作业

【实验平台】

Windows 10

Ubuntu 16.04 server x64

【实验步骤】

在线测试信息：

<http://165.227.54.253/webapp/index.php>

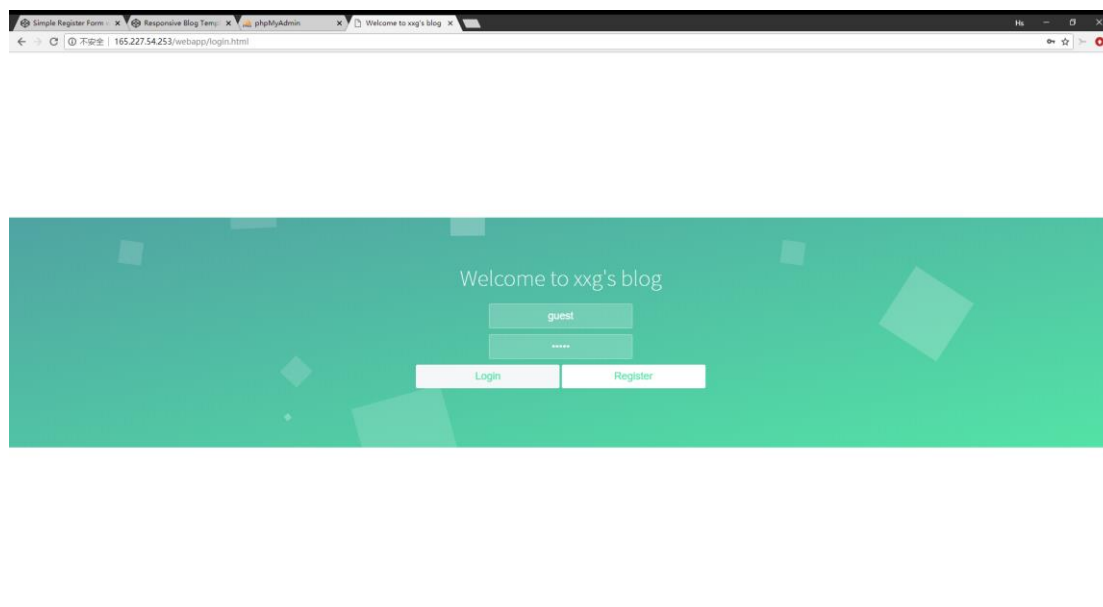
管理员账号：admin 密码 adminpassword

游客账号：guest 密码 123456，也可以自行注册

效果展示：

实现了一个简单的 web 博客，提供了用户注册的功能。Admin 用户可以对博客进行增加、修改、删除操作，而普通用户只能够查看博客。

登陆界面：



注册界面：

← → ↻ ① 不安全 | 165.227.54.253/webapp/register.html

Register

用户名:

登陆密码:

重复密码:

性别: ☐ 男 ☐ 女

个人介绍:

登陆成功界面：

← → ↻ ① 不安全 | 165.227.54.253/webapp/login.php

登录成功

Welcome : admin

Turn to index.php in 3second

管理员登陆主页：

Simple Register Form | Responsive Blog Temp | 165.227.54.253 / local | My blog

← → ↻ ① 不安全 | 165.227.54.253/webapp/index.php

Xxg's Blog

Home add blog Logout

文章3

Posted 2019-01-11

3..

[Read More...](#) | [edit](#) | [delete](#)

文章2

Posted 2019-01-11

2..

[Read More...](#) | [edit](#) | [delete](#)

