第 15 章

PHP 操作 MySQL 数据库

学习指引

Web 开发是离不开数据库操作的，当然不止 Web 开发会用到数据库，任何一种编程语言都需要对数据库进行操作，PHP 也不例外。现在流行的数据库有很多，如 Oracle、SQL Server、MySQL 等。由于 MySQL 是开元的、跨平台的，使用方便，从而获得了广泛应用。本章就来介绍一下 PHP 如何操作 MySQL 数据库。

重点导读

- 熟悉 PHP 访问 MySQL 数据库的一般步骤。
- 掌握 PHP 操作 MySQL 数据库的方法。
- 掌握 PHP 操作 MySQL 数据库。

15.1 PHP 访问 MySQL 数据库的一般步骤

MySQL 是一个开源的，市场的占有率比较高，所以一直以来都被认为是 PHP 的最佳搭档。同时 PHP 也具有很强大的数据库支持能力，本节主要介绍 PHP 访问 MySQL 数据库的一般步骤。

PHP 访问 MySQL 数据库的一般步骤如图 15-1 所示。

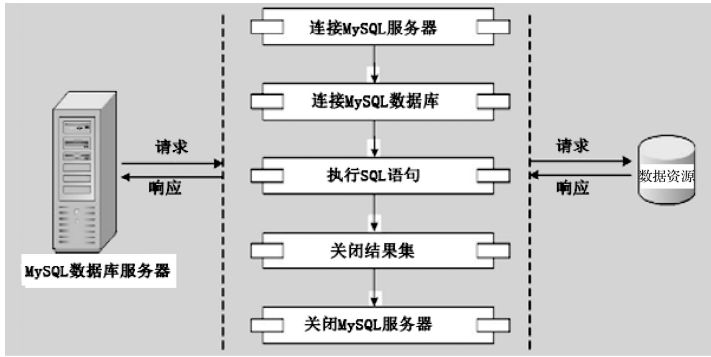


图 15-1 PHP 访问 MySQL 数据库的一般步骤

1. 连接 MySQL 数据库

使用 mysqli_connect()函数与 MySQL 服务器的建立连接。有关 mysqli_connect()函数的使用请参考 15.2.1 节。

2. 选择 MySQL 数据库

使用 mysqli_select_db()函数选择 MySQL 数据库服务器的数据库，并与数据库建立连接，有关 mysqli_select_db()函数的使用请参考 15.2.2 节。

3. 执行 SQL 语句

在选择数据库中使用 mysqli_query()函数执行 SQL 语句，对数据的操作方式主要包括 5 种方式，下面分别进行介绍。

（1）查询数据：使用 select 语句实现数据的查询功能。

（2）显示数据：使用 select 语句显示数据的查询结果。

（3）插入数据：使用 insert into 语句向数据库中插入数据。

（4）更新数据：使用 update 语句更新数据库中的记录。

（5）删除数据：使用 delete 语句删除数据库中的记录。

mysqli_query()函数的具体使用请参考 15.2.3 节。

4. 关闭结果集

数据库操作完成后，需要关闭结果集，以释放系统资源，语法格式如下：

```
mysqli_free_result($result);
```

5. 关闭 MySQL 服务器

每使用一次 mysqli_connect()或者 mysqli_query()函数，都会消耗系统资源，少量用户访问 Web 网站时问题还不大，但如果用户连接超过一定数量时，就会造成系统性能下降，甚至是死机。为了避免这种现象的发生，在完成数据库的操作后，应该使用 mysql_close()函数关闭与 MySQL 服务器的连接，以节省系统资源。

语法格式如下：

```
mysqli_close($con);
```

15.2　PHP 操作 MySQL 数据库的方法

PHP 提供了大量的 MySQL 数据库函数，方便对 MySQL 数据库进行操作，使 Web 程序的开发更加方便灵活。

15.2.1　使用 mysqli_connect()函数连接 MySQL 服务器

要操作 MySQL 数据库，首先要先与 MySQL 数据库建立连接，连接使用 mysqli_connect()函数来完成。该函数语法格式如下：

```
mysqli_connect(host,username,password);
```

mysqli_connect()函数的参数说明如表 15-1 所示。

表 15-1　mysqli_connect()函数的参数说明

参　数	说　明
host	MySQL 服务器的主机名或 IP 地址
username	登录 MySQL 数据库服务器的用户名
password	MySQL 服务器的用户密码

【例 15-1】（实例文件：ch15\Chap15.1.php）连接 MySQL 服务器。

```
<?php
$connect=mysqli_connect("localhost","root","123456");
//判断连接是否成功
    if($connect){
        echo "服务器连接成功";
    }else{
        echo "服务器连接失败";
    }
?>
```

