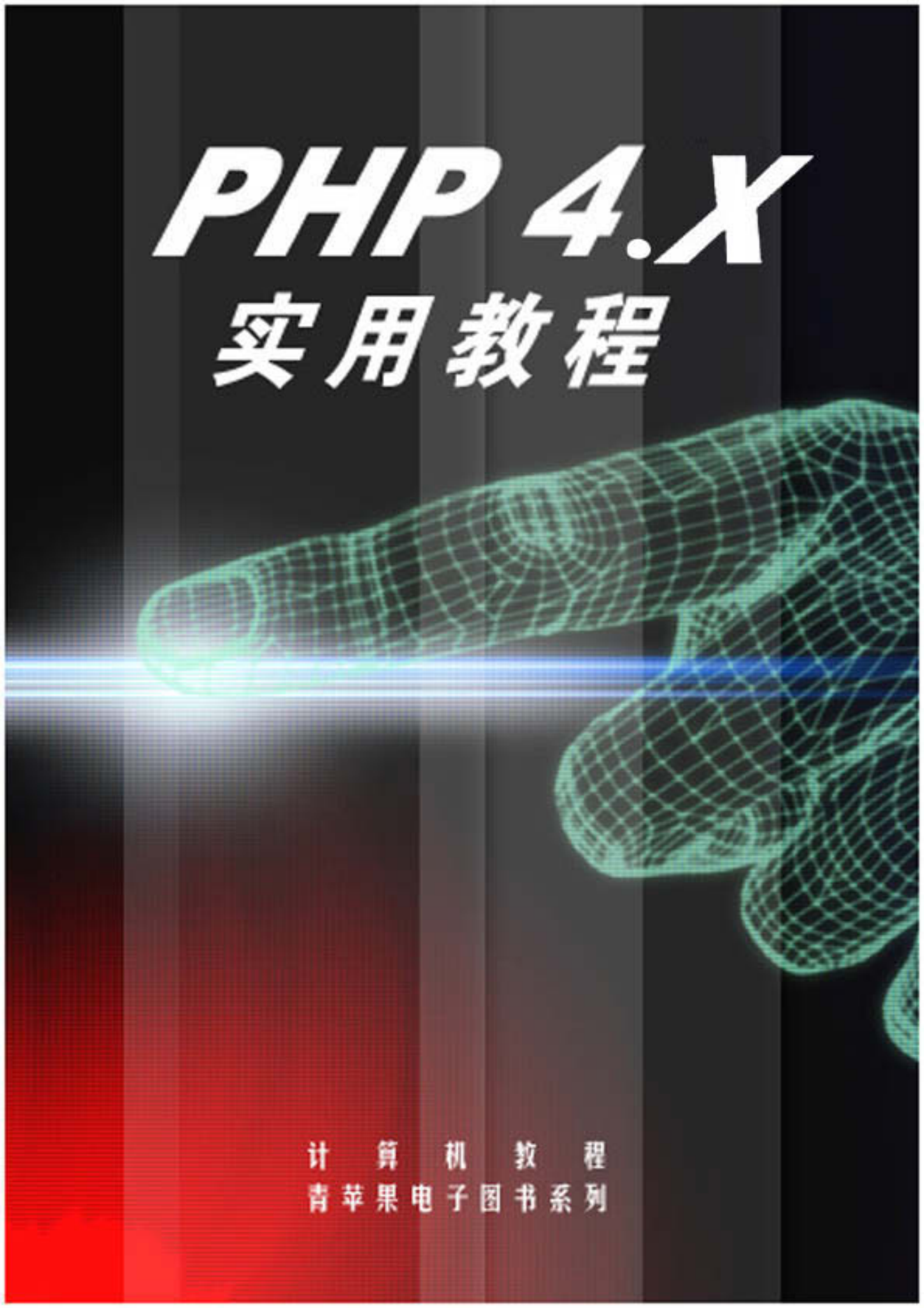
PHP 4.X
实用教程



计 算 机 教 程
青苹果电子图书系列

PHP 4.X 实用教程

王孟奎　韩　笑

Green Apple Data Center

前　言

PHP 是 Linux Web 服务器端与 Microsoft ASP 类似的一种脚本编程语言，PHP 4.04 是在 2001 年 5 月份发布的最新版本，它可以高性能、高效率运行在 Linux Web 服务器平台上，PHP 容易学习和使用并且具有强大的功能，所以逐步得到广大程序员的喜爱，它将在 Web 程序设计领域中掀起翻天覆地的变化。

PHP 是一种采用与 Linux 相同发行方式的基于 GNU 协议的自由软件，它的语法借鉴了 C、Java 及 Perl 语言，使得有上述语言基础的程序员可以轻松过渡过来。PHP 相对于 ASP 来讲，除了支持 Linux 以外最重要的特点是效率很高，整个 PHP 4.04 采用 C/C++编写，压缩后不到 1.9M，尤其是对于 MySQL 等数据库的存取非常简练，没有额外的开销，简直就如同一只手直接伸入数据库中抓取东西一样。这一点与 ASP 完全不一样，ASP 必须通过中间层 ADO 或者 ODBC 才能对数据库中的信息进行存取，效率当然要打折扣。PHP 不仅支持 Linux 和 Unix 操作系统，而且还支持各种版本的 Windows 环境，所以喜爱 Windows 平台的用户，也可以非常方便地在 Windows 上面建立支持 PHP 的开发和运行环境的 Web 站点,所开发的程序几乎不用改写就直接可以在 Linux 平台上运行。

在动态网页兴起的同时，为了存储大量的数据，数据库设计渐渐登上了 Web 程序设计的前台，由于瑞典 T.e.X 公司的 MySQL 数据库的全新设计，使得该数据库工作速度和效率得到充分的发挥，引起了世人的瞩目，而且它和 PHP 几乎成了一种完美的组合，使用这两个工具可以很容易地进行网页数据库编程，并且具有强大的功能。

本书共分为 7 章，下面是每章的内容概括。

第 1 章对 PHP 语言进行简单介绍，让读者了解 PHP 语言的发展历史、特点、功能等。

第 2 章对 PHP 在 Linux/Unix 和 Windows 平台的安装和配置进行讲解。

第 3 章介绍 PHP 的基本编程语法。

第 4 章对 PHP 常用的函数进行讲解。

第 5 章介绍了几个有用的 PHP 实例。

第 6 章详细讲解 PHP 与 MySQL 的网页数据库编程。

第 7 章通过一个具体的综合实例讲解由 PHP 和 MySQL 组建的一个论坛程序。

在本书的结尾附录中主要包括 2 个部分，附录 A 列出了 PHP 的函数表，附录 B 中列出了国内外著名的 PHP 资源网站。本书还附带一张光盘，包括 Windows 和 Linux 平台下面最新的 PHP、Apache 和 MySQL 软件及其本书全部范例程序。

本书由五洋工作室策划，王孟奎、韩笑等编著。在本书的编写过程中，得到了许多同行及网友的支持和帮助，他们对本书的内容和结构提出了宝贵的意见，在此向他们表示感谢。

如果在阅读过程中发现本书有疏漏或错误之处，请广大读者批评指正。

编　者

Green Apple Data Center

内 容 提 要

本书以 PHP 4.04 为蓝本，介绍了 PHP 的基础知识和应用方法，包括 PHP 简介、PHP 的安装和设置、PHP 的语法、PHP 常用函数说明、PHP 简单应用实例、PHP 数据库编程、PHP 与 MySQL 综合应用实例等 7 章内容。另外，本书的附录 A 提供了 PHP 的函数列表，附录 B 中列出了国内外著名的 PHP 资源网站。

本书配套光盘提供了 Windows 和 Linux 平台下面最新版本的 PHP、Apache 和 MySQL 软件及其本书全部范例程序。

本书适合初、中级 PHP 程序员，也适合其他脚本编程语言如 Perl、ASP、JSP 的程序员参考，同时适合网站所有从业人员比如网页制作人员及其他管理人员参考。

Green Apple Data Center

目 录

Green Apple Data Center

Green Apple Data Center

Green Apple Data Center

Green Apple Data Center

第 1 章　PHP 简介

本章要点：

- PHP 的发展历史
- PHP 的新功能和特征

1.1　什么是 PHP

PHP 是一种服务器端 HTML 嵌入式脚本描述语言，全称是 Professional Hypertext Pre- processor，目前正式发布的最高版本为 4.04。服务器端脚本技术又分为嵌入式与非嵌入式两种，PHP 是嵌入式的，类似的如 ASP。它是一种功能非常强大的面向 Internet/Intranet 的编程语言，可以开发动态交互的 Web 应用程序，可在多种系统平台和多种 Web 服务器中使用，是真正的跨平台、跨服务器的开发语言。

1.2　PHP 的发展历史

1994 年秋季，RasmusLerdorf 开始构思 PHP，早期的非发行版本被用在他的主页上，以追踪谁在看他的在线简历。1995 年年初第一版本出台，当时 PHP 只被认为是个人主页开发工具。它由一个非常单纯的只能理解很少数特殊宏的分析引擎和一些用在主页后端通用的工具组成。如留言簿，计数器和其他一些东西。这个分析器在 1995 年中被重写并被命名为 PHP/FI 第二版。FI 来自 Rasmus 写的另外一个包，用于解释 HTML 形式的数据。他结合了个人主页工具脚本和形式解析器，并加上 mSQL 支持。这样就产生 PHP/FI 了。PHP/FI 以令人惊奇的步调成长，人们开始把自己的代码贡献给它。

很难给出它的详细统计数据，但可以估计在 1996 年末，整个世界至少有 15,000 个网站在用 PHP/FI。到 1997 年年中，这个数字已经超过 50,000 了。而在此时 PHP 的发展也发生了变化。由 Rasmus 自己偏爱的和几个人开发的项目变成一个更有组织的团体成就。ZeevSuraski 和 AndiGutmans 重写了解析器。这个新的解析器成为 PHP 版本 3 的基础。许多有用的代码从 PHP/FI 继承到 PHP 3，并且很多是完全重写的。

1999 年年中，不管是 PHP/FI 或 PHP 3 与很多商业产品捆绑在一块，例如 C2 级强度的 Web 服务器和红帽子 Linux。根据 NetCraft 提供的数据推断，保守估计全世界应用 PHP 的网站已超过 150,000 个。由此看来，它比在因特网上运行 Netscape 的旗舰企业服务器的站点还多。

如今是数以百万计的网站使用 PHP，其网站数目远远大于使用 ASP 网站的数目。

与此手册一样，下一代 PHP 的开发工作正在进行中。它将利用强大的 Zend 脚本引擎产生更高的性能，并且它将支持 Apache 以外的 Web 服务器以本地服务器模块运行。

1.3　PHP 的功能概述

作为最基本的应用 PHP 能够完成任何其他 CGI 程序所能完成的任务，如收集表格数据、产生动态页内容或者发送接收 cookies 等。

也许 PHP 最强大最突出的特性在于它能支持大量的数据库，这使得编写支持数据库的网页变得越来越简单，以下是 PHP 能够支持的数据库的列表：

AdabasD	InterBase	Solid
dBase	mSQL	Sybase
Empress	MySQL	Velocis
FilePro	Oracle	Unixdbm

Green Apple Data Center

Informix PostgreSQL

PHP 同时也支持使用其他协议的服务如 IMAP、SNMP、NNTP、POP3 甚至还支持 HTTP，用户也可以开创全新的网络服务方式与其他协议相互作用。

基于服务器端——由于 PHP 是在 Web 服务器端运行的，所以 PHP 程序可以很大、很复杂而不会降低客户端的运行速度。

跨平台——虽然本书是以 Windows 为重点介绍的，但 PHP 程序可以运行在 UNIX、Linux 操作系统下。

嵌入 HTML——因为 PHP 语言可以嵌入到 HTML 内部，所以 PHP 很容易学习。

简单的语言——和 Java、C++不同，PHP 语言坚持以基本语言为基础，但它的功能也强大到足以支持任何类型的 Web 站点。

效率高——和其他的解释性语言相比，PHP 消耗较少的系统资源。当 PHP 作为 ApacheWeb 服务器的一部分时，运行代码不需要调外部二进制程序，服务器解释脚本不需要承担任何额外负担。

分析 XML——用户可以组建一个可以读取 XML 信息的 PHP 版本。

数据库模块——用户可以使用 PHP 存取 Oracle、Sybase、MSSQL、MySQL、mSQL、PostgreSQL、dBase、FilePro、Solid、Unixdbm、Informix/Illustra 等类型的数据库，以及任何支持 ODBC 标准的数据库。

文件存取——PHP 有许多支持文件存取的函数。

文本处理——PHP 有许多函数处理字符串，其中包括模式匹配的能力。

复杂的变量——PHP 支持标量、数组、关联数组等变量，这给用户提供了支持其他高级数据结构的坚实基础。

图像处理——用户可以使用 PHP 动态地创建图象。

1.4 PHP 的特征

PHP 是一种采用 Linux 同样发行方式，基于 GNU 协议的自由软件 Freeware。与 ASP 一样，是一种内嵌于 HTML 中的服务器端脚本编程语言，它的语法借鉴了 C、Java 及 Perl 语言，使得有上述语言基础的程序员可以轻松过渡过来。PHP 相对于 ASP 来讲，除了前面谈到的支持 Linux 以外，最重要的特点是效率最高，整个 PHP 4.0 采用 C/C++编写压缩后不到 1.9M，尤其是对于 MySQL 等数据库的存取非常直接简练，没有额外的开销，简直就如同一只手直接伸入数据库中抓取东西一样，这一点与 ASP 完全不一样。ASP 必须通过中间层 ADO 或者 ODBC 才能对数据库中的信息进行存取，效率当然要打折扣。当然 PHP 不仅支持 Linux，还支持各种版本的 Windows，甚至可以在 Windows95 上构造 PHP 的运行环境。所以在 Windows 平台非常方便地利用各种桌面工具开发的程序几乎不用改写直接就可以在 Linux 上运行。PHP 语言主要具有以下特征：

（1）免费、轻巧快速、真正跨平台。

（2）PHP 是一种遵守 GNU 条约的软件。根据此条约，所有用户都可以免费使用 PHP 并可以得到它的源代码，还可以在源代码上进行修改和完善，开发成适合自己使用的新的版本。当然这个新形成的版本同样是遵守 GNU 的。这就意味着全世界成千上万的程序员都在不断的完善和加强 PHP 的功能，这也是 PHP 能够迅速发展的根本原因。

（3）PHP 易学易用。因为 PHP 3.0 以上版本是用 C 实现的，而且它自身的语法风格同 C 极其相似，有许多的语句、函数 PHP 与 C 是完全相同的，而 C 语言的普及性是不容置疑的，因此 PHP 对于程序员而言非常容易上手。

（4）PHP 还有一个非常重要的特点，那就是 PHP 具有十分强大的数据库操作功能，可直接链接多种数据库，并完全支持 ODBC，这一特点是其他脚本语言所不能比拟的。可支持的数据库，包括常用的 Oracle，mSQL，dBase，Sybase，Informix，MySQL 等等。PHP 的使用者都认为在开发数据库的网站时，使用 PHP 与 Mysql 是最佳组合。

（5）PHP 语言可以嵌入 HTML 中

当使用者使用经典程序设计语言（如 C 或 Pascal）编程时，所有的代码必须编译成一个可执行的文件，然后该可执行文件在运行时，为远程的 Web 浏览器产生可显示的 HTML 标记。但另一方面，PHP 并不需要编译（至少不编译成可执行文件）。使用者可以把自己的代码混合到 HTML 中。例如，下面的代码将显示

Green Apple Data Center

“Hello,world!”，PHP 代码在下面以黑体字显示。

```
<HTML>
    <HEAD><TITLE>Test</TITLE></HEAD>
    <BODY>
        <?PHP$string='world!';?>
        <H1>Hello,<?phpecho$string?></H1>
    </BODY>
</HTML>
```

PHP 应用程序服务器（本书的主要写作目的）是紧密集成到 ApacheWeb 服务器中的，可以在一个程序内同时调用它们两个。当 Web 浏览器请求 PHP Web 页面的时候，Web 服务器的 PHP 部分将被调用进行解释。Web 服务器在请求的 Web 页中寻找<?PHP...?>标记，并按要求执行这些 PHP 代码。

由 PHP 解释后生成的代码将去掉<?PHP...?>标记。例如，当 PHP 代码运行后，以前的 Web 页面将变成如下所示的内容：

```
<HTML>
    <HEAD><TITLE>Test</TITLE></HEAD>
    <BODY>
        <H1>Hello,world!</H1>
    </BODY>
</HTML>
```

注意，所有的 PHP 代码都消失了，仅仅留下了 HTML 语句。而由 PHP 代码生成的 HTML 语句在上例中以黑体的形式显示。

PHP 4.0 是更有效的，更可靠的动态 Web 页开发工具，在大多数情况运行比 PHP 3.0 快，其脚本描述功能更强大并且更复杂，最显著的特征是速率比的增加。PHP 4.0 这些优异的性能是 PHP 脚本引擎重新设计产生的结果：引擎由 AndiGutmans 和 ZeevSuraski 从底层全面重写。PHP 4.0 脚本引擎（Zend 引擎）使用了一种更有效的“编译－执行”模型，而不是 PHP 3.0 采用的“执行－当解析时”模型。

PHP 4.0 在 PHP 3.0 版的基础上增加或增强了许多有用的特征，主要如下：

（1）别名。在 PHP 4.0 中，可以利用引用为变量赋值，这给编程带来了很大的灵活性。

（2）扩充了 API 模块。PHP 4.0 为扩展的 API 模块提供了扩展 PHP 接口模块，它比旧的 API 版本明显地快，PHP 模块已有的及最常用的接口多数被转换到使用这个扩展的接口。

（3）自动资源释放。PHP 4.0 增加了引用计数功能，这种新技术的引入使 PHP4.0 具有了自动内存管理功能，减轻了开发人员的负担。

（4）PHP 4.0 支持布尔类型。

（5）进程生成。在 UNIX 环境下的 PHP 4.0 提供了一个很智能和通用的生成进程，使用了一种名为基于 automake/libtool 的系统生成技术。

（6）COM/DCOM 支持。PHP 4.0 提供 COM/DCOM 支持（仅用于 Windows 环境）可以无缝地存取和访问 COM 对象。

（7）与 PHP 3.0 相容性很好。PHP 4.0 是与 PHP 3.0 代码向后兼容性接近 100%，由于 PHP4 的改进的体系结构，两者有一些细微的差别，但是大多数人将可能永远不可能遇上这种情况。

（8）配置。PHP 4.0 重新设计和增强了 PHP.ini 文件，这使得用 PHP.ini 来配置 PHP 显得极为容易，这个文件可以在运行时被 Apache（UNIX 系统）或由 Windows 注册（Windows 环境）。

（9）加密支持。PHP 4.0 实现了完整的加密，这些加密功能是一个完整的 mycrypt 库，并且 PHP4.0 支持

Green Apple Data Center

哈希函数，Blowfish，TripleDES，MD5，并且 SHA1 也是可使用的加密算法。

（10）类型检查。PHP 4.0 支持同一操作符，用于类型检查：“ === ”（3 等号运算符），为在两个值和其类型之间作检查。例如，“ 3 ” ===3 将视为假（类型是不同的），而 “ 3 ” ==3（相等判断）将视为真。

（11）FTP 支持。PHP 4.0 支持 FTP，通常会为通过一个调制解调器链接下载一个大文件提供一个网络接口。

（12）PHP 4.0 新增函数或功能增强函数。PHP 4.0 新增了许多函数，同时也将许多现有的函数功能进行了增强。

（13）“ Here ” 打印。PHP 4.0 的 “ Here ” 打印是与 Perl 类似的，尽管完全不相同，“ Here ” 是打印大容量文章的一个有用的方法，例如在 HTML 文件中，不会漏掉任何一个字符，例如目录标记。

（14）HTTPSessionfallback 系统。为 HTTPSession 管理的一个 fallback 系统在 PHP 4.0 被实现，缺省情况下，Session 标识符由 cookies 存储，如果没有 cookies 支持或一项 cookies 任务失败，Session 标识符自动被创建并在 URL 的查询字符串中被携带。

（15）ISAPI 支持。PHP 4.0 能作为一个个性化的 ISAPI 模块作为 IIS 插件，这比 PHP 3.0 更有效，它作为 CGI 运行（一个外部的程序）。

（16）内存。PHP 4.0 能更有效的使用内存，导致较少的内存占用消耗，这主要归功于引用计数技术的实现。

1.5 PHP 与其他 CGI 的比较

写 CGI 的方式有很多种，PHP 只是其中的一种选择，下面表格比较它们之间的特点。

表 1-1 PHP 与其他 CGI 的比较

	PHP	ASP	CGI	NSAPI	ISAPI
操作系统	均可	Win32	均可	均可	Win32
Web 服务器	数种	IIS	均可	NetscapeServer	IIS
执行效率	快	快	慢	极快	极快
稳定性	佳	中等	最高	差	差
开发时间	短	短	中等	长	长
修改时间	短	短	中等	长	长
程序语言	PHP	VB	不限	C/C++	C/Delphi
网页结合	佳	佳	差	差	差
学习门槛	低	低	高	极高	高
函数支持	多	少	不定	中等	少
系统安全	佳	极差	最佳	佳	尚可
使用网站	超多	多	多	极少	少
改版速度	快	慢	无	慢	慢

其中的 PHP 可用在数种 Web 服务器上；传统 CGI 就不限是哪种操作系统或 Web 服务器平台；NSAPI 一定要在 Netscape 的服务器（如 NetscapeEnterpriseServer 或 FastTrackServer）上才可以执行，但可支持多种操作系统（UNIX 或 Win32）；ASP 及 ISAPI 只在 IIS 上有完整的功能。

在稳定性上，由于 NSAPI 或 ISAPI 是动态链接的方式，因此在执行过程中若出现问题，会使得 Web 服务器一起瘫痪。而 ASP 在笔者实际应用经验上，隔阵子就会使系统不稳定，需要重新启动操作系统。PHP 在许多网站上使用，不但长期使用都没有问题，而且程序的稳定性也不错。

在开发及维护时间上，PHP 及 ASP 都表现不错。要比较和网页结合的能力，PHP 和 ASP 是并驾齐驱的，

Green Apple Data Center

其它的方式就不能内嵌 HTML 语法了。而这也是影响开发时间的因素之一。

就系统安全性而言，ASP 是最差的，在没有经过微软的 IISServicePack 处理时，使用::$DATA 就可以看到 ASP 的源代码，这真是叫人不敢领教。当然，传统 CGI 的程序，由于是由操作系统直接管理，要破解的难度最高，黑客必须由操作系统下手，而不能由 Web 服务器下手。PHP 在许多商业及非商业使用时，也没有听过有什么安全的问题。

在新增功能及改版方面，PHP 是最有活力的，数天至数周就有一个新版本出现，每次的新版，就代表更多的功能及修正更多的错误。

Green Apple Data Center

第 2 章　PHP 的安装和设置

本章要点：

- PHP 在 Linux/Unix 下的安装设置
- PHP 在 Windows 下的安装设置

2.1　系统需求及其准备工作

在安装 PHP 做为 WWW 服务器的一部分时，可以考虑用 Unix 操作系统；或者是 Windows NT/98/2000 等 Win32 API 的平台。当然，大部分的人都会使用 Unix 来当作 PHP 的执行平台（在 Windows NT/2000 的使用者大多数都会选择 IIS+ASP），实际上，Linux+Apache+PHP 应是最经济的选择，因为这样的组合几乎是不用钱的，Apache 服务器则是目前最多 WWW 网站所采用的服务器。可以至 http://www.apache.org 下载最新版的程序及相关文件，若觉得从国外下载要很久的话，国内很多著名站点都有下载。

Linux 操作系统方面，可以选择各式的 Linux 套件，包括 RedHat、Slackware Linux、Open Linux、Turbo Linux 还有中文版的 BluePoint、红旗 Linux 等版本，反正这方面的软件很容易而且很便宜就可以买到，也可以去各大 FTP 站下载完整的系统安装。现在的很多版本，比如 RedHat7.0、BluePoint2.0 在安装系统时就已经把 PHP、Apache 等组件安装好了。

PHP 则可以去它的官方网站 http://www.php.net 下载所需要的程序，如图 2-1 所示。

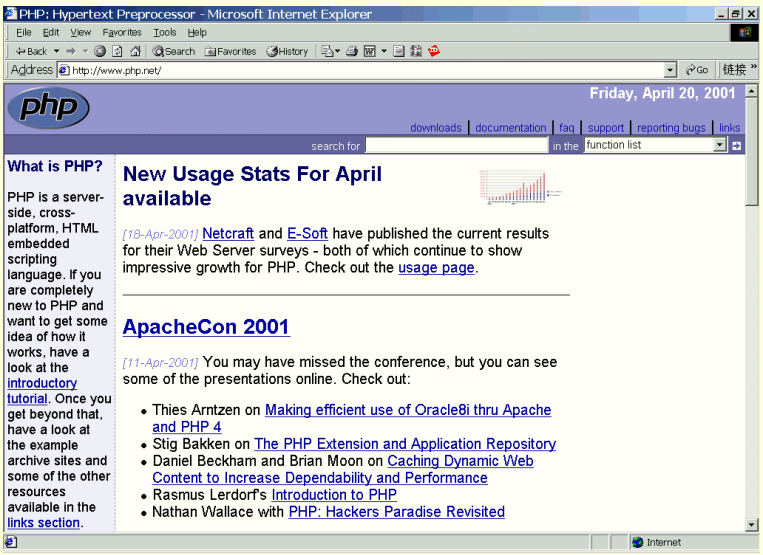


图 2-1　PHP 官方网站首页

虽然目前 WindowsNT 或者 Windows2000 等 Win32 的操作平台也能安装 PHP 及 Apache 服务器，不过 PHP 和 Apache 在 Unix 下可以跑得更快更好。但是 Windows 系列操作系统是用户最熟悉、使用最多、工具软件最丰富的操作系统，在制作网页时，有优秀的 FrontPage、Dreamveaver 等可视化软件，所以很多程序员都是在 Windows 下面开发 PHP 程序，再拿到 Unix 或者 Linux 下运行，所以本书所有的程序都是 Windows 下开发的，由于 PHP 本身就是跨平台语言，几乎不需要任何改动就可以在 Unix 或者 Linux 下运行。

当然，若想使用商业化的操作平台，SUN、IBM、HP、DEC、SGI、NEC 等公司都有提供相关的 Unix 或者是 WindowsNT 的操作平台。加上高安全性调整过后的 Apache 服务器，Stronghold 或是其它支援 SSL 的 Apache 版本。这种组合，相信能满足商业化的需求。而 PHP 就扮演着快速方便的 CGI 角色，让客户对站点的服务品质更加满意。

Green Apple Data Center

2.2 在 Unix/Linux 下的安装和设置

2.2.1 基本命令

在 Unix/Linux 下的安装比较复杂，需要读者熟悉 Unix/Linux 系统，也许有的读者以前从没有编译过 Unix/Linux 应用程序，在此介绍一些基本命令，以便能够顺利安装 PHP。

（1）tar

tar 即 tapearchiver，它可以把几个文件组合成一个文件，并可以选择是否进行压缩。这个命令过去通常用来进行备份，以便使数据存储在磁带中。当 tar 文件被压缩时，它们有一个.gz 的后缀；当 tar 文件没有压缩时，它们有一个.tar 的后缀。

（2）gcc

gcc 是 GNU 的 C 编译器。它的工作是把人可以看懂的源代码文件编译成机器可以读懂的目标文件。C 源文件通常有.c 的后缀名，目标文件通常有.o 的后缀名。如果编译工作不能正常进行，就是碰到了一个编译期的错误，或者说：语法错误。在大多数情况下，不彻底的编译通常是编译器找不到一个或几个包含文件而产生的。包含文件都有一个.h 的扩展名，通常用来定义不同的系统信息，以及将多个不同的.c 文件所共有的信息收集在一起。

（3）make

make 是一个常用的工具程序，是用来进行帮助编译的。它的工作是用来只编译那些还没编译的.c 文件，编译后将生成一个.o 文件，如果.c 文件比.o 文件新，也就是说，到上次编译之前源文件被编辑过，那么 make 将会重新编译.c 文件。make 指令一般是寻找一个 Make file 文件，在这个文件中包含有一个或多个能执行的目标，例如，make clean 会告诉 make 执行清除目标。

（4）ld

ld 是 GNU 的链接程序。它的工作是把所有的目标文件和库链接起来，创建一个单一的可执行文件。幸运的是，基本上不用手工运行这个程序，因为 Make file 将会考虑到所有的编译细节。

（5）ldconfig

ldconfig 会在多个库目录（在/etc/ld.so.conf 中指定）中寻找共享库。共享库常被多个应用程序使用，它们的文件名中的某个地方有.so，例如，libqt.so.1.42 是一个共享库。在编译完毕之后，有可能需要在/etc/ld.so.conf 文件中增加一个目录，并且运行 ldconfig-v 命令。

（6）./configure

configure 将会在计算机中寻找一些关键信息，例如，安装的是哪一种 C 编译器、包含文件在哪里等等。然后，configure 将会按照所用的计算机配置重新修改 Make file 文件。应该使用./configure 在当前目录下运行程序，以避免偶然运行$PATH 环境变量中的目录下的其他程序。

2.2.2 安装 Apache Web 服务器和 PHP

在安装前还是要先下载 apache_1.3.x.tar.gz 和 php-4.0.x.tar.gz 的安装程序，这里是以 apache_1.3.12.tar.gz 和 php-4_0.x.tar.gz。

（1）tar xvfz apache_1.3.12.tar.gz 解压缩 apache 安装程序。

（2）tar xvf php-4_0.x.tar.gz 解压缩 php4bata1 安装程序。

（3）cd ../apache_1.3.12 切换到 apache 目录。

（4）./configure 进行编译前的配置,默认安装路径为/usr/local/apache)。

（5）cd ../php-4.0.x 切换到 php4 目录。

（6）./configure--with-apache=/root/apache_1.3.12--enable-track-vars 配置 php4 成 apache module 型式，--enable-track-vars 预设 PHP4 启动 GET/POST/Cookie 的功能，如有需要使用其他的参数请使用./configure--help 观看其说明。

（7）make 编译 PHP4。

Green Apple Data Center

（8）make install 安装 PHP4。

（9）cd ../apache_1.3.12 切换回 apache 目录。

（10）./configure--activate-module=src/modules/php4/libphp4.a 设定 apache 模块启动 php4 的模块。（注意：在 apache module php4 路径下并不会有 libphp4.a 这个档案，这是正常的，libphp4.a 在之后会被 apache 创造出来。）

（11）make 编译 apache。

（12）make install 安装 apache。

（13）如果需要的话，将执行中的 http server 停止，并将/usr/local/apache/bin/下的 httpd daemon 拷贝至原本 httpd daemon 的路径，将旧的 httpd daemon 更换成新版，可使用 httpd-l 检查是否将 PHP4 编辑进入 apacheserver，再重新启动 httpserver。

（14）cd ../php-4.0.x 切换回 PHP4 路径。

（15）cp php.ini-dist /usr/local/lib/php.ini 将 php.ini-dist 拷贝至/usr/local/lib/php.ini。

（16）编辑 httpd.conf 或 srm.conf 档案，增加下面几行：

```
AddType application/x-httpd-php.php
AddType application/x-httpd-php.php3
AddType application/x-httpd-php.php4
```

（17）重新启动 apache server。

在细节选项上，除了上述的安装外，也可以在编译时加入其它的选项。

Apache 模块

语法:--with-apache=DIR

说明:用本选项可以让 PHP 以 Apache 的模块方式使用，DIR 的字串可以是/usr/local/apache 或其它安装 Apache 的目录。

例如:--with-apache=/var/lib/apache

fhttpd 服务器模块

语法:--with-fhttpd=DIR

说明:若使用 fhttpd 服务器，可以使用本指令编译 PHP。用模块的方式配合 fhttpd 服务器，可以有较好的功能。

Adabas D 数据库

语法:--with-adabas=DIR

说明:数据库系统为 Adabas D 数据库时需要加本选项。关于 Adabas D 数据库的细节，可以参考 http://www.adabas.com。

例如:--with-adabas=/usr/local/adabasd

dBase 数据库

语法:--with-dbase

说明:只要加本选项，不用其它的参数或函数库，PHP 就会让系统有存取 dBase 数据库的功能。

filePro 数据库

语法:--with-filepro

说明:不用指定数据库路径及其它函数库等，可以读取 filePro 数据库（只读）。

mSQL 数据库

语法:--with-msql=DIR

说明:提供存取 mSQL 数据库。更多的细节请参考 mSQL 的网站 http://www.hughes. com.au。

例如:--with-msql=/usr/local/Hughes

MySQL 数据库

语法:--with-mysql=DIR

说明:提供存取 MySQL 数据库。更多的细节请参考 MySQL 的网站 http://www.tcx.se。

例如:--with-mysql=/usr/local/mysql

iODBC 数据库支持

Green Apple Data Center

语法:--with-iodbc=DIR

说明:**提供** ODBC **数据库支持,用来存取后端数据库。更多的细节请参考** iODBC **的网站** http://www.iodbc.org。

例如:--with-iodbc=/usr/local/iodbc

OpenLink ODBC 数据库支持

语法:--with-openlink=DIR

说明:**使用** OpenLink ODBC **数据库支持，用来存取后端数据库。更多的细节请参考** OpenLink ODBC **的网站** http://www.openlinksw.com。

例如:--with-openlink=/usr/local/openlink

Oracle 数据库

语法:--with-oracle=DIR

说明:**使用** Oracle **数据库。** Oracle **的版本要在** 7.3 **版以上。也可以在** PHP **程序中使用环境变量** ORACLE_HOME **来指定** Oracle **的路径。更多有关** Oracle **的信息请参考** Oracle **的网站** http://www.oracle.com。

例如:--with-oracle=/export/app/oracle/product/7.3.2

Postgre SQL 数据库

语法:--with-pgsql=DIR

说明:**使用** Postgre SQL **数据库。更多有关** Postgre SQL **的细节请参考** Postgre SQL **的网站** http://www.postgreSQL.org。

例如:--with-pgsql=/usr/local/pgsql

Solid 数据库

语法:--with-solid=DIR

说明:**使用** Solid **数据库。更多有关** Solid **的细节请参考** Solid **的网站** http://www.solidtech.com。

例如:--with-solid=/usr/local/solid

Sybase 数据库

语法:--with-sybase=DIR

说明:**使用** Sybase **数据库。更多有关** Sybase **的细节请参考** Sybase **的网站** http://www.sybase.com。

例如:--with-sybase=/home/sybase

Sybase-CT 数据库

语法:--with-sybase-ct=DIR

说明:**使用** Sybase-CT **数据库。**

例如:--with-sybase-ct=/home/sybase

Velocis 数据库

语法:--with-velocis=DIR

说明:**使用** Velocis **数据库。有关** Velocis **数据库的进一步细节请参考** Raima **公司的网站** http://www.raima.com。

例如:--with-velocis=/usr/local/velocis

自订 ODBC 数据库驱动程序

语法:--with-custom-odbc=DIR

说明:**使用自订的** ODBC **函数库。当然，在使用时要指定** CUSTOM_ODBC_LIBS **及** CFLAGS **变量。例如在** QNX **机器上使用** Sybase SQL Anywhere **时可能要设定系统环境变量** CFLAGS=-DODBC_QNX、LDFLAGS=-lunix **及** CUSTOM_ODBC_LIBS="-ldblib-lodbc"，**并要在** PHP **设定加入**--with-custom-odbc=/usr/lib/sqlany50。

例如:--with-custom-odbc=/usr/local/odbc

不使用 ODBC 数据库驱动程序

语法:--disable-unified-odbc

说明:**使用本选项将使所有的** ODBC **数据库驱动程序不起作用。本选项不用指定路径，而受本选项影响的选项有**--with-iodbc、--with-solid、--with-adabas、--with-velocis **及**--with-custom-odbc。

LDAP 目录协定

语法:--with-ldap=DIR

Green Apple Data Center

说明:若要使用目录协定（Lightweight Directory Access Protocol,LDAP）**则必须要打开本选项。有关** LDAP **的细节，可以参考** RFC **文件的** RFC1777 **及** RFC1778**。**

例如:--with-ldap=/usr/local/ldap.

mcrypt 编码函数库

语法:--with-mcrypt=DIR

说明:当安装了 mcrypt **函数库后，可在编译** PHP **时加入本选项，让程序可以使用编解码功能。**

例如:--with-mcrypt=/usr/local/include

SysV 信号

语法:--enable-sysvsem

说明:要使用 SysV **的信号**（semaphores）**机制，则要打开本选项。**

XML 支持

语法:--with-xml

说明:打开本选项可以支持 James Clark's **写的** XML **解析程序库。**

维护模式

语法:--enable-maintainer-mode

说明:本选项一般不会打开，除非是 PHP **开发人员比较有用。**

常规表示程序库

语法:--with-system-regex

说明:若需要额外的常规表示功能，可以加入本选项。

PHP 设定

语法:--with-config-file-path=DIR

说明:用来指定 php3.ini **或** php4.ini **的路径，供** PHP **初始化时使用。**

例如:--with-config-file-path=/usr/local/lib

PHP 执行路径

语法:--with-exec-dir=DIR

说明:有时为了系统的安全性考虑，会指定 PHP **程序一定要在哪个目录执行。**

例如:--with-exec-dir=/usr/local/bin

调试模式

语法:--enable-debug

说明:本选项一般不会使用，除非在开发 PHP **程序时比较有用。它可以显示额外的错误信息。**

安全模式

语法:--enable-safe-mode

说明:内定值是打开的，可以对系统安全提供比较多的保护。

变量追踪

语法:--enable-track-vars

说明:让 PHP **能追踪** HTTP_GET_VARS、HTTP_POST_VARS **及** HTTP_COOKIE_VARS **三个变数，一般是打开的。**

自动加引入字符

语法:--enable-magic-quotes

说明:可让程序在执行时自动加入反斜线的引入字符。

开启调试器

语法:--enable-debugger

说明:打开内建的 PHP **调试器。**

取消路径（**discard path**）

语法:--enable-discard-path

Green Apple Data Center

说明:打开这个选项，用户就不能透过浏览器读取.htaccess **等和系统安全相关的档案。**

高精确度数学函数

语法:--enable-bcmath

说明:打开高精确度函数。必须要先安装本函数库，本选项方有效。

强迫 CGI **重新引导**

语法:--enable-force-cgi-redirect

例如:**若使用** CGI VERSION **模式来进行** PHP **的设置，打开本选项会增加安全性。例如用户读** http://my.host/cgi-bin/php/secret/doc.html **遇到比较了解** PHP **系统的黑客级用户可能会自已输入以下网址** http://my.host/secret/doc.html **来读取相关信息。若** PHP **和** Apache **编译在一起，让** PHP **变成** Apache **的一部分，则不需要加入本选项。**

不使用短的标记

语法:--disable-short-tags

说明:设定本选项后，PHP **的程序就不能使用短的**<? .... ?>**标记，一定要用**<?php.....?>**的长标记。**

引入远程文件

语法:--enable-url-includes

说明:设定本选项可让 PHP **程序可以引入（**include**）远端的** HTTP **或** FTP **服务器中的文件。**

关闭语法效果

语法:--disable-syntax-hl

说明:使用本选项会关闭 PHP **语法的彩色效果。**

函数库路径

语法:CPPFLAGS=-IDIR **及** LDFLAGS=-LDIR

说明:若 PHP **在安全或编译所需的函数库在特别的路径，需要加入本选项，**LDFLAGS **表示函数库的路径，**CPPFLAGS **表示标头档的路径。**

例如:LDFLAGS=-L/my/lib/dirCPPFLAGS=-I/my/include/dir./configure

2.3 在 Windows 2000 下的安装和设置

由于很多朋友都只有一台计算机而且对 Windows **比较熟悉,要用一台计算机在** Windows **上同时完成服务器和客户机的功能，这样可以用如下的方案：**

（1）**操作系统**:Microsoft Windows 2000 Professional **简体中文版**

（2）Web **服务器**:Apache 1.3.12 for WIN32

（3）PHP4:PHP 4.0.0 for WIN32

（4）ZEND:Zend Optimizer beta4 for PHP4.0.0 for WIN32

事实上 Web **服务器可以选择** IIS **等其他** Web **服务器，但是考虑到** PHP **天生就和** Apache **是最佳组合，而且有些功能只有** Apache **服务器才能用到，所以建议选用** Apache **服务器。**

2.3.1 安装和配置 PHP

安装 PHP **的过程相当简单，下载最新的** Windows **版的** PHP**（当前最新的稳定版本是** 4.04**），将下载下来的文件解压缩到指定的目录中，如** C:\PHP4 **就可以了。**

配置 PHP **的过程也比较简单，与** PHP **一起分发的文件包里有一个** Readme **文件，里面讲述了如何配置。有一个文件名为** PHP.INI-dist**，它是** PHP **的主要配置文件。只需要将它拷贝到** Windows **系统目录（如果系统安装在** C**：盘，则就是** c**：**Winnt **目录下），并且改名为** PHP.INI**。需要对此文件作适当的修改，最主要的事情是加入一些可能要用到的模块，比如支持** MySQL **数据库等。用文本编辑器打开** PHP.INI**，在里面找到如下的内容：**

;;;;;;;;;;;;;;;;;;;;;;

；Dynamic Extensions；

;;;;;;;;;;;;;;;;;;;;;;

；if you wish to have an extension loaded automaticly,usethe

Green Apple Data Center

```
；following syntax:extension=modulename.extension
；for example,on windows,
；extension=msql.dll；or under Unix,
；extension=msql.so
；Note that it should be the name of the module only,
；no directory information needs to go here.
；Specify the location of the extension with the
；extension_dir directive above.
```

接下来几行被注释的，如

```
；Windows Extensions
；extension=php_mysql.dll
；extension=php_nsmail.dll
；extension=php_calendar.dll
；extension=php_dbase.dll
；extension=php_filepro.dll
...
```

要做的就是将一些行的注释去掉，即删除打头的分号。需要 MySQL **的支持，就从**"extension=php3_mysql.dll" **一行中去掉了分号**"；"**，在特定情况下，要用的** DLL **文件不在这个列表中，只需要简单地加上一行，如：**"extension=mydll.dll"**。这样，**PHP **就配置好了。**

2.3.2 安装 Apache Web 服务器

（1）运行 apache_1_3_12_win32.exe**，解压缩完毕后将进入如图** 2-2 **所示的安装开始界面，单击“** Next **”按钮进入如图** 2-3 **所示的** Apache **安装软件协议界面。**

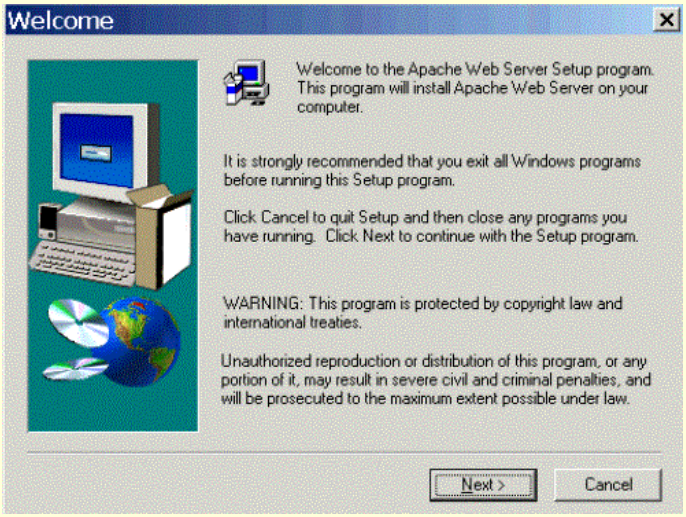


图 2-2 Apache **安装开始界面**

Green Apple Data Center

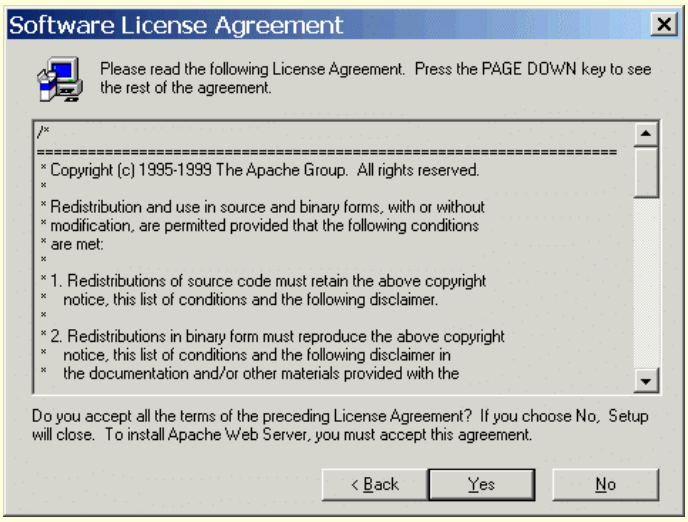


图 2-3 Apache 安装协议界面

（2）在这个界面中，是安装 Apache 服务器必须遵守的一些协议，这里当然接受这些协议，单击“Yes”按钮进入如图 2-4 所示的信息界面。

（3）这个窗口界面主要是给出一些警告信息，单击“Next”按钮进入如图 2-5 所示的选择安装路径界面。

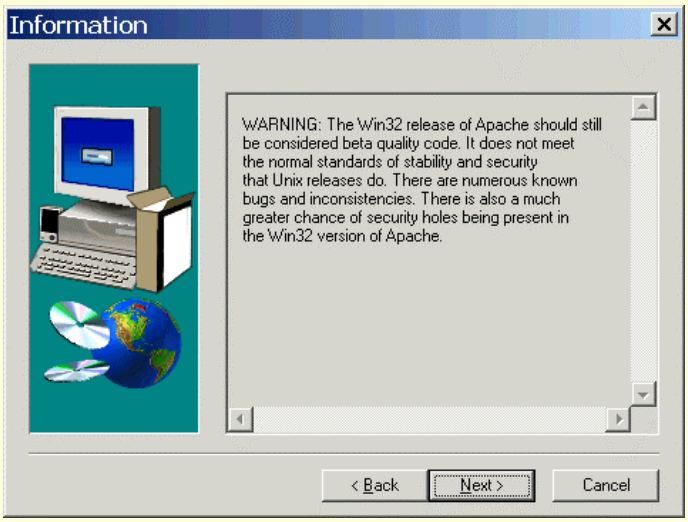


图 2-4 Apache 安装信息界面

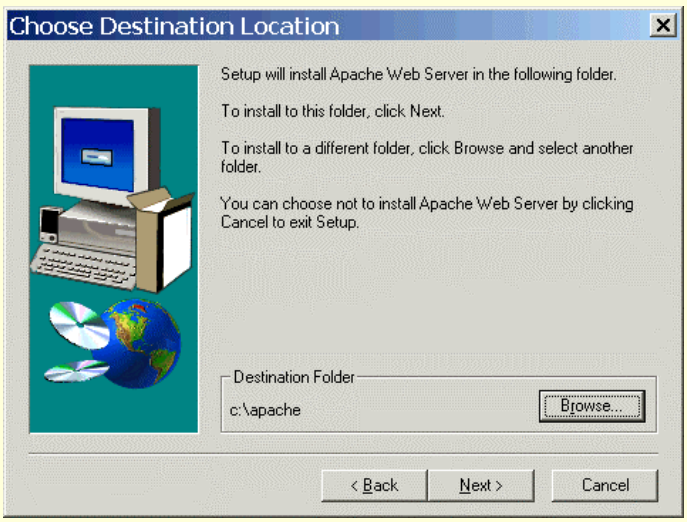


图 2-5 Apache 安装择路径选择界面

Green Apple Data Center

（4）在这个窗口选择 Apache 服务器的安装路径，用户可以把它安装到计算机的硬盘中，这里设定安装在 c:\Apache 目录中。选择好安装路径后，单击“ Next ”按钮进入如图 2-6 所示的选择安装类型界面窗口。

（5）在安装类型界面窗口中，有三种安装方式：Typical（典型安装）、Compact（完全安装）、Custom（自定义安装），通常选用典型安装方式。选择完毕后单击“ Next ”进入如图 2-7 所示的“ Set Program Folder ”界面窗口。

（6）这个窗口主要是在 Windows“ 开始 ”－“ 程序 ”菜单中设置名字，通常不用改变，单击“ Next ”按钮，将复制文件到计算机硬盘上。复制文件完毕后，将进入如图 2-8 所示的“ 安装完毕 ”窗口，单击“ Finish ”按钮完成安装过程。

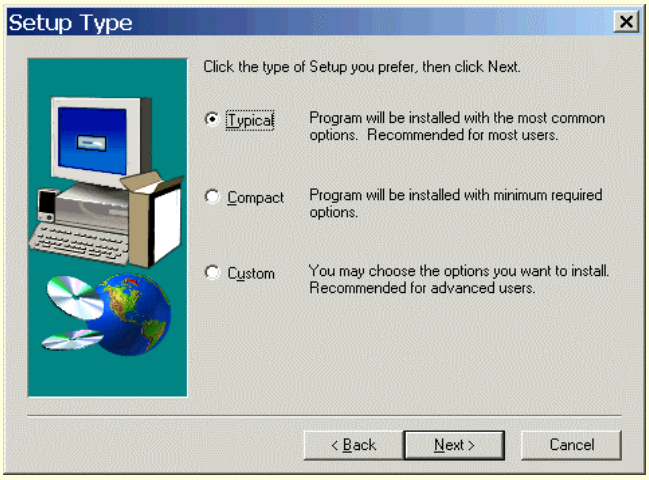


图 2-6　安装类型界面窗口

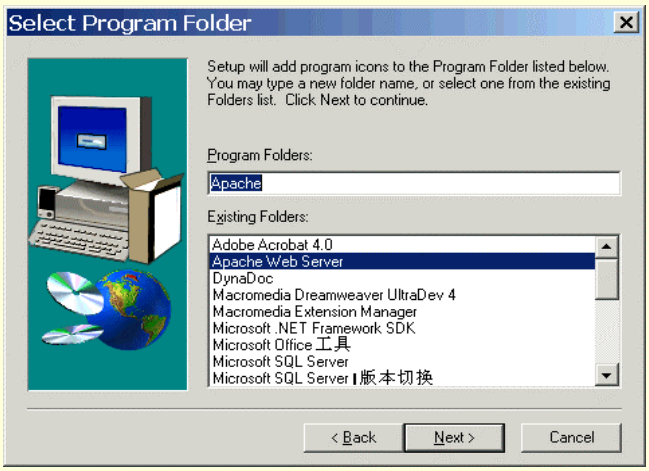


图 2-7　“ Set Program Folder ”界面窗口

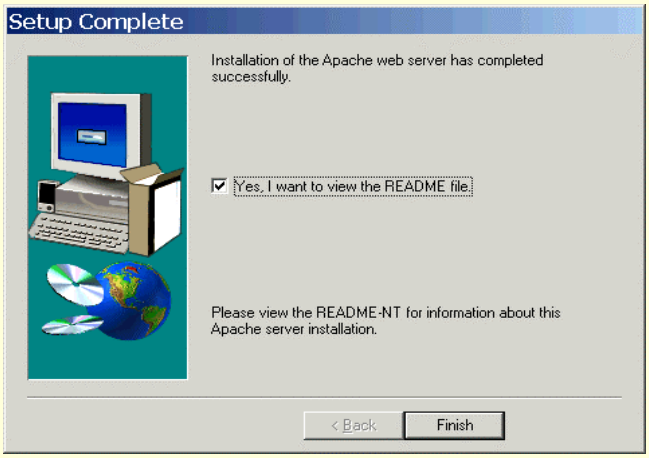


图 2-8　Apache 安装完毕界面窗口

Green Apple Data Center

2.3.3 设置 Apache Web 服务器

在上面安装 Apache 的目录，用文件编辑器（最好是支持 Win32 长文件名格式的，如 Edit,Ultraedit 等）打开.\conf\httpd.conf 文件，在本例中是 c:\Apache\conf\httpd.conf，如图 2-9 所示，这个是 APACHE 的最主要的配置文件，不要轻易修改，除非有特定把握，这里面的配置选项非常多，如果要详细了解 Apache 服务器，可去参考一下其他 Apache 技术的书籍。这里只介绍 Apache 概要说明以及与 PHP 一起工作的一些配置。

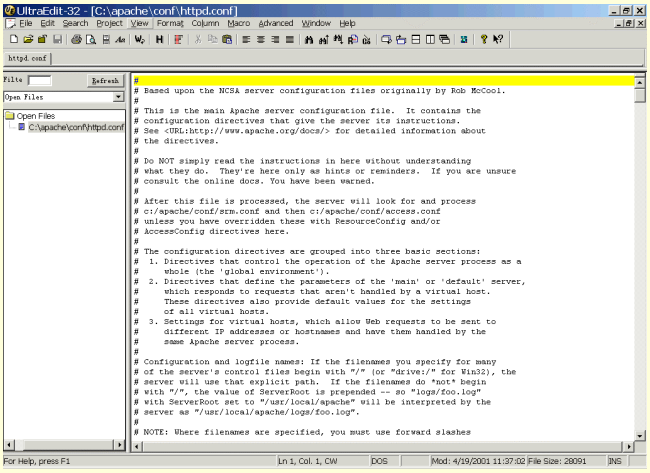


图 2-9 httpd.conf 文件内容

1. Apache 概要说明

通常至少需要修改的是 ServerName 这一项，把注释符号“#”去掉，最简单是后面跟 IP 地址，对于有 WINS 的局域网用户可以用机器名，对于有 DNS 的网络用户可以用域名。在 Windows2000 下没有设置 ServerName 这一项会造成 Apache 不能启动。还有一个影响是当用如下方式使用 Alias 时：Alias/userdir"c:/userdir/"假如这样调用：

http://localhost/userdir/不会产生任何问题，可是如果这样调用：

http://localhost/userdir 那么 localhost 会被替换成 ServerName 所指定的值，如果这个值不能被正确解析，就无法访问。用户的配置项目添加在 httpd.conf 或者 srm.conf 中都可以，一说是将所有的配置项目都放在 httpd.conf 中，这样只需要修改维护这一个配置文件，比较方便，而且避免了冲突（这是 srm.conf 文件中说的！）。可是 httpd.conf 确实太大了，要在里面搜索特定的项目就不很容易。还有一说是将所有的用户自己添加修改的配置项目都放在 srm.conf 中，这个文件很小，便于察看修改，本人倾向于后者。在 httpd.conf 中，Apache 其他常用的可能需要修改的设置有：ServerAdmin your@email.address#管理员的 e-mail 地址

DocumentRoot "c:/apache/htdocs"# 默认的根路径。注意路径全都是用斜线而不是反斜线来分隔。DirectoryIndex index.html index.php index.php4 index.php3 index.htm index.shtml 这一项指明了在每个目录中的默认文档及其顺序。

AccessFileName.htaccess#目录访问的配置文件名用资源管理器会发现无法将一个文件改名为“.htaccess”，不过在命令行方式下可以，用 notepad 也可以另存为这个文件名。这样就可以用一台计算机进行程序调试，而不必进入 XWindows。

2. Apache 设置步骤

（1）修改 c:\apache\httpd.conf，去掉 ServerName 前的“#”号，后面跟本机的 IP 地址或者域名。作者的机器的域名是：duoduo.it.tsinghua.edu.cn，以后在浏览器里敲入地址 http://duoduo.it.tsinghua.edu.cn 就可以访问 ApacheWeb 服务器了。这一行必须修改，否则在 Windows 2000 下不能正常启动 Apache。

（2）要让 Apache 与 PHP 一起工作，只要在 httpd.conf 文件中加入以下几行即可：

```
ScriptAlias/php4/"C:/PHP4/"
AddType application/x-httpd-php3.php
AddType application/x-httpd-php3.php3
```

Green Apple Data Center

```
AddType application/x-httpd-php3.php4
AddType application/x-httpd-php3.phtml
Action application/x-httpd-php3"/php4/php.exe"
```

需要注意的是上面的这几行不能写错。其中第一行的最后一部分是安装的 PHP 的目录（比如 C:/PHP4/），中间的四行，是 Apache 服务器所能识别的后缀名，经常上网的细心读者可能会发现有些网站网页文件名是以．php 为后缀名，而有些则是以．php3 或者其他名称为后缀名，正是由于这里的设置。应该注意的是这几行配置命令都区分大小写。

（3）运行 c:\apache\apache-i 将 Apache 安装成为 Windows 2000 的一个服务器。

（4）运行 net start apache 启动 Apache。

（5）将 c:\apache\htdocs\index.html.en 改名为 index.html，然后在浏览器地址栏中输入“ http://localhost ”，如果能够在 Web 浏览器中看到如图 2-10 所示的内容时，表明 Apache Web 服务器安装成功。

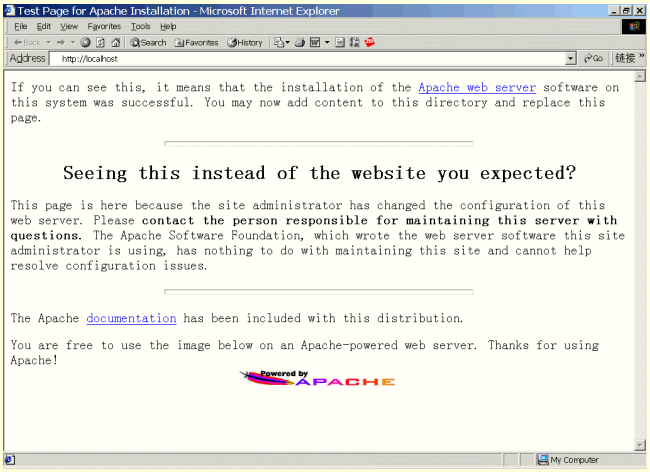


图 2-10　index.html 页面内容

3．测试 PHP 和 Apache 系统

如果 Apache 服务器工作正常，接下来，继续测试 PHP 文件。用文本编辑器，在 Apache 安装目录下的 htdocs 子目录中创建一个名为 test.php 的文件，内容如下：

```
<?
phpinfo();
?>
```

如果看不懂这 3 行代码，没有关系，后面的章节将详细介绍这些代码的含义，只要一步一步的学习，很快就会成为 PHP 高手。再打开浏览器，输入如下地址：http://localhost/test.php。如果前面的配置没有问题的话，将会看到一张如图 2-11 所示很长的表格，里面有 PHP 的各种配置信息及相关的环境变量，包含 Apache 的有关信息。如果它未工作，请检查 httpd.conf 的设置是否正确，这很重要。如果未设置好，Apache 将不知道如何去处理后缀为 php 的文件。

Green Apple Data Center

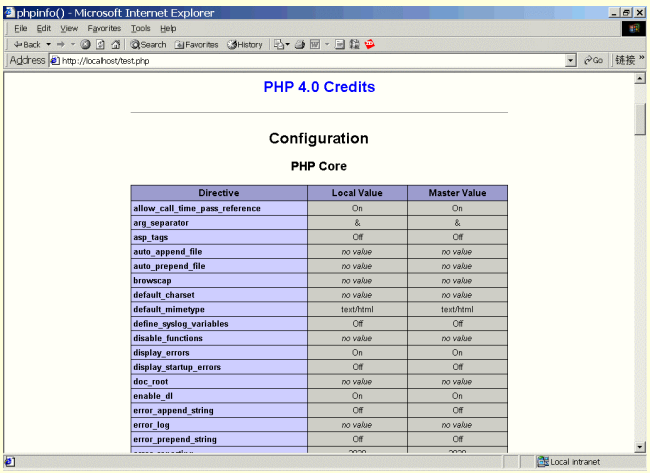


图 2-11　test.php **的文件执行结果**

2.3.4　Zend Optimizer for PHP 4 **快速安装**

（1**）**ZendOptimizer-Beta4-WindowsNT.zip **解压缩到一个目录下，例如** c:\zend

（2**）修改** Windows **根目录下** tphp.ini **文件，在任何地方加入如下两行：**

```
zend_optimizer.optimization_level=7
zend_extension_ts="c:\apache\zend\ZendOptimizer.dll"
```

（3**）浏览器中刷新** test.php**，可以看到多了这么一行：**with Zend Optimizer v0.98,Copyright (c) 1998-2000,by Zend Technologies **完全按照以上步骤进行，就会在不出错的情况下拥有** Windows 2000＋Apache＋PHP4 **的运行环境了！**

2.4　PHP **的使用配置**

当 PHP **启动时将读取配置文件（**php.ini**），作为** PHP **的服务器模块版本，配置文件只在网络服务器启动时读取一次；作为** CGI **版本每次调用时都会读取该文件。下面介绍该文件的使用配置。**

2.4.1　**常用配置语句**

*auto_append_file*string

确定主文件之后自动解析的文件名，如果这个文件被 include 函数调用它就会被包含在内，include_path **也会被使用。所有特殊值都不会关闭文件搜索路径的自动设置功能。**

☞ **注意**

如果脚本被 exit **中断就不会进行文件搜索路径的自动设置。**

*auto_prepend_file*string

确定主文件之前自动解析的文件名，如果这个文件被 include 函数调用，它就会被包含在内，include_path **也会被使用。所有特殊值都不会关闭文件搜索路径的自动设置功能。**

*cgi_ext*string

display_errors boolean

用于确定错误信息是否作为 HTML **输出的一部分打印在屏幕上。**

doc_root string

服务器上 PHP **的根目录仅用于非空的情况下，如果** PHP **是按安全模式配置的，那么该路径之外的文件都不能使用。**

Green Apple Data Center

engine boolean

该语句只在 PHP 的 Apache 版本中起作用，它通常适用于针对每一个目录或虚拟服务器，设置 PHP 解析开启或关闭的站点，通过设置 httpd.conf 文件中适当位置上的 php_engineoff 值可以开启或关闭 PHP。

error_log string

用于记录脚本错误的文件名。如果使用特殊值 syslog，错误信息将被传送到系统记录器，在 Unix 中这意味着 syslog(3)而在 Windows NT 中这意味着记录过程,Windows95/98 不支持系统记录器。

error_reporting integer

设置错误报告级别。参数是一个整数，把下表中的想要报告错误级别的 bit 值相加起来就得到错误报告级别。

表 2-1 错误报告级别

Bit 值	错误类型
1	正常错误
2	正常警告
3	解析错误
4	非严重型相关警告

该语句的缺省值是 7（1＋2＋4），显示正常错误、正常警告和解析错误。

open_basedir string

限制 PHP 可以在特定目录树中打开的文件。

当脚本企图用 fopen 或 gzopen 等命令打开文件时，系统就会检测文件的位置，当文件位于特定的目录树之外时 PHP 将拒绝打开文件，所有的符号链接都经过了分解，所以不可能用 symlink 来回避，这种限制特殊值表明脚本所保存的路径，将被作为基本路径。在 Windows 中，用分号分隔各个目录，在其他所有系统中，目录是用冒号分隔的。作为 Apache 模块，缺省状态是允许打开所有文件。

设置 GET/POST/COOKIE 的顺序为可变解析该语句的，缺省设置是 GPC。如果将其设为 GP 将会导致 PHP 完全忽略 cookies，并且覆盖所有与 POST 方法变量同名的 GET 方法变量。

ignore_user_abort string

缺省状态下该选项值为 OFF，如果改为 ON，即使远程客户机中途断开脚本也会完全运行 ignore_user_abort。

include_path string

定义一列供 require include 和 fopen_with_path 函数查找文件的路径。这一格式就类似于系统的 PATH 环境变量，Unix 系统中是用冒号分隔的一列目录；Windows 中是用分号分隔的一列目录。

isapi_ext string

log_errors boolean

确定是否要将脚本的错误信息记录到服务器的错误日志中，该选项专用于服务器。

magic_quotes_gpc boolean

将 magic_quotes 状态设置为 GPC（Get/Post/Cookie）语句当 magic_quotes 处于 ON 状态时所有的（’）单引号、（”）双引号、（\）反斜杠及 NUL 都会以斜杠自动溢出，如果 magic_quotes_sybase 也是开启的，单引号将会以单引号溢出，而不是以反斜杠溢出。

magic_quotes_runtime boolean

如果 magic_quotes_runtime 是可行的，从外部数据源包括数据库和文本文件返回数据的大部分函数的引号都会以反斜杠溢出。

magic_quotes_sybase boolean

当 magic_quotes_gpc 或 magic_quotes_runtime 处于开启状态时，如果 *magic_quotes_ sybase* 也处于开启状态，单引号将会以单引号溢出，而不是以反斜杠溢出。

max_execution_time integer

设置一个脚本被解析器中断之前，可以占用的最长时间以秒数计，这可以防止编写粗糙的脚本链接服务器。

memory_limit integer

设置脚本可以分配的最大内存数以位计，这可以防止编写粗糙的脚本占据服务器的所有可用内存。

*nsapi_ext*s tring

Green Apple Data Center

short_open_tag boolean

表示是否允许使用短格式的 PHP **提示符**<??>**。如果要结合** XML **一起使用，**PHP **就必须将该选项设置为无效，如果该选项有效，就必须使用长格式的提示符（**<?php?>**）。**

sql.safe_mode boolean

track_errors boolean

如果该选项值有效，则最后的错误信息将一直保留在全局变量$php_errormsg **中。**

track_vars boolean

如果该选项值有效，就可以分别在相关的全局数组$HTTP_GET_VARS**、**$HTTP_POST _VARS **和** $HTTP_COOKIE_VARS **中查找** GET**、**POST **和** cookie **输入。**

upload_tmp_dir string

下载文件时用以存储文件的临时目录。无论 PHP **以何种方式运行，它都必须保持可写状态。**

user_dir string

存放 PHP **文件的用户，主目录的目录基本名称如** public_html**。**

warn_plus_overloading boolean

如果该选项值有效，则当串中使用加号时系统会发出警告信息。这一设置的目的是使用户能够更快地找到需要将串链接符改成(.)**的脚本。**

2.4.2 邮件配置语句

SMTP string

Windows **中需要和** mail **函数一起发送的** SMTP **服务器的** DNS **名或** IP **地址。**

sendmail_from string

Windows **中需要在** PHP **发出的邮件中使用的**"From:"**邮件地址。**

sendmail_path string

能够找到 sendmail **程序的位置通常是**/usr/sbin/sendmail **或**/usr/lib/sendmailconfigure**，试图定位该位置并将它设为缺省值。但如果定位过程失败，用户可以在这个语句中进行设置。对于不使用** sendmail **的系统来说，如果需要将这一语句设置为该邮件系统所提供的** sendmail **来替代。**

2.4.3 安全模式配置语句

safe_mode boolean

设置是否允许使用 PHP **安全模式。**

safe_mode_exec_dir string

如果 PHP **在安全模式下运行，**system **和其他整型系统程序的函数将拒绝运行该目录之外的文件。**

2.4.4 调试器配置语句

debugger.host string

调试器所使用的主机 DNS **名或** IP **地址。**

debugger.port string

调试器所使用的端口号。

debugger.enabled boolean

设置调试器是否可用。

2.4.5 扩展加载语句

enable_dl boolean

该语句只有在 PHP **的** Apache **模块版本中才有用，可以针对每一个虚拟服务器或每一个目录，使用** dl（）**函数。动态加载** PHP **扩展的功能设置为开启或关闭。关闭动态加载功能的主要原因是考虑安全问题，在动态加载时很容易被忽略的。**

