《软件安全》实验报告

姓名：邢清画　　学号：2211999　　班级： 1023

实验名称：

Web 开发实践

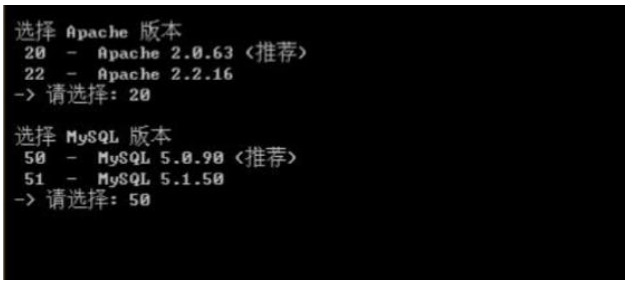
实验要求：

复现课本第十章的实验三(10.3.5 节):利用 php，编写简单的数据库插入、查询和删除操作的示例。基于课本的完整例子，进一步了解 WEB 开发的细节。

实验过程：

1. 安装 PHPnow

在安装过程中发现，在 win7,win10 版本存在管理员机制，必须使用管理员权限运行 Init.cmd,否则会运行失败,可以使用 Windows XP 系统进行本次实验,选择 PHPnow 的 Apache 和 Mysql 版本：



设置 Mysql 密码并完成安装：

打开默认界面（http://127.0.0.1/index.php）

127.0.0.1

Let's PHP now !

为何只能本地访问？
此服务器互联网 IP
111.33.78.4

Server Information	
SERVER_NAME	127.0.0.1
SERVER_ADDR:PORT	127.0.0.1:80
SERVER_SOFTWARE	Apache/2.0.63 (Win32) PHP/5.2.14
PHP_SAPI	apache2handler
php.ini	C:\PHPnow-1.5.6\php-5.2.14-Win32\php-apache2handler.ini
网站主目录	C:/PHPnow-1.5.6/htdocs
Server Date / Time	2024-05-27 12:52:13 (+08:00)
Other Links	phpinfo() | phpMyAdmin

PHP 组件支持	
Zend Optimizer	Yes / 3.3.3
MySQL 支持	Yes / client lib version 5.0.90
GD library	Yes / bundled (2.0.34 compatible)
eAccelerator	No

MySQL 连接测试			
MySQL 服务器	localhost	MySQL 数据库名	test
MySQL 用户名	root	MySQL 用户密码	
			连接

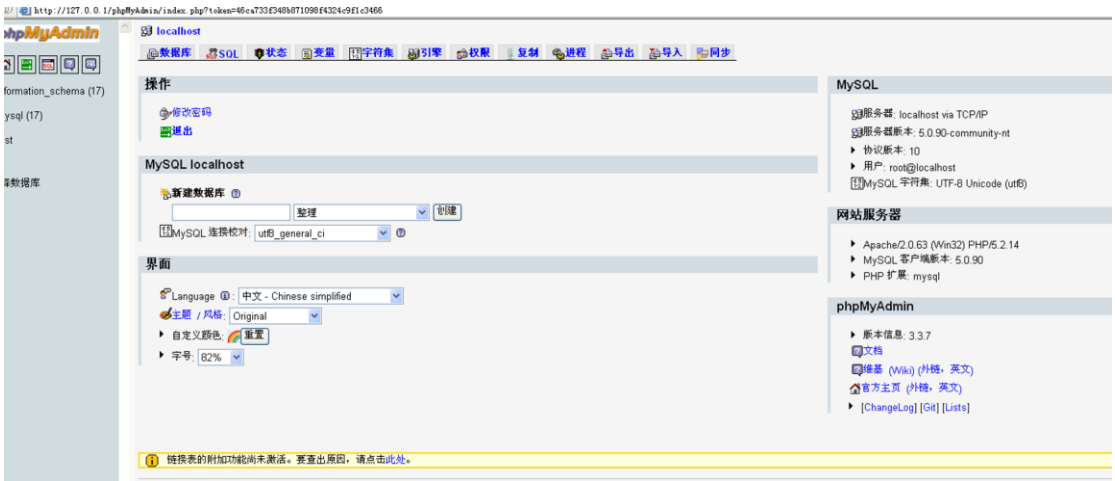
Valid XHTML 1.0 Strict / Copyleft ! 2007-? by PHPnow.org