在 IE 浏览器中运行结果如图 15-2 所示。

图 15-2　连接 MySQL 服务器

15.2.2　使用 mysqli_select_db()函数选择数据库

在连接到 MySQL 数据库之后，可以使用 mysqli_select_db()函数选择数据库。该函数的语法格式如下：

```
mysqli_select_db(connection,dbname);
```

mysqli_select_db()函数的参数说明如表 15-2 所示。

表 15-2　mysqli_select_db()函数的参数说明

参　数	说　明
connection	必须参数，规定要使用的 MySQL 连接
dbname	必须参数，规定要使用的数据库

【例 15-2】（实例文件：ch15\Chap15.2.php）选择数据库。

```
<?php
    $connect=mysqli_connect("localhost","root","123456");      //连接服务器
    $connect1=mysqli_select_db($connect,"user_database");      //选择 user_database 数据库
    //判断 user_database 数据库是否连接成功
    if($connect1){
        echo "数据库连接成功";
    }else{
        echo "数据库连接失败";
    }
?>
```

在 IE 浏览器中运行结果如图 15-3 所示。

图 15-3　选择数据库

15.2.3　使用 mysqli_query()函数执行 SQL 语句

要对数据库中的表进行操作，通常使用 mysqli_query()函数执行 SQL 语句。该函数语法格式如下：

```
mysqli_query(connection,query,resultmode);
```

mysqli_query()函数的参数说明如表 15-3 所示。

表 15-3　mysqli_query()函数的参数说明

参　数	说　明
connection	必须参数，规定要使用的 MySQL 连接
query	必须参数，规定查询字符串
resultmode	可选参数，一个常量。可以是下列值中的任意一个： MYSQLI_USE_RESULT：需要检索大量数据时使用 MYSQLI_STORE_RESULT：默认

例如，下面以管理员信息表 u_admin 为例，举例说明常见的 SQL 语句用法。

```
mysqli_query($connect,"update u_admin set name='李四',pwd='456' where id=1");  //修改数据库记录
mysqli_query($connect,"delete from u_admin where name='朱八'");                 //删除数据库记录
SELECT * FROM 'u_admin' WHERE id<5;                                             //查询数据库记录
mysqli_query($connect,"insert into u_admin(name, pwd, email) value('三毛','789','sanmao@qq.
com')");                                                                        //添加记录
```

15.2.4　使用 mysqli_fetch_array()函数从数组结果集中获取信息

在 15.2.3 节中，介绍了使用 mysqli_query()函数执行 SQL 语句，接下来使用 mysqli_fetch_array()函数从结果集中获取信息。该函数的语法格式如下：

```
mysqli_fetch_array(result,resulttype);
```

mysqli_fetch_array()函数的参数说明如表 15-4 所示。

表 15-4　mysqli_fetch_array()函数的参数说明

参　数	说　明
result	必须参数，规定由 mysqli_query()、mysqli_store_result()或 mysqli_use_result()返回的结果集标识符
resulttype	可选参数，规定应该产生哪种类型的数组。可以是以下值中的一个： • MYSQLI_ASSOC：关联数组； • MYSQLI_NUM：数字数组； • MYSQLI_BOTH：默认值，同时产生关联和数字数组

【例 15-3】（实例文件：ch15\Chap15.3.php）mysqli_fetch_array()函数。

```
<?php
$connect=mysqli_connect("localhost","root","123456","user_database");
if(!$connect) {
    die('连接失败: ' . mysqli_error($connect));
}
//设置编码，防止中文乱码
mysqli_set_charset($connect,"utf8");
$sql="SELECT * FROM u_admin";
$data= mysqli_query( $connect, $sql );
if(!$data){
    die('无法读取数据:'. mysqli_error($connect));
```

```
    }
    echo '<h2>管理人员信息表<h2>';
    echo '<table border="1"><tr><td>ID</td><td>姓名</td><td>
密码</td><td>邮箱</td></tr>';
    //使用 while 循环语句以表格的形式输出数组结果集$output 中的数据
    while($output=mysqli_fetch_array($data, MYSQLI_ASSOC)){
       echo "<tr>
              <td> {$output['id']}</td>".
              "<td>{$output['name']}</td>".
              "<td>{$output['pwd']}</td>".
              "<td>{$output['email']}</td>".
          "</tr>";
    }
    echo '</table>';
    mysqli_close($connect);          //关闭数据库连接
    ?>
```

在 IE 浏览器中运行结果如图 15-4 所示。

图 15-4　mysqli_fetch_array()函数

15.2.5　使用 mysqli_fetch_object()函数从结果集中获取一行作为对象

使用 mysqli_fetch_object()函数同样可以获取查询结果中的数据。该函数的语法格式如下：

```
mysqli_fetch_object(result,classname,params);
```

mysqli_fetch_object()函数的参数说明如表 15-5 所示。

表 15-5　mysqli_fetch_object()函数的参数说明

参　数	说　明
result	必须参数。规定由 mysqli_query()返回的结果集标识符
classname	可选参数。规定要实例化的类名称，设置属性并返回
params	可选参数。规定一个传给 classname 对象构造器的参数数组

