《软件安全》实验报告

姓名：李雅帆　　学号：2213041　　班级：信安班

一、实验名称：

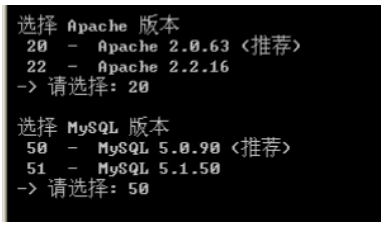
Web 开发实践

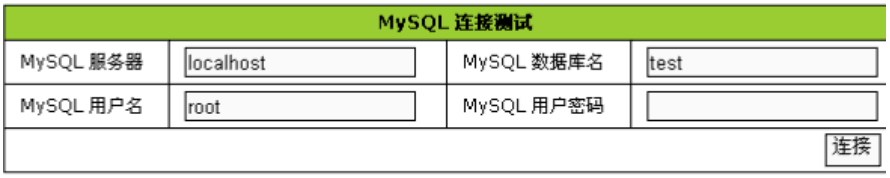
二、实验要求：

复现课本第十章实验三：利用 PHP，编写简单的数据库插入、查询和删除操作示例。基于课本的完整的例子，进一步了解 Web 开发细节。

三、实验过程：

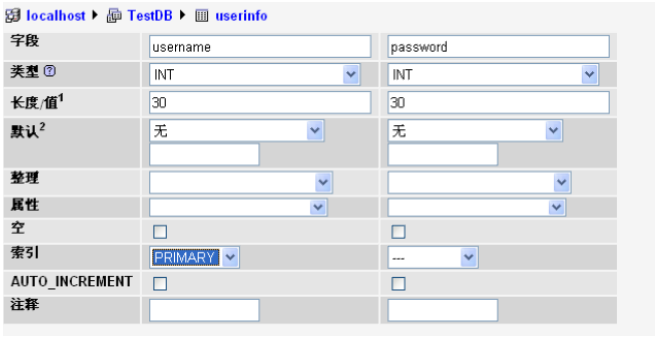
1.在虚拟机 xp 系统安装 PHP，选择版本并设置 MySQL 的密码，并打开默认界面并连接数据库。



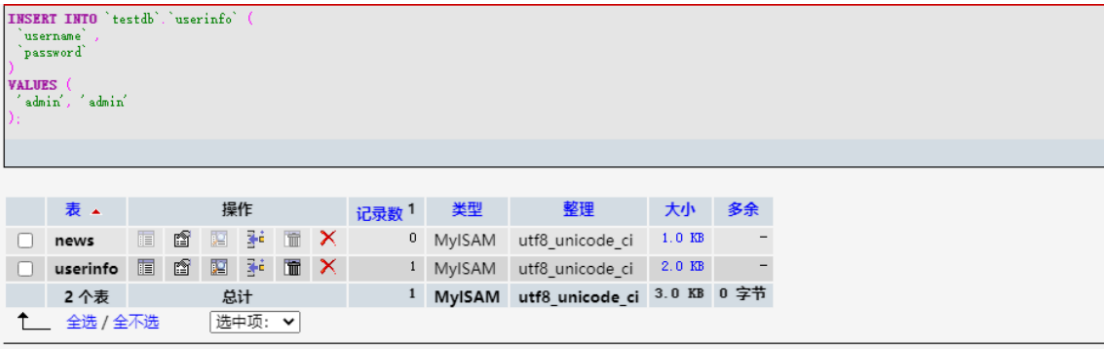


2.实现实验三，利用 PHP，编写简单的数据库插入、查询和删除操作。

新建数据库，以及 news(newsid,topic,content)和 userinfo(username,password)两张表。



向表中插入数据



打开 dreamwaver，创建网页文件

3.编写 web 程序

（1）web.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>
```

这段代码创建了一个简单的 HTML 登录表单，用户可以在表单中输入用户名和密码。表单由一个表格布局，其中包含两行输入字段：一行用于用户名，另一行用于密码。在表格的最后一行，有一个提交按钮，用户点击该按钮后，表单数据（即用户名和密码）将通过 HTTP POST 方法发送到服务器上的 loginok.php 文件进行处理。在提交过程中，表单数据会按照 name 属性的值进行标识，即用户名对应 username，密码对应 pwd。为了保证数据的安全性，最佳实践是通过 HTTPS 协议来传输表单数据。

（2）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">姓名</td>
<td height="20"><label>
<input name="username" type="text" id="username" />
</label></td>
</tr>
<tr>
<td height="20">口令</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>
```