输入密码 123456 检查数据库连接是否正常：

MySQL 连接测试			
MySQL 服务器	localhost	MySQL 数据库名	test
MySQL 用户名	root	MySQL 用户密码	
			连接

MySQL 测试结果	
服务器 localhost	OK (5.0.90-community-nt)
数据库 test	OK

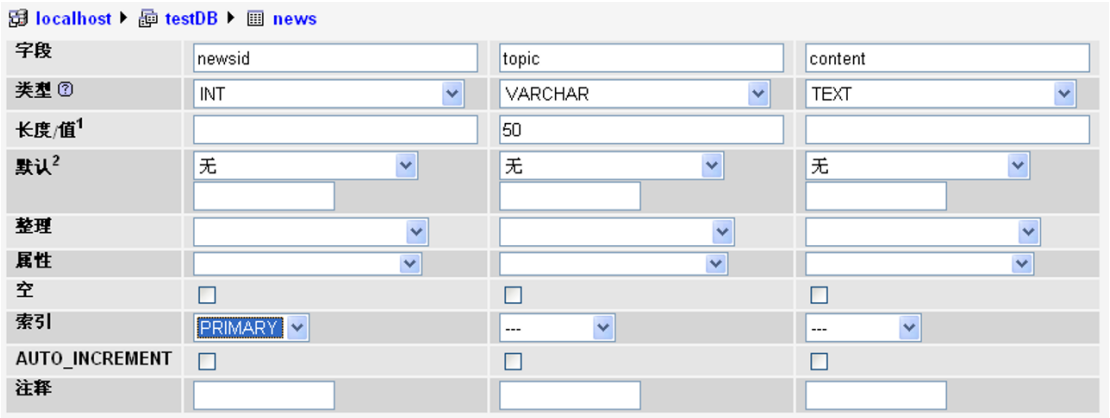
数据库成功连接之后进入数据库管理系统；



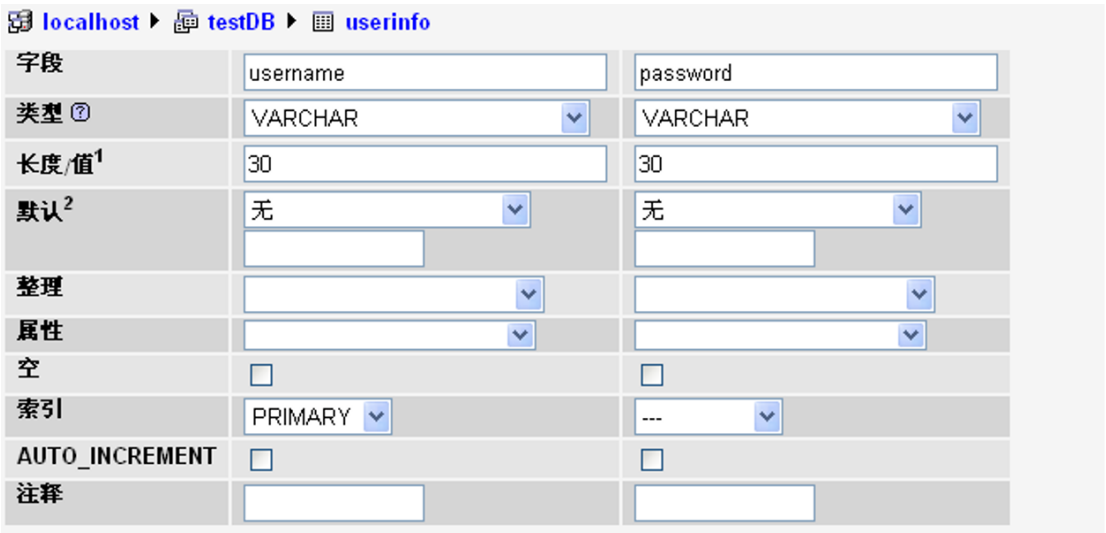
2. 利用 PHP，编写简单的数据库插入、查询和删除操作的示例复现

新建数据库 TestDB

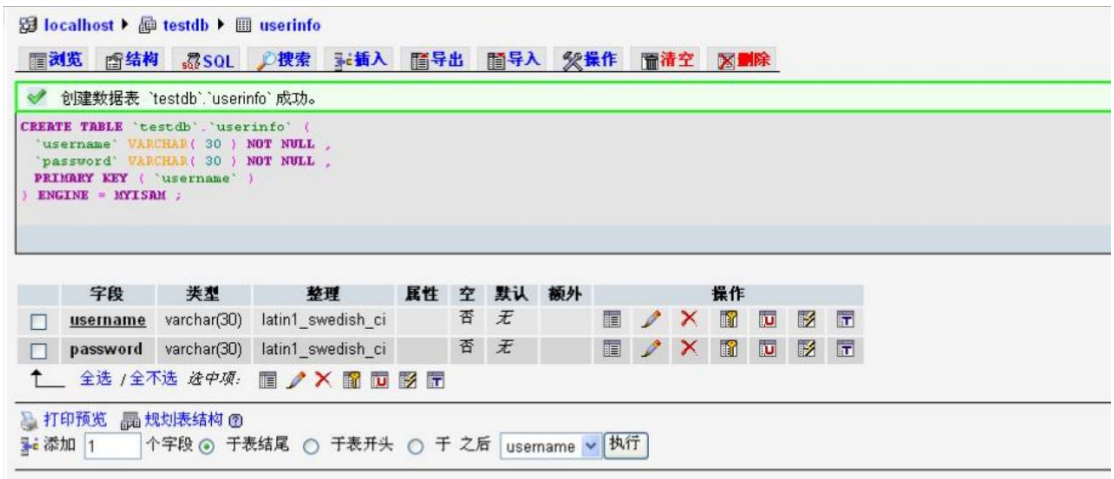
在数据库中创建数据表 News（newsid, topic, content）



userinfo（username, password）



均创建成功



向表中插入管理员数据 admin：

新建 html 文档，准备编写 html 代码，新建 html 文件，准备编写 html 和 PHP 代码。

3. **html 以及 php 代码编写：**

下面进行 html 以及 php 代码编写：

3.1 login.html：

```
<html>
<body>
<form id="form1" name="form1" method="post" action="loginok.php">
<table width="900" border="0" cellspacing="0" cellpadding="0">
<tr> <td height="20">username</td>
<td height="20"><label>
<input name="username" type="text" id="username" />
</label></td>
</tr>
<tr>
```

```
<td height="20">password</td>
<td height="20"><label>
<input name="pwd" type="password" id="pwd" />
</label></td>
</tr>
<tr>
<td height="20"> </td>
<td height="20"><label>
<input type="submit" name="Submit" value="提交" />
</label></td>
</tr>
</table>
</form>
</body>
</html>
```

表单 <form>： 这个表单的 ID 是 "form1"，并且使用 POST 方法将数据发送到 "loginok.php"。POST 方法通常用于提交需要处理的数据，比如登录信息。"loginok.php" 是一个 PHP 文件，用来处理提交的数据。

输入字段 <input>：

1. 用户名输入框：
   类型为 "text"，用于输入用户名。
   name 和 id 都是 "username"。
   用户在这里输入他们的用户名。
2. 密码输入框：
   类型为 "password"，用于输入密码。
   name 和 id 都是 "pwd"。
   用户在这里输入他们的密码，输入的内容会被隐藏以保护隐私。