Green Apple Data Center

safe_mode 和 open_basedir 限制
只要不使用安全模式，缺省设置是允许使用动态加载，在安全模式中绝不可能使用 dl（）函数。
extension_dir string
设置 PHP 查找可动态加载的扩展功能的路径。
extension string
PHP 启动时所加载的可动态加载的扩展功能。

2.4.6 MySQL 配置语句

mysql.allow_persistent boolean
设置是否允许 MySQL 永久链接。
mysql.max_persistent integer
每次可处理的 MySQL 永久链接的最大数目。
mysql.max_links integer
每次可处理的 MySQL 链接的最大数目，包括永久链接。

2.4.7 mSQL 配置语句

msql.allow_persistent boolean
设置是否允许 mSQL 永久链接。
msql.max_persistent integer
每次可处理的 mSQL 永久链接的最大数目。
msql.max_links integer
每次可处理的 mSQL 链接的最大数目，包括永久链接。

2.4.8 Postgres 配置语句

*pgsql.allow_persisten t*boolean
设置是否允许 Postgres 永久链接。
pgsql.max_persistent integer
每次可处理的 Postgres 永久链接的最大数目。
pgsql.max_links integer
每次可处理的 Postgres 链接的最大数目，包括永久链接。

2.4.9 Sybase 配置语句

sybase.allow_persistent boolean
设置是否允许 Sybase 永久链接。
sybase.max_persistent integer
每次可处理的 Sybase 永久链接的最大数目。
sybase.max_links integer。
每次可处理的 Sybase 链接的最大数目，包括永久链接。

2.4.10 Sybase-CT 配置语句

sybct.allow_persistent boolean
设置是否允许 Sybase-CT 永久链接，缺省值是允许。
sybct.max_persistent integer
每次可处理的 Sybase-CT 永久链接的最大数目，缺省值为-1，表示没有限制。
sybct.max_links integer
每次可处理的 Sybase-CT 链接的最大数目，包括永久链接缺省值为-1，表示没有限制。

Green Apple Data Center

sybct.min_server_severity integer

以警告形式显示严重性超过或等同于 sybct.min_server_severity **值的服务器信息。该值可以通过在脚本中调用** sybase_min_server_severity **来设置，缺省值为** 10**，报告错误信息或更严重的错误。**

sybct.min_client_severity integer

以警告形式显示严重性超过或等同于 sybct.min_client_severity **值的客户机信息。该值可以通过在脚本中调用** sybase_min_client_severity **来设置，缺省值为** 10**，能有效防止错误报告。**

sybct.login_timeout integer

在返回失败信息之前尝试链接所能持续的最长时间（以秒计），注意当尝试链接的所用的时间超过 max_execution_time **所设置的时间时，脚本会在对所发生的错误采取措施之前被中断，缺省值为** 1 **分钟。**

sybct.timeout integer

在返回失败信息之前 select_db **或查询操作所等待的最长时间（以秒计），注意当一项操作所用的时间超过** max_execution_time **所设置的时间时，脚本会在对所发生的错误采取措施之前被中断，缺省值为没有限制。**

sybct.hostname string

要链接的主机名用于 sp_who **显示，缺省值为空。**

2.4.11 Informix 配置语句

ifx.allow_persistent boolean

设置是否允许 Informix **永久链接。**

ifx.max_persistent integer

每次可处理的 Informix **永久链接的最大数目。**

ifx.max_links integer

每次可处理的 Informix **链接的最大数目，包括永久链接。**

ifx.default_host string

当 ifx_connect **或** ifx_pconnect **都没有设置主机时缺省链接的主机。**

ifx.default_user string

当 ifx_connect **或** ifx_pconnect **中没有设置用户** id **号时缺省使用的用户** id **号。**

ifx.default_password string

当 ifx_connect **或** ifx_pconnect **中没有设置密码时缺省采用的密码。**

*ifx.blobinfile*boolean

如果需要返回文件中的团点列（blobcolumns**）请将该项设为真。如果只需要保存在内存中就设为假，运行时使用** ifx_blobinfile_mode **可以忽略该设置。**

ifx.textasvarchar boolean

如果需要将选定部分的文本列以正常的串形式返回，请将该项设为真。如果要使用 id **参数就设为假。运行时使用** ifx_textasvarchar **可以忽略该设置。**

ifx.byteasvarchar boolean

如果在选择查询时需要以正常串形式返回 BYTE **列，请将该项设为真。如果要使用** id **参数就设为假，运行时使用** ifx_textasvarchar **可以忽略该设置。**

ifx.charasvarchar boolean

如果希望从 CHAR **列中取数据时删除后面的空格，则该项设置为真。**

ifx.nullformat boolean

如果要将 NULL **列按字面返回** NULL**，请将该项设为真。如果直接返回空串就设为假，运行时使用** ifx_nullformat **可以忽略该设置。**

2.4.12 BCMath 配置语句

bcmath.scale integer

设置所有 bcmath **函数的小数点位数。**

Green Apple Data Center

2.4.13 Browser Capability 配置语句

browscap string
设置 browser capabilities 文件的名称。

2.4.14 Unified ODBC 配置语句

uodbc.default_db string
当 odbc_connect 或 odbc_pconnect 中没有设置数据源时使用的 ODBC 数据源。
uodbc.default_user string
当 odbc_connect 或 odbc_pconnect 中没有设置用户名时使用的用户名。
uodbc.default_pw string
当 odbc_connect 或 odbc_pconnect 中没有设置密码时使用的密码。
uodbc.allow_persistent boolean
设置是否允许 ODBC 永久链接。
uodbc.max_persistent integer
设置每次可处理的 ODBC 永久链接的最大数目。
uodbc.max_links integer
设置每次可处理的 ODBC 链接的最大数目，包括永久链接。

2.5 PHP 编程和调试环境

PHP 使用的各种操作平台上，都有许多集成的编程环境。在 Windows 下，编辑和集成工具是最多最好的，这也是程序员通常都是在 Windows 下开发 PHP 程序，再拿到 Unix/Linux 下面去用的原因。其实编写 PHP 程序只要一个文本编辑器就可以了（比如 Windows 下的记事本），但是它的功能狭小，需要功能更强大的编辑环境和开发工具。这里介绍 UltraEdit 和 PHP Editor 编辑工具。

2.5.1 Ultra Edit 编辑工具

UltraEdit 是 Windows 下比较著名的适合各种编程的编辑器，是著名的文本/二进制/十六进制编辑器，对文件大小无限制。还有宏定义、书签、查找/替换、自动换行、自动缩进、拼写检查、行/列显示、FTP 远端编辑等等特色功能，尤其值得称道的是打开文件速度极快。自 8.0b 版本开始更增加了复制（或剪切）并附加到剪贴板的功能，用户自定的特定配置和整理 HTML 代码的独创性功能绝对是编辑器中的老大了。最新版本增加了一个同步滚动图标，并且修正了以前版本的 BUG。

新版增加了复制（或剪切）并附加到剪贴板，用户自定的特定具置和整理 HTML 代码等新特性，可以说在编辑器类的软件中都是独创性的，其他编辑器中从未见过，如图 2-12 所示。

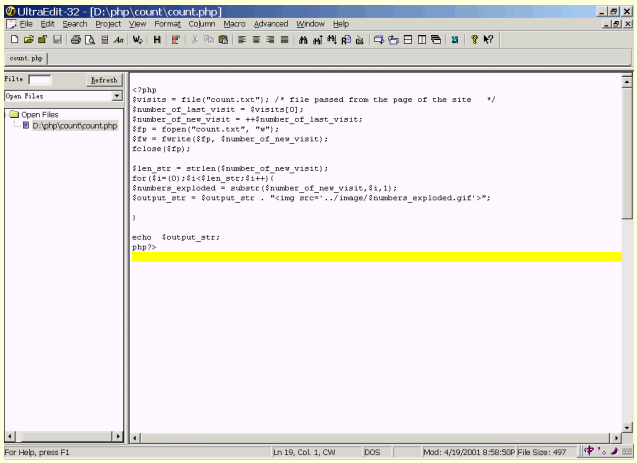


图 2-12 UltraEdit 编辑工具

Green Apple Data Center

它支持关键字高亮显示，一套极棒的文字、Hex、ASCII 码编辑器，可以取代记事本，内建英文单字检查、C++、VB、HTML、PHP 指令突出显示，可同时编辑多个文件，而且即使开启很大的文件速度也不会慢。

并且它具有 HTML Tag 颜色显示、搜寻替换以及无限制的还原功能；甚至用它来修改 EXE 或 DLL 文件。UltraEdit 的基本功能，就是一般的文书输入，它和 Win32Pad 同样有不错的编辑功能，也都比“记事本”要强上许多。

在编辑时，如果要拿走左边的“文件树状”结构区，如此可以让编辑范围扩大，那么可以按下 Ctrl+U 来切换。

Ultra Edit 最强的地方是在 16 进位的编辑功能，如图 2-13 所示，也就是说，如果要修改特殊文件的内码，那么可以使用 Ultra Edit。

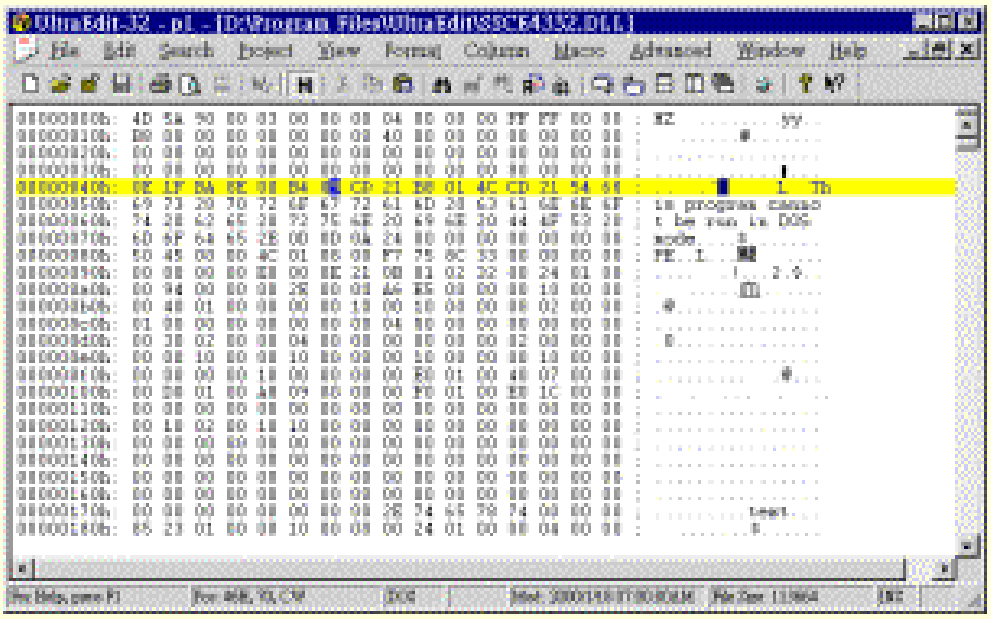


图 2-13　UltraEdit16 进位的编辑功能

例如，有许多游戏软件，都是以资料文件来记录主角的游戏进度和拥有的武器、体力、和金钱等，如果会使用 Ultra Edit 来修改这些 16 进位资料，那么就可以永远不死、而且轻易过关。而修改 16 进位资料的功能，正是 Ultra Edit 的过人之处。

2.5.2　PHP Editor

对熟悉 Windows 系列操作系统的使用者，推荐使用 PHP Editor 编辑器，其界面如图 2-14 所示，对 javascript 支持得很好。

若对这个编辑软件有兴趣，可以到 http://www.soysal.com /PHPEd 下载最新的版本，同时还需要下载 PHP 程序 Win32 版本。安装好 PHP Win32 后，在 PHP Editor 中设定好 PHP Win32 的路径，就可以轻松地开发 PHP 程序了。

至于在开发 PHP 程序的工作平台上，有没有 Web 服务器倒是不重要，因为在 PHP Editor 的环境中写好的程序，按下执行的键后，由 PHP Editor 直接将所写的 PHP 程序送给 PHP Win32，并将执行结果显示在编程者的面前。

Green Apple Data Center

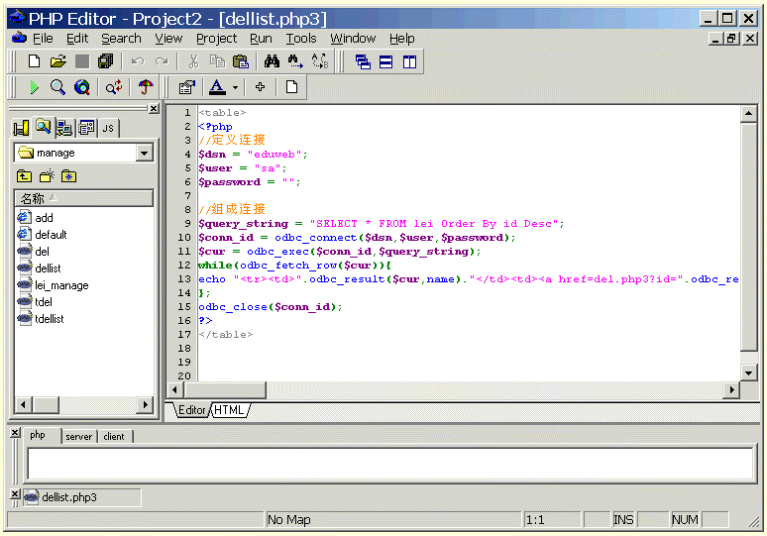


图 2-14 PHP Editor 界面

Green Apple Data Center

第 3 章　PHP 的语法

同其他高级语言一样，在 PHP 语言中也存在有数据类型、常量、变量和数组等概念，考虑到 PHP 语言的语法和 C（C++）、Java 等语言的语法极其相似，稍具有 C 语言基础的读者很容易就能上手，而 C 语言非常普及，所以本章只是简单的讲解 PHP 语言语法，使读者能够顺利地从其他语言转到 PHP 程序的开发上来。

3.1　PHP 语法概述

若只会用 Frontpage、Dreamweaver 的所见即所得编辑模式来写网页，而完全不懂 HTML 的语法，恐怕要先下点功夫了解 HTML 语法，才能顺利编写 PHP 程序，对于 PHP 这种后端服务器的程序语言，下苦功夫去学习 HTML 是编写 PHP 程序的必备条件。若早就非常了解 HTML 语言的语法，那么应该可以马上开始 PHP 的程序编写了。如果有一定的 C 语言基础，并了解 JavaScript 或者 VBScript 脚本语言，那么学习 PHP 语言将是一件十分简单轻松的事情。

编写 PHP 程序最好的方法，就是先处理好纯 HTML 格式的 Homepage 文件之后，再将需要变量或其它处理的地方改成 PHP 程序。这种方法，可以在开发上达到事半功倍的效果。

3.1.1　从一个简单的 PHP 程序说起

下面先看一个最简单的 PHP 程序。

```
<html>
<head>
<title>最简单的 PHP 程序</title>
</head>
<body>
<h1>
<?php
echo "这是我编写的第一个令人激动的 PHP 程序!\n";
?>
</h1>
</body>
</html>
```

这十多行程序在 PHP 中不需经过编译等复杂的过程，只要将它放在设定好可执行 PHP 语法的服务器中（第 2 章讲的 Apache 的 htdos 目录下），把它存成文件 1.php。在用户的浏览器里输入 http://localhost/1.php，就可以在浏览器上看到 echo 命令后面的字符串出现。如图 3-1 所示。

Green Apple Data Center

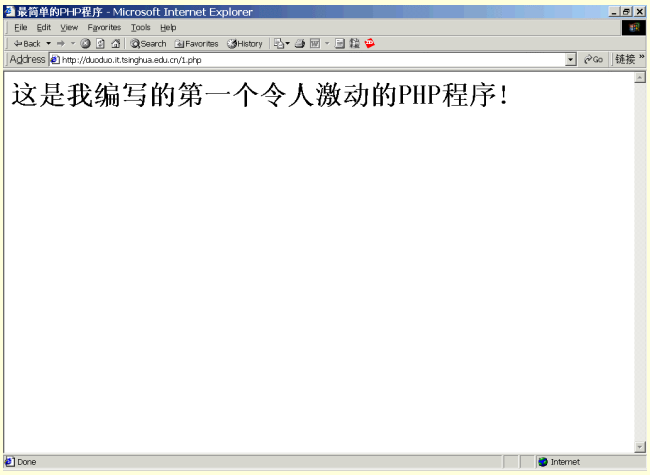


图 3-1 一个简单的 PHP 程序

实际上这个程序只有下面三行有用，其它几行都是标准的 HTML 语法，这里不再介绍 HTML 语法，稍有点 HTML 语法基础的读者都会明白其中的 HTML 部分内容的含义。

```
<?php
echo "这是我编写的第一个令人激动的 PHP 程序!\n";
?>
```

（1）HTML 转义

<?php 和?>分别是 PHP 的开始及结束的嵌入符号，中间一行才是服务器端执行的程序。而它在返回浏览器时和 JavaScript 或 VBScript 完全不一样，PHP 的程序没有传到浏览器，只在浏览器上看到如图所示的“这是我编写的第一个令人激动的 PHP 程序!”。在这个例子中，"\n"和 C 语言的表示都是这样，代表换行的意思。在第一章也有介绍过 PHP 是混合多种语言而成，而 C 正是含量最多的语言，在一个表达式结束后，要加上分号代表结束。

对 HTML 代码进行转义进入 PHP 代码模式的方法有四种：

```
① <? echo ("这是一行 PHP 语言代码\n"); ?>
② <?php echo("这是一行 PHP 语言代码\n"); ?>
③ <script language="php">
echo ("这是一行与 JavaScript 和 VBScript 语法相似 PHP 语言代码");
</script>
④ <% echo ("这是一行与 ASP 语法相似的 PHP 语言代码"); %>
```

其中第一种及第二种是最常用的二个方法，在小于号加上问号后，可以加也可以不加 php 三个字，之后就是 PHP 的代码。在代码结束后，加入问号和大于号就可以了。第三种方法对熟悉 Netscape 服务器产品的 Webmaster 人员而言，有相当的亲切感，它是类似 JavaScript 的写作方式。第四种方法对于在 Windows NT /2000 平台下开发过 ASP 的使用者来说似曾相识，以百分比大于号结束 PHP 的代码段，但想用第四种方法的使用者别忘了在 php.ini 加入 asp_tags 或是在编译时配置-enable-asp-tags 选项，才能使第四种方法有效。建议最好不用第四种方法，因为如果有 PHP 与 ASP 源代码混在一起时就不好区分了。

（2）语句分隔

与 C 或 Perl 相同语句中的各声明之间是用分号分隔的，闭合的提示符（?>）同时也意味着该声明的结束，所以，以下语句是等价的

```
<?php
```

Green Apple Data Center

```
echo "可爱的 PHP ";
?>
<?php echo "可爱的 PHP" ?>
```

（3）程序的注释

PHP 支持 C 语言、C++语言或者是 Unix 的 Shell 解释程序类型语言的注释方式，注释的方法也很灵活。可以单独使用，而且也可以一起混合使用。这可以让每个写 PHP 网页程序的程序员挑选出属于自己喜爱的风格。例如：

```
<?php
echo "我是可爱的 PHP "; //  这是一单行 C++风格的注释
/*  这是多行的
  注释方式*/
echo "我是最可爱的 PHP ";
echo "我是超级可爱的 PHP"; #  这是 UNIX  Shell  风格的注释
?>
```

不过在使用多行注释时请注意，不能让注释陷入嵌套循环当中，否则会引起错误。

```
<?php
   /*
   echo "这是错误的例子。\n"; /*  嵌套注释会引起错误  */
   */
?>
```

3.1.2 文件的引用

PHP 最吸引人的特色之一大概就是它的文件的引用了，用这个方法可以将常用的功能写成一个函数，放在文件之中，引用之后就可以调用这个函数了。

引用文件的方法有二种：require 及 include。二种方式提供不同的使用弹性。

require 的使用方法如“ require（"MyRequireFile.php"）; ”，这个函数通常放在 PHP 程序的最前面，PHP 程序在执行前，就会先读入 require 所指定引入的文件，使它变成 PHP 程序网页的一部分。include 使用方法如“ include（"MyIncludeFile.php"）; ”，这个函数一般是放在流程控制的处理代码段中，PHP 程序网页在读到 include 的文件时，才将它读进来，这种方式，可以把程序执行时的流程简单化。

例如在上面的 1.php 文件中加入一行代码“ require（"2.php"）; ”，再编辑一个 2.php 文件放在与 1.php 同一个目录下。

程序 1.php

```
<html>
<head>
<title>最简单的 PHP 程序</title>
</head>
<body>
<h1>
<? require("2.php"); ?>
<?php
   echo  "这是我编写的第一个令人激动的 PHP 程序!\n";
```

Green Apple Data Center

```
?>
</h1>
</body>
</html>
```

下面是 2.php **文件中的内容。**

```
<?php
   echo  "这是一个文件的引用例子。\n";
?>
```

结果在浏览器中如图 3-2 **所示。**

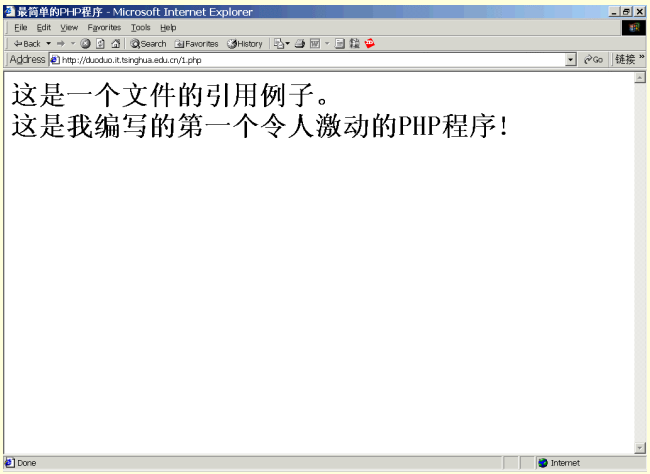


图 3-2　**文件引用例子**

3.2　数据类型

在变量使用上 PHP **和** C **语言等不同，**PHP **的任何变量都不需要在程序头部说明就可以拿来使用，变量类型由** PHP **自动识别。这一点虽然对于编程很方便，但是对程序的可读性来讲却是很不利的。程序员如果想到哪里就随手定义一个变量，那么过一段时间可能连他自己也不知道这个变量是什么意思。所以我们建议在变量使用之前用注释将变量说明一下，以便增强程序的易读性。为了不同** HTML **文档中其他字符混淆，**PHP **变量在使用的时候都要在前面加一个字符“** $ **”。**

PHP **的变量定义不多，**PHP **支持如下数据类型有整型（**integer**）、双精（**double**）、字符串（**string**）、数组（**array**）、对象（**object**）五种类型。**

（1**）整型（**integer**）。在** 32 **位操作系统中，它的有效范围是**-2,147,483,648 **到**+2,147,483,647**。要使用** 16 **进制整数可以在前面加** 0x**。**

整型变量可以用以下任意一种语法进行设置：

```
$number = 11; //十进制
$ number = -11; //  负数
$ number = 0x11; //十六进制
```

（2**）双精（**double**）。在** 32 **位的操作系统中，它的有效范围是** 1.7E-308 **到** 1.7E+308**。双精度数字可以用以下任意一种语法进行设置：**

Green Apple Data Center

```
$a = 1.234; // 表示数 1.234
$a = 1.2e3; //表示数 1200
```

（3）字符串（string）。string 即为字符串变量，值得注意的是要指定字符串给一个字符串变量，要在头尾加上双引号（例如:$str="这是字符串"）。就像在 C 和 Perl 中一样，反斜杠（"\"）可以用于定义特殊字符,在要使字符串换行时，可使用溢出字符，也就是反斜线加上指定的符号，若是\x 加上二位数字，如\xFE 即表示十六进位字符，详见表 3-1：

表 3-1 特殊的字符型变量表示

符号	意义
\ n	换行
\ r	回车
\ t	制表符
\ $	美元符号
\ 0 n n n	任一个八进制数
\ x n n	任一个十六进制数
\ \	反斜杠字符
\"	双引号

可以使用以下语法定义字符串变量：

```
$duoduo = "我是本书的作者";
$money = "何时我才有 100 万\$";
```

可以知道使用$字符时，为什么需要使用反斜杠。

（4）数组（array）。Array 是数组变量，可以是一维、二维、三维或者多维数组，其中的元素类型可以是 string、integer 或者 double，甚至是 array。一个数组就是把一系列数字和字符串作为一个单元来处理,数组中的每一个信息都被认为是数组的一个元素。一维数组元素有三种方法设置初始值：

① 可以对每一个元素分别赋值：

```
$a[0] = "abc";          //数组 a 第一个元素为字符串 abc
$a[1] = "def";          //数组 a 第二个元素为字符串 def
$b["foo"] = 13;         //数组 b 下标为 foo 的元素为数字 13
$a[] = "hello";         // $a[2] == "hello"
$a[] = "world";         // $a[3] == "world"
```

② 也可以使用 array()函数为数组同时对多个元素赋值：

```
$a = array(
  "color"   => "red",
  "taste"   => "sweet",
  "shape"  => "round",
  "name"  => "apple",
  3        => 4
);
```

这个语句效果和下面的语句是等价的：

```
$a["color"] = "red";
$a["taste"] = "sweet";
```

Green Apple Data Center

```
$a["shape"]       = "round";
$a["name"]= "apple";
$a[3]         = 4;
```

③ 最快的方法是简单在数组的下一个空余位置上增加一个元素，第一个位置是 0，第二个位置是 1，依次类推。例如，下面的代码给$arr_names 数组增加了三个元素，这三个元素的下标分别为 1、2 和 3（假设这个数组没有其他元素存在）。

```
$arr_names[] = ' duoduo';
$arr_names[] = ' duoduo1 ';
$arr_names[] = ' duoduo2 ';
```

多维数组实际上也是很简单的，对数组中的每一维来说，用户只是在数组的末尾添加另一个关键字值。定义多维数组可以用 array()函数为多维数组进行赋值：

```
<?
$a = array(
        "apple"    => array(
                "color"    => "red",
                "taste"    => "sweet",
                "shape"    => "round"
         ),
        "orange"    => array(
                "color"    => "orange",
                "taste"    => "tart",
                "shape"    => "round"
         ),
      "banana"    => array(
                "color"    => "yellow",
                "taste"    => "paste-y",
                "shape"    => "banana-shaped"
         )
         );
echo $a["apple"]["taste"];      # 输出 "sweet"
?>
```

根据需要，可以使用以下函数对数组进行排序 asort、arsort、ksort、rsort、sort、uasort、usort 和 uksort。

使用函数 count 可以对数组中的元素进行计数。

使用函数 next 和 prev 可以对数组进行遍历，更常用的遍历方法是使用函数 each。

（5）对象（object）。object 为对象变量，目前在 PHP 中的对象不多，若论及对象，Microsoft 的 ASP 对象仍然比 PHP 的内定对象多，相信这有赖大家的努力。不过话又说回来，Web CGI 程序要求的是效率，以完全面向对象的方式，恐怕使用者在浏览时也会因为程序执行速度慢而很不耐烦吧。

要使用变量，只要在英文字符串前面加个$即可，目前变量名称仍不能使用中文。至于变量的大小写是不一样的，对开发 PHP 程序的小组来说，最好使用相同的变量使用风格，以免因为变量大小的问题，花许多无谓的时间去找寻问题点，那就麻烦了。

对象的使用上就比较麻烦了，要先声明类别，甚至必须先有方法，方可使用对象，如下例：

```
class  foo  {
```

Green Apple Data Center

```
    function do_foo () {
      echo "Doing foo.";
    }
}
$bar = new foo;
$bar -> do_foo ();
```

（6）PHP 的类型转换

前面已经谈到，PHP **允许不说明变量就对变量进行应用。由** PHP **本身根据变量的赋值来规定变量是什么数据类型，也就是说，**PHP **语言中变量是自动转换的。请看下面的** PHP **的类型自动转换例子：**

```
$foo = "0";                          // $foo 变量是字符型，为字符“0”
$foo++;                          // $foo 变量是字符型，为字符“1”
$foo += 1;                           // $foo 变量是整型，为整数 2
$foo = $foo + 1.3;               // $foo 变量是浮点型，为实数 3.3
$foo = 5 + "10 Little Piggies";  // $foo 变量是整型，为整数 15
$foo = 5 + "10 Small Pigs";      // $foo 变量是整型，为整数 15
```

使用 gettype()**函数可以得到变量目前是什么类型。在编程的时候，如果程序员不满意** PHP **自动识别变量类型的结果，可以强制将某个变量转换到指定的类型，强制转换类型函数是** cast()**函数，或者** settype()**函数。转换方法如下：**

```
 (int), (integer)          类型强制转换成整型
(real), (double), (float)  类型强制转换成实型
(string)                   类型强制转换成字符型
(array)                        类型强制转换成数组
(object)                       类型强制转换成类
```

类型强制转换成数组以后，原来的变量值变成数组的第一个元素值；类型强制转换成类以后，原来的变量成为转换后类型的一个属性成员，属性成员的名字叫'scalar'**。请看下面** PHP **的强制类型转换例子：**

```
$foo = 10;                   // $foo 是一个整型变量
$bar = (double) $foo;    // $bar 强制转换成浮点型变量
$var = 'ciao';               // $var 是一个字符型变量
$arr = (array) $var;
echo $arr[0];            // $arr 强制转换成数组变量，数组第一个元素是变量$var 的值
$var = 'ciao';
$obj = (object) $var;
echo $obj->scalar;       // $obj 强制转换成类变量，类的 scalar 成员是变量$var 的值
```

3.3 常量和变量

3.3.1 常量类型

常量可以一种简单的方法使程序增加可读性，更容易让人理解。此外，由于大多数常量都在程序文件的开头部分定义，因此对它们进行更改也非常容易。

Green Apple Data Center

PHP 中定义了几种常量，同时还提供了在运行时定义更多常量的机制，常量与变量非常相似，但以下两点是不同的：

（1）常量必须用函数 define 定义。

（2）定义完之后不能再重新定义成其他值。

1. 系统预定义的常量

系统预定义的一些常量，用户写程序时不必自己再去定义，可以直接用。

（1）__FILE__

这个内定常量是 PHP 程序文件名。若引用文件（include 或 require）则在引用文件内的该常量为引用文件名，而不是引用它的文件名。

（2）__LINE__

这个内定常量是 PHP 程序行数。若引用文件（include 或 require）则在引用文件内的该常量为引用文件的行，而不是引用它的文件行。

（3）PHP_VERSION

当前使用的解析器版本号。

（4）PHP_OS

执行该 PHP 解析器的操作系统名称。

（5）TRUE

真值。

（6）FALSE

假值。

（7）E_ERROR

标识一个不能恢复的错误而不是解析错误。

（8）E_WARNING

标识一个 PHP 已经辨别出来的但仍将继续执行的错误，这类错误可能由脚本本身引起。

如 ereg 中无效的 regexp

（9）E_PARSE

由于脚本中无效语法导致的解析器错误，该错误是不可恢复的。

（10）E_NOTICE

当前发生的一些情况，可能是错误，也可能不是，程序将继续执行。

2. 使用 define 函数自定义常量

使用 define 函数可以同时定义数字常量和字符串常量，例如：

```
<?php
define(“PI”,”3.1415926”);
define(“HOST”,”http://duoduo.it.tsinghua.edu.cn”);
?>
```

除了不需要在常量名前加$符号外，存取常量值和存取变量值非常类似。在上面定义的两个常量可以用如下的方式存取：

```
<?
echo “主机名称是”.HOST;
?>
```

因为不使用初始的$符号，所以变量替换并不适合常量。下面看一个综合例子。

Green Apple Data Center

```
<?php
define("HOST","http://duoduo.it.tsinghua.edu.cn");
echo "主机名称是：".HOST;
echo "  所用的操作系统是",PHP_OS;
?>
```

在浏览器里的显示结果如图 3-3 所示。

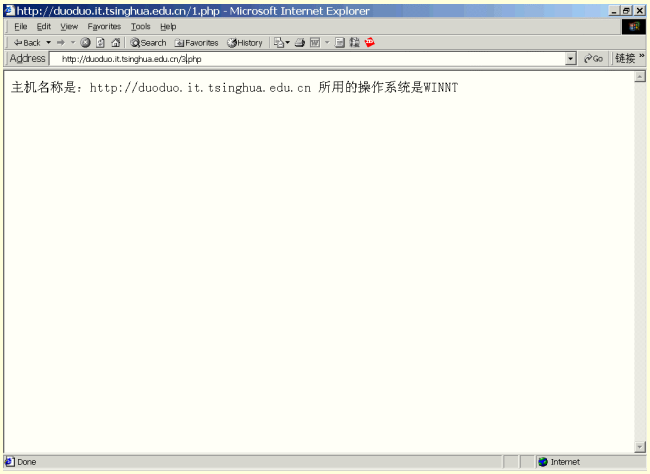


图 3-3　PHP 常量执行示例

3.3.2　变量设定

由于 PHP 许多语法都是 C 语言的翻版，故 PHP 在使用变量时，随时都可以使用新的变量，只要在使用前将变量初始化就好了，不必像 Pascal 语言那样严谨，所有要使用的变量都要事先声明。

（1）变量作用域

变量的作用域就是该变量所定义的运行环境。绝大多数时候，所有的 PHP 变量都只有一个作用域，但在用户自定义函数中，引入了局部函数作用域的定义，缺省情况下，任何一个函数内部的变量都限制在该局部函数作用域内。例如：

```
$a = 888; /*  全局变量  */
Function Test () {
echo $a; /*  显示局部变量  */ }
Test ();
```

这段脚本程序将不会造成任何的输出，因为这句话的目的是显示变量$a，同时这个变量没有在它的作用域中被指定一个值。您可以注意到这和 C 语言有一些不同，在 C 语言中全局变量是自动被设定为可用的，除非在函数中进行了特别的说明。由于人们可能会不小心的改变了全局变量的值，所以这可能会在程序中导致许多问题。

在 PHP 的程序执行时，系统会在内存中保留一块全局变量的区域。实际运用时，可以通过$GLOBALS["变量名称"]将需要的变量取出。在使用者自定义的函数或程序中，就可以用$GLOBALS 数组取出需要的变量，当然别忘了 PHP 的变量有分大小写。$GLOBALS 数组是 PHP 程序中比较特殊的变量，不必声明，系统会自动匹配相关的变量在里面。在函数中，也不必管$GLOBALS 数组是否已经做全局声明，就可以直接使用了。

在 PHP 中如果要在函数中使用全局变量，必须在函数内部将该变量声明为全局变量。举例如下：

```
$a =8;
$b =88;
```

Green Apple Data Center

```
Function Sum () {
global $a, $b;
$b = $a + $b;
}
Sum ();
echo $b;
```

上面的脚本将输出“96”。在函数中申明了全局变量$a 和$b，任何对这两个变量的引用都被指定到了该全局变量。这里没有函数可以操作限制全局变量的数目。

还有一种接受全局变量的方法是使用 PHP 特殊的定义数组$GLOBALS，举例如下：

```
$a = 1;
$b = 2;
Function Sum () {
$GLOBALS["b"] = $GLOBALS["a"] + $GLOBALS["b"];
}
Sum ();
echo $b;
```

$GLOBALS 数组是一个联合数组使用“global”为变量的名字，全局变量作为该变量数组其中某个元素的值。

另一个关于变量作用域的重要特性是“静态变量”。一个静态变量仅仅在本地函数的作用域中存在，但是当程序离开这个范围时，它的值并不丢失。请参考下面的例子：

```
Function Test () {
$a = 0;
echo $a;
$a++;
}
```

这个函数每次被调用时都把变量$a 置为 0 同时打印“0”，所以几乎是没有作用的。表达式“$a++”将增加变量的值，但是每次退出函数时变量$a 就消失了。想要使用一个不丢失当前计算的记数函数，用户可以将变量$a 设置为静态的，示例如下：

```
Function Test () {
static $a = 0;
echo $a;
$a++;
}
```

现在，每一次当 Test()函数被调用的时候，它将打印出变量$a 和它当时增加之后的数值。当函数被递归调用的时候，使用静态变量是一种很重要的方法。递归函数就是可以调用自身的函数。当编写递归函数的时候，必须注意可能会发生的循环定义。您必须有一个适当的方法来中断这个递归过程。下面的例子递归了 10 次：

```
Function Test () {
static $count = 0;
$count++;
```

Green Apple Data Center

```
echo $count;
if ($count < 10) {
Test (); }
$count--;
}
```

（2）变化变量

有些时候使用变化变量是十分方便的。也就是说，一个变量的名字将被动态的设置和使用。一个普通的变量将会使用如下的申明：

```
$a = "duoduo";
```

一个变化变量获得一个变量的值，并将其视为该变量的名字。在上面的例子中的“ duoduo ”，能够使用变量的名字加上两个$来进行使用，例如。

```
$$a = "love php";
```

在这一点上，两个变量被定义和存储在 PHP 的符号树上；$a 的内容为“ hello ”，而$hello 的值为“ world ”。因此如下的申明：

```
echo "$a ${$a}";
```

制作了和如下相同的输出：

```
echo "$a $duoduo";
```

它们都输出：“ duoduo love php ”

要在数组中使用变化变量，必须解决一个含糊的问题：如果写入了“ $$a[1] ”，然后解析程序将需要知道您想使用$a[1]作为变量还是使用$$a 作为变量，这样索引“[1]”可能就会发生歧义。解决这种歧义的语法如下：“ ${$a[1]} ”或者使用“ ${$a}[1] ”（对上述的第二种情况）。

3.3.3 外界 PHP 变量

（1）HTML 表单（GET 和 POST）

当表单被提交给 PHP 脚本时，从该表单获得的变量将自动地被该 PHP 脚本设置为可用。例如：

```
<form action="post.php" method="post">
Name: <input type="text" name="name"><br>
<input type="提交">
</form>
```

当提交之后，PHP 将建立一个变量“ $name ”，它将包含任何在表单中输入在“ Name ”中的内容，同样 PHP 也能理解表单变量形式的数组，但是只能使用一维数组，例如，可以将相关的数组组合到一个组中，或者利用该特性对多重选定的输入进行检索，从而获得值。

例如，图 3-4 所示的表单提交实例，可以用一个数组 personal 分别存放姓名和邮件输入值，用数组 study 存放学历。

Green Apple Data Center

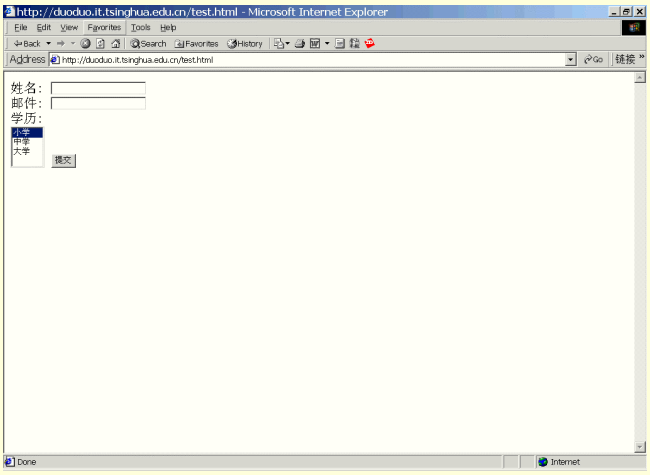


图 3-4　一个表单提交实例

```
<form action="post.html" method="post">
姓名: <input type="text" name="personal[name]"><br>
邮件: <input type="text" name="personal[email]"><br>
学历: <br>
<select multiple name="study[]">
<option value="primary">小学
<option value="middle">中学
<option value="high">大学
</select>
<input type="submit" value="提交" >
</form>
```

如果 PHP 的 track_var 特性被打开了，任何关于它的结构设置或者<?php_track_vars?>指示经由 POST 或 GET 模式的变量提交，将发觉使用全局联合数组“ $HTTP_POST _VARS ”和“ $HTTP_GET_VARS ”是合适的。

（2）图形提交变量名字

当提交一个表单时，有可能使用一幅图象来代替标准的带标签的提交按钮，例如：

```
<input type=image src="image.gif" name="sub">
```

当用户点击该图象的任何位置时，相应的表单将被使用两个附加的变量（sub_x 和 sub_y）来传送到服务器上，包含用户点击在该图形上的相应位置信息。这将包含从浏览器发出的真实变量名字（甚至包含下划线），但是 PHP 将会把它自动的转化成带下划线的形式。

（3）HTTP Cookies

PHP 支持使用 Netscape's Spec 定义的 HTTP Cookies，Cookies 是一种在远程浏览器上存放数据的装置，用于跟踪和鉴定用户的身份。可以使用 SetCookie()函数来对 cookies 进行设置，Cookies 是 HTTP 头的一部分，所以 SetCookie 必须在程序向浏览器输出前被调用，这和 Header()函数的设置是类似的。任何从服务器送到用户处的 cookies 将自动地被转化成为一个 PHP 变量，就像使用 GET 和 POST 模式的数据一样。

如果希望对单个的 cookie 分派多值，只要在 cookie 的名字后面简单的加上一个“ [ ] ”即可。例如：

```
SetCookie ("MyCookie[]", "Testing", time()+3600);
```

注意，如果不是在浏览器上的域，或者路径不一样的话，cookie 将会使用相同的名字来替代先前的一个 cookie。所以，对于购物单应用程序，可能希望保持一个计数器同时通过 cookie 来传送它，例如：

Green Apple Data Center

```
$Count++;
SetCookie ("Count", $Count, time()+3600);
SetCookie ("Cart[$Count]", $item, time()+3600);
```

（4）环境变量

PHP **自动地使用环境变量作为** PHP **的普通变量，例如：**

```
echo $HOME; /*  如果设置了环境变量，则显示出来。  */
```

既然信息随同 GET,Post,Cookie **等机制被传递进来，并且自动创建了** PHP **变量，有时最好是准确的从外界环境中读出一个变量以确信正在使用正确的版本。**getenv()**函数就是做这个工作的，也可以使用** putenv()**函数来设置一个环境变量。**

3.4 运算符号

3.4.1 算术运算符

算术运算符号，就是用来处理四则运算的符号，这是最简单，也最常用的符号，尤其是数字的处理，几乎都会使用到算术运算符号。

表 3-2 **算术运算符号**

实例	**名称**	**结果**
$a+ $b	**加法**	$a $b **之和**
$a -$b	**减法**	$a **减去**$b **后剩余的值**
$a *$b	**乘法**	$a **和**$b **的乘积**
$a /$b	**除法**	$a **被**$b **除之后的结果**
$a %$b	**取模**	$a **被**$b **除之后的余数**

3.4.2 字符运算符

字符串运算（string operator**）的运算符号只有一个，就是英文的句号（.）。它可以将字符串链接起来，变成合并的新字符串。以下是字符串运算的例子，将显示字符“** Duoduo love php. **”。**

```
<?php
$a  =  "Duoduo";
$b  =  "love php.";
echo  $a.":  ".$b;
?>
```

3.4.3 赋值运算符

基本的赋值运算符是“ = **”，它将运算符左侧的操作数值设为右侧表达式的值。赋值表达式的值就是被赋予的值，也就是说**"$a = 3"**的值就是** 3**，这样，读者就可以编写一些复杂的语句如：**

```
$a = ($b = 8) + 8 // $a  现在的值是 16，  $b  的值是 8。
```

除了基本赋值运算符之外，还有许多由二进制的运算和字符串的合成运算符，使得读者可以在表达式中使用一个值，然后将该值作为该表达式的结果，例如：

```
$a = 1;
$a += 7; // $a  的值为 8，相当于$a = $a + 5;
```

Green Apple Data Center

```
$b = "Duoduo ";
$b .= "love php!"; // 字符$b 为 "Duoduo love php!", j 相当于 $b = $b . "love php!";
```

3.4.4 逻辑和比较运算符

逻辑运算（logical operators）通常用来测试真假值。最常见到的逻辑运算就是循环的处理，用来判断是否该离开循环或继续执行循环内的指令。比较运算符用来比较两个值。

表 3-3 逻辑和比较运算符

实例	名称	结果
$a and $b	与	如果$a $b 都为真结果为真
$a or $b	或	$a $b 中任意为真时结果为真
$a xor $b	异或	当$a $b 中有一个为真但又不同时为真时结果为真
! $a	非	当$a 为真时结果为假
$a && $b	与	如果$a $b 都为真结果为真
$a || $B	或	$a $b 中任意为真时结果为真
$a == $b	等于	当$a 等于$b 时结果为真
$a != $b	不等于	当$a 不等于$b 时结果为真
$a >$b	大于	当 $a 严格大于$b 时值为真
$a < $b	小于	当 $a 严格小于$b 时值为真
$a <= $b	小于等于	当$a 小于等于$b 时值为真
$a >= $b	大于等于	当$a 大于等于$b 时值为真

3.4.5 位运算符

PHP 的位操作符（bitwise operators）共有六个，位运算符允许读者将整型值中的特定位设为开启或关闭状态。

表 3-4 位运算符

实例	名称	结果
$a & $b	与	将$a 和 $b 中都为 1 的位置 1
$a | $b	或	将$a 或 $b 中为 1 的位置 1
$a ^ $b	异或	将$a 或$b 中为 1 但又不同时为 1 的位置为 1
$a >> $b	右移	将$a 中的位右移$b 位每移一位表示原值除以 2
$a <<$b	左移	将$a 中的位左移$b 位每移一位表示原值乘以 2
~$a	否	将$a 中为 1 的位置为 0 将为 0 的位置为 1

3.5 流程控制

任意一个脚本都是由一系列语句组成的，语句可以是一个赋值、一个函数、调用一个循环，甚至还可以是一个什么都不做的空语句，语句通常用分号结尾。此外可以用花括弧{}将一组语句括起来，形成一个语句组。

3.5.1 if...else 条件语句

if...else 结构是 PHP 中最重要的结构，它允许进行代码段的条件执行。PHP 的 if...else 结构和 C 语言中的 if...else 结构相似，if...else 结构有 4 种：

（1）只有用到 if 条件，当作单纯的判断。语法如下：

```
if (expr) {
statement
}
```

Green Apple Data Center

根据本部分中对表达式的介绍，需要对 expr 值进行计算。如果 expr 值为 TRUE，PHP 将执行语句；如果值为 FALSE，将忽略语句。

以下的实例在$a 大于$b 时将显示 ：a 比 b 大。

```
if ($a > $b) print " a 比 b 大";
```

如果根据条件执行多于一条语句。当然，需要给每条语句都加上 if 判断。可以把多条语句组成一个语句组，用花括号{}把它们括起来。例如，当$a 比$b 大时以下程序会显示：a 比 b 大，并把$a 的值赋给$b:

```
if ($a > $b) {
print " a  比 b 大";
$b = $a;
}
```

if 语句可以嵌套于其他 if 语句中，这样更能够灵活有条件的执行程序的各个部分。

例如：

```
if ($a > $b) {
if  (date("D")  ==  "Sat")  {
      print “该是去玩的时候了”
                        }
            }
```

（2）除了 if 之外，加上了 else 条件，可理解成“ 如果条件成立，则怎样处理，否则就另外处理 ”。语法如下：

```
if (expr) {
   statement1
} else {
   statement2
      }
```

例如以下代码在$a >$b 时显示：a 比 b 大；在$a<$b 时显示：a 比 b 小。

```
if ($a > $b) {
print " a  比 b 大";
}else{
print " a  比 b 小";
}
```

（3）else if 结构，就像它的名称所显示的那样，else if 是 if 和 else 的组合，和 els 相似，它将 if 语句进行拓展，以便在初始的 if 表达式值为 FALSE 时执行语句，但是与 else 不同的是它只在 else if 条件表达式值为 TRUE 时，才会执行相应的语句。语法如下：

```
if(expr) {
   statement1
} else if{
   statement2
      }
```

Green Apple Data Center

```
else{
   statement3
}
```

（4）if 语句的交替格式：if():... endif。

PHP 允许用另外一种办法在 if 语句中使用一组语句。这通常用于嵌套一段 HTML 代码块于 if 语句里，但也可以用于任何地方。if（expr）后面必须用一个冒号取代原来的大括号 ，后跟一条或几条语句组成的语句组，并以 endif 结束。例如：

```
<html>
<body>
<h1>
<?php if ($a==8): ?>
我发财了！
<?php endif; ?>
</h1>
</body>
</html>
```

上例中，HTML 代码块"我发财了！"嵌套在一个 if 语句内，这段 HTML 代码仅在$a 等于 8 时显示 。

交替语法也可应用于 esle 和 elseif（expr）。例如下面就是一个含有 elseif 和 else 的交替格式的 if 语句：

```
if ($a==5):
print "a 等于 5";
print "...";
elseif ($a==6):
print "a 等于 6";
print "!!!";
else:
print "a 既不是 5 也不是 6";
endif;
```

3.5.2 do...while 循环语句

do..while 是重复的循环，可以分成 2 种循环方式。

（1）单纯 while 的循环。用来在指定的条件内，只要 WHILE 表达式为 TRUE 就重复执行嵌套的语句。每次循环开始时检查 WHILE 表达式的值，所以即使在嵌套语句内改变了它的值，本次执行也不会终止,直到本次循环结束（每次 PHP 运行嵌套的语句称为一次循环）。如果一开始 WHILE 表达式的值就是 FALSE，这样嵌套语句一句也不会被执行。语法如下：

```
while (expr) {
   statement
            }
```

其中的 expr 为判断的条件，通常都是用逻辑运算符号（logical operators）当判断条件。而 statement 为符合条件的执行代码段程序，若程序只有一行，可以省略大括号{}。

例如下面代码在浏览器里显示 10 行“ PHP 是最优秀的脚本语言 ”。

```
<?php
$i = 1;
```

Green Apple Data Center

```
 while ($i <= 10) {
   print $i++;
   echo " PHP 是最优秀的脚本语言<br>\n";
}
?>
```

while 还类似于 if 语句，可以用大括号把一组语句括起来，或使用下面另一种语法，从而在同一个 while 循环中执行多条语句。语法如下：

```
while (expr):
statement 1
statement 2
...
 endwhile;
```

（2）do...while 循环则先执行，再判断是否要继续执行，也就是说循环至少执行一次。它和单用 while 的循环的主要区别是 do...while 的第一次循环肯定要执行（真值表达式仅在循环结束时检查），而不必执行严格的 while 循环（每次循环开始时就检查真值表达式，如果在开始时就为 FALSE，循环会立即终止执行）。语法如下：

```
do {
   statement
} while (expr);
```

3.5.3 for 循环语句

for 循环是 PHP 中最复杂的循环，也是用得非常多的循环，它和 C 语言中相应语句的使用方法相同，其语法如下：

```
for (expr1; expr2; expr3) statement
```

第一个表达式（expr1）将在循环的开始处将无条件计算执行一次，而在每次重复的开始处 expr2 都会被计算。如果值为 TRUE，循环将继续，同时继续执行嵌套的语句；如果值为 FALSE，循环的执行就结束了。

下列代码用 for 循环在浏览器里显示 10 行“ PHP 是最优秀的脚本语言 ”，实现和上面 while 相同的功能。

```
for ($i=1; $i<=10; $i++) {
print $i;
echo " PHP 是最优秀的脚本语言<br>\n";
      }
```

实际应用中，若循环有初始值，且都要累加（或累减），则使用 for 循环比用 while 循环好。这也就是 for 循环用的比 while 循环多的原因之一了。

类似 C 语言，PHP 在 for 循环中也支持冒号语法，语法如下：

```
for (expr1; expr2; expr3): statement; ...; endfor;
```

Green Apple Data Center

3.5.4 其他流程控制语句

（1）breake 语句

break 用来跳出目前执行的循环，例如：

```
<?php
$i = 0;
 while ($i < 10) {
   if ($arr[$i] == "stop") {
break;
   }
   $i++;
}
?>
```

（2）continue 语句

continue 立刻停止目前执行循环，并回到循环的条件判断处，例如：

```
<?php
while (list($key,$value) = each($arr)) {
   if ($key % 2) { // 略过偶数
continue;
   }
   do_something_odd ($value);
 }
?>
```

（3）switch 语句

switch 语句类似于对应同一个表达式的一系列 if 语句。许多情况下需要将同一个变量和不同的值相比较，并且根据比较的值执行不同的代码段，这就是 switch 语句的作用。

如果需要同时测试、判断多个条件值时，if 语句处理起来就显得比较烦琐，因为所有的 elseif 语句都要执行一遍。在这种情况下许多人会发现，使用 switch 语句会更容易、快速。其语法如下：

```
switch (expr) {
   case expr1:
statement1;
break;
   case expr2:
statement2;
break;
:
:
   default:
statementN;
break;
}
```

Green Apple Data Center

以下的两个实例用不同的方法完成了相同的任务，一个是使用一系列 if 语句，另一个用了 switch 语句。例一：

```
if ($i == 0) {
print "i 等于 0";
if ($i == 1) {
print " i 等于 1";
if ($i == 2) {
print "i equals 2";
}
```

例二：

```
switch ($i) {
case 0:
print "i 等于 0";
break;
case 1:
print "i 等于 1";
break;
case 2:
print "i 等于 2";
break;
}
```

3.6 函数

3.6.1 自定义函数

函数帮助程序员组织自己的代码，使其成为比较容易理解和使用的代码段。函数让程序员一步步地编写出程序，并以这种方式测试代码。

在对所编程序有一个初步构想之后，会需要在脑子里或在纸上列出编程大纲。所列大纲中的每个步骤，可能就是一个函数，这就称之为模块化的程序设计，这种技术将编程的详细过程隐藏起来，以便读源程序的人明白整个程序的设计目标。

例如，如果在所编程序中包含有一个计算圆面积的函数，可以调用如下一行程序编码：

```
$flt_area_of_circle = area_of_circle(5);
```

读者在看到所调用的函数以后，一般都会明白程序在做什么了，而对函数的实际内容并不需要过多了解。

☞ 注意

调用函数意味着 PHP 将停止程序当前行的执行，跳转到所调用的函数中去。在函数执行完毕以后，PHP 会重新跳回到程序中调用函数的地方，继续往下进行。

下面看一看函数调用过程，在命令行中首先出现的是一个标量和一个赋值操作符，应该知道其作用是将赋值操作符右边的数值赋予变量$flt_area_of_circle，但是赋值符右边的究竟是些什么呢？

最先看到的是函数名：area_of_circle()，紧跟其后的圆括号表示这是一次函数调用，在圆括号中的是准备传给函数的一些参数或数值。可以将参数设想为一个足球，传球时，接收方（比如函数）会有几种选择：运球（以某种方式进行修改），传球出去（调用其他程序），犯规（调用错误处理程序）。

Green Apple Data Center

一个函数的语法结构如下：

```
function functionName ( parameterList ) {
   //代码行
}
```

函数取名有几个规定——其中最重要的一条就是，函数名不能以数字开头，且中间不能有空格。参数表可以任选，它提供了函数可以使用的特定数值。在函数中可以使用任何有效的 PHP 代码，甚至包括调用其他的函数和类定义，应注意的是函数必须在引用之前定义。

3.6.2 返回值

PHP 中的函数（function）和 C 语言一样，包括有返回值的和无返回值的，不像 Pascal 分成函数（function）和程序（procedure）那么复杂。

函数可以通过可选的 return 语句返回值。返回值可以是任何类型，包括列表和对象。

```
function my_sqrt （ $num ） {
    return $num * $num;
}
echo my_sqrt( 4 );      // outputs '16'.
```

函数不能同时返回多个值,但可以通过返回列表的方法来实现：

```
function foo() {
    return array( 0, 1, 2 );
}
list( $zero, $one, $two ) = foo();
```

3.6.3 函数参数

一般来说，PHP 会将参数的值传递给函数，这就意味着函数不能改变参数表中任何变量的值。在看过下例以后，就会比较清楚为什么会这样了：

```
function one( $parameter ) {
   $parameter++;
}
$a = 10;
one($a);
echo "a=$a<br>";
```

如果函数 one()中的加一操作对变量$a 有影响的话，那么可以预期函数 echo 的输出结果为 a=11，而事实上并非如此。与此相反，本例的输出显示为 a=10。

设计要改变其参数的函数并不是一个好主意。如果要弄明白这是为什么，就要研究一下编程原理。当两段代码共享信息时，它们就被称为是紧耦合的，此时当一段程序改变时－比如加入某种功能－很可能另外一段程序也要跟着改变。因为两段程序需要同时进行改动，所以这时发生差错的几率就比较高。另一方面，如果编制的两段程序的关系是松耦合的话，那么，对程序的修改是相互隔离的，所以发生错误的几率相应就会降低。只有在某些特定的情况下，有些函数需要修改其参数值。

当函数必须要修改其参数时，那些参数需要通过引用的方式传递给函数。在函数中使用变量引用作为参数

Green Apple Data Center

时，提供的是存放变量的内存地址，下面给出的函数 one 就是使用引用变量的例子：

```
function one( &$parameter ) {
   $parameter++;
}

$a = 10;
one($a);
echo "a=$a<br>";
```

这段小程序的结果输出为 a=11，表明函数 one()改变了变量$a 的值。注意，采用引用变量的唯一不同就是，需在所定义函数的参数名前面加上&号。

如果使用另一个程序员编的程序，就不要奢望去改变函数的定义。在那种情况下，可以在所调用函数的参数前面加上一个&号，如下示：

```
one(&$a);
```

在结束参数传递这一个话题之前，有必要讲一讲数组作为函数的参数传递的情况，下例显示了如何从函数中传递出数组元素值：

```
function array_first( $arr_parameter ) {
   return($arr_parameter[0]);
}

$a = array_first( array(3,5) );
echo "a=$a<br>";
```

在这个例子中定义了一个查找第一个数组元素的函数。请注意，在此用到了标准的数组元素记号，函数使用数组作为参数和使用标量作为参数的情况并无太大的区别。

Green Apple Data Center

第 4 章　PHP 常用函数说明

PHP 语言的函数有上千个，光是介绍 PHP 函数就可以写一本上千页的书，本章只介绍一些在编程时最常用的一些函数，而其他一些函数，编程时基本上用不到，若实在需要那些不常用的函数时，用户可以参考本书附录 A 的函数列表或者其他专门详细介绍 PHP 全部函数的书籍。

4.1　程序执行功能函数库

本函数库共有 4 个函数。
escapeshellcmd：除去字符串中的特殊符号。
exec：执行外部程序。
system：执行外部程序并显示输出。
passthru：执行外部程序并不加处理输出。

escapeshellcmd

除去字符串中的特殊符号。
语法：string escapeshellcmd(string command);
返回值：字符串
内容说明
本函数除去了字符串中的特殊符号，可以防止使用者耍花招来破解该服务器系统。可以用本函数搭配 exec() 或是 system()二个函数，减少网络上的使用者恶意破坏的机会。
使用范例

```
<?php
system(EscapeShellCmd($cmdline));
?>
```

exec

执行外部程序。
语法：string exec(string command, string [array], int [return_var]);
返回值：字符串
内容说明
本函数执行输入 command 的外部程序或外部指令。它的返回字符串只是外部程序执行后返回的最后一行；若需要完整的返回字符串，可以使用 PassThru()这个函数。
要是参数 array 存在，command 会将 array 加到参数中执行，若不欲 array 被处理，可以在执行 exec()之前呼叫 unset()。若是 return_var 跟 array 二个参数都存在，则执行 command 之后的状态会填入 return_var 中。
值的注意的是若需要处理使用者输入的指令，而又要防止使用者耍花招破解系统，则可以使用 EscapeShellCmd()。

system

执行外部程序并显示输出。
语法：string system(string command, int [return_var]);
返回值：字符串
内容说明
本函数就像是 C 语言中的函数 system()，用来执行指令，并输出结果。若是 return_var 参数存在，则执行 command 之后的状态会填入 return_var 中。同样值得注意的是若需要处理用户输入的指令，而又要防止用户耍

Green Apple Data Center

花招破解系统，则可以使用 EscapeShellCmd()。若 PHP 以模块式的执行，本函数会在每一行输出后自动更新 Web 服务器的输出缓冲区。若需要完整的返回字符串，且不想经过不必要的其他中间的输出界面，可以使用 PassThru()。

passthru

执行外部程序并不加处理输出。
语法：string passthru(string command, int [return_var]);
返回值：字符串
内容说明
本函数类似 Exec()用来执行 command 指令，并输出结果。若是 return_var 参数存在，则执行 command 之后的状态会填入 return_var 中。若输出的是二进位的，并且需要输出到浏览器中的话，使用本函数就相当合适了。例如使用 pbmplus 工具来执行指令，并返回二进位的图形。可以先配置返回的标头(header)为 Content-type：image/gif，然后呼叫 pbmplus 程序处理图形，并将二进位的图形直接返回浏览器。

4.2 字符串处理函数库

本函数库共有 51 个函数，它们是：
AddSlashes：字符串加入斜线。
bin2hex：二进位转成十六进位。
Chop：去除连续空白。
Chr：返回序数值的字符。
chunk_split：将字符串分成小段。
convert_cyr_string：转换古斯拉夫字符串成其他字符串。
crypt：将字符串用 DES 编码加密。
echo：输出字符串。
explode：切开字符串。
flush：清除输出缓冲区。
get_meta_tags：抽出文件所有 meta 标记的数据。
htmlspecialchars：将特殊字符转成 HTML 格式字符。
htmlentities：将所有的字符都转成 HTML 字符串。
implode：将数组变成字符串。
join：将数组变成字符串。
ltrim：去除连续空白。
md5：计算字符串的 MD5 杂凑值。
nl2br：将换行字符转成
。
Ord：返回字符的序数值。
parse_str：解析 query 字符串成变量。
print：输出字符串。
printf：输出格式化字符串。
quoted_printable_decode：将 qp 编码字符串转成 8 位字符串。
QuoteMeta：加入引用符号。
rawurldecode：从 URL 专用格式字符串还原成普通字符串。
rawurlencode：将字符串编码转成 URL 专用格式。
setlocale：配置地域化信息。
similar_text：计算字符串相似度。
soundex：计算字符串的读音值
sprintf：将字符串格式化。
strchr：寻找第一个出现的字符。
strcmp：字符串比较。
strcspn：不同字符串的长度。
strip_tags：去掉 HTML 及 PHP 的标记。
StripSlashes：去掉反斜线字符。

Green Apple Data Center

strlen：取得字符串长度。
strpos：寻找字符串中某字符最先出现处。
strrpos：寻找字符串中某字符最后出现处。
strrchr：取得某字符最后出现处起的字符串。
strrev：颠倒字符串。
strspn：找出某字符串在另一字符串掩码的数目。
strstr：返回字符串中某字符串开始处至结束的字符串。
strtok：切开字符串。
strtolower：字符串全转为小写。
strtoupper：字符串全转为大写。
str_replace：字符串取代。
strtr：转换某些字符。
substr：取部分字符串。
trim：截去字符串首尾的空格。
ucfirst：将字符串第一个字符改大写。
ucwords：将字符串每个字第一个字母改大写。

AddSlashes

字符串加入斜线。
语法：string addslashes(string str);
返回值：字符串
内容说明
本函数使需要让数据库处理的字符串引号的部分加上斜线，以供数据库查询(query)能顺利运作。这些会被改的字符包括单引号(')、双引号(")、反斜线 backslash(\)以及空字符 NUL (the null byte)。

bin2hex

二进位转成十六进位。
语法：string bin2hex(string str);
返回值：字符串
内容说明
本函数让二进位字符串转成十六进位字符串。
使用范例
下面的函数将 16 进制变成 2 进制数字

```
<?
function hex2bin($data) {
   $len = strlen($data);
   for($i=0;$i<$len;$i+=2) {
         $newdata .= pack("C",hexdec(substr($string,$i,2)));
   }
   return $newdata;
}
?>
```

Chop

去除连续空白。
语法：string chop(string str);
返回值：字符串
内容说明
本函数将字符串的连续空白清除。

Green Apple Data Center

使用范例

```
<?
$trimmed = Chop($line);
?>
```

Chr

返回序数值的字符。
语法：string chr(int ascii);
返回值：字符串
内容说明
本函数将字符的序数转成 ASCII **的字符。本函数和** ord()**成对照。**

使用范例

```
<?
$str .= chr(27);
$str = sprintf("字符串的结束字符是：%c", 27);
?>
```

chunk_split

将字符串分成小段。
语法：string chunk_split(string string, int [chunklen] , string [end]);
返回值：字符串
内容说明
本函数将字符串变成小段供其他函数使用。例如，base64_encode**。内定是参数** chunklen(76 **个字符)每隔** 76 **个字符插入** end("\r\n")**。返回新字符串而不改动原字符串。**
使用范例
将字符串$data **格式化成** MIME BASE64 **格式**

```
<?
$new_string = chunk_split(base64_encode($data));
?>
```

convert_cyr_string

转换古斯拉夫字符串成其他字符串。
语法：string convert_cyr_string(string str, string from, string to);
返回值：字符串
内容说明
本函数将古斯拉夫字符串转成其他的字符串。from **及** to **二个参数是字符，其代表意义如下：**
k - koi8-r
w - windows-1251
i - iso8859-5
a - x-cp866
d - x-cp866
m - x-mac-cyrillic

Green Apple Data Center

crypt

将字符串用 DES **编码加密。**
语法：string crypt(string str, string [salt]);
返回值：字符串
内容说明
本函数将字符串用 UNIX **的标准加密** DES **模块加密。这是单向的加密函数，无法解密。欲比对字符串，将已加密的字符串的头二个字符放在** salt **的参数中，再比对加密后的字符串。**
在一些较新的 UNIX **版本中，除了** DES **之外还提供了其他的加密模块，如** MD5**。甚至有些系统还用** MD5 **取代** DES**。在** salt **参数还有一些变化，要看传给** salt **参数的字符串长度而定：**
CRYPT_STD_DES-**标准的** DES **编码，输入** 2 **字符的** salt**。**
CRYPT_EXT_DES-**延伸的** DES **编码，输入** 9 **字符的** salt**。**
CRYPT_MD5-MD5 **编码，输入** 12 **字符加上**1**的** salt**。**
CRYPT_BLOWFISH-**延伸的** DES **编码，输入** 16 **字符加上**2**的** salt**。**
此外，若不使用 salt **参数，则程序会自动产生。**

echo

输出字符串。
语法：echo "string arg1, string [argn]...";
返回值：无
内容说明
本函数将字符串输出。由于它不是真正的函数，因此也没有返回值。
使用范例

```
<?php
echo "Hello World";
?>
```

explode

切开字符串。
语法：array explode(string separator, string string);
返回值：数组
内容说明
本函数将字符串依指定的字符串或字符 separator **切开。将切开后的字符串返回到数组变量中。**