【例 15-4】（实例文件：ch15\Chap15.4.php）mysqli_fetch_object()函数。

```
<?php
$connect=mysqli_connect("localhost","root","123456","user_database");
if(!$connect) {
    die('连接失败: ' . mysqli_error($connect));
}
//设置编码，防止中文乱码
mysqli_set_charset($connect,"utf8");
$sql="SELECT * FROM u_admin";
$data= mysqli_query( $connect, $sql );
if(!$data){
    die('无法读取数据:'. mysqli_error($connect));
}
echo '<h2>管理人员信息表<h2>';
echo '<table border="1"><tr><td>ID</td><td>姓名</td><td>密码</td><td>邮箱</td></tr>';
//使用 while 循环语句以“结果集->列名”的方式输出结果集$output 中的管理人员信息
while($output=mysqli_fetch_object($data)){
    echo "<tr>
           <td> {$output->id}</td>".
           "<td>{$output->name}</td>".
           "<td>{$output->pwd}</td>".
```

```
            "<td>{$output->email}</td>".
        "</tr>";
}
echo '</table>';
mysqli_close($connect);          //关闭数据库连接
?>
```

在 IE 浏览器中运行结果如图 15-5 所示。

图 15-5　mysqli_fetch_object()函数

15.2.6　使用 mysqli_fetch_row()函数逐行获取结果集中的每条记录

除了前面介绍的 mysqli_fetch_array()函数和 mysqli_fetch_object()函数可以从结果集中获取数据外，还可以使用 mysqli_fetch_row()函数来获取数据。使用 mysqli_fetch_row()函数逐行获取结果集中的每条记录。mysqli_fetch_row()函数的语法格式如下：

```
mysqli_fetch_row($result);
```

其中，$result 是必须参数，规定由 mysqli_query()、mysqli_store_result()或 mysqli_use_result()返回的结果集标识符。

【例 15-5】（实例文件：ch15\Chap15.5.php）mysqli_fetch_arow()函数。

```
<?php
$connect=mysqli_connect("localhost","root","123456","user_database");
if(!$connect) {
    die('连接失败:'. mysqli_error($connect));
}
mysqli_set_charset($connect,"utf8");
$sql="SELECT * FROM u_admin";
$data= mysqli_query( $connect, $sql );
if(!$data){
    die('无法读取数据:'. mysqli_error($connect));
}
echo '<h2>管理人员信息表<h2>';
echo '<table border="1"><tr><td>ID</td><td>姓名</td><td>密码</td><td>邮箱</td></tr>';
//使用 while 循环语句以“结果集->列名”的方式输出结果集$output 中的管理人员信息
while($output=mysqli_fetch_row($data)){
    echo "<tr>
            <td> {$output[0]}</td>".
            "<td>{$output[1]}</td>".
            "<td>{$output[2]}</td>".
            "<td>{$output[3]}</td>".
        "</tr>";
}
echo '</table>';
mysqli_close($connect);          //关闭数据库连接
?>
```

在 IE 浏览器中运行结果如图 15-6 所示。

图 15-6　mysqli_fetch_arow()函数

15.2.7 使用 mysqli_num_rows()函数获取查询结果集中的记录数

有时候需要获取 select 语句查询到的结果集中的数目，使用 mysqli_num_rows()函数来完成。该函数的语法格式如下：

```
mysqli_num_rows($result);
```

其中，$result 是必须参数，规定由 mysqli_query()、mysqli_store_result()或 mysqli_use_result()返回的结果集标识符。

【例 15-6】（实例文件：ch15\Chap15.6.php）mysqli_num_rows()函数。

```
<?php
$connect=mysqli_connect("localhost","root","123456","user_database");
if(!$connect) {
    die('连接失败: ' . mysqli_error($connect));
}
//设置编码，防止中文乱码
mysqli_set_charset($connect,"utf8");
$sql="SELECT * FROM u_admin";
echo "<h2>数据表共有记录: $num"."条</h2>";          //输出结果集中行的数目
if(!$data){
    die('无法读取数据:'. mysqli_error($connect));
}
echo '<h2>管理人员信息表<h2>';
echo '<table border="1"><tr><td>ID</td><td>姓名</td><td>密码
</td><td>邮箱</td></tr>';
//使用 while 循环语句以表格的形式输出数组结果集$output 中的数据
while($output=mysqli_fetch_array($data, MYSQLI_ASSOC)){
    echo "<tr>
            <td> {$output['id']}</td>".
            "<td>{$output['name']}</td>".
            "<td>{$output['pwd']}</td>".
            "<td>{$output['email']}</td>".
        "</tr>";
}
echo '</table>';
$data= mysqli_query( $connect, $sql );
$num=mysqli_num_rows($data);
mysqli_close($connect);              //关闭数据库连接
?>
```

图 15-7 mysqli_num_rows()函数

在 IE 浏览器中运行结果如图 15-7 所示。

15.3 PHP 操作 MySQL 数据库

PHP 数据库操作技术是 Web 开发过程中的核心技术。本节通过 PHP 和 MySQL 数据库实现学生成绩的简单管理系统，主要实现动态添加、查询、修改和删除学生的成绩。