提交按钮 <input type="submit">： 这是一个提交按钮。用户点击它后，表单会被提交。按钮上的文字是 "提交"。当用户点击这个按钮时，表单数据会通过 POST 方法发送到 "loginok.php" 进行处理。

3.2 Loginok.php：

```
<?php
$loginok=0;
$conn=mysql_connect("localhost", "root", "123456");
$username = $_POST['username'];
$pwd = $_POST['password'];
$SQLStr = "SELECT * FROM userinfo where username='$username' and pwd='$pwd'";
echo $SQLStr;
$result=mysql_db_query("testDB", $SQLStr, $conn);
```

```
if ($row=mysql_fetch_array($result))//通过循环读取数据内容
{
$loginok=1;
}
// 释放资源
mysql_free_result($result);
// 关闭连接
mysql_close($conn);
if ($loginok==1)
{
?>
<script>
alert("login succes");
window.location.href="sys.php";
</script>
<?php
}
else{?>
<script>
alert("login failed");
history.back();
</script>
<?php
}
?>
```

这段 PHP 代码的主要作用是处理用户登录。

1. **连接数据库**

   `$conn = mysql_connect("localhost", "root", "123456");`

   代码尝试连接到本地的 MySQL 数据库，使用用户名 "root" 和密码 "123456"。

2. **获取表单数据：**

   `$username = $_POST['username'];`

   `$pwd = $_POST['password'];`

   获取用户在 HTML 表单中提交的用户名和密码，分别存储在 $username 和 $pwd 变量中。

3. **构建 SQL 查询：**

   ```
   $SQLStr = "SELECT * FROM userinfo where username='$username' and pwd='$pwd'";
   echo $SQLStr;
   ```

   构建一个 SQL 查询字符串，用于在 userinfo 表中查找与提交的用户名和密码匹配的记录。

输出查询字符串（用于调试）。

4. **执行查询：**

   ```
   $result = mysql_db_query("testDB", $SQLStr, $conn);
   ```

   在名为 testDB 的数据库中执行查询，并将结果存储在 $result 变量中。

5. **处理查询结果：**

```
    if ($row = mysql_fetch_array($result)) {
        $loginok = 1;
    }
```

使用 mysql_fetch_array 函数通过循环读取查询结果，如果找到了匹配的记录，就将 $loginok 变量设置为 1，表示登录成功。

6. **释放资源并关闭连接：**

```
    mysql_free_result($result);
    mysql_close($conn);
```

释放数据库查询的结果资源。

关闭与数据库的连接。

7. **检查登录状态并响应：**

```
if ($loginok == 1) {
    ?>
    <script>
    alert("login success");
    window.location.href = "sys.php";
    </script>
    <?php
} else {
    ?>
    <script>
    alert("login failed");
    history.back();
    </script>
    <?php
}
?>
```