使用范例

```
<?
$pizza = "第一片 第二片 第三片 第四片 第五片 第六片";
$pieces = explode(" ", $pizza);
?>
```

flush

清除输出缓冲区。
语法：void flush(void);
返回值：无
内容说明
本函数无输入亦无输出。将输出缓冲区的内容送出，并清除。

Green Apple Data Center

get_meta_tags

抽出文件所有 meta **标记的数据。**
语法：array get_meta_tags(string filename, int [use_include_path]);
返回值：数组
内容说明
本函数将 homepage **中的所有**<meta .....>**标记的取出并放置数组变量返回。例如：**

```
<html>
<head>
<meta name="author" content="彭武兴">
<meta name="title" content="PHP BIBLE">
<title>PHP BIBLE</title>
</head><!-- 本函数处理只到这儿结束 -->
```

本函数找出 meta **标记，**name **属性的字符串为数组索引，而** content **属性字符串则为数组的内容了。值得注意的是本函数为** PHP **原生函数，在** UNIX **系列平台上无法直接处理麦金塔的文件格式，因为换行字符不同。参数** filename **也可以是** URL**，函数将抽出远端服务器的文件** meta **标记。而函数在处理时，遇到**</head>**标记结束，要是**<meta ....>**标记放在**</head>**后将无法处理。**

htmlspecialchars

将特殊字符转成 HTML **格式。**
语法：string htmlspecialchars(string string);
返回值：字符串
内容说明
本函数将特殊字符转成 HTML **的字符串格式(** &....; **)。最常用到的场合可能就是处理客户留言的留言版了。**
& (**和**) **转成** &
" (**双引号**)**转成**"
< (**小于**) **转成** <
> (**大于**) **转成** >
此函数只转换上面的特殊字符，并不会全部转换成 HTML **所定的** ASCII **转换。**

使用范例

```
<FORM ACTION=bla>
<H2>Restaurant Description<H2>
Name of restaurant：
<INPUT TYPE=text NAME="restname" VALUE="<?
        echo HTMLSpecialChars($restname); ?>">
<!-- 变量 $restname 是这样子的 $restname="\"The White Horse\""; -->
<BR>
输入描述 (若您会 HTML，可直接使用)：<BR>
<TEXTAREA NAME="descript"><?
    echo HTMLSpecialChars($descript);
?></TEXTAREA>
<INPUT TYPE=submit>
</FORM>
```

htmlentities

将所有的字符都转成 HTML **字符串。**

Green Apple Data Center

语法：string htmlentities(string string);
返回值：字符串
内容说明
本函数有点像 htmlspecialchars()**函数，但本函数会将所有** string **的字符都转成** HTML **的特殊字集字符串。不过在转换后阅读网页源代码的方面，会有很多困扰，尤其是网页源代码的中文字会变得不知所云，浏览器上看到的还是正常的。**

implode

将数组变成字符串。
语法：string implode(string glue, array pieces);
返回值：字符串
内容说明
本函数将数组的内容组合成一个字符串，参数 glue **是字符之间的分隔符号。**

使用范例

```
<?
$colon_separated = implode("：", $array);
echo $colon_separated;
?>
```

join

将数组变成字符串。
语法：string join(string glue, array pieces);
返回值：字符串
内容说明
本函数是 implode **函数的别名。**

ltrim

去除连续空白。
语法：string ltrim(string str);
返回值：字符串
内容说明
本函数用来删除字符串中的连续空白带(whitespace)**。**

md5

计算字符串的 MD5 **杂凑值。**
语法：string md5(string str);
返回值：字符串
内容说明
本函数用来计算 MD5 **杂凑值。关于** MD5 **编码法，可以参考** RSA Data Security, Inc. MD5 Message-Digest Algorithm. RFC1321**；**

nl2br

将换行字符转成
。
语法：string nl2br(string string);
返回值：字符串
内容说明
本函数将换行字符转换成 HTML **换行的**
指令。

Green Apple Data Center

Ord

返回字符的序数值。
语法：int ord(string string);
返回值：整数
内容说明
本函数返回字符的 ASCII(**美国国家标准交换码**)**序数值。本函数和** chr()**函数相反。**

使用范例

```
<?php
if (ord($str) == 10) {
   echo("字符串\$str 的第一个字是换行字符。\n");
}
?>
```

parse_str

解析 query **字符串成变量。**
语法：void parse_str(string str);
返回值：无
内容说明
本函数可将浏览器返回的 GET **方法的** QUERY_STRING **字符串解析。返回的变量名及值就依** QUERY_STRING **的名称及值。**

使用范例

```
<?php
$str = "first=value&second[]=this+works&second[]=another";
parse_str($str);
echo $first;          // 显示出 "value" 字符串
echo $second[0];   // 显示 "this works" 字符串
echo $second[1];   // 显示 "another" 字符串
?>
```

print

输出字符串。
语法：boolean print(string arg);
返回值：布尔值
内容说明
本函数输出字符串。若成功则返回 1**，失败则返回** 0**。例如传输中途客户的浏览器突然挂了，则会造成输出失败的情形。**

printf

输出格式化字符串。
语法：int printf(string format, mixed [args]...);
返回值：整数
内容说明
本函数依参数 format **指定的内容格式将字符串格式化。格式的细节可以参考** sprintf()**。**

Green Apple Data Center

quoted_printable_decode

将 qp **编码字符串转成** 8 **位字符串。**
语法：string quoted_printable_decode(string str);
返回值：字符串
内容说明
本函数可以将 quoted-printable **后的字符串解码成为** 8 **位编码的字符串。而本函数类似** imap_qprint()**函数，唯一不同的地方是使用** imap_qprint()**函数需要让系统加入** IMAP **的模块，而本函数不需要** IMAP **模块。**

QuoteMeta

加入引用符号。
语法：string quotemeta(string str);
返回值：字符串
内容说明
本函数将字符串中含有 . \\ + * ? [^] ($) **等字符的前面加入反斜线**"\"**符号。**

rawurldecode

从 URL **专用格式字符串还原成普通字符串。**
语法：string rawurldecode(string str);
返回值：字符串
内容说明
本函数将字符串解码。从 URL **的字符串专用格式解成普通字符串。详细的编码解码信息及规格文件可以参考** RFC 1738**。**

rawurlencode

将字符串编码成 URL **专用格式。**
语法：string rawurldecode(string str);
返回值：字符串
内容说明
本函数将字符串编码成 URL **的字符串专用格式，特殊的字符会转换成百分比符号后面加上二个** 16 **位数字的格式。例如，空格就会变成**%20**。**

使用范例

```
<?php
echo '<a href="ftp：//guest：', rawurlencode ('foo @+%/'), '@localhost/x.txt">';
?>
```

setlocale

配置地域化信息。
语法：string setlocale(string category, string locale);
返回值：字符串
内容说明
本函数用来配置地域的信息。参数 category **有下列的选择：**
LC_ALL **包括下面的全部选项都要。**
LC_COLLATE **配置字符串比较，**PHP **目前尚未实现本项。**
LC_CTYPE **配置字符类别及转换。例如全变大写** strtoupper()**。**
LC_MONETARY **配置金融货币，**PHP **目前尚未实现。**
LC_NUMERIC **配置小数点后的位数。**
LC_TIME **配置时间日期格式，与** strftime()**合用。**
而参数 locate **若是空字符串**""**，则会使用系统环境变量的** locate **或是** LANG **的值。若** locate **为零，则不会**

Green Apple Data Center

改变地域化配置，返回新的地域。若系统尚未实现则返回 false。

使用范例

```
<?
setlocale("LC_ALL", "pl");
$net = "1234,56";
$gross = "1,22" * $net;
printf("毛利：%s,  净利：%s", $gross, $net");
?>
```

返回值则为
毛利：1234,56,**净利：**1506,1632

similar_text

计算字符串相似度。
语法：int similar_text(string first, string second, double [percent]);
返回值：整数
内容说明
本函数用来计算比较二字符串的相似程度。

soundex

计算字符串的读音值
语法：string soundex(string str);
返回值：字符串
内容说明
Soundex **值是利用英文字的读音近似值所求得的值，值由四个字符构成，第一个字符为英文字母，后三个为数字。在拼音文字中有时会有会念但不能拼出正确字的情形，特别是在做搜寻引擎时面对用户传入的英文字符串，可用本函数做类似模糊比对的效果。例如** Knuth **和** Kant **二个字符串，它们的** soundex **值都是** H416。

使用范例

```
<?
$str1=soundex("Wilson");
$str2=soundex("Waillsume");
echo "soundex(\"Wilson\")=$str1  等于\n";
echo "soundex(\"Waillsume\")=$str2\n";
echo "值均为  $str1";
?>
```

上例返回的字符串为
soundex("Wilson")=W425 **等于**
soundex("Waillsume")=W425
值均为 W425

sprintf

将字符串格式化。
语法：string sprintf(string format, mixed [args]...);
返回值：字符串
内容说明
本函数用来将字符串格式化。参数 format **是转换的格式，以百分比符号%开始到转换字符为止。而在转换**

Green Apple Data Center

的格式间依序包括了：

（1）填空字符。0 表示空格填 0；空格是默认值，表示空格。

（2）对齐方式。默认值为向右对齐，负号表向左对齐。

（3）字段宽度。为最小宽度。

（4）精确度。指在小数点后的浮点数位数。

类型如下：,

% 印出百分比符号，不转换。

b 整数转成二进位。

c 整数转成对应的 ASCII 字符。

d 整数转成十进位。

f 双精确度数字转成浮点数。

o 整数转成八进位。

s 整数转成字符串。

x 整数转成小写十六进位。

X 整数转成大写十六进位。

使用范例

```
<?
$money1 = 68.75;
$money2 = 54.35;
$money = $money1 + $money2;
// 此时变量$money 值为"123.1";
$formatted = sprintf ("%01.2f", $money);
// 此时变量$formatted 值为"123.10"
?>
```

strchr

寻找第一个出现的字符。

语法：string strchr(string haystack, string needle);

返回值：字符串

内容说明

本函数也就是 strstr()函数。

strcmp

字符串比较。

语法：int strcmp(string str1, string str2);

返回值：整数

内容说明

本函数用来比较二字符串的大小。返回负数表示 str1 小于 str2；返回正数表示 str1 大于 str2；返回零表示二字符串相同。

strcspn

不同字符串的长度。

语法：int strcspn(string str1, string str2);

返回值：整数

内容说明

本函数用来比较二字符串并计算出不同的字符串长度。

strip_tags

去掉 HTML 及 PHP 的标记。

Green Apple Data Center

语法：string strip_tags(string str);
返回值：字符串
内容说明
本函数可去掉字符串中包含的任何 HTML **及** PHP **的标记字符串。若是字符串的** HTML **及** PHP **标签原来就有错，例如少了大于符号，则也会返回错误。而本函数和** fgetss()**有着相同的功能。**

StripSlashes

去掉反斜线字符。
语法：string stripslashes(string str);
返回值：字符串
内容说明
本函数可去掉字符串中的反斜线字符。若是连续二个反斜线，则去掉一个，留下一个。若只有一个反斜线，就直接去掉。

strlen

取得字符串长度。
语法：int strlen(string str);
返回值：整数
内容说明
本函数返回指定的字符串长度。

strrpos

寻找字符串中某字符最后出现处。
语法：int strrpos(string haystack, char needle);
返回值：整数
内容说明
本函数用来寻找字符串 haystack **中的字符** needle **最后出现的位置。值得注意的是** needle **只能是一个字符，中文字等就不适合了。若找不到指定的字符，则返回** false **值。**

strpos

寻找字符串中某字符最先出现处。
语法：int strpos(string haystack, string needle, int [offset]);
返回值：整数
内容说明
本函数用来寻找字符串 haystack **中的字符** needle **最先出现的位置。值得注意的是** needle **只能是一个字符，中文字等就不适合了。若找不到指定的字符，则返回** false **值。参数** offset **可省略，用来表示从** offset **开始找。**

strrchr

取得某字符最后出现处起的字符串。
语法：string strrchr(string haystack, string needle);
返回值：整数
内容说明
本函数用来寻找字符串 haystack **中的字符** needle **最后出现位置，并将此位置起至字符串** haystack **结束之间的字符串返回。若没有找到** needle **则返回** false**。**
使用范例
下例取回环境变量 PATH **之最后一个路径**

```
<?php
$dir = substr( strrchr( $PATH, "：" ), 1 );
echo "最后的路径为：".$dir;
?>
```

Green Apple Data Center

strrev

颠倒字符串。
语法：string strrev(string string);
返回值：字符串
内容说明
将字符串前后颠倒。

使用范例
下例的返回字符串为"gneP nosliW"

```
<?
$str=strrev("Wilson Peng");
echo $str;
?>
```

strspn

找出某字符串落在另一字符串掩码的数目。
语法：int strspn(string str1, string str2);
返回值：整数
内容说明
本函数将 str2 **字符串当掩码，可用来计算** str1 **字符串中有几个字符落在** str2 **中。**

strstr

返回字符串中某字符串开始处至结束的字符串。
语法：string strstr(string haystack, string needle);
返回值：字符串
内容说明
本函数将 needle **最先出现在** haystack **处起至** haystack **结束的字符串返回。若找不到** needle **则返回** false**。**

strtok

切开字符串。
语法：string strtok(string arg1, string arg2);
返回值：字符串
内容说明
本函数将字符串 arg1 **依字符串** arg2 **的值切开成小段小段的字符串。**

使用范例
本例将 I will be back **字符串依空白切开。**

```
<?php
$string = "I will be back";
$tok = strtok($string," ");
while($tok) {
      echo "单字=$tok<br>";
      $tok = strtok(" ");
}
?>
```

Green Apple Data Center

strtolower

字符串全转为小写。
语法：string strtolower(string str);
返回值：字符串
内容说明
本函数将字符串 str **全部变小写字符串。**

strtoupper

字符串全转为大写。
语法：string strtoupper(string str);
返回值：字符串
内容说明
本函数将字符串 str **全部变大写字符串。**

str_replace

字符串取代。
语法：string str_replace(string needle, string str, string haystack);
返回值：字符串
内容说明
本函数将字符串 str **代入** haystack **字符串中，将所有的** needle **置换成** str**。**

使用范例
下例将%body%**以** black **取代**

```
<?php
$bodytag = str_replace("%body%", "black", "<body text=%body%>");
echo $bodytag;
?>
```

strtr

转换某些字符。
语法：string strtr(string str, string from, string to);
返回值：字符串
内容说明
本函数将字符串 str **中和** from **有关的字符一一转成** to **的字符。**

substr

取部分字符串。
语法：string substr(string string, int start, int [length]);
返回值：字符串
内容说明
本函数将字符串 string **的第** start **位起的字符串取出** length **个字符。若** start **为负数，则从字符串尾端算起。若可省略的参数** length **存在，但为负数，则表示取到倒数第** length **个字符。**

使用范例

```
<?
echo substr("abcdef", 1, 3);    // 返回 "bcd"
echo substr("abcdef", -2);      // 返回 "ef"
echo substr("abcdef", -3, 1); // 返回 "d"
```

Green Apple Data Center

```
echo substr("abcdef", 1, -1); // 返回 "bcde"
?>
```

trim

截去字符串首尾的空格。
语法：string trim(string str);
返回值：字符串
内容说明
本函数返回字符串 string **首尾的空白字符去除后的字符串。**

ucfirst

将字符串第一个字符改大写。
语法：string ucfirst(string str);
返回值：字符串
内容说明
本函数返回字符串 str **第一个字的字首字母改成大写。**

ucwords

将字符串每个字第一个字母改大写。
语法：string ucwords(string str);
返回值：字符串
内容说明
本函数返回字符串 str **每个字的字首字母全都改成大写。**

4.3 变量处理函数库

本函数库共有 17 **个函数，他们是：**
gettype**：取得变量的类型。**
intval**：变量转成整数类型。**
doubleval**：变量转成双精度类型。**
empty**：判断变量是否已配置。**
is_array**：判断变量类型是否为数组类型。**
is_double**：判断变量类型是否为双精度类型。**
is_float**：判断变量类型是否为浮点数类型。**
is_int**：判断变量类型是否为整数类型。**
is_integer**：判断变量类型是否为长整数类型。**
is_long**：判断变量类型是否为长整数类型。**
is_object**：判断变量类型是否为对象类型。**
is_real**：判断变量类型是否为实数类型。**
is_string**：判断变量类型是否为字符串类型。**
isset**：判断变量是否已配置。**
settype**：配置变量类型。**
strval**：将变量转成字符串类型。**
unset**：删除变量。**

gettype

取得变量的类型。
语法：string gettype(mixed var);
返回值：字符串
内容说明
本函数用来取得变量的类型。返回的类型字符串可能为下列字符串其中之一：integer、double、string、array、

Green Apple Data Center

object、unknown type。

intval

变量转成整数类型。
语法：int intval(mixed var, int [base]);
返回值：整数
内容说明
本函数可将变量转成整数类型。可省略的参数 base **是转换的基底，默认值为** 10**。转换的变量** var **可以为数组或对象之外的任何类型变量。**

doubleval

变量转成双精度类型。
语法：double doubleval(mixed var);
返回值：双精度
内容说明
本函数可将变量转成双精度类型。转换的变量 var **可以为数组或类之外的任何类型变量。**

empty

判断变量是否已配置。
语法：int empty(mixed var);
返回值：整数
内容说明
本函数用来测试变量是否已经配置。若变量已存在、非空字符串或者非零，则返回 false **值；反之返回** true。

is_array

判断变量类型是否为数组类型。
语法：int is_array(mixed var);
返回值：整数
内容说明
若变量为数组类型则返回 true**，否则返回** false。

is_double

判断变量类型是否为双精度类型。
语法：int is_double(mixed var);
返回值：整数
内容说明
若变量为双精度类型则返回 true**，否则返回** false。

is_float

判断变量类型是否为浮点数类型。
语法：int is_float(mixed var);
返回值：整数
内容说明
若变量为浮点数类型则返回 true**，否则返回** false。

is_int

判断变量类型是否为整数类型。
语法：int is_int(mixed var);
返回值：整数
内容说明
若变量为整数类型则返回 true**，否则返回** false。

Green Apple Data Center

is_integer

判断变量类型是否为长整数类型。
语法：int is_integer(mixed var);
返回值：整数
内容说明
若变量为长整数类型则返回 true，**否则返回** false。**本函数其实就是** is_long()。

is_long

判断变量类型是否为长整数类型。
语法：int is_long(mixed var);
返回值：整数
内容说明
若变量为长整数类型则返回 true，**否则返回** false。

is_object

判断变量类型是否为对象类型。
语法：int is_object(mixed var);
返回值：整数
内容说明
若变量为对象类型则返回 true，**否则返回** false。

is_real

判断变量类型是否为实数类型。
语法：int is_real(mixed var);
返回值：整数
内容说明
若变量为实数类型则返回 true，**否则返回** false。

is_string

判断变量类型是否为字符串类型。
语法：int is_string(mixed var);
返回值：整数
内容说明
若变量为字符串类型则返回 true，**否则返回** false。

isset

判断变量是否已配置。
语法：int isset(mixed var);
返回值：整数
内容说明
本函数用来测试变量是否已经配置。若变量已存在则返回 true **值。其他情形返回** false **值。**
使用范例

```
<?php
$a = "test";
echo isset($a); // true
unset($a);
echo isset($a); // false
?>
```

Green Apple Data Center

settype

配置变量类型。
语法：int settype(string var, string type);
返回值：整数
内容说明
本函数用来配置或转换变量类型。成功返回 true **值，其他情形返回** false **值。参数** var **为原来的变量名，参数** type **为下列的类型之一：**integer、double、string、array **与** object**。**

strval

将变量转成字符串类型。
语法：string strval(mixed var);
返回值：字符串
内容说明
本函数可将数组及类之外的变量类型转换成字符串类型。

4.4 数学运算函数库

本函数库共有 33 **个函数**
本函数库能处理的数值范围只能达到长整数与双精度的范围。若要处理超过上述范围的数值，要使用 BC **高精确度函数库。本函数库定义了圆周率的常量** M_PI **值为** 3.14159265358979323846**。数学运算函数库包括：**

Abs**：取得绝对值。**
Acos**：取得反余弦值。**
Asin**：取得反正弦值。**
Atan**：取得反正切值。**
Atan2**：计算二数的反正切值。**
base_convert**：转换数字的进位方式。**
BinDec**：二进位转成十进位。**
Ceil**：计算大于指定数的最小整数。**
Cos**：余弦计算。**
DecBin**：十进位转二进位。**
DecHex**：十进位转十六进位。**
DecOct**：十进位转八进位。**
Exp**：自然对数** e **的次方值。**
Floor**：计算小于指定数的最大整数。**
getrandmax**：随机数的最大值。**
HexDec**：十六进位转十进位。**
Log**：自然对数值。**
Log10**：**10 **为基底的对数值。**
max**：取得最大值。**
min**：取得最小值。**
mt_rand**：取得随机数值。**
mt_srand**：配置随机数种子。**
mt_getrandmax**：随机数的最大值。**
number_format**：格式化数字字符串。**
OctDec**：八进位转十进位。**
pi**：圆周率。**
pow**：次方。**

Green Apple Data Center

rand：**取得随机数值。**
round：**四舍五入。**
Sin：**正弦计算。**
Sqrt：**开平方根。**
srand：**配置随机数种子。**
Tan：**正切计算。**

Abs

取得绝对值。
语法：mixed abs(mixed number);
返回值：混合类型
内容说明
返回参数 number **的绝对值。若** number **是双精度数，则返回值也是双精度数；其他的类型则返回类型为整数。**

Acos

取得反余弦值。
语法：float acos(float arg);
返回值：浮点数
内容说明
返回参数 arg **的反余弦值**(arc cosine)**。**
参考
asin()　atan()

Asin

取得反正弦值。
语法：float asin(float arg);
返回值：浮点数
内容说明
返回参数 arg **的反正弦值**(arc sine)**。**
参考
acos()　atan()

Atan

取得反正切值。
语法：float atan(float arg);
返回值：浮点数
内容说明
返回参数 arg **的反正切值**(arc tangent)**。**
参考
acos()　asin()

Atan2

计算二数的反正切值。
语法：float atan2(float y, float x);
返回值：浮点数
函数种类：数学运算

Green Apple Data Center

内容说明

参考

acos() atan() asin()

base_convert

转换数字的进位方式。

语法：string base_convert(string number, int frombase, int tobase);

返回值：字符串

内容说明

本函数将数字字符串 number **从以** frombase **进位转换到以** tobase **进位。本函数能够处理由以二进位到以三十六进位之间的进位方式。在十进位之前都是以数字表示，而在超过十进位之后就用英文字母表示。例如十六进位个位数依序为** 123456789abcdef，10 **的顺序是第十七个，这时才进一位。而三十六进位** a **是第十个、**b **为第十一个、**z **为第三十六个、**10 **是第三十七个，这时才进位。**

使用范例

本例将十六进位字符串转成二进位字符串

```
<?php
$binary = base_convert($hexadecimal, 16, 2);
echo "十六进位字符串\"$hexadecimal\"转成二进位为\"$binary\"。";
?>
```

BinDec

二进位转成十进位。

语法：int bindec(string binary_string);

返回值：整数

内容说明

本函数将二进位数字字符串转成十进位的整数。由于 PHP **使用** 32 **位有正负号整数计算，能处理的最大十进位数字为** 2147483647，**也就是二进制数字的** 11111111111111111111 1111111111(31 **个** 1)。

参考

DecBin()

Ceil

计算大于指定数的最小整数。

语法：int ceil(float number);

返回值：整数

内容说明

本函数用来计算比浮点参数 number **大的最小整数。**

使用范例

本例返回值为 4。

```
<?php
$nextint=ceil(3.14);
echo $nextint;
?>
```

Green Apple Data Center

参考

Floor()　round()

Cos

余弦计算。

语法：float cos(float arg);

返回值：浮点数

内容说明

本函数计算参数 arg **的余弦值**(cosine)**。**

参考

Sin()　Tan()

DecBin

十进位转二进位。

语法：string decbin(int number);

返回值：字符串

内容说明

本函数将十进位数字转成二进位字符串。由于 PHP **使用** 32 **位有正负号整数计算，能处理的最大十进位数字为** 2147483647**，也就是二进位数字的** 11111111111111111111 1111111111(31 **个** 1)**。**

参考

BinDec()

DecHex

十进位转十六进位。

语法：string dechex(int number);

返回值：字符串

内容说明

本函数将十进位数字转成十六进位字符串。由于 PHP **使用** 32 **位有正负号整数计算，能处理的最大十进位数字为** 2147483647**，也就是十六进位数字** 7fffffff**。**

参考

HexDec()

DecOct

十进位转八进位。

语法：string decoct(int number);

返回值：字符串

内容说明

本函数将十进位数字转成八进位字符串。由于 PHP **使用** 32 **位有正负号整数计算，能处理的最大十进位数字为** 2147483647**，也就是八进位数字** 17777777777**。**

参考

OctDec()

Exp

自然对数 e **的次方值。**

语法：float exp(float arg);

返回值：浮点数

Green Apple Data Center

内容说明

本函数计算自然对数(natural logarithm)**的** arg **次方值。**

参考

pow()　Log()

Floor

计算小于指定数的最大整数。

语法：int floor(float number);

返回值：整数

内容说明

本函数用来计算比浮点参数 number **小的最大整数。**

使用范例

本例返回值为 3**。**

```
<?php
$lastint=floor(3.14);
echo $lastint;
?>
```

参考

Ceil()　round()

getrandmax

随机数的最大值。

语法：int getrandmax(void);

返回值：整数

内容说明

本函数计算随机数函数 rand()**可能取得的取得的最大随机数值。本函数不需参数。**

参考

rand()　srand()　mt_rand()　mt_srand()　mt_getrandmax()

HexDec

十六进位转十进位。

语法：int hexdec(string hex_string);

返回值：整数

内容说明

本函数将十六进位字符串转成十进位数字。由于 PHP **使用** 32 **位有符号整数计算，能处理的最大十六进位数字为** 7fffffff**，也就是十进位数字的** 2147483647**。**

参考

DecHex()

Log

自然对数值。

语法：float log(float arg);

返回值：浮点数

Green Apple Data Center

内容说明

本函数计算参数 arg **的自然对数**(natural logarithm)**值。**

Log10

10 **为基底的对数值。**

语法：float log10(float arg);

返回值：浮点数

内容说明

本函数计算参数 arg **的常用对数值。**

max

取得最大值。

语法：mixed max(mixed arg1, mixed arg2 .... mixed argn);

返回值：混合类型

内容说明

本函数计算参数间的最大值。若第一个参数是数字数组，则会找出该数组的最大数字。若第一个参数非数组，则需二个以上的参数。这些数字可以是整数、倍精确数或数字字符串的类型。参数的数目不限，视用户的需求而定。在计算时，只要有一个参数是倍精确数，本函数会将所有的参数都转成倍精确数，并返回倍精确数。若参数只有整数及数字字符串，则会将所有的参数转换成整数，并返回整数。

参考

min()

min

取得最小值。

语法：mixed min(mixed arg1, mixed arg2 .... mixed argn);

返回值：混合类型

内容说明

本函数计算参数间的最小值。若第一个参数是数字数组，则会找出该数组的最小数字。若第一个参数非数组，则需二个以上的参数。这些数字可以是整数、倍精确数或数字字符串的类型。参数的数目不限，视用户的需求而定。在计算时，只要有一个参数是倍精确数，本函数会将所有的参数都转成倍精确数，并返回倍精确数。若参数只有整数及数字字符串，则会将所有的参数转换成整数，并返回整数。

参考

max()

mt_rand

取得随机数值。

语法：int mt_rand([int min], [int max]);

返回值：整数

内容说明

本函数不使用一般常用的 libc **来计算随机数值，而是使用计算速度至少快四倍的马其赛特旋转**(Mersenne Twister)**演算法来计算随机数值。有关马特赛特旋转演算法可在松本真的** http：//www.math.keio.ac.jp/~matumoto/emt.html **找到更多的相关信息，最佳化的原始程序则在** http：//www.scp.syr.edu/~marc/hawk/twister.html**。若没有指定随机数的最大及最小范围，本函数会自动的从** 0 **到** RAND_MAX **中取一个随机数。若有指定** min **及** max **的参数，则从指定参数中取一个数字，例如** mt_rand(38,49)**则会从** 38 **到** 49 **之间取一个随机数值。值得注意的是为使随机数的随机度最大，每次在取随机数之前最好使用** mt_srand()**以配置新的随机数种**

Green Apple Data Center

子。

参考

rand() srand() getrandmax() mt_srand() mt_getrandmax()

mt_srand

配置随机数种子。

语法：void mt_srand(int seed);

返回值：无

内容说明

本函数传入参数 seed **后，配置随机数的种子。值得注意的是参数** seed **值最好也是随机出现的数字，例如利用加入时间做为变量的来源就是不错的方法。**

使用范例

本例加入时间的因素，以执行时的百万分之一秒当随机数种子

```
<?php
mt_srand((double)microtime()*1000000);
$randval = mt_rand();
echo $randval;
?>
```

参考

rand() srand() getrandmax() mt_rand() mt_getrandmax()

mt_getrandmax

随机数的最大值。

语法：int mt_getrandmax(void);

返回值：整数

内容说明

本函数计算随机数函数 mt_rand()**可能取得的最大随机数值。本函数不需参数。**

参考

rand() srand() getrandmax() mt_srand() mt_rand()

number_format

格式化数字字符串。

语法：string number_format(float number,int[decimals],string[dec_point],string[thousands _sep]);

返回值：字符串

内容说明

本函数用来将浮点参数 number **格式化。若没加参数** decimals **则返回的字符串只要整数部分，加了此参数才依参数指定的小数点位数返回。参数** dec_point **表示小数点的表示方法，默认值是**"."**，若需要转换成其他的小数点就可以在这个参数修改。参数** thousands_sep **为整数部分每三位的分隔符号，默认值是**","**。本函数最特别的地方就是参数数目，最少要有一个，也就是欲格式化的字符串；也可以有二个或者四个参数，但不能用三个参数。值得注意的是指定小数点的位数之后的数字直接舍弃，没有四舍五入的情形。**

使用范例

```
<?
```

Green Apple Data Center

```
$short_pi = "3.14159";
$my_pi = number_format($short_pi, 2);
echo $my_pi."\n";      // 3.14
$foo = 850017.9021;
$new_foo = number_format($foo, 3, ".", " ");
echo $new_foo."\n";    // 850 017.902
?>
```

OctDec

八进位转十进位。
语法：string decoct(int number);
返回值：字符串
内容说明
本函数将八进位字符串转成十进位数字。由于 PHP **使用** 32 **位有符号整数计算，能处理最大的八进位数字为** 17777777777**，也就是十进位数字的** 2147483647**。**
参考
DecOct()

pi

圆周率。
语法：double pi(void);
返回值：倍精确数
内容说明
本函数返回圆周率。不需输入参数。

pow

次方。
语法：float pow(float base, float exp);
返回值：浮点数
内容说明
本函数计算次方值。参数 base **为基底，**exp **为幂数。**

使用范例

```
<?php
print(pow(2,3));    // 8
print(pow(5,4));    // 625
?>
```

参考
Log10() Exp()

rand

取得随机数值。
语法：int rand([int min], [int max]);
返回值：整数
内容说明
本函数用来取得随机数值。若没有指定随机数的最大及最小范围，本函数会自动的从 0 **到** RAND_MAX **中**

Green Apple Data Center

取一个随机数。若有指定 min **及** max **的参数，则从指定参数中取一个数字。例如** rand(38,49)**则会从** 38 **到** 49 **之间取一个随机数值，**UNIX **系统包含** 49**、**Win32 **系统不包含** 49(zkimmel@earthlink.net 10-May-1999)**。值得注意的是为使随机数的随机度最大，每次在取随机数之前最好使用** srand()**以配置新的随机数种子。**

参考

srand() getrandmax() mt_rand() mt_srand() mt_getrandmax()

round

四舍五入。

语法：double round(double val);

返回值：倍精确数

内容说明

本函数用来将数字小数点后四舍五入。

使用范例

```
<?php
$foo1 = round(3.4);
$foo2 = round(3.5);
$foo3 = round(3.6);
echo "round(3.4)：".$foo1."<br>\n";
echo "round(3.5)：".$foo2."<br>\n";
echo "round(3.6)：".$foo3;
?>
```

参考

Ceil() Floor()

Sin

正弦计算。

语法：float sin(float arg);

返回值：浮点数

内容说明

本函数计算参数 arg **的正弦值**(sine)**。**

参考

Cos() Tan()

Sqrt

开平方根。

语法：float sqrt(float arg);

返回值：浮点数

内容说明

本函数将参数 arg **开平方。**

srand

配置随机数种子。

语法：void srand(int seed);

返回值：无

内容说明

Green Apple Data Center

本函数传入参数 seed **后，配置随机数的种子。值得注意的是参数** seed **值最好也是随机出现的数字，例如利用加入时间做为变量的来源就是不错的方法。**

使用范例

本例加入时间的因素，以执行时的百万分之一秒当随机数种子

```
<?php
srand((double)microtime()*1000000);
$randval = rand();
echo $randval;
?>
```

参考

rand()　getrandmax()　mt_srand()　mt_rand()　mt_getrandmax()

Tan

正切计算。

语法：float tan(float arg);

返回值：浮点数

内容说明

本函数计算参数 arg **的正切值**(tangent)**。**

参考

Sin()　Cos()

```
$result = odbc_prepare ($conn, $sql);
odbc_setoption ($result, 2, 0, 30);
odbc_execute ($result);
?>
```

4.5　FTP **文件传输函数库**

本函数库共有 20 **个函数，**FTP **的全名为** File Transfer Protocol**，也就是文件传输协议。利用本函数库可以让** PHP **也能处理** FTP **的相关功能。本函数库需要** PHP 3.0.13 **版之后才支持。本函数包括：**

ftp_connect**：打开** FTP **链接。**

ftp_login**：登入** FTP **服务器。**

ftp_pwd**：取得目前所在路径。**

ftp_cdup**：回上层目录。**

ftp_chdir**：改变路径。**

ftp_mkdir**：建新目录。**

ftp_rmdir**：删除目录。**

ftp_nlist**：列出指定目录中所有文件。**

ftp_rawlist**：详细列出指定目录中所有文件。**

ftp_systype**：显示服务器系统。**

ftp_pasv**：切换主被动传输模式。**

ftp_get**：下载文件。**

ftp_fget**：下载文件，并存在已开的档中。**

ftp_put**：上传文件。**

ftp_fput**：上传已打开文件。**

ftp_size**：获得指定文件的大小。**

Green Apple Data Center

ftp_mdtm：获得指定文件的最后修改时间。
ftp_rename：将文件改名。
ftp_delete：将文件删除。
ftp_quit：关闭 FTP 链接。

ftp_connect

打开 FTP 链接。
语法：int ftp_connect(string host, int [port]);
返回值：整数
内容说明
本函数可打开 FTP 服务器的链接。参数 host 为 FTP 服务器的网址。参数 port 通常省略，若 FTP 服务器的埠号(port)不是 21 时才需要加本参数。若无错误则返回链接代码，失败则返回 false 值。
参考
ftp_quit()

ftp_login

登入 FTP 服务器。
语法：boolean ftp_login(int ftp_stream, string username, string password);
返回值：布尔值
内容说明
本函数可登入已链接的 FTP 服务器。参数 ftp_stream 为 FTP 的链接代码。参数 username 及 password 分别为服务器的使用者帐号及密码，通常 anonymous 为公开的使用帐号，密码则为 Email。成功则返回 true 值。

ftp_pwd

取得目前所在路径。
语法：string ftp_pwd(int ftp_stream);
返回值：字符串
内容说明
本函数用来取得目前在 FTP 服务器中的路径。参数 ftp_stream 为 FTP 的链接代码。若有错误则返回 NULL 值。

ftp_cdup

回上层目录。
语法：boolean ftp_cdup(int ftp_stream);
返回值：布尔值
内容说明
本函数用来回到上层目录，也就是目前目录的父目录。参数 ftp_stream 为 FTP 的链接代码。成功则返回 true 值。

ftp_chdir

改变路径。
语法：boolean ftp_chdir(int ftp_stream, string directory);
返回值：布尔值
内容说明
本函数用来改变路径。参数 ftp_stream 为 FTP 的链接代码。参数 directory 为欲前往的目录。成功则返回 true 值，失败则返回 false 值。

Green Apple Data Center

ftp_mkdir

建新目录。

语法：string ftp_mkdir(int ftp_stream, string directory);

返回值：字符串

内容说明

本函数用来建立新的目录。参数 ftp_stream **为** FTP **的链接代码。参数** directory **为欲建立的新目录。成功则返回已建立的目录名，失败则返回** false **值。**

ftp_rmdir

删除目录。

语法：boolean ftp_chdir(int ftp_stream, string directory);

返回值：布尔值

内容说明

本函数用来删除空目录。参数 ftp_stream **为** FTP **的链接代码。参数** directory **为欲删除的目录。成功则返回** true **值，失败则返回** false **值。**

ftp_nlist

列出指定目录中所有文件。

语法：array ftp_nlist(int ftp_stream, string directory);

返回值：数组

内容说明

本函数用来列出指定路径中的所有文件名称。参数 ftp_stream **为** FTP **的链接代码。参数** directory **为指定的目录。成功则返回文件名称的数组，失败则返回** false **值。**

ftp_rawlist

详细列出指定目录中所有文件。

语法：array ftp_rawlist(int ftp_stream, string directory);

返回值：数组

内容说明

本函数可详细列出指定路径中的所有文件名称。参数 ftp_stream **为** FTP **的链接代码。参数** directory **为指定的目录。成功则返回文件名称的数组，失败则返回** false **值。**

ftp_systype

显示服务器系统。

语法：string ftp_systype(int ftp_stream);

返回值：字符串

内容说明

本函数可显示远端 FTP **服务器的系统，也就等于对** FTP **服务器下** system **或** syst **指令。参数** ftp_stream **为** FTP **的链接代码。成功则返回字符串，如：**"215 UNIX Type：L8"**，失败则返回** false **值。**

ftp_pasv

切换主被动传输模式。

语法：boolean ftp_systype(int ftp_stream);

返回值：布尔值

内容说明

Green Apple Data Center

本函数可以切换成主动传输或者被动传输模式，也就等于对 FTP 服务器下 passive 或 pass 指令。参数 ftp_stream 为 FTP 的链接代码。成功则返回 true 值，失败则返回 false 值。

ftp_get

下载文件。

语法：boolean ftp_get(int ftp_stream, string local_file, string remote_file, int mode);

返回值：布尔值

内容说明

本函数用来下载指定的文件。参数 ftp_stream 为 FTP 的链接代码。参数 local_file 为欲存成本地端的文件名。参数 remote_file 为欲下载的文件名。参数 mode 的值有 FTP_ASCII 及 FTP_BINARY 二种，分别表示文本文件或者是二进制文件。成功则返回 true 值，失败则返回 false 值。

ftp_fget

下载文件，并存在已开的文件中。

语法：boolean ftp_fget(int ftp_stream, int fp, string remote_file, int mode);

返回值：布尔值

内容说明

本函数用来下载指定的文件。参数 ftp_stream 为 FTP 的链接代码。参数 fp 为本地端的已开文件的文件指针。参数 remote_file 为欲下载的文件名。参数 mode 的值有 FTP_ASCII 及 FTP_BINARY 二种，分别表示文本文件或者是二进制文件。成功则返回 true 值，失败则返回 false 值。

ftp_put

上传文件。

语法：boolean ftp_put(int ftp_stream, string remote_file, string local_file, int mode);

返回值：布尔值

内容说明

本函数用来上传指定的文件。参数 ftp_stream 为 FTP 的链接代码。参数 remote_file 为欲存在远端的文件名。参数 local_file 为欲上传文件的文件名。参数 mode 的值有 FTP_ASCII 及 FTP_BINARY 二种，分别表示文本文件或者是二进制文件。成功则返回 true 值，失败则返回 false 值。

ftp_fput

上传已打开文件。

语法：boolean ftp_fput(int ftp_stream, string remote_file, int fp, int mode);

返回值：布尔值

内容说明

本函数用来上传指定的文件。参数 ftp_stream 为 FTP 的链接代码。参数 remote_file 为欲存在远端的文件名。参数 fp 为欲上传的已打开文件的文件指针。参数 mode 的值有 FTP_ASCII 及 FTP_BINARY 二种，分别表示文本文件或者是二进制文件。成功则返回 true 值，失败则返回 false 值。

ftp_size

获得指定文件的大小。

语法：int ftp_size(int ftp_stream, string remote_file);

返回值：整数

内容说明

本函数用来获取 FTP 服务器上指定文件的大小。参数 ftp_stream 为 FTP 的链接代码。参数 remote_file 为欲获取大小的文件名。返回值为文件大小，失败则返回-1 值。

Green Apple Data Center

ftp_mdtm

获得指定文件的最后修改时间。

语法：int ftp_mdtm(int ftp_stream, string remote_file);

返回值：整数

内容说明

本函数用来获取 FTP **服务器上指定文件的最后修改时间。参数** ftp_stream **为** FTP **的链接代码。参数** remote_file **为欲获取修改时间的文件名。返回值为** UNIX **的时间格式**(timestamp)**，失败则返回**-1 **值。**

ftp_rename

将文件改名。

语法：boolean ftp_rename(int ftp_stream, string from, string to);

返回值：布尔值

内容说明

本函数可将远端 FTP **服务器的文件改名字，值得注意的是权限不符时无法改动。参数** ftp_stream **为** FTP **的链接代码。参数** from **为原来的文件名。参数** to **为欲改的新文件名。成功则返回** true **值，失败则返回** false **值。**

ftp_delete

将文件删除。

语法：boolean ftp_delete(int ftp_stream, string remote_file);

返回值：布尔值

内容说明

本函数可将远端 FTP **服务器的文件删除，若是权限不符则无法删除。参数** ftp_stream **为** FTP **的链接代码。参数** remote_file **为欲删除的文件名。成功则返回** true **值，失败则返回** false **值。**

ftp_quit

关闭 FTP **链接。**

语法：boolean ftp_quit(int ftp_stream);

返回值：布尔值

内容说明

本函数用来将远端 FTP **服务器链接关闭。参数** ftp_stream **为** FTP **的链接代码。成功则返回** true **值，失败则返回** false **值。**

4.6 网络函数库

本函数库共有 13 **个函数**

fsockopen**：打开网络的** Socket **链接。**

pfsockopen**：永久打开网络的** Socket **链接。**

set_socket_blocking**：切换阻塞与无阻塞模式。**

gethostbyaddr**：返回机器名称。**

gethostbyname**：返回** IP **地址。**

gethostbynamel**：返回机器名称的所有** IP**。**

checkdnsrr**：检查指定网址的** DNS **记录。**

getmxrr**：取得指定网址** DNS **记录之** MX **字段。**

openlog**：打开系统记录。**

syslog**：记录至系统记录。**

Green Apple Data Center

closelog：关闭系统记录。

debugger_on：使用内建的 PHP 调试器。

debugger_off：关闭内建的 PHP 调试器。

fsockopen

打开网络的 Socket 链接。

语法：int fsockopen(string hostname, int port, int [errno], string [errstr], int [timeout]);

返回值：整数

内容说明

目前这个函数提供二个 Socket 数据流界面，分别为 Internet 用的 AF_INET 及 Unix 用的 AF_UNIX。当在 Internet 情形下使用时，参数 hostname 及 port 分别代表网址及端口。在 UNIX 情形可做 IPC，hostname 参数表示到 socket 的路径，port 配置为 0。可省略的 timeout 选项表示多久没有连上就中断。在使用本函数之后会返回文件指针，供文件函数使用，包括 fgets()、fgetss()、fputs()、fclose()与 feof()。参数 errno 及 errstr 也是可省略的，主要当做错误处理使用。使用本函数，会使用阻塞模式(blocking mode)处理，可用 set_socket_blocking()转换成无阻塞模式。

使用范例

本例用来模拟 HTTP 链接。

```
<?php
$fp = fsockopen("php.wilson.gs", 80, &$errno, &$errstr, 10);
if(!$fp) {
        echo "$errstr ($errno)<br>\n";
} else {
        fputs($fp,"GET / HTTP/1.0\nHost：php.wilson.gs\n\n");
        while(!feof($fp)) {
                echo fgets($fp,128);
        }
        fclose($fp);
}
?>
```

pfsockopen

永久打开网络的 Socket 持续链接。

语法：int pfsockopen(string hostname, int port, int [errno], string [errstr], int [timeout]);

返回值：整数

内容说明

本函数和 fsockopen()类似，但本函数在 PHP 程序结束时，不会将网络 socket 链接关闭，仍保持链接。用这种方式，可以增加效率，但较耗系统资源。

set_socket_blocking

切换阻塞与无阻塞模式。

语法：int set_socket_blocking(int socket descriptor, int mode);

返回值：整数

内容说明

若参数 mode 值为 false，会将 socket 切换到无阻塞模式(non-blocking mode)；若 mode 值为 true，则切换成阻塞模式。当使用 fgets()等函数读取 socket 时，无法确定返回时间，用无阻塞模式可使继续进行，不会因无法

Green Apple Data Center

读取而阻塞。

gethostbyaddr

返回机器名称。
语法：string gethostbyaddr(string ip_address);
返回值：字符串
内容说明
本函数可返回某个 IP **地址的机器名称**(Domain Name)**。若执行失败，则返回原来的** IP **地址。**
使用范例
下例的返回值为 dns.tsinghua.net.cn

```
<?
echo gethostbyaddr("202.166.255.97");
?>
```

gethostbyname

返回 IP **地址。**
语法：string gethostbyname(string hostname);
返回值：字符串
内容说明
本函数可返回某个机器名称(Domain Name)**的** IP **地址**(IP Address)**。若执行失败，则返回原来的机器名称。**
使用范例
下例的返回值为 140.137.33.246

```
<?
echo gethostbyaddr("www.tsinghua.edu.cn");
?>
```

gethostbynamel

返回机器名称的所有 IP**。**
语法：array gethostbynamel(string hostname);
返回值：数组
内容说明
若一个机器名称有很多个 IP **地址(例如一些** FTP **或是** WWW **网站)，使用本函数可以取得全部的** IP **地址，返回到数组变量中。**
使用范例
本范例列出所有网景 FTP **站的** IP**。(注：网景的** FTP **站是一个** Domain Name **却对应到许多** IP Address **的网站。)**

```
<?php
$netscapeftp=gethostbynamel("ftp.netscape.com");
echo "Netscape FTP 网站 IP Address：<ol type=1>";
for ($i=0; $i<count($netscapeftp); $i++) {
   echo "<li>".$netscapeftp[$i];
}
echo "</ol>";
?>
```

Green Apple Data Center

checkdnsrr

检查指定网址的 DNS **记录。**

语法：int checkdnsrr(string host, string [type]);

返回值：整数

内容说明

本函数用来检查 DNS **的字段记录。指定的参数** host **可以是** IP **地址**(IP Address)**，也可以用机器名称**(Domain Name)**。参数** type **可以省略，内定值为** MX**。而参数** type **的值可为以下的其中之一：**A、MX、NS、SOA、PTR、CNAME **或** ANY**。若找到了指定网址的** DNS **字段，返回** true**；若未找到指定的** DNS **字段或是有错误均会返回** false**。**

getmxrr

取得指定网址 DNS **记录的** MX **字段。**

语法：int getmxrr(string hostname, array mxhosts, array [weight]);

返回值：整数

内容说明

本函数用来检查 DNS **字段记录中的** MX **字段，也就是电子邮件服务器** Mail eXchanger **字段。若找到了指定网址** DNS **记录的** MX **字段，返回** true**；若未找到指定的** DNS MX **字段或是有错误均会返回** false**。指定网址的所有** MX **字段记录的机器都会传入数组参数** mxhosts **中。若有指定数组参数** weight**，则同时返回** MX **机器的优先顺序。**

openlog

打开系统记录。

语法：int openlog(string ident, int option, int facility);

返回值：整数

内容说明

本函数会打开操作系统的记录机制(logger)**。参数** ident **会加到记录的字符串中。参数** option **的值包括了** LOG_PID、LOG_CONS、LOG_ODELAY、LOG_NDELAY、LOG_NOWAIT、LOG_PERROR**，在** Win32 **系统中，只有** LOG_PID **有效。参数** facility **的值可能为** LOG_KERN、LOG_USER、LOG_MAIL、LOG_DAEMON、LOG_AUTH、LOG_SYSLOG、LOG_LPR、LOG_NEWS、LOG_UUCP、LOG_CRON **或** LOG_AUTHPRIV**，在** Win32 **系统上，本参数是无效的字段。本函数呼叫** UNIX **系统的** openlog()**函数，因此在** Windows **系列的操作系统中，本函数没有完全的作用。**

使用范例

```
<?php
openlog("FUN",LOG_PID|LOG_CONS,LOG_USER);
syslog(LOG_INFO, "Wa ha ha ....");
closelog();
?>
```

syslog

记录至系统记录。

语法：int syslog(int priority, string message);

返回值：整数

内容说明

本函数将 message **字符串写到系统记录中，参数** priority **的值可能为** LOG_EMERG、LOG_ALERT、

Green Apple Data Center

LOG_CRIT、LOG_ERR、LOG_WARNING、LOG_NOTICE、LOG_INFO、LOG_DEBUG。本函数呼叫 UNIX 操作系统的 syslog()函数，在 Windows NT 上，使用事件监视器模拟出本功能。

closelog

关闭系统记录。

语法：int closelog(void);

返回值：整数

内容说明

本函数用来关闭已打开的系统记录。本函数无传入参数，亦不是必须要的函数，PHP 脚本程序在执行完成后就会自动关闭打开的资源。

debugger_on

使用内建的 PHP 调试器。

语法：int debugger_on(string address);

返回值：整数

内容说明

本函数用来对远程的机器进行 PHP 调试。调试使用的端口(Port)在 php.ini 中的 debugger.port 字段配置，默认端口是 7869。

使用范例

```
<?php
debugger_on("123.123.123.123");
?>
```

debugger_off

关闭内建的 PHP 调试器。

语法：int debugger_off(void);

返回值：整数

内容说明

本函数用来关闭远端的机器 PHP 调试功能。本函数无传入参数。

4.7 拼写检查函数库

aspell_new

载入一个新的字典。

语法：int aspell_new(string master, string personal);

返回值：整数

内容说明

本函数载入一个新的字典，并赋与一个新的身份值(整数)，以供程序中使用。

使用范例

```
$aspell_link=aspell_new("english");
```

aspell_check

检查一个单字。

语法：boolean aspell_check(int dictionary_link, string word);

Green Apple Data Center

返回值：布尔值

内容说明

本函数检查单字的拼写。若拼写正确则返回 true**，不正确则返回** false**。**

使用范例

```
$aspell_link=aspell_new("english");
if (aspell_check($aspell_link,"testt")) {
    echo "This is a valid spelling";
} else {
    echo "Sorry, wrong spelling";
}
```

aspell_check-raw

检查一个单字，即使拼错也不改变或修正。

语法：boolean aspell_check_raw(int dictionary_link, string word);

返回值：布尔值

内容说明

本函数检查单字的拼写。若拼写正确则返回 true**，不正确则返回** false**。本函数不会改变或者修正使用者的拼写。**

使用范例

```
$aspell_link=aspell_new("english");
if (aspell_check_raw($aspell_link,"testt")) {
    echo "This is a valid spelling";
} else {
    echo "Sorry, wrong spelling";
}
```

aspell_suggest

检查一个单字，并提供拼写建议。

语法：array aspell_suggest(int dictionary_link, string word);

返回值：数组

内容说明

本函数检查单字的拼写。并给予可能的拼法及正确的建议，以数组类型将结果返回。

使用范例

```
<?
$aspell_link=aspell_new("english");
if (!aspell_check($aspell_link,"testt")) {
    $suggestions=aspell_suggest($aspell_link,"testt");
    for($i=0; $i < count($suggestions); $i++) {
        echo "Possible spelling：" . $suggestions[$i] . "<br>";
    }
}
```

Green Apple Data Center

4.8 目录管理函数库

chdir

改变目录。
语法：int chdir(string directory);
返回值：整数
函数种类：文件存取
内容说明
本函数用来改变目前 PHP **执行的目录到新的** directory **目录中。若无法改变则返回** false**，成功则返回** true**。**

dir

目录类别类。
语法：new dir(string directory);
返回值：类
函数种类：文件存取
内容说明
这是一个类似面向对象的类别类，用来读取目录。当目录参数 directory **打开之后，有二个属性可用：**handle **属性就像其他非对象的函数所用的** readdir()、rewinddir()**及** closedir()**一样；**path **属性则配置打开目录后的路径参数。本类有三个方法**(method)**：**read、rewind **与** close**。**
使用范例

```
<?
$d = dir("/etc");
echo "Handle：".$d->handle."<br>\n";
echo "Path：".$d->path."<br>\n";
while($entry=$d->read()) {
    echo $entry."<br>\n";
}
$d->close();
?>
```

closedir

关闭目录 handle**。**
语法：void closedir(int dir_handle);
返回值：无
函数种类：文件存取
内容说明
本函数用来关闭目录资料流的 dir_handle**。这个** dir_handle **参数所操作的目录必须要** opendir()**打开的方可使用。**

opendir

打开目录 handle**。**
语法：int opendir(string path);
返回值：整数

Green Apple Data Center

函数种类：文件存取
内容说明
本函数用来打开目录资料流。返回的整数是可供其他目录函数操作的 handle。

readdir

读取目录 handle。
语法：string readdir(int dir_handle);
返回值：字符串
函数种类：文件存取
内容说明
本函数用来读取目录。返回目录中的文件名称，读取不按照任何特殊的顺序。
使用范例
本例列出目前目录的所有文件

```
<?php
$handle=opendir('.');
echo "目录 handle：$handle\n";
echo "文件：\n";
while ($file = readdir($handle)) {
      echo "$file\n";
}
closedir($handle);
?>
```

rewinddir

重设目录 handle。
语法：void rewinddir(int dir_handle);
返回值：无
函数种类：文件存取
内容说明
本函数用来重设目录资料流到开始处。

4.9 HTTP 相关函数库

header

送出 HTTP **协议的标头到浏览器**
语法：int header(string string);
返回值：整数
函数种类：网络系统
内容说明
标头(header)**是服务器以** HTTP **协议传** HTML **资料到浏览器前所送出的字符串，在标头与** HTML **文件之间尚需空一行分隔。有关** HTTP **的详细说明，可以参考相关书籍或更详细的** RFC 2068 **官方文件**(http：//www.w3.org/Protocols/rfc2068/rfc2068)。**在** PHP **中送回** HTML **资料前，需先传完所有的标头。**
注意：传统的标头一定包含下面三种标头之一，并只能出现一次。
Content-Type：xxxx/yyyy

Green Apple Data Center

Location：xxxx：yyyy/zzzz
Status：nnn xxxxxx
新的多型标头规格(Multipart MIME)**方可以出现二次以上。**
使用范例
范例一：本例用来重导用户到 PHP **的官方网站。**

```
<?php
Header("Location：http：//www.php.net");
exit;
?>
```

范例二：欲让用户每次都能得到最新的资料，而不是 Proxy **或** cache **中的资料，可以使用下列的标头**

```
header("Expires：Mon, 26 Jul 1997 05：00：00 GMT");
header("Last-Modified：" . gmdate("D, d M Y H：i：s") . "GMT");
header("Cache-Control：no-cache, must-revalidate");
header("Pragma：no-cache");
```

范例三：让用户的浏览器出现找不到文件的信息。

```
<?php
header("Status：404 Not Found");
?>
```

范例四：用户下载文件的范例。

```
header("Content-type：application/x-gzip");
header("Content-Disposition：attachment; filename=some-file.tar.gz");
header("Content-Description：PHP3 Generated Data");
```

setcookie

送出 Cookie **信息到浏览器。**
语法：int setcookie(string name, string value, int expire, string path, string domain, int secue);
返回值：整数
内容说明
本函数会跟着标头 Header **送出一段小信息字符串到浏览器。使用本函数需要在送出** HTML **资料前，实际上** cookie **也算标头的一部分。本函数的参数除了第一个** name **之外，都是可以省略的。参数** name **表示** cookie **的名称；**value **表示这个** cookie **的值，这个参数为空字符串则表示取消浏览器中该** cookie **的资料；**expire **表示该** cookie **的有效时间；**path **为该** cookie **的相关路径；**domain **表示** cookie **的网站；**secure **则需在** https **的安全传输时才有效。想得到更多的** cookie **信息可以到** http：//www.netscape.com/newsref/std/cookie_spec.html**，有** cookie **原创者** Netscape **所提供的完整信息。**
使用范例

```
<?php
$status = 0;
if (isset($myTstCky) && ($myTstCky == "ChocChip")) $status = 1;
if (!isset($CCHK)) {
```

Green Apple Data Center

```
    setcookie("myTstCky", "ChocChip");
    header("Location：$PHP_SELF?CCHK=1");
    exit;
}
?>
<html>
<head><title>Cookie Check</title></head>
<body bgcolor="#FFFFFF" text="#000000">
Cookie Check Status：
<?php
printf ('<font color="#%s">%s</font><br>;',
    $status ? "00FF00"  ："FF0000",
    $status ? "PASSED!"  ："FAILED!");
?>
</body>
</html>
```

4.10 电子邮件函数库

mail

寄出电子邮件。

语法：boolean mail(string to, string subject, string message, string [additional_headers]);

返回值：布尔值

函数种类：网络系统

内容说明

本函数寄出电子邮件到指定的邮件地址 to，subject **表示主题，** message **为信件内容。额外的选项** additional_headers **可省略，表示其他的邮件文件头。**

使用范例

```
<?
$message="abcdefghijklmnopqrstuvwxyz";
mail("php@wilson.gs","没有主题",$message,  "From：someone@wahaha.edu.cn\nReply-To：reply@wahaha.edu.cn\nX-Mailer：PHP/" . phpversion());
?>
```

4.11 mhash 哈稀函数库

本函数库共有 4 **个函数**

本函数库支持多种哈稀演算法，例如最出名的 MD5、SHA1 **或** GOST，**还有其他多种的哈稀演算法，列示如下：**

MHASH_MD5
MHASH_SHA1
MHASH_HAVAL
MHASH_RIPEMD160
MHASH_RIPEMD128
MHASH_SNEFRU

Green Apple Data Center

MHASH_TIGER
MHASH_GOST
MHASH_CRC32
MHASH_CRC32B

欲使用本函数库要先下载 mhash-x.x.x.tar.gz**，网址为** http**：**//sasweb.de/mhash**。当然还要编译** mhash **程序库，之后才能编译** PHP **程序，在编译** PHP **程序时，记得要加**--with-mhash **选项打开系统的** mhash **功能。**

本函数库适合用来产生检查码(checksums)**、数位代码信息或者其他功能，如下例：**

```
<?php
$input = "Let us meet at 9 o' clock at the secret place.";
$hash = mhash(MHASH_SHA1, $input);
print "哈稀值为 ".bin2hex($hash)."\n";
?>
```

在浏览器看到的字符串是
哈稀值为 d3b85d710d8f6e4e5efd4d5e67d041f9cecedafe

mhash_get_hash_name

取得哈稀演算法名称。
语法：string mhash_get_hash_name(int hash);
返回值：字符串
函数种类：编码处理
内容说明
本函数取得哈稀演算法的名称。返回值为名称字符串，若没有指定的哈稀演算法则返回 false **或输入的名称。**
使用范例
下例返回的字符串为 MD5**。**

```
<?php
$hash = MHASH_MD5;
print mhash_get_hash_name($hash);
?>
```

mhash_get_block_size

取得哈稀方式的区块大小。
语法：int mhash_get_block_size(int hash);
返回值：整数
函数种类：编码处理
内容说明
本函数用来取得哈稀演算的区块大小。参数为编码名称，返回整数值，单位为位组(byte)**。**

mhash_count

取得哈稀 ID **的最大值。**
语法：int mhash_count(void);
返回值：整数
函数种类：编码处理

Green Apple Data Center

内容说明

本函数用来取得哈稀演算的最大 ID **值。在使用哈稀计算时，会从** 0 **开始计数到使用的数值。本函数不用输入参数。**

使用范例

```
<?php
$nr = mhash_count();
for($i = 0; $i <= $nr; $i++) {
   echo sprintf("哈稀%s 的区块大小为%d\n",mhash_get_hash_name($i),mhash_get_block_size($i));
}
?>
```

mhash

计算哈稀值。

语法：string mhash(int hash, string data);

返回值：字符串

函数种类：编码处理

内容说明

本函数依指定的哈稀演算法计算哈稀值。参数 hash **为指定的哈稀演算法；参数** data **为欲计算的字符串值。**

4.12 正则表达式函数库

本函数库共有 6 **个函数**

关于正则表达式(Regular expression)**，实际上是负责字符串解析比对，并对字符串做相关的处理。本函数库让** PHP **也能处理复杂的字符串操作。它采用了** POSIX 1003.2 **的扩充常规处理**(regular expression)**的标准。更多关于正则表达式的信息可以参考** UNIX Shell**、**Perl **或是** awk **等相关的书籍。**

ereg

字符串比对解析。

语法：int ereg(string pattern, string string, array [regs]);

返回值：整数/数组

内容说明

本函数以 pattern **的规则来解析比对字符串** string**。比对结果的返回值放在数组参数** regs **之中，**regs[0]**内容就是原字符串** string**、**regs[1]**为第一个合乎规则的字符串、**regs[2]**就是第二个合乎规则的字符串，其余类推。若省略参数** regs**，则只是单纯地比对，找到则返回值为** true**。**

使用范例

这个例子可对使用者输入的 E-Mail **作简单的检查，检查使用者的** E-Mail **字符串是否有@字符，在@字符前有英文字母或数字，在之后有数节字符串，最后的小数点后只能有二个或三个英文字母。**super@mail.wilson.gs **就可以通过检查，**super@mail.wilson **就不能通过检查。**

```
<?php
if (eregi("^[_\.0-9a-z-]+@([0-9a-z][0-9a-z-]+\.)+[a-z]{2,3}$",$email)) {
   echo "您的 E-Mail 通过初步检查";
}
?>
```

Green Apple Data Center

ereg_replace

字符串比对解析并取代。

语法：string ereg_replace(string pattern, string replacement, string string);

返回值：字符串

内容说明

本函数以 pattern **的规则来解析比对字符串** string**，欲取而代之的字符串为参数** replacement**。返回值为字符串类型，为取代后的字符串结果。**

eregi

字符串比对解析，与大小写无关。

语法：int eregi(string pattern, string string, array [regs]);

返回值：整数/数组

内容说明

本函数和 ereg()**类似，用法也相同。不同之处在于** ereg()**有区分大小写，本函数与大小写无关。**

eregi_replace

字符串比对解析并取代，与大小写无关。

语法：string eregi_replace(string pattern, string replacement, string string);

返回值：字符串

内容说明

本函数和 ereg_replace()**类似，用法也相同。不同之处在于** ereg_replace()**有区分大小写，本函数与大小写无关。**

split

将字符串依指定的规则分开。

语法：array split(string pattern, string string, int [limit]);

返回值：数组

内容说明

本函数可将字符串依指定的规则分开。分开后的返回值为数组变量。参数 pattern **为指定的规则字符串、参数** string **则为待处理的字符串、参数** limit **可省略，表示欲处理的最多合乎值。值得注意的是本函数的** pattern **参数有区分大小写。**

sql_regcase

将字符串逐字返回大小写字符。

语法：string sql_regcase(string string);

返回值：数组

内容说明

本函数可将字符串的字符逐字返回大小写。在 PHP **使用上，本函数没有什么作用，但可以提供外部程序或数据库处理。**

4.13 Session 函数库

session_start

初始 session**。**

Green Apple Data Center

语法：boolean session_start(void);
返回值：布尔值
内容说明
本函数初始化一个新的 Session**，若该客户已在** Session **之中，则连上原** Session**。本函数没有参数，且返回值均为** true**。**

session_destroy

结束 session**。**
语法：boolean session_destroy(void);
返回值：布尔值
内容说明
本函数结束目前的 Session**。本函数没有参数，且返回值均为** true**。**

session_name

存取目前 session **名称。**
语法：string session_name(string [name]);
返回值：字符串
内容说明
本函数可取得或者重新配置目前 Session **的名称。若无参数** name **则表示获取目前** Session **名称，加上参数则表示将** Session **名称设为参数** name**。**
使用范例
下面的范例为 Session **片段程序**

```
<?php
$username="guest1";
if(isset($username)) {
      session_name($username);
}
echo "您是 ".session_name()."\n";
?>
```

session_module_name

存取目前 session **模块。**
语法：string session_module_name(string [module]);
返回值：字符串
内容说明
本函数可取得或者重新配置目前的 Session **模块。若无参数** module **则表示只获取目前** Session **的模块，加上参数则表示将** Session **模块设为参数** module**。**

session_save_path

存取目前 session **路径。**
语法：string session_save_path(string [path]);
返回值：字符串
内容说明
本函数可取得或者重新配置目前存放 Session **的路径。若无参数** path **则表示只有取得目前** Session **的路径目录名，加上参数** path **则表示将** Session **存在新的** path **上。**

Green Apple Data Center

session_id

存取目前 session **代号。**

语法：string session_id(string [id]);

返回值：字符串

内容说明

本函数可取得或者重新配置目前存放 Session **的代号。若无参数** id **则表示只有取得目前** Session **的代号，加上参数则表示将** Session **代号设成新指定的** id**。输入及返回均为字符串。**

session_register

注册新的变量。

语法：boolean session_register(string name);

返回值：布尔值

内容说明

本函数在全域变量中增加一个变量到目前的 Session **之中。参数** name **即为欲加入的变量名。成功则返回** true **值。**

session_unregister

删除已注册变量。

语法：boolean session_unregister(string name);

返回值：布尔值

内容说明

本函数在目前的 Session **中删除全域变量上的变量。参数** name **即为欲删除的变量名。成功则返回** true **值。**

session_is_registered

检查变量是否注册。

语法：boolean session_is_registered(string name);

返回值：布尔值

内容说明

本函数可检查目前的 Session **中是否已有指定的变量注册。参数** name **即为欲检查的变量名。成功则返回** true **值。**

session_decode

Session **资料解码。**

语法：boolean session_decode(string data);

返回值：布尔值

内容说明

本函数可将 Session **资料解码。参数** data **即为欲解码的资料。成功则返回** true **值。**

session_encode

Session **资料编码。**

语法：boolean session_encode(void);