文章1

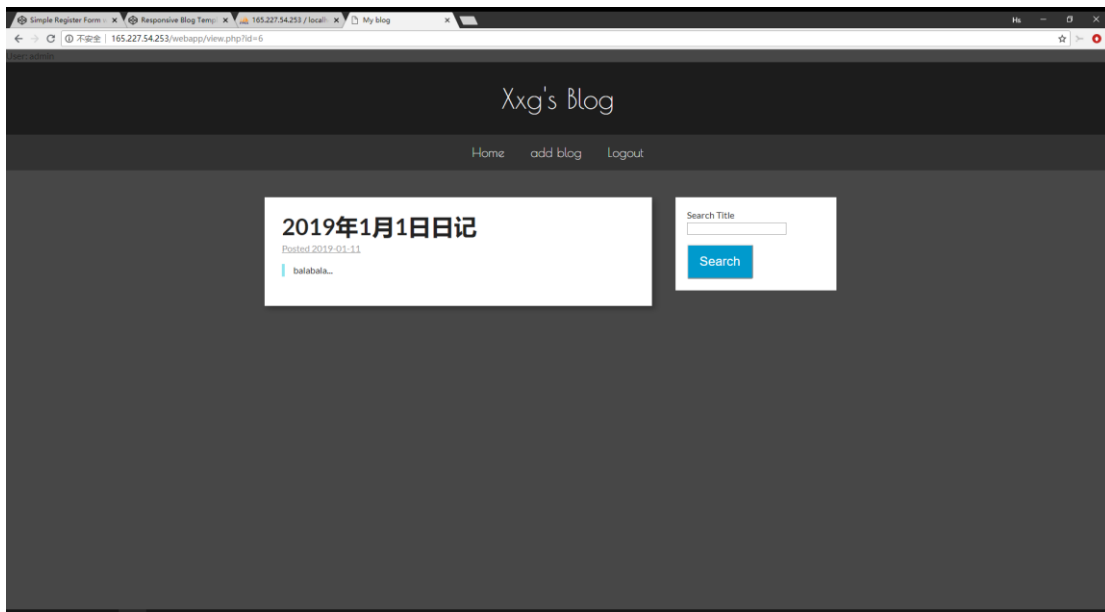
Posted 2019-01-11

1..

[Read More...](#) | [edit](#) | [delete](#)

Search Title

浏览博客：



新增博客界面：

← → ↻ ① 不安全 | 165.227.54.253/webapp/add.php

User: admin
[index](#) [add blog](#) [Logout](#)

title :

contents:

修改博客界面：

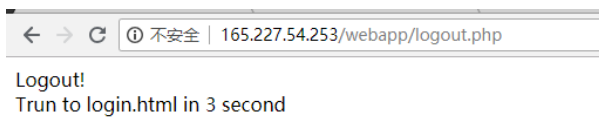
← → ↻ ① 不安全 | 165.227.54.253/webapp/edit.php?id=5

User: admin
[index](#) [add blog](#) [Logout](#)

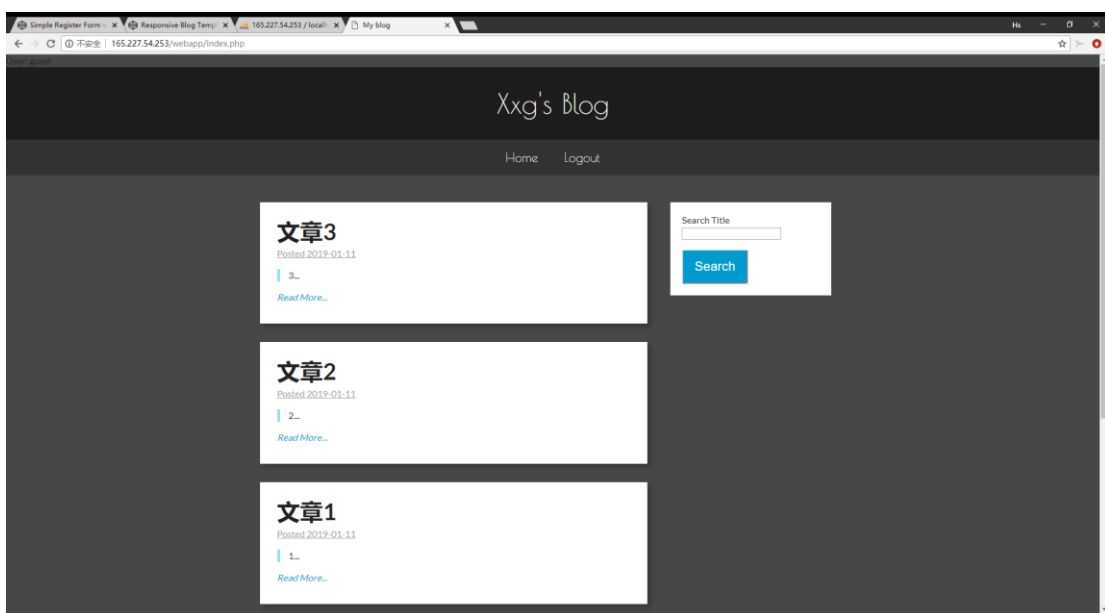
title :

contents:

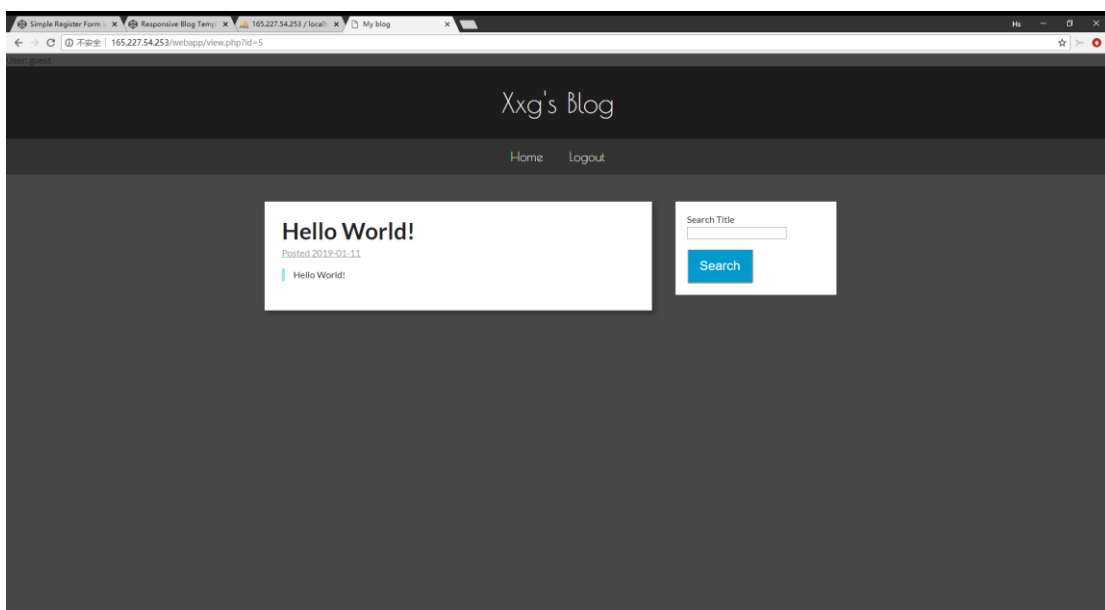
退出登录:



游客界面:



浏览文章界面:



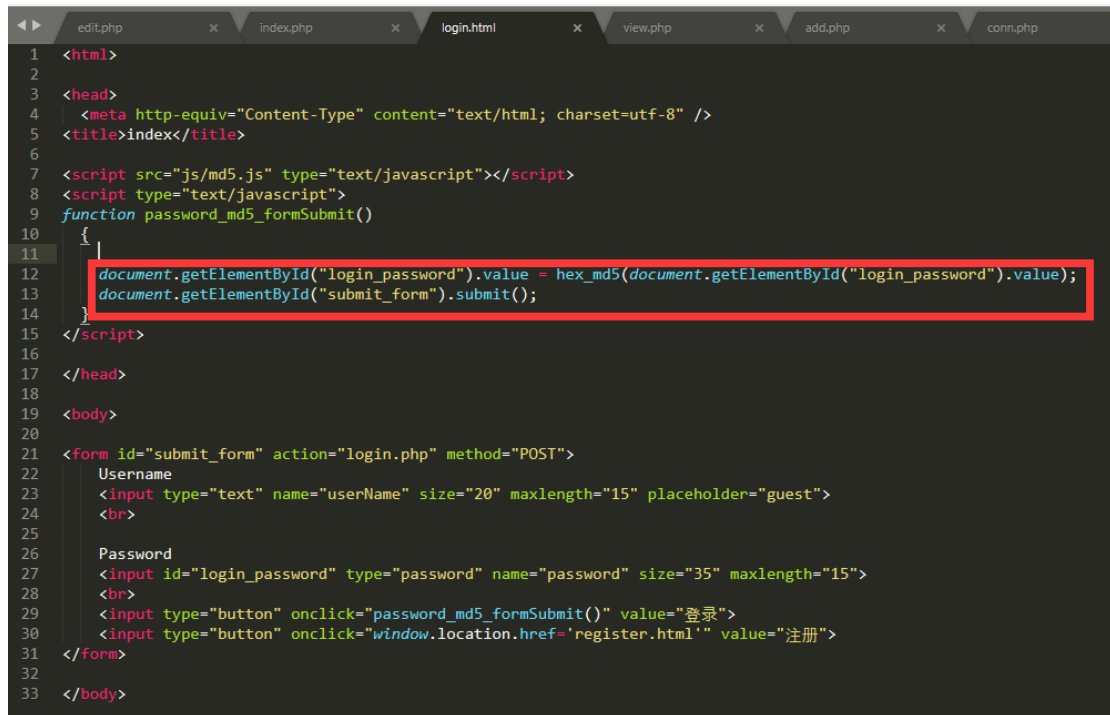
各项要求的实现：

1、 不同角色的登陆、访问控制

如上文中效果展示，实现了 admin 和 guest 及未登录用户的不同权限，admin 具有读写权限，guest 只具有读权限，未登录用户无任何权限。

2、 密码 hash

使用 md5 对密码进行 hash，如在前端使用 hash，降低密码在信道上被截获的危险。



```
1 <html>
2
3 <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
5   <title>index</title>
6
7   <script src="js/md5.js" type="text/javascript"></script>
8   <script type="text/javascript">
9     function password_md5_formSubmit()
10    {
11      document.getElementById("login_password").value = hex_md5(document.getElementById("login_password").value);
12      document.getElementById("submit_form").submit();
13    }
14  </script>
15
16
17 </head>
18
19 <body>
20
21 <form id="submit_form" action="login.php" method="POST">
22   Username
23   <input type="text" name="userName" size="20" maxlength="15" placeholder="guest">
24   <br>
25   Password
26   <input id="login_password" type="password" name="password" size="35" maxlength="15">
27   <br>
28   <input type="button" onclick="password_md5_formSubmit()" value="登录">
29   <input type="button" onclick="window.location.href='register.html'" value="注册">
30 </form>
31
32
33 </body>
```

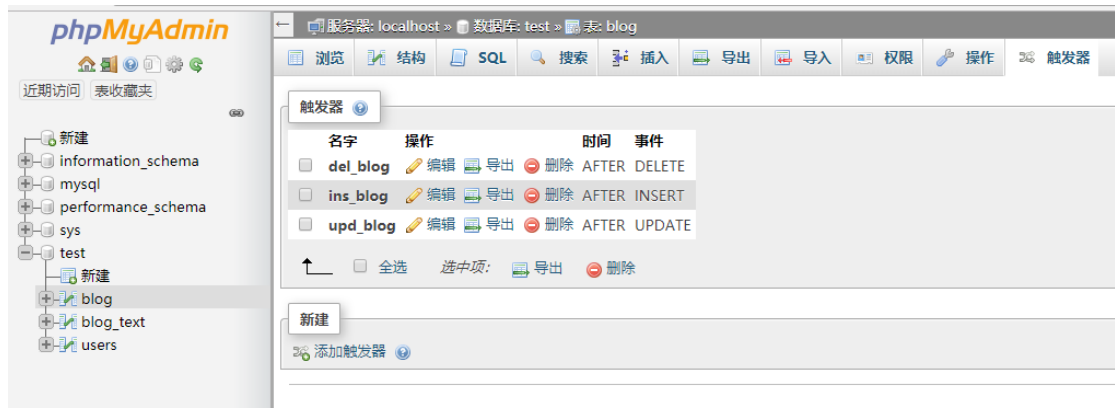
数据库内的密码 hash 存储，防止被脱库获取大量用户明文密码。



user_id	UserName	password	sex	remark
25	admin	e3274be5c857fb42ab72d786e281b4b8	male	
26	guest	7cec85c75537840dad40251576e5b757		

