**第11章 PHP访问MYSQL数据库**

**1. web数据库架构的工作原理回顾**

**以Book-O-Rama书店为例。步骤如下：**

1）用户的web浏览器发出HTTP请求，请求特定的web页面。例如，该用户可能以HTML表单形式搜索Book-O-Rama书店所有Laura编写的图书，搜索结果页为result.php

2）web服务器收到result.php的请求，获取该文件，并将它传到php引擎，要求它处理

3）PHP引擎开始解析脚本。脚本中有一条连接数据库的明亮，还有执行一个查询的命令。PHP打开通向MYSQL数据库的连接，发送适当的查询

4）mysql服务器接受数据库查询并处理。将结果返回到PHP引擎。

5）PHP引擎完成脚本运行，将查询结果格式化成HTML格式，将输出的HTML返回到web服务器。

6）web服务器将HTML发送到浏览器。 这样用户就可以看到所搜索的图书了。



**search.html——Book-O-Rama的数据库搜索页面**

**————————————————————————————————**

**<html>**

**<head>**

**<title>Book-O-Rama Catalog Search</title>**

**</head>**

**<body>**

**<h1>Book-O-Rama Catalog Search</h1>**

**<form action="results.php" method="post">**

**Choose Search Type:<br />**

**<select name="searchtype">**

**<option value="author">Author**

**<option value="title">Title**

**<option value="isbn">ISBN**

**</select>**

**<br />**

**Enter Search Term:<br />**

**<input name="searchterm" type="text" size="40">**

**<br />**

**<input type="submit" name="submit" value="Search">**

**</form>**

**</body>**

**</html>**

**————————————————————————————————**

**results.php——从mysql数据库获取并格式化搜索结果，以便显示结果**

**————————————————————————————————**

**<html>**

**<head>**

**<title>Book-O-Rama Search Results</title>**

**</head>**

**<body>**

**<h1>Book-O-Rama Search Results</h1>**

**<?php**

**// create short variable names**

**$searchtype=$\_POST['searchtype'];**

**$searchterm=trim($\_POST['searchterm']);**

**if (!$searchtype || !$searchterm) {**

**echo 'You have not entered search details. Please go back and try again.';**

**exit;**

**}**

**if (!get\_magic\_quotes\_gpc()){**

**$searchtype = addslashes($searchtype);**

**$searchterm = addslashes($searchterm);**

**}**

**//PHP登录mysql ( -h -u -p -D )**

**@ $db = new mysqli('localhost', 'bookorama', 'bookorama123', 'books');**

**//连接错误提示**

**if (mysqli\_connect\_errno()) {**

**echo 'Error: Could not connect to database. Please try again later.';**

**exit;**

**}**

**//执行查询**

**$query = "select \* from books where ".$searchtype." like '%".$searchterm."%'";**

**$result = $db->query($query);**

**//结果集的记录数目**

**$num\_results = $result->num\_rows;**

**echo "<p>Number of books found: ".$num\_results."</p>";**

**//循环逐行输出结果集**

**for ($i=0; $i <$num\_results; $i++) {**

**$row = $result->fetch\_assoc();**

**echo "<p><strong>".($i+1).". Title: ";**

**echo htmlspecialchars(stripslashes($row['title']));**

**echo "</strong><br />Author: ";**

**echo stripslashes($row['author']);**

**echo "<br />ISBN: ";**

**echo stripslashes($row['isbn']);**

**echo "<br />Price: ";**

**echo stripslashes($row['price']);**

**echo "</p>";**

**}**

**$result->free();**

**$db->close();**

**?>**

**</body>**

**</html>**

**————————————————————————————————**

**2. 从web查询数据库的基本步骤**

**1）检查并过滤来自用户的数据**

**2）建立一个到适当数据库的连接**

**3）查询数据库**

**4）获取查询结果**

**5）将查询结果显示给用户**

**2.1 检查与过滤用户输入的数据**

**$searchterm=trim($\_POST['searchterm']);**

**如果输入的信息都是空白字符，trim()函数将删除它们：**

**if (!