15.3.1 使用 insert 语句动态添加学生成绩信息

在实现动态添加学生成绩前，首先需要创建数据库以及数据表，具体代码如下：

```
<?php
```

```
    $connect=mysqli_connect('localhost','root','123456');      //连接服务器
    if (!$connect) {                                           //检测是否连接成功
        die("连接服务器失败");                                  //连接服务器失败退出程序
    }
    mysqli_query($connect,"utf8");                             //设置编码类型
    //创建数据库命名为 student
    $sql_database = "CREATE DATABASE student";
    if (mysqli_query($connect,$sql_database)){                 //检测数据库是否创建成功
        echo "数据库 student 创建成功</br>";
    } else {
        echo "数据库 student 创建失败 " ."</br>";
    }
    //连接数据库 student
    $sele=mysqli_select_db($connect,"student");
    if(!$sele){                                                //检测数据库是否连接成功
        die("连接数据库失败");                                  //连接数据库失败退出程序
    }
    //创建数据表命名为 score，主键为 id(不为空整型)，变量名为 name(255 位不为空字符串)，变量名为 chinese(4 位不为空整型)，变量名为 english(4 位不为空整型)，变量名为 math(4 位不为空整型)
    $sql_table = "CREATE TABLE score( ".
        "id INT NOT NULL AUTO_INCREMENT, ".
        "name CHAR(255) NOT NULL, ".
        "chinese INT(4) NOT NULL, ".
        "english INT(4) NOT NULL, ".
        "math INT(4) NOT NULL, ".
        "PRIMARY KEY ( id )); ";
    $table = mysqli_query($connect,$sql_table);
    if($table){
        echo '数据表 score 创建成功</br>';
    }else{
        echo "数据表 score 创建失败</br>";
    }
    mysqli_close($connect);    //关闭数据库连接
    ?>
```

在 IE 浏览器中运行结果如图 15-8 所示，说明数据库以及数据表已经创建完成。在 phpMyAdmin 图像化管理工具中可以看到已经创建的数据库，如图 15-9 所示。

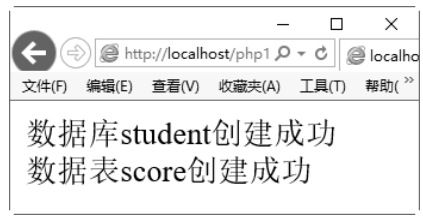


图 15-8　创建数据库

图 15-9　查看数据库

下面就来具体实现动态添加学生成绩信息。

【例 15-7】（实例文件：ch15\Chap15.7.php）使用 insert 语句动态添加学生成绩信息。

创建 left.php 的文件，作为左侧的功能导航。代码如下：

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body style="border: 1px solid rgba(6,4,5,0.67);overflow: hidden;width: 600px;">
<div>
    <h2 style="width: 600px;text-align: center;">欢迎来到学生成绩管理系统</h2>
    <div style="float:left;width: 150px;border: 2px solid #76eec6">
        <h4>管理功能</h4>
        <ul >
            <li><a href="index.html" target="iframe_a">添加学生成绩</a></li>
            <li><a href="Chap15.11.php" target="iframe_a">查询学生成绩</a></li>
            <li><a href="Chap15.12.php" target="iframe_a">修改学生成绩</a></li>
            <li><a href="Chap15.13.php" target="iframe_a">删除学生成绩</a></li>
        </ul>
    </div>
    <div style="float:left;">
        <iframe  src="index.html"  name="iframe_a"  frameborder="1"  width="400"  height="400"
style="*=auto;"></iframe>
    </div>
</div>
</body>
</html>
```

创建 add.php 文件，用来添加学生成绩的页面。在添加页面中，对添加的信息进行一些判断，当添加的学生成绩满足要求时，才会通过 POST 提交数据信息。当一切都满足时，提交数据到 Chap15.7.php 页面。

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <script>
        var reg=/[0-9]/;                //定义正则，用于判断学生成绩是否符合规范
        function check(){
            //获取页面中 text1，text2，text3，text4 元素
            var name1=document.getElementById('text1').value;
            var name2=document.getElementById('text2').value;
            var name3=document.getElementById('text3').value;
            var name4=document.getElementById('text4').value;
            //定义添加信息的规则
            if(name1==''){
                alert("姓名不能为空");
                return false;
            } else if(!reg.test(name2)||!reg.test(name3)||!reg.test(name4)){
                alert("输入格式不对");
                return false;
            } else if(name2==''){
                alert("语文成绩不能为空");
                return false;
```

```
            }else if(name3==''){
                alert("英语成绩不能为空");
                return false;
            }else if(name4==''){
                alert("数学成绩不能为空");
                return false;
            }else if(name2<0||name2>100){
                alert("语文成绩不合规范");
                return false;
            }else if(name3<0||name3>100){
                alert("英语成绩不合规范");
                return false;
            }else if(name4<0||name4>100){
                alert("数学成绩不合规范");
                return false;
            }else{
                return true;
            }
        }
    </script>
    <style>
        div{width: 200px;height: 200px;margin: 100px auto;}    <!--定义 div 的宽度、高度、外边距-->
    </style>
</head>
<body>
<div>
    <form action="Chap15.7.php" method="post" onsubmit="check()" id="form1" >
        <table>
            <h3>添加学生成绩界面</h3>
            <tr><td ><input type="text" name="text1" id="text1" placeholder="姓名"></td></tr>
            <tr><td><input type="text" name="text2" id="text2" placeholder="语文成绩"></td></tr>
            <tr><td><input type="text" name="text3" id="text3" placeholder="英语成绩"></td></tr>
            <tr><td><input type="text" name="text4" id="text4" placeholder="数学成绩"></td></tr>
            <tr><td><input type="submit" name="submit" value="添加"></td></tr>
        </table>
    </form>
</div>
</body>
</html>
```