3、 完整性检查、触发器

在 blog 表定义了三个触发器，如下：



```
DELIMITER ;;
CREATE TRIGGER `ins_blog` AFTER INSERT ON `blog` FOR EACH ROW BEGIN
    INSERT INTO blog_text (id, hits, title, date, contents)
    VALUES (new.id, new.hits, new.title, new.date, new.contents);
END;;

CREATE TRIGGER `upd_blog` AFTER UPDATE ON `blog` FOR EACH ROW BEGIN
    IF (old.title != new.title) OR (old.hits != new.hits) OR (old.id != new.id) OR (old.date != new.date) OR (old.contents != new.contents)
    THEN
        UPDATE blog_text
        SET id = new.id,
            hits = new.hits,
            title = new.title,
            date = new.date,
            contents = new.contents
        WHERE id = old.id;
    END IF;
END;;

CREATE TRIGGER `del_blog` AFTER DELETE ON `blog` FOR EACH ROW BEGIN
    DELETE FROM blog_text WHERE id = old.id;
END;;

DELIMITER ;
```

- 4、 对于 sql 注入的防范
在每个输入的地方，均加有防范 sql 注入。

```
File Edit Selection Find View Goto Tools Project Preferences Help
edit.php x index.php x login.html x login.php x view.php x
1 <?php
2 include_once("function/database.php");
3 include_once("function/util.php");
4
5 session_start();
6 if(!empty($_SESSION['UserName'])){
7     echo "You have been login!<br>";
8     auto_jump("index.php");
9     exit();
10 }
11
12
13 // $userName = $_POST['userName'];
14 // $password = $_POST['password'];
15 $userName = addslashes($_POST['userName']);
16 $password = addslashes($_POST['password']);
17 getConnect();
18 $loginSQL = "select * from users where UserName='$userName' and password='$password'";
19 //echo $loginSQL;
20 $resultLogin = mysql_query($loginSQL);
21
22 $user = mysql_fetch_array($resultLogin);
23
24 if (mysql_num_rows($resultLogin) > 0) {
25     echo "登录成功";
26     echo "<br>". "Welcome : ".$user['UserName'];
27     set_session($user['UserName']);
28     echo "<br>". "Turn to index.php in 3second";
29
30     auto_jump("index.php");
31 } else {
32     echo "登录失败";
33
34     auto_jump("login.php");
35 }
36 closeConnect();
37 ?>
```

5、并发测试

由于测试性能与主机的性能关系较大，测试主机配置为：1 CPU、1GB 内存
使用 100 个并发连接、100000 次查询，结果如下：

```
root@ubuntu-s-1vcpu-1gb-sfo2-01:/var/www/html# mysqlslap --concurrency=100 --iterations=1 --create-schema='test' --query='select * from blog;' --number-of-queries=100000 --uroot -p
Enter password:
Benchmark
Average number of seconds to run all queries: 9.990 seconds
Minimum number of seconds to run all queries: 9.990 seconds
Maximum number of seconds to run all queries: 9.990 seconds
Number of clients running queries: 100
Average number of queries per client: 1000
```

1000 个并发、100000 次查询，结果：

```
root@ubuntu-s-1vcpu-1gb-sfo2-01:/var/www/html# mysqlslap --concurrency=1000 --iterations=1 --create-schema='test' --query='select * from blog;' --number-of-queries=100000 --uroot -p
Enter password:
Benchmark
Average number of seconds to run all queries: 11.458 seconds
Minimum number of seconds to run all queries: 11.458 seconds
Maximum number of seconds to run all queries: 11.458 seconds
Number of clients running queries: 1000
Average number of queries per client: 100
```

6、事务的实现

在 register.php 中：

```
29 $registerSQL = "insert into users values(null, '$userName', '$password', '$sex', '$remark')";
30
31 echo $registerSQL."<br>";
32
33 mysqli_query($conn, "SET AUTOCOMMIT=0"); // 设置为不自动提交，因为MYSQL默认立即执行
34 mysqli_begin_transaction($conn);         // 开始事务定义
35
36 if(!mysql_query($registerSQL))
37 {
38     mysqli_query($conn, "ROLLBACK");      // 判断当执行失败时回滚
39 }
40 mysqli_commit($conn);                     // 执行事务
```