$searchtype || !$searchterm) {**

**echo 'You have not entered search details. Please go back and try again.';**

**exit;**

**}**

**如果魔术引号特性没有打开，需要转义数据，从数据库出来的数据的反斜杠必须过滤掉。**

**if (!get\_magic\_quotes\_gpc()){**

**$searchtype = addslashes($searchtype);**

**$searchterm = addslashes($searchterm);**

**}**

**2.2 建立一个连接**

**PHP为连接mysql提供了函数库：mysqli**

**实例化mysqli类，并创建到主机localhost的连接，用户名是： 密码是： 数据库是：books**

**@ $db = new mysqli('localhost', 'bookorama', 'bookorama123', 'books');**

**面向过程的方法：**

**@ $db = mysqli\_connect('localhost', 'bookorama', 'bookorama123', 'books'); 返回一个资源。**

**检查连接的结果：**

**if (mysqli\_connect\_errno()) {**

**echo 'Error: Could not connect to database. Please try again later.';**

**exit;**

**}**

**2.3 选择使用的数据库**

**命令行中： use books；**

**对应PHP中：**

**$db->select\_db(*dbname*)**

**或**

**mysqli\_select\_db(*db\_resource,db\_name*)**

**2.4 查询数据库**

**$query = "select \* from books where ".$searchtype." like '%".$searchterm."%'";**

**$result = $db->query($query); 返回一个结果对象**

**面向过程的函数是： $result = mysqli\_query($db,$query);返回一个结果资源**

**2.5 检索查询结果**

**$num\_results = $result->num\_rows; 返回的行数保存在结果对象的num\_rows成员变量中。**

**面向过程，查询返回的行数：**

**$num\_results = mysqli\_num\_rows($result);**

**在每一轮循环中，调用$result->fetch\_assoc(); （或mysqli\_fetch\_assoc($result)函数）**

**for ($i=0; $i <$num\_results; $i++) {**

**$row = $result->fetch\_assoc();**

**。。。**

**}**

**给定相关数组$row，遍历每一个字段，并适当地显示它们**

**也可以将一行取回到一个对象中：**

**$row = $result->fetch\_object();**

**或**

**$row = mysqli\_fetch\_object($result);**

**然后通过$row->title,$row>author等，访问每个属性。**

**2.6 从数据库断开连接**

**$result->free();**

**或**

**mysqli\_free\_result($result);**

**然后使用：**

**$db->close();**

**或者**

**mysqli\_close($db);**

**来关闭一个数据库连接。脚本执行完毕，它们也将被自动关闭。**

**3. 将新信息放入数据库**

**类似地，插入数据到数据库中的步骤：**

**建立一个连接、发送查询，最后检查结果。这里的查询，是insert而不是select。**



**newbook.html——图书输入页的HTML**

**—————————————————————————————————**

**<html>**

**<head>**

**<title>Book-O-Rama - New Book Entry</title>**

**</head>**

**<body>**

**<h1>Book-O-Rama - New Book Entry</h1>**

**<form action="insert\_book.php" method="post">**

**<table border="0">**

**<tr>**

**<td>ISBN</td>**

**<td><input type="text" name="isbn" maxlength="13" size="13"></td>**

**</tr>**

**<tr>**

**<td>Author</td>**

**<td> <input type="text" name="author" maxlength="30" size="30"></td>**

**</tr>**

**<tr>**

**<td>Title</td>**

**<td> <input type="text" name="title" maxlength="60" size="30"></td>**

**</tr>**

**<tr>**

**<td>Price $</td>**

**<td><input type="text" name="price" maxlength="7" size="7"></td>**

**</tr>**

**<tr>**

**<td colspan="2"><input type="submit" value="Register"></td>**

**</tr>**

**</table>**

**</form>**

**</body>**

**</html>**

**—————————————————————————————————**

**insert\_book.php ——该脚本将新的图书写入到数据库**

**—————————————————————————————————**

**<html>**

**<head>**

**<title>Book-O-Rama Book Entry Results</title>**

**</head>**

**<body>**

**<h1>Book-O-Rama Book Entry Results</h1>**

**<?php**

**// create short variable names**

**$isbn=$\_POST['isbn'];**

**$author=$\_POST['author'];**

**$title=$\_POST['title'];**

**$price=$\_POST['price'];**

**if (!