在 Chap15.7.php 页面中通过$_POST 获取提交的信息，获取到信息后，执行 insert 语句把数据添加到数据库中。具体代码如下：

```
<?php
$connect=mysqli_connect("localhost","root","123456","student");
if(!$connect){
    echo "数据库连接失败";
}
mysqli_query($connect,"set names utf8");      //设置编码格式
//获取表单 POST 方法传递的数据
$username=$_POST["text1"];
$score1=$_POST["text2"];
$score2=$_POST["text3"];
$score3=$_POST["text4"];
//向数据库添加数据
$sql="INSERT INTO 'score' ('name', 'chinese', 'english', 'math') VALUES ('$username','$score1',
'$score2','$score3')";
```

```
    $result=mysqli_query($connect,$sql);
    if($result){ //判断是否添加成功，成功后跳转到 index.html 页面
        echo "<script>alert('添加成绩成功');this.location.href='index.html'</script>";
    } else{
        echo "<script>alert('添加成绩失败');this.location.href='index.html'</script>";
    }
    mysqli_close($connect);
    ?>
```

在 IE 浏览器中运行结果如图 15-10 所示。

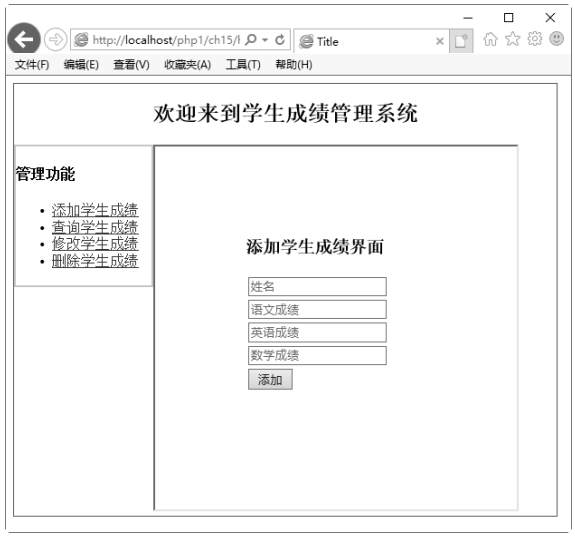


图 15-10　学生成绩添加界面

15.3.2　使用 select 语句查询学生成绩信息

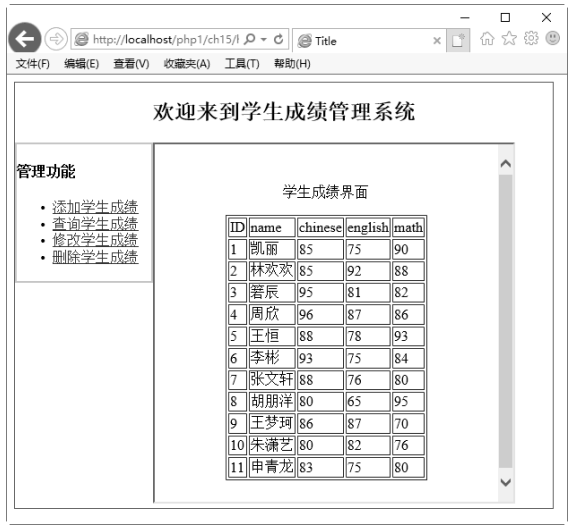
实现添加学生成绩后，即可对学生成绩表 score 进行查询操作了。下面使用 mysqli_query()函数执行 select 查询语句，使用 mysqli_fetch_object()函数获取查询结果集，通过 while 循环语句输出查询的结果。

【例 15-8】（实例文件：ch15\Chap15.8.php）使用 select 语句查询学生成绩信息。

```
    <?php
    $connect=mysqli_connect("localhost","root","123456","student");
    if(!$connect){
        echo "连接数据库失败";
    }
    mysqli_query($connect,"set names utf8");        //设置编码格式
    $sql="SELECT * FROM 'score'";
    $data=mysqli_query($connect,"$sql");
    echo '<table border="1"><caption>学生成绩界面</caption><tr><td>ID</td><td>name</td><td>chinese
</td><td>english</td><td>math</td></tr>';
    //使用 while 循环语句以“结果集->列名”的方式输出结果集$output 中的学生成绩信息
    while($output=mysqli_fetch_object($data)){
        echo "<tr>
                <td> {$output->id}</td>".
            "<td>{$output->name}</td>".
            "<td>{$output->chinese}</td>".
            "<td>{$output->english}</td>".
            "<td>{$output->math}</td>".
```

```
            "</tr>";
    }
    echo '</table>';
    ?>
    <style>*{margin:15px auto;}</style>  //设置外边距
```