返回值：布尔值

内容说明

本函数可将 Session **资料编码，编码以** ZEND **引擎做哈稀编码。本函数没有参数。成功则返回** true **值。**

Green Apple Data Center

4.14 压缩文件函数库

欲使用本函数库需先安装 zlib，**可到** http：//www.cdrom.com/pub/infozip/zlib/**取得该函数库。**

gzclose

关闭压缩文件。
语法：boolean gzclose(int zp);
返回值：布尔值
内容说明
本函数将已打开的压缩文件关闭。参数 zp **为压缩文件的指针代码。成功则返回** true **值。**

gzeof

判断是否在压缩文件尾。
语法：boolean gzeof(int zp);
返回值：布尔值
内容说明
本函数用来判断目前打开的压缩文件指针是否指到文件尾(EOF, End OF File)**。参数** zp **为压缩文件的指针代码。在文件尾则返回** true **值。**

gzfile

读压缩文件到数组中。
语法：array gzfile(string filename);
返回值：数组
内容说明
本函数将压缩文件读出并解压缩至数组变量中。参数 filename **为文件名称。**

gzgetc

读压缩文件中的字符。
语法：string gzgetc(int zp);
返回值：字符串
内容说明
本函数将压缩文件解压缩并取出一个字符。参数 zp **为压缩文件打开代码。若至文件尾则会返回** false**。**

gzgets

读压缩文件中的字符串。
语法：string gzgets(int zp, int length);
返回值：字符串
内容说明
本函数将压缩文件解压缩并取出指定长度的字符串。参数 zp **为压缩文件打开代码。参数** length **为指定字符串长度加一，意即读出的字符串长度为** length-1**。若至文件尾或至行尾即停止，因此本函数通常用来读取一行。**

gzgetss

读压缩文件中的字符串，并去掉 HTML **指令。**
语法：string gzgetss(int zp, int length);
返回值：字符串

Green Apple Data Center

内容说明

本函数将压缩文件解压缩并取出指定长度的字符串，并将字符串中的 HTML **或** PHP **指令去掉，返回纯文字。参数** zp **为压缩文件打开代码。参数** length **为指定字符串长度加一，意即读出的字符串长度为** length-1**。若至文件尾或至行尾即停止，因此本函数通常用来读取一行。**

gzopen

打开压缩文件。

语法：int gzopen(string filename, string mode);

返回值：整数

内容说明

本函数用来打开压缩文件。参数 filename **为文件名称。参数** mode **为打开文件的状态。若有失败则返回** false **值。**

使用范例

下面为部份程序

```
<?php
$fp=gzopen("/tmp/gzfile.gz", "r");
?>
```

gzpassthru

解压缩打开压缩文件指针后的全部资料。

语法：boolean gzpassthru(int zp);

返回值：布尔值

内容说明

本函数将已打开压缩文件指针后的资料全部解压缩，并输出至标准输出装置(stdout)**。参数** zp **为打开文件的代码。若有失败则返回** false **值。**

gzputs

资料写入压缩文件。

语法：boolean gzputs(int zp, string str, int [length]);

返回值：布尔值

内容说明

本函数其实就是 gzwrite()**。参数** zp **为打开文件的代码。参数** str **为欲写入的字符串。参数** length **可省略，为指定长度。若有失败则返回** false **值。**

gzread

读出压缩文件中指定长度字符串。

语法：string gzread(int zp, int length);

返回值：字符串

内容说明

本函数用来读取指定长度的字符串。参数 zp **为打开文件的代码。参数** length **为指定长度。**

使用范例

```
<?php
$filename = "/temp/sosofile.txt.gz";
$zd = gzopen($filename, "r");
$contents = gzread($zd, 10000);
```

Green Apple Data Center

```
gzclose($zd);
?>
```

gzrewind

重设压缩文件指针。
语法：boolean gzrewind(int zp);
返回值：布尔值
内容说明
本函数将重设压缩文件的文件操作指针到文件头处。参数 zp **为打开文件代码。**

gzseek

重设压缩文件指针至指定处。
语法：int gzseek(int zp, int offset);
返回值：整数
内容说明
本函数将重设压缩文件的文件操作指针到指定的位置。参数 zp **为打开文件代码。参数** offset **为第几个位。成功则返回** 0**，失败返回**-1**。**

gztell

取得压缩文件指针处。
语法：int gztell(int zp);
返回值：整数
内容说明
本函数用来取得压缩文件的文件操作指针在某位处。参数 zp **为打开文件代码。**
使用范例

```
<?php
$exfile=gzopen("/tmp/haha.gz", "r");
$aline=gzgets($exfile, 80);
print("现在文件指针在第".gztell($exfile)."个位");
gzclose($exfile);
?>
```

readgzfile

读出压缩文件。
语法：boolean readgzfile(string filename);
返回值：布尔值
内容说明
本函数将压缩文件全部读出并解压缩，之后将内容送到标准输出设备上(stdout)**。参数** filename **为文件名称。本函数其实也可以读取非压缩文件至标准输出设备中。**

gzwrite

资料写入压缩文件。
语法：boolean gzwrite(int zp, string string, int [length]);
返回值：布尔值
内容说明

Green Apple Data Center

本函数用来将资料写入指定的压缩文件中。参数 zp 为打开文件的代码。参数 str 为欲写入的字符串。参数 length 可省略，为指定长度。若有失败则返回 false 值。

4.15 MySQL 函数库

PHP 可以访问多种数据库，但是最常用的是 MySQL 数据库，所以本节只介绍 MySQL 函数库。

mysql_connect

打开 MySQL 服务器链接。

语法：int mysql_connect(string [hostname] [：port]， string [username]， string [password]);

返回值：整数

内容说明

本函数建立与 MySQL 服务器的链接。其中所有的参数都可省略。当使用本函数却不加任何参数时，参数 hostname 的默认值为 localhost、参数 username 的默认值为 PHP 执行行程的拥有者、参数 password 则为空字符串(即没有密码)。而参数 hostname 后面可以加冒号与埠号，代表使用那个埠与 MySQL 链接。当然在使用数据库时，早点使用mysql_close()将链接关掉可以节省资源。

使用范例

```
<?php
$dbh = mysql_connect('localhost：3306'，'root'，'');
mysql_select_db('admreqs');
$query = "insert into requests(date，request，email，priority，status) values (NOW()，'$description'，'$email'，'$priority'，'NEW')";
$res = mysql_query($query， $dbh);
$query = "select max(id) from requests";
$res = mysql_query($query， $dbh);
$err = mysql_error();
if($err){
  echo "发生错误，请通知<a href=mailto：webmaster@mysite>站长</a>";
}
  $row = mysql_fetch_row($res);
  echo "未来您使用的号码为： "。$row[0];
?>
```

参考

mysql_close() mysql_pconnect()

mysql_create_db

建立一个 MySQL 新数据库。

语法：int mysql_create_db(string database name， int [link_identifier]);

返回值：整数

内容说明

本函数用来建立新的数据库(database)。在建立前，必须先与服务器链接。

mysql_select_db

选择一个数据库。

语法：int mysql_select_db(string database_name， int [link_identifier]);

返回值：整数

Green Apple Data Center

内容说明

本函数选择 MySQL **服务器中的数据库以供之后的资料查询作业**(query)**处理。成功返回** true，**失败则返回** false。

参考

mysql_connect() mysql_pconnect() mysql_query()

mysql_query

送出一个 query **字符串。**

语法：int mysql_query(string query， int [link_identifier]);

返回值：整数

内容说明

本函数送出 query **字符串供** MySQL **做相关的处理或者执行。若没有指定** link_identifier **参数，则程序会自动寻找最近打开的** ID。**当** query **查询字符串是** UPDATE、INSERT **及** DELETE **时，返回的可能是** true **或者** false；**查询的字符串是** SELECT **则返回新的** ID **值。当返回** false **时，并不是执行成功但无返回值，而是查询的字符串有错误。**

参考

mysql_select_db() mysql_connect()

mysql_fetch_array

返回数组资料。

语法：array mysql_fetch_array(int result， int [result_typ]);

返回值：数组

内容说明

取出下一行，返回一个数组。可以用数字下标访问(第一个字段是下标 0)，**也可以用字符串下标访问(即使用各字段名)，如果已取了最后一行，返回** false。

使用范例

```
<?php
mysql_connect($host，$user，$password);
$result = mysql_db_query("database"，"select * from table");
while($row = mysql_fetch_array($result)) {
   echo $row["user_id"];
   echo $row["fullname"];
}
mysql_free_result($result);
?>
```

mysql_fetch_field

取得字段信息。

语法：object mysql_fetch_field(int result， int [field_offset]);

返回值：类

内容说明

本函数返回的类资料为 result **的字段**(Column)**信息。返回类的属性如下：**

name-**字段名称**

table-**字段所在表格的资料表名称**

max_length-**字段的最大长度**

Green Apple Data Center

not_null-若为 1 表示本字段不能是空的(null)
primary_key-若为 1 表示本字段是主要键(primary key)
unique_key-若为 1 表示本字段为不可重覆键(unique key)
multiple_key-若为 1 表示本字段为可重覆键(non-unique key)
numeric-若为 1 表示本字段为数字类型(numeric)
blob-若为 1 表示本字段为位类型(BLOB)
type-字段类型
unsigned-若为 1 表示本字段为无记号(unsigned)
zerofill-若为 1 表示本字段为被零填满(zero-filled)

mysql_num_rows

取得返回列的数目。
语法：int mysql_num_rows(int result);
返回值：整数
内容说明
本函数可以得到返回列的数目。
参考
mysql_query()　mysql_fetch_row()

mysql_num_fields

取得返回字段的数目。
语法：int mysql_num_fields(int result);
返回值：整数
内容说明
本函数可以得到返回字段的数目。
参考
mysql_query()　mysql_fetch_field()　mysql_num_rows()

mysql_free_result

释放返回占用内存。
语法：boolean mysql_free_result(int result);
返回值：布尔值
内容说明
本函数可以释放目前 MySQL 数据库 query 返回所占用的内存，一般只有在非常担心在内存的使用上可能会不足的情形下才会用本函数，PHP 程序会在结束时自动释放。

mysql_list_dbs()

列出 MySQL 服务器可用的数据库(database)。
语法：int mysql_list_dbs(int [link_identifier]);
返回值：整数
内容说明
本函数可以得到指定数据库中的所有资料表名称。

mysql_list_tables

列出指定数据库的资料表(table)。
语法：int mysql_list_tables(string database，　int [link_identifier]);

Green Apple Data Center

返回值：整数

mysql_close

关闭 MySQL 服务器链接。

语法：int mysql_close(int [link_identifier]);

返回值：整数

内容说明

本函数关闭与 MySQL 数据库服务器的链接。若无指定参数 link_identifier 则会关闭最后的一次链接。用 mysql_pconnect()**链接则无法使用本函数关闭。实际上本函数不是一定需要的，当 PHP 整页程序结束后，将会自动关闭与数据库的非永久性**(non-persistent)**链接。成功返回** true**，失败返回** false **值。**

参考

mysql_connect()　mysql_pconnect()

mysql_pconnect

打开 MySQL 服务器持续链接。

语法：int mysql_pconnect(string [hostname] [：port]， string [username]， string [password]);

返回值：整数

内容说明

本函数和mysql_connect()**相似。不同的地方在于使用本函数打开数据库时，程序会先寻找是否曾经执行过本函数，若执行过则返回先前执行的** ID**。另一个不同的地方是本函数无法使用**mysql_close()**关闭数据库。**

mysql_select_db

选择一个数据库。

语法：int mysql_select_db(string database_name， int [link_identifier]);

返回值：整数

内容说明

本函数选择 MySQL 服务器中的数据库以供之后的资料查询作业(query)**处理。成功返回** true**，失败则返回** false**。**

参考

mysql_connect()　mysql_pconnect()　mysql_query()

Green Apple Data Center

第 5 章　PHP 简单应用实例

经过前面几章的介绍，读者对 PHP 有了一定了解，特别是对 C 语言熟悉的读者掌握起来更是容易。要用好 PHP 这个编程工具，必须一些实战训练。本章介绍一些在做网页时常用的实例，介绍 PHP 编程的基本技巧和要点。

5.1　HTTP 认证

也许有的朋友想在网站放上自己的照片集，而且只想给自己知心的朋友看，这时就需要一个密码验证的程序。如何用 PHP 来实现密码验证的功能呢？可以使用简短的 PHP 代码，使用函数 header()发送 HTTP 标头强制验证，客户端浏览器则弹出供输入用户名和密码的对话框。在 PHP 中，客户端用户输入的信息传送到服务端之后自动保存在$PHP_AUTH _USER，$PHP_AUTH_PW，以及$PHP_AUTH_TYPE 这三个全局变量中。利用这些变量，就可以根据保存在数据文件或数据库中的用户帐号信息验证用户身份。

不过在这里需要提醒使用者注意的一点是：只有在 Apache 模块方式运行的时候，PHP 脚本才能使用 $PHP_AUTH_USER，$PHP_AUTH_PW，以及$PHP_AUTH_TYPE 这三个变量。如果用户使用的是 CGI 模式的 PHP 则无法实现基于 HTTP 的验证功能。

下面，就来详细介绍一下如何使用 PHP 对用户身份进行验证。在下例中是使用$PHP_AUTH_USER 和 $PHP_AUTH_PW 这两个变量来验证进入者是否合法并允许进入。在本例中被允许登录的用户名称和密码对分别为 tnc 和 nature：

```
<?
if(!isset($PHP_AUTH_USER))
{
Header("WWW-Authenticate: Basic realm=\"My Realm\"");
Header("HTTP/1.0 401 Unauthorized");
echo "Text to send if user hits Cancel button\n";
exit;
}
else
{
if ( !($PHP_AUTH_USER=="tnc" && $PHP_AUTH_PW=="nature") )
{
// 如果是错误的用户名称/密码对，强制再验证
Header("WWW-Authenticate: Basic realm=\"My Realm\"");
Header("HTTP/1.0 401 Unauthorized");
echo "ERROR : $PHP_AUTH_USER/$PHP_AUTH_PW is invalid.";
exit;
}
else
{
echo "Welcome tnc!";
}
?>
```

Green Apple Data Center

事实上在实际引用中，不大可能如上面使用明显的用户名称/密码对代码段，而是利用数据库或者加密的密码文件存取它们。数据库的知识在下一章将作详细讲述。

下面再讲述怎样根据指定的验证信息来核实用户身份？

首先，可以使用以下代码确定用户是否已经输入了用户名和密码，并显示出用户输入的信息。

```
<?php
if (!isset($PHP_AUTH_USER)) {
header('WWW-Authenticate: Basic realm="My Private Stuff"');
header('HTTP/1.0 401 Unauthorized');
echo 'Authorization Required.';
exit;
}
else {
echo "<P>You have entered this username: $PHP_AUTH_USER<br>
You have entered this password: $PHP_AUTH_PW<br>
The authorization type is: $PHP_AUTH_TYPE</p>";
}
?>
```

说明：

isset（）函数用于确定某个变量是否已被赋值。根据变量值是否存在，返回 true 或 false。

header（）函数用于发送特定的 HTTP 标头。注意，使用 header（）函数时，一定要在任何产生实际输出的 HTML 或 PHP 代码前面调用该函数。

虽然上述代码相当简单，没有根据任何实际值对用户输入的用户名和密码进行有效验证，但是至少了解了如何使用 PHP 在客户端产生输入对话框。

下面，就来了解一下如何根据指定的验证信息核实用户身份。代码如下：

```
<?php
if (!isset($PHP_AUTH_USER)) {
header('WWW-Authenticate: Basic realm="My Private Stuff"');
header('HTTP/1.0 401 Unauthorized');
echo 'Authorization Required.';
exit;
}
else if (isset($PHP_AUTH_USER)) {
if (($PHP_AUTH_USER != "admin") || ($PHP_AUTH_PW != "123")) {
header('WWW-Authenticate: Basic realm="My Private Stuff"');
header('HTTP/1.0 401 Unauthorized');
echo 'Authorization Required.';
exit;
} else {
echo "<P>You're authorized!</p>";
}
}
?>
```

Green Apple Data Center

在这里，首先检查用户是否已经输入了用户名称和密码，如果没有则弹出相应对话框要求用户输入身份信息。随后，通过判断用户输入的信息是否符合 admin/123 这一指定用户帐号来授予用户访问权限或提示用户再次输入正确的信息。这种方法适用于所有用户都使用同一登录帐号的网站。

5.2 访客计数器

现在，上网的人越来越多，许多网友尝试着制作自己的主页，访客计数器是必不可少的一部分，因为它是向来访者炫耀自己网站人气指数的最好方法。虽然很多网站提供免费的计数器程序，可毕竟不是自己亲手制作的。有的朋友可能认为它很难，不敢去尝试，其实利用 PHP 语言编程，制作一个计数器不仅不难，甚至可以说它非常容易，几行代码就可以了。

首先，介绍访客计数器的程序设计原理：

（1）第一位访问者浏览此网页，计数程序从一个文件（下文以 count.txt 为例）中读取记录该页已被浏览的次数，并且再加上一，然后存回 count.txt，并在浏览器中显示加一后的次数。

（2）如果第二位访问者浏览此网页，计数程序又重复上述过程，从而实现了访客计数器。

PHP 没有直接的计数器函数，但利用它对文件读写的强大功能，可以很容易地自己编写一个计数器。现对程序需要用到的函数进行说明：

（1）打开文件操作：int fopen(string filename, string mode)；

其中 string filename 是要打开的文件名，必须为字符串形式。例如"count.txt"。

string mode 是打开文件的方式，必须为字符形式。

'r'，只读形式，文件指针指向文件的开头。

'r+'，可读可写，文件指针指向文件的开头。

'w'，只写形式，文件指针指向文件的开头，把文件长度截成 0，如果文件不存在，将尝试建立文件。

'w+'，可读可写，文件指针指向文件的开头，把文件长度截成 0，如果文件不存在，将尝试建立文件。

'a'，追加形式（只可写入），文件指针指向文件的最后，如果文件不存在，将尝试建立文件。

'a+'，可读可写，文件指针指向文件的最后，如果文件不存在，将尝试建立文件。

（2）读文件操作：array file(string filename);

将文件全部读出，并输出到数组的变量中，每行都是单独的数组元素。

（3）写文件操作：int fputs(int fp, string str, int [length]);

其中 int fp 是要写入信息的文件流指针，由 fopen 函数返回数值。

string str 是要写入文件的字符串。

int length 是写入的长度，可选，如果不选 length，则整个串将被写入。否则，写入 length 长度个字符。

（4）关闭文件操作：int fclose(int fp);

其中 int fp 是 fopen 函数返回的文件流指针。

下面，来看一下计数器的源程序：（假设 count.txt 文件存在）

```
<html>
<head>
<title>计数器</title>
</head>
<body>
<?php
$visits = file("count.txt");
//读取 count.txt 文件，把每行内容输出到数组的变量$visits 中
$number_of_last_visit = $visits[0];
//读取数组变量的第一个元素，即文件第一行内容
$number_of_new_visit = ++$number_of_last_visit;
//浏览次数加一
```

Green Apple Data Center

```
$fp = fopen("count.txt", "w");
//只写方式打开 count.txt 文件
$fw = fputs($fp, $number_of_new_visit);
//写入加一后结果
echo "$number_of_new_visit";
//浏览器输出浏览次数
fclose($fp);
//关闭文件
?>
</body>
</html>
```

在浏览器中运行结果如图 5-1 所示。

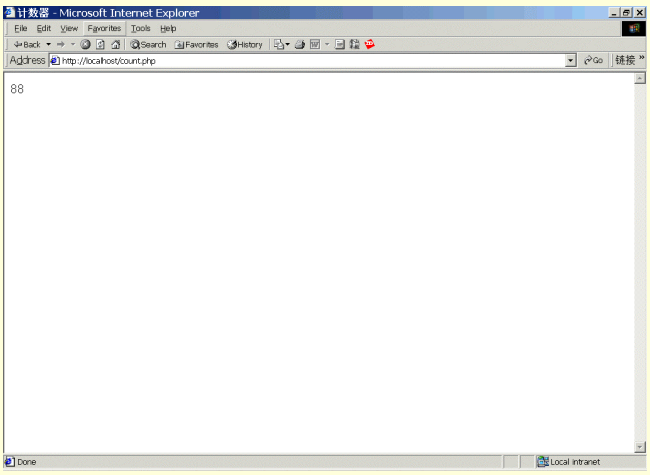


图 5-1　数字访客计数器

需要说明的是，这只是计数器的原型，它只能以文本方式显示次数，并不美观，没有一个网站会把这样一个纯数字计数器直接拿来用。下面介绍一个基于上述代码的图形计数器。原理如下：

（1）先用图形工具（如 Photoshop）制作 0~9 十个数字，分别为 0.gif~9.gif。

（2）将文件 count.txt 中得到的数字用 substr()函数分开为单独的数字。

（3）将每个数字转化为图形显示。

图形计数器的源程序如下，此程序的前面部分代码和上述文本计数器几乎一样。

先熟悉以下字符串处理函数：

1.**字符串长度函数：**int strlen(string str);

其中 string str **是要计算长度的字符串。**

2.**字符串相加：**

如，把$string1 **和**$string2 **相加：**

$string = $string1.$string2

3.**取部分字符串函数：**string substr(string string, int start, int [length]);

Substr **返回** string **中由参数** start **和** length **指定的部分。**

```
<html>
<head>
<title>计数器</title>
</head>
<body>
```

Green Apple Data Center

```
<?php
$visits = file("count.txt");
$number_of_last_visit = $visits[0];
$number_of_new_visit = ++$number_of_last_visit;
$fp = fopen("count.txt", "w");
$fw = fwrite($fp, $number_of_new_visit);
fclose($fp);

$len_str = strlen($number_of_new_visit);
//取出数字的长度
for($i=(0);$i<$len_str;$i++){
$numbers_exploded = substr($number_of_new_visit,$i,1);
//分开为单独的数字。
$output_str = $output_str . "<img src='../image/$numbers_exploded.gif'>";
//将每个数字转化为图形显示
}
echo   $output_str;
//浏览器输出图形
php?>
</body>
</html>
```

在浏览器中运行结果如图 5-2 **所示。**

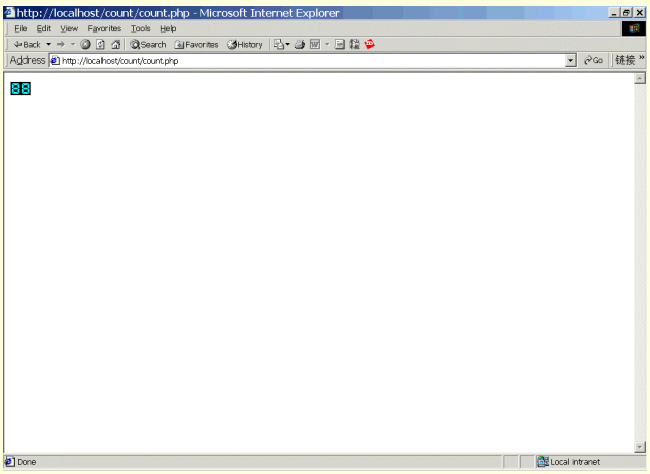


图 5-2　**图形访客计数器**

PHP **具有极其强大的图像处理能力，用它可以很轻易的动态生成** WEB **图像。不过有点麻烦的是需要对** PHP **的图像处理函数相当熟悉，下面对此感兴趣的读者介绍一个利用** PHP **图像处理函数生成的图形访客计数器。**

它的原理是：用上二例方法得到访问次数后，再把数字转为标准格式进行图像处理，并输出成图片显示。

如果要生成计数图像，需要以下函数：

1.**新建图像函数：**int imagecreate(int x_size, int y_size);

其中 x_size，y_size **分别是新建图像的宽度和高度(以像素为单位)。**

2.**颜色函数：**int imagecolorallocate(int im, int red, int green, int blue);

其中 int im **是图像识别号。**

int red、int green、int blue **分别是红绿蓝三种颜色的分量，取值范围** 0-255，**即相应颜色的** RGB。

3.**给图像填充颜色的函数：**int imagefill(int im, int x, int y, int col);

Green Apple Data Center

其中 int x, int y **为开始填充颜色的图像坐标，以图像的左上角为（**0，0**）。**
int col **是颜色的识别号。**
4.**在图像中写入水平文字的函数：**int imagestring(int im, int font, int x, int y, string s, int col);
其中 int im **是图像的识别号。**
int font **是字体识别号。**
int x，int y **是开始写入字体的坐标，**(0,0)**为左上角。**
string s **是要写入的字符串。**
int col **是字体的颜色识别号。**
5.**在图像中划直线的函数：**int imageline(int im, int x1, int y1, int x2, int y2, int col);
其中 int im **是图像的识别号。**
int x1，int y1，int x2，int y2 **是划线的起止坐标。**
int col **是线的颜色识别号。**
6.**把图像输出成** GIF **格式的函数：**int imagegif(int im, string filename);
其中 int im **是图像的识别号。**
string filename **是生成图片的名字，可选，如果** filename **为空，则直接地输出。**
7.**释放图像：**int imagedestroy(int im);
其中 int im **是要释放的图像识别号。**
该函数释放识别号 im **的图像及图像所占用的系统资源。**
下面是 counter.php **的程序清单：**

```
<?
Header("Content-type: image/gif");
//定义输出为图像类型
$n=10;
//变量$n 是显示位数
$fp = fopen("count.txt", "r");
$str1 = fgets($fp,$n+1);
$str1++;
fclose($fp);
$fp = fopen("count.txt", "w");
fputs($fp, $str1);
fclose($fp);
$str2 = "";
$len1 = strlen($str1);
for ($i=1;$i<=$n;$i++) {
$str2 = "0".$str2;
};
//得到$n 位 0
$len2 = strlen($str2);
//计算访问人数的位数
$dif = $len2 - $len1;
$rest = substr($str2, 0, $dif);
$string = $rest.$str1;
//位数如果不够$n 位，在前面补 0
for ($i=0;$i<=$n-1;$i++) {
$str[$i]=substr($string,$i,1);
};
```

Green Apple Data Center

```
//以数组存储每位
$font = 4;
//定义字号
$im = imagecreate($n*11-1,16);
//新建图象
$black = ImageColorAllocate($im, 0,0,0);
$white = ImageColorAllocate($im, 255,255,255);
//定义颜色
imagefill($im, 0,0,$black);
//把计数器的底色设置成黑色
ImageString($im,$font,1,0,$str[0],$white);
for ($i=1;$i<=$n-1;$i++) {
imageline($im, $i*11-1,0,$i*11-1,16, $white);
ImageString($im,$font,$i*11+1,0,$str[$i],$white);
};
//将每位写入图象，并以竖线分隔
ImageGif($im);
//图象输出
ImageDestroy($im);
//释放图象
?>
```

另外，为了方便，还可以将计数器作为一个函数 MyCounter()，这样只需在主页开头加入 require("filename")，使 MyCounter()成为此主页的一部分，需要的时候将<? MyCounter();?>加在需要计数器的地方就可以了。

5.3 文件上载

在互联网站点上经常可以看到关于文件上传的问题，比如免费电子邮件系统中实现附件的添加，网上虚拟社区让用户上载照片等。如何不用 FTP 就能实现文件的上载？PHP 对文件的上载是无比强大和简单的。下面介绍一个基于 Unix/Linux 下运行的 PHP 文件上载程序。

5.3.1 设计上传表格

文件上载主要的任务是完成文件从本地计算机上传到服务器上去。为了做到这一点，需要做一个如图所示的表格，单击“ Browse... ”按钮就可以显示本地磁盘文件，如图 5-3 所示，允许用户选择一个文件并可以提交它。

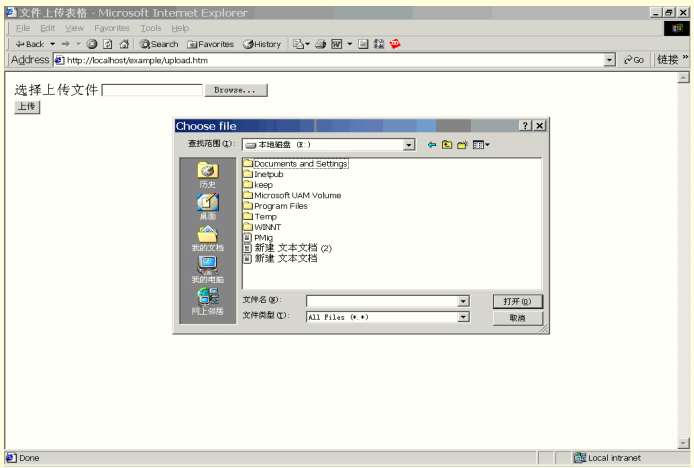


图 5-3 文件上载

Green Apple Data Center

下面是前台外观程序代码清单：

```
<html>
<head>
<title>文件上传表格</title>
</head>
<body>
<table>
<form ENCTYPE="multipart/form-data" name=myform    action=submit.php
method="post">
<input type="hidden" name="MAX_FILE_SIZE" value="100000">
<tr><td>选择上传文件</td><td><input name="myfile"
type="file"></td></tr>
<tr><td colspan="2"><input name="submit" value="上传"
type="submit"></td></tr>
</table>
</body>
</html>
```

注意程序中的 ENCTYPE="multipart/form-data"**部分，这一定不能错，否则服务器不知道在上载文件。**<input type="hidden" name="MAX_FILE_SIZE" value="100000"> **这行代码是为了限制上载文件的大小，**MAX_FILE_SIZE **隐藏字段必须在文件的输入字段之前，其值必须是系统所能接收的文件大小的最大值，该值是以字节计算。这里将不允许用户上传超过** 100kb **的文件。**

5.3.2 设计上传程序

现在已经完成了前台部分，再仔细地考虑后台是如何接收文件并保存它到指定的目录下去。下面就用 php **编写** submit.php **的程序代码清单：**

```
<?
if($myfile != "none") {
copy($myfile,"/home/duoduo/$myfile_name");
echo "file".$myfile_name."已经上载完成。"."<br>";
        echo "文件大小是".$myfile_size."字节。"."”<br>";
        echo "文件类型是".$myfile_type."<br>";
unlink($myfile);
}
else {
echo"你没有上载任何文件。";
}
?>
```

这就是整个处理过程。实现文件上载的整个步骤是：

（1）检查是否一个文件已经上传到服务器，通过 if($myfile != "none");

（2）用 copy **函数拷贝文件到指定位置。**

（3）用 echo **函数显示上载文件属性。**

（4）用 unlink **函数删除临时文件。**

Green Apple Data Center

之所以这个程序要在 Unix/Linux 下运行，熟悉 Unix/Linux 的读者立刻就会明白"/home/duoduo/$myfile_name"是 Unix/Linux 操作系统的路径。当按下了“上传”按钮后，文件将会从本地计算机上传到服务器的临时目录下，临时目录可以通过在 PHP 所运行的环境中设置环境变量进行修改，在 PHP 脚本中使用 putenv（）函数进行设置是不起作用的。在临时目录下的文件是临时文件，真正的文件名应该使用 file 字段的 name 值加上"_name"来访问它，例如上例是$MyFile_name。为了显示文件大小，可以通过 file 字段 name 属性值加上"_size"这个变量来访问。为了显示文件类型，可以通过 file 字段 name 属性值加上"_type"这个变量来访问。如表 5-1 所示。

表 5-1 变量名称及含义

变量名称	变量的含义
$file	上载文件在服务器上临时保存的文件名
$file_name	发送系统中文件的原名
$file_size	以字节计算的上载文件的大小
$file_type	如果浏览器提供该信息，那就表示文件的模仿类型，例如"image/gif"

使用 copy()函数，将临时文件$MyFile 拷贝到指定目录下，拷贝后的文件名为$MyFile_name。完成后不要忘了删除临时文件，不然临时目录下有许多不想要的文件。

最后值得注意的是需要对上载文件的目录有写的权限。如果一个用户用匿名上传文件，那他的用户名应该是"nobody"。这个用户必须对该目录有写的权限，否则可能得到一些如下面的信息：

```
Warning: Unable to create '/home/duoduo/php.txt':
Permission denied
in /home/duoduo/submit.php on line 5
```

5.4 发送电子邮件

当使用者浏览网页时，有时想发送电子邮件给 Webmaster，但是执行 Email 程序总是不方便，使用者在按下如 mailto:Webmaster@some.com，还要花段时间打开自己这儿的 Outlook。这时，如果在网页上就能提供写信发电子邮件功能的意见信箱就太好了。

整个意见信箱其实就像 Outlook 或者其他电子邮件软件，打开寄发新邮件的功能，不同的地方在于使用 Outlook 时，寄件人是固定的，而要填上收件人的地址；而网站上的意见信箱，收件人几乎都是 Webmaster，反而是要填上寄件人的电子邮件地址。当然另一个不同之处是 Outlook 处理寄信；而意见信箱是由 Web 服务器处理使用者发送的信件。

在 UNIX 的系统中，大部分和电子邮件有关的问题都和 sendmail（）函数有关，因此，意见信箱的设计开发，也是使用 sendmail 来达成所需要的功能。而 Windows 系统中，由于没有 sendmail 程序，因此本节程序无法在 Windows 系统上执行。

程序的流程如下：

（1）送出填写意见的表格到使用者的浏览器上。

（2）使用者填好后送出数据到服务器。

（3）服务器将使用者填的数据整理后，存入文件。

（4）利用 UNIX 的管道指令及 sendmail 程序将意见送给系统管理人员。

（5）服务器通知使用者意见已送出。

程序代码如下：

```
<html>
<head>
<title>意见信箱</title>
</head>
```

Green Apple Data Center

```
<body>
<?php
$mailto="yourname@hahaha.com.tw";
if (($topic!="") and ($Email!="") and ($body!="")) {
   $tmpfilename = tempnam("/tmp", "dm");
   $fp = fopen($tmpfilename, "w");
   fwrite($fp, "From: ".$Email."\n");
   fwrite($fp, "Subject: ".$topic."  <访客来信>\n\n");
   fwrite($fp, $body."\n\n");
   fwrite($fp, "送信人:".$sender."\n");
   fwrite($fp, "发信 IP:".$REMOTE_ADDR."\n");
   fclose($fp);
   $execstr="cat ".$tmpfilename." | /usr/lib/sendmail ".$mailto;
   exec($execstr);
   $execstr="echo $sender $REMOTE_HOST >> /var/log/mail.log";
   exec($execstr);
   echo "信件已送出！！本站工作人员尽快处理您的问题<p><br><br><br><br><br>";
} else {
   ?>
   <form action=<? echo($GLOBALS["PHP_SELF"]); ?> method=post>
   <table border=0>
   <tr><td>主题</td><td><input type=text size=20 name=topic></tr>
   <tr><td>姓名</td><td><input type=text size=20 name=sender></tr>
   <tr><td>Email</td><td><input type=text size=20 name=Email></tr>
   <tr><td colspan=2>内容<br><textarea cols=26 rows=10 name=body></textarea></td></tr>
   <tr><td colspan=2><div align=right><input type=submit value="送出"></td></tr>
   </table>
   </form>
   <?
}
?>
</body>
</html>
```

程序在 PHP 处理解析时，先判断使用者是否填入数据。若没有数据则送出意见表单给使用者，若有数据则表示使用者已输入相关的数据，则进行处理。

处理的原则是先将使用者填写的数据写入暂存文件中，但为了防止多使用者同时填写意见时，会造成文件被覆盖，因此需要每次都有不同的暂存档，这个问题可以使用 tempnam()函数来解决，用来建立独一无二的临时档。在文件名的问题处理完后，利用 PHP 提供的文件处理功能，将使用者填写的数据写入刚才建立的文件中，将文件关闭就初步完成。即使数据没有邮寄出去，系统仍能保存意见文件。值得注意的是，若存放在/tmp 中，有些 UNIX 的系统(如 SUN Solaris)会在重新启动系统时遗失这些数据，而有些则不会(如 Slackware Linux)，这方面可能要先规划好，要保存的话需要存在不会被清掉的目录下。

UNIX 中最强的功能就是管道，可以利用管道来处理寄信的动作，如下：

```
cat tmpfilename | /usr/lib/sendmail Webmaster@some.com
```

Green Apple Data Center

这个指令的意思为将文件送给管道彼端的 sendmail 程序，而 sendmail 将该文件寄给 Webmaster@some.com。因此可利用本管道指令将意见寄给 Webmaster 或是客服部门的人员。若要寄给多人，可利用 mailing list 或是多用几次寄信的管道指令。

在 PHP 程序中要使用 UNIX 的程序或者外部指令，可以使用 exec()函数来做。寄完信后，通知使用者已经在处理了，就完成了意见处理的初步工作。当然之后要如何处理，就不是 PHP 书中所能讨论的。

当然执行寄信的方式不只一种，可以利用 mail()函数来寄信，亦可利用 UNIX 的网络 socket 来做，所谓戏法人人会变，巧妙各有不同。

5.5 简易 banner 动态更替

不知大家有没有发现各大站点上的标头广告 banner，每次访问这些站点时，都会看到不同的广告图标，或者如果每次刷新页面时，这些广告 banner 就会不断地随机更替变换。要实现这种效果虽然用 javascript 也可以达到（象天极网站的动态变换广告 banner 就是通过调用 javascript 来实现的），但是如果用 PHP 的话，同样很容易实现这个功能，下面就立即来看看如何用 PHP 来实现 banner 的动态更替功能。

简易 banner 动态更替 PHP 文件（banner.php）：

```
<?
//产生随机数
srand((double)microtime()*1000000);
//在 0 和 4 之间取一个数字
$randval = rand(0,5);
// 显示结果
echo "<a href=//URL/index.html><img alt=进入 php 的世界 border=0 src=$randval.gif></a>";
?>
```

可以发现，实现的程序非常简单：主要是先利用 srand 这一初始化随机数产生器产生随机数，再调用 rand 函数在定义的有效范围内来获取其中一个随机值，最后显示$randval.gif 各图片 banner，即 0.gif、1.gif、2.gif、3.gif 或 4.gif。为了便于大家理解，将 rand 函数的语法及相关说明如下：

rand

语法：int rand([int min], [int max]);

返回值：整数

函数种类：数学运算

内容说明：本函数用来取得随机值。若没有指定随机数的最大及最小范围，本函数会自动地从 0 到 RAND_MAX 中取一个随机数。若有指定 min 及 max 的参数，则从指定参数中取一数字。例如 rand(38,49)则会从 38 到 49 之间取得一个随机值。其中 UNIX 系统包含 49，Win32 系统不包含 49。值得注意的是为了使随机数的随机率最大，每次在取随机数前最好使用 srand()来设定新的随机数。本例中在用 srand()来产生新的随机数时加入了时间因素，执行时以百万分之一的随机率来产生随机数

当然可以更改下面的 head.inc 文件以应用该简易的 banner 动态更替功能，同时还得为不同的广告 banner 链接到它们对应的网址。首先必须先准备好用于更换的 banner 图标，同时也给页面标头加上自己网站的徽标（如 01DC.gif）。

新的标头文件（header.inc）：

```
<?
// 定义通用页面头部
?>
<HTML>
<HEAD>
```

Green Apple Data Center

```
<TITLE> <? echo "$MySiteName - $title"; ?> </TITLE>
<style type="text/css">
<!--
.text { font-family: "宋体"; font-size: 12pt; color: #006633; text-decoration: none}
-->
</style>
</HEAD>
<body topmargin=2>
<table width="100%" border="0">
<tr>
<td rowspan="3" width="19%"><img src="http://gophp.heha.net/test/images/01dc.gif" width="140" height="60"></td>
<td rowspan="3" width="29%">
<?
//取得随机数种子
srand((double)microtime()*1000000);
//在 0 和 4 之间取一个数字
$randval = rand(0,5);
//  显示结果
switch($randval)
{
case "0";
echo "<a href=//gophp.heha.net/index.html><img border=0 src=$randval.gif></a>";
break;
case "1";
echo "<a href=//personal.668.cc/haitang/index.htm><img border=0 src=$randval.gif></a>";
break;
case "2";
echo "<a href=//gophp.heha.net/index.html><img border=0 src=$randval.gif></a>";
break;
case "3";
echo "<a href=//gophp.heha.net/index.html><img border=0 src=$randval.gif></a>";
break;
case "4";
echo "<a href=//personal.668.cc/haitang/index.htm><img border=0 src=$randval.gif></a>";
break;
}
?>
</td>
<td width="52%">
<div align="center"><a href="../test/form.php3" class="text">自动发送邮件测试</a></div>
</td>
</tr>
<tr>
<td width="52%">
<div align="center"><a href="../test/php/php1.php3" class="text">简易轮回广告更替</a></div>
</td>
</tr>
```

Green Apple Data Center

```
<tr>
<td width="52%">
<div align="center"><a href="../password/password.php3" class="text">简易密码验证实例</a></div>
</td>
</tr>
</table>
<hr color="#ff9900" size="4">
</body>
</html>
```

5.6 投票系统

在许多时候，需要收集上网者和网友们的意见。例如：新版页面与旧版页面的比较；对某一事情的看法；对体育比赛结果的预测等等。这时候，需要一个非常有效的网上调查系统，使用 PHP 就可以非常方便地实现这一构想。程序代码如下：

mypolls.php

```
<?
$status=0;
if(isset($polled)&&($polled=="c-e")){
$status=1;
}
#echo "$status";
if(isset($poll)&&($status==0)){
setcookie("polled","c-e",time()+86400,"/");#time=24h
}
?>
<html>
<head>
<title>新版页面调查</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
.tb { border="1" bordercolor="#009933" cellspacing="0" font-size: 9pt; color: #000000}
.head { font-family: "宋体"; font-size: 12pt; font-weight: bold; color: #009933; text-decoration: none}
.pt9 { font-size: 9pt}
a.p9:link { font-size: 9pt; color: #000000; text-decoration: none}
a.p9:visited { font-size: 9pt; color: #000000; text-decoration: none }
a.p9:hover { font-size: 9pt; color: #FF0000; text-decoration: underline}
a.p9:active { font-size: 9pt; color: #FF0000; text-decoration: underline }
-->
</style>
</head>
<body bgcolor="#FFFFFF">
<div class="head">与旧版页面相比较您觉得新版页面：</div><br>
<?
if(!isset($submit)){
```

Green Apple Data Center

```
?>
<form action="myPolls.php3" method="get">
<input type="radio" name="poll_voteNr" value="1" checked >
<span class="pt9">信息量更大</span> <br>
<input type="radio" name="poll_voteNr" value="2" >
<span class="pt9">网页更精美</span> <br>
<input type="radio" name="poll_voteNr" value="3" >
<span class="pt9">没什么改进</span> <br>
<input type="radio" name="poll_voteNr" value="4" >
<span class="pt9">其他</span> <br>
<input type="submit" name="submit" value="OK">
<input type="hidden" name="poll" value="vote">
<A HREF="myPolls.php3?submit=OK" class="p9">查看调查结果</A>
</form>
<?
/*
如果想增加其他的选项可直接加上即可
*/
}else{
$descArray=array(1=>"信息量更大",
2=>"网页更精美",
3=>"没什么改进",
4=>"其他"
);
$poll_resultBarHeight = 9; // height in pixels of percentage bar in result table
$poll_resultBarScale = 1; // scale of result bar (in multiples of 100 pixels)
$poll_tableHeader="<table border=1 class=\"tb\">";
$poll_rowHeader="<tr>";
$poll_dataHeader="<td align=center>";
$poll_dataFooter="</td>";
$poll_rowFooter="</tr>";
$poll_tableFooter="</table>";
$coutfile="data.pol";
$poll_sum=0;

// read counter-file
if (file_exists( $coutfile))
{
$fp = fopen( $coutfile, "rt");
while ($Line = fgets($fp, 10))
{
// split lines into identifier/counter
if (ereg( "([^ ]*) *([0-9]*)", $Line, $tmp))
{
$curArray[(int)$tmp[1]] = (int)$tmp[2];
$poll_sum+=(int)$tmp[2];
}
```

Green Apple Data Center

```
}
// close file
fclose($fp);
}else{//
for ($i=1;$i<=count($descArray);$i++){
$curArray[$i]=0;
}
}
if(isset($poll)){
$curArray[$poll_voteNr]++;
$poll_sum++;
}
echo $poll_tableHeader;

// cycle through all options 遍历数组
reset($curArray);
while (list($K, $V) = each($curArray))
{
$poll_optionText = $descArray[$K];
$poll_optionCount = $V;
echo $poll_rowHeader;
if($poll_optionText != "")
{
echo $poll_dataHeader;
echo $poll_optionText;
echo $poll_dataFooter;
if($poll_sum)
$poll_percent = 100 * $poll_optionCount / $poll_sum;
else
$poll_percent = 0;
echo $poll_dataHeader;

if ($poll_percent > 0)
{
$poll_percentScale = (int)($poll_percent * $poll_resultBarScale);
}
printf(" %.2f %% (%d)", $poll_percent, $poll_optionCount);
echo $poll_dataFooter;
}
echo $poll_rowFooter;
}
echo "总共投票次数:<font color=red> $poll_sum</font>";
echo $poll_tableFooter;
echo "<br>";
echo "<input type=\"submit\" name=\"Submit1\" value=\"返回主页\" onClick=\"javascript:location='http: //localhost/index.html'\">";
echo " <input type=\"submit\" name=\"Submit2\" value=\" 重 新 投 票 \" onClick=\"javascript:location='http:
//localhost/mypolls.php'\">";
```

Green Apple Data Center

```
if(isset($poll)){
// write counter file
$fp = fopen($coutfile, "wt");
reset($curArray);
while (list($Key, $Value) = each($curArray))
{
$tmp = sprintf( "%s %d\n", $Key, $Value);
fwrite($fp, $tmp);
}
// close file
fclose($fp);
}
}
?>
</body>
</html>
```

注释：从上面程序代码可以看出该投票系统的基本过程：

（1）打开文件取得数据到数组$curArray**（文件不存在则初始化数组**$curArray**）；**

（2）遍历数组，处理数据得到所需值；

（3）计算百分比，控制统计 bar **图像宽度；**

（4）将数据保存到"data.pol"**中。**

5.7 用 Session 对 Web 页面进行保护

在很多时候，都要对某些 Web **页面进行安全保护，典型的例子就是前台浏览页面与后台管理页面的安全性，这也是** Web **上用得最多的一种页面安全模式。**

要求目的：同一站点、无权用户、一般授权用户和超级用户能看到和使用不同的页面。

实现办法：在要保护的页面 include **不同级别的安全检验模板。**

如何使用：

（1）在需要一般保护的页面的代码最前边加上 include("secturity2.php");**就行了；**

（2）如果需要特殊保护，只要在需要保护的页面的代码最前边加上 include("sectu，rity1.php");**和** include("secturity2.php")**；就行了（假设所有文件都在同一个文件夹里）。**

程序代码及详细解释：

security1.php **特殊用户页面保护模板**

security2.php **一般用户页面保护模板**

login2.php **用户登录页面**

先来看 login2.php**（用户登录页面）的代码：**

```
<?php
session_register("user");#增加用户名变数
session_register("password");  #增加密码变数
session_register("tmLast");  #增加时间变数
if($user==""){#判断是否是第一次登录
$error="Chooseyounameandinputthepasswordplease!";
}
$tmLast=date("U");  #记录登录时间
if($user1)
```

Green Apple Data Center

```
$user=trim($user1); #记录用户名（引用 user1 变量是为什么？请读者自己思考。）
$password=trim($password1); #记录密码
if($user1&&$password1){
if($password1==888){ #判断登录密码是否是默认密码 888 结束 PHP 程式
$sid="PHPSESSID=".session_id(); #保存当前 session 的 ID 号
$warning="Yourpasswordisstillthedefaultpassword888,pleasechangeit.";
header("Location:changePassword.php?$sid&warning=$warning"); #传递警告参数 warning 到 changePassword.php 页面
exit(); #立刻结束 PHP 程式
}
if(strtolower($user)=="root"){ #判断登录用户是否是超级用户，可以自行扩充用户
$fileName="backend_index.php";
}
else{
if(!$fileName) #判断进入登录页面的上一页是否是受保护页面
$fileName="index.php";
}
$sid="PHPSESSID=".session_id(); #保存当前 session 的 ID 号
header("Location:$fileName?$sid); #登录成功进入指定页面，传递当前 session 的 ID 号，防止用户不使用 cookie 而读不到 session 值
exit(); #立刻结束 PHP 程式
}
?>
<html>
<title></title>
<head>
<linkrel="stylesheet"href="class/style.css">
<metahttp-equiv="Content-Type"content="text/html;charset=gb2312">
</head>
<h2>LoginPage</h2>
<?php
echo"$error"; #显示登录提示
?>
<formaction="<?phpecho$PHP_SELF；#提交到当前页?>"method=post>
<P><b>Name：</b>
<?php
include("class/dbclass.inc"); #调用 dbclass.inc 类，用法和 mysql.inc 类一样
$q=newDB_Sql; #定义一个新的对象
$q->connect($Host,$Database,$User,$Password); #链接 mysql 数据库
$query="selectchrUserName,chrFirstName,chrLastName".
"fromUser".
"wherechrFirstName!="".
"orderbychrFirstName";
$q->query($query); #执行 sql 语句
echo"<selectname=user1size=1>";
while($q->next_record()){ #从数据库中调出一般用户
if($user==$q->f(0)) #判断是否是当前用户
$select="selected"; #是当前用户则设置为默认值
```

Green Apple Data Center

```
else
$select="";
echo"<optionvalue='".$q->f(0)."'$select>".
ucfirst($q->f(1))."". #用户名首字大写
ucfirst($q->f(2))."</option>";
}
echo"</select>";
?></P>
<P><b>Password：</b><INPUTname=password1type=password></P>
<INPUTname=tmLasttype=hiddenvalue=<?phpechodate("U")?>>
<INPUTname=fileNametype=hiddenvalue=<?phpecho$fileName?>>
<P><INPUTname=submittype=submitvalue=确认></P>
</form>
```

security2.php（一般用户页面保护模板）

```
<?php
session_register("user"); #说明同上
session_register("password");
session_register("tmLast");
if($fileName=="")
$fileName=$PHP_SELF; #记录当前页面路径
if($durtime=="")
$durtime=300; #设置 session “失效”时间
$currtime=date("U");
if(($currtime-$tmLast)>$durtime){ #判断 session 是否“失效”
//session_destroy();
$error=urlencode("Seesionexpired.Loginagainplease!");
header("Location:login2.php?fileName=$fileName&error=$error&user=$user"); #跳到重新登录页
exit();
}
else{
$tmLast=$currtime; # session 没“失效”则更新最后“登录”时间
}
include("class/dbclass.inc");
$q=newDB_Sql;
$q->connect($Host,$Database,$User,$Password);

$query="selectidUserfromUser".
"wherechrUserName='$user'".
"andchrPasswd='$password'";
$q->query($query);
if(!$q->num_rows()){ #判断是否找到密码匹配的用户
$error=urlencode("PasswordiswrongorNoprivilegeuser.");
header("Location:login2.php?fileName=$fileName&error=$error&user=$user"); #跳到密码错误登录页
}
else{
$sid="PHPSESSID=".session_id();
```

Green Apple Data Center

```
$q->next_record();
$USERID=$q->f(idUser); #保存通过验证用户的 ID 号，方便以后使用
}
?>
```

security1.php （特殊用户页面保护模板）

```
<?php
session_register("user"); #说明同上
$privilege="root,macro,jackie"; #设置超级用户名单列表，用“，”隔开
$pieces=explode(",",$privilege); #取得单个超级用户名单
for($i=0;$i<count($pieces);$i++){
if(strtolower($user)==$pieces[$i]){ #判断是否是超级用户
$hasPrivilege=1;
break; #跳出判断循环
}
}
if(!$hasPrivilege){
if($fileName=="")
$fileName=$PHP_SELF;
$error=urlencode("Youhavenoprivilegetoviewthispage!");
header("Location:login2.php?fileName=$fileName&error=$error&id=$id");
exit(); #跳到无权用户登录页面
}
?>
```

要想使 seeion 和 cookies 完全正确地工作，关键就是在于配置好 php.ini。下面就把 php.ini 文件中关于 session 和 cookies 部分贴出来，供大家参考。

```
[Session]
session.save_handler=files   ; handler used to store/retrieve data
session.save_path=c:\temp   ; argument passed to save_handler
; in the case of files, this is the
; path where data files are stored
session.use_cookies=1      ; whether to use cookies
session.name=PHPSESSID
; name of the session
; is used as cookie name
session.auto_start=0       ; initialize session on request startup
session.cookie_lifetime=0      ; lifetime in seconds of cookie
; or if 0, until browser is restarted
session.cookie_path=/      ; the path the cookie is valid for
session.cookie_domain=; the domain the cookie is valid for
session.serialize_handler=php     ; handler used to serialize data
; php is the standard serializer of PHP
session.gc_probability=1       ; percentual probability that the
; 'garbage collection' process is started
; on every session initialization
```

Green Apple Data Center

```
session.gc_maxlifetime=1440   ; after this number of seconds, stored
; data will be seen as 'garbage' and
; cleaned up by the gc process
session.referer_check=; check HTTP Referer to invalidate
; externally stored URLs containing ids
session.entropy_length=0      ; how many bytes to read from the file
session.entropy_file=; specified here to create the session id
; session.entropy_length=16
; session.entropy_file=/dev/urandom
session.cache_limiter=nocache ; set to {nocache,private,public} to
; determine HTTP caching aspects
session.cache_expire=180      ; document expires after n minutes
```

其中最关键的是这两个地方：

```
session.save_path=c:\ temp
session.cookie_path=/;
```

第一个可以保证 session **正确使用，但如果没有第二个的支持，**cookies **就无法正确显示。当** session **和** cookies **出问题的时候，不要怀疑是程序的错误，而应该考虑一下自己的** php.ini **文件的设置。**

Green Apple Data Center

第 6 章　PHP 数据库编程

本章要点：

- MySQL 数据库的安装和设置
- SQL 语言基础
- MySQL 数据库的基本操作及应用实例

6.1　PHP 数据库功能简介

在当前互联网发展迅速、电子商务网站层出不穷的形势下，对网站开发的效率和质量提出了越来越高的要求。对于大型和结构复杂、内容繁多的网站，都要实现网站的动态化和方便的管理。数据管理离不开数据库系统的支持，而衡量一种 CGI 语言的重要标志，就是它对后台数据库的访问能力、效率等。

PHP 是一个建立动态网站的强大工具，而它最大的优点就是能够轻松处理多种数据库。PHP 具有强大的数据库支持能力，PHP 提供对 10 余种常见数据库的支持，如 Oracle、dBase、Informix、SQL Server、Sysbase、MySQL 等。正是由于广泛的数据库支持，才拓展了 PHP 的应用范围，使得各种数据库应用都可以利用 PHP 进行开发。PHP 对各种数据库的访问方法进行封装，针对不同数据库系统的函数也很相似，增加了使用的方便性。

Oracle 是典型的大型数据库应用系统。如果网站数据量大，性能、效率要求高的话，Oracle 是个不错的选择。在 Win32 平台上，SQL Server 占有较大的市场，PHP 同样可以访问 SQL Server。

在各种数据库中，MySQL 由于其免费、跨平台、使用方便、访问效率较高，获得了很大的应用。很多中心型网站都使用 PHP+MySQL 这一最佳搭档。MySQL 是一个轻型 SQL 数据库服务器，可运行在多种平台上，包括 Windows NT 和 Linux，它还有一个 GPL 版本，MySQL 被认为是建立数据库驱动的动态网站的最佳产品。PHP、MySQL 和 Apache 是 Linux 平台网站的最佳拍档。堪称 Web 数据库黄金组合的 PHP/MySQL 如何构筑一个网络数据库应用的方法是大家所关心的。所以本书就是以 MySQL 数据库介绍 PHP 强大的数据库处理功能。

MySQL 是一个小巧灵珑的数据库服务器软件，对于小型（当然也不一定很小）应用系统是非常理想的。除了支持标准的 ANSI SQL 语句，它还支持多种平台，而在 Unix 系统上该软件支持多线程运行方式，从而能获得相当好的性能。对于不使用 Unix 的用户，它可以在 Windows NT 系统上以系统服务方式运行，或者在 Windows 95/98 系统上以普通进程方式运行。MySQL 是用于服务器平台的网络数据库，自由软件。特点是速度快，比一般的网络数据库要快 2-3 倍。目前已有大量的 Linux 网站使用 MySQL。缺点是：目前的版本尚不能用于访问量特别大的网站。（同一时刻的访问人数最好不要超过 1000 人）。但 MySQL 现在发展很快，或许这个问题可以很快得到解决。

6.2　安装 MySQL

6.2.1　在 Unix 上安装 MySQL

可得到几种版本的 MySQL 分发包。当前稳定的发行版有 3.22 版本系列。当前正在开发的版本为 3.23 系列。一般，应该使用系列中最高编号的版本。

MySQL 分发包可以以二进制代码、RPM 和源代码的格式得到。二进制代码和 RPM 分发包容易安装，但必须接受建立在分发包内的安装设计和缺省配置。源代码分发包安装很困难，因为必须对软件进行编译，但可对参数进行更多的控制。例如，可以只编译客户机程序的分发包，而不用管服务器，可以更改安装软件的目标位置等。

分发包含有下列一个或多个组件：

（1）mysqld 服务器。

Green Apple Data Center

（2）客户机程序（mysql、mysqladmin 等）以及客户机编程支持环境（库和头文件）。

（3）文档。

（4）标准数据库。

（5）语言支持环境。

源程序和二进制代码分发包含有上述所有内容。每个 RPM 文件只包含其中一些内容，因此为了得到所需的东西，可能需要安装多个 R P M。如果打算链接到其他机器上运行的服务器，则不需要安装服务器，但应该安装客户机软件。

（6）如果不运行服务器，那么只需要客户机，以便能够链接到其他机器的服务器上。

（7）如果确实运行一个服务器，将希望能够从该服务器的主机对此服务器进行链接，而不是在其他具有客户机软件的机器上登录，然后再来测试您的服务器。

1. MySQL 安装综述

在 Unix 上安装 MySQL 涉及下列步骤：

（1）创建一个用户的 Unix 账号和服务器将操纵的组（如果正在安装服务器）。

（2）获得和打开想安装的分发包。如果使用的是源代码分发包，编译并安装它。

（3）运行 mysql_install_db 脚本来初始化数据目录和权限表（仅允许第一次安装）。

（4）启动服务器。

（5）熟悉一般的管理过程。特别应该阅读服务器设置和关闭以及作为无特权。

2. 安装二进制分发包

对于安装二进制分发，遵循以下步骤：

（1）选择一个想解开分发包的目录，进入该目录。在下面的例子中，将分发包解包在/usr/local 下并且创建一个安装 MySQL 的/usr/local/mysql 目录。（因此，下列指令假设有权限在/usr/local 中创建文件，并且如果该目录被保护，需要以 root 实施安装。）

（2）解开分发包并且创建立安装目录：

```
shell> gunzip < mysql-VERSION-OS.tar.gz |tar    xvf -
shell> ln - s    mysql-VERSION-OS    mysql
```

第一个命令创建一个名为 mysql-VERSION-OS 的目录，第二个命令生成到该目录的一个符号链接。这样更容易把安装目录指定为/usr/local/mysql。

（3）进入安装目录：

```
shell> cd    mysql
```

会在 mysql 目录下发现几个文件和子目录，安装目录最重要的是 bin 和 scripts 子目录。bin 这个目录包含客户程序和服务器，应该把这个目录的完整路径加到 PATH 环境变量，以便 shell 能正确的找到 MySQL 程序。scripts 这个目录包含 mysql_install_db 脚本，用来初始化服务器存取权限。

（4）启动 MySQL 服务器：

```
shell> bin/safe_mysqld &
```

3. 安装源代码分发包

MySQL 源代码分发以压缩的 tar 档案提供，并且有类似于 mysql-VERSION.tar.gz 的名字，这里的 VERSION 是一个类似 3.23.7-alpha 的数字。

（1）以 root 身份进入，选择一个要在其下面解包分发包的目录，并且进入该目录下解包分发包：

Green Apple Data Center

```
shell> gunzip < mysql-VERSION.tar.gz | tar    xvf -
```

这个命令创建名为 mysql-VERSION 的一个目录。

（2）进入解包分发的顶级目录：

```
shell> cd    mysql-VERSION
```

（3）设置发行版本并且编译：

```
shell>./configure--prefix=/usr/local/mysql
shell>make
```

（4）安装所有东西：

```
shell>make install
```

（5）创造 MySQL 授权表（只有以前没安装 MySQL 需要）。

```
shell>scripts/mysql_install_db
```

（6）启动 MySQL 服务器，这里 BINDIR 是 safe_mysqld 被安装的目录（缺省为/usr/local/bin）。

```
shell> BINDIR/safe_mysqld &
```

4. 安装 RPM 分发包

在 Linux 上安装 MySQL 推荐的方法是使用一个 RPM 文件。在 REDHAT 版本的 LINUX 上安装 MYSQL 是一件非常容易的事情。需要使用的 RPM 软件包有：

（1）MySQL-VERSION.i386.rpm MySQL 服务器。除非只是想要与运行在其他机器上 MySQL 服务器链接，否则将需要它。

（2）MySQL-client-VERSION.i386.rpm 标准 MySQL 客户程序。可能总是需要安装这个包。

（3）MySQL-bench-VERSION.i386.rpm 测试和基准程序。需要 Perl 和 msql-mysql- modules RPM。

（4）MySQL-devel-VERSION.i386.rpm 所需的库和包含文件。如果想要编译其他 MySQL 客户程序,例如 Perl 模块。

（5）MySQL-VERSION.src.rpm 包含上述所有包的源代码。它也能被用来尝试为其他硬件平台构造 RPM(例如，Alpha 或 SPARC)。

下面以 MySQL-3.22.27-2c 版本介绍安装步骤如下：

（1）以 root 身份登入 Redhat 操作系统。

（2）依次用 rpm-ivh 命令将软件包安装入 Redhat 操作系统中。

```
[root@test /root]# rpm -ivh MySQL-3.22.27-2c.i386.rpm
[root@test /root]# rpm -ivh MySQL-client-3.22.27-2c.i386.rpm
[root@test /root]# rpm -ivh MySQL-devel-3.22.27-2c.i386.rpm
```

（3）以 root 身份登入 Redhat 系统后，进入 MySQ 数据库系统。

```
[root@test /root]# mysql mysql
```

（4）若安装成功则可以看到下列文字和 mysql 的提示符。

Green Apple Data Center

```
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 212 to server version: 3.22.27
Type 'help' for help.
mysql>
```

（5）更改 MySQL 系统的管理员密码。

```
mysql> UPDATE user SET password=password('新密碼') where user='root';
Query OK, 0 rows affected (0.00 sec)
Rows matched: 2  Changed: 0  Warnings: 0
```

（6）删除空帐号，使系统更安全。

```
mysql> DELETE FROM user WHERE User = '';
Query OK, 0 rows affected (0.00 sec)
Rows matched: 2  Changed: 0  Warnings: 0
```

6.2.2 在 Windows 上安装 MySQL

可在 Windows 95/98、Windows NT 和 Windows 2000 下运行 MySQL。为了做到这一点，必须安装 TCP/IP 支持环境，而且 Winsock 软件必须至少为版本 2。

在 Windows 下可安装两种软件：

（1）独立程序，如为 Unix 安装的那种程序（mysqld 服务器与诸如 mysql 和 mysladmin 这样的程序）。

（2）MyODBC，允许其他程序（如 Access）与 MySQL 服务器通信的 ODBC 的 MySQL 驱动。

Windows 分发包全都可从 MySQL 网站上的 zip 文件得到。为了打开这样的文件，只需双击它即可。如果这样不行，可使用诸如 Winzip 或 pkunzip 这样的程序来打开它。主要分发包如下：

（1）mysqlwin.version.zip 完全分发包（服务器和客户机）。

（2）winclients-version.zip 客户机软件（mysql、mysqladmin、mysqldump 等等）。如果不想在 Windows 下运行服务器，可使用这个软件。

（3）myodbc-version-win 95.zip

myodbc-version-nt.zip

Windows 95/98/2000 或 Windows NT 的 MyODBC 支持环境。

（4）mysqlclient-version-cygwin-b20.tar.gz 用 Cygnus 工具集编译的 MySQL 客户机。它包括 mysqlc，具有命令行历史编辑功能的一种 mysql 客户机版本。如果安装它，需要从 c:\mysql\lib 拷贝库文件 cygwinb19.dll 到 Windows 系统目录。

下面以 Windows 2000 为例介绍它的安装过程：

（1）先安装 mysql，用 winzip8.0 打开软件包进行完全安装，默认安装路径为:c:\mysql。

（2）安装完成后，单击“开始”按钮中的“运行”，输入命令：C:\mysql\bin\mysqld-nt.exe --install，并执行。

（3）单击“开始”按钮下的“程序”→“管理工具”→“服务”，找到“mysql”服务，启动它。

（4）至此，mysql 安装完成，重启 Windows 2000。还可以打开 C:\mysql\bin\winmy-sqladmin.exe，在第一次用它时，需要建立管理员名及密码，可以分别设置为 root 和 yourpassword。关闭它后，程序自动在状态行下建立一个“红绿灯”的小图标。

Green Apple Data Center

6.3 SQL 语言基础

为了建立交互网站，需要使用数据库来存储来自访问者的信息。例如，要建立一个职业介绍服务的网站，就需要存储诸如个人简历，所感兴趣的工作等等这样的信息。创建动态网页也需要使用数据库，如果想显示符合来访者要求的最好的工作，就需要从数据库中取出这份工作的信息，在许多情况下需要使用数据库。

在这一节里，介绍怎样使用“结构化查询语言”（SQL）来操作数据库。SQL 语言是数据库的标准语言。在 PHP 数据库编程中，无论何时要访问一个数据库，就要使用 SQL 语言。因此，掌握好 SQL 对 PHP 编程是非常重要的。如图读者对 SQL 语言比较熟悉，则可以跳过这一节的内容。

☞ 注意

可以把“SQL”读作“sequel”，也可以按单个字母的读音读作 S－Q－L。两种发音都是正确的，每种发音各有大量的支持者。在本书里，认为“SQL”读作“sequel”。

通过这一节的学习，读者将理解怎样用 SQL 实现 PHP 数据库查询，将学会怎样使用这种查询从数据表中取出信息，最后，将学会怎样设计和建立自己的数据库，尤其是和 PHP 紧密结合的 mysql 数据库。

☞ 注意

通过下面对 SQL 的介绍，读者将对 SQL 有足够的了解，从而可以有效地使用 PHP 数据库编程。但是，SQL 是一种复杂的语言，本书不可能包括它的全部细节，只是介绍最常用和一些基础知识，只要掌握了这些知识，就可以进行大多数的 PHP 数据库编程。如果要全面掌握 SQL 语言，可以参考其他详细介绍 SQL 语言的书籍。

6.3.1 SQL 介绍

SQL 是操作数据库的标准语言，事实上，关于 SQL 语言有一个专门的 ANSI 标准。在学习 SQL 的细节之前，需要理解它的两大特点。一个特点容易掌握，另一个掌握起来有点困难。

第一个特点是所有 SQL 数据库中的数据都存储在表中。一个表由行和列组成。例如，下面这个简单的表包括 name 和 e-mail address：

Name	Email Address
Bill Gates	billg@microsoft.com
president Clinton	president@whitehouse.com
Stephen Walther	swalther@somewhere.com