检查 $loginok 的值：

如果 $loginok 为 1，表示登录成功，弹出一个提示框显示 "login success"，然后跳转到 "sys.php" 页面。

如果 $loginok 不为 1，表示登录失败，弹出一个提示框显示 "login failed"，然后返回到上一个页面。

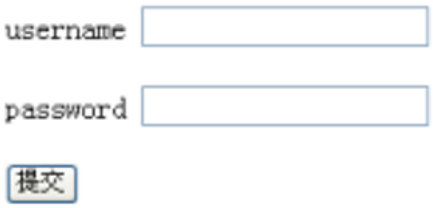
3.3 news.php

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>主页</title>
</head>
<body>
<div align="center">
<table width="900" border="0" cellspacing="0" cellpadding="0">
<tr>
<td height="40"><form id="form1" name="form1" method="post" action="loginok.php">
```

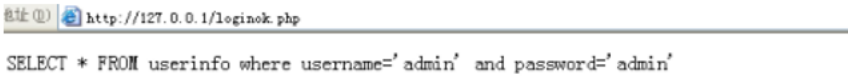
```
<div align="right">用户名：
<input name="username" type="text" id="username" size="12" />
密码：
<input name="password" type="password" id="password" size="12" />
<input type="submit" name="Submit" value="提交" />
</div>
</form>
</td>
</tr>
<tr>
<td><hr /></td>
</tr>
<tr>
<td height="300" align="center" valign="top"><p> </p>
<?php
$conn=mysql_connect("localhost", "root", "123456");
$newsid = $_GET['newsid'];$SQLStr = "select * from news where newsid=$newsid";
$result=mysql_db_query("testDB", $SQLStr, $conn);
if ($row=mysql_fetch_array($result))//通过循环读取数据内容
{
// 定位到第一条记录
mysql_data_seek($result, 0);
// 循环取出记录
while ($row=mysql_fetch_row($result))
{
echo "$row[1]<br>";
echo "$row[2]<br>";
}
}
// 释放资源
mysql_free_result($result);
// 关闭连接
mysql_close($conn);
?>
</td>
</tr>
</table>
</div>
</body>
</html>
```

通过上述代码实现了一个简单的用户登录和新闻显示系统。首先，通过 HTML 表单收集用户名和密码，并将数据提交给 PHP 脚本 ″loginok.php″ 进行验证。如果验证通过，用户会看到登录成功的提示并跳转到 ″sys.php″ 页面；

如果验证失败，会提示登录失败并返回登录页面。此外，页面还包含一个 PHP 脚本，通过新闻 ID 从数据库中检索并显示相应的新闻内容。



输入 username：admin，password：admin



显示登录成功

4. 用 php 编写简单的数据库插入、查询和删除操作的示例

4.1 编写 sys.php：

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>主页</title>
</head>
<?php
$conn=mysql_connect("localhost", "root", "123456");
?>
<body>
<div align="center">
<table width="900" border="0" cellspacing="0" cellpadding="0">
<tr>
<td height="40"><form id="form1" name="form1" method="post" action="add.php">
<div align="right">新闻标题：
<input name="topic" type="text" id="topic" size="50" />
<BR>
新闻内容：
<textarea name="content" cols="60" rows="8" id="content"></textarea><BR>
```

```
<input type="submit" name="Submit" value="添加" />
</div>
</form>
</td>
</tr>
<tr>
<td><hr /></td>
</tr>
<tr>
<td height="300" align="center" valign="top"><table width="600" border="0" cellspacing="0"
cellpadding="0">
<tr>
<td width="100" height="30"><div align="center">新闻序号</div></td>
<td><div align="center">新闻标题</div></td>
<td><div align="center">删除</div></td>
</tr><?php
$SQLStr = "select * from news";
$result=mysql_db_query("testDB", $SQLStr, $conn);
if ($row=mysql_fetch_array($result))//通过循环读取数据内容
{
// 定位到第一条记录
mysql_data_seek($result, 0);
// 循环取出记录
while ($row=mysql_fetch_row($result))
{
?>
<tr>
<td height="30"><div align="center"> <?php echo $row[0] ?> </div></td>
<td width="400"> <div align="center"> <?php echo $row[1] ?>
</div></td>
<td><div align="center"><a href="del.php?newsid=<?php echo $row[0] ?> " >
删 除 </a>
</div></td>
</tr>
<?php
}
}
?>
</table></td>
</tr>
</table>
</div>
</body>
</html>
```

```
<?php
// 释放资源
mysql_free_result($result);
// 关闭连接
mysql_close($conn);
?>
```

这段代码实现了一个简单的新闻管理系统，用户可以通过网页表单添加新闻标题和内容，这些新闻将存储在数据库中并显示在页面上，同时提供删除新闻的功能。PHP 脚本连接数据库读取新闻记录并在页面上显示，同时提供删除链接以便用户删除特定新闻条目。