新增博客的 Add.php 中：

```

28 $sql= "insert into blog values(null,'0','$title',now(),'$con')";
29
30 mysqli_query($conn, "SET AUTOCOMMIT=0"); // 设置为不自动提交, 因为MySQL默认立即执行
31 mysqli_begin_transaction($conn); // 开始事务定义
32
33 if(!mysqli_query($sql))
34 {
35     mysqli_query($conn, "ROLLBACK"); // 判断当执行失败时回滚
36 }
37 mysqli_commit($conn); // 执行事务
38

```

会判断是否执行成功, 失败时回滚数据库。

7、 读写锁

如在 index.php 中使用了共享锁:

```

25
26
27 <ul>
28 <li class="nav-item"><a href="index.php">Home</a></li>
29 </ul>
30 <php
31 if($current_user == "admin")
32     echo "<li class='nav-item'><a href='add.php'>add blog</a></li>";
33 </php>
34 <ul>
35 <li class="nav-item"><a href="logout.php">Logout</a></li>
36 </ul>
37 <div class="menu-bar">Menu
38 <span class="hamburger-icon"><i class="fa fa-bars"></i></span>
39 </div>
40
41 <div class="container">
42 <div class="action">
43 <div class="col span_2_of_3">
44
45
46
47 <?php
48 include_once("conn.php"); // 引入连接数据库
49
50 if (empty($_GET['keys'])) {
51     $key = $_GET['keys'];
52     $w = " title like '$key%'";
53 } else {
54     $w = "";
55 }
56
57 $sql = "select * from blog where $w order by id desc limit 5 lock in share mode";
58 $query = mysqli_query($conn, $sql);
59 while ($rs = mysqli_fetch_array($query)) {
60
61 <div class="blog-post">
62 <li class="blog-title">
63 <a href="view.php?id=<?php echo $rs['id']; ?>"><?php echo $rs['title']; ?></a></li>
64
65 <li class="date">Posted <?php echo $rs['date']; ?></li>
66
67 <div class="blog-content"><?php echo $rs['content']; ?></div>
68 <a href="view.php?id=<?php echo $rs['id']; ?>" class="post-link">Read More...</a>
69
70 <?php
71 if($current_user == "admin"){ ?>
72 <a href="edit.php?id=<?php echo $rs['id']; ?>" class="post-link">edit</a>
73 <a href="del.php?id=<?php echo $rs['id']; ?>" class="post-link">delete</a> |
74
75 }
76

```

8、 完整性校验

User 表的完整性约束:

phpMyAdmin 界面显示了数据库 'test' 中的表 'users' 的结构。表结构如下：

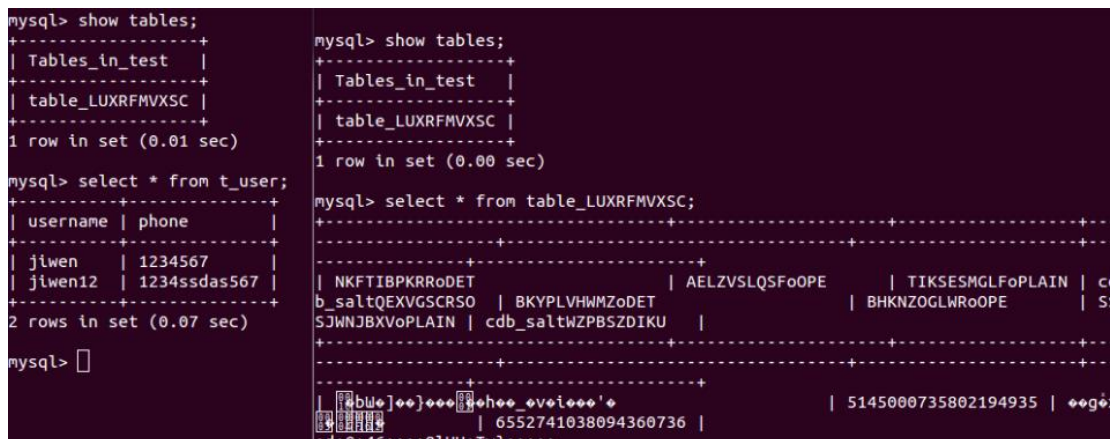
#	名字	类型	排序规则	属性	空	默认	注释	操作
1	user_id	int(11)		否	无	AUTO_INCREMENT		修改 删除 更多
2	UserName	varchar(30)	utf8_general_ci	否	无			修改 删除 更多
3	password	varchar(32)	utf8_general_ci	否	无			修改 删除 更多
4	sex	varchar(10)	utf8_general_ci	是	NULL			修改 删除 更多
5	remark	varchar(200)	utf8_general_ci	是	NULL			修改 删除 更多