这段 HTML 代码实现了一个简单的登录表单页面。页面包含一个表单，表单内部使用表格布局了两个输入字段：一个用于输入姓名（通常应该是用户名），另一个用于输入密码（以密码形式隐藏）。用户可以在对应的输入框中输入信息，并在完成输入后，点击提交按钮。点击提交按钮后，表单数据（即姓名和密码）将通过 HTTP POST 方法发送到 loginok.php 文件进行处理。其中，姓名和密码分别通过 name 属性标识为 username 和 pwd。整个表单的 HTML 结构被包含在<html>和</html>标签之间，并位于<body>标签内。

此时的运行效果为：

当得到用户名和密码之后，将 PHP 脚本连接数据库，如果输入的的用户名和密码对的话，显示 OK，不对的话报错。

完整的 PHP 脚本代码为：

```php
<?php
$username=$_POST['username'];
$password=$_POST['password'];
$SQLStr="select * from userinfo where username='$username' and pwd='$password'";

echo $SQLStr;

?>
```

然后对其进行校验，输入正确的用户名和密码显示 OK

接下来要将其与数据库进行连接，连接数据库的代码为：

```php
$conn=mysql_connect("localhost", "root", "123456");  //连接数据库
$username = $_POST['username'];
$pwd = $_POST['password'];
$SQLStr = "SELECT * FROM userinfo where username='$username' and pwd='$pwd'";
echo $SQLStr ;
$result=mysql_db_query("testDB", $SQLStr, $conn); //执行数据库SQL语句
// 获取查询结果
if ($row=mysql_fetch_array($result))//通过循环读取数据内容
     echo "<br>OK<br>";
 else
     echo "<br>false<br>";
   // 释放资源
   mysql_free_result($result);
   // 关闭连接
   mysql_close($conn);
```

运行 MySQL 语句，此时输入不匹配的用户名和密码，打印结果为 false

```
SELECT * FROM userinfo where username='111111112222' and pwd='111'
false
```

输入正确的，打印结果为

```
SELECT * FROM userinfo where username='admin' and pwd='admin'
OK
```

此时，修改代码，我希望如果登陆成功，就转到一个管理界面，如果登陆失败了，就返回重新输入。

```
if ($isOK==1)
{
    ?>
    <script language="javascript">
    alert("ok");
    window.location.href="sys.php";
    </script>
    <?php
}else{
    ?>
    <script language="javascript">
    alert("false");
    history.back();
    </script>
    <?php
}
?>
```