代码的主要作用

1. 连接数据库：

代码连接到本地的 MySQL 数据库，使用指定的用户名和密码。

2. 添加新闻功能：

在页面的主体部分，创建了一个用于添加新闻的表单。表单包含两个输入字段：新闻标题和新闻内容，以及一个提交按钮。当用户填写新闻标题和内容后，点击“添加”按钮，将通过 POST 方法将数据发送到 "add.php" 文件进行处理。

3. 显示新闻列表：

页面上有一个表格用于显示新闻列表。这个列表从数据库的 "news" 表中获取所有新闻记录，并将每一条新闻的序号、标题和一个删除链接显示出来。每个新闻的删除链接会指向 "del.php" 文件，并带上新闻的 id 作为参数。

4. 删除新闻功能：

用户可以通过点击每条新闻对应的“删除”链接，删除特定的新闻条目。删除操作由 "del.php" 文件处理，利用新闻 ID 执行删除操作。

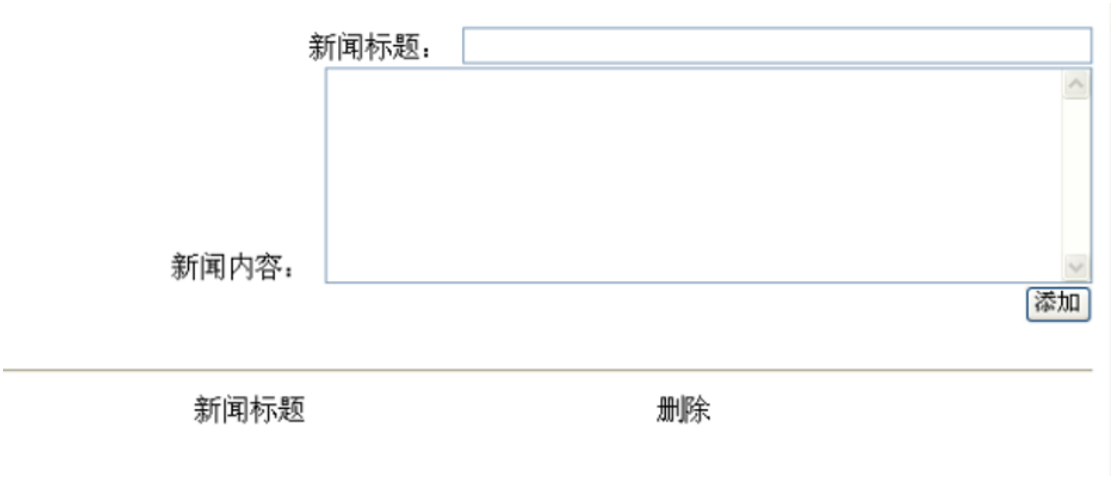
4.2 Add.php：

```
<?php
$conn=mysql_connect("localhost", "root", "123456");
mysql_select_db("testDB");
$topic = $_POST['topic'];
$content = $_POST['content'];
$SQLStr = "insert into news(topic, content) values('$topic', '$content')";
echo $SQLStr;
$result=mysql_query($SQLStr);
// 关闭连接
mysql_close($conn);
if ($result)
{
?>
<script>
alert("insert succes");
window.location.href="sys.php";
</script>
<?php
}
```

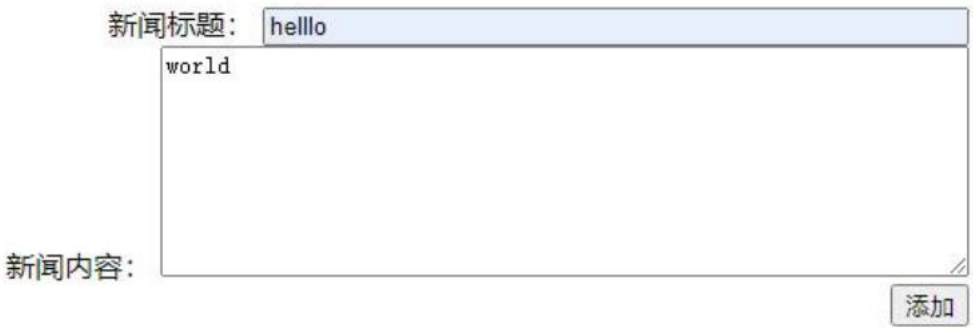
```
else{
?>
<script>
alert("insert failed");
history.back();
</script>
<?php
}
?>
```