这个表有两列（列也称为字段，域）：Name 和 Email Address。有三行，每一行包含一组数据。一行中的数据组合在一起称为一条记录。无论何时向表中添加新数据，就添加了一条新记录。一个数据表可以有几十个记录，也可以有几千甚至几十亿个记录。虽然也许永远不需要存储十亿个 Email 地址，但知道能这样做总是好的，也许有一天会有这样的需要。数据库很有可能包含几十个表，所有存储在数据库中的信息都被存储在这些表中。当考虑怎样把信息存储在数据库中时，应该考虑怎样把它们存储在表中。

SQL 的第二个特点有些难于掌握。这种语言被设计为不允许按照某种特定的顺序来取出记录，因为这样做会降低 SQL Sever 读取记录的效率。使用 SQL，只能按查询条件来读取记录。当考虑如何从表中取出记录时，自然会想到按记录的位置读取它们。例如，也许会尝试通过一个循环，逐个记录地扫描，来选出特定的记录。在使用 SQL 时，必须训练自己不要有这种思路。假如想选出所有的名字是“Bill Gates”的记录，如果使用传统的编程语言，也许会构造一个循环，逐个查看表中的记录，看名字域是否是“Bill Gates”。

这种选择记录的方法是可行的，但是效率不高。使用 SQL 只要选择所有名字域等于 Bill Gates 的记录，SQL 就会选出所有符合条件的记录，SQL 会确定实现查询的最佳方法。

如果想取出表中的前十个记录。使用传统的编程语言，可以做一个循环，取出前十个记录后结束循环。但

Green Apple Data Center

使用标准的 SQL 查询，这是不可能实现的。从 SQL 的角度来说，在一个表中不存在前十个记录这种概念。

综上所述，SQL 有两个特点：所有数据存储在表中，从 SQL 的角度来说，表中的记录没有顺序。在下一节，将学会怎样用 SQL 从表中选择特殊的记录。

6.3.2 用 SQL 创建新表

如果要创建一个表来存储一个网站访问者的信息，可用 CREATE TABLE 语句创建新表，例如：

```
CREATE TABLE guestbook (visitor VARCHAR(40),comments TEXT,entrydate
                                    DATETIME)
```

所创建的表名为 guestbook，可以使用这个表来存储网站访问者的信息。这个语句有两部分：第一部份指定表的名子；第二部分是括在括号中的各字段的名称和属性，相互之间用逗号隔开。

表 guestbook 有三个字段：visitor,comments 和 entrydate。visitor 字段存储访问者的名字，comments 字段存储访问者对网站的意见，entrydate 字段存储访问者访问网站的日期和时间。

注意每个字段名后面都跟有一个专门的表达式。例如，字段名 comments 后面跟有表达式 TEXT。这个表达式指定了字段的数据类型。数据类型决定了一个字段可以存储什么样的数据。因为字段 comments 包含文本信息，其数据类型定义为文本型。

字段有许多不同的数据类型。下面讲述 SQL 所支持的一些重要的数据类型。

1. 字段类型

不同的字段类型用来存放不同类型的数据。创建和使用表时，更应该理解五种常用的字段类型：字符型、文本型、数值型、逻辑性和日期型。

（1）字符型数据

字符型数据非常有用。当需要存储短的字符串信息时，总是要用到字符型数据。例如，可以把从 HTML form 的文本框中搜集到的信息放在字符型字段中。

要建立一个字段用来存放可变长度的字符串信息，可以使用表达式 VARCHAR。考虑前面创建的表 guestbook：

```
CREATE TABLE guestbook (visitor VARCHAR(40),comments TEXT,entrydate
                                    DATETIME)
```

在这个例子中，字段 visitor 的数据类型为 VARCHAR。注意跟在数据类型后面的括号中的数字。这个数字指定了这个字段所允许存放的字符串的最大长度。在这个例子中，字段 visitor 能存放的字符串最长为四十个字符。如果名字太长，字符串会被截断，只保留四十个字符。

VARCHAR 类型可以存储的字符串最长为 255 个字符。要存储更长的字符串数据，可以使用文本型数据（TEXT）。

另一种字符型数据用来存储固定长度的字符数据。下面是一个使用这种数据类型的例子：

```
CREATE TABLE guestbook (visitor CHAR(40),comments TEXT,entrydate
                                    DATETIME)
```

在这个例子中，字段 visitor 被用来存储四十个字符的固定长度字符串。表达式 CHAR 指定了这个字段应该是固定长度的字符串。

VARCHAR 型和 CHAR 型数据的这个差别是细微的，但是非常重要。假如向一个长度为四十个字符的 VARCHAR 型字段中输入数据 Bill Gates。当以后从这个字段中取出此数据时，取出的数据其长度为十个字符——字符串 Bill Gates 的长度。

假如把字符串输入一个长度为四十个字符的 CHAR 型字段中，那么当取出数据时，所取出的数据长度将是

Green Apple Data Center

四十个字符。字符串的后面会被附加多余的空格。

当建立自己的网站时，会发现使用 VARCHAR 型字段要比 CHAR 型字段方便的多。使用 VARCHAR 型字段时，不需要为剪掉数据中多余的空格而操心。

VARCHAR 型字段的另一个突出的好处是它可以比 CHAR 型字段占用更少的内存和硬盘空间。当数据库很大时，这种内存和磁盘空间的节省会变得非常重要。

（2）文本型数据

字符型数据限制了字符串的长度不能超过２55 个字符。而使用文本型数据，可以存放超过二十亿个字符的字符串。当需要存储大串的字符时，应该使用文本型数据。

这里有一个使用文本型数据的例子：

```
CREATE  TABLE  guestbook (visitor VARCHAR(40),comments TEXT,entrydate
                                    DATETIME)
```

在这个例子中，字段 comments 被用来存放访问者对网站的意见。注意文本型数据没有长度，而上面所讲的字符型数据是有长度的。一个文本型字段中的数据通常要么为空，要么很大。

当从 HTML form 的多行文本编辑框（TEXTAREA）中收集数据时，应该把收集的信息存储于文本型字段中。但是，无论何时，只要能避免使用文本型字段，就应该不适用它。文本型字段既大且慢，滥用文本型字段会使服务器速度变慢。文本型字段还会吃掉大量的磁盘空间。

☞ 注意

一旦向文本型字段中输入了任何数据（甚至是空值），就会有 2K 的空间被自动分配给该数据。除非删除该记录，否则无法收回这部分存储空间。

（3）数值型数据

MySQL 支持许多种不同的数值型数据，可以存储整数、小数和货币等。

通常，当需要在表中存放数字时，要使用整型（INT）数据。INT 型数据的表数范围是从-2，147，483，647 到 2，147，483，647 的整数。下面是一个如何使用 INT 型数据的例子：

```
CREATE  TABLE  visitlog (visitor VARCHAR(40),numvisits  INT)
```

这个表可以用来记录网站被访问的次数。只要没有人访问该网站超过 2，147，483，647 次，nubvisits 字段就可以存储访问次数。

为了节省内存空间，可以使用 SMALLINT 型数据。SMALLINT 型数据可以存储从-32768 到 32768 的整数，这种数据类型的使用方法与 INT 型完全相同。

最后，如果实在需要节省空间，可以使用 TINYINT 型数据。同样，这种类型的使用方法也与 INT 型相同，不同的是这种类型的字段只能存储从０到 255 的整数。TINYINT 型字段不能用来存储负数。

通常，为了节省空间，应该尽可能的使用最小的整型数据。一个 TINYINT 型数据只占用一个字节；一个 INT 型数据占用四个字节。这看起来似乎差别不大，但是在比较大的表中，字节数的增长是很快的。另一方面，一旦已经创建了一个字段，要修改它是很困难的。因此，为安全起见，应该预测一下，一个字段所需要存储的数值最大有可能是多大，然后选择适当的数据类型。

为了能对字段所存放的数据有更多的控制，可以使用 NUMERIC 型数据来同时表示一个数的整数部分和小数部分。NUMERIC 型数据能表示非常大的数——比 INT 型数据要大得多。一个 NUMERIC 型字段可以存储从 -10^{38} 到 10^{38} 范围内的数。NUMERIC 型数据还能表示有小数部分的数。例如，可以在 NUMERIC 型字段中存储小数 3.14。

当定义一个 NUMERIC 型字段时，需要同时指定整数部分的大小和小数部分的大小。这里有一个使用这种数据类型的例子：

```
CREATE TABLE  numeric_data (bignumber  NUMERIC(28,0),
                                    fraction      NUMERIC (5,4) )
```

Green Apple Data Center

当这个语句执行时，将创建一个名为 numeric_data 包含两个字段的表。字段 bignumber 可以存储直到 28 位的整数。字段 fraction 可以存储有五位整数部分和四位小数部分的小数。

一个 NUMERIC 型数据的整数部分最大只能有 28 位，小数部分的位数必须小于或等于整数部分的位数，小数部分可以是零。

可以使用 INT 型或 NUMERIC 型数据来存储货币。但是，专门有另外两种数据类型用于此目的。如果希望存储很多钱，可以使用 MONEY 型数据。如果钱不多，可以使用 SMALLMONEY 型数据。MONEY 型数据可以存储从-922，337，203，685，477.5808 到 922，337，203，685，477.5807 的钱数。如果需要存储比这还大的金额，可以使用 NUMERIC 型数据。

SMALLMONEY 型数据只能存储从-214，748.3648 到 214，748.3647 的钱数。同样，如果可以的话，应该用 SMALLMONEY 型来代替 MONEY 型数据，以节省空间。下面的例子显示了如何使用这两种表示货币的数据类型：

```
CREATE TABLE products (product VARCHAR(40),price    MONEY,
                                    Discount_price SMALLMONEY)
```

这个表可以用来存储商品的折扣和普通售价。字段 price 的数据类型是 MONEY，字段 discount_price 的数据类型是 SMALLMONEY。

（4）存储逻辑值

如果使用复选框（CHECKBOX）从网页中搜集信息，可以把此信息存储在 BIT 型字段中。BIT 型字段只能取两个值：0 或 1。这里有一个如何使用这种字段的例子：

```
CREATE    TABLE    opinion (visitor VARCHAR(40),good BIT)
```

这个表可以用来存放对网站进行民意调查所得的信息。访问者可以投票表示是否喜欢该网站。如果投 YES，就在 BIT 型字段中存入 1。反之，就在字段中存入 0。

注意，在创建好一个表之后，不能向表中添加 BIT 型字段。如果打算在一个表中包含 BIT 型字段，必须在创建表时完成。

（5）存储日期和时间

当建立一个网站时，也许需要记录在一段时间内的访问者数量。为了能够存储日期和时间，需要使用 DATETIME 型数据，如下例所示：

```
CREATE    TABL visitorlog( visitor VARCHAR (40), arrivaltime DATETIME ,
                                    departuretime    DATETIME)
```

这个表可以用来记录访问者进入和离开网站的时间和日期。一个 DATETIME 型的字段可以存储的日期范围是从１７５３年１月１日第一毫秒到 9999 年 12 月 31 日最后一毫秒。

如果不需要覆盖这么大范围的日期和时间，可以使用 SMALLDATETIME 型数据。它与 DATETIME 型数据同样使用，只不过它能表示的日期和时间范围比 DATETIME 型数据小，而且不如 DATETIME 型数据精确。一个 SMALLDATETIME 型的字段能够存储从 1900 年 1 月 1 日到 2079 年 6 月 6 日的日期，它只能精确到秒。

DATETIME 型字段在输入日期和时间之前并不包含实际的数据，认识这一点是重要的。实际应用时，可以在 VBScript 和 JScript 中使用日期和时间函数来向一个 DATETIME 型字段中输入日期和时间。

2．字段属性

上一小节介绍了如何建立包含不同类型字段的表。在这一小节中，将学会如何使用字段的三个属性。这些属性允许控制空值、缺省值和标识值。

（1）允许和禁止空值

Green Apple Data Center

大多数字段可以接受空值（NULL）。当一个字段接受了空值后，如果不改变它，它将一直保持空值。空值（NULL）和零是不同的，严格的说，空值表示没有任何值。

为了允许一个字段接受空值，要在字段定义的后面使用表达式 NULL。例如，下面的表中两个字段都允许接受空值：

```
CREATE TABLE empty (empty1 CHAR (40) NULL,empty2 INT NULL)
```

☞ 注意

BIT 型数据不能是空值。一个这种类型的字段必须取 0 或者 1。

有时需要禁止一个字段使用空值。例如，假设有一个表存储着信用卡号码和信用卡有效日期，不希望有人输入一个信用卡号码但不输入有效日期。为了强制两个字段都输入数据，可以用下面的方法建立这个表：

```
CREATE    TABLE creditcards (creditcard_number CHAR(20) NOT NULL,
                                Creditcard_expire    DATETIME    NOT NULL)
```

注意字段定义的后面跟有表达式 NOT NULL。通过包含表达式 NOT NULL，可以禁止任何人只在一个字段中插入数据，而不输入另一个字段的数据。

在建设网站过程中，这种禁止空值的能力是非常有用的。如果指定一个字段不能接受空值，那么当试图输入一个空值时，会有错误警告。这些错误警告可以为程序调试提供有价值的线索。

（2）缺省值

假设有一个存储地址信息的表，这个表的字段包括国家、省份、城市、街道、邮政编码。如果预计地址的大部分是在中国，可以把这个值作为 country 字段的缺省值。

为了在创建一个表时指定缺省值，可以使用表达式 DEFAULT。请看下面这个在创建表时使用缺省值的例子：

```
CREATE TABLE addresses (street VARCHAR(60)    NULL,
                        city VARCHAR(40) NULL,
                        province    VARCHAR(20) NULL
                        zip    VARCHAR(20) NULL,
                        country VARCHAR(30) DEFAULT    '中国' )
```

在这个例子中，字段 country 的缺省值被指定为中国。注意单引号的使用，引号指明这是字符型数据。为了给非字符型的字段指定缺省值，不要把该值扩在引号中：

```
CREATE TABLE orders(price    MONEY DEFAULT $38.00,
                    quantity INT DEFAULT 50,
                    entrydate DATETIME DEFAULT GETDATE())
```

在这个 CREATE TABLE 语句中，每个字段都指定了一个缺省值。注意 DATETIME 型字段 entrydate 所指定的缺省值，该缺省值是函数 Getdate()的返回值，该函数返回当前的日期和时间。

（3）标识字段

每个表可以有一个也只能有一个标识字段。一个标识字段是唯一标识表中每条记录的特殊字段。例如，数据库 pubs 中的表 jobs 包含了一个唯一标识每个工作标识字段：

```
job_id      job_desc
```

Green Apple Data Center

New Hire Job not specified
Chief Executive officer
Bushness Operations Manager
Chief Financial Officier
Publisher

字段 job_id 为每个工作提供了唯一的一个数字。如果决定增加一个新工作，新增记录的 job_id 字段会被自动赋给一个新的唯一值。

为了建立一个标识字段，只需在字段定义后面加上表达式 IDENTITY 即可。只能把 NUMERIC 型或 INT 型字段设为标识字段，这里有一个例子：

```
CREATE TABLE visitorID (theID NUBERIC(18) IDENTITY,name VARCHAR(40))
```

这个语句所创建的表包含一个名为 theid 的标识字段。每当一个新的访问者名字添加到这个表中时，这个字段就被自动赋给一个新值。可以用这个表为网站的每一个用户提供唯一标识。

技巧

建立一个标识字段时，注意使用足够大的数据类型。例如使用 TINYINT 型数据，那么只能向表中添加 255 个记录。如果预计一个表可能会变得很大，应该使用 NUMERIC 型数据。

标识字段的存在会有许多想得到，但不可能做到的事情。例如，也许想利用标识字段来对记录进行基于它们在表中位置的运算（应该抛弃这种意图）。每个记录的标识字段的值是互不相同的，但是，这并不禁止一个标识字段的标识数字之间存在间隔。例如，永远不要试图利用一个表的标识字段来取出表中的前十个记录。这种操作会导致失败，比如说 6 号记录和 7 号记录根本不存在。

6.3.3 删除和修改表

应该在建立表之前仔细设计它们，因为在改变一个已经存在的表时会受到很大的限制。例如，一旦已经建立了一个表，就不能删除表中的字段或者改变字段的数据类型。在这种情况下所能做的是删除这个表，然后重头开始。

要删除一个表，可以使用 SQL 语句 DROP TABLE。例如，在数据库中彻底删除表 mytable，要使用如下的语句：

```
DROP TABLE mytable
```

注意

使用 DROP TABLE 命令时一定要小心。一旦一个表被删除之后，将无法恢复它。

当建设一个网站时，很可能需要向数据库中输入测试数据。而当测试没有错误时，会想清空表中的这些测试信息。如果要清除表中的所有数据但不删除这个表，可以使用 TRUNCATE TABLE 语句。例如，下面的这个 SQL 语句从表 mytable 中删除所有数据：

```
TRUNCATE TABLE mytable
```

虽然不能删除和修改已经存在的字段，但可以增加新字段，使用 SQL 语句 ALTER TABLE。下面是一个如何使用这种语句的例子：

```
ALTER TABLE mytable ADD mynewcolumn INT NULL
```

这个语句向表 mytable 中增加了一个新字段 mynewcolumn。当增加新字段时，必须允许它接受空值，原因非常简单，因为表中原来可能已经有了许多记录。

Green Apple Data Center

6.3.4 使用 SQL 从表中取记录

1. 操作一个表

SQL 的主要功能之一是实现数据库查询。如果读者熟悉 Internet 引擎，那么就已经熟悉查询了。使用查询来取得满足特定条件的信息。例如，如果想找到有 PHP 信息的全部网站，可以链接到 Yahoo!并执行一个对 PHP 的搜索。在输入这个查询后，就会收到一个列表，表中包括所有其描述中包含搜索表达式的网站。

多数 Internet 引擎允许逻辑查询。在逻辑查询中，可以包括特殊的运算符如 AND、OR 和 NOT，使用这些运算符来选择特定的记录。例如，可以用 AND 来限制查询结果,如果执行一个对 PHP AND SQL 的搜索，将会得到其描述中包含 PHP 和 SQL 的记录。如果需要扩展查询的结果，可以使用逻辑操作符 OR。例如，如果执行一个搜索，搜索所有的描述中包含 PHP OR SQL 的网站，收到的列表中将包括所有描述中同时包含两个表达式或其中任何一个表达式的网站。

如果想从搜索结果中排除特定的网站，可以使用 NOT。例如，查询“ PHP ”AND NOT“ SQL ”,将返回一个列表，列表中的网站包含 PHP，但不包含 SQL。当必须排除特定的记录时，可以使用 NOT。

用 SQL 执行的查询与用 Internet 搜索引擎执行的搜索非常相似。当执行一个 SQL 查询时，通过使用包括逻辑运算符的查询条件，可以得到一个记录列表。此时查询结果是来自一个或多个表。

SQL 查询的方法非常简单。假设有一个名为 email_table 的表，包含名字和地址两个字段，要得到 Bill Gates 的 e_mail 地址,可以使用下面的查询：

```
SELECT email from email_table WHERE name="Bill Gates"
```

当这个查询执行时，就从名为 email_table 的表中读取 Bill Gates 的 e_mail 地址。这个简单的语句包括三部分：

（1）SELECT 语句的第一部分指明要选取的列。在此例中，只有 email 列被选取。当执行时，只显示 email 列的值 billg@microsoft.com。

（2）SELECTT 语句的第二部分指明要从哪个（些）表中查询数据。在此例中，要查询的表名为 email_table。

（3）最后，SELECT 语句的 WHERE 子句指明要选择满足什么条件的记录。在此例中，查询条件为只有 name 列的值为 Bill Gates 的记录才被选取。

Bill Gates 很有可能拥有不止一个 email 地址。如果表中包含 Bill Gates 的多个 email 地址。用上述的 SELECT 语句可以读取所有的 email 地址。SELECT 语句从表中取出所有 name 字段值为 Bill Gates 的记录的 email 字段的值。

前面说过，查询可以在查询条件中包含逻辑运算符。假如想读取 Bill Gates 或 Clinton 总统的所有 email 地址，可以使用下面的查询语句：

```
SELECT email FROM email_table WHERE name="Bill Gates" OR
name="president Clinton"
```

此例中的查询条件比前一个复杂了一点。这个语句从表 email_table 中选出所有 name 列为 Bill Gates 或 president Clinton 的记录。如果表中含有 Bill Gates 或 president Clinton 的多个地址，所有的地址都被读取。

2. 操作多个表

到现在为止，只用一句 SQL 查询从一个表中取出数据。也可以用一个 SELECT 语句同时从多个表中取出数据，只需在 SELECT 语句的 FROM 从句中列出要从中取出数据的表名称即可：

```
SELECT au_lname ,title FROM authors, titles
```

这个 SELECT 语句执行时，同时从表 authors 和表 titles 中取出数据。从表 authors 中取出所有的作者名字，

Green Apple Data Center

从表 titles 中取出所有的书名。出了什么差错？问题在于没有指明这两个表之间的关系。没有通过任何方式告诉 SQL 如何把表和表关联在一起。由于不知道如何关联两个表，服务器只能简单地返回取自两个表中的记录的所有可能组合。

要从两个表中选出有意义的记录组合，需要通过建立两表中字段的关系来关联两个表。要做到这一点的途径之一是创建第三个表，专门用来描述另外两个表的字段之间的关系。

表 authors 有一个名为 au_id 的字段，包含有每个作者的唯一标识。表 titles 有一个名为 title_id 的字段，包含每个书名的唯一标识。如果能在字段 au_id 和字段 title_id 之间建立一个关系，就可以关联这两个表。数据库 pubs 中有一个名为 titleauthor 的表，正是用来完成这个工作。表中的每个记录包括两个字段，用来把表 titles 和表 authors 关联在一起。下面的 SELECT 语句使用了这三个表以得到正确的结果：

```
SELECT    au_name,title FROM authors,titles,titleauthor
                    WHERE    authors.au_id=titleauthor.au_id
                      AND       titles.title_id=titleauthor.title_id
```

当这个 SELECT 语句执行时，每个作者都将与正确的书名相匹配。表 titleauthor 指明了表 authors 和表 titles 的关系，它通过包含分别来自两个表的各一个字段实现这一点。第三个表的唯一目的是在另外两个表的字段之间建立关系。它本身不包含任何附加数据。

注意在这个例子中字段名是如何书写的。为了区别表 authors 和表 titles 中相同的字段名 au_id，每个字段名前面都加上了表名前缀和一个句号。名为 author.au_id 的字段属于表 authors，名为 titleauthor.au_id 的字段属于表 titleauthor，两者不会混淆。

通过使用第三个表，可以在两个表的字段之间建立各种类型的关系。例如，一个作者也许写了许多不同的书，或者一本书也许有许多不同的作者共同完成。当两个表的字段之间有这种“多对多”的关系时，需要使用第三个表来指明这种关系。

但是，在许多情况下，两个表之间的关系并不复杂。比如需要指明表 titles 和表 publishers 之间的关系。因为一个书名不可能与多个出版商相匹配，不需要通过第三个表来指明这两个表之间的关系。要指明表 titles 和表 publishers 之间的关系，只要让这两个表有一个公共的字段就可以了。在数据库 pubs 中，表 titles 和表 publishers 都有一个名为 pub_id 的字段。如果想得到书名及其出版商的一个列表，可以使用如下的语句：

```
SELECT    title,pub_name    FROM    titles,publishers
                    WHERE titles.pub_id=publishers.pub_id
```

当然，如果一本书是由两个出版商联合出版的，那么需要第三个表来代表这种关系。通常，当先知道两个表的字段间存在“多对多”关系时，就使用第三个表来关联这两个表。反之，如果两个表的字段间只有“一对一”或“一对多”关系，可以使用公共字段来关联它们。

3. 操作字段

通常，当从一个表中取出字段值时，该值与创建该表时所定义的字段名联系在一起。如果从表 authors 中选择所有的作者名字，所有的值将会与字段名 au_lname 相联系。但是在某些情况下，需要对字段名进行操作。在 SELECT 语句中，可以在缺省字段名后面紧跟一个新名字来取代它。例如，可以用一个更直观易读的名字 Author Last Name 来代替字段名 au_lname：

```
SELECT au_lname "Author Last Name" FROM authors
```

当这个 SELECT 语句执行时，来自字段 au_lname 的值会与“Author Last Name”相联系。查询结果可能是这样：

Green Apple Data Center

```
Author Last Name
.....
White
Green
Carson
O' Leary
Straight
…
(23 row(s) affected)
```

注意字段标题不再是 au_lname**，而是被** Author Last Name **所取代。**

也可以通过执行运算，来操作从一个表返回的字段值。例如，如果想把表 titles **中的所有书的价格加倍，可以使用下面的** SELECT **语句：**

```
SELECT  price*2 FROM titles
```

当这个查询执行时，每本书的价格从表中取出时都会加倍。但是，通过这种途径操作字段不会改变存储在表中的书价。对字段的运算只会影响 SELECT **语句的输出，而不会影响表中的数据。为了同时显示书的原始价格和涨价后的新价格，可以使用下面的查询：**

```
SELECT  price  "Original  price", price*2  "New price" FROM  titles
```

当数据从表 titles **中取出时，原始价格显示在标题** Original price **下面，加倍后的价格显示在标题** New price **下面。结果可能是这样：**

```
original  price          new  price
.....
39.98
11.95                    23.90
5.98
39.98
…
(18 row(s)  affected)
```

可以使用大多数标准的数学运算符来操作字段值，如加（+），减（-），乘（*）和除（/）。也可以一次对多个字段进行运算，例如：

```
SELECT  price*ytd_sales "total revenue" FROM titles
```

在这个例子中，通过把价格与销售量相乘，计算出了每种书的总销售额。这个 SELECT **语句的结果将是这样的：**

```
total  revenue
.....
```

Green Apple Data Center

```
81,859,05
46,318,20
55,978,78
81,859,05
40,619,68
…
(18 row(s)  affected)
```

最后，还可以使用链接运算符（它看起来像个加号）来链接两个字符型字段：

```
SELECT  au_fname+" "+au_lname  "author name" FROM authors
```

在这个例子中，把字段 au_fname **和字段** au_lname **粘贴在一起，中间用一个逗号隔开，并把查询结果的标题指定为** author name**。这个语句的执行结果将是这样的：**

```
 author  names
 ·····.
 Johnson White
 Marjorie Green
 Cheryl  Carson
Michael O' Leary
Dean Straight
…
(23 row(s)  affected)
```

可以看到，SQL **提供了对查询结果的许多控制。应该在** PHP **编程过程中充分利用这些优点。使用** SQL **来操作查询结果几乎总是比使用有同样作用的脚本效率更高。**

4. 排序查询结果

本节的介绍中曾强调过，SQL **表没有内在的顺序。例如，从一个表中取第二个记录是没有意义的。从** SQL **的角度看来，没有一个记录在任何其他记录之前。**

然而，可以操纵一个 SQL **查询结果的顺序。在缺省情况下，当记录从表中取出时，记录不以特定的顺序出现。例如，当从表** authors **中取出字段** au_lname **时，查询结果显示成这样：**

```
 au_lname
 ·····
 White
 Green
 Carson
 O' Leary
 Straight
 …
 (23 row(s) affected)
```

Green Apple Data Center

看一列没有特定顺序的名字是很不方便的。如果把这些名字按字母顺序排列，读起来就会容易得多。通过使用 ORDER BY 子句，可以强制一个查询结果按升序排列，就像这样：

```
SELECT au_lname FROM    authors    ORDER    BY    au_lname
```

当这个 SELECT 语句执行时，作者名字的显示将按字母顺序排列。ORDER BY 子句将作者名字按升序排列。

也可以同时对多个列使用 ORDER BY 子句。例如，如果想同时按升序显示字段 au_lname 和字段 au_fname，需要对两个字段都进行排序：

```
SELECT au_lname,au_fname FROM authors ORDER BY au_lname ,au_fname
```

这个查询首先把结果按 au_lname 字段进行排序，然后按字段 au_fname 排序。记录将按如下的顺序输出：

```
au_lname                    au_fname
.....
Bennet                      Abraham
Ringer                      Albert
Ringer                      Anne
Smith                       Meander
…
(23 row(s) affected)
```

注意有两个作者有相同的名字 Ringer。名为 Albert Ringer 的作者出现名为 Anne Ringer 的作者之前，这是因为姓 Albert 按字母顺序应排在姓 Anne 之前。

如果想把查询结果按相反的顺序排列，可以使用关键字 DESC。关键字 DESC 把查询结果按降序排列，如下例所示：

```
SELECT    au_lname,au_fname    FROM authors
        WHERE au_lname=”Ringer” ORDER BY    au_lname ,au_fname    DESC
```

这个查询从表 authors 中取出所有名字为 Ringer 的作者记录。ORDER BY 子句根据作者的名字和姓，将查询结果按降序排列。结果是这样的：

```
au_lname                    au_fname
.....
Ringer                      Anne
Ringer                      Albert
(2 row(s)    affectec)
```

注意在这个表中，姓 Anne 出现在姓 Albert 之前。作者名字按降序显示。

也可以按数值型字段对一个查询结果进行排序。例如，如果想按降序取出所有书的价格，可以使用如下的 SQL 查询：

```
SELECT    price    FROM titles    ORDER BY    price    DESC
```

这个 SELECT 语句从表中取出所有书的价格，显示结果时，价格低的书先显示，价格高的书后显示。

Green Apple Data Center

☞ 注意

不是特别需要时，不要对查询结果进行排序，因为服务器完成这项工作要费些力气。这意味着带有 ORDER BY 子句的 SELECT 语句执行起来比一般的 SELECT 语句花的时间长。

5. 取出互不相同的记录

一个表有可能在同一列中有重复的值。例如，数据库 pubs 的表 authors 中有两个作者的名字是 Ringer。如果从这个表中取出所有的名字，名字 Ringer 将会显示两次。

在特定情况下，可能只有兴趣从一个表中取出互不相同的值。如果一个字段有重复的值，也许希望每个值只被选取一次，可以使用关键字 DISTINCT 来做到这一点：

```
SELCET  DISTINCT au_lname  FROM  authors  WHERE  au_lname="Ringer"
```

当这个 SELECT 语句执行时，只返回一个记录。通过在 SELECT 语句中包含关键字 DISTINCT，可以删除所有重复的值。例如，假设有一个关于新闻组信息发布的表，想取出所有曾在这个新闻组中发布信息的人的名字，那么可以使用关键字 DISTINCT。每个用户的名字只取一次——尽管有的用户发布了不止一篇信息。

☞ 注意

如同 ORDER BY 子句一样，强制服务器返回互不相同的值也会增加运行开销。服务器不得不花费一些时间来完成这项工作。因此，不是必须的时候不要使用关键字 DISTINCT。

6.3.5 建立索引

假设想找到本书中的某一个句子，可以一页一页地逐页搜索，但这会花很多时间。而通过使用本书的索引，可以很快地找到要搜索的主题。表的索引与书后面的索引非常相似，它可以极大地提高查询的速度。对一个较大的表来说，通过添加索引，一个通常要花费几个小时来完成的查询只要几分钟就可以完成，因此有理由对需要频繁查询的表增加索引。

（1）聚簇索引和非聚簇索引

假设通过书的索引找到了一个句子所在的页码。一旦已经知道了页码后，很可能漫无目的翻寻这本书，直至找到正确的页码。通过随机的翻寻，最终可以到达正确的页码。但是，有一种找到页码的更有效的方法。

首先，把书翻到大概一半的地方，如果要找的页码比半本书处的页码小，就把书翻到四分之一处，否则，就把书翻到四分之三的地方。通过这种方法，可以继续把书分成更小的部分，直至找到正确的页码附近，这是找到书页的非常有效的一种方法。

表索引以类似的方式工作。一个表索引由一组页组成，这些页构成了一个树形结构。根页通过指向另外两个页，把一个表的记录从逻辑上分成和两个部分。而根页所指向的两个页又分别把记录分割成更小的部分，每个页都把记录分成更小的分割，直至到达叶级页。

索引有两种类型：聚簇索引和非聚簇索引。在聚簇索引中，索引树的叶级页包含实际的数据：记录的索引顺序与物理顺序相同。在非聚簇索引中，叶级页指向表中的记录：记录的物理顺序与逻辑顺序没有必然的联系。

聚簇索引非常像目录表，目录表的顺序与实际的页码顺序是一致的。非聚簇索引则更像书的标准索引表，索引表中的顺序通常与实际的页码顺序是不一致的。一本书也许有多个索引。例如，它也许同时有主题索引和作者索引。同样，一个表可以有多个非聚簇索引。

通常情况下，使用的是聚簇索引，但是应该对两种类型索引的优缺点都有所理解。

每个表只能有一个聚簇索引，因为一个表中的记录只能以一种物理顺序存放。通常对一个表按照标识字段建立聚簇索引。但是，也可以对其他类型的字段建立聚簇索引，如字符型，数值型和日期时间型字段。

从建立了聚簇索引的表中取出数据要比建立了非聚簇索引的表快。当需要取出一定范围内的数据时，用聚簇索引也比用非聚簇索引好。例如，假设用一个表来记录访问者在网站上的活动。如果想取出在一定时间段内的登录信息，应该对这个表的 DATETIME 型字段建立聚簇索引。

对聚簇索引的主要限制是每个表只能建立一个聚簇索引。但是，一个表可以有不止一个非聚簇索引。实际

Green Apple Data Center

上，对每个表最多可以建立 249 个非聚簇索引。也可以对一个表同时建立聚簇索引和非聚簇索引。

假如不仅想根据日期，而且想根据用户名从网站活动日志中取数据。在这种情况下，同时建立一个聚簇索引和非聚簇索引是有效的。可以对日期时间字段建立聚簇索引，对用户名字段建立非聚簇索引。如果发现需要更多的索引方式，可以增加更多的非聚簇索引。

非聚簇索引需要大量的硬盘空间和内存。另外，虽然非聚簇索引可以提高从表中获取数据的速度，它也会降低向表中插入和更新数据的速度。每当改变一个建立了非聚簇索引的表中的数据时，必须同时更新索引。因此对一个表建立非聚簇索引时要慎重考虑。如果预计一个表需要频繁地更新数据，那么不要对它建立太多非聚簇索引。另外，如果硬盘和内存空间有限，也应该限制使用非聚簇索引的数量。

（2）索引属性

这两种类型的索引都有两个重要属性：可以用两者中任一种类型同时对多个字段建立索引（复合索引）；两种类型的索引都可以指定为唯一索引。

可以对多个字段建立一个复合索引，甚至是复合的聚簇索引。假如有一个表记录了网站访问者的姓和名字。如果希望根据完整姓名从表中取数据，需要建立一个同时对姓字段和名字字段进行的索引。这和分别对两个字段建立单独的索引是不同的。当希望同时对不止一个字段进行查询时，应该建立一个对多个字段的索引。如果希望对各个字段进行分别查询，应该对各字段建立独立的索引。

两种类型的索引都可以被指定为唯一索引。如果对一个字段建立了唯一索引，将不能向这个字段输入重复的值。一个标识字段会自动成为唯一值字段，但也可以对其他类型的字段建立唯一索引。假设用一个表来保存的网站的用户密码，当然不希望两个用户有相同的密码。通过强制一个字段成为唯一值字段，可以防止这种情况的发生。

（3）用 SQL 建立索引

为了给一个表建立索引，输入下面的语句：

```
CREATE INDEX mycolumn_index ON mytable (myclumn)
```

这个语句建立了一个名为 mycolumn_index 的索引。可以给一个索引起任何名字，但应该在索引名中包含索引的字段名，这对将来弄清楚建立该索引的意图是有帮助的。

索引 mycolumn_index 对表 mytable 的 mycolumn 字段进行。这是个非聚簇索引，也是个非唯一索引。（这是一个索引的缺省属性）

如果需要改变一个索引的类型，必须删除原来的索引并重建一个。建立了一个索引后，可以用下面的 SQL 语句删除它：

```
DROP INDEX mytable.mycolumn_index
```

注意在 DROP INDEX 语句中要包含表的名字。在这个例子中，删除的索引是 mycolumn_index，它是表 mytable 的索引。

要建立一个聚簇索引，可以使用关键字 CLUSTERED，记住一个表只能有一个聚簇索引。这里有一个如何对一个表建立聚簇索引的例子：

```
CREATE CLUSTERED INDEX mycolumn_clust_index ON mytable(mycolumn)
```

如果表中有重复的记录，当试图用这个语句建立索引时，会出现错误。但是有重复记录的表也可以建立索引；只要使用关键字 ALLOW_DUP_ROW 即可：

```
CREATE CLUSTERED INDEX mycolumn_cindex ON mytable(mycolumn)
WITH ALLOW_DUP_ROW
```

Green Apple Data Center

这个语句建立了一个允许重复记录的聚簇索引。应该尽量避免在一个表中出现重复记录，但是，如果已经出现了，可以使用这种方法。

要对一个表建立唯一索引，可以使用关键字 UNIQUE。对聚簇索引和非聚簇索引都可以使用这个关键字。这里有一个例子：

```
CREATE UNIQUE COUSTERED INDEX myclumn_cindex ON mytable(mycolumn)
```

这是将经常使用的索引建立语句。无论何时，只要可以，应该尽量对一个表建立唯一聚簇索引来增强查询操作。

最后，要建立一个对多个字段的索引——复合索引——在索引建立语句中同时包含多个字段名。下面的例子对 firstname 和 lastname 两个字段建立索引：

```
CREATE INDEX name_index ON username(firstname,lastname)
```

这个例子对两个字段建立了单个索引。在一个复合索引中，最多可以对 16 个字段进行索引。

6.3.6 SQL 核心语句

在上面几节讲述了如何用 SQL SELECT 语句从一个表中取数据。但是，到现在为止，还没有讨论如何添加，修改或删除表中的数据。在这一节中，将学习这些内容。

（1）插入数据

向表中添加一个新记录，要使用 SQL INSERT 语句。这里有一个如何使用这种语句的例子：

```
INSERT mytable (mycolumn) VALUES (‘ some data’ )
```

这个语句把字符串’some data’插入表 mytable 的 mycolumn 字段中。将要被插入数据的字段的名字在第一个括号中指定，实际的数据在第二个括号中给出。

INSERT 语句的完整句法如下：

```
INSERT [INTO]   {table_name|view_name} [(column_list)] {DEFAULT VALUES |
Values_list | select_statement}
```

如果一个表有多个字段，通过把字段名和字段值用逗号隔开，可以向所有的字段中插入数据。假设表 mytable 有三个字段 first_column,second_column,和 third_column。下面的 INSERT 语句添加了一条三个字段都有值的完整记录：

```
INSERT mytable (first_column,second_column,third_column)
VALUES (‘ some data’ ,’ some more data’ ,’ yet more data’ )
```

如果在 INSERT 语句中只指定两个字段和数据会怎么样呢？换句话说，向一个表中插入一条新记录，但有一个字段没有提供数据。在这种情况下，有下面的四种可能：

如果该字段有一个缺省值，该值会被使用。例如，假设插入新记录时没有给字段 third_column 提供数据，而这个字段有一个缺省值’some value’。在这种情况下，当新记录建立时会插入值’some value’。

如果该字段可以接受空值，而且没有缺省值，则会被插入空值。

如果该字段不能接受空值，而且没有缺省值，就会出现错误,会收到错误信息：

最后，如果该字段是一个标识字段，那么它会自动产生一个新值。当向一个有标识字段的表中插入新记录时，只要忽略该字段，标识字段会给自己赋一个新值。

Green Apple Data Center

向一个有标识字段的表中插入新记录后，可以用 SQL 变量@@identity 来访问新记录的标识字段的值。考虑如下的 SQL 语句：

```
INSERT mytable (first_column) VALUES(‘ some value’ )
INSERT anothertable(another_first,another_second)
VALUES(@@identity,’ some value’ )
```

如果表 mytable 有一个标识字段，该字段的值会被插入表 anothertable 的 another_first 字段。这是因为变量@@identity 总是保存最后一次插入标识字段的值。

字段 another_first 应该与字段 first_column 有相同的数据类型。但是，字段 another_first 不能是标识字段。Another_first 字段用来保存字段 first_column 的值。

（2）删除记录

要从表中删除一个或多个记录，需要使用 SQL DELETE 语句。可以给 DELETE 语句提供 WHERE 子句。WHERE 子句用来选择要删除的记录。例如，下面的这个 DELETE 语句只删除字段 first_column 的值等于’Delete Me’ 的记录：

```
DELETE mytable WHERE first_column=’ Deltet Me’
```

DELETE 语句的完整句法如下：

```
DELETE [FROM] {table_name|view_name}  [WHERE clause]
```

在 SQL SELECT 语句中可以使用的任何条件都可以在 DELECT 语句的 WHERE 子句 中使用。例如，下面的这个 DELETE 语句只删除那些 first_column 字段的值为’ goodbye’ 或 second_column 字段的值为’so long’ 的记录：

```
DELETE mytable WHERE first_column=’ goodby’  OR second_column=’ so long’
```

如果不给 DELETE 语句提供 WHERE 子句，表中的所有记录都将被删除。前面曾讲过 TRUNCATE TABLE 可以删除表中的所有记录，如果想删除该表中的所有记录，应使用 TRUNCATE TABLE 语句。为什么要用 TRUNCATE TABLE 语句代替 DELETE 语句？当使用 TRUNCATE TABLE 语句时，记录的删除是不作记录的。也就是说，这意味着 TRUNCATE TABLE 要比 DELETE 快得多。

（3）更新记录

要修改表中已经存在的一条或多条记录，应使用 SQL UPDATE 语句。同 DELETE 语句一样，UPDATE 语句可以使用 WHERE 子句来选择更新特定的记录。请看这个例子：

```
UPDATE mytable SET first_column=’ Updated!’  WHERE second_column=’ Update Me!’
```

这个 UPDATE 语句更新所有 second_column 字段的值为’ Update Me!’ 的记录。对所有被选中的记录，字段 first_column 的值被置为’ Updated!’ 。

下面是 UPDATE 语句的完整句法：

```
UPDATE {table_name|view_name} SET [{table_name|view_name}]
    {column_list|variable_list|variable_and_column_list}
    [,{column_list2|variable_list2|variable_and_column_list2}...
[,{column_listN|variable_listN|variable_and_column_listN}]]
[WHERE clause]
```

Green Apple Data Center

☞ **注意：**

可以对文本型字段使用 UPDATE 语句。但是，如果需要更新很长的字符串，应使用 UPDATETEXT 语句。这部分内容对本书来说太高级了，因此不加讨论。

如果不提供 WHERE 子句，表中的所有记录都将被更新。有时这是有用的，例如，想把表 titles 中的所有书的价格加倍。

也可以同时更新多个字段。例如，下面的 UPDATE 语句同时更新 first_column，second_column，和 third_column 这三个字段：

```
UPDATE mytable    SET    first_column=’Updated!’
 Second_column=’Updated!’
 Third_column=’Updated!’
 WHERE first_column=’Update Me1’
```

☞ **技巧**

SQL 忽略语句中多余的空格，可以把 SQL 语句写成任何最容易读的格式。

（4）用 SELECT 创建记录和表

读者也许已经注意到，INSERT 语句与 DELETE 语句和 UPDATE 语句有一点不同，它一次只操作一个记录。然而，有一个方法可以使 INSERT 语句一次添加多个记录。要作到这一点，需要把 INSERT 语句与 SELECT 语句结合起来，像这样：

```
INSERT mytable (first_column,second_column)
SELECT another_first,another_second
FROM anothertable
WHERE another_first=’Copy Me!’
```

这个语句从 anothertable 拷贝记录到 mytable.只有表 anothertable 中字段 another_first 的值为’Copy Me！’的记录才被拷贝。

当为一个表中的记录建立备份时，这种形式的 INSERT 语句是非常有用的。在删除一个表中的记录之前，可以先用这种方法把它们拷贝到另一个表中。

如果需要拷贝整个表，可以使用 SELECT INTO 语句。例如，下面的语句创建了一个名为 newtable 的新表，该表包含表 mytable 的所有数据：

```
SELECT * INTO newtable FROM mytable
```

也可以指定只有特定的字段被用来创建这个新表。要做到这一点，只需在字段列表中指定想要拷贝的字段。另外，可以使用 WHERE 子句来限制拷贝到新表中的记录。下面的例子只拷贝字段 second_columnd 的值等于’Copy Me!’的记录的 first_column 字段。

```
SELECT first_column INTO newtable
FROM mytable
WHERE second_column=’Copy Me!’
```

使用 SQL 修改已经建立的表是很困难的。例如，如果向一个表中添加了一个字段，没有容易的办法来去除它。另外，如果不小心把一个字段的数据类型给错了，将没有办法改变它。但是，使用本小节中讲述的 SQL 语句，可以绕过这两个问题。

Green Apple Data Center

例如，假设想从一个表中删除一个字段。使用 SELECT INTO 语句，可以创建该表的一个拷贝，但不包含要删除的字段。这样既删除了该字段，又保留了不想删除的数据。

如果想改变一个字段的数据类型，可以创建一个包含正确数据类型字段的新表。创建好该表后，就可以结合使用 UPDATE 语句和 SELECT 语句，把原来表中的所有数据拷贝到新表中。通过这种方法，既可以修改表的结构，又能保存原有的数据。

6.3.7 集合函数

到现在为止，讲述了如何根据特定的条件从表中取出一条或多条记录。但是，假如想对一个表中的记录进行数据统计。例如，如果想统计存储在表中的一次民意测验的投票结果。或者想知道一个访问者在网站上平均花费了多少时间。要对表中的任何类型的数据进行统计，都需要使用集合函数。

SQL 支持五种类型的集合函数。可以统计记录数目，平均值，最小值，最大值，或者求和。当使用一个集合函数时，它只返回一个数，该数值代表这几个统计值之一。

要在 PHP 网页中使用集合函数的返回值，需要给该值起一个名字，要做到这一点，可以在 SELECT 语句中，在集合函数后面紧跟一个字段名，如下例所示：

SELECT AVG(vote) ‘ the_average’ FROM opinion

在这个例子中，vote 的平均值被命名为 the_average。现在可以在 PHP 网页的数据库方法中使用这个名字。

（1）统计字段值的数目

函数 COUNT（）也许是最有用的集合函数，可以用这个函数来统计一个表中有多少条记录。例如：

```
SELECT COUNT(au_lname) FROM authors
```

这个例子计算表 authors 中名字（last name）的数目。如果相同的名字出现了不只一次，该名字将会被计算多次。如果想知道名字为某个特定值的作者有多少个，可以使用 WHERE 子句，如下例所示：

```
SELECT COUNT(au_lname) FROM authors WHERE au_lname=’ Ringer’
```

这个例子返回名字为’ Ringer’ 的作者的数目。如果这个名字在表 authors 中出现了两次，则此函数的返回值是 2。

假如想知道有不同名字的作者的数目。可以通过使用关键字 DISTINCT 来得到该数目。如下例所示：

```
SELECT COUNT(DISTINCT au_lname) FROM authors
```

如果名字 Ringer’ 出现了不只一次，它将只被计算一次。关键字 DISTINCT 决定了只有互不相同的值才被计算。

通常，当使用 COUNT（）时，字段中的空值将被忽略。一般来说，这正是所希望的。但是，如果仅仅想知道表中记录的数目，那么需要计算表中所有的记录——不管它是否包含空值。下面是一个如何做到这一点的例子：

```
SELECT COUNT(*) FROM authors
```

注意函数 COUNT（）没有指定任何字段。这个语句计算表中所有记录数目，包括有空值的记录。因此，不需要指定要被计算的特定字段。

函数 COUNT（）在很多不同情况下是有用的。例如，假设有一个表保存了对网站的质量进行民意调查的结果。这个表有一个名为 vote 的字段，该字段的值要么是 0，要么是 1。0 表示反对票，1 表示赞成票。要确定赞成票的数量，可以所有下面的 SELECT 语句：

```
SELECT COUNT(vote) FROM opinion_table WHERE vote=1
```

Green Apple Data Center

（2）计算字段的平均值

使用函数 COUNT（），可以统计一个字段中有多少个值。但有时需要计算这些值的平均值。使用函数 AVG（），可以返回一个字段中所有值的平均值。

假如对网站进行一次较为复杂的民意调查。访问者可以在 1 到 10 之间投票，表示他们喜欢该网站的程度。把投票结果保存在名为 vote 的 INT 型字段中，要计算用户投票的平均值，需要使用函数 AVG（）：

```
SELECT AVG(vote) FROM opinion
```

这个 SELECT 语句的返回值代表用户对网站的平均喜欢程度，函数 AVG（）只能对数值型字段使用，这个函数在计算平均值时也忽略空值。

（3）计算字段值的和

假设网站被用来出售卡片，已经运行了两个月，是该计算赚了多少钱的时候了。假设有一个名为 orders 的表用来记录所有访问者的订购信息，要计算所有订购量的总和，可以使用函数 SUM（）：

```
SELECT SUM(purchase_amount) FROM orders
```

函数 SUM（）的返回值代表字段 purchase_amount 中所有值的平均值，字段 purchase_amount 的数据类型也许是 MONEY 型，但也可以对其它数值型字段使用函数 SUM（）。

（4）返回最大值或最小值

假设有一个表用来保存对网站进行民意调查的结果。访问者可以选择从 1 到 10 的值来表示他们对网站的评价。如果想知道访问者对网站的最高评价，可以使用如下的语句：

```
SELECT MAX(vote) FROM opinion
```

也许有人对网站给予了很高的评价。通过函数 MAX（），可以知道一个数值型字段的所有值中的最大值。如果有人对网站投了数字 10，函数 MAX（）将返回该值。

另一方面，假如想知道访问者对网站的的最低评价，可以使用函数 MIN（），如下例所示：

```
SELECT MIN(vote) FROM opinion
```

函数 MIN（）返回一个字段的所有值中的最小值。如果字段是空的，函数 MIN（）返回空值。

（6）通过匹配一定范围的值来取出数据

假设有一个表用来保存对网站进行民意调查的结果。现在想向所有对网站的评价在 7 到 10 之间的访问者发送书面的感谢信。要得到这些人的名字，可以使用如下的 SELECT 语句：

```
SELECT username FROM opinion WHERE vote>6    and    vote<11
```

这个 SELECT 语句会实现这样的要求，使用下面的 SELECT 语句也可以得到同样的结果：

```
SELECT username FROM opinion WHERE vote BETWEEN 7 AND 10
```

这个 SELECT 语句与上一个语句是等价的，使用哪一种语句是编程风格的问题，但使用表达式 BETWEEN 的语句更易读。

现在假设只想取出对网站投了 1 或者 10 的访问者的名字。要从表 opinion 中取出这些名字，可以使用如下的 SELECT 语句：

Green Apple Data Center

```
SELECT username FROM opinion WHERE vote=1 or vote＝10
```

这个 SELECT 语句会返回正确的结果，没有理由不使用它，但是，存在一种等价的方式。使用如下的 SELECT 可以得到相同的结果：

```
SELECT username FROM opinion WHERE vote IN (1,10)
```

注意表达式 IN 的使用。这个 SELECT 语句只取出 vote 的值等于括号中的值之一的记录。

也可以使用 IN 来匹配字符数据。例如，假设只想取出 Bill Gates 或 President Clinton 的投票值。可以使用如下的 SELECT 语句：

```
SELECT vote FROM opinion    WHERE    username IN (‘ Bill Gates’ ,’ President Clinton’ )
```

最后，可以在使用 BETWEEN 或 IN 的同时使用表达式 NOT。例如，要取出那些投票值不在 7 到 10 之间的人的名字，可以使用如下的 SELECT 语句：

```
SELECT username FROM opinion WHERE vote NOT BETWEEN 7 and 10
```

要选取那些某个字段的值不在一列值之中的记录，可以同时使用 NOT 和 IN，如下例所示：

```
SELECT vote FROM opinion
WHERE username NOT IN (‘ Bill Gates’ ,’ President Clinton’ )
```

不是必须在 SQL 语句中使用 BETWEEN 或 IN，但是，要使查询更接近自然语言，这两个表达式是有帮助的。

6.3.8 操作字符串数据

SQL 有许多函数和表达式能对字符串进行有趣的操作，包括各种各样的模式匹配和字符转换。

（1）匹配通配符

假设想建立一个与 Yahoo 功能相似的 Internet 目录。可以建立一个表用来保存一系列的网站名称，统一资源定位器（URL），描述和类别，并允许访问者通过在 HTML form 中输入关键字来检索这些内容。

假如有一个访问者想从这个目录中得到其描述中包含关键字 trading card 的网站的列表。要取出正确的网站列表，也许试图使用这样的查询：

```
SELECT site_name FROM site_directory    WHERE site_desc=’ trading card’
```

这个查询可以工作。但是，它只能返回那些其描述中只有 trading card 这个字符串的网站。例如，一个描述为 We have the greatest collection of trading cards in the world!的网站不会被返回。

要把一个字符串与另一个字符串的一部分相匹配，需要使用通配符。使用通配符和关键字 LIKE 来实现模式匹配。下面的语句使用通配符和关键字 LIKE 重写了上面的查询，以返回所有正确网站的名字：

```
SELECT SITE_name FROM site_directory
 WHERE site_desc LIKE ‘ %trading cark%’
```

在这个例子中，所有其描述中包含表达式 trading card 的网站都被返回。描述为 We have the greatest collection of trading cards in the world!的网站也被返回。当然，如果一个网站的描述中包含 I am trading cardboard boxes online，该网站的名字也被返回。

Green Apple Data Center

注意本例中百分号的使用。百分号是通配符的例子之一。它代表 0 个或多个字符。通过把 trading card 括在百分号中，所有其中嵌有字符串 trading card 的字符串都被匹配。

现在，假设网站目录变得太大而不能在一页中完全显示。决定把目录分成两部分。在第一页，想显示所有首字母在 A 到 M 之间的网站。在第二页，想显示所有首字母在 N 到 Z 之间的网站。要得到第一页的网站列表，可以使用如下的 SQL 语句：

```
SELECT site_name FROM site_directory WHERE site_name LIKE ‘ [A-M]%’
```

在这个例子中使用了表达式[A-M]，只取出那些首字母在 A 到 M 之间的网站。中括号（[]）用来匹配处在指定范围内的单个字符。要得到第二页中显示的网站，应使用这个语句：

```
SELECT site_name FROM    site_directory
 WHERE site_name LIKE ‘ [N-Z]%’
```

在这个例子中，括号中的表达式代表任何处在 N 到 Z 之间的单个字符。

假设网站目录变得更大了，现在需要把目录分成更多页。如果想显示那些以 A，B 或 C 开头的网站，可以用下面的查询来实现：

```
SELECT site_name FROM site_directory WHERE site_name LIKE   ‘ [ABC]%’
```

在这个例子中，括号中的表达式不再指定一个范围，而是给出了一些字符。任何一个其名字以这些字符中的任一个开头的网站都将被返回。

通过在括号内的表达式中同时包含一个范围和一些指定的字符，可以把这两种方法结合起来。例如，用下面的这个查询，可以取出那些首字母在 C 到 F 之间，或者以字母 Y 开头的网站：

```
SELECT site_name FROM site_directory WHERE site_name LIKE ‘ [C-FY]%’
```

在这个例子中，名字为 Collegescape 和 Yahoo 的网站会被选取，而名字为 Magicw3 的网站则不会被选取。

也可以使用脱字符（^）来排除特定的字符。例如，要得到那些名字不以 Y 开头的网站，可以使用如下的查询：

```
SELECT site_name FROM site_directory WHERE site_name LIKE ‘ [^Y]%’
```

对给定的字符或字符范围均可以使用脱字符。

最后，通过使用下划线字符（_），可以匹配任何单个字符。例如，下面这个查询返回每一个其名字的第二个字符为任何字母的网站：

```
SELECT site_name FROM site_directory WHERE site_name LIKE ‘ M_crosoft’
```

这个例子既返回名为 Microsoft 的网站，也返回名为 Macrosoft 的网站。但是，名字为 Moocrosoft 的网站则不被返回。与通配符’%’不同，下划线只代表单个字符。

☞ 注意

如果想匹配百分号或下划线字符本身，需要把它们括在方括号中。如果想匹配连字符(-)，应把它指定为方括号中的第一个字符。如果想匹配方括号，应把它们也括在方括号中。例如，下面的语句返回所有其描述中包含百分号的网站：

Green Apple Data Center

```
SELECT site_name FROM site_directory WHERE site_desc LIKE ‘ %[%]%’
```

（2）匹配发音

SQL 有两个允许按照发音来匹配字符串的函数。函数 SOUNDEX（）给一个字符串分配一个音标码，函数 DIFFERENCE（）按照发音比较两个字符串。当不知道一个名字的确切拼写，但多少知道一点它的发音时，使用这两个函数将有助于取出该记录。

例如，如果建立一个 Internet 目录，也许想增加一个选项，允许访问者按照网站名的发音来搜索网站，而不是按名字的拼写。考虑如下的语句：

```
SELECT site_name FROM site_directory
 WHERE DIFFERENCE(site_name , ‘ Microsoft’ >3
```

这个语句使用函数 DEFFERENCE（）来取得其名字的发音与 Microsoft 非常相似的网站。函数 DIFFERENCE（）返回一个 0 到 4 之间的数字。如果该函数返回 4，表示发音非常相近；如果该函数返回 0，说明这两个字符串的发音相差很大。

例如，上面的语句将返回网站名 Microsoft 和 Macrosoft。这两个名字的发音与 Microsoft 都很相似。如果把上一语句中的大于 3 改为大于 2，那么名为 Zicrosoft 和 Megasoft 的网站也将被返回。最后，如果只需要差别等级大于 1 即可，则名为 Picosoft 和 Minisoft 的网站也将被匹配。

要深入了解函数 DIFFERENCE（）是如何工作的，可以用函数 SOUNDEX（）来返回函数 DIFFERENCE（）所使用的音标码。这里有一个例子：

```
SELECT site_name ‘ site name’ ,SOUNDEX(site_name) ‘ sounds like’
```

这个语句选取字段 site_name 的所有数据及其音标码。下面是这个查询的结果：

```
site name sounds like
.....
Yahoo Y000
Mahoo M000
Microsoft M262
Macrosoft M262
Minisoft    M521
MicroshoftM262
Zicrosoft Z262
Zaposoft    Z121
Millisoft M421
Nanosoft    N521
Megasoft    M221
Picosoft    P221
(12 row(s) affected)
```

如果仔细看一下音标码，会注意到音标码的第一个字母与字段值的第一个字母相同。例如，Yahoo 和 Mahoo 的音标码只有第一个字母不同，还可以发现 Microsoft 和 Macrosoft 的音标码完全相同。

函数 DIFFERENDE（）比较两个字符串的第一个字母和所有的辅音字母。该函数忽略任何元音字母（包括 y），除非一个元音字母是一个字符串的第一个字母。

不幸的是，使用 SOUNDEX（）和 DIFFERENCE()有一个欠缺。WHERE 子句中包含这两个函数的查询执

Green Apple Data Center

行起来效果不好。因此，应该小心使用这两个函数。

（3）删除空格

有两个函数，TTRIM（）和 LTRIM（），可以用来从字符串中剪掉空格。函数 LTRIM（）去除字符串前面的所有空格；函数 RTRIM（）去除一个字符串尾部的所有空格。这里有一个使用函数 RTRIM（）的例子：

```
SELECT RTRIM(site_name) FROM site_directory
```

在这个例子中，如果任何一个网站的名字尾部有多余的空格，多余的空格将从查询结果中删去。

可以嵌套使用这两个函数，把一个字符串前后的空格同时删去：

```
SELECT LTRIM(RTRIM(site_name) FROM site_directory
```

会发现，在从 CHAR 型字段中剪掉多余的空格时，这两个函数非常有用。记住，如果把一个字符串保存在 CHAR 型字段中，该字符串会被追加多余的空格，以匹配该字段的长度。用这两个函数，可以去掉无用的空格，从而解决这个问题。

这一节加深了 SQL 的知识。学会了如何建立索引，使的查询速度更快。还学会了如何插入，删除和更新一个表中的数据，如何使用集合函数得到一个表中数据的统计信息。最后，学会了许多有价值的表达式，函数和过程，用来操作字符串。

6.4 MySQL 数据库常用操作

本节通过演示如何使用 MySQL 客户程序创造和使用一个简单的数据库，提供一个 MySQL 的入门教程。MySQL（有时称为“终端监视器”或只是“监视”）是一个交互式程序，允许链接一个 MySQL 服务器，运行查询并察看结果。MySQL 可以用于批模式：预先把查询放在一个文件中，然后告诉 MySQL 执行文件的内容。使用 MySQL 的两个方法都在这里涉及。

为了看清由 MySQL 提供的一个选择项目表，可用--help 选项调用它：

```
shell> mysql –help
mysql    Ver 11.13 Distrib 3.23.36, for Win95/Win98 (i32)
Copyright (C) 2000 MySQL AB & MySQL Finland AB & TCX DataKonsult AB
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL license

Usage: mysql [OPTIONS] [database]

  -?, --help              Display this help and exit.
  -A, --no-auto-rehash    No automatic rehashing. One has to use 'rehash' to
                          get table and field completion. This gives a quicker
                          start of mysql and disables rehashing on reconnect.
  -B, --batch             Print results with a tab as separator, each row on
                          a new line. Doesn't use history file.
  --character-sets-dir=...
                                  Directory where character sets are located.
  -C, --compressUse compression in server/client protocol.
  -D, --database=..       Database to use.
  --default-character-set=...
                                  Set the default character set.
```

Green Apple Data Center

```
-e, --execute=...       Execute command and quit. (Output like with --batch)
-E, --vertical          Print the output of a query (rows) vertically.
-f, --force             Continue even if we get an sql error.
-g, --no-named-commands
                Named commands are disabled. Use \* form only, or
                        use named commands only in the beginning of a line
                        ending with a semicolon (;) Since version 10.9 the
                        client now starts with this option ENABLED by
                        default! Disable with '-G'. Long format commands
                        still work from the first line.
-G, --enable-named-commands
                        Named commands are enabled. Opposite to -g.
-i, --ignore-space      Ignore space after function names.
-h, --host=...  Connect to host.
-H, --html              Produce HTML output.
-L, --skip-line-numbers
                        Don't write line number for errors.
--no-tee                Disable outfile. See interactive help (\h) also.
-n, --unbuffered        Flush buffer after each query.
-N, --skip-column-names
                        Don't write column names in results.
-O, --set-variable var=option
                Give a variable an value. --help lists variables.
-o, --one-database      Only update the default database. This is useful
                for skipping updates to other database in the update
                log.
-p[password], --password[=...]
                Password to use when connecting to server
                If password is not given it's asked from the tty.
-W, --pipe              Use named pipes to connect to server

-P, --port=...  Port number to use for connection.
-q, --quick             Don't cache result, print it row by row. This may
                slow down the server if the output is suspended.
                Doesn't use history file.
-r, --raw       Write fields without conversion. Used with --batch
-s, --silent            Be more silent.
-S    --socket=...      Socket file to use for connection.
-t, --table             Output in table format.
-T, --debug-info        Print some debug info at exit.
--tee=...               Append everything into outfile. See interactive help
                        (\h) also. Does not work in batch mode.
-u, --user=#            User for login if not current user.
-U, --safe-updates[=#], --i-am-a-dummy[=#]
                Only allow UPDATE and DELETE that uses keys.
-v, --verbose           Write more. (-v -v -v gives the table output format)
-V, --version           Output version information and exit.
```

Green Apple Data Center

```
  -w, --wait          Wait and retry if connection is down.

Default options are read from the following files in the given order:
E:\WINNT\my.ini c:\my.cnf
The following groups are read: mysql client
The following options may be given as the first argument:
--print-defaults   Print the program argument list and exit
--no-defaults          Don't read default options from any options file
--defaults-file=# Only read default options from the given file #
--defaults-extra-file=# Read this file after the global files are read

Possible variables for option --set-variable (-O) are:
connect_timeout          current value: 0
max_allowed_packet       current value: 16777216
net_buffer_length        current value: 16384
select_limit             current value: 1000
max_join_size            current value: 1000000
```

6.4.1 链接与断开服务者

为了链接服务器，当调用 MySQL **时，通常将需要提供一个** MySQL **用户名和一个口令。如果服务器运行在没有登录的一台机器上，也需要指定主机名。联系管理员以找出应该使用什么链接参数进行链接(即那个主机，用户名字和使用的口令)。一旦知道正确的参数，应该能像这样链接：**

```
shell> mysql -h host -u user -p
Enter password: ********
```

********代表的口令；当** mysql **显示** Enter password:**提示时输入它。**
如果能工作，应该看见 mysql>**提示后的一些介绍信息：**

```
shell> mysql -h host -u user -p
Enter password: ********
Welcome to the MySQL monitor.   Commands end with ; or \g.
Your MySQL connection id is 7t o server version: 3.22.36

Type 'help' for help.

mysql>
```

提示符告诉 MySQL **准备输入命令。**
一些 MySQL **安装允许用户以**“anoymous”(**匿名)用户链接在本地主机上运行的服务器。如果本地机器是这种情况，应该能通过没有任何选项地调用** MySQL **与该服务器链接：**

```
shell> mysql
```

在成功地链接后，可以在 mysql>**提示下打入** QUIT **随时断开：**

```
mysql> QUIT
```

Green Apple Data Center

```
Bye
```

也可以键入 control-D 断开。

下列章节的大多数例子都假设链接到服务器。由 mysql>提示指明他们。

6.4.2 输入查询

确保链接上了服务器，首先介绍一点关于如何查询的知识，比立刻跳至讲述创建表、装载数据并且检索数据要来的重要写。本节描述输入命令的基本原则，使用几个查询，能尝试让自己 mysql 是如何工作的。

这是一个简单的命令，要求服务器告诉它的版本号和当前日期。在 mysql>提示打入如下命令并按回车键：

```
mysql> SELECT VERSION(), CURRENT_DATE;
+--------------+--------------+
| version()    | CURRENT_DATE |
+--------------+--------------+
| 3.22.36 | 2001-06-19   |
+--------------+--------------+
1 row in set (0.01 sec)
mysql>
```

这查询说明了关于 mysql 几件事:：

一个命令通常由 SQL 语句组成，随后有一个分号。（有一些例外不需要分号，早先提到的 QUIT 是其中命令之一。）

当发出一个命令时，mysql 发送它给服务器并显示结果，然后打出另外一个 mysql>，显示它准备好接受另外的命令。

mysql 以一张表格(行和列)显示查询输出。第一行包含列的标签，随后的行是询问结果。通常，列标签是取自数据库表的列的名字。如果正在检索一个表达式而非表列的值(如刚才的例子)，mysql 用表达式本身标记列。

mysql 显示多少行被返回，和查询花了多长时间，它给提供服务器性能的一个大致概念。因为他们表示时钟时间(不是 CPU 或机器时间)，并且因为他们受到诸如服务器负载和网络延时的影响，因此这些值是不精确的。（为了简洁，在后面例子中不再显示“集合中的行”。）