此时输入正确的用户名和密码，会弹出窗口显示 OK，输入错误的用户名和密码，则会返回登陆界面，允许用户重新输入。

(3)接下来转到 sys.php 实现新闻内容管理，实现添加。

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);
```

```
?>
```

这段代码是一个简单的 HTML 页面，结合了 PHP 来处理与 MySQL 数据库的交互。首先，HTML 部分定义了页面的基本结构和样式，包括一个表单用于输入新闻标题和内容，以及一个表格用于展示已存在的新闻列表。在 PHP 部分，代码首先尝试连接到本地 MySQL 数据库，使用用户名"root"和密码"123456"。然后，它定义了一个 SQL 查询语句 $SQLStr，用于从名为"news"的表中选取所有记录。这个查询在连接到数据库后被执行，结果集存储在$result 中。接下来，代码通过 mysql_fetch_array 尝试获取第一行数据，但这里有一个逻辑错误，因为紧接着代码重置了结果集的指针到第一行（mysql_data_seek($result, 0)），并随后使用 mysql_fetch_row 来循环遍历结果集中的每一行。对于每一行数据，它都被用于填充表格中的一行，显示新闻的序号、标题，并提供一个链接用于删除该新闻。在 HTML 表格下方，PHP 代码释放了结果集资源并关闭了数据库连接。

实现插入的代码 add.php 为

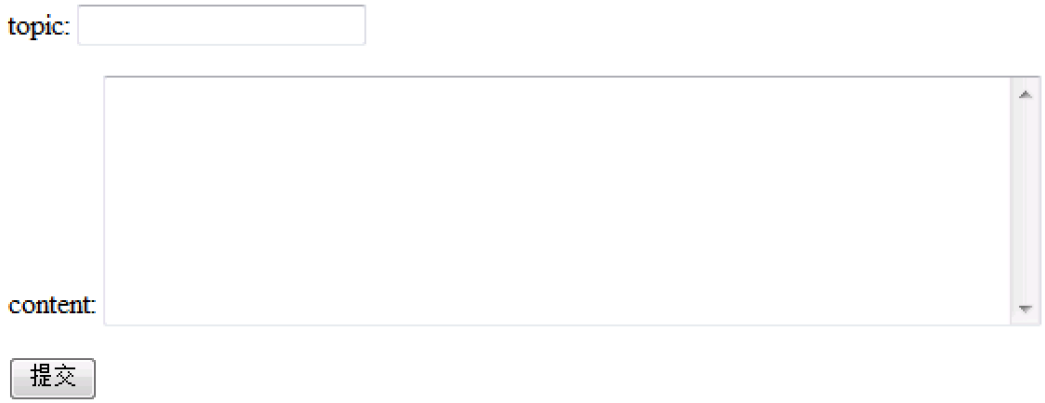
```
<?php
        $conn=mysql_connect("localhost", "root", "123456");
        mysql_select_db("testDB");
        $topic = $_POST['topic'];
        $content = $_POST['content'];
        $SQLStr = "insert into news(topic, content) values('$topic', '$content')";
        echo $SQLStr;
        $result=mysql_query($SQLStr);

        // 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
        }

    ?>
```

PHP 首先尝试使用提供的凭据（"root"用户名和"123456"密码）连接到本地 MySQL 服务器，并选择名为"testDB"的数据库。接着，PHP 从表单的 POST 数据中获取用户输入的新闻标题（$_POST['topic']）和内容（$_POST['content']）。然后，PHP 构造一个 SQL 插入语句（$SQLStr），将用户输入的标题和内容插入到"news"表的相应字段中。为了调试或日志记录，PHP 将构造的 SQL 语句输出到屏幕上（echo $SQLStr;）。随后，PHP 执行这个 SQL 插入语句（mysql_query($SQLStr)），并将结果存储在$result 变量中。在执行插入操作后，PHP 关闭与 MySQL 数据库的连接（mysql_close($conn)）。PHP 检查$result 变量以确定插入操作是否成功。如果$result 为真（即插入成功），则输出一段 JavaScript 代码，该代码会在用户浏览器中弹出一个警告框显示"insert succes"，并将用户重定向到"sys.php"页面。如果插入操作失败（$result 为假），则输出另一段 JavaScript 代码，该代码会弹出一个警告框显示"insert failed"，并将用户导航回上一个页面（通过 history.back()方法）。

运行代码，登陆后进入到系统界面



此时，输入内容，点击提交，显示：

```
insert into news(topic, content) values('1','11111')
```

此时已经添加成功，进入数据库中，可以看见添加的字段。

(4)实现显示新闻内容的功能。

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>
```

首先，页面加载时，在 HTML 的<body>部分中，用户会看到一个居中的表格。表格的第一行包含一个表单，用于用户输入用户名和密码，并有一个提交按钮。当用户填写用户名和密码并点击提交按钮时，表单数据会被 POST 到 loginok.php 进行处理（虽然这部分逻辑在提供的代码中并未展示）。表格的第二行是一个水平线，用于分隔表单和下面的内容。表格的第三行用于展示新闻内容。在这里，PHP 代码首先尝试使用 mysql_connect 函数连接到本地的 MySQL 数据库，使用用户名"root"和密码"123456"。然后，它从 URL 的 GET 参数中获取 newsid 的值，并构造一个 SQL 查询语句来从"news"表中选取与给定 newsid 对应的新闻记录。如果查询成功，并且返回了结果集，PHP 代码会首先定位到结果集的第一条记录（尽管这一步是多余的，因为紧接着的 while 循环会从头开始遍历），然后通过一个 while 循环逐条取出新闻记录，并输出新闻的标题和内容（假设新闻表的第二列是标题，第三列是内容）。在输出完所有新闻记录后，PHP 代码会释放结果集占用的资源，并关闭与数据库的连接。