这段代码实现了从表单获取用户输入的新闻标题和内容，并将其插入到数据库中。插入操作完成后，根据结果显示相应的提示信息，成功时跳转到新闻管理页面，失败时返回到输入页面。PHP 脚本负责处理数据库连接、数据插入操作及结果反馈。

执行结果如下：



插入新闻标题为 hello，新闻内容为 world，插入成功



4.3 Del.php：

```
<?php
$conn=mysql_connect("localhost", "root", "123456");
mysql_select_db("testDB");
```

```
$newsid = $_GET['newsid'];
$SQLStr = "delete from news where newsid=$newsid";echo $SQLStr;
$result=mysql_query($SQLStr);
// 关闭连接
mysql_close($conn);
if ($result)
{
?>
<script>
alert("delete succes");
window.location.href="sys.php";
</script>
<?php
}
else{
?>
<script>
alert("delete failed");
history.back();
</script>
<?php
}
?>
```

这段代码实现了从 GET 请求中获取新闻 ID 并删除对应的新闻记录。删除操作完成后，根据结果显示相应的提示信息，成功时跳转到新闻管理页面，失败时返回到之前的页面。PHP 脚本负责处理数据库连接、数据删除操作及结果反馈。

1. 连接数据库：

代码连接到本地的 MySQL 数据库，并选择使用 testDB 数据库。

2.获取新闻 ID：

从 GET 请求中获取用户要删除的新闻 ID (newsid)。

3.构造并执行 SQL 语句：

构造 SQL 删除语句，删除 news 表中与获取的新闻 ID 对应的记录。执行删除操作，并输出 SQL 语句以便调试。

4.关闭数据库连接：

删除操作完成后，关闭数据库连接以释放资源。

5.结果反馈：

根据删除操作的结果，使用 JavaScript 弹出相应的提示信息。如果删除成功，弹出“删除成功”的提示并跳转到 sys.php 页面。如果删除失败，弹出“删除失败”的提示并返回到上一页面。

删除对比：

心得体会：

本次 Web 开发实验让我深入理解了 PHP 语言及其与 HTML、SQL 数据库的交互机制。通过本次实验，我更好地理解了前端与后端的交互过程。我学会了如何在前端获取用户的输入，然后通过 POST 或 GET 方法发送到后端。在后端，我学会了如何接收这些输入，处理它们，并将结果发送回前端。

在这次实验中，我通过实现一个简单的用户登录和新闻管理系统，深入理解了 Web 开发中的基本流程和技术要点。通过编写 HTML 表单和 PHP 脚本，我学会了如何进行用户数据的收集与处理，如何连接和操作 MySQL 数据库，以及如何在前后端之间传递信息。特别是在处理数据的增删改查操作时，我认识到 SQL 注入的风险及其防范的重要性。

虽然在这个过程中遇到了一些问题，但通过寻找资料和试验，我最终都成功解决了这些问题。这使我意识到，开发过程中不仅需要扎实的编程技能，还需要不断学习和适应新问题的能力。与此同时，我也更加关注在开发过程中必须关注的安全性和用户体验问题。

通过本次实验，我不仅巩固了我的编程技能，还提高了对 Web 开发中前后端交互的理解。这次实验让我意识到，在实际开发中，安全性和用户体验同样重要，必须时刻保持对这些问题的警惕和关注。

总的来说，这次实验对我来说是一次非常宝贵的学习经历，不仅提高了我的技术水平，还增强了我在实际开发中解决问题的能力。我期待在未来的学习和工作中，能将这些经验和技能应用到更复杂的项目中去。