在底部，可以看到 'user_id' 被设置为 PRIMARY 键，类型为 BTREE，索引为 A。

Blog 表的完整性约束:

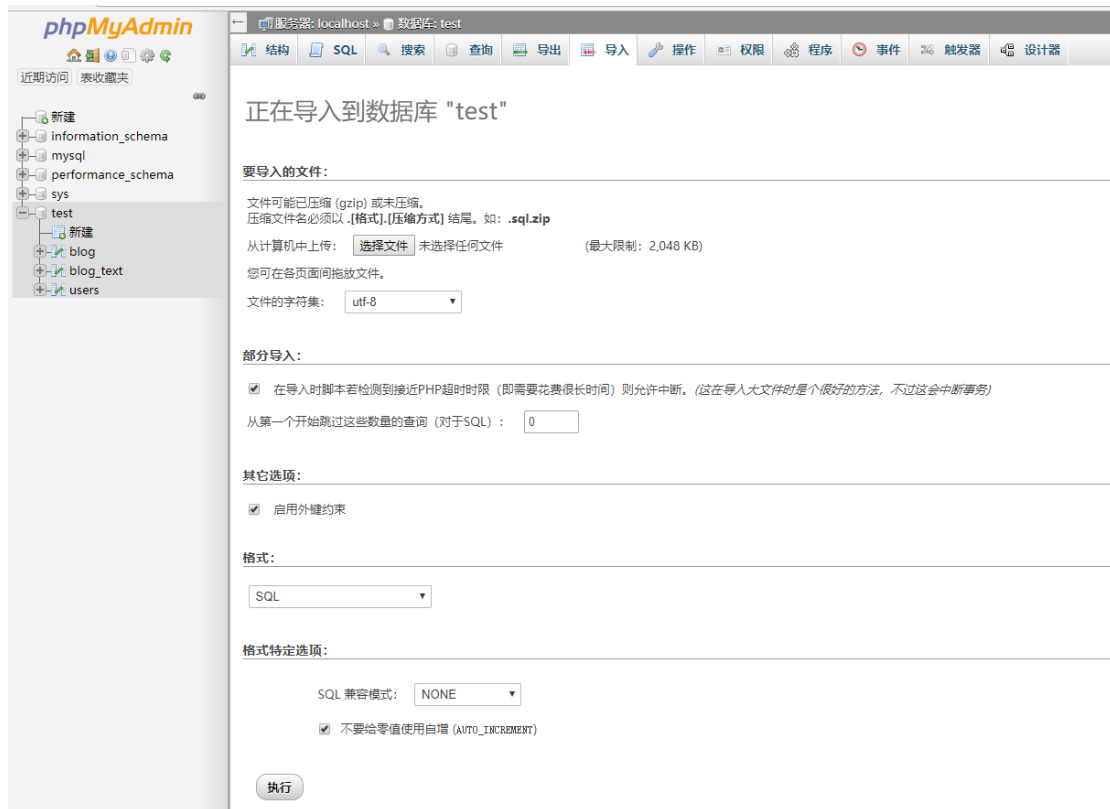


9、 数据加密 配置了 Cryptdb:



10、 数据库备份与恢复 可在 phpmyadmin 中很方便的备份、恢复。





【实验小结】

通过本次实验，对 web 应用的开发、数据库的安全操作有了更深的了解。

评语：

成绩：

签名：

日期： 年 月 日