在 IE 浏览器中运行结果如图 15-11 所示。



ID	name	chinese	english	math
1	凯丽	85	75	90
2	林欢欢	85	92	88
3	蓉辰	95	81	82
4	周欣	96	87	86
5	王恒	88	78	93
6	李彬	93	75	84
7	张文轩	88	76	80
8	胡朋洋	80	65	95
9	王梦珂	86	87	70
10	朱潇艺	80	82	76
11	申青龙	83	75	80

图 15-11　使用 select 语句查询学生成绩信息

15.3.3　使用 update 语句修改学生成绩信息

学生的信息查询出来后，可以根据情况来进行修改。下面使用 update 语句动态编辑数据库中学生成绩。

首先创建 Chap15.9.php 文件，在该页面中把学生信息查询出来，然后在循环输出时添加 “修改”列，单击某条记录的“修改”链接，会链接到 updata.php 文件，并把对应的 id 参数也一起传递过去。

【例 15-9】（实例文件：ch15\Chap15.9.php）用 update 语句修改学生成绩信息。

```
    <?php
    $connect=mysqli_connect("localhost","root","123456","student");
    if(!$connect){
        echo "连接数据库失败";
    }
    mysqli_query($connect,"set names utf8");     //设置编码格式
    $sql="SELECT * FROM 'score'";
    $data=mysqli_query($connect,"$sql");

    echo '<table border="1"><caption>修改学生成绩界面</caption><tr><td>ID</td><td>name</td><td>chinese
</td><td>english</td><td>math</td><td>EDIT</td></tr>';
    //使用 while 循环语句以“结果集->列名”的方式输出结果集$output 中的学生成绩信息
    while($output=mysqli_fetch_object($data)){
        echo "<tr>
              <td>{$output->id}</td>".
            "<td>{$output->name}</td>".
            "<td>{$output->chinese}</td>".
            "<td>{$output->english}</td>".
            "<td>{$output->math}</td>".
```

```
            "<td><a href='update.php?id=$output->id'>修改</a></td>".
            "</tr>";
    }
    echo '</table>';
    ?>
    <style>*{margin:15px auto;}</style>
```

在 update.php 文件中通过$_GET 获取链接过来的 id 参数，通过该 id 查询数据库中对应的信息并显示出来，然后修改需要更正的内容。修改完成后，单击“修改”按钮，表单会把修改的内容发送到 update1.php 文件。

```
    <?php
    $connect=mysqli_connect("localhost","root","123456","student");
    if(!$connect){
       echo "数据库连接失败";
    }
    $id=$_GET["id"];
    mysqli_query($connect,"set names utf8");     //设置编码格式
    $sql="SELECT * FROM 'score' where id=$id";
    $data=mysqli_query($connect,"$sql");
    $output=mysqli_fetch_object($data);
    ?>
    <div>
       <h2>学生成绩的修改界面</h2>
       <form action="update1.php" method="post" name="form1">
          姓        名：<input type='text' name="txt1"
value="<?php echo $output->name ?>"><br/>
          <input type='hidden' name='id' value="<?php echo $output->id ?>">
          语文成绩：<input type='text' name="txt2" value="<?php echo $output->chinese ?>"><br/>
          英语成绩：<input type='text' name="txt3" value="<?php echo $output->english ?>"><br/>
          数学成绩：<input type='text' name="txt4" value="<?php echo $output->math ?>"><br/>
          <input type="submit" value="修改">
       </form>
    </div>
    <style>
       div{width: 300px;height: 200px;margin: 100px auto;}    <!--定义 div 的宽度、高度、外边距-->
    </style>
```

在 update1.php 文件中对数据库中的信息进行更改操作，修改完成后会跳转到 Chap15.9.php 页面。具体代码如下：

```
    <?php
    $connect=mysqli_connect("localhost","root","123456","student");
    if(!$connect){
       echo "连接数据库失败";
    }
    mysqli_query($connect,"set names utf8");       //设置编码格式
    $name=$_POST['txt1'];
    $chinese=$_POST["txt2"];
    $english=$_POST["txt3"];
    $math=$_POST["txt4"];
    $id=$_POST["id"];
    $mysql="update score set name='$name',chinese='$chinese',english='$english',math='$math' where
id=$id";
    $result=mysqli_query($connect,"$mysql");
    if($result){
       echo "<script>alert('成绩修改成功');this.location.href='Chap15.9.php'</script>";
    }
    ?>
```