运行，此时可以显示添加的新闻信息。

在其中加入链接，也可以实现查找之后单击连接，进入新闻页面。

id	topic
1	1
2	22
3	444

(5)实现删除新闻的功能

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
        }
    ?>
```

当该 PHP 脚本被执行时，它首先尝试使用提供的凭据（用户名"root"和密码"123456"）连接到本地 MySQL 服务器，并选择"testDB"数据库。接着，它从 URL 的 GET 参数中捕获 newsid 的值，该值代表要删除的新闻记录的 ID。然后，脚本构造了一个 SQL 删除语句，该语句将删除"news"表中 newsid 字段值等于捕获到的 newsid 的新闻记录。这个 SQL 语句被存储在变量$SQLStr 中，并且为了调试或日志记录的目的，该语句被输出到屏幕上。随后，脚本尝试执行这个 SQL 删除语句，并将结果存储在$result 变量中。如果删除操作成功（即$result 为真），则输出一段 JavaScript 代码，该代码会在用户浏览器中弹出

一个警告框显示"delete succes"，并将用户重定向到"sys.php"页面。如果删除操作失败（即$result 为假），则输出另一段 JavaScript 代码，该代码会弹出一个警告框显示"delete failed"，并使用浏览器的历史记录将用户导航回上一个页面。最后，无论删除操作是否成功，脚本都会关闭与 MySQL 数据库的连接。

运行代码，删除某条新闻，提示删除成功。

```
delete from news where newsid=1
```

查看数据库

删除后的数据库

表面删除成功。

(6)最后编写允许用户产看新闻和进行登陆界面。

index.php：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>主页</title>
</head>
<?php
  $conn=mysql_connect("localhost", "root", "NANkai@101");
?>
<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"><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>
            </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> <div align="center"> <a href="news.php?newsid=<?php echo $row[0] ?> " >
<?php echo $row[1] ?>     </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);
?>
```

当网页被加载时，PHP 代码首先尝试使用提供的凭据（用户名"root"和密码"NANkai@101"）连接到本地的 MySQL 数据库。如果连接成功，页面继续加载 HTML 内容。在 HTML 内容中，页面中央有一个表格，表格的第一行包含一个用户登录表单，用户可以在此输入用户名和密码，并通过提交按钮将信息发送到"loginok.php"进行处理（注意，具体的登录逻辑和验证在"loginok.php"中处理，这段代码中没有给出）。表格的第二行是一个水平线，用于分隔登录表单和下面的内容。表格的第三行包含一个嵌套的表格，用于展示新闻列表。在 PHP 部分，代码执行了一个 SQL 查询，从"testDB"数据库的"news"表中选取所有新闻记录。如果查询成功并返回了结果集，代码将定位到结果集的第一条记录，并通过一个循环逐条取出新闻记录。对于每一条新闻记录，代码会输出一个表格行，其中包含新闻序号和新闻标题。新闻序号直接来自结果集中的第一列（*row*[0]），而新闻标题则来自第二列（row[1]）。新闻标题被包装在一个链接中，该链接指向"news.php"页面，并附带一个名为"newsid"的 GET 参数，其值为新闻序号。这样，当用户点击新闻标题时，他们将被导航到显示具体新闻内容的页面，并通过"newsid"参数指定要查看的新闻。

运行程序，最终效果为：

此时支持用户查看新闻具体界面。

四、心得体会：

在这次 Web 开发实践实验中，我体会到了 PHP 与 MySQL 结合使用的强大之处。通过复现课本第十章实验三，我学到了如何编写简单的数据库插入、查询和删除操作，并且更加深入地了解了 Web 开发的细节。

首先，我在虚拟机 XP 系统中安装了 PHP。在选择版本并设置 MySQL 的密码后，我成功地打开了默认界面并连接上了数据库。 接着，我按照课本中的例子，编写了数据库插入、查询和删除的 PHP 脚本。

通过这次实验，我不仅加深了对 PHP 和 MySQL 的理解，还掌握了一些实用的 Web 开发技巧。学会了如何进行基本的数据库操作，进一步了解了 Web 开发的流程和细节。