关键词可以任何大小写字符输入，下列的查询是等价的：

```
mysql> SELECT VERSION(), CURRENT_DATE;
mysql> select version(), current_date;
mysql> SeLeCt vErSiOn(), current_DATE;
```

这里有另外一个查询，它说明能将 MySQL 用作一个简单的计算器：

```
mysql> SELECT SIN(PI()/4), (4+1)*5;
+-------------+---------+
| SIN(PI()/4) | (4+1)*5 |
+-------------+---------+
|    0.707107 |       25 |
+-------------+---------+
```

上面显示的命令都是单行语句，是相当短的，还可以在单行上输入多条语句，只是以分号结束每一条：

```
mysql> SELECT VERSION(); SELECT NOW();
```

Green Apple Data Center

```
+--------------+
| version()    |
+--------------+
| 3.22.36 |
+--------------+

+---------------------+
| NOW()               |
+---------------------+
| 2001－07－01 20:15:33 |
+---------------------+
```

一个命令不必全在一个单独行给出，所以需要多行的长命令不是问题。MySQL 通过寻找终止的分号而不是寻找输入行的结束来决定的语句在哪儿结束。（换句话说，MySQL 接受自由格式输入：它收集输入行但执行他们直到看见分号。）

这里是一个简单的多行语句的例子：

```
mysql> SELECT
    -> USER()
    -> ,
    -> CURRENT_DATE;
+--------------------+--------------+
| USER()             | CURRENT_DATE |
+--------------------+--------------+
| joesmith@localhost | 2001-07-01   |
+--------------------+--------------+
```

在这个例子中，在输入一个多行查询的第一行后，要注意提示符如何从 mysql>变为->，这正是 MySQL 如何指出它没见到完整的语句并且正在等待剩余的部分。提示符是朋友，因为它提供有价值的反馈，如果使用该反馈，将总是知道 MySQL 正在等待什么。

如果决定不想执行在输入过程中输入的一个命令，打入\c 取消它：

```
mysql> SELECT
    -> USER()
    -> \c
mysql>
```

这里也要注意提示符，在打入\c 以后，它切换回到 mysql>，提供反馈以表明 MySQL 准备接受一个新命令。表 6-1 显示出可以看见的各个提示符意味着 MySQL 在什么状态下。

表 6-1　提示符及其含义

提示符	意　义
mysql>	准备好接受新命令
->	等待多行命令的下一行
'>	等待下一行，收集以单引号（“'”）开始的字符串
">	等待下一行，收集以双引号（“"”）开始的字符串

当打算在单行上发出一个命令时，多行语句通常“偶然”出现，但是忘记终止的分号。在这种情况中，MySQL 等待进一步输入：

Green Apple Data Center

```
mysql> SELECT USER()
    ->
```

如果输完了语句但是唯一的反应是一个->提示符，很可能 MySQL 正在等待分号，只要键入一个分号完成语句，MySQL 就可以执行它：

```
mysql> SELECT USER()
    -> ;
+--------------------+
| USER()             |
+--------------------+
| joesmith@localhost |
+--------------------+
```

'>和">提示符出现在字符串收集期间。在 MySQL 中，可以写由“ ' ”或“ " ”字符括起来的字符串（例如，'hello'或"goodbye"），并且 MySQL 让进入跨越多行的字符串。当看到一个'>或">提示符时，这意味着已经输入了包含以“ ' ”或“ " ”括号字符开始的字符串的一行，但是还没有输入终止字符串的匹配引号。'>和">提示符经常显示由于粗心省掉了一个引号字符。例如：

```
mysql> SELECT * FROM my_table WHERE name = "Smith AND age < 30;
    ">
```

如果输入该 SELECT 语句，然后按回车键并等待结果，什么都没有出现。不要惊讶，“ 为什么该查询这么长呢？ ”，注意">提示符提供的线索。它告诉 MySQL 期望见到一个未终止字符串的余下部分。（在语句中看见错误吗？字符串"Smith 正好丢失第二个引号。）

到这一步，该做什么？最简单的是取消命令。然而，在这种情况下，不能只是打入\c，因为 MySQL 作为它正在收集的字符串的一部分来解释它。相反，输入关闭的引号字符(这样 MySQL 知道完成了字符串)，然后打入\c：

```
mysql> SELECT * FROM my_table WHERE name = "Smith AND age < 30;
    "> "\c
mysql>
```

提示符回到 mysql>，显示 MySQL 准备好接受一个新命令了。

知道'>和">提示符意味着什么是很重要的，因为如果错误地输入一个未终止的字符串，任何下一步输入的行好像将要被 MySQL 忽略（包括包含 QUIT 命令的行）。这可能相当含糊，特别是在能取消当前命令前，如果不知道则需要提出终止引号。

6.4.3 常用查询的例子

下面是一些学习如何用 MySQL 解决一些常见问题的例子，如果读者理解了上一节 SQL 语言的一些基础内容，相信这些例子是很容易理解的。

例如一个例子使用数据库表“shop”，包含某个商人的物品号的价格。假定每个商人的每种物品有一个单独的固定价格，那么(物品，商人)是记录的主键。

能这样创建例子数据库表：

```
USE DATABASE；
```

Green Apple Data Center

```
CREATE TABLE shop (
  article INT(4) UNSIGNED ZEROFILL DEFAULT '0000' NOT NULL,
  dealer    CHAR(20)                  DEFAULT ''        NOT NULL,
  price     DOUBLE(16,2)              DEFAULT '0.00' NOT NULL,
  PRIMARY KEY(article, dealer));

INSERT INTO shop VALUES
(1,'A',3.45),(1,'B',3.99),(2,'A',10.99),(3,'B',1.45),(3,'C',1.69),
(3,'D',1.25),(4,'D',19.95);
```

上面先选择数据库，然后在这个数据库里创建一个叫 shop **的数据表，并插入一些值，要查询这个表里的数据记录，用如下命令：**

```
SELECT * FROM shop

+---------+--------+-------+
| article | dealer | price |
+---------+--------+-------+
|    0001 | A      |  3.45 |
|    0001 | B      |  3.99 |
|    0002 | A      | 10.99 |
|    0003 | B      |  1.45 |
|    0003 | C      |  1.69 |
|    0003 | D      |  1.25 |
|    0004 | D      | 19.95 |
+---------+--------+-------+
```

（1）列的最大值

如果要选择最大的物品号是什么，上一节讲过用 MAX **函数就可以了，命令如下：**

```
SELECT MAX(article) AS article FROM shop

+---------+
| article |
+---------+
|       4 |
+---------+
```

（2）拥有某个列的最大值的行

如果要找出最贵的物品的编号、商人和价格，在 ANSI-SQL **中很容易用一个子查询做到：**

```
SELECT article, dealer, price
FROM    shop
WHERE   price=(SELECT MAX(price) FROM shop)
```

在 MySQL **中（还没有子查询），就必须用** 2 **步做到：**

第 1 **步是用一个** SELECT **语句从表中得到** price **最大值：**

Green Apple Data Center

```
SELECT MAX(price) FROM shop
+--------------------+
| MAX(price)         |
+--------------------+
|              19.95 |
+--------------------+
```

第 2 步使用该值得出实际的查询：

```
SELECT article, dealer, price
FROM    shop
WHERE   price=19.95
```

另一个解决方案是按价格降序排序所有行并用 MySQL 特定 LIMIT 子句只得到的第一行：

```
SELECT article, dealer, price
FROM    shop
ORDER BY price DESC
LIMIT 1
```

注意:如果有多个最贵的物品(例如每个 19.95)，LIMIT 方案就不能解决这个问题了，因为这只能显示之一！如果要按组排序显示每种物品的最高的价格，则要用到 GROUP BY 语句就可以了。

```
SELECT article, MAX(price) AS price
FROM    shop
GROUP BY article
```

```
+---------+-------+
| article | price |
+---------+-------+
|    0001 |  3.99 |
|    0002 | 10.99 |
|    0003 |  1.69 |
|    0004 | 19.95 |
+---------+-------+
```

（3）拥有某个字段的组间最大值的行

如果要选出对每种物品最贵的价格的交易者，在 ANSI SQL 语言中，可以用这样一个子查询做到：

```
SELECT article, dealer, price
FROM    shop s1
WHERE   price=(SELECT MAX(s2.price)
                FROM shop s2
                WHERE s1.article = s2.article)
```

Green Apple Data Center

在 MySQL 中，最好是分几步做到：
第 1 步创建一个临时表。
第 2 步选出物品的最贵价格并插到临时表中。
第 3 步对两个表查询每种物品，得到对应于存储最大价格的行。
这可以很容易用一个临时表做到：

```
CREATE TEMPORARY TABLE tmp (
        article INT(4) UNSIGNED ZEROFILL DEFAULT '0000' NOT NULL,
        price   DOUBLE(16,2)             DEFAULT '0.00' NOT NULL);

LOCK TABLES article read;
```

把查询出来的最贵价格的物品插入到临时表中。

```
INSERT INTO tmp SELECT article, MAX(price) FROM shop GROUP BY article;
```

在 shop 表和临时表中查询实现要得到的结构

```
SELECT article, dealer, price FROM shop, tmp
WHERE shop.article=tmp.articel AND shop.price=tmp.price;

UNLOCK TABLES;

DROP TABLE tmp;
```

如果不使用 TEMPORARY 表，也必须锁定“tmp”表。同样能够通过一个单个查询实现目的，但是只有使用“MAX-CONCAT”才能实现：

```
SELECT article,
        SUBSTRING( MAX( CONCAT(LPAD(price,6,'0'),dealer) ), 7) AS dealer,
  0.00+LEFT(        MAX( CONCAT(LPAD(price,6,'0'),dealer) ), 6) AS price
FROM    shop
GROUP BY article;

+---------+--------+-------+
| article | dealer | price |
+---------+--------+-------+
|    0001 | B      |  3.99 |
|    0002 | A      | 10.99 |
|    0003 | C      |  1.69 |
|    0004 | D      | 19.95 |
+---------+--------+-------+
```

（5） 使用外键

MySQL **唯一不做的事情是** CHECK**，以保证使用的键确实在正在引用表中存在，并且它不自动从有一个外键定义的表中删除行。**

Green Apple Data Center

```
CREATE TABLE persons (
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
    name CHAR(60) NOT NULL,
    PRIMARY KEY (id)
);

CREATE TABLE shirts (
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
    style ENUM('t-shirt', 'polo', 'dress') NOT NULL,
    color ENUM('red', 'blue', 'orange', 'white', 'black') NOT NULL,
    owner SMALLINT UNSIGNED NOT NULL REFERENCES persons,
    PRIMARY KEY (id)
);

INSERT INTO persons VALUES (NULL, 'Antonio Paz');

INSERT INTO shirts VALUES
(NULL, 'polo', 'blue', LAST_INSERT_ID()),
(NULL, 'dress', 'white', LAST_INSERT_ID()),
(NULL, 't-shirt', 'blue', LAST_INSERT_ID());

INSERT INTO persons VALUES (NULL, 'Lilliana Angelovska');

INSERT INTO shirts VALUES
(NULL, 'dress', 'orange', LAST_INSERT_ID()),
(NULL, 'polo', 'red', LAST_INSERT_ID()),
(NULL, 'dress', 'blue', LAST_INSERT_ID()),
(NULL, 't-shirt', 'white', LAST_INSERT_ID());

SELECT * FROM persons;
+----+--------------------+
| id | name               |
+----+--------------------+
|  1 | Antonio Paz        |
|  2 | Lilliana Angelovska |
+----+--------------------+

SELECT * FROM shirts;
+----+---------+--------+-------+
| id | style   | color  | owner |
+----+---------+--------+-------+
|  1 | polo    | blue   |     1 |
|  2 | dress   | white  |     1 |
|  3 | t-shirt | blue   |     1 |
|  4 | dress   | orange |     2 |
|  5 | polo    | red    |     2 |
|  6 | dress   | blue   |     2 |
```

Green Apple Data Center

```
|    7 | t-shirt | white    |         2 |
+----+---------+--------+-------+

SELECT s.* FROM persons p, shirts s
  WHERE p.name LIKE 'Lilliana%'
     AND s.owner = p.id
     AND s.color &lt;&gt; 'white';

+----+-------+--------+-------+
| id | style | color    | owner |
+----+-------+--------+-------+
|    4 | dress | orange |         2 |
|    5 | polo    | red       |         2 |
|    6 | dress | blue      |         2 |
+----+-------+--------+-------+
```

6.4.4 创建并使用一个数据库

创建数据库是最基本的操作，因为一切其他关于表的操作（查询、插入数据等）都是基于数据库的。本节将讲述：怎样创建一个数据库、怎样创建一个数据库表、怎样装载数据到数据库表、怎样以各种方法从表中检索数据、怎样使用多个表。

首先要知道服务器上已经有哪些数据库，使用 SHOW 语句找出在服务器上当前存在什么数据库：

```
mysql> SHOW DATABASES;
+----------+
| Database |
+----------+
| mysql       |
| test         |
| tmp          |
+----------+
```

数据库的列表可能在机器上是不同的，但是 mysql 和 test 数据库很可能在其间。因为它们是 MySQL 必需的，它们描述用户存取权限，test 数据库经常作为一个工作区提供给新用户试试身手。

如果 test 数据库存在，尝试存取它：

```
mysql> USE test
Database changed
```

注意，USE，类似 QUIT，不需要分号（如果喜欢，可以用分号终止这样的语句）。USE 语句在使用上也有另外一个特殊的地方：它必须在一个单行上给出。

可在后面的操作中使用 test 数据库(如果能访问它)，但是在该数据库创建的任何东西可以被访问它的其他人删除，为了这个原因，可能应该询问 MySQL 的管理员许可自己使用的一个数据库。假定想要调用的 menagerie，管理员需要执行一个这样的命令：

```
mysql> GRANT ALL ON menagerie.* TO your_mysql_name;
```

Green Apple Data Center

这里 your_mysql_name 是分配给的 MySQL 用户名。

（1）创建并选用一个数据库

如果在设置的权限时，管理员创建了数据库，可以开始使用它。否则，需要自己创建它：

```
mysql> CREATE DATABASE menagerie;
```

在 Unix 下，数据库名字是区分大小写的，因此必须总是以 menagerie 引用数据库，不是 Menagerie、MENAGERIE 或一些其他名称。对表名也是区分大小写的。（在 Windows 下，该限制不适用，尽管必须在一个给定的查询中使用同样的大小写来引用数据库和表。）

创建了一个数据库必须选定了以后才可以使用它。为了使 menagerie 称为当前的数据库，使用 USE 这个命令：

```
mysql> USE menagerie
Database changed
```

数据库只需要创建一次，但是必须在每次启动一个 MySQL 会话时选择它，可以发出 USE 命令做到。另外，当调用时 MySQL，可在命令行上选择数据库，就在可能需要提供的任何链接参数之后指定其名字。例如：

```
shell> mysql -h host -u user -p menagerie
Enter password: ********
```

注意，menagerie 不是刚才所示命令的口令。如果想要在命令行上在-p 选项后提供的口令，必须做到没有多余的空格(例如，是-pmypassword，而不是-p mypassword)。然而，不建议把口令放在命令行上，因为这样做把它暴露出来，能被在的机器上登录的其他用户窥探到。

（2）创建一个数据库表

创建数据库是容易的部分，但是在这时它是空的，正如 SHOW TABLES 将告诉：

```
mysql> SHOW TABLES;
Empty set (0.00 sec)
```

较难的部分是决定数据库的结构应该是什么：将需要什么数据库表，和在他们中有什么样的列。如果需要一个包含每个宠物的记录的表。它可称为 pet 表，并且它应该包含至少每个动物的名字。因为名字本身不是很有趣，表应该包含另外的信息。例如，如果在豢养宠物的家庭有超过一个人，可能想要列出每个动物的主人，可能也想要记录如种类和性别的一些基本的描述信息。

年龄呢？那可能有趣，但是在一个数据库中存储不是一件好事情。年龄随着时间流逝而变化，这意味着将要不断地更新记录。相反,存储一个固定值例如生日比较好，那么，无论何时需要年龄，可以以当前日期和出生日期之间的差别来计算它。MySQL 为日期运算提供了函数，因此这并不困难。存储出生日期而非年龄也有其他优点：

可以将数据库用于这样的任务，如生日即将到来的宠物生日的提示。

可以相对于日期而不只是当前日期来计算年龄。例如，如果在数据库存储死亡日期，能容易计算一只宠物是何时死的。

可能想到 pet 表中有用的其他类型信息，但是到目前为止这些现在是足够了：名字、主人、种类，性别、出生和死亡日期。

使用一个 CREATE TABLE 语句指定的数据库表的布局：

```
mysql> CREATE TABLE pet (name VARCHAR(20), owner VARCHAR(20),
    -> species VARCHAR(20), sex CHAR(1), birth DATE, death DATE);
```

Green Apple Data Center

VARCHAR 对 name、owner 和 species 列是个好的选择，因为列值是会变长的。这些列的长度都不必是相同的，而且不必是 20。可以挑选从 1 到 255 的任何长度，只要合理就行了。（如果做了较差的选择，以后会变得需要一个更长的字段，MySQL 提供一个 ALTER TABLE 语句。）

动物性别表可以用许多方法表示，例如，"m"和"f"，或许"male"和"female"，使用单个字符"m"和"f"是最简单的。

为 birth 和 death 列使用 DATE 数据类型是相当明显的选择。

既然创建了一个表，SHOW TABLES 应该产生一些输出：

```
mysql> SHOW TABLES;
+--------------------+
| Tables in menagerie |
+--------------------+
| pet                |
+--------------------+
```

为了验证刚才建立表是按期望的方式被创建，使用一个 DESCRIBE 语句：

```
mysql> DESCRIBE pet;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | YES  |     | NULL    |       |
| owner   | varchar(20) | YES  |     | NULL    |       |
| species | varchar(20) | YES  |     | NULL    |       |
| sex     | char(1)     | YES  |     | NULL    |       |
| birth   | date        | YES  |     | NULL    |       |
| death   | date        | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
```

如果忘记在表中的列的名字或他们是什么类型，可以随时 DESCRIBE 一个表结构。

（3）将数据装入一个数据库表

在创建表后，需要在里面装载数据他，LOAD DATA 和 INSERT 语句都是用于此。

假定的宠物记录描述如表 6-2 所示（观察到 MySQL 期望日期时以 YYYY-MM-DD 格式；这可能与习惯的不同）。

表 6-2 宠物记录描述

name	owner	species	sex	birth	death
Fluffy	Harold	cat	f	1993-02-04	
Claws	Gwen	cat	m	1994-03-17	
Buffy	Harold	dog	f	1989-05-13	
Fang	Benny	dog	m	1990-08-27	
Bowser	Diane	dog	m	1998-08-31	2001-01-29
Chirpy	Gwen	bird	f	1998-09-11	
Whistler	Gwen	bird		1997-12-09	
Slim	Benny	snake	m	1996-04-29	

因为是从一张空表开始的，装载它的一个容易方法是创建包含每种动物的每一行为一个文本文件，然后用一个单个语句装载文件的内容到表中。

可以创建一个文本文件“ pet.txt ”，每行包含一个记录，用定位符（tab）把值分开，并且以在 CREATE TABLE 语句中列出的列次序给出。对于丢失的值（例如未知的性别，或仍然活着的动物的死亡日期），可以使用 NULL 值。为了在的文本文件表示这些，使用\N。

Green Apple Data Center

为了装载文本文件“ pet.txt ”到 pet 表中，使用这个命令：

```
mysql> LOAD DATA LOCAL INFILE "pet.txt" INTO TABLE pet;
```

如果愿意，能明确地在 LOAD DATA 语句中指出列值的分隔符和行尾标记，但是缺省值是定位符和换行符，这些对读入文件“pet.txt”的语句是足够的。

当想要一次增加一个新记录时，INSERT 语句是有用的。用它最简单的形式，为每一列提供值，按照列在 CREATE TABLE 语句被列出的顺序。假定 Diane 把一只新仓鼠命名为 Puffball，可以使用一个这样 INSERT 语句增加一条新记录：

```
mysql> INSERT INTO pet
    -> VALUES ('Puffball','Diane','hamster','f','1999-03-30',NULL);
```

注意，这里字符串和日期值被指定为引号扩起来的字符串。另外，用 INSERT 能直接插入 NULL 代表不存在的值。不能使用\N，就像用 LOAD DATA 做的那样。

从这个例子，应该能看到用多个 INSERT 语句而非单个 LOAD DATA 语句装载初始记录。

（4） 从一个数据库表检索信息

SELECT 语句被用来从一张桌子拉出信息。语句的一般格式是：

```
SELECT what_to_select
FROM which_table
WHERE conditions_to_satisfy
```

what_to_select 指出想要看到的，这可以是列的一张表，或*表明“ 所有的列 ”，which_table 指出想要从其检索数据的表，WHERE 子句是可选的，如果它存在，conditions_to_satisfy 指定行是必须满足的检索条件。

① 选择所有数据

SELECT 最简单的形式是从一张表中检索全部记录：

```
mysql> SELECT * FROM pet;
+----------+--------+---------+------+------------+------------+
| name     | owner  | species | sex  | birth      | death      |
+----------+--------+---------+------+------------+------------+
| Fluffy   | Harold | cat     | f    | 1993-02-04 | NULL       |
| Claws    | Gwen   | cat     | m    | 1994-03-17 | NULL       |
| Buffy    | Harold | dog     | f    | 1989-05-13 | NULL       |
| Fang     | Benny  | dog     | m    | 1990-08-27 | NULL       |
| Bowser   | Diane  | dog     | m    | 1998-08-31 | 2001-01-29 |
| Chirpy   | Gwen   | bird    | f    | 1998-09-11 | NULL       |
| Whistler | Gwen   | bird    | NULL | 1997-12-09 | NULL       |
| Slim     | Benny  | snake   | m    | 1996-04-29 | NULL       |
| Puffball | Diane  | hamster | f    | 1999-03-30 | NULL       |
+----------+--------+---------+------+------------+------------+
```

如果想要考察整个表，这种形式的 SELECT 是很有用的。例如，在刚刚给它装载了初始数据以后，如果刚才显示的输出揭示了在的数据文件的一个错误，如发现 Bowser 正确的出生年是 1989，而不是 1998，则有下面修正它的方法：

Green Apple Data Center

编辑文件“ pet.txt ”改正错误，然后使用 DELETE 和 LOAD DATA 弄空表并且再次装载它:

```
mysql> DELETE FROM pet;
mysql> LOAD DATA LOCAL INFILE "pet.txt" INTO TABLE pet;
```

然而，如果这样做，必须重新输入 Puffball 记录。
用一个 UPDATE 语句仅修正错误记录：

```
mysql> UPDATE pet SET birth = "1989-08-31" WHERE name = "Bowser";
```

如上所示，检索整个表是容易的，但是一般不想那样做，特别是当表变得很大时，相反，在这种情况下为想要的信息指定一些限制。

② 选择特定行

在表中只选择特定的行。例如，如果想要验证对 Bowser 的出生日期所做的改变，像这样精选 Bowser 的记录：

```
mysql> SELECT * FROM pet WHERE name = "Bowser";
+--------+-------+---------+------+------------+------------+
| name   | owner | species | sex  | birth      | death      |
+--------+-------+---------+------+------------+------------+
| Bowser | Diane | dog     | m    | 1989-08-31 | 1995-07-29 |
+--------+-------+---------+------+------------+------------+
```

输出正确年份，现在正确记录为 1989，而不是 1998。
字符串比较通常是大小写无关的，因此可以指定名字为"bowser"、"BOWSER"等等，查询结果将是相同的。
可以在任何列上指定条件，不只是 name。例如，如果想要知道哪个动物在 1998 以后出生的，测试 birth 列：

```
mysql> SELECT * FROM pet WHERE birth >= "1998-1-1";
+----------+-------+---------+------+------------+-------+
| name     | owner | species | sex  | birth      | death |
+----------+-------+---------+------+------------+-------+
| Chirpy   | Gwen  | bird    | f    | 1998-09-11 | NULL  |
| Puffball | Diane | hamster | f    | 1999-03-30 | NULL  |
+----------+-------+---------+------+------------+-------+
```

要组合条件，例如，找出雌性的狗：

```
mysql> SELECT * FROM pet WHERE species = "dog" AND sex = "f";
+-------+--------+---------+------+------------+-------+
| name  | owner  | species | sex  | birth      | death |
+-------+--------+---------+------+------------+-------+
| Buffy | Harold | dog     | f    | 1989-05-13 | NULL  |
+-------+--------+---------+------+------------+-------+
```

上面的查询使用 AND 逻辑操作符，也有一个 OR 操作符：

```
mysql> SELECT * FROM pet WHERE species = "snake" OR species = "bird";
+----------+-------+---------+------+------------+-------+
| name     | owner | species | sex  | birth      | death |
```

Green Apple Data Center

```
+----------+-------+---------+------+------------+-------+
| Chirpy   | Gwen  | bird    | f    | 1998-09-11 | NULL  |
| Whistler | Gwen  | bird    | NULL | 1997-12-09 | NULL  |
| Slim     | Benny | snake   | m    | 1996-04-29 | NULL  |
+----------+-------+---------+------+------------+-------+
```

AND 和 OR 可以混用。如果这样做，使用括号指明条件应该如何被分组是一个好主意：

```
mysql> SELECT * FROM pet WHERE (species = "cat" AND sex = "m")
    -> OR (species = "dog" AND sex = "f");
+-------+--------+---------+------+------------+-------+
| name  | owner  | species | sex  | birth      | death |
+-------+--------+---------+------+------------+-------+
| Claws | Gwen   | cat     | m    | 1994-03-17 | NULL  |
| Buffy | Harold | dog     | f    | 1989-05-13 | NULL  |
+-------+--------+---------+------+------------+-------+
```

③ 选择特定列

如果不想看到表的整个行，就命名感兴趣的列，用逗号分开。例如，如果想要知道动物什么时候出生的，只选 name 和 birth 列：

```
mysql> SELECT name, birth FROM pet;
+----------+------------+
| name     | birth      |
+----------+------------+
| Fluffy   | 1993-02-04 |
| Claws    | 1994-03-17 |
| Buffy    | 1989-05-13 |
| Fang     | 1990-08-27 |
| Bowser   | 1989-08-31 |
| Chirpy   | 1998-09-11 |
| Whistler | 1997-12-09 |
| Slim     | 1996-04-29 |
| Puffball | 1999-03-30 |
+----------+------------+
```

找出谁拥有宠物，使用这个查询：

```
mysql> SELECT owner FROM pet;
+--------+
| owner  |
+--------+
| Harold |
| Gwen   |
| Harold |
| Benny  |
| Diane  |
```

Green Apple Data Center

```
| Gwen    |
| Gwen    |
| Benny   |
| Diane   |
+--------+
```

然而，注意到查询简单地检索每个记录的 owner 字段，并且他们中的一些出现多次。为了使输出减到最少，通过增加关键词 DISTINCT 检索出每个唯一的输出记录：

```
mysql> SELECT DISTINCT owner FROM pet;
+--------+
| owner  |
+--------+
| Benny  |
| Diane  |
| Gwen   |
| Harold |
+--------+
```

能使用一个 WHERE 子句把行选择与列选择相结合。例如，为了只得到狗和猫的出生日期，使用这个查询：

```
mysql> SELECT name, species, birth FROM pet
    -> WHERE species = "dog" OR species = "cat";
+--------+---------+------------+
| name   | species | birth      |
+--------+---------+------------+
| Fluffy | cat     | 1993-02-04 |
| Claws  | cat     | 1994-03-17 |
| Buffy  | dog     | 1989-05-13 |
| Fang   | dog     | 1990-08-27 |
| Bowser | dog     | 1989-08-31 |
+--------+---------+------------+
```

④ 排序行

可能已经注意到前面的例子中结果行没有以特定的次序被显示。然而，当行以某个有意义的方式排序，检验查询输出通常是更容易的。为了排序结果，使用一个 ORDER BY 子句。

这里是动物生日，按日期排序：

```
mysql> SELECT name, birth FROM pet ORDER BY birth;
+----------+------------+
| name     | birth      |
+----------+------------+
| Buffy    | 1989-05-13 |
| Bowser   | 1989-08-31 |
| Fang     | 1990-08-27 |
| Fluffy   | 1993-02-04 |
| Claws    | 1994-03-17 |
```

Green Apple Data Center

```
| Slim     | 1996-04-29 |
| Whistler | 1997-12-09 |
| Chirpy   | 1998-09-11 |
| Puffball | 1999-03-30 |
+----------+------------+
```

为了以逆序排序，增加 DESC（下降）**关键字到正在排序的列名上：**

```
mysql> SELECT name, birth FROM pet ORDER BY birth DESC;
+----------+------------+
| name     | birth      |
+----------+------------+
| Puffball | 1999-03-30 |
| Chirpy   | 1998-09-11 |
| Whistler | 1997-12-09 |
| Slim     | 1996-04-29 |
| Claws    | 1994-03-17 |
| Fluffy   | 1993-02-04 |
| Fang     | 1990-08-27 |
| Bowser   | 1989-08-31 |
| Buffy    | 1989-05-13 |
+----------+------------+
```

能在多个列上排序。例如，按动物的种类排序，然后按生日，首先是动物种类中最年轻的动物，使用下列查询：

```
mysql> SELECT name, species, birth FROM pet ORDER BY species, birth DESC;
+----------+---------+------------+
| name     | species | birth      |
+----------+---------+------------+
| Chirpy   | bird    | 1998-09-11 |
| Whistler | bird    | 1997-12-09 |
| Claws    | cat     | 1994-03-17 |
| Fluffy   | cat     | 1993-02-04 |
| Fang     | dog     | 1990-08-27 |
| Bowser   | dog     | 1989-08-31 |
| Buffy    | dog     | 1989-05-13 |
| Puffball | hamster | 1999-03-30 |
| Slim     | snake   | 1996-04-29 |
+----------+---------+------------+
```

注意 DESC **关键词仅适用于紧跟在它之前的列名字**(birth)；species **值仍然以升序被排序。**

⑤ 日期计算

MySQL **提供几个函数，能用来执行在日期上的计算，例如，计算年龄或提取日期的部分。**

为了决定每个宠物有多大，用出生日期和当前日期之间的差别计算年龄。通过变换 2 **个日期到天数，取差值，并且用** 365 **除(在一年里的天数)：**

```
mysql> SELECT name, (TO_DAYS(NOW())-TO_DAYS(birth))/365 FROM pet;
```

Green Apple Data Center

```
+----------+-------------------------------------+
| name         | (TO_DAYS(NOW())-TO_DAYS(birth))/365 |
+----------+-------------------------------------+
| Fluffy     |                                6.15 |
| Claws       |                                5.04 |
| Buffy       |                                9.88 |
| Fang         |                                8.59 |
| Bowser     |                                9.58 |
| Chirpy     |                                0.55 |
| Whistler |                               1.30 |
| Slim         |                                2.92 |
| Puffball |                               0.00 |
+----------+-------------------------------------+
```

尽管查询可行，关于它还有能被改进的一些事情。首先，如果行以某个次序表示，其结果能更容易被扫描。第二，年龄列的标题不是很有意义的。

第一个问题通过增加一个 ORDER BY name **子句按名字排序输出来解决。为了处理列标题，为列提供一个名字以便一个不同的标签出现在输出中(这被称为一个列别名)：**

```
mysql> SELECT name, (TO_DAYS(NOW())-TO_DAYS(birth))/365 AS age
    -> FROM pet ORDER BY name;
+----------+------+
| name         | age    |
+----------+------+
| Bowser     | 9.58 |
| Buffy       | 9.88 |
| Chirpy     | 0.55 |
| Claws       | 5.04 |
| Fang         | 8.59 |
| Fluffy     | 6.15 |
| Puffball | 0.00 |
| Slim         | 2.92 |
| Whistler | 1.30 |
+----------+------+
```

为了按 age **而非** name **排序输出，只要使用一个不同** ORDER BY **子句：**

```
mysql>    SELECT name, (TO_DAYS(NOW())-TO_DAYS(birth))/365 AS age
    ->    FROM pet ORDER BY age;
+----------+------+
| name         | age    |
+----------+------+
| Puffball | 0.00 |
| Chirpy     | 0.55 |
| Whistler | 1.30 |
| Slim         | 2.92 |
| Claws       | 5.04 |
```

Green Apple Data Center

```
| Fluffy    | 6.15 |
| Fang      | 8.59 |
| Bowser    | 9.58 |
| Buffy     | 9.88 |
+----------+------+
```

一个类似的查询可以被用来确定已经死亡动物的死亡年龄。通过检查 death 值是否是 NULL 来决定是哪些动物，然后，对于那些有非 NULL 值的动物，计算在 death 和 birth 值之间的差别：

```
mysql>  SELECT name, birth, death, (TO_DAYS(death)-TO_DAYS(birth))/365 AS age
    ->  FROM pet WHERE death IS NOT NULL ORDER BY age;
+--------+------------+------------+------+
| name   | birth      | death      | age  |
+--------+------------+------------+------+
| Bowser | 1989-08-31 | 1995-07-29 | 5.91 |
+--------+------------+------------+------+
```

查询使用 death IS NOT NULL 而非 death != NULL，因为 NULL 是特殊的值。

如果想要知道哪个动物下个月过生日，怎么办？对于这类计算，年和天是无关的，则要提取 birth 列的月份部分。MySQL 提供几个日期部分的提取函数，例如 YEAR()、MONTH()和 DAYOFMONTH()，在这里 MONTH()是适合的函数。为了看它怎样工作，运行一个简单的查询，显示 birth 和 MONTH(birth)的值：

```
mysql> SELECT name, birth, MONTH(birth) FROM pet;
+----------+------------+--------------+
| name     | birth      | MONTH(birth) |
+----------+------------+--------------+
| Fluffy   | 1993-02-04 |            2 |
| Claws    | 1994-03-17 |            3 |
| Buffy    | 1989-05-13 |            5 |
| Fang     | 1990-08-27 |            8 |
| Bowser   | 1989-08-31 |            8 |
| Chirpy   | 1998-09-11 |            9 |
| Whistler | 1997-12-09 |           12 |
| Slim     | 1996-04-29 |            4 |
| Puffball | 1999-03-30 |            3 |
+----------+------------+--------------+
```

用下个月的生日找出动物也是容易的。假定当前月是 4 月，那么月值是 4 并且寻找在 5 月出生的动物（5 月），像这样：

```
mysql> SELECT name, birth FROM pet WHERE MONTH(birth) = 5;
+-------+------------+
| name  | birth      |
+-------+------------+
| Buffy | 1989-05-13 |
+-------+------------+
```

Green Apple Data Center

当然如果当前月份是 12 月，就有点复杂了。不是只把加 1 到月份数（12）上并且寻找在 13 月出生的动物，因为没有这样的月份。相反，寻找在 1 月出生的动物（1 月）。

甚至可以编写查询以便不管当前月份是什么它都能工作。这种方法不必在查询中使用一个特定的月份数字，DATE_ADD()允许把时间间隔加到一个给定的日期。如果把一个月加到 NOW()值上，然后用 MONTH()提取月份部分，结果产生寻找生日的月份：

```
mysql> SELECT name, birth FROM pet
    -> WHERE MONTH(birth) = MONTH(DATE_ADD(NOW(), INTERVAL 1 MONTH));
```

完成同样任务的一个不同方法是加 1 以得出当前月份的下一个月（在使用取模函数(MOD)后，如果它当前是 12，则“绕回”月份到值 0）：

```
mysql> SELECT name, birth FROM pet
    -> WHERE MONTH(birth) = MOD(MONTH(NOW()), 12) + 1;
```

注意，MONTH 返回在 1 和 12 之间的一个数字，且 MOD(something,12)返回在 0 和 11 之间的一个数字，因此必须在 MOD()以后加 1，否则将从 11 月(11)跳到 1 月(1)。

⑥ NULL 值操作

NULL 值开始时可能觉得很奇怪直到习惯于它。概念上，NULL 意味着“没有值”或“未知值”，且它被看作有点与众不同的值。为了测试 NULL，不能使用算术比较运算符例如=、<或!=。为了说明它，试试下列查询：

```
mysql> SELECT 1 = NULL, 1 != NULL, 1 < NULL, 1 > NULL;
+----------+-----------+----------+----------+
| 1 = NULL | 1 != NULL | 1 < NULL | 1 > NULL |
+----------+-----------+----------+----------+
|     NULL |      NULL |     NULL |     NULL |
+----------+-----------+----------+----------+
```

很清楚从这些比较中得到毫无意义的结果。只能使用 IS NULL 和 IS NOT NULL 操作符：

```
mysql> SELECT 1 IS NULL, 1 IS NOT NULL;
+-----------+---------------+
| 1 IS NULL | 1 IS NOT NULL |
+-----------+---------------+
|         0 |             1 |
+-----------+---------------+
```

在 MySQL 中，0 意味着假而 1 意味着真。

NULL 这样特殊的处理是为什么，在前面的章节中，为了确定哪个动物不再是活着的，使用 death IS NOT NULL 而不是 death != NULL。

⑦ 模式匹配

MySQL 提供标准的 SQL 模式匹配，以及一种基于像 Unix 实用程序如 vi、grep 和 sed 的扩展正则表达式模式匹配的格式。

SQL 的模式匹配允许使用“_”匹配任何单个字符，而“%”匹配任意数目字符（包括零个字符）。在 MySQL 中，SQL 的模式缺省是忽略大小写的。下面显示一些例子。注意在使用 SQL 模式时，不能使用=或!=；而使用 LIKE 或 NOT LIKE 比较操作符。

Green Apple Data Center

为了找出以“ b ”开头的名字：

```
mysql> SELECT * FROM pet WHERE name LIKE "b%";
+--------+--------+---------+------+------------+------------+
| name   | owner  | species | sex  | birth      | death      |
+--------+--------+---------+------+------------+------------+
| Buffy  | Harold | dog     | f    | 1989-05-13 | NULL       |
| Bowser | Diane  | dog     | m    | 1989-08-31 | 1995-07-29 |
+--------+--------+---------+------+------------+------------+
```

为了找出以“ fy ”结尾的名字：

```
mysql> SELECT * FROM pet WHERE name LIKE "%fy";
+--------+--------+---------+------+------------+-------+
| name   | owner  | species | sex  | birth      | death |
+--------+--------+---------+------+------------+-------+
| Fluffy | Harold | cat     | f    | 1993-02-04 | NULL  |
| Buffy  | Harold | dog     | f    | 1989-05-13 | NULL  |
+--------+--------+---------+------+------------+-------+
```

为了找出包含一个“ w ”的名字：

```
mysql> SELECT * FROM pet WHERE name LIKE "%w%";
+----------+-------+---------+------+------------+------------+
| name     | owner | species | sex  | birth      | death      |
+----------+-------+---------+------+------------+------------+
| Claws    | Gwen  | cat     | m    | 1994-03-17 | NULL       |
| Bowser   | Diane | dog     | m    | 1989-08-31 | 1995-07-29 |
| Whistler | Gwen  | bird    | NULL | 1997-12-09 | NULL       |
+----------+-------+---------+------+------------+------------+
```

为了找出包含正好 5 个字符的名字，使用“ _ ”模式字符：

```
mysql> SELECT * FROM pet WHERE name LIKE "_____";
+-------+--------+---------+------+------------+-------+
| name  | owner  | species | sex  | birth      | death |
+-------+--------+---------+------+------------+-------+
| Claws | Gwen   | cat     | m    | 1994-03-17 | NULL  |
| Buffy | Harold | dog     | f    | 1989-05-13 | NULL  |
+-------+--------+---------+------+------------+-------+
```

由 MySQL 提供的模式匹配的其他类型是使用扩展正则表达式。当对这类模式进行匹配测试时，使用 REGEXP 和 NOT REGEXP 操作符（或 RLIKE 和 NOT RLIKE，它们是同义词）。

扩展正则表达式的一些字符是：

（1）“ . ”匹配任何单个的字符。

（2）一个字符类“ [...] ”匹配在方括号内的任何字符。例如，“ [abc] ”匹配“ a ”、“ b ”或“ c ”。为了命名字

Green Apple Data Center

符的一个范围，使用一个“ - ”。“ [a-z] ”匹配任何小写字母，而“ [0-9] ”匹配任何数字。

（3）“ * ”匹配零个或多个在它前面的东西。例如，“ x* ”匹配任何数量的“ x ”字符，“ [0-9]* ”匹配的任何数量的数字，而“ .* ”匹配任何数量的任何东西。

（4）正则表达式是区分大小写的，但是如果希望，能使用一个字符类匹配两种写法。例如，“ [aA] ”匹配小写或大写的“ a ”而“ [a-zA-Z] ”匹配两种写法的任何字母。

（5）如果它出现在被测试值的任何地方，模式就匹配（只要他们匹配整个值，SQL 模式匹配）。

（6）为了定位一个模式以便它必须匹配被测试值的开始或结尾，在模式开始处使用“ ^ ”或在模式的结尾用“ $ ”。

为了说明扩展正则表达式如何工作，上面所示的 LIKE 查询在下面使用 REGEXP 重写：

为了找出以“ b ”开头的名字，使用“ ^ ”匹配名字的开始并且“ [bB] ”匹配小写或大写的“ b ”：

```
mysql> SELECT * FROM pet WHERE name REGEXP "^[bB]";
+--------+--------+---------+------+------------+------------+
| name   | owner  | species | sex  | birth      | death      |
+--------+--------+---------+------+------------+------------+
| Buffy  | Harold | dog     | f    | 1989-05-13 | NULL       |
| Bowser | Diane  | dog     | m    | 1989-08-31 | 1995-07-29 |
+--------+--------+---------+------+------------+------------+
```

为了找出以“ fy ”结尾的名字，使用“ $ ”匹配名字的结尾：

```
mysql> SELECT * FROM pet WHERE name REGEXP "fy$";
+--------+--------+---------+------+------------+-------+
| name   | owner  | species | sex  | birth      | death |
+--------+--------+---------+------+------------+-------+
| Fluffy | Harold | cat     | f    | 1993-02-04 | NULL  |
| Buffy  | Harold | dog     | f    | 1989-05-13 | NULL  |
+--------+--------+---------+------+------------+-------+
```

为了找出包含一个“ w ”的名字，使用“ [wW] ”匹配小写或大写的“ w ”：

```
mysql> SELECT * FROM pet WHERE name REGEXP "[wW]";
+----------+-------+---------+------+------------+------------+
| name     | owner | species | sex  | birth      | death      |
+----------+-------+---------+------+------------+------------+
| Claws    | Gwen  | cat     | m    | 1994-03-17 | NULL       |
| Bowser   | Diane | dog     | m    | 1989-08-31 | 1995-07-29 |
| Whistler | Gwen  | bird    | NULL | 1997-12-09 | NULL       |
+----------+-------+---------+------+------------+------------+
```

既然一个正规表达式可以出现在值的任何地方，其模式都匹配，就不必在先前的查询中在模式的两方面放置通配符，以使得它匹配整个值，就像使用了 SQL 模式一样。

为了找出包含正好 5 个字符的名字，使用“ ^ ”和“ $ ”匹配名字的开始和结尾，和 5 个“ . ”实例在两者之间：

```
mysql> SELECT * FROM pet WHERE name REGEXP "^.....$";
+-------+--------+---------+------+------------+-------+
| name  | owner  | species | sex  | birth      | death |
```

Green Apple Data Center

```
+-------+--------+---------+------+------------+-------+
| Claws | Gwen   | cat     | m    | 1994-03-17 | NULL  |
| Buffy | Harold | dog     | f    | 1989-05-13 | NULL  |
+-------+--------+---------+------+------------+-------+
```

也可以使用“ {n} **”重复** n **次操作符重写先前的查询：**

```
mysql> SELECT * FROM pet WHERE name REGEXP "^.{5}$";
+-------+--------+---------+------+------------+-------+
| name  | owner  | species | sex  | birth      | death |
+-------+--------+---------+------+------------+-------+
| Claws | Gwen   | cat     | m    | 1994-03-17 | NULL  |
| Buffy | Harold | dog     | f    | 1989-05-13 | NULL  |
+-------+--------+---------+------+------------+-------+
```

⑧ 行计数

数据库经常用于回答这样一个问题，某个类型的数据在一张表中出现的频率？例如，可能想要知道有多少宠物，或每位主人有多少宠物，或可能想要对动物施行各种类型的普查。

计算拥有动物的总数字与“在 pet **表中有多少行?”是同样的问题，因为每个宠物有一个记录。**COUNT() **函数计数非** NULL **结果的数目，所以数的动物的查询看起来像这样：**

```
mysql> SELECT COUNT(*) FROM pet;
+----------+
| COUNT(*) |
+----------+
|        9 |
+----------+
```

在前面，检索了拥有宠物的人的名字。如果想要知道每个主人有多少宠物，可以使用 COUNT()**函数：**

```
mysql> SELECT owner, COUNT(*) FROM pet GROUP BY owner;
+--------+----------+
| owner  | COUNT(*) |
+--------+----------+
| Benny  |        2 |
| Diane  |        2 |
| Gwen   |        3 |
| Harold |        2 |
+--------+----------+
```

注意，使用 GROUP BY **对每个** owner **分组计算所有记录，没有它，得到的一切都是错误消息：**

```
mysql> SELECT owner, COUNT(owner) FROM pet;
ERROR 1140 at line 1: Mixing of GROUP columns (MIN(),MAX(),COUNT()...)
with no GROUP columns is illegal if there is no GROUP BY clause
```

COUNT()**和** GROUP BY **对以各种方式分类的数据很有用。下列例子显示出实施动物普查操作的不同方式。**

Green Apple Data Center

每种动物数量：

```
mysql> SELECT species, COUNT(*) FROM pet GROUP BY species;
+---------+----------+
| species | COUNT(*) |
+---------+----------+
| bird    |        2 |
| cat     |        2 |
| dog     |        3 |
| hamster |        1 |
| snake   |        1 |
+---------+----------+
```

每种性别的动物数量：

```
mysql> SELECT sex, COUNT(*) FROM pet GROUP BY sex;
+------+----------+
| sex  | COUNT(*) |
+------+----------+
| NULL |        1 |
| f    |        4 |
| m    |        4 |
+------+----------+
```

（在这个输出中，NULL **表示“未知性别”。）**
按种类和性别组合的动物数量：

```
mysql> SELECT species, sex, COUNT(*) FROM pet GROUP BY species, sex;
+---------+------+----------+
| species | sex  | COUNT(*) |
+---------+------+----------+
| bird    | NULL |        1 |
| bird    | f    |        1 |
| cat     | f    |        1 |
| cat     | m    |        1 |
| dog     | f    |        1 |
| dog     | m    |        2 |
| hamster | f    |        1 |
| snake   | m    |        1 |
+---------+------+----------+
```

当使用 COUNT()**时，不必检索整个一张表。例如，先前的查询，当只在狗和猫上施行时，看起来像这样：**

```
mysql> SELECT species, sex, COUNT(*) FROM pet
    -> WHERE species = "dog" OR species = "cat"
    -> GROUP BY species, sex;
+---------+------+----------+
```

Green Apple Data Center

```
| species | sex    | COUNT(*) |
+---------+------+----------+
| cat     | f      |        1 |
| cat     | m      |        1 |
| dog     | f      |        1 |
| dog     | m      |        2 |
+---------+------+----------+
```

或者，如果只需要知道已知性别的动物数目：

```
mysql> SELECT species, sex, COUNT(*) FROM pet
    -> WHERE sex IS NOT NULL
    -> GROUP BY species, sex;
+---------+------+----------+
| species | sex    | COUNT(*) |
+---------+------+----------+
| bird    | f      |        1 |
| cat     | f      |        1 |
| cat     | m      |        1 |
| dog     | f      |        1 |
| dog     | m      |        2 |
| hamster | f      |        1 |
| snake   | m      |        1 |
+---------+------+----------+
```

6.4.5 使用多个数据库表

pet 表只追踪有哪些宠物。如果想要记录他们的其他信息，例如在他们一生中的事件，如何时看兽医或何时后代出生，需要另外的表。这张表应该像什么呢？

（1）它需要包含宠物名字，因此知道哪个事件属于此动物。

（2）它需要一个日期因此知道事件什么时候发生的。

（3）需要一个字段描述事件。

（4）如果想要分类事件，有一个事件类型字段将是有用的。

给出了这些考虑，为 event 表的 CREATE TABLE 语句可能看起来像这样：

```
mysql> CREATE TABLE event (name VARCHAR(20), date DATE,
    -> type VARCHAR(15), remark VARCHAR(255));
```

就像 pet 表，最容易的是，通过创建包含信息的一个定位符分隔的文本文件装载初始记录：

表 6-3 pet 表

Fluffy	1995-05-15	litter	4 kittens, 3 female, 1 male
Buffy	1993-06-23	litter	5 puppies, 2 female, 3 male
Buffy	1994-06-19	litter	3 puppies, 3 female
Chirpy	1999-03-21	vet	needed beak straightened
Slim	1997-08-03	vet	broken rib
Bowser	1991-10-12	kennel	
Fang	1991-10-12	kennel	
Fang	1998-08-28	birthday	Gave him a new chew toy
Claws	1998-03-17	birthday	Gave him a new flea collar
Whistler	1998-12-09	birthday	First birthday

Green Apple Data Center

像这样装载记录：

```
mysql> LOAD DATA LOCAL INFILE "event.txt" INTO TABLE event;
```

基于从已经运行在 pet 表上的查询中学到的，应该能执行在 event 表中记录的检索，原则是一样的。但是什么是 event 表本身不能回答但可能问的问题呢？

当他们有了一窝小动物时，假定想要找出每只宠物的年龄。event 表指出何时发生，但是为了计算母亲的年龄，需要她的出生日期。既然它被存储在 pet 表中，为了查询需要两张表：

```
mysql> SELECT pet.name, (TO_DAYS(date) - TO_DAYS(birth))/365 AS age, remark
    -> FROM pet, event
    -> WHERE pet.name = event.name AND type = "litter";
+--------+------+-----------------------------+
| name   | age  | remark                      |
+--------+------+-----------------------------+
| Fluffy | 2.27 | 4 kittens, 3 female, 1 male |
| Buffy  | 4.12 | 5 puppies, 2 female, 3 male |
| Buffy  | 5.10 | 3 puppies, 3 female         |
+--------+------+-----------------------------+
```

关于该查询要注意的几件事情：

（1）FROM 子句列出两个表，因为查询需要从两个表中拉出信息。

（2）当组合（联结-join）来自多个表的信息时，需要指定在一个表中的记录怎样能匹配其他表的记录。这很简单，因为它们都有一个 name 列。查询使用 WHERE 子句基于 name 值来匹配 2 个表中的记录。

（3）因为 name 列出现在两个表中，当引用列时，一定要指定哪个表。这通过把表名附在列名前实现。

不必有 2 个不同的表来执行一个联结。如果想要将一个表的记录与同一个表的其他记录进行比较，联结一个表到自身有时是有用的。例如，为了在的宠物之中繁殖配偶，可以用 pet 联结自身来进行相似种类的雄雌配对：

```
mysql> SELECT p1.name, p1.sex, p2.name, p2.sex, p1.species
    -> FROM pet AS p1, pet AS p2
    -> WHERE p1.species = p2.species AND p1.sex = "f" AND p2.sex = "m";
+--------+------+--------+------+---------+
| name   | sex  | name   | sex  | species |
+--------+------+--------+------+---------+
| Fluffy | f    | Claws  | m    | cat     |
| Buffy  | f    | Fang   | m    | dog     |
| Buffy  | f    | Bowser | m    | dog     |
+--------+------+--------+------+---------+
```

在这个查询中，我们为表名指定别名以便能引用列，并且使得每一个列引用是关联于哪个表显得更直观。

6.4.6 获得数据库和表的信息

如果忘记一个数据库或表的名字，或一个给定的表的结构是什么（例如，它的列叫什么），怎么办？MySQL 通过提供数据库及其支持的表的信息的几个语句解决这个问题。

已经见到了 SHOW DATABASES，它列出由服务器管理的数据库。为了找出当前选择了哪个数据库，使用 DATABASE()函数：

Green Apple Data Center

```
mysql> SELECT DATABASE();
+------------+
| DATABASE() |
+------------+
| menagerie  |
+------------+
```

如果还没选择任何数据库，结果是空的。
为了找出当前的数据库包含什么表（例如，当不能确定一个表的名字），使用这个命令：

```
mysql> SHOW TABLES;
+---------------------+
| Tables in menagerie |
+---------------------+
| event               |
| pet                 |
+---------------------+
```

如果想要知道一个表的结构，DESCRIBE 命令是很有用的；它显示有关一个表的每个列的信息：

```
mysql> DESCRIBE pet;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | YES  |     | NULL    |       |
| owner   | varchar(20) | YES  |     | NULL    |       |
| species | varchar(20) | YES  |     | NULL    |       |
| sex     | char(1)     | YES  |     | NULL    |       |
| birth   | date        | YES  |     | NULL    |       |
| death   | date        | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
```

Field 显示列名字，Type 是列的数据类型，Null 表示列是否能包含 NULL 值，Key 显示列是否被索引而 Default 指定列的缺省值。

如果在一个表上有索引，SHOW INDEX FROM tbl_name 生成有关它们的信息。

6.4.7 以批处理模式使用 mysql

在前面的章节中，交互式地使用 MySQL 输入查询并且查看结果。也可以批模式运行 MySQL。为了做到这些，把想要运行的命令放在一个文件中，然后告诉 MySQL 从文件读取它的输入：

```
shell> mysql < batch-file
```

如果需要在命令行上指定链接参数，命令可能看起来像这样：

```
shell> mysql -h host -u user -p < batch-file
Enter password: ********
```

Green Apple Data Center

当这样使用 mysql 时，正在创建一个脚本文件，然后执行脚本。

为什么要使用一个脚本？有很多原因：

（1）如果重复地运行查询（比如说，每天或每周），把它做成一个脚本使得在每次执行它时避免重新键入。

（2）能通过拷贝并编辑脚本文件从类似的现有的查询生成一个新查询。

（3）当正在开发查询时，批模式也是很有用的，特别对多行命令或多行语句序列。如果犯了一个错误，不必重新打入所有一切，只要编辑的脚本来改正错误，然后告诉 MySQL 再次执行它。

（4）如果有一个产生很多输出的查询，可以通过一个分页器而不是盯着它翻屏到屏幕的顶端来运行输出：

```
shell> mysql < batch-file | more
```

（5）能输出到一个文件中进行更一步的处理：

```
shell> mysql < batch-file > mysql.out
```

（6）可以散发脚本给另外的人，因此他们也能运行命令。

（7）一些情况不允许交互地使用，例如，当从一个 cron 任务中运行查询时。在这种情况下，必须使用批模式。

当以批模式运行 MySQL 时，比起交互地使用它时，其缺省输出格式是不同的（更简明些）。例如，当交互式运行 SELECT DISTINCT species FROM pet 时，输出看起来像这样：

```
+---------+
| species |
+---------+
| bird    |
| cat     |
| dog     |
| hamster |
| snake   |
+---------+
```

但是当以批模式运行时，像这样：

```
species
bird
cat
dog
hamster
snake
```

如果想要在批模式中得到交互的输出格式，使用 mysql -t。为了显示已输出被执行的命令，使用 mysql -vvv。

在上述几节中，只是详细介绍了 MySQL 的常用的操作，而没有实际的例子，有点纸上谈兵的感觉，从这一节起就介绍一些实例，从简单到复杂的应用，希望读者能更好的掌握 PHP 与 MySQL 黄金组合的数据库编程。

6.5 PHP 与 MySQL 入门实例

前面几节介绍了 SQL 语言基础知识和 MySQL 数据库的常用操作，这些都是学习 PHP 与 MySQL 数据库编程必须的基础，但是光有这些知识还不够，必须适当学习一些编程实例才能更好地掌握 PHP 与 MySQL 数据库

Green Apple Data Center

编程，本节从最基本的网页数据库编程开始，由简单到复杂一步步地讲述 PHP 与 MySQL 数据库编程，使读者能全面掌握这些数据库编程知识。

6.5.1 建立数据库表

从 MySQL 数据库中读取数据之前，必须得先在数据库中创建一些数据库表，MySQL 在安装时就创建了 mysql 和 test 两个数据库，mysql 数据库里面存放有用户权限表。用户权限表里会创建一个默认的用户（root），该用户是没有口令的。数据库管理员可以根据需要来增加用户并赋予用户各种不同的权限，本书只使用 root 用户。如果自己管理服务器和数据库，为了系统的安全，必须及时修改 root 用户的口令。

要做的第一件事情是实际创建出数据库。在 MySQL 的 bin 目录下的命令行下，键入下列命令：

```
mysqladmin -u root create mydb
```

这样就创建了一个名为“ mydb ”的数据库。-u 选项告诉 MySQL 使用的是 root 用户。下一步就要加入一些数据，这里用的示例数据是大家都熟悉的员工数据库。建立这些数据库和数据表将会用到上一节的一些 MySQL 基础知识。如果读者对这些建立数据库和数据表还不太清楚，请详细阅读 6.4 节内容。把下面的文字复制到一个文件中，把该文件存在 MySQL 的 bin 目录下（假定文件名是 mydb.dump），这段文字内容十分简单，就是创建一个名叫 employees 的数据表，并在这个表中添加 3 条数据记录。

```
CREATE TABLE employees (
                id tinyint(4) DEFAULT '0' NOT NULL AUTO_INCREMENT,
                first varchar(20),
                last varchar(20),
                address varchar(255),
                position varchar(50),
                PRIMARY KEY (id),
                UNIQUE id (id)
                );
INSERT INTO employees VALUES (1,'Bob','Smith','128 Here St, Cityname','Marketing Manager');
INSERT INTO employees VALUES (2,'John','Roberts','45 There St , Townville','Telephonist');
INSERT INTO employees VALUES (3,'Brad','Johnson','1/34 Nowhere Blvd, Snowston','Doorman');
```

如果文字是折行的，请确保每一个 INSERT 语句都是另起一行的。下一步就是要把数据加入到 mydb 数据库中了。在命令行下，键入下面的命令：

```
mysql -u root mydb < mydb.dump
```

此时应该不会遇到什么错误，如果真的出错了，请仔细检查一下是否因上面的文字折行而引起错误。

现在已经把数据导入到数据库中了，接着来处理这些数据，把下面的文字存入一个文件中，把该文件存在 Web 服务器的文档目录下，后缀名为 testdb.php。

```
<html>
<body>
<?php
$db = mysql_connect("localhost", "root","");
mysql_select_db("mydb",$db);
$result = mysql_query("SELECT * FROM employees",$db);
printf("First Name: %s<br>\n", mysql_result($result,0,"first"));
```

Green Apple Data Center

```
printf("Last Name: %s<br>\n", mysql_result($result,0,"last"));
printf("Address: %s<br>\n", mysql_result($result,0,"address"));
printf("Position: %s<br>\n", mysql_result($result,0,"position"));
?>
</body>
</html>
```

下面解释一下上面的代码。要用 PHP 实现数据库编程，首先要链接数据库。mysql_connect()函数就是以指定的用户名（本例中用户名是 root）链接到指定机器（在本例中机器是本机 localhost）上的 MySQL 数据库。用户名 root 后面是它的口令，注意""里面什么字符也没有，千万不要输入一个空格，否则就连不上服务器（这里 MySQL 服务器上的 root 用户没有口令，所以也可以省略），链接的结果保存在变量$db 中。关于这些 PHP 的有关 MySQL 数据库函数的详细用法请参考附录。

连上服务器后，就应该选择要读取的数据库名称。mysql_select_db()函数就是告诉 PHP 要读取的数据库是 mydb。可以在程序中同时链接到多台机器上的多个数据库，但目前这个程序还是限于链接一个数据库。

接下来就是要对选择的数据库进行操作了，mysql_query()函数完成最复杂也是最重要的部分。利用刚才得到的链接结果标识$db，该函数把一行 SQL 语句送给 MySQL 服务器去处理，这里的 SELECT * FROM employees 是查询数据库 mydb 中的 employees 表中的全部记录，返回的结果保存在变量$result 中。

最后，mysql_result()函数显示 SQL 查询命令所得到的各个字段的值。利用变量$result，就可以找到第一条记录，记录号是 0，并将其中各字段的值显示出来。

如果读者以前没用过 Perl 或 C 语言，那么 printf 函数的语法格式会显得很奇怪。在上面的每一行程序中，%s 代表表达式第二部分中的那个变量（例如，mysql_result ($result,0,"position")）应该以字符串的形式显示出来。

在浏览器里运行 testdb.php 这个程序文件，则会显示出如图 6-1 所示的结果。虽然这个程序非常简单，但是用 PHP 程序来处理数据库中的信息基本上就是这样一个过程，即链接服务器、选择数据库、操作数据库、在浏览器里返回结果的这样一个简单的过程。下面进行一些稍为复杂的工作，来显示多行记录的数据，甚至与数据库互相交换数据。

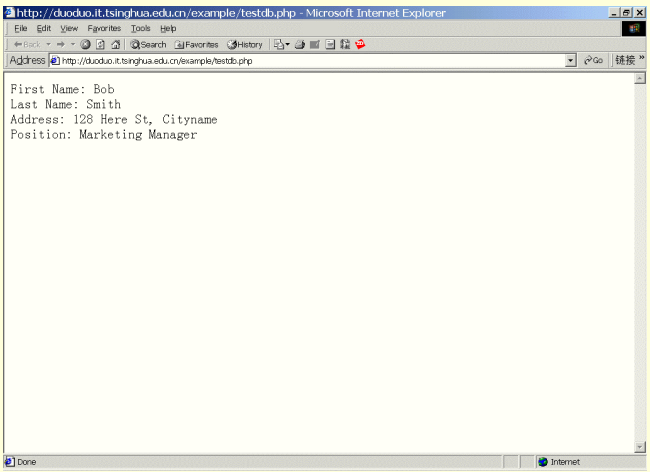


图 6-1　testdb.php 运行结果

6.5.2　while 循环

6.5.1 小节从创建的数据库开始，讲述怎样显示库中的数据，下面稍微加以润色继续深入下去讲述怎样使用 PHP 和 MySQL 来写出一些简单而有用的页面。

首先，看下面的用来查询数据库记录的 testdb1.php 代码内容，注意和上一小节有什么不同的地方。

```
<html>
<body>
```

Green Apple Data Center

```
<?php
$db = mysql_connect("localhost", "root");
mysql_select_db("mydb",$db);
$result = mysql_query("SELECT * FROM employees",$db);
echo "<table border=1>\n";
echo "<tr><td>姓名</td><td>地址</td></tr>\n";
while ($myrow = mysql_fetch_row($result)) {
printf("<tr><td>%s %s</td><td>%s</td></tr>\n", $myrow[1], $myrow[2], $myrow[3]);
}
echo "</table>\n";
?>
</body>
</html>
```

在浏览器里运行结果如图 6-2 所示，读者可能已经注意到，在这个程序里加进了一些新东西，最明显的是 while()循环。该循环的含义是只要数据库里还有记录可读（使用 mysql_fetch_row()函数），那就把该记录赋给变量$myrow，然后执行大括号（{}）内的指令。仔细理解这里，这部分是比较重要的。

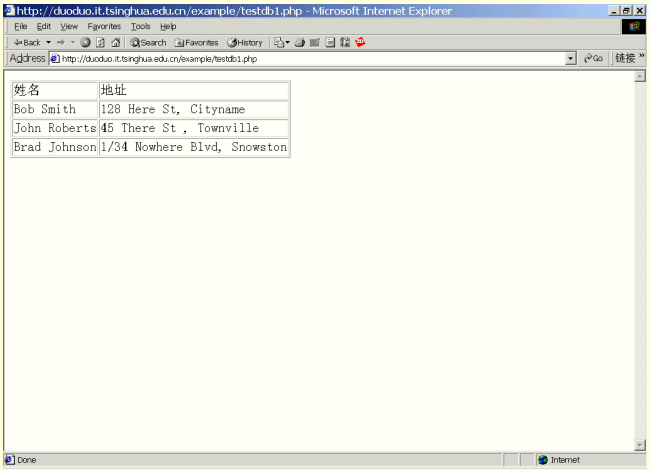


图 6-2 testdb1.php 运行结果

应该注意一下 mysql_fetch_row()函数，它返回的是一个数组，必须以数组下标来访问其中的某个字段，第一个字段下标为 0，第二个是 1，依此类推。在执行某些复杂查询时，这么做实在是太繁琐了，不过这个数据表中的记录很简单，这样编写代码也显得非常简单。

现在更仔细地研究一下循环过程。程序前几行在上一小节例子中已经看到过了。然后，在 while()循环里，从查询结果中读取一条记录并把该记录赋给数组$myrow。接着，用 printf 函数把数据中的内容显示在屏幕上。随后，循环反复执行，读取下一条记录赋给$myrow。这样继续下去，直到所有记录都已被读取完为止。

使用 while()循环的一个好处是，如果数据库查询没有返回任何记录，那也不会收到错误信息。在刚才执行循环语句时，循环条件就不满足，不会有任何数据赋给$myrow，程序就直接往下运行了。

但是如果查询未返回任何数据，一个优秀的程序员应该考虑给出提示，让用户知道这点相关的消息。这是可以做到的，下面就讲述是怎么实现的。

6.5.3 if-else

请看下面用来查询数据库记录的 testdb2.php 代码内容，注意和上面的 testdb1.php 有什么不同的地方。

```
<html>
<body>
<?php
```

Green Apple Data Center

```
$db = mysql_connect("localhost", "root");
mysql_select_db("mydb",$db);
$result = mysql_query("SELECT * FROM employees",$db);
if ($myrow = mysql_fetch_array($result)) {
echo "<table border=1>\n";
echo "<tr><td>姓名</td><td>地址</td></tr>\n";
do    {
printf("<tr><td>%s %s</td><td>%s<td></tr>\n", $myrow["first"],
$myrow["last"], $myrow["address"]);
      }
while ($myrow = mysql_fetch_array($result));
echo "</table>\n";
                                  }
else   {
echo "对不起，没有找到记录！";
      }
?>
</body>
</html>
```

这段程序中包含有不少新内容，不过这些内容都相当简单，在浏览器里的结果如图 6-3 所示，和 testdb1.php 运行结果是一样的。这个程序首先是 mysql_fetch_array()函数，该函数与 mysql_fetch_row()十分相近，只有一点不同：使用这个函数时，可以通过字段名而不是数组下标来访问它返回的字段，比如$myrow["first"]。这样就可以省不少力气了。另外，程序中还加进了 do/while 循环和 if-else 条件判定语句。