$isbn || !$author || !$title || !$price) {**

**echo "You have not entered all the required details.<br />"**

**."Please go back and try again.";**

**exit;**

**}**

**if (!get\_magic\_quotes\_gpc()) {**

**$isbn = addslashes($isbn);**

**$author = addslashes($author);**

**$title = addslashes($title);**

**$price = doubleval($price);**

**}**

**@ $db = new mysqli('localhost', 'bookorama', 'bookorama123', 'books');**

**if (mysqli\_connect\_errno()) {**

**echo "Error: Could not connect to database. Please try again later.";**

**exit;**

**}**

**$query = "insert into books values**

**('".$isbn."', '".$author."', '".$title."', '".$price."')";**

**$result = $db->query($query);**

**if ($result) {**

**echo $db->affected\_rows." book inserted into database.";**

**} else {**

**echo "An error has occurred. The item was not added.";**

**}**

**$db->close();**

**?>**

**</body>**

**</html>**

**—————————————————————————————————**

**doubleval()函数，对数字字段过滤，去除所有临时字符，不会在小数点的前后插入斜杠。**

**$result = $db->query($query);**

**或者 mysqli\_query($query)**

**insert 和 select 的显著不同在于：对mysqli\_affected\_rows()**

**echo $db->affected\_rows." book inserted into database.";**

**确定select返回多少行记录，我们使用mysqli\_num\_rows()**

**当编写一个修改数据库的查询时，例如 insert delete update，应该使用mysqli\_affected\_rows()函数。**

**4. 使用prepared语句**

**对于在执行大量具有不同数据的相同查询时，可以提高执行速度，也可以避免SQL注射风格的攻击。**

**prepared语句的基本思想是：可以向mysql发送一个想要执行的查询模板，然后单独发送数据。**

**在insert\_book.php脚本中，可以使用prepared语句。如下所示：**

**$query = "insert into books values(?,?,?,?) ";**

**$stmt = $db->prepare($query);**

**$stmt->bind\_param("sssd",$isbn,$author,$title,$price);**

**$stmt->execute();**

**echo $stmt->affected\_rows.' book inserted into database. ';**

**$stmt->close();**

**$db->prepare()对应的过程版本是，mysqli\_stmt\_prepare()函数。此行构建一个语句对象或需要用来完成实际处理的资源。**

**语句对象有一个bind\_param()方法。对应过程版本是mysqli\_stmt\_bind\_param()函数。 用途是，告诉PHP哪些变量应该被问号所替换。这里，所传递的值意味着4个参数分别是字符串、字符串、字符串、双精度。**

**格式化字符串中的其他可能字符还有：i表示整数，b表示bool。在这个参数之后，必须列出与语句中的问号数量相同的变量。它们将一次被替换。**

**调用$stmt->execute()函数，对应过程版本是mysqli\_stmt\_execute()函数，将真正运行这个查询。我们可以访问受影响的行数，并关闭这个语句。**

**prepared语句，优点：可以改变这4个绑定变量的值，然后重新执行这个语句。这对于循环执行批量插入操作来说，非常有用。**

**与绑定参数一样，也可以绑定结果。对于select查询，可以使用$stmt->bind\_result()函数 或 mysqli\_stmt\_bind\_result()函数。**

**调用$stmt->fetch()函数 或mysqli\_stmt\_fetch()函数，结果集下一行的列值，将被填充到这些绑定变量中。例如，**

**$stmt->bind\_result($isbn,$author,$title,$price);**

**将这4个变量，绑定到将通过查询返回的4列。**

**调用$stmt->execute();后，可以在循环中调用 $stmt->fetch();它将获得下一结果行，并填充到4个绑定变量中。**

**相应地，可以使用mysqli\_stmt\_bind\_param() 和 mysqli\_stmt\_bind\_result()函数**