在 IE 浏览器中运行，单击其中一个“修改”链接，页面将跳转到图 15-12 所示页面；更改信息，然后单击“修改”按钮，弹出“成绩修改成功”对话框，如图 15-13 所示；单击“确定”按钮，会跳转到 Chap15.9.php 页面，如图 15-14 所示。

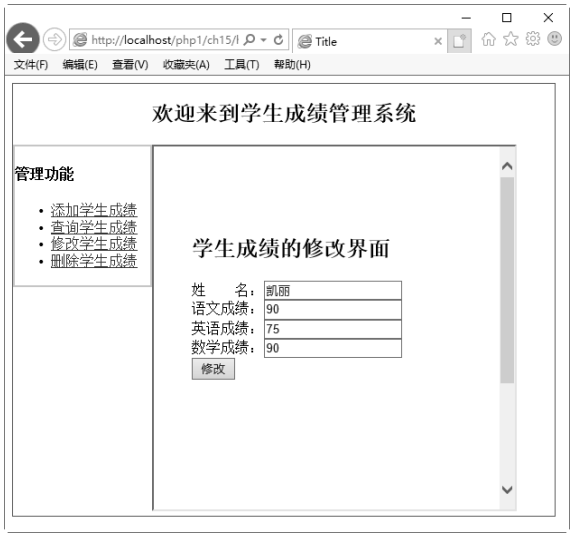


图 15-12　修改界面

图 15-13　“成绩修改成功”对话框

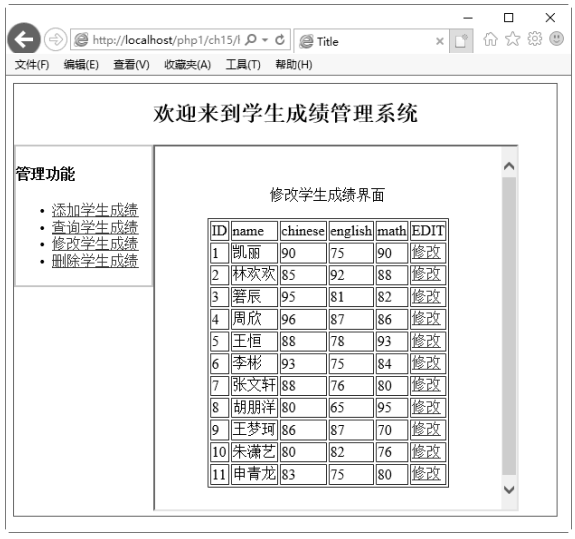


图 15-14　修改完成

15.3.4　使用 delete 语句删除学生成绩信息

删除学生成绩与更新类似。首先创建 Chap15.10.php 文件，在该页面中把学生信息查询出来，然后在循环输出时添加“删除”列，单击某条记录的“删除”链接，会链接到 delete.php 文件，并把对应的 id 参数也一起传递过去。

【例 15-10】（实例文件：ch15\Chap15.10.php）使用 delete 语句删除学生成绩信息。

```
<?php
```

```
    $connect=mysqli_connect("localhost","root","123456","student");
    if(!$connect){
        echo "连接数据库失败";
    }
    mysqli_query($connect,"set names utf8");
    $sql="SELECT * FROM 'score'";
    $data=mysqli_query($connect,"$sql");
    echo  '<table  border="1"><caption> 删 除 学 生 成 绩 界 面 </caption><tr><td>ID</td><td>name
</td><td>chinese</td><td>english</td><td>math</td><td>DEL</td></tr>';
    //使用 while 循环语句以“结果集->列名”的方式输出结果集$output 中的学生成绩信息
    while($output=mysqli_fetch_object($data)){
        echo "<tr>
              <td>{$output->id}</td>".
            "<td>{$output->name}</td>".
            "<td>{$output->chinese}</td>".
            "<td>{$output->english}</td>".
            "<td>{$output->math}</td>".
            "<td><a href='delete.php?id=$output->id' onclick='click()'>删除</a></td>".
            "</tr>";
    }
    echo '</table>';
    ?>
    <style>*{margin:15px auto;}</style>
```

在 delete.php 文件中通过$_GET 获取链接过来的 id 参数，通过该 id 查询数据库中对应的信息并显示出来。单击“删除”按钮，表单会把内容发送到 delete1.php 文件。

```
    <?php
    $connect=mysqli_connect("localhost","root","123456","student");
    if(!$connect){
        echo "数据库连接失败";
    }
    mysqli_query($connect,"set names utf8");     //设置编码格式
    $id=$_GET["id"];
    $sql="SELECT * FROM 'score' where id=$id";
    $data=mysqli_query($connect,"$sql");
    $output=mysqli_fetch_object($data);
    ?>
    <div>
        <h2>学生成绩的删除界面</h2>
        <form action="delete1.php" method="post" name="form1">
            姓          名 : <input type='text' name="txt1"
value="<?php echo $output->name ?>"><br/>
            <input type='hidden' name='id' value="<?php echo $output->id ?>">
            语文成绩: <input type='text' value="<?php echo $output->chinese ?>"><br/>
            英语成绩: <input type='text' value="<?php echo $output->english ?>"><br/>
            数学成绩: <input type='text' value="<?php echo $output->math ?>"><br/>
            <input type="submit" value="删除">
        </form>
    </div>
    <style>
        div{width: 300px;height: 200px;margin: 100px auto;}    <!--定义 div 的宽度、高度、外边距-->
    </style>
```