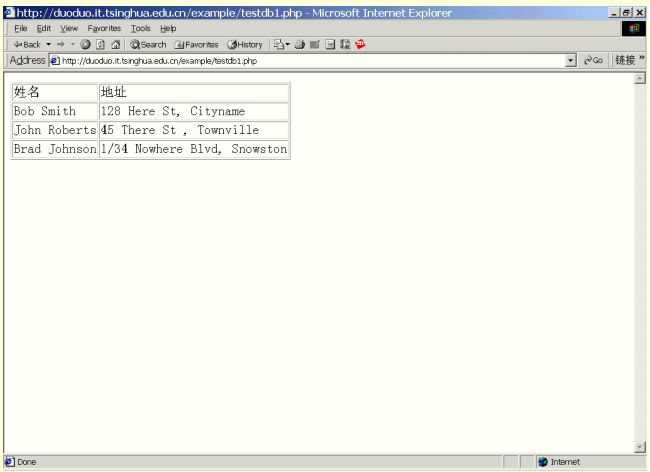


图 6-3　testdb2.php 运行结果

if-else 条件判定语句的含意是，如果成功地把一条记录赋给了$myrow 变量，那就继续；否则，就跳到 else 部分，执行那里的指令。

do/while 循环是 while()循环的一个变体。这里要用到 do/while 的原因是：在最初的 if 语句中，已经把查询返回的第一条记录赋给变量$myrow 了，如果这时执行一般的 while 循环（比如，while ($myrow = mysql_fetch_row($result)），那就会把第二条记录赋给$myrow，而第一条记录就被冲掉了。但是 do/while 循环可以执行一次循环体内容之后再来判定循环条件。因此，就不会漏掉第一条记录了。

最后，如果查询结果没有任何记录的话，程序就会执行包含在 else{}部分的那些语句。如果想看到这部分程序的执行情况，可以把 SQL 语句改为 SELECT * FROM employees WHERE id=6，或改成其他形式，使得查询结果中没有任何记录。结果如图 6-4 所示。

Green Apple Data Center

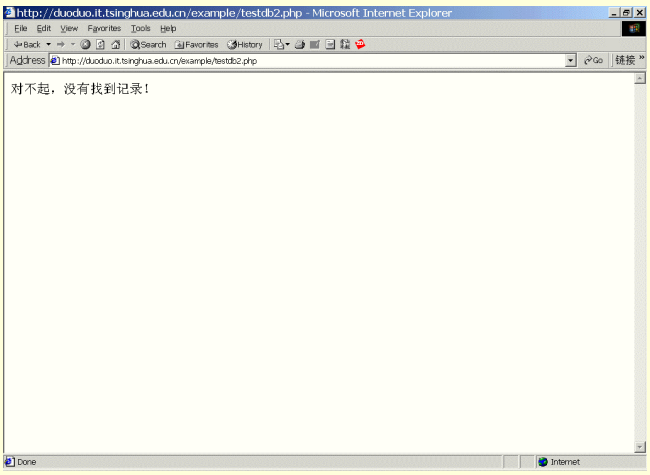


图 6-4 testdb2.php 运行结果

下面扩充一下循环 if-else 代码，使得页面内容更加丰富。

6.5.4 一个实际的程序脚本

上面介绍了重要的循环语句，下面将在一个更加实际一点的例子中看看如何运用它。但是在这之前，应该知道如何处理 Web 表格、查询参数串，以及表单的 GET 方法和 POST 方法。

现在要处理查询参数串，有三种方法可以把参数内容写入到查询参数串中：第一种是在表格中使用 GET 方法；第二种是在浏览器的地址栏中输入网址时直接加上查询参数；第三种是把查询参数串嵌入到网页的超链接中，使得超链接的内容像下面这样：< href="http://duoduo.it.tsinghua.edu.cn/example/mypage.php?id=1">。现在就要用到最后这一种方法。

再来查询数据库列出员工姓名，看下面的 testdb3.php 程序代码，其中大部分内容都已经很熟悉了。

```
<html>
<body>
<?php
$db = mysql_connect("localhost", "root");
mysql_select_db("mydb",$db);
$result = mysql_query("SELECT * FROM employees",$db);
if ($myrow = mysql_fetch_array($result)) {
do    {
printf("<a href=\"%s?id=%s\">%s %s</a><br>\n",
$PATH_INFO, $myrow["id"], $myrow["first"], $myrow["last"]);
}
while ($myrow = mysql_fetch_array($result));
}
else {
echo "对不起，没有找到记录！";
}
?>
</body>
</html>
```

结果如图 6-5 所示，这里没什么特别的，只是 printf 函数有些不同。首先要注意的是，所有的引号前面都有一个反斜杠。这个反斜杠告诉 PHP 直接显示后面的字符，而不能把后面的字符当作程序代码来处理。另外要注意变量$PATH_INFO 的用法，该变量在所用程序中都可以访问，是用来保存程序自身的名称与目录位置的，

Green Apple Data Center

之所以用到它是因为要在页面中再调用这个程序本身。使用$PATH_INFO 可以做到，即使程序被挪到其他目录，甚至是其它机器上时，也能保证正确地调用到这个程序，所以优秀的程序员都应该保持这些良好的编程习惯。

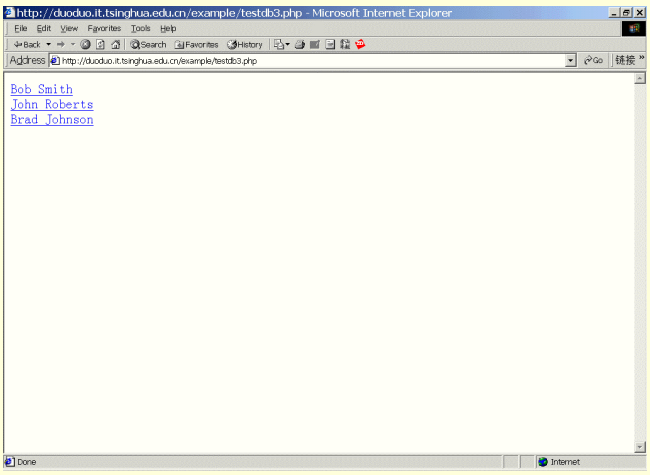


图 6-5 testdb3.php 运行结果

程序所生成的网页，其中包含的超链接会再次调用程序本身。不过，再次调用时，会加入一些查询参数，如图 6-6 所示。

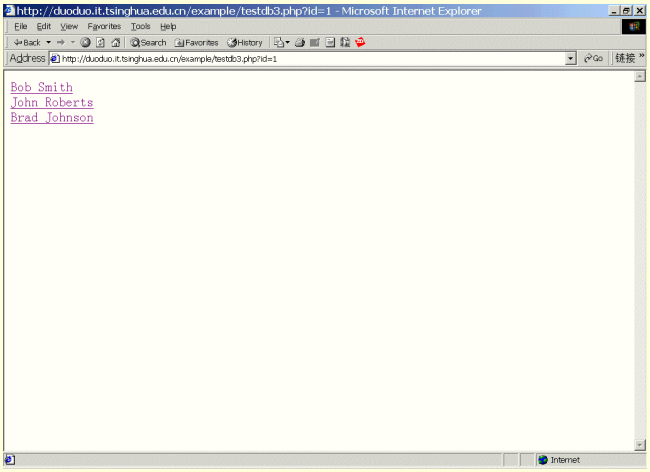


图 6-6 testdb3.php 运行结果

PHP 见到查询参数串中包含有“名字=值”这样的成对格式时，会作一些特别的处理，它会自动生成一个变量，变量名称与取值都与查询参数串中所给定的名称和取值相同。这一功能使得可以在程序中判断出是第一次执行本程序还是第二次，所要做的只是问 PHP$id 这个变量是否存在。

知道这个问题的答案后，可以在第二次调用程序时显示一些不同的结果出来。请看看下面的 testdb4.php 程序代码，程序开始变得复杂了，所以在这里面加了注释，来解释一下到底发生了什么，可以用//加入单行注释，或者用/*和*/来括住大段的注释。

```
<html>
<body>
<?php
$db = mysql_connect("localhost", "root");
mysql_select_db("mydb",$db);
// display individual record
// 显示单条记录内容
if ($id) {
```

Green Apple Data Center

```
$result = mysql_query("SELECT * FROM employees WHERE id=$id",$db);

$myrow = mysql_fetch_array($result);
printf("名: %s\n<br>", $myrow["first"]);
printf("姓: %s\n<br>", $myrow["last"]);
printf("住址: %s\n<br>", $myrow["address"]);
printf("职位: %s\n<br>", $myrow["position"]);
        }
else {
// 显示员工列表
$result = mysql_query("SELECT * FROM employees",$db);
if ($myrow = mysql_fetch_array($result)) {
// 如果有记录，则显示列表
do    {
printf("<a href=\"%s?id=%s\">%s %s</a><br>\n", $PATH_INFO,
$myrow["id"], $myrow["first"], $myrow["last"]);
        }
while ($myrow = mysql_fetch_array($result));
                                    }
else {
// 没有记录可显示
echo "对不起，没有找到记录！";
}
}
?>
</body>
</html>
```

程序的运行结果如图 6-7 所示，单击任意一个链接后，$id 这个变量就存在了，所以 PHP 就执行 if 语句中的内容，例如单击了 Brad Johnson 链接后，浏览器就显示 Brad Johnson 的有关信息，如图 6-8 所示。

到此为止，这是第一个真正有用的 PHP/MySQL 脚本程序！下一步要看看怎样把 Web 表格加进来，并且向数据库发送数据。

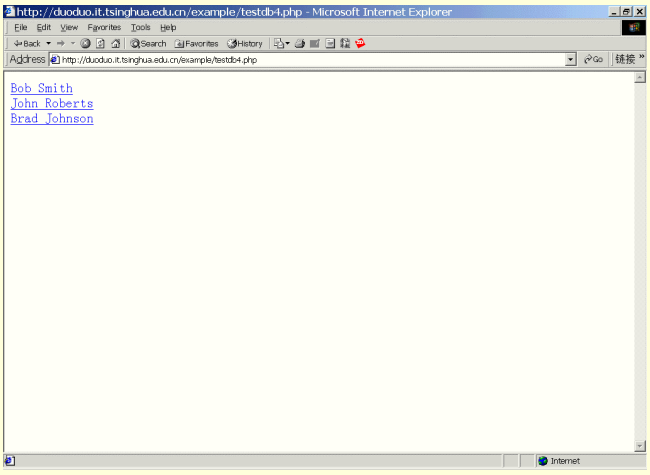


图 6-7 testdb3.php 运行结果

Green Apple Data Center

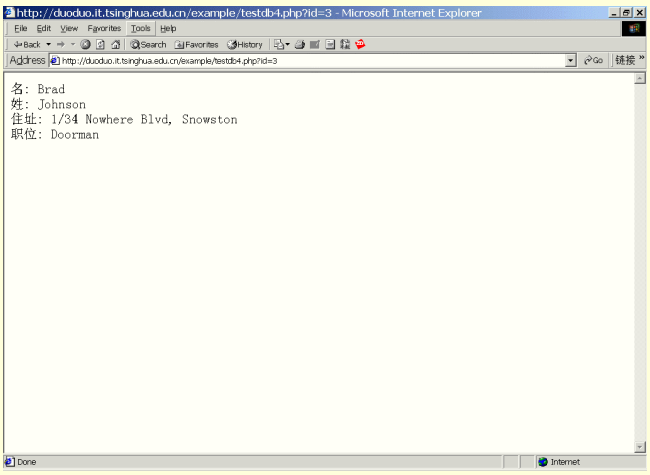


图 6-8　testdb3.php 运行结果

6.5.5　向服务器发送数据

现在从数据库读取数据已经没有太多困难了。但是怎么反过来向数据库发送数据呢？首先，创建一个带有简单表格的网页。

```
<html>
<body>
<form method="post" action="<?php echo $PATH_INFO?>">
名：<input type="Text" name="first"><br>
姓：<input type="Text" name="last"><br>
住址：<input type="Text" name="address"><br>
职位：<input type="Text" name="position"><br>
<input type="Submit" name="submit" value="输入信息">
</form>
</body>
</html>
```

这个网页在浏览器中如图 6-9 所示，看上去可能不美观，不过不要紧，本书着重程序功能的介绍，重点在于掌握程序的编写，至于网页的美观性就交给美工去做吧。这里同样要注意$PATH_INFO 的用法，可以在 HTML 代码中的任意位置使用 PHP。读者也会注意到，表格中的每一个元素都对应着数据库中的一个字段。这种对应关系并不是必须的，这么做只是更直观一些，便于以后理解这些代码。

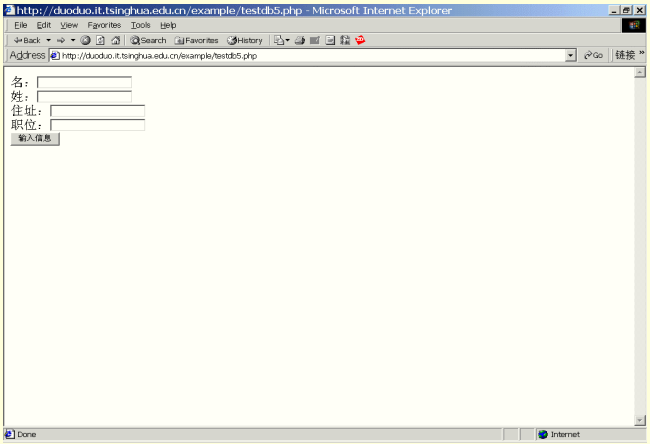


图 6-9　testdb5.php 运行结果

Green Apple Data Center

还要注意的是，在 Submit 按钮中加入了 name 属性，这样在程序中可以试探$submit 变量是否存在。于是，当网页被再次调用时，就会知道调用页面时是否已经填写了表格。

应该指出，不一定要把上面的网页内容写到 PHP 程序中，再返过来调用程序本身。完全可以把显示表格的网页和处理表格的程序分开放在两个网页、三个网页甚至更多网页中，放在一个文件中只是可以使内容更加紧凑而已。

现在加入一些代码，向数据库发送在表格中填写的内容，程序用$HTTP_POST_VARS 把所有查询参数变量都显示出来，这只不过是为了证明 PHP 确实把所有变量都传给了程序，这种方法是一个很有用的调试手段，如果要想看全部的变量，可以用$GLOBALS。

```
<html>
<body>
<?php
if ($submit) {
while (list($name, $value) = each($HTTP_POST_VARS)) {
echo "$name = $value <br>\n";
        }
// 处理表格输入
$db = mysql_connect("localhost", "root");
mysql_select_db("mydb",$db);
$sql = "INSERT INTO employees (first,last,address,position)
        VALUES ('$first','$last','$address','$position')";
$result = mysql_query($sql);
echo "信息已经加入数据库。\n";
}
else{
// 显示表格
 ?>
<form method="post" action="<?php echo $PATH_INFO?>">
名：<input type="Text" name="first"><br>
姓：<input type="Text" name="last"><br>
住址：<input type="Text" name="address"><br>
职位：<input type="Text" name="position"><br>
<input type="Submit" name="submit" value="输入信息">
</form>
<?php
} // end if，if 结束

?>
</body>
</html>
```

把上面代码存为 testdb5.php 文件，用户填写表格内容后，单击输入信息按钮，如果出现如图 6-10 所示的结果，就表明向数据库中插入数据成功了。

Green Apple Data Center

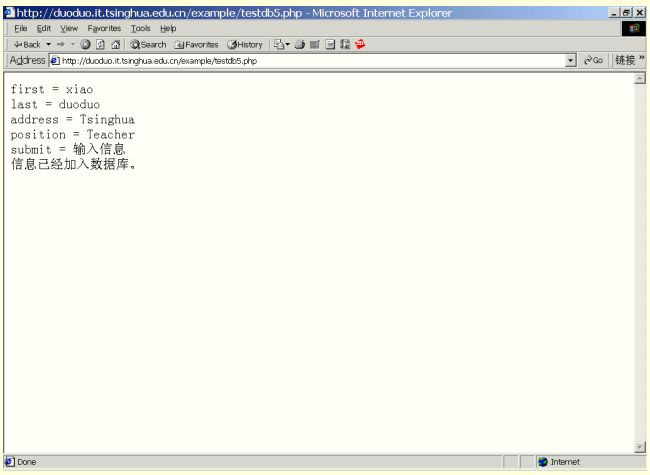


图 6-10　testdb5.php 运行结果

6.5.6　修改数据

上面已经讲述了如何把数据插入到数据库中，现在讲述如何修改数据库中已有的记录。对数据库的读写，都把要执行的 SQL 语句放到一个变量（$sql）中，然后才用 mysql_query()来执行数据库查询。在调试时这是很有用的，如果程序出了什么问题，随时可以把 SQL 语句的内容显示出来，检查其中的语法错误。

首先，前面讲述了在网页中显示员工姓名，这里要把数据显示在表格中，请看看下面的 testdb6.php 程序代码。

```
<html>
<body>
<?php
$db = mysql_connect("localhost", "root");
mysql_select_db("mydb",$db);
if ($id) {
// 查询数据库
$sql = "SELECT * FROM employees WHERE id=$id";
$result = mysql_query($sql);
$myrow = mysql_fetch_array($result);
?>
<form method="post" action="<?php echo $PATH_INFO?>">
<input type=hidden name="id" value="<?php echo $myrow["id"] ?>">
名：<input type="Text" name="first" value="<?php echo $myrow["first"] ?>"><br>
姓：<input type="Text" name="last" value="<?php echo $myrow["last"] ?>"><br>
住址：<input type="Text" name="address" value="<?php echo
    $myrow["address"] ?>"><br>
职位：<input type="Text" name="position" value="<?php echo
    $myrow["position"] ?>"><br>
<input type="Submit" name="submit" value="输入信息">
</form>
<?php
}
else {
// 显示员工列表
$result = mysql_query("SELECT * FROM employees",$db);
```

Green Apple Data Center

```
while ($myrow = mysql_fetch_array($result)) {
printf("<a href=\"%s?id=%s\">%s %s</a><br>\n", $PATH_INFO,
                        $myrow["id"], $myrow["first"], $myrow["last"]);
    }
}
?>
</body>
</html>
```

在浏览器运行如图 6-11 **所示，单击其中一个链接，比如** xiao duoduo**，就会出现如图** 6-12 **所示的结果，关于** xiao duoduo **的数据信息全部都显示在表格中。**

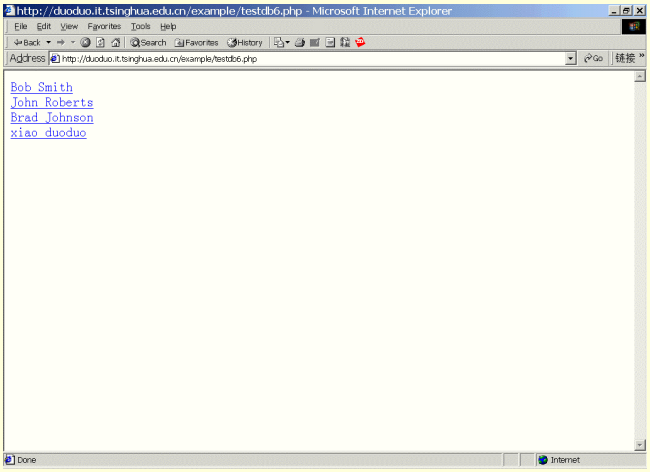


图 6-11　testdb6.php **运行结果**

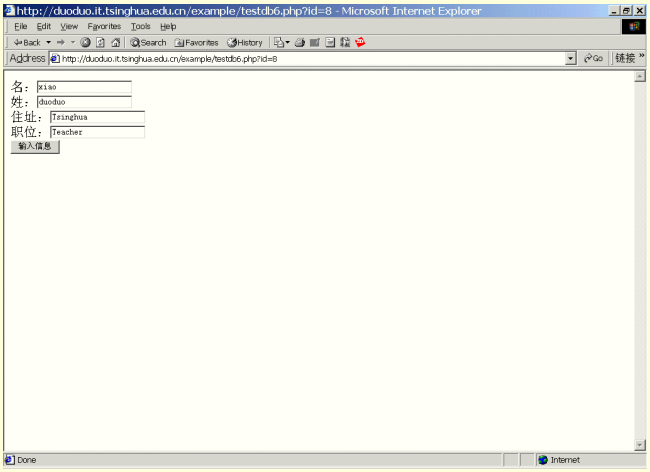


图 6-12　testdb6.php **运行结果**

刚才是把字段内容写入到相应表格元素中的 value **属性里，这是相当简单的，再往前一步，使程序可以把用户修改过的内容写回数据库去，通过** Submit**（输入信息）按钮来判断是否处理表格输入内容，只是用的** SQL **语句稍稍有些不同，代码如下：**

```
<html>
<body>
<?php
$db = mysql_connect("localhost", "root");
mysql_select_db("mydb",$db);
```

Green Apple Data Center

```
if ($id) {
        if ($submit) {
         $sql = "UPDATE employees SET first='$first',last='$last',
         address='$address',position='$position' WHERE id=$id";
         $result = mysql_query($sql);
         echo "谢谢！数据更改完成\n";
        }
else {
// 查询数据库
$sql = "SELECT * FROM employees WHERE id=$id";
$result = mysql_query($sql);
$myrow = mysql_fetch_array($result);
?>
<form method="post" action="<?php echo $PATH_INFO?>">
<input type=hidden name="id" value="<?php echo $myrow["id"] ?>">
名：<input type="Text" name="first" value="<?php echo $myrow["first"] ?>"><br>
姓：<input type="Text" name="last" value="<?php echo $myrow["last"] ?>"><br>
住址：<input type="Text" name="address" value="<?php echo
    $myrow["address"] ?>"><br>
职位：<input type="Text" name="position" value="<?php echo
    $myrow["position"] ?>"><br>
<input type="Submit" name="submit" value="输入信息">
</form>
<?php
}
}
else {
// 显示员工列表
$result = mysql_query("SELECT * FROM employees",$db);
while ($myrow = mysql_fetch_array($result)) {
printf("<a href=\"%s?id=%s\">%s %s</a><br>\n", $PATH_INFO,
                        $myrow["id"], $myrow["first"], $myrow["last"]);
    }
}
?>
</body>
</html>
```

在这个程序中已经包含了讲述过的大多数知识，这里只是在一个 if()**条件判别语句中又加了一个** if()**语句，来检查多重条件，判断所要执行的** SQL **语句，同时可见** SQL **语句在数据库编程中的重要性。**

下面，把所有东西全都加在一起，写出一个很好的完整的程序来。

6.5.7 完整的程序

要把所有功能加入到一个程序中，使它具有增加、编辑修改、删除记录的功能，这是前面所有内容的一个延伸，也可以作为极好的复习方法，看看下面的 testdb8.php **程序。**

```
<html>
```

Green Apple Data Center

```
<body>
<?php
$db = mysql_connect("localhost", "root");
mysql_select_db("mydb",$db);
if ($submit) {
        // 如果没有 ID，则我们是在增加记录，否则我们是在修改记录
    if ($id) {
       $sql = "UPDATE employees SET first='$first',last='$last',
       address='$address',position='$position' WHERE id=$id";
    }
    else {
       $sql = "INSERT INTO employees (first,last,address,position)
       VALUES ('$first','$last','$address','$position')";
    }
    // 向数据库发出 SQL 命令
    $result = mysql_query($sql);
    echo "记录修改成功！<br>";
}
elseif ($delete) {
// 删除一条记录
        $sql = "DELETE FROM employees WHERE id=$id";
        $result = mysql_query($sql);
        echo "记录删除成功！<br>";
}
else {
     // 如果我们还没有按 submit 按钮，那么执行下面这部分程序
    if (!$id) {
       // 如果不是修改状态，则显示员工列表
       $result = mysql_query("SELECT * FROM employees",$db);
       while ($myrow = mysql_fetch_array($result)) {
          printf("<a href=\"%s?id=%s\">%s %s</a> \n",
          $PATH_INFO, $myrow["id"], $myrow["first"], $myrow["last"]);
    printf("<a href=\"%s?id=%s&delete=yes\">(DELETE)</a><br>", $PATH_INFO, $myrow["id"]);
       }
    }
    ?>
    <br>
    <a href="<?php echo $PATH_INFO?>"> 一条记录</a>
    <br>
    <form method="post" action="<?php echo $PATH_INFO?>">
    <?php
    if ($id) {
       // 我们是在编辑修改状态，因些选择一条记录
       $sql = "SELECT * FROM employees WHERE id=$id";
       $result = mysql_query($sql);
       $myrow = mysql_fetch_array($result);
       $id = $myrow["id"];
```

Green Apple Data Center

```
        $first = $myrow["first"];
        $last = $myrow["last"];
        $address = $myrow["address"];
        $position = $myrow["position"];
        // 显示 id，供用户编辑修改
        ?>
        <input type=hidden name="id" value="<?php echo $id ?>">
        <?php
    }
    ?>
    名：<input type="Text" name="first" value="<?php echo $first ?>"><br>
    姓：<input type="Text" name="last" value="<?php echo $last ?>"><br>
    住址：<input type="Text" name="address" value="<?php echo $address ?>"><br>
    职位：<input type="Text" name="position" value="<?php echo $position ?>"><br>
    <input type="Submit" name="submit" value="输入信息">
    </form>
<?php
}
?>
</body>
</html>
```

这段程序看起来很复杂，但实际上并不难，程序主要有三个部分：第一个 if()语句检查是否已经按下了那个“输入信息”的数据提交按钮。如果是，程序再检查$id 是否存在；如果不存在，那我们就是在增加记录状态，否则，是在修改记录状态。

接下来检查变量$delete 是否存在，如果存在，是要删除记录。注意，第一个 if()语句检查的是用 POST 方法发送来的变量，而这一次检查的是 GET 方法中传递过来的变量。

最后，程序默认的动作是显示员工列表和表格。同样，要检查变量$id 是否存在，如果存在，就根据它的值检索出相应的记录显示出来。否则，会显示一个空的表格。该程序在浏览器里执行结果如图 6-13 所示。

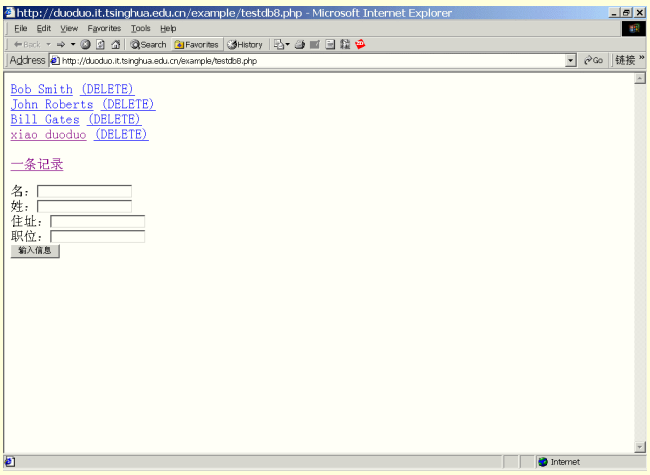


图 6-13　testdb8.php 运行结果

6.6　PHP 与 MySQL 一个留言薄实例

大部分的网站，都会考虑到和使用者之间的互动关系。这时，用留言板的功能，可让使用者留下到此一游，

Green Apple Data Center

或者是一些和网站的互动讯息。

在设计上，可以很简单的只留下使用者的短篇留言，也可以设计到依性质分门别类很复杂的 Web BBS 系统。下例是一个非常实用的留言本程序，简洁，清晰，对于改变源程序设置成自己特色的留言也非常容易，所以也非常适合初学 PHP 的读者学习 PHP+MYSQL 的结合和 Cookies 的使用。

6.6.1 建立数据库及数据表

首先要建立一个名叫 dowebs 的数据库，可以在 mysql 命令行输入以下命令创建：

```
mysql> CREATE DATABASE dowebs ；
```

然后要创建一个名叫 dat 的数据表，它包括的字段及其字段含义如表 6-4 所示。

表 6-4 数据表：dat

字段	类型	Null	缺省值	额外	说明
id	tinyint(4)	否	0	auto_increment	编号
name	varchar(20)	否			留言人姓名
mail	varchar(80)	是			留言人电子邮
host	varchar(80)	是			留言人主页
title	varchar(70)	是			主题
text	text	是			内容
addtime	datetime	否	0000-00-00 00:00:00		留言时间

读者可以用 phpMyAdmin 等工具可视化建立数据表，也可以用下列的 SQL 语句建立数据表：

```
CREATE TABLE dat (
        id tinyint(4) DEFAULT '0' NOT NULL AUTO_INCREMENT,
        name varchar(20) NOT NULL,
        mail varchar(80),
        host varchar(80),
        title varchar(70),
        text text,
        addtime datetime DEFAULT '0000-00-00 00:00:00' NOT NULL,
PRIMARY KEY (id),
        UNIQUE id (id)
        );
```

6.6.2 写留言页面

写留言的页面程序（guestbook.php）非常简单，只是一些 HTML 代码，没有与 PHP 和 MySQL 有关的知识，这里完全可以用 FrontPage 或者 Dreamveaver 制作一个如下图 6-14 所示的漂亮的界面。其源程序如下：

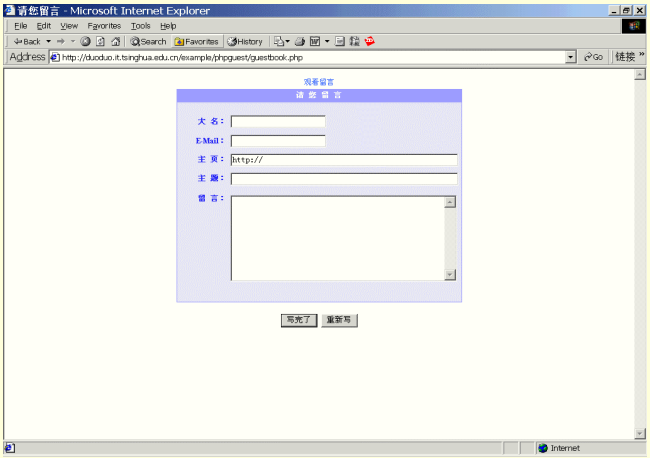


图 6-14 留言的页面

Green Apple Data Center

```
<html>
<head>
<meta NAME="GENERATOR" Content="Microsoft FrontPage 3.0">
<meta HTTP-EQUIV="Content-Type" content="text/html; charset=gb2312">
<meta HTTP-EQUIV="Expires" CONTENT="0">
<title>请您留言</title>
<style type="text/css">
body { font-family: 宋体; font-size: 9pt; }
table { font-family: 宋体; font-size: 9pt; }
a { text-decoration: none;color:3366ff}
a:active { text-decoration: none }
a:hover { color: red; font-size: 9pt; text-decoration: underline }
</style>
</head>
<body bgcolor="#FFFFFF">
<form method="post" action="savedata.php" name="frmGstBook">
 <div align="center"><center>
 <table border="0" cellspacing="1" width="450"
 bgcolor="#9999FF">
 <caption><span class="smallFont"><a href="index.php">观看留言</a></span></caption>
 <tr>
 <td height="20">
 <div align="center"></div>
 <p align="center"><strong><font color="#ffffff">请 您 留 言</font></strong>
 </td>
 </tr>
 <tr align="center" bgcolor="#e3e3FF">
 <td height="270" valign="center" align="middle">
 <div align="center"><center>
 <table
 border="0" cellpadding="2" width="450" height="310">
 <tr>
 <td width="92" height="25">
 <p align="right"><font color="#3333FF"><strong>大 名：</strong></font>
 </td>
 <td width="312" height="25"><font>
 <input class="smallInput" name="name" size="20"
 value="">
 </font></td>
 </tr>
 <tr>
 <td width="92" height="25">
 <p align="right"><font color="#3333FF"><strong><font
 face="Times New Roman">E-Mail</font>：</strong></font>
 </td>
 <td width="312" height="25">
 <input class="smallInput" name="mail" size="20"
```

Green Apple Data Center

```
value="">
</td>
</tr>
<tr>
<td width="92" height="25">
<p align="right"><font color="#3333FF"><strong>主 页：</strong></font>
</td>
<td width="312" height="25">
<input class="smallInput" name="host" size="50"
value="http://">
</td>
</tr>
<tr>
<td width="92" height="25">
<p align="right"><font color="#3333FF"><strong>主 题：</strong></font>
</td>
<td width="312" height="25">
<input class="smallInput" name="title" size="50">
</td>
</tr>
<tr style="COLOR: #ff9933">
<td width="92" height="5">
<div align="right"><font color="#3333FF"></font></div>
</td>
<td width="312" height="5"></td>
</tr>
<tr>
<td width="92" height="163" valign="top">
<p align="right"><font color="#3333FF"><strong>留 言：</strong></font>
</td>
<td width="312" height="163" valign="top">
<p><font color="#000000">
<textarea
class="smallArea" cols="48" name="text" rows="9"></textarea>
</font>
</td>
</tr>
</table>
</center></div></td>
</tr>
</table>
</center></div><div align="center"><center><p><input type="submit" value="写完了"
class="buttonface" name="cmdOk"> <input type="reset" value="重新写" name="cmdReset"
class="buttonface"></p>
</center></div>
</form>
</body>
```

Green Apple Data Center

```
</html>
```

6.6.3 留言处理程序

留言处理程序（savedata.php**）是处理上面输入的留言，并把这些留言写入数据库。其源程序如下：**

```
<html>
<head>
<title>〖欢迎给 xiaoduoduo 留言〗</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
body { font-family: 宋体; font-size: 9pt; }
table { font-family: 宋体; font-size: 9pt; }
a { text-decoration: none;color:3366ff}
a:active { text-decoration: none }
a:hover { color: red; font-size: 9pt; text-decoration: underline }
</style>
</head>
<body bgcolor="#FFFFFF">
<p align=center>〖xiaoduoduo 的留言薄〗</p>
<?
$dbhost="localhost"; //主机地址

$dbname="dowebs"; //数据库名称
$dbusername=""; //数据用户
$dbpassword=""; //数据口令
$name=htmlspecialchars(trim($name));
$mail=htmlspecialchars(trim($mail));
$host=htmlspecialchars(trim($host));
if ($host=="http://"){$host="";}
$title=htmlspecialchars(trim($title));
$text=htmlspecialchars(trim($text));
$errormsg="";
if ($name=="")
{$errormsg="请输入你姓名！";}
else {
if ($title=="" and $text=="")
{$errormsg="请输入你的主题或内容!";}
else
{
if($title==""){$title="无主题";}
if($text==""){$text="无内容";}
$db=mysql_connect($dbhost,$dbusername,$dbpassword);
mysql_select_db($dbname,$db);
$sql="insert into dat (name,mail,host,title,text,addtime,attrib) values ('$name','$mail','$host','$title', '$text',NOW(),1)";
$result=mysql_query($sql);
echo("<center>");
echo "您已经成功留下了宝贵意见！</b>谢谢您的留言！<p><a href=index.php>返回</a></b>";
```

Green Apple Data Center

```
echo("</center>");
}}
if($errormsg<>""){echo "$errormsg <a href=guestbook.php>返 回 重 写</a>";}
?>
</body></html>
```

6.6.4 留言显示程序

留言显示程序（index.php），注意其分页功能是怎样实现的，其源程序如下：

```
<html>
<head>
<title>〖欢迎给 xiaoduoduo 留言〗</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css"> body { font-family: 宋体; font-size: 9pt; } table { font-family: 宋体; font-size: 9pt; }
 a { text-decoration: none;color:3366ff} a:active { text-decoration: none }
 a:hover { color: red; font-size: 9pt; text-decoration: underline } </style></head>
 <body bgcolor="#FFFFFF"> <p>〖xiaoduoduo 的留言簿〗<center><a href="../index.htm">返回主页</a></center></p> <a href="admin.php">管理员由此进</a>
<?
$sign=$HTTP_COOKIE_VARS["sign"];
$dbhost="localhost"; //主机地址
$dbname="dowebs"; //数据库名称
$dbusername="root"; //数据用户
$dbpassword=""; //数据口令
$showrow=5; //每页最大显示数
function showpages($maxrow,$thispage)
{global $showrow; //每页最大显示数
 if (($maxrow/$showrow)==Floor($maxrow/$showrow))
{$n=Floor($maxrow/$showrow);}
else
{$n=Floor($maxrow/$showrow)+1;}
if ($n==1) {echo "<p align='center'><a href=guestbook.php>写写留言</a></p>";}
else
 {echo "<p align='left'>&gt;&gt; 留言分页 ";
 for ($k=1; $k<$n+1;$k++)
 {if ($k==$thispage) {echo "[<b>$k</b>] ";}
else
{echo "[<b><a href='index.php?page=$k'>$k</a></b>] ";} }
echo "<a href=guestbook.php>写写留言</a></p>"; }}//显示数据函数
 function show_announce($id)
{global $db;
global $board_id;
$query="select * from dat where id='$id' and attrib=1";
 $result=mysql_query($query,$db);
$myrow=mysql_fetch_array($result);
echo "<table width=\"700\" border=\"1\" cellspacing=\"0\" bordercolorlight=\"#9999FF\" bordercolordark =\"#FFFFFF\">";
echo "<tr bgcolor=\"#ddddFF\"><td width=\"70\"><div align=\"right\"><font color=\"#3300CC\"><b> 大 名 ：
```

Green Apple Data Center

```
</b></font></div></td><td >$myrow[name]</td></tr>";

    if($myrow[mail]<>""){echo "<tr><td width=\"70\"><div align=\"right\"><font color=\"#3300CC\"> <b>E_MAIL ：
</b></font></div></td><td ><a href=\"mailto:$myrow[mail]\">$myrow[mail]</td></tr>";}
    if($myrow[host]<>"") {echo "<tr><td width=\"70\"><div align=\"right\"><font color=\"#3300CC\"><b> 主 页 ：
</b></font></div></td><td><a href=\"$myrow[host]\" target=\"_blank\">$myrow[host]</td></tr>";}
    echo "<tr><td width=\"70\"><div align=\"right\"><font color=\"#3300CC\"><b>主题：</b></font> </div></td><td>$myrow[title]
($myrow[addtime])</td></tr>";echo "<tr><td width=\"70\" valign=\"top\">
    <div align = \"right\"><font color = \"#3300CC\"><b>内容：</b></font></div></td><td>$myrow[text] </td></tr></table>";
    global $sign; if ($sign=="dowebs") {echo "<a href=delete.php?id=$myrow[id]>删除</a>";}
     mysql_free_result($result);}
    if ($page<2) {$page=1;}
    $db=mysql_connect($dbhost,$dbusername,$dbpassword);
    mysql_select_db($dbname,$db);
    $offset=($page-1)*$showrow; $query="select id from dat where attrib=1 order by id desc";
    $result_top=mysql_query($query,$db);
    $mum=mysql_num_rows($result_top);
    showpages($mum,$page);
    $query="select id from dat where attrib=1 order by id desc limit $offset, $showrow";
     $result_top=mysql_query($query,$db);
     while($myrow_top=mysql_fetch_array($result_top)){show_announce($myrow_top[id]);}
     showpages($mum,$page);mysql_close($db); ?>
    </body>
    </html>
```

在浏览器中结果如图 6-15 所示。

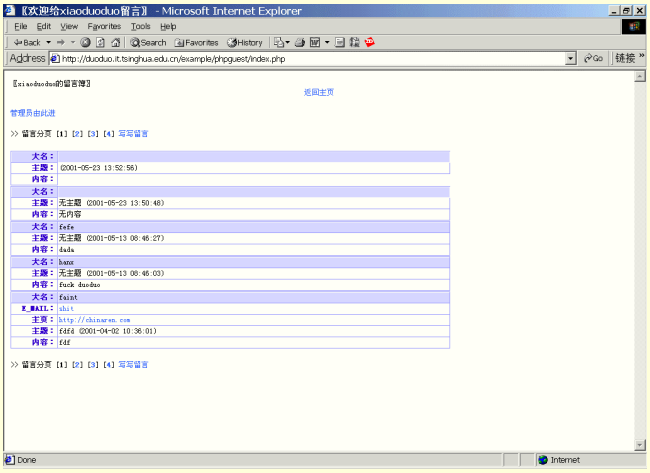


图 6-15　留言显示

6.6.5　管理员程序

admin.php

```
<?
if ($password=="123")
{SetCookie("sign", "dowebs");
echo"<html><head><meta http-equiv=\"refresh\"content=\"0;url=index.php\"></head><body></body> </html>";}
```

Green Apple Data Center

```
else{
if ($password<>"") echo "对不起，你的密码不对，不能通过！";
echo "<html><head></head><body bgcolor=\"#FFFFFF\"><form name=\"form1\" action=\"admin.php\"  method=\"post\">请输入留言本管理密码:
<input type=\"password\" name=\"password\"><input type=\"submit\" value=\"提交\"></form></body> </html>";
}
?>
```

delete.php

```
<html>
<head>
<title>〖xiaoduoduo 留言〗</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
body        { font-family: 宋体; font-size: 9pt; }
table       { font-family: 宋体; font-size: 9pt; }
a           { text-decoration: none;color:3366ff}
a:active { text-decoration: none }
a:hover   { color: red; font-size: 9pt; text-decoration: underline }
</style>
</head>
<body bgcolor="#FFFFFF">
<p>〖xiaoduoduo〗</p>
<?
if (($HTTP_COOKIE_VARS["sign"]="dowebs")&&($id>0)){
$dbhost="localhost"; //主机地址
$dbname="dowebs";   //数据库名称
$dbusername="root";   //数据用户
$dbpassword="";          //数据口令
$db=mysql_connect($dbhost,$dbusername,$dbpassword);
mysql_select_db($dbname,$db);
$sql="delete from dat where id='$id'";
$result=mysql_query($sql);
echo "<b>OK!!删除成功 ！</b><a href=index.php>返 回</a></b>";
}else
{echo "<b>ERROR!!不能删除 ！</b><a href=index.php>返 回</a></b>";}
?>
</body></html>
```

Green Apple Data Center

第 7 章　PHP 与 MySQL 综合实例

通过前面几章的介绍，读者对 PHP 与 MySQL 有了基本的了解，也能够编写一些简单的应用程序，本章通过一个具体的 BBS 论坛实例展示 PHP 与 MySQL 在网站建设上面的非凡能力，读者完全理解这个实例之后，相信对 PHP 与 MySQL 这对黄金组合的也就完全掌握了。

BBS 是英文 Bulletin Board System 的缩写，意为电子公告牌系统，它是在计算机网络上设立一个或多个电子论坛，一般以匿名的方式向公众提供访问的权利，使得公众以电子信息的方法发表自己的观点。利用 PHP 与 MySQL 的强大功能，很容易创建一个 BBS 系统。

对一个商业论坛的建设，要考虑的问题很多，这里单从技术上讲，一个 BBS 论坛系统要有两方面的内容，一是针对普通用户的程序，二是这针对管理员的管理程序。

以本章论坛为例，用户服务程序包括用户注册登录、发表文章、查询文章等；管理员程序包括管理员登录、文章维护、用户帐号管理等。

7.1　建立数据库

BBS 论坛有众多的用户帐号和密码，有很多用户发表的文章，所用这些都要通过数据库来管理，这里选择优秀的 MySQL 数据库。安装 MySQL 时已经创建好了一个名叫 mysql 的数据库，在这里面再添加 4 个表（board、boarduser、online、plan），这 4 个表包含了本论坛所用的数据库信息。其中：

（1）Board 表主要是建立论坛各个版面，包括版面代号、版面名称、版面的版主等信息。

```
CREATE TABLE board(
boardid int(11) NOT NULL auto_increment,
boardname char(255),
boardmaster char(50),
masteremail char(50),
masterpassword char(50),
plan int(5) default '0',
PRIMARY KEY (boardid)
);
```

（2）Boarduser 表主要是建立用户的有关信息，包括用户帐号、用户密码、用户电子邮件等信息。

```
CREATE TABLE boarduser (
                    id int(11) NOT NULL auto_increment,
                    username char(50),
                    password char(50),
                    email char(50),
                    comefrom char(50),
                    sex char(2),
                    homepage char(50),
                    addtime datetime,
                    writename char(255),
                    num int(6) default '0',
                    PRIMARY KEY (id)
```

Green Apple Data Center

```
);
```

（3）Online 表主要是记录在线用户，包括用户的 IP 地址、上站时间等信息。

```
CREATE TABLE online (
                    id int(11) NOT NULL auto_increment,
                    ip char(30) NOT NULL,
                    conntime datetime DEFAULT '0000-00-00 00:00:00' NOT NULL,
                    PRIMARY KEY (id)
);
```

（4）Plan 表是非常重要的表，主要记录用户发表的文章相关信息，包括发表文章的用户信息、文章的内容、文章的回复情况及其内容等。

```
CREATE TABLE plan (
    planid int(11) NOT NULL auto_increment,
    boardid int(11),
    rootid int(11),
    layer int(11),
    title varchar(255),
    body text,
    username varchar(50),
    email varchar(50),
    link varchar(50),
    picture varchar(50),
    renum int(11) DEFAULT '0',
    clicknum int(11) DEFAULT '0',
    writename varchar(255),
    linktitle varchar(255),
    emote varchar(20),
    retoemail char(2) DEFAULT '否',
    addtime datetime,
    parentid int(11),
    glod char(2) DEFAULT '否',
    hote int(11) DEFAULT '0',
    PRIMARY KEY (planid)
);
```

7.2 进站页面

如图 7-1 所示的进站主页面，其程序名称是 index.php，这里采用框架结构（frame）方便用户浏览整个论坛，源程序如下：

Green Apple Data Center

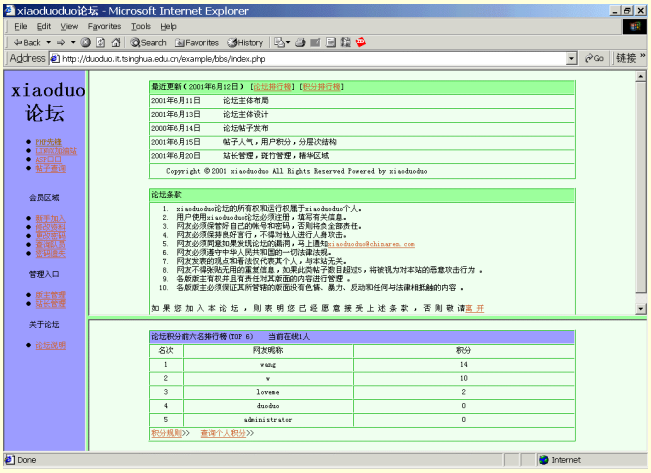


图 7-1　进站主页面

index.php

```
<html>
<head><title>xiaoduoduo 论坛</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<frameset cols="130,646*" frameborder="YES" border="3" framespacing="3" rows="*" bordercolor ="#99FF99">
   <frame name="leftFrame" scrolling="NO" noresize src="left.php">
   <frameset rows="253,238*" frameborder="YES" border="5" framespacing="5" cols="*" bordercolor="#99FF99">
      <frame src="up.php" name="BoardList" target="mian" scrolling="auto">
         <frame src="down.php" name="BoardAnnounce">
   </frameset>
</frameset>
<noframes><body bgcolor="#FFFFFF">
</body></noframes>
</html>
```

index.php **都是由** HTML **语言代码编写而成，没有用到** PHP **的有关语法，它只是调用了** 3 **个程序：**left.php、up.php、down.php。

其中 left.php **主要用于显示论坛的入口，包括各个版面程序入口、用户服务程序入口和管理员程序入口，程序源代码如下：**

left.php

```
<html>
<head>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
```

Green Apple Data Center

```
</style>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#9999FF">
<div align="center"><h2>xiaoduoduo 论坛 </h2></div>
<ul>
   <li><a href="list.php?boardid=1&amp;page=1" target="BoardList">PHP 先锋</a></li>
   <li><a href="list.php?boardid=2&amp;page=1" target="BoardList">LINUX 加油站</a></li>
   <li><a href="list.php?boardid=3&amp;page=1" target="BoardList">ASP 口口</a></li>
   <li><a href="search.php" target="BoardAnnounce">帖子查询</a></li>

</ul>
<div align="center"><br>
   会员区域 </div>
<ul>
   <li><a href="addnew.php" target="BoardAnnounce">新手加入</a></li>
   <li><a href="modify.php" target="BoardAnnounce">修改资料</a></li>
   <li><a href="modify.php" target="BoardAnnounce">更改密码</a></li>
   <li><a href="queryuser.php" target="BoardAnnounce">查询队员</a></li>
   <li><a href="forgetpassword.php" target="BoardAnnounce">密码遗失</a></li>
</ul>
<div align="center">管理入口<br>
</div>
<ul>
   <li><a href="banzhu/mastermanage.php" target="BoardList">版主管理</a></li>
   <li><a href="admin/master.php" target="BoardList">站长管理</a></li>
</ul>
<div align="center">关于论坛<br>
</div>
<ul>
    <li><a href="banzhu/mastermanage.php" target="BoardList">论坛说明</a></li>
</ul>
</html>
```

up.php 主要用于显示本论坛的最新情况和用户在本论坛发表文章必须遵守的一些条款，其源代码如下：

```
up.php
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
```

Green Apple Data Center

```
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
</head>

<body bgcolor="#ECFFEC">
<table width = "80%" border = "1" cellspacing = "0" cellpadding = "3" bordercolorlight = "#FFFFFF" bordercolordark = "#00B000" align = "center">
  <tr bgcolor="#99FF99">
    <td>最近更新（2001 年 6 月 12 日） [<a href="plantop.php" target="BoardAnnounce">论坛排行榜</a>]
      [<a href="down.php" target="BoardAnnounce">积分排行榜</a>]</td>
  </tr>
  <tr>
    <td>2001 年 6 月 11 日      论坛主体布局</td>
  </tr>
  <tr>
    <td>2001 年 6 月 13 日      论坛主体设计     </td>
  </tr>
  <tr>
    <td height="21">2000 年 6 月 14 日      论坛帖子发布 </td>
  </tr>
  <tr>
    <td>2001 年 6 月 15 日      帖子人气，用户积分，分层次结构 </td>
  </tr>
  <tr>
    <td>2001 年 6 月 20 日      站长管理，斑竹管理，精华区域</td>
  </tr>
  <tr>
    <td>    Copyright &copy; 2001 xiaoduoduo All Rights Reserved  Powered by xiaoduoduo</td>
  </tr>
</table>
<br>
<table width="80%" border="1" cellspacing="0" cellpadding="3" bordercolorlight="#FFFFFF" bordercolordark="#00B000" align="center">
  <tr bgcolor="#99FF99">
    <td>论坛条款</td>
  </tr>
  <tr>
    <td>
      <ol>
        <li> xiaoduoduo 论坛的所有权和运行权属于 xiaoduoduo 个人。 </li>
        <li>用户使用 xiaoduoduo 论坛必须注册，填写有关信息。 </li>
        <li>网友必须保管好自己的帐号和密码，否则将负全部责任。 </li>
        <li>网友必须保持良好言行，不得对他人进行人身攻击。</li>
```

Green Apple Data Center

```
                <li> 网友必须同意如果发现论坛的漏洞，马上通知 <a href="mailto:xiaoduoduo @chinaren.com
">xiaoduoduo@chinaren.com
                  </a></li>
                <li>网友必须遵守中华人民共和国的一切法律法规。</li>
                <li>网友发表的观点和看法仅代表其个人，与本站无关。</li>
                <li>网友不得张贴无用的重复信息，如果此类帖子数目超过 5，将被视为对本站的恶意攻击行为 。</li>
                <li>各版版主有权并且有责任对其版面的内容进行管理。</li>
                <li>各版版主必须保证其所管辖的版面没有色情、暴力、反动和任何与法律相抵触的内容 。</li>
              </ol>
              <p> 如果您加入本论坛，则表明您已经愿意接受上述条款，否则敬请<a href="javascript: window.close()">离
                开</a> </p>
            </td>
          </tr>
        </table>
        </body>
        </html>
```

down.php 主要用于显示论坛积分前 6 位的网友名称，请注意其中关于数据库的查询，非常简单，第 6 章讲述过类似例子，其源代码如下：

```
        down.php
        <?include "online.php"?>
        <html>
        <head>
        <title>Untitled Document</title>
        <style type="text/css">
        <!--
        a:active {    color: #666600; text-decoration: underline}
        a:hover {    color: #FF0000; text-decoration: none}
        a:link {    color: #CC6633; text-decoration: underline}
        a:visited {    color: #996600; text-decoration: underline}
        body {    font-size: 12px}
        td {    font-size: 12px}
        -->
        </style>
        <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
        </head>
        <body bgcolor="#ECFFEC">
        <?
        $conn=mysql_connect('localhost:3306','root','');
        mysql_select_db("mysql");
        $sql="select*from boarduser order by num desc limit 0,6";
        $rs=mysql_query($sql,$conn);
        ?>
        <table width = "80%" border = "1" cellspacing = "0" cellpadding = "3" bordercolorlight = "#FFFFFF" bordercolordark="#00B000"
align="center">
          <tr bgcolor="#9999FF">
```

Green Apple Data Center

```
    <td colspan="3">论坛积分前六名排行榜(TOP 6)    当前在线<?echo $online;?>人</td>
  </tr>
  <tr>
    <td width="8%" nowrap>
      <div align="center">名次</div>
    </td>
    <td width="40%" nowrap>
      <div align="center">网友昵称</div>
    </td>
    <td width="52%" nowrap>
      <div align="center">积分</div>
    </td>
  </tr>
  <?$i=1;
     while ($result=mysql_fetch_array($rs)){
          ?>
  <tr>
    <td width="8%" nowrap align="center"><?echo $i;?>
      <div align="center"></div>
    </td>
    <td width="40%" nowrap align="center"><?echo $result["username"];?>
      <div align="center"></div>
    </td>
    <td width="52%" nowrap align="center"><?echo $result["num"];?>
      <div align="center"></div>
    </td>
  </tr>
  <?$i++;
     };?>
  <tr>
    <td colspan="3"><a href="toporder.htm" target="_blank">积分规则</a>&gt;&gt;   <a href="queryuser.php">查询个人积分</a>&gt;&gt;   </td>
  </tr>
</table>
</body>
</html>
```

7.3 用户服务程序

用户服务程序主要用于新用户注册，用户修改其注册资料、查询其它用户等。

7.3.1 新用户注册

新用户注册由论坛条款程序（addnew.php**）、用户注册页面程序（**regnew.php**）、和注册处理程序（**saveruser.php**）组成。**

论坛条款程序（addnew.php**）非常简单，主要是介绍一些用户在本论坛必须遵守的一些条款，其源程序如下：**

Green Apple Data Center

addnew.php

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#ECFFEC">
<table width="80%" border="1" cellspacing="0" cellpadding="3"
bordercolorlight="#FFFFFF" bordercolordark="#00B000" align="center">
  <tr bgcolor="#99FF99">
    <td>请阅读完论坛条款</td>
  </tr>
  <tr>
    <td>
      <ol>
        <li> xiaoduoduo 论坛的所有权和运行权属于 xiaoduoduo。  </li>
        <li>用户使用 xiaoduoduo 论坛必须注册，填写有关信息。  </li>
        <li>网友必须保管好自己的帐号和密码，否则将负全部责任。  </li>
        <li>网友必须保持良好言行，不得对他人进行人身攻击。</li>
        <li>  网友必须同意如果发现 xiaoduoduo 论坛的漏洞,马上通知 xiaoduoduo 论坛站长 xiaoduoduo@chinaren.com </li>
        <li>网友必须遵守中华人民共和国的一切法律法规  </li>
        <li>网友发表的观点和看法仅代表其个人，与本站无关网友不得张贴垃圾信息（美元美分、上网赚钱等等）  </li>
        <li>网友不得张贴无用的重复信息，如果此类帖子数目超过 5，将被视为对本站的恶意攻击行为  </li>
        <li>各版版主有权并且有责任对其版面的内容进行管理  </li>
        <li>各版版主必须保证其所管辖的版面没有色情、暴力、反动和任何与法律相抵触的内容  </li>
        <li>xiaoduoduo 论坛保留对恶意攻击采取行动的权利，情节严重的我们将采取法律手段  </li>
        <li>xiaoduoduo 论坛有权对登记网友的账号进行管理，有权对违反上述条款的网友账号进行关闭、删除的处罚  </li>
      </ol>
      <p align="center">
     <input type="button" name="Button" value="  我同意" onclick="document.location='regnew.php'">
     <input type="button" name="Submit2" value="我不同意" >
      </p>
    </td>
  </tr>
</table>
</body>
```

Green Apple Data Center

```
</html>
```

用户注册页面程序（regnew.php）用于注册新用户，注意其中的对用户所填写的资料作正确性检查的程序部分，源程序如下：

```
regnew.php
<html>
<head>
<title>xiaoduoduo 论坛>>--->>注册新用户</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#ECFFEC">
// 注意下面的 javascrip 语句，用来判断输入非空，非常实用
<script languge="javascript">
<!--
function check(the){
        var checkok=true;
        if( the.username.value=="" || the.password.value=="" || the.repassword.value==""){
                alert("请把资料填写完整,谢谢!!!");
                checkok=false;
        };
        return checkok;
};
//-->
</script>
// 将用户输入的特殊字符转为 HTML 格式
<?
$username=htmlspecialchars($username);
$email=htmlspecialchars($email);
$homepage=htmlspecialchars($homepage);
$sex=htmlspecialchars($sex);
$comefrom=htmlspecialchars($comefrom);
$writename=htmlspecialchars($writename);
?>
<form method = "post" action =" saveuser.php" name = "regnew" onsubmi t= "return check (document.regnew)">
```

Green Apple Data Center

```
<table width="80%" border="1" cellspacing="0" cellpadding="3" bordercolorlight="#FFFFFF" bordercolordark="#00B000" align="center">
  <tr bgcolor="#99FF99">
    <td colspan="2">xiaoduoduo 论坛注册新用户 (带<font color="#FF0000">*</font>号的必须填写)</td>
  </tr>
  <tr>
    <td>用户名： </td>
    <td>
<input type="text" name="username" maxlength="50" size="20" value="<?echo $username;?>">
      <font color="#FF0000">*</font> </td>
  </tr>
  <tr>
    <td> 密码：</td>
    <td>
      <input type="password" name="password" maxlength="50" size="20">
      <font color="#FF0000">*</font> </td>
  </tr>
  <tr>
    <td>密码确认：</td>
    <td>
      <input type="password" name="repassword" maxlength="50" size="20">
      <font color="#FF0000">*</font> </td>
  </tr>
  <tr>
    <td>E-mail：</td>
    <td>
      <input type="text" name="email" maxlength="50" size="30" value="<?echo $email;?>">
    </td>
  </tr>
  <tr>
    <td>个人主页：</td>
    <td>
<input type="text" name="homepage" maxlength="50" size="30" value="<?echo $homepage;?>">
    </td>
  </tr>
  <tr>
    <td>性别：</td>
    <td>
      <input type="radio" name="sex" value="男" <?if($sex!="女"){?>checked<?};?>>
      男   
      <input type="radio" name="sex" value="女" <?if($sex=="女"){?>checked<?};?>>
      女 </td>
  </tr>
  <tr>
    <td>来自何方：</td>
    <td>
<input type="text" name="comefrom" maxlength="50" size="30" value="<?echo $comefrom;?>">
```

Green Apple Data Center

```
        </td>
      </tr>
      <tr>
        <td>签名：</td>
        <td>
  <input type="text" name="writename" maxlength="255" size="40" value="<?echo $writename;?>">
        </td>
      </tr>
      <tr>
        <td colspan="2">
          <input type="submit" name="Submit" value="下一步&gt;&gt;">
          <input type="reset" name="Submit2" value="  重写  ">
        </td>
      </tr>
  </table>
</form>
</body>
</html>
```

本程序运行结果如图 7-2 所示。

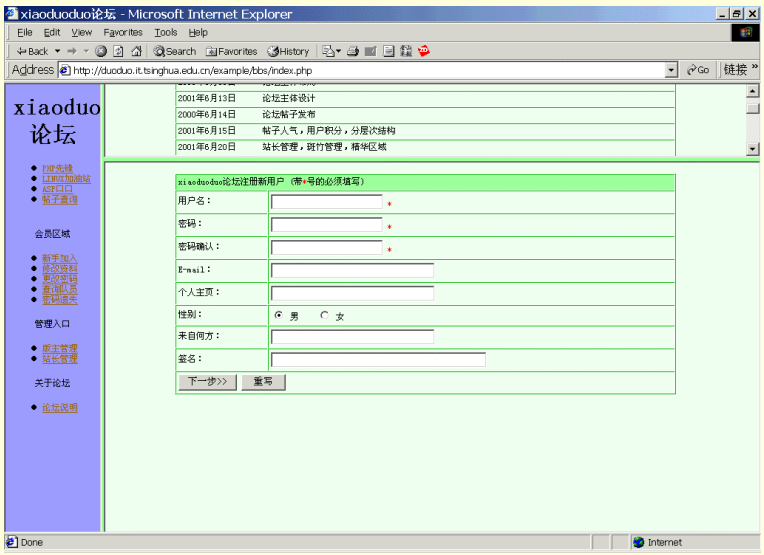


图 7-2　新用户注册界面

注册处理程序（saveruser.php）是用来处理用户注册页面程序（regnew.php）中用户输入的资料，如果用户输入了合法的资料，就把它写入 MySQL 数据库中，如果用户输入了不合法的资料，比如 2 次输入密码不一致或者数据库中已经注册了此用户名称，则给出一些提示，让用户重新输入资料。其源程序如下：

saveruser.php

```
<html>
<head>
<title>xiaoduoduo 论坛>>--->>注册新用户</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
```

Green Apple Data Center

```
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#ECFFEC">
// 将用户输入的特殊字符转为 HTML 格式
<?
$username=htmlspecialchars($username);
$password=htmlspecialchars($password);
$repassword=htmlspecialchars($repassword);
$email=htmlspecialchars($email);
$homepage=htmlspecialchars($homepage);
$sex=htmlspecialchars($sex);
$comefrom=htmlspecialchars($comefrom);
$writename=htmlspecialchars($writename);
$regok=true;
// 查询数据库，判断输入的用户名是否已经存在
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
$text="select*from boarduser where username='$username'";
$rs=mysql_query($text,$conn);
$allready=mysql_fetch_row($rs);
if (! $allready){
        ;
}else{
        $regok=false;
        $tit="此用户名已经存在;<br>";
};
// 检查用户名是否为空
if ($username==""){
        $tit="用户名为空;<br>";
        $regok=false;
};
if ($password=="" || $password!=$repassword){
        $regok=false;
        $tit=$tit."密码错误!<br>";
};
?>
// 显示注册错误信息
<?if ($regok==false){
?>
<br>
```

Green Apple Data Center

```
<br>

有以下错误，以至注册不成功：<br>
<ul><?echo $tit?></ul>
      
<input type = "button" name = "Button" valu e= "&lt;&lt;返回" onclick = "document.location ='regnew.php ?username=<?echo $username;s?>&email=<?echo $email;?>&homepage=<?echo $homepage;?> &sex=<?echo $sex;?>&comefrom=<?echo $comefrom;?>&writename=<?echo $writename;?>'">
<br>
<br>
// 显示注册成功信息
<?}else{
$addtime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
$tex ="insert into boarduser(username,email,homepage,comefrom,sex,writename,addtime,password) values('$username','$email','$homepage','$comefrom','$sex','$writename','$addtime','$password')";
mysql_query($text,$conn);
$err=mysql_error();
echo $err;
?>
<br>
<br>
<table width = "80%" border = "1" cellspacing = "0" cellpadding = "3" bordercolorlight = "#FFFFFF" bordercolordark="#00B000" align="center">
  <tr bgcolor="#99FF99">
    <td colspan="2">xiaoduoduo 论坛注册新用户成功,请记住你的用户名与密码,谢谢你使用本论坛 </td>
  </tr>
  <tr>
    <td>用户名： </td>
    <td> <?echo $username;?></td>
  </tr>
  <tr>
    <td> 密码：</td>
    <td>********** </td>
  </tr>
  <tr>
    <td>E-mail：</td>
    <td> <?echo $email;?> </td>
  </tr>
  <tr>
    <td>个人主页：</td>
    <td> <?echo $homepage;?> </td>
  </tr>
  <tr>
    <td>性别：</td>
    <td><?echo $sex;?> </td>
  </tr>
  <tr>
```

Green Apple Data Center

```
        <td>来自何方：</td>
        <td> <?echo $comefrom;?> </td>
    </tr>
    <tr>
        <td>签名：</td>
        <td> <?echo $writename;?> </td>
    </tr>
    <tr>
        <td colspan="2"> <a href="plantop.php">按此进入论坛&gt;&gt;</a></td>
    </tr>
</table>
<br>
<br>
<br>
<br>
<?};?>
</body>
</html>
```

如图 7-3 所示的是显示注册成功的用户资料。

图 7-3 用户注册成功界面

7.3.2 用户资料修改

用户注册的资料并不是一成不变的，比如用户可能会经常修改自己的密码，可能要更新一下自己的主页或者电子邮件变动情况，这是一个好的 BBS 论坛必须具备的最基本的功能。用户资料修改由用户名输入界面程序（modify.php）、用户资料显示并修改程序（modifyuser.php）、用户资料修改处理程序（savemodifyuser.php）组成。

用户名输入界面程序（modify.php）非常简单，在浏览器里显示如图 7-4 所示，源程序如下：

Green Apple Data Center

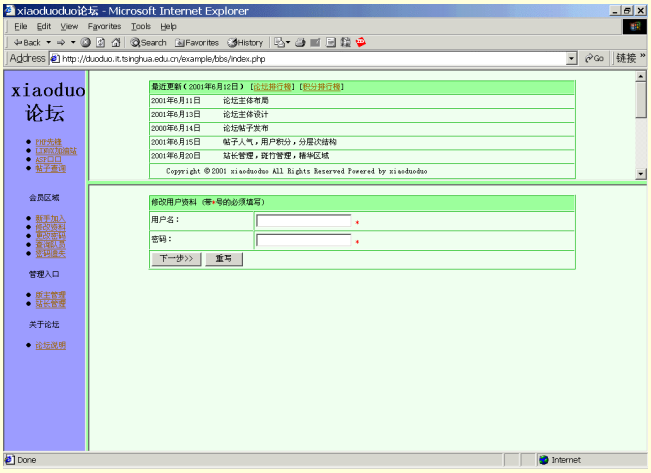


图 7-4　用户名输入界面

modify.php

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
</head>
<body bgcolor="#ECFFEC">
<form method = "post" action = "modifyuser.php" name = "regnew" onSubmit = "return check (document.regnew)">
<table width = "80%" border = "1" cellspacing = "0" cellpadding = "3" bordercolorlight = "#FFFFFF" bordercolordark="#00B000"
align="center">
    <tr bgcolor="#99FF99">
      <td colspan="2" height="25">修改用户资料 (带<font color="#FF0000">*</font>号的必须填写)</td>
    </tr>
    <tr>
      <td>用户名： </td>
      <td>
        <input type="text" name="username" maxlength="50" size="20">
        <font color="#FF0000">*</font> </td>
    </tr>
    <tr>
      <td> 密码：</td>
      <td>
```

Green Apple Data Center

```
            <input type="password" name="password" maxlength="50" size="20">
            <font color="#FF0000">*</font> </td>
        </tr>
        <tr>
          <td colspan="2">
            <input type="submit" name="Submit" value="下一步&gt;&gt;">
            <input type="reset" name="Submit2" value="  重写  ">
          </td>
        </tr>
    </table>
</form>
// 判断用户名和密码是否为空
<script languge="javascript" language="">
<!--
function check(the){
      var checkok=true;
      if( the.username.value=="" || the.password.value=="" ){
            alert("请把资料填写完整,谢谢!!!");
            checkok=false;
      };
      return checkok;
};
//-->
</script>
</body>
</html>
```

用户资料显示并修改程序（modifyuser.php**）主要作用是从** MySQL **数据库服务器中读入要修改用户的资料并显示出来以便修改。源程序如下：**

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
</head>

<body bgcolor="#ECFFEC">
```

Green Apple Data Center

```
<?
$username=htmlspecialchars($username);
$password=htmlspecialchars($password);
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
$text="select*from boarduser where username='$username' and password='$password'";
$rs=mysql_query($text,$conn);
$allready=mysql_fetch_row($rs);
if (! $allready){
    $tit="用户名或密码错误,以至操作不能进行!<br>";
}else{
    $tit="你的资料如下,修改后请按保存按钮";
    $email=$allready[3];
    $homepage=$allready[6];
    $sex=$allready[5];
    $comefrom=$allready[4];
    //$num=$allready[9];
    $writename=$allready[8];
};
?>
<? if (!$allready){?>
<ul>
<?echo $tit;?>
<br>
<br>
<input type="button" value="<<返回" onclick="document.location='modify.php'">
</ul>
<?}else{
?>
<script languge="javascript">
<!--
function check(the){
    var checkok=true;
    if( the.username.value=="" || the.password.value==""){
        alert("请把资料填写完整,谢谢!!!");
        checkok=false;
    };
    return checkok;
};
//-->
</script>
<form method = "post" action = "savemodify.php" name = "regnew" onSubmit = "return check(document.regnew)">
<table width = "80%" border = "1" cellspacing = "0" cellpadding = "3" bordercolorligh = "#FFFFFF" bordercolordark="#00B000"
align="center">
    <tr bgcolor="#99FF99">
      <td colspan="2"> <?echo $tit;?></td>
    </tr>
```

Green Apple Data Center

```
<tr>
  <td height="34">用户名： </td>
  <td height="34"> <?echo $username;?>
    <input type="hidden" name="username" value="<?echo $username;?>">
    <input type="hidden" name="password" value="<?echo $password;?>">
  </td>
</tr>
<tr>
  <td> 新密码：</td>
  <td>
    <input type="password" name="newpassword" maxlength="50" size="20">
    (可以不用填写,当然密码也不会改变哦) </td>
</tr>
<tr>
  <td>密码确认：</td>
  <td>
    <input type="password" name="repassword" maxlength="50" size="20">
    (可以不用填写,当然密码也不会改变哦) </td>
</tr>
<tr>
  <td>E-mail：</td>
  <td>
    <input type="text" name="email" maxlength="50" size="30" value="<?echo $email;?>">
  </td>
</tr>
<tr>
  <td>个人主页：</td>
  <td>
    <input type ="text" name = "homepage" maxlength =" 50" siz e= "30" value = "<?echo $homepage;?>">
  </td>
</tr>
<tr>
  <td>性别：</td>
  <td>
    <input type="radio" name="sex" value="男" <?if($sex!="女"){?>checked<?};?>>
    男   
    <input type="radio" name="sex" value="女" <?if($sex=="女"){?>checked<?};?>>
    女 </td>
</tr>
<tr>
  <td>来自何方：</td>
  <td>
    <input type = "text" name = "comefrom" maxlength = "50" size = "30" value = "<?echo $comefrom;?>">
  </td>
</tr>
<tr>
  <td>签名：</td>
```

Green Apple Data Center

```
        <td>
    <input type="text" name="writename" maxlength="255" size="40" value="<?echo $writename;?>">
        </td>
      </tr>
      <tr>
        <td colspan="2">
          <input type="submit" name="Submit" value="保存&gt;&gt;">
          <input type="reset" name="Submit2" value="  重写  ">
        </td>
      </tr>
    </table>
</form>
<p><?};?> </p>
</body>
</html>
```

如图 7-5 所示显示的是要修改用户 wangmenker 的一些资料。

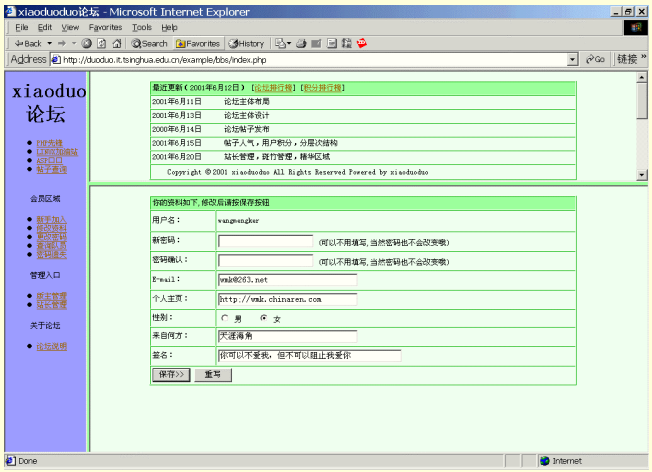


图 7-5　显示要修改的用户资料

用户资料修改处理程序（savemodifyuser.php）是对如图 7-5 所示的 modifyuser.php 界面用户重新输入的资料进行处理，如果输入的资料合法，则把它写入数据库，更新以前的资料；如果输入的资料不合法，比如 2 次输入的密码不一致，则给出提示信息。源程序如下：

savemodifyuser.php

```
<html>
<head>
<title>xiaoduoduo 论坛>>--->>注册新用户</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
```

Green Apple Data Center

```
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#ECFFEC">
<?
$username=htmlspecialchars($username);
$password=htmlspecialchars($password);
$repassword=htmlspecialchars($repassword);
$email=htmlspecialchars($email);
$homepage=htmlspecialchars($homepage);
$sex=htmlspecialchars($sex);
$comefrom=htmlspecialchars($comefrom);
$writename=htmlspecialchars($writename);
$regok=true;
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
$text="select*from boarduser where username='$username' and password='$password'";
$rs=mysql_query($text,$conn);
$allready=mysql_fetch_row($rs);
if (! $allready){
        $regok=false;
        $tit="用户名密码错误;<br>";
}else{
        ;
};
if ($username==""){
        $tit="用户名为空;<br>";
        $regok=false;
};
if ($newpassword!="" && $newpassword!=$repassword){
        $regok=false;
        $tit=$tit."密码错误!<br>";
}else{
        if ($newpassword!="" && $newpassword==$repassword){
                $password=$newpassword;
        };
};
?>
<?if ($regok==false){
?>
<br>
<br>
// 显示错误信息
有以下错误，以至修改资料不成功：<br>
<ul><?echo $tit?></ul>
```

Green Apple Data Center

```
      
<input type = "button" name = "Button" value = "&lt;&lt;返回" onclick = "document.location ='modifyuser.php?username=<?echo $username;s?>&password=<?echo $password;?>'">
<br>
<br>
// 更新数据库，并显示修改成功后的信息
<?}else{
//$addtime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
$text = "update boarduser set email = '$email', homepage = '$homepage', comefrom = '$comefrom', sex = '$sex', writename='$writename',password='$password' where username='$username' ";
mysql_query($text,$conn);
$err=mysql_error();
echo $err;
?>
<br>
<br>
<table width = "80%" border = "1" cellspacing = "0" cellpadding = "3" bordercolorlight = "#FFFFFF" bordercolordark="#00B000" align="center">
  <tr bgcolor="#99FF99">
    <td colspan="2">资料或者密码修改成功,请记住你的密码,谢谢你使用本论坛 </td>
  </tr>
  <tr>
    <td>用户名： </td>
    <td> <?echo $username;?></td>
  </tr>
  <tr>
    <td> 密码：</td>
    <td>********** </td>
  </tr>
  <tr>
    <td>E-mail：</td>
    <td> <?echo $email;?> </td>
  </tr>
  <tr>
    <td>个人主页：</td>
    <td> <?echo $homepage;?> </td>
  </tr>
  <tr>
    <td>性别：</td>
    <td><?echo $sex;?> </td>
  </tr>
  <tr>
    <td>来自何方：</td>
    <td> <?echo $comefrom;?> </td>
  </tr>
  <tr>
    <td>签名：</td>
```

Green Apple Data Center

```
        <td> <?echo $writename;?> </td>
    </tr>
    <tr>
        <td colspan="2"> <a href="plantop.php">按此进入论坛&gt;&gt;</a></td>
    </tr>
</table>
<br>
<br>
<br>
<br>
<?};?>
</body>
</html>
```

如图 7-6 所示的是显示修改成功后的用户资料。

图 7-6　显示修改成功后的用户资料

7.3.3　查询用户信息

用户经常在论坛上可能要查询别的用户信息，比如想访问某个用户的主页或者想知道他/她的电子邮件地址，就可以通过查询用户信息来得到所要的资料，查询用户信息包括对要查询的用户名输入界面程序（queryuser.php）、用户资料显示程序（requeryuser.php）。本论坛几乎是显示用户的全部资料（除用户密码外），读者可以修改这个论坛使其功能更为强大，比如只显示用户愿意公开的资料，这也当然要修改用户注册界面了。用户名输入界面程序（queryuser.php）十分简单，在浏览器中显示如图 7-7 所示，源程序如下：

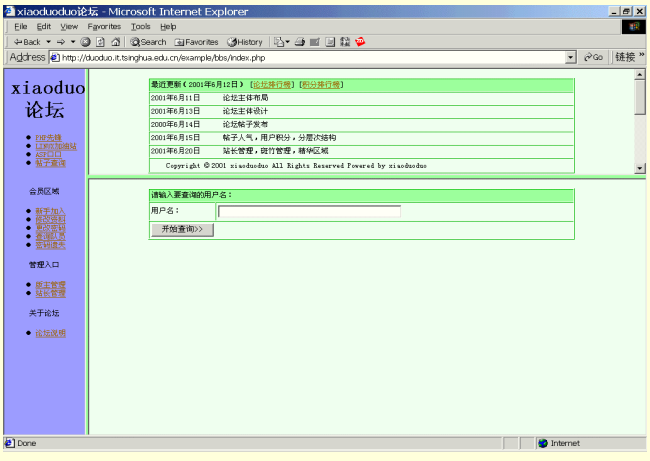


图 7-7　用户名输入界面

Green Apple Data Center

```
<html>
<head>
<title>查询用户资料</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#ECFFEC">
<script language="javascript">
<!--
function search(username){
url="requeryuser.php?username="+username;
open(url,"_self");
};
//-->
</script>
<form name="search" action="requeryuser.php">
   <table width = "80%" border = "1" cellspacing = "0" cellpadding = "3" bordercolorlight = "#FFFFFF" bordercolordark="#00B000"
align="center">
   <tr>
      <td colspan="2" bgcolor="#99FF99">请输入要查询的用户名：</td>
   </tr>
   <tr>
      <td>用户名： </td>
      <td valign="middle">
         <input type="text" name="username" size="40" maxlength="50">
      </td>
   </tr>
   <tr>
         <td colspan="2">
            <input type="submit" name="Submit" value="开始查询&gt;&gt;">
         </td>
   </tr>
</table></form>
</body>
</html>
```

Green Apple Data Center

用户资料显示程序（requeryuser.php）是从数据库中读出要查询的用户资料，并在浏览器里显示出来。源程序如下：

```
<html>
<head>
<title>查询用户资料</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#ECFFEC">
<?
$username=htmlspecialchars($username);
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
$text="select*from boarduser where username='$username'";
$rs=mysql_query($text,$conn);
$allready=mysql_fetch_array($rs);
if (! $allready){
        $tit="此用户不存在！";
        $username="";
//    显示查询的用户资料
}else{
        $tit="你要查询的用户信息如下：";
        $email=$allready[email];
        $homepage=$allready[homepage];
        $sex=$allready[sex];
        $comefrom=$allready[comefrom];
        $num=$allready[num];
};
?>
<table width="80%" border="1" cellspacing="0" cellpadding="3" bordercolorlight="#FFFFFF" bordercolordark="#00B000" align="center">
    <tr>
      <td colspan="2" bgcolor="#99FF99"><?echo $tit;?> <a href="queryuser.php">重新查询&gt;&gt;</a></td>
    </tr>
    <tr>
```

Green Apple Data Center

```
    <td>用户名： </td>
    <td> <?echo $username;?></td>
  </tr>
  <tr>
    <td>E-mail：</td>
    <td> <?echo $email;?> </td>
  </tr>
  <tr>
    <td>个人主页：</td>
    <td> <?echo $homepage;?> </td>
  </tr>
  <tr>
    <td>性别：</td>
    <td><?echo $sex;?> </td>
  </tr>
  <tr>
    <td>来自何方：</td>
    <td> <?echo $comefrom;?> </td>
  </tr>
  <tr>
    <td>个人积分</td>
    <td> <?echo $num;?> </td>
  </tr>
  <tr>
    <td colspan="2"><a href="queryuser.php">重新查询&gt;&gt;</a></td>
  </tr>
</table>
</body>
</html>
```

如图 7-8 所示的是查询出用户 wangmengker 的一些资料。

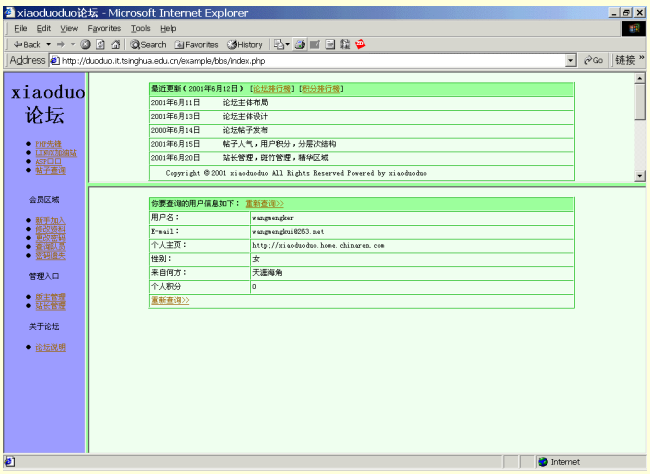


图 7-8　查询用户资料结果

用户服务程序最基本的功能就包括这些，还有一些功能比如用户密码丢失等功能详见本书附带的光盘，当然一个优秀的论坛程序功能当然是越强大越好，读者如果对此论坛程序感兴趣，可以对本书附带的光盘中的源

Green Apple Data Center

程序加以扩充和修改，使之具有更为强大的功能。

7.4 论坛版面程序

7.4.1 版面显示程序

版面显示程序由版面分页程序（list.php）和显示文章（showplan.php）等程序组成。

版面分页程序（list.php）是根据不同的 boardid（论坛版面号）和 page（该版面页号）显示该版面中所指定页的文章标题，如图 7-9 所示，在程序中通常都是像以下方式调用超级链接：

http://duoduo.it.tsinghua.edu.cn/example/bbs/list.php?boardid=1&page=2

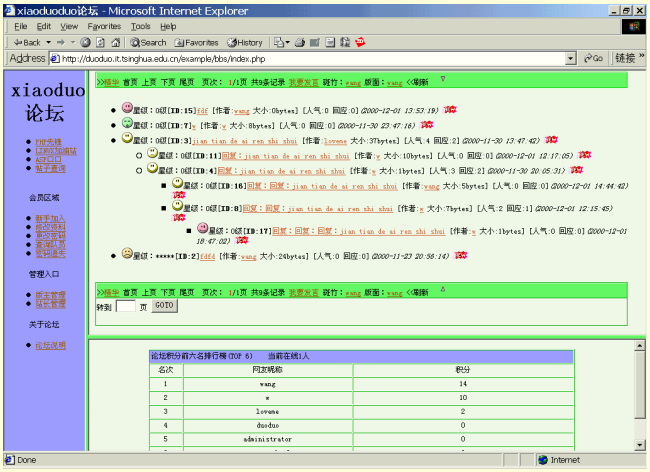


图 7-9 版面显示界面

上面超级链接表示要显示论坛版面号为 1 并且是该版面页数为 2 的文章标题，list.php 源程序如下：

list.php

```
<?include "showplan.php";?>
<html>
<head>
<title>Untitled Document</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<?
$boardid=htmlspecialchars ( $boardid );
if (!($boardid<>"") || $boardid<1){
        $boardid=1;
};
```

Green Apple Data Center

```
$conn = mysql_connect('localhost:3306','root','');
mysql_select_db('mysql');
$text = "select*from board where boardid=$boardid ";
$rs = mysql_query ($text,$conn);
$rom = mysql_fetch_row ($rs);
$err=mysql_error();
if (! $rom) {
                echo $err."<br>";
        echo "此版面不存在！！！！ ";
        }
else{
        $page=htmlspecialchars($page);
        if ($page==""){
                $page=1;
        };
    $boardname=$rom[1];
    $boardmaster=$rom[2];
    $masteremail=$rom[3];
        $text="select count(*) as planid from plan where boardid='$boardid'";
        $rs=mysql_query($text,$conn);
        $pagesize=30;
        $count=mysql_result($rs,"0","planid");
        if (!$count){
                $tit="此版面还没有帖子，你想成为第一个吗，赶快开始发言吧！";
                $pagecount=0;
        }
        else{
                $pagecount=ceil($count/$pagesize);
                $start=($page-1)*$pagesize;
        }
};

?>
</head>

<body bgcolor="#ECFFEC" topmargin="2">
<a name="top"></a>
<table width="100%" border="1" cellspacing="0" cellpadding="0" align="center" bordercolordark="#FFFFFF"
bordercolorlight="#00CC00">
   <tr bgcolor="#66FF66">
     <td nowrap colspan="2">&gt;&gt;<a href="glod.php?boardid=<?echo $boardid?>">精华</a>
       <?if ($page>1){?><a href="list.php?boardid=<?echo $boardid;?>&page=1">首页</a>
       <?}else{?>首页 <?};?><?if ($page>1){?>
   <a href="list.php?boardid=<?echo $boardid;?>&page=<?echo $page-1;?>">上页</a>
       <?}else{?>上页 <?};?><?if ($page<$pagecount){?>
   <a href="list.php?boardid=<?echo $boardid;?>&page=<?echo $page+1;?>">下页</a>
       <?}else{?>下页 <?};?><?if ($page<$pagecount){?>
```

Green Apple Data Center

```
        <a href="list.php?boardid=<?echo $boardid;?>&page=<?echo $pagecount;?>">尾页</a>
            <?}else{?>尾页 <?};?>  页次：
            <font color=red><?echo $page;?></font>/<?echo $pagecount;?>页
            共<?echo $count ;?>条记录
        <a href="addplan.php?boardid=<?echo $boardid;?>" target="BoardAnnounce">我要发言</a>
            斑竹：<a href="mailto:<? echo $masteremail?>"><? echo $boardmaster;?></a>
            版面：<a href="list.php?boardid=<? echo $boardid?>&page=<?echo $page;?>"><? echo $boardname;?></a>
            &lt;&lt; 刷 新   <a href="#bottom"><img src="images/arrow-down.gif" width="9" height="19" border="0"></a></td>
          </tr>
      </table>
      <?if (!$count){
       echo "<ul>".$tit."</ul>";}else{
          $text="select*from plan where boardid='$boardid' order by rootid desc,layer limit $start,$pagesize";
         showplan($text);
       };
       ?><br>
      <table width="100%" border="1" cellspacing="0" cellpadding="0" align="center" bordercolordark="#FFFFFF" bordercolorlight="#00CC00">
        <tr bgcolor="#66FF66">
          <td nowrap colspan="2">&gt;&gt;<a href="glod.php?boardid=<?echo $boardid?>">精华</a>
            <?if ($page>1){?><a href="list.php?boardid=<?echo $boardid;?>&page=1">首页</a>
            <?}else{?>首页 <?};?><?if ($page>1){?>
        <a href="list.php?boardid=<?echo $boardid;?>&page=<?echo $page-1;?>">上页</a>
            <?}else{?>上页 <?};?><?if ($page<$pagecount){?>
        <a href="list.php?boardid=<?echo $boardid;?>&page=<?echo $page+1;?>">下页</a>
            <?}else{?>下页 <?};?><?if ($page<$pagecount){?>
        <a href="list.php?boardid=<?echo $boardid;?>&page=<?echo $pagecount;?>">尾页</a>
            <?}else{?>尾页 <?};?>  页次：
            <font color=red><?echo $page;?></font>/<?echo $pagecount;?>页
            共<?echo $count ;?>条记录
        <a href="addplan.php?boardid=<?echo $boardid;?>" target="BoardAnnounce">我要发言</a>
            斑竹：<a href="mailto:<? echo $masteremail?>"><? echo $boardmaster;?></a>
            版面：<a href="list.php?boardid=<? echo $boardid?>&page=<?echo $page;?>"><? echo $boardname;?></a>
            &lt;&lt; 刷 新   <a href="#bottom"><img src="images/arrow-down.gif" width="9" height="19" border="0"></a></td>
          </tr>
        <tr>
          <td nowrap colspan="2">
            <form method="post" action="list.php" name="goto">
              转到
              <input type="text" name="page" maxlength="6" size="3">
              页
              <input type="submit" name="Submit" value="GOTO">
              <input type="hidden" name="boardid" value="<?echo $boardid;?>">
            </form>
          </td>
```

Green Apple Data Center

```
    </tr>
</table>
</body>
</html>
```

在 list.php 中调用了 showplan.php 程序，该程序主要作用是读取数据库中文章记录，显示文章标题及其相关的作者、回文次数等信息，其源程序如下：

showplan.php

```
<?
function showplan($text){
      $neworold=100;
    $qconn = mysql_connect('localhost:3306','root','');
    mysql_select_db('mysql');
    $rs=mysql_query ($text,$qconn);
    $i= 0 ;
    echo "<ul>";
    while ($rowt=mysql_fetch_array ($rs) ){
        if ( $rowt["planid"]!=$rowt["parentid"]) {
           if ($rowt["renum"]>=1 ){
                  if ($parentid[$i]!=$rowt["parentid"]){
                       for($j=$i;$j>=1;$j--){
                              if ($parentid[$j]!=$rowt["parentid"]){
                             echo "</ul>";//.$rowt["parentid"];
                             $i--;
                          }else{
                            // echo "</ul>" ;
                             $j=0;
                          };
                        };
                  }else{
                     $parentid[$i]=$rowt["parentid"];
                  };
              $parentid[$i+1] = $rowt["planid"];
              $i++;
              echo "<li>";//.$rowt["parentid"]."fdafdas";
              echo "<img src='images/mod" ;
              echo $rowt["emote"] ;
              echo ".gif'>" ;
            echo "星级：";
             if ($rowt["hote"]==0){echo "0 级";};
                  for($jm=$rowt["hote"];$jm>=1;$jm--){echo "*";};
              echo "[<B>ID:".$rowt["planid"]."</B>]"."<
              a href='replan.php?boardid=".$rowt["boardid"]."&planid=".$rowt["planid"]."'
              target='BoardAnnounce' >".$rowt["title"]."</a>"." [作者:"."
              <a href='mailto:".$rowt["email"]."'>".$rowt["username"]."</a>  大小:"
              .strlen($rowt["body"])."bytes]"." [人气:".$rowt["clicknum"]."  回应:".$
```

Green Apple Data Center

```
        rowt["renum"]."]"."<i>(".$rowt["addtime"].")</i>    ";

        $ctime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
            //echo checkdate($ctime);
        if ( $ctime - $rowt["addtime"]<=$neworold){
            echo "<img src='images/new.gif'>";
        };
          echo "</li><ul>";
    }else{
                    if ($parentid[$i]!=$rowt["parentid"]){
                for($j=$i;$j>=1;$j--){
                            if ($parentid[$j]!=$rowt["parentid"]){
                    echo "</ul>";//.$rowt["parentid"];
                    $i--;
                    }else{
                     // echo "</ul>" ;
                    $j=0;
                    };
                };
            }else{
               $parentid[$i]=$rowt["parentid"];
            };
            echo "<li>";
            echo "<img src='images/mod" ;
            echo $rowt["emote"] ;
            echo ".gif'>" ;
     echo "星级：";
       if ($rowt["hote"]==0){echo "0 级";};
            for($jm=$rowt["hote"];$jm>=1;$jm--){echo "*";};
            echo "[<B>ID:".$rowt["planid"]."</B>]"."
            <a href='replan.php?boardid=".$rowt["boardid"]."&planid=".$rowt["planid"]."'
              target='BoardAnnounce' >".$rowt["title"]."</a>"." [作者:"."
              <a href='mailto:".$rowt["email"]."'>".$rowt["username"]."</a>  大小:"
              .strlen($rowt["body"])."bytes]"." [人气:".$rowt["clicknum"]."  回应:".
              $rowt["renum"]."]"."<i>(".$rowt["addtime"].")</i>    ";

            $ctime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
            if ( $ctime - $rowt["addtime"]<$neworold){
               echo "<img src='images/new.gif'>";
            };
       echo "</li>";
    };
}else{
    if ($i==1){
        echo "</ul>";
        $i=0;
    }else{
```

Green Apple Data Center

```
                    for($j=$i;$j>=1;$j--){
                        //echo "fdafdasfdasfdsafdafdsa";
                      if ($parentid[$j]!=$rowt["parentid"]){
                        echo "</ul>";//.$rowt["parentid"];
                        $i--;
                      }else{
                       // echo "</ul>" ;
                        $j=0;
                      };
                    };
                $i=0;
            };
            echo "<li>";
            echo "<img src='images/mod" ;
            echo $rowt["emote"] ;
            echo ".gif'>" ;
             echo "星级：";
               if ($rowt["hote"]==0){echo "0 级";};
                    for($jm=$rowt["hote"];$jm>=1;$jm--){echo "*";};
            echo "[<B>ID:".$rowt["planid"]."</B>]"."
            <a href='replan.php?boardid=".$rowt["boardid"]."&planid=".$rowt["planid"]."'
            target='BoardAnnounce' >".$rowt["title"]."</a>"." [作者:"."
            <a href='mailto:".$rowt["email"]."'>".$rowt["username"]."</a>  大小:"
            .strlen($rowt["body"])."bytes]"." [人气:".$rowt["clicknum"]."  回应:"
            .$rowt["renum"]."]"."<i>(".$rowt["addtime"].")</i>    ";

            $ctime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
            if ( $ctime - $rowt["addtime"]<$neworold){
                echo "<img src='images/new.gif'>";
            };
            echo "</li>";
            if ($rowt["renum"]>=1 ){
                $parentid[$i+1] = $rowt["planid"];
                $i++;
                echo "<ul>";
            };
            //$i=$i+1;
        };
    };
  // $i=$i+($i+1);
    for($j=0;$j<=$i;$j++){
        echo "</ul>";
    };
};
?>
```

Green Apple Data Center

7.4.2 发表文章程序

发表文章程序包括发表文章页面程序（addplan.php）和处理发表文章程序（saveplan.php）。发表文章页面程序（addplan.php）主要用于写文章，这个界面应该加入一些图片制作的稍漂亮一些，如图 7-10 所示程序中加入了许多表情图片，使用户发表文章时可以根据自己的心情选择表情图片。

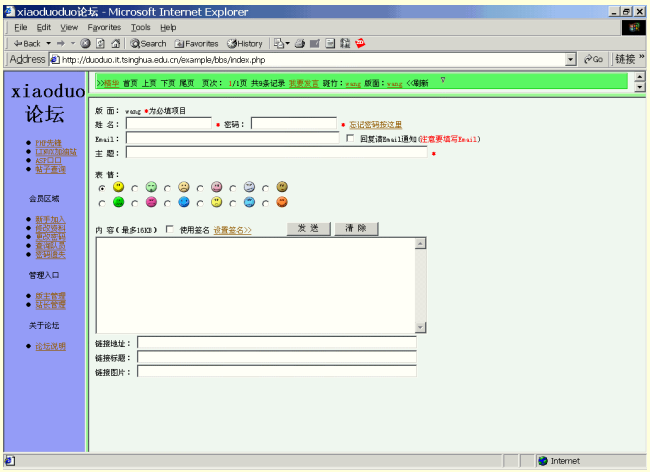


图 7-10 发表文章界面

源程序如下：

```
<html>
<head>
<title>新加帖子</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#ECFFEC">

<?
$boardid=htmlspecialchars ( $boardid );
if (!($boardid<>"") || $boardid<1){
        $boardid=1;
};
$conn = mysql_connect('localhost:3306','root','');
mysql_select_db('mysql');
$text = "select*from board where boardid=$boardid ";
$rs = mysql_query ($text,$conn);
```

Green Apple Data Center

```
$rom = mysql_fetch_row ($rs);
$err=mysql_error();
if (! $rom) {
                echo $err."<br>";
               echo "此版面不存在！！！！ ";
               }
else{

        $boardname=$rom[1];
};

?>
<script language="javascript">
<!--
function chkSubmit(){
var subok=true;
if (document.planform.username.value=="" || document.planform.password.value=="" || document. planform.title.value==""){
        alert("请把数据输入完整,带*号的是必填项!");
        subok=false;
}
return subok;
}
//-->
</script>
<form action="saveplan.php" method="POST" name="planform" onsubmit="return chkSubmit()">
   <input type="hidden" name="boardid" value="<?echo $boardid?>">
   <span class=smallFont>版 面： <? echo $boardname; ?> <font color=red><strong>*</strong></font>为必填项目</span><br>
   <input type="hidden" name="boardname" value="<? echo $boardname?>">
   <span class=smallFont>姓 名：</span>
   <input class = "smallInput" name = "username" size = "18" maxlength = "50" value = "<?echo $username;?>">
   <font color=red><strong>*</strong></font> <span class=smallFont>密码： </span>
   <input class="smallInput" name="password" size="18" maxlength="10" type="password">
   <span class=smallFont><font color=red><strong>*</strong></font></span>
  <a href="forgetpassword.php" target="_blank">忘记密码按这里</a><br>
   <span class=smallFont>Email：</span>
   <input class="smallInput" name="email" size="47" maxlength="40" value="">
   <input type=checkbox    value="是" name=retoemail>
   回复请 Email 通知(<font color="#ff0000">注意要填写 Email</font>)<br>
   <span class=smallFont>主 题：</span>
   <input class="smallInput" name="title" size="67" maxlength="255" value="">
   <font color=red><strong>*</strong></font>
   <p><span class=smallFont>表 情：</span><br>
      <input type="radio" value="1" name="emote" checked>
      <img src="images/mod1.gif"
               width="20" height="20" alt="">
      <input type="radio" value="2" name="emote" >
      <img src="images/mod2.gif"
```

Green Apple Data Center

```
            width="20" height="20" alt="">
        <input type="radio" value="3" name="emote" >
        <img src="images/mod3.gif"
            width="20" height="20" alt="">
        <input type="radio" value="4" name="emote" >
        <img src="images/mod4.gif"
            width="20" height="20" alt="">
        <input type="radio" value="5" name="emote" >
        <img src="images/mod5.gif"
            width="20" height="20" alt="">
        <input type="radio" value="6" name="emote" >
        <img src="images/mod6.gif"
            width="20" height="20" alt=""><br>
        <input type="radio" value="7" name="emote" >
        <img src="images/mod7.gif"
            width="20" height="20" alt="">
        <input type="radio" value="8" name="emote" >
        <img src="images/mod8.gif"
            width="20" height="20" alt="">
        <input type="radio" value="9" name="emote" >
        <img src="images/mod9.gif"
            width="20" height="20" alt="">
        <input type="radio" value="10" name="emote" >
        <img src="images/mod10.gif"
            width="20" height="20" alt="">
        <input type="radio" value="11" name="emote" >
        <img src="images/mod11.gif"
            width="20" height="20" alt="">
        <input type="radio" value="12" name="emote" >
        <img src="images/mod12.gif"
            width="20" height="20" alt=""></p>

    <p>
    <table width="450" cellspacing="0" cellpadding="0" border=0>
    <tr>
      <td width="60%"><span class=smallFont>内 容（最多 16KB）</span>
          <input type=checkbox   value="是" name=writename>
          <span class=smallFont>使用签名 <a href="modify.php" target="_blank">设置签名>></a></span></td>
      <td width="40%" align="right"><input class="buttonface" type="submit" value=" 发 送 " id=submit1 name=submit1> <input
class="buttonface" type="reset" value=" 清 除 " id=reset1 name=reset1></td>
    </tr>
  </table>
    <textarea class="smallarea" cols="72" name="body" rows="10"></textarea><br>
    <span class=smallFont>链接地址： </span>
    <input class="smallInput" name="link" size="62">
    <br>
    <span class=smallFont>链接标题： </span>
```

Green Apple Data Center

```
    <input class="smallInput" name="linktitle" size="62" value="">
    <br>
    <span class=smallFont>链接图片： </span>
    <input class="smallInput" name="picturelink" size="62">
</form>
<div align=center> <br>
</div>
<br>
</body>
</html>
```

处理发表文章程序（saveplan.php**）主要是把** addplan.php **页面用户提交发表的文章写入数据库。源程序如下：**

```
saveplan.php
<? include "plansave.php" ;?>
<html>
<head>
<title>新加帖子</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#ECFFEC">

<?

$boardid=htmlspecialchars ( $boardid );
if (!($boardid<>"") || $boardid<1){
            $boardid=1;
};
$text = "select*from board where boardid=$boardid ";
$rs = mysql_query ($text,$conn);
$rom = mysql_fetch_row ($rs);
$err=mysql_error();
if (! $rom) {
                    echo $err."<br>";
            echo "此版面不存在！！！！ ";
            }
else{
```

Green Apple Data Center

```
        $boardname=$rom[1];
            $username=htmlspecialchars($username);
            $password=htmlspecialchars($password);
            setcookie("username",$username);
            $email=htmlspecialchars($email);
            $title=htmlspecialchars($title);
            $retoemail=htmlspecialchars($retoemail);
            $writename=htmlspecialchars($writename);
            $emote=htmlspecialchars($emote);
            $link=htmlspecialchars($link);
            $linktitle=htmlspecialchars($linktitle);
            $picturelink=htmlspecialchars($picturelink);
            $text="select*from boarduser where username='$username' and password='$password'";
            $rs=mysql_query($text,$conn);
            $check=mysql_fetch_row($rs);
            if ($writename=="是"){
                    $writename=$check[8];
                    //echo $writename;
            };

            $err=mysql_error();
            $body=htmlspecialchars($body);
            $addtime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
            echo $err;
            if (! $check || $username=="" || $password=="" ||$title==""){
                    if (! $check){
                            $tit="有以下错误,以至帖子不能提交成功<br>用户名密码错误!  申请用户<a  href='addnew.php'>
按这里</a><br>";
                            $username="";
                    }else{          $tit="数据填写不完整!";          };

    ?>

    <script language="javascript">
    <!--
    function chkSubmit(){
    var subok=true;
    if (document.planform.username.value=="" || document.planform.password.value=="" || document. planform.title.value==""){
            alert("请把数据输入完整,带*号的是必填项!");
            subok=false;
    }
    return subok;
    }
    //-->
    </script>
    <span class=smallFont><? echo $tit; ?></span>
    <form action="saveplan.php" method="POST" name="planform" onsubmit="return chkSubmit()">
```

Green Apple Data Center

```
<input type="hidden" name="boardid" value="<?echo $boardid?>">
<span class=smallFont>版面：<? echo $boardname; ?> <font color=red><strong>*</strong></font>为必填项目</span><br>
<input type="hidden" name="boardname" value="<? echo $boardname?>">
<span class=smallFont>姓 名：</span>
<input class="smallInput" name="username" size="18" maxlength="50" value="<?  echo $username?>">
<font color=red><strong>*</strong></font> <span class=smallFont>密码： </span>
<input class="smallInput" name="password" size="18" maxlength="10" type="password">
<span class=smallFont><font color=red><strong>*</strong></font></span> <a href=
"forgetpassword.php" target="_blank">忘记密码按这里</a><br>
<span class=smallFont>Email：</span>
<input class="smallInput" name="email" size="47" maxlength="40" value="<?  echo $email?>">
<input type=checkbox  value="是" name=retoemail>
回复请 Email 通知(<font color="#ff0000">注意要填写 Email</font>)<br>
<span class=smallFont>主 题：</span>
<input class="smallInput" name="title" size="67" maxlength="255" value="<?  echo $title?>">
<font color=red><strong>*</strong></font>
<p><span class=smallFont>表 情：</span><br>
  <input type="radio" value="1" name="emote" checked>
  <img src="images/mod1.gif"
      width="20" height="20" alt="">
  <input type="radio" value="2" name="emote" >
  <img src="images/mod2.gif"
      width="20" height="20" alt="">
  <input type="radio" value="3" name="emote" >
  <img src="images/mod3.gif"
      width="20" height="20" alt="">
  <input type="radio" value="4" name="emote" >
  <img src="images/mod4.gif"
      width="20" height="20" alt="">
  <input type="radio" value="5" name="emote" >
  <img src="images/mod5.gif"
      width="20" height="20" alt="">
  <input type="radio" value="6" name="emote" >
  <img src="images/mod6.gif"
      width="20" height="20" alt=""><br>
  <input type="radio" value="7" name="emote" >
  <img src="images/mod7.gif"
      width="20" height="20" alt="">
  <input type="radio" value="8" name="emote" >
  <img src="images/mod8.gif"
      width="20" height="20" alt="">
  <input type="radio" value="9" name="emote" >
  <img src="images/mod9.gif"
      width="20" height="20" alt="">
  <input type="radio" value="10" name="emote" >
  <img src="images/mod10.gif"
      width="20" height="20" alt="">
```

Green Apple Data Center

```
        <input type="radio" value="11" name="emote" >
        <img src="images/mod11.gif"
                width="20" height="20" alt="">
        <input type="radio" value="12" name="emote" >
        <img src="images/mod12.gif"
                width="20" height="20" alt=""></p>

      <p>
      <table width="450" cellspacing="0" cellpadding="0" border=0>
      <tr>
        <td width="60%"><span class=smallFont>内 容（最多 16KB）</span>
            <input type=checkbox    value="是" name=writename>
            <span class=smallFont>使用签名 <a href="modify.php" target="_blank">设置签名>></a></span></td>
        <td width="40%" align="right"><input class="buttonface" type="submit" value=" 发 送 " id=submit1 name=submit1> <input
class="buttonface" type="reset" value=" 清 除 " id=reset1 name=reset1></td>
      </tr>
    </table>
      <textarea class="smallarea" cols="72" name="body" rows="10"><?    echo $body?></textarea><br>
      <span class=smallFont>链接地址： </span>
      <input class="smallInput" name="link" size="62" value="<?    echo $link?>">
      <br>
      <span class=smallFont>链接标题： </span>
      <input class="smallInput" name="linktitle" size="62" value="<?    echo $linktitle?>">
      <br>
      <span class=smallFont>链接图片： </span>
      <input class="smallInput" name="picturelink" size="62" value="<?    echo $picturelink?>">
    </form>
    <div align=center> <br>
    </div>
    <br>
    <?
    }else{
    if ($emote>12 || $emote<1){
            $emote=1;
    };
    $addtime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
    //echo $addtime;
    saveplan($boardid,$username,$email,$title,$body,$link,$picturelink,$linktitle,$emote,$writename,$addtime,1,$retoemail,$planid);
    ?>主题：<? echo $title ; ?><hr>
    <? echo $username ?> 于 <? echo $addtime ?> 加贴在 <? echo $boardname ?> <br>
    <hr>
    <?addusernum($username);
    if    ($body==""){
          echo "无内容";}
    else{
    echo "内容（".strlen($body)."bytes）:<BR><ul>";
    $Tbody=str_replace("\n","<br>",$body);
```

Green Apple Data Center

```
echo $Tbody;
echo "</ul>";
}?>        [<a href = "addplan.php?boarded = <?echo $boardid; ?>">继续发帖子</a>][<a href="list.php?boardid=<?echo $boardid;?>&amp;page=1" target="BoardList">刷新版面</a>]
<br>
<br><?
};};
?>
<p> </p><p> </p></body>
</html>
```

如图 7-11 所示的是该程序把发表文章的作者、发表时间、发表版面、文章大小等信息写入数据库后并把处理后的结果显示在浏览器中。

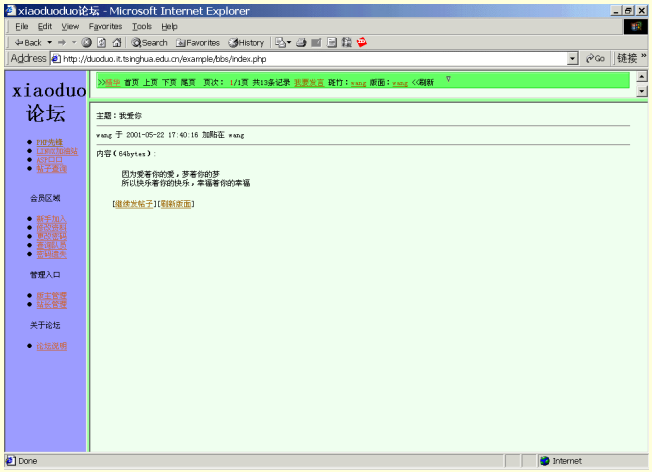


图 7-11　显示文章发表信息

在这个程序的第一行调用了 plansave.php 文件，这个文件主要是自定义了一些处理函数，由于还有许多页面程序要用到相同的数据处理，为了使编程方便，所以自定义一些函数存放在这个文件里，需要时调用这个文件就行了。其源程序如下：

```
plansave.php
<?
$conn = mysql_connect('localhost:3306','root','');
mysql_select_db('mysql');

function
saveplan($boardid,$username,$email,$title,$body,$link,$picturelink,$linktitle,$emote,$writename,$addtime,$layer,$retoemail){
        $Tconn = mysql_connect('localhost:3306','root','');
        mysql_select_db('mysql');
      //$text="insert into plan (boardid,username,email,title,body,link,picture,linktitle,emote,writename, addtime,retoemail) values ($boardid,'$username','$email','$title','$body','$link','$picturelink','$linktitle','$emote', '$writename',$addtime,'$retoemail')";
    //echo $text;
    $text="INSERT INTO plan (boardid, title, body, username, email, link, picture, writename, linktitle, emote, retoemail, addtime) VALUES( '$boardid', '$title', '$body', '$username', '$email', '$link', '$picturelink', '$writename', '$linktitle', '$emote', '$retoemail', '$addtime' )";
    mysql_query($text,$Tconn);
```

Green Apple Data Center

```
            //$er = mysql_error ();
                //echo $er;
    $planid=getplanid($boardid,$username,$email,$title,$body,$link,$picturelink,$linktitle,$emote,$writename,$addtime);
                updateplan($planid,$layer,$planid,$planid);
    };

    function updateplan($planid,$layer,$rootid,$parentid){
                $mconn = mysql_connect('localhost:3306','root','');
                mysql_select_db('mysql');
                $text="update plan set rootid='$rootid',layer='$layer',parentid='$parentid' where planid='$planid' ";
                return mysql_query($text,$mconn);
    };

    function getplanid($boardid,$username,$email,$title,$body,$link,$picturelink,$linktitle,$emote,$writename,$addtime){
                $aconn = mysql_connect('localhost:3306','root','');
                mysql_select_db('mysql');
                $text="select*from plan where boardid='$boardid' and username='$username' and title='$title' and email='$email' and
body='$body' and link='$link' and picture='$picturelink' and linktitle='$linktitle' and emote='$emote' and addtime='$addtime'";
                $rs=mysql_query($text,$aconn);
                $err=mysql_error();
                echo $err;
                $wrs=mysql_fetch_row($rs);

                $planid=$wrs[0];
                //echo    $planid ;
                return $planid;
    }
    function addclicknum($planid,$clicknum){
                //echo $clicknum;
                $hconn = mysql_connect('localhost:3306','root','');
                mysql_select_db('mysql');
                $text="update plan set clicknum='$clicknum' where planid='$planid' ";
                return mysql_query($text,$hconn);
    };

    function addusernum($username){
                $uconn = mysql_connect('localhost:3306','root','');
                mysql_select_db('mysql');
                $text="update boarduser set num=boarduser.num+2 where username='$username' ";
                $err=mysql_error();
                //echo $err.$username;
                return mysql_query($text,$uconn);
    };

    function
resaveplan($boardid,$username,$email,$title,$body,$link,$picturelink,$linktitle,$emote,$writename,$addtime,$layer,$retoemail,$rootid,$p
arentid){
```

Green Apple Data Center

```
        $lconn = mysql_connect('localhost:3306','root','');
        mysql_select_db('mysql');
                //echo $rootid;
                //echo $layer;
        $text="select*from plan where boardid='$boardid' and rootid='$rootid' and layer>='$layer' order by layer";
                $rs=mysql_query($text,$lconn);
                $err=mysql_error();
                echo $err;
                //$mt=mysql_fetch_row($rs);
                //echo $text;
                while ($layerow=mysql_fetch_array ($rs)){
                        //echo "fjdklasjfkldjasfldjasklf";
                        $planid=$layerow["planid"];
                        //echo $planid;
                        $text="update plan set layer=plan.layer+1 where planid='$planid'";
                        mysql_query($text,$lconn);
                };
        //echo $text;
       $text="INSERT INTO plan (boardid, title, body, username, email, link, picture, writename, linktitle, emote, retoemail, addtime) VALUES( '$boardid', '$title', '$body', '$username', '$email', '$link', '$picturelink', '$writename', '$linktitle', '$emote', '$retoemail', '$addtime' )";
        mysql_query($text,$lconn);
        //$er = mysql_error ();
        //echo $er;
        $planid=getplanid($boardid,$username,$email,$title,$body,$link,$picturelink,$linktitle,$emote,$writename,$addtime);
                //echo #layer;
        updateplan($planid,$layer,$rootid,$parentid);
        reupdateplan($parentid);
          retoemail($parentid,$planid);
};

function reupdateplan($planid){
        $mconn = mysql_connect('localhost:3306','root','');
        mysql_select_db('mysql');
        $text="update plan set renum=plan.renum+1    where planid='$planid' ";
        return mysql_query($text,$mconn);
};
function retoemail($parentid,$planid){
        $mailconn = mysql_connect('localhost:3306','root','');
        mysql_select_db('mysql');
          $text="select*from plan where planid='$parentid'";
          $rs=mysql_query($text,$mailconn);
          $rot=mysql_fetch_row($rs);
          //echo $rot[15];
          if ($rot[15]=="是" || $planid==$parentid){
                $text="select*from plan where planid='$planid'";
                $rss=mysql_query($text,$mailconn);
```

Green Apple Data Center

```
                $srom=mysql_fetch_row($rss);
                $username=$srom[6];
                $title=$srom[4];
                $body=$srom[5];
                $addtime=$srom[16];
                mail($rot[7],$title,$body.$addtime."用户名：".$username);
            }


};
function decplan($planid){
        $mconn = mysql_connect('localhost:3306','root','');
        mysql_select_db('mysql');
        $text="update plan set renum=plan.renum-1    where planid='$planid' ";
        return mysql_query($text,$mconn);
};
function addglod($username){
        $uconn = mysql_connect('localhost:3306','root','');
        mysql_select_db('mysql');
        $text="update boarduser set num=boarduser.num+10 where username='$username' ";
        $err=mysql_error();
        //echo $err.$username;
        return mysql_query($text,$uconn);
};
function decusernum($username){
        $uconn = mysql_connect('localhost:3306','root','');
        mysql_select_db('mysql');
        $text="update boarduser set num=boarduser.num-4 where username='$username' ";
        $err=mysql_error();
        //echo $err.$username;
        return mysql_query($text,$uconn);
};

function gotoemail($to,$planid){

        $mailconn = mysql_connect('localhost:3306','root','');
        mysql_select_db('mysql');
          $text="select*from plan where planid='$planid'";
          $rss=mysql_query($text,$mailconn);
          $srom=mysql_fetch_row($rss);
          $username=$srom[6];
          $title=$srom[4];
          $body=$srom[5];
          $addtime=$srom[16];
          mail($to,$title,$body.$addtime."用户名：".$username);

};
```

Green Apple Data Center

```
?>
```

7.4.3 显示文章内容程序

显示文章内容程序由一个程序 replan.php **实现，当然它也调用了** plansave.php **和** showplan.php **用于显示文章内容，如图** 7-12 **所示。用户可以根据这篇文章进行文章的转发、回复等操作。**

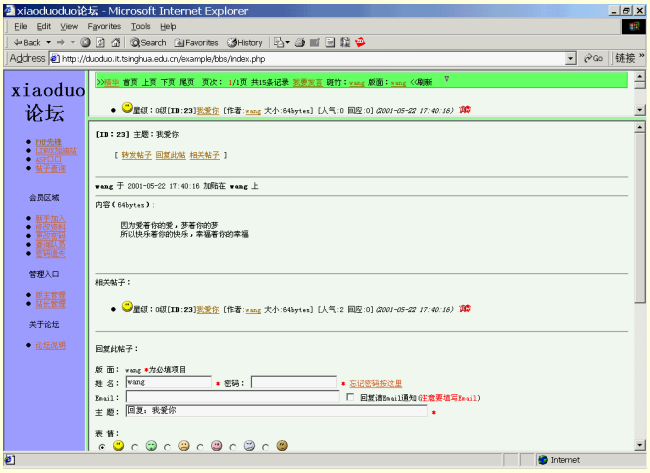


图 7-12 **显示文章内容**

```
replan.php
<? include "plansave.php";?>
<? include "showplan.php"; ?>
<?$ue=$username;
$boardid=htmlspecialchars ( $boardid );
if (!($boardid<>"") || $boardid<1){
        $boardid=1;
};
$conn = mysql_connect('localhost:3306','root','');
mysql_select_db('mysql');
$text = "select*from board where boardid=$boardid ";
$rs = mysql_query ($text,$conn);
$rom = mysql_fetch_row ($rs);
$err=mysql_error();
if (! $rom) {
                echo $err."<br>";
              echo "此版面不存在 !!!! ";
              }
else{
         $boardname=$rom[1];
};
$text="select*from plan where planid='$planid'";
$rs=mysql_query($text,$conn);
$srom=mysql_fetch_row($rs);
$err=mysql_error();
if (! $srom){
        echo $err."<br>";
```

Green Apple Data Center

```
        echo "此帖子已经不存在!!!!";
}else{
        $username=$srom[6];
        $title=$srom[4];
        $body=$srom[5];
        $addtime=$srom[16];
        $link=$srom[8];
        $email=$srom[7];
        $linktitle=$srom[13];
        $picturelink=$srom[9];
        $clicknum=$srom[11];
        $writename=$srom[12];
        //echo $clicknum."fdasfdsa";
        addclicknum($planid,$clicknum+1);
        $rootid=$srom[2];
        $layer=$srom[3];
};
?>
<html>
<head>
<title>回复帖子</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#ECFFEC">
<p>
   <script language="javascript">
<!--
function chkSubmit(){
var subok=true;
if (document.planform.username.value=="" || document.planform.password.value=="" || document.planform.title.value==""){
        alert("请把数据输入完整,带*号的是必填项!");
        subok=false;
}
return subok;
}
//-->
</script>
```

Green Apple Data Center

```
   <b> [ID：<? echo $planid ; ?>]</b>  主题：<? echo $title ; ?></p>
<p>     [ <a href="#retoemail">转发帖子</a> <a href="#reply">回复此帖</a>
   <a href="#xiangguan">相关帖子</a> ] <br>
</p>
<hr>
<b><? echo $username; ?></b>  于  <? echo $addtime ; ?>  加贴在  <b><? echo $boardname ; ?></b>  上<br>
<hr>
<? if    ($body==""){
        echo "无内容";}
else{
echo "内容（".strlen($body)."bytes）:<BR><ul>";
$Tbody=str_replace("\n","<br>",$body);
echo $Tbody;
echo "</ul>";
}?>
<br>
<div align="center"> <?
if ($writename!=""){
echo "<b>";
echo $writename;
echo "</b><br><br>";
};
if ($picturelink!=""){
        echo "<img src='".$picturelink."' alt="".$linktitle.">";
};
echo "<br>";
if ($link!=""){
        echo "<a href='".$link."' target='_blank'>";
        if ($linktitle!=""){
                echo $linktitle;
        }else{
                echo $link;
        };
        echo "</a>";
}else{
        echo $linktitle;
};
?>
   <div align="center"></div>
</div>
<hr>
<a name="xiangguan">相关</a>帖子：  <br>
<?
$text="select*from plan where boardid='$boardid' and rootid='$rootid' order by layer ";
showplan($text);
?>
<hr>
```

Green Apple Data Center

```
<br>

回复此帖子：
<br>
<form action="resaveplan.php" method="POST" name="planform" onsubmit="return chkSubmit()">
  <input type="hidden" name="boardid" value="<?echo $boardid;?>">
  <span class=smallFont><a name="reply">版 面</a>：<? echo $boardname; ?> <font color=red><strong>*</strong></font>为必填项目</span>
  <input type="hidden" name="rootid" value="<? echo $rootid;?>">
  <input type="hidden" name="layer" value="<? echo $layer+1;?>">
  <input type="hidden" name="planid" value="<? echo $planid;?>">
  <br>
  <input type="hidden" name="boardname" value="<? echo $boardname;?>">
  <span class=smallFont>姓 名：</span>
  <input class="smallInput" name="username" size="18" maxlength="50" value="<?echo $ue;?>">
  <font color=red><strong>*</strong></font> <span class=smallFont>密码： </span>
  <input class="smallInput" name="password" size="18" maxlength="10" type="password">
  <span class=smallFont><font color=red><strong>*</strong></font></span> <a href="forgetpassword.php" target="_blank">忘记密码按这里</a><br>
  <span class=smallFont>Email：</span>
  <input class="smallInput" name="email" size="47" maxlength="40" value="">
  <input type=checkbox  value="是" name=retoemail>
  回复请 Email 通知(<font color="#ff0000">注意要填写 Email</font>)<br>
  <span class=smallFont>主 题：</span>
  <input class="smallInput" name="title" size="67" maxlength="255" value="回复：<?echo $title;?>">
  <font color=red><strong>*</strong></font>
  <p><span class=smallFont>表 情：</span><br>
    <input type="radio" value="1" name="emote" checked>
    <img src="images/mod1.gif"
        width="20" height="20" alt="">
    <input type="radio" value="2" name="emote" >
    <img src="images/mod2.gif"
        width="20" height="20" alt="">
    <input type="radio" value="3" name="emote" >
    <img src="images/mod3.gif"
        width="20" height="20" alt="">
    <input type="radio" value="4" name="emote" >
    <img src="images/mod4.gif"
        width="20" height="20" alt="">
    <input type="radio" value="5" name="emote" >
    <img src="images/mod5.gif"
        width="20" height="20" alt="">
    <input type="radio" value="6" name="emote" >
    <img src="images/mod6.gif"
        width="20" height="20" alt=""><br>
```

Green Apple Data Center

```
        <input type="radio" value="7" name="emote" >
        <img src="images/mod7.gif"
             width="20" height="20" alt="">
        <input type="radio" value="8" name="emote" >
        <img src="images/mod8.gif"
             width="20" height="20" alt="">
        <input type="radio" value="9" name="emote" >
        <img src="images/mod9.gif"
             width="20" height="20" alt="">
        <input type="radio" value="10" name="emote" >
        <img src="images/mod10.gif"
             width="20" height="20" alt="">
        <input type="radio" value="11" name="emote" >
        <img src="images/mod11.gif"
             width="20" height="20" alt="">
        <input type="radio" value="12" name="emote" >
        <img src="images/mod12.gif"
             width="20" height="20" alt=""></p>

      <p>
      <table width="450" cellspacing="0" cellpadding="0" border=0>
      <tr>
        <td width="60%"><span class=smallFont>内 容（最多 16KB）</span>
            <input type=checkbox   value="是" name=writename>
            <span class=smallFont>使用签名 <a href="modify.php" target="_blank">设置签名>></a></span></td>
        <td width="40%" align="right"><input class="buttonface" type="submit" value=" 发 送 " id=submit1 name=submit1> <input
class="buttonface" type="reset" value=" 清 除 " id=reset1 name=reset1></td>
      </tr>
    </table>
      <textarea class="smallarea" cols="72" name="body" rows="10"></textarea><br>
      <span class=smallFont>链接地址： </span>
      <input class="smallInput" name="link" size="62">
      <br>
      <span class=smallFont>链接标题： </span>
      <input class="smallInput" name="linktitle" size="62" value="">
      <br>
      <span class=smallFont>链接图片： </span>
      <input class="smallInput" name="picturelink" size="62">
    </form>

    <hr><br><b>
    <a name="retoemail">帖子</a>转发：</b><br>
    <form method="post" action="retoemail.php?plianid=<?echo $planid; ?>" name="retoemail" target="_blank">
      E-mail 地址：
      <input type="text" name="textfield" size="20" maxlength="50">
```

Green Apple Data Center

```
    <input type="submit" name="submit" value="转发">
</form>
</body>
</html>
```

此外用户还可以阅读精华区的文章内容，精华区的文章都是各版面版主把一些优秀的文章加入精华区，普通用户只有发表和浏览文章的权限，版主则有删除文章和把文章加入精华区的权限（下一节将作介绍）。本论坛用户可以点击“精华”超级链接访问精华区的文章，是通过 gold.php 程序文件实现的。这个程序和版面显示程序 list.php 所用的 PHP 与 MySQL 技术都极其相似，这里就不再赘述了，有兴趣的读者可参考光盘中的源程序。

7.5 管理员程序

7.5.1 版主管理程序

一个版面通常有许多用户发表自己的言论，这通常需要一个或者多个版主来维护本版面的日常工作，比如删除垃圾文章、把优秀的文章加入精华区等工作，也就是说，版主具有比一般用户更高的权限，只要读者弄懂了前面几节的程序，相信后面的管理程序也非常容易懂，编程的思路和技术都没有变化，只是赋给管理员更多的权限而已。版主管理程序包括版主入口程序(mastermanage.php)、入口处理程序(boardlogin.php)以及版主版面显示程序（manageboard.php）。版主入口程序(mastermanage.php)非常简单，主要是用于输入版主帐号和密码界面。

mastermanage.php

```
<html>
<head>
<title>Untitled Document</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#ECFFEC">
<?
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
$text="select*from board";
$rs=mysql_query($text,$conn);
?>
<form method="post" action="boardlogin.php" name="adminlogin">
    <table width="80%" border="1" cellspacing="0" cellpadding="3" bordercolorlight="#FFFFFF" bordercolordark="#00B000" align="center">
```

Green Apple Data Center

```
        <tr bgcolor="#99FF99">
          <td colspan="2">斑竹管理 需要输入密码:</td>
        </tr>
        <tr>
          <td width="10%" nowrap>请选择版面:</td>
          <td><select name="boardid"><?while($mot=mysql_fetch_array($rs)){?><option value="<?echo $mot["boarded"];?>"><?
echo $mot["boardname"];?></option>
    <?};?>
                    </select>
          </td>
        </tr>
        <tr>
          <td width="10%" nowrap>用户名:</td>
          <td>
            <input type="text" name="masterusername" maxlength="50" size="30">
          </td>
        </tr>
        <tr>
          <td width="10%" nowrap>请输入密码:</td>
          <td>
            <input type="password" name="masterpassword" maxlength="50" size="30">
          </td>
        </tr>
          <td colspan="2">
            <input type="submit" name="Submit" value="登录&gt;&gt;">
            <input type="reset" name="Submit2" value=" 重写 ">
          </td>
        <tr>
        </tr>
      </table>
    </form>
    </body>
    </html>
```

在浏览器中如图 7-13 所示。

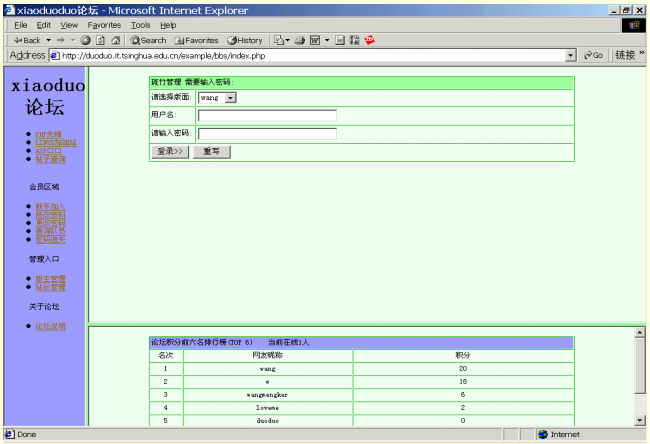


图 7-13 版主管理入口界面

Green Apple Data Center

入口处理程序(boardlogin.php)**主要是通过查询数据库检验入口程序界面里输入的版主帐号和密码是否正确，特别注意的是其中关于** session **部分。源程序如下：**

boardlogin.php

```
<?
session_start();
$boardid=htmlspecialchars($boardid);
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
//echo $loginusername;
//echo $loginpassword;
$text="select * from board where boardmaster='$masterusername' and masterpassword='$masterpassword' and boardid='$boardid'";
$check=mysql_query($text,$conn);
$recheck=mysql_fetch_row($check);
if (! $recheck){ //如果没有找到查询结果，即输入的用户不是版主
    $masterok=false;
    session_unregister("masterusername");
    session_unregister("masterpassword");
    session_unregister("boardid");
    session_unregister("masterok");
    //session_register("loginusername");
    //session_register("loginpassword");
    session_register("loginok");
    header("location:mastermanage.php"); //重新定位到输入界面
    }else{//如果有查询结果
    $masterok=true;
    session_register("masterusername");
    session_register("masterpassword");
    session_register("boardid");
    session_register("masterok");
    //echo $loginusername;
    //echo $loginpassword;
    //echo $loginok;
    header("location:manageboard.php");//定位到版主版面显示程序（manageboard.php）
    };
?>
```

版主版面显示程序（manageboard.php**）是最核心的程序，它的显示情况如图** 7-14 **所示，从图中可以看出和一般用户的界面差不多，只是多了一些权限比如删除文章、评定星级和加入精华区等。源程序如下：**

Green Apple Data Center

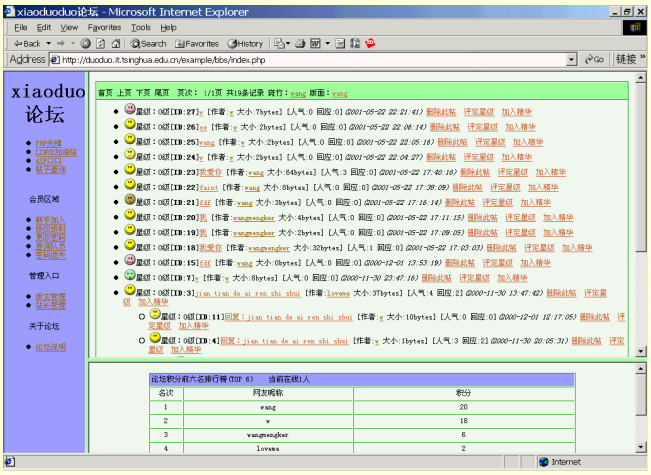


图 7-14　版主管理显示界面

```
<?include "boardchecklogin.php";?>
<?include "mastershow.php";?>
<?$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
$text="select*from board where boardid='$boardid'";
$rs=mysql_query($text,$conn);
$board=mysql_fetch_row($rs);
$tit="有以下错误以至操作不能进行：<br><ul>";
$comeok=true;
$err=mysql_error();
echo $err;
if (! $board){
        $tit=$tit."此版面不存在；<br>";
        $comeok=false;
}else{
    $boardname=$board[1];
    $boardmaster=$board[2];
    $masteremail=$board[3];
};
        $tit="";
        $page=htmlspecialchars($page);
        if ($page==""){
                $page=1;
        };
        $text="select count(*) as planid from plan where boardid='$boardid'";
        $rs=mysql_query($text,$conn);
        $pagesize=30;
        $count=mysql_result($rs,"0","planid");
        if (!$count){
                $tit="此版面还没有帖子";
                $pagecount=0;
        }
```

Green Apple Data Center

```
        else{
                $pagecount=ceil($count/$pagesize);
                $start=($page-1)*$pagesize;
        };
                        ?>
<html>
<head>
<title>Untitled Document</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#ECFFEC">
<table width="100%" border="1" cellspacing="0" cellpadding="2" bordercolorlight="#009900" bordercolordark="#FFFFFF">
  <tr bgcolor="#99FF99">
    <td>  <a  name="top"></a><?if  ($page>1){?><a  href="manageboard.php?boardid=<?echo  $boardid;?>&amp;page=
1&amp;username=<?echo  $username;?>">首页</a>
      <?}else{?> 首 页  <?};?><?if  ($page>1){?><a  href="manageboard.php?boardid=<?echo  $boardid;?>&amp;page=<?echo
$page-1;?>&amp;username=<?echo  $username;?>">上页</a>
      <?}else{?> 上 页  <?};?><?if  ($page<$pagecount){?><a  href="manageboard.php?boardid=<?echo  $boardid;?>&amp;
page=<?echo $page+1;?>&amp;username=<?echo  $username;?>">下页</a>
      <?}else{?> 下 页  <?};?><?if  ($page<$pagecount){?><a  href="manageboard.php?boardid=<?echo  $boardid;?>&amp;
page=<?echo $pagecount;?>&amp;username=<?echo  $username;?>">尾页</a>
      <?}else{?>尾页 <?};?>  页次： <font color=red><?echo $page;?></font>/<?echo $pagecount;?>页
      共<?echo $count ;?>条记录 斑竹：<a href="mailto:<? echo $masteremail?>"><? echo $boardmaster;?></a>
      版面：<a href="manageboard.php?boardid=<? echo $boardid?>&page=<?echo $page;?>&username=<?echo  $username;?
>&password=<?echo $password;?>"><? echo $boardname;?></a>
        <a href="#bottom"><img src="images/arrow-down.gif" width="9" height="19" border="0"></a>
    </td>
  </tr>
  <tr bgcolor="#ECFFEC">
    <td><?if (!$count){
  echo "<ul>".$tit."</ul>";}else{
    $text="select*from plan where boardid='$boardid' order by rootid desc,layer limit $start,$pagesize";
   showplan($text,$username,$password,$boardid);
  };
  ?> </td>
  </tr>
```

Green Apple Data Center

```
<tr bgcolor="#99FF99">
  <td> <?if ($page>1){?><a href="manageboard.php?boardid=<?echo $boardid;?>&amp;page=1&amp;username=<?echo $username;?>">首页</a>
  <?}else{?> 首 页 <?};?><?if ($page>1){?><a href="manageboard.php?boardid=<?echo $boardid;?>&amp;page=<?echo $page-1;?>&amp;username=<?echo $username;?>">上页</a>
  <?}else{?>上页 <?};?><?if ($page<$pagecount){?><a href="manageboard.php?boardid=<?echo $boardid;?>&amp; page=<?echo $page+1;?>&amp;username=<?echo $username;?>">下页</a>
  <?}else{?>下页 <?};?><?if ($page<$pagecount){?><a href="manageboard.php?boardid=<?echo $boardid;?>&amp; page=<?echo $pagecount;?>&amp;username=<?echo $username;?>">