在 delete1.php 文件中对数据库中的信息进行删除操作，删除完成后会跳转到 Chap15.10.php。具体代码如下：

```
    <?php
    $connect=mysqli_connect("localhost","root","123456","student");
    if(!$connect){
```

```
    echo "数据库连接失败";
}
mysqli_query($connect,"set names utf8");          //设置编码格式
$id=$_POST["id"];                                 //获取表单 POST 传递过来的 id
$txt=$_POST["txt1"];                              //获取表单 POST 传递过来的姓名
$mysql="delete from score where id=$id";          //使用 delete 语句删除数据
$result=mysqli_query($connect,"$mysql");
if($result){
    echo "<script>alert('$txt.的成绩删除成功');this.location.href='Chap15.13.php'</script>";
}
?>
```

在 IE 浏览器中运行，单击其中一个“删除”时，页面将跳转到图 15-15 所示页面；单击“删除”按钮，会弹出成绩删除对话框，如图 15-16 所示；单击“确定”按钮会跳转到 Chap15.10.php 页面，如图 15-17 所示。

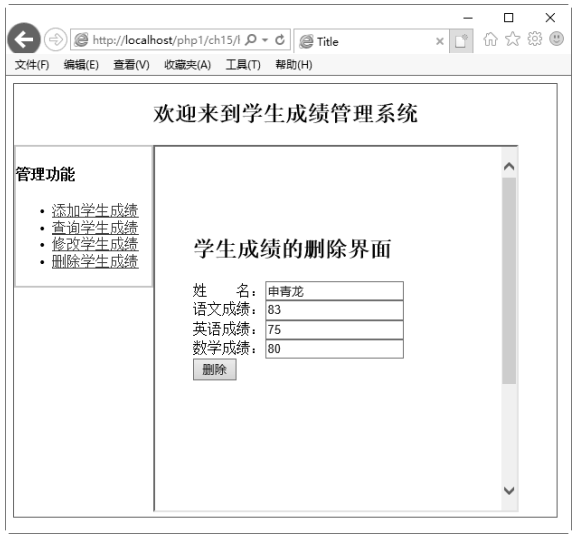


图 15-15　删除界面

图 15-16　删除成功对话框

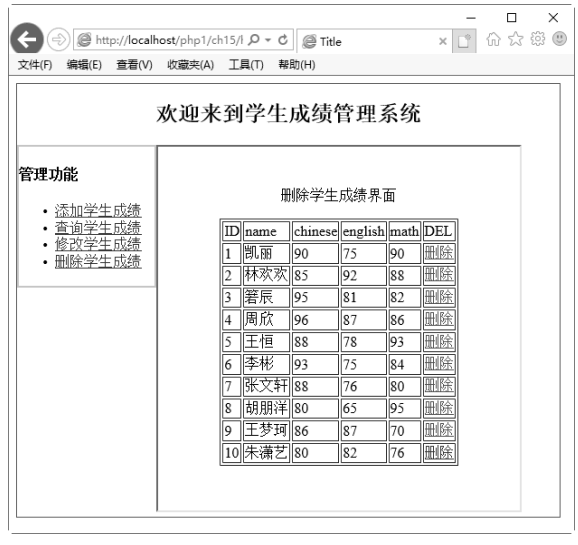


图 15-17　删除完成页面

15.4 就业面试技巧与解析

15.4.1 面试技巧与解析（一）

面试官：MySQL 优化是怎么做的？

应聘者：MySQL 优化主要从以下几个方面来实现。

（1）设计角度：存储引擎的选择，字段类型的选择。

（2）功能角度：可以利用 MySQL 自身的特性，如索引、查询缓存、碎片整理、分区、分表等。

（3）SQL 语句优化方面：尽量简化查询语句，能查询字段少就尽量少查询字段，优化分页语句、分组语句等。

（4）从硬件上升级数据库服务器。

15.4.2 面试技巧与解析（二）

面试官：索引有几种？

应聘者：索引主要有以下几种。

（1）主键索引：数据库表经常有一列或列组合，其值唯一标识表中的每一行。该列称为表的主键。在数据库关系图中为表定义主键将自动创建主键索引，主键索引是唯一索引的特定类型。该索引要求主键中的每个值都唯一。当在查询中使用主键索引时，允许对数据的快速访问。

（2）普通索引：使用字段关键字建立的索引，主要是提高查询速度。

（3）唯一索引：唯一索引是不允许其中任何两行具有相同索引值的索引。当现有数据中存在重复的键值时，大多数数据库不允许将新创建的唯一索引与表一起保存。数据库还可防止在表中创建重复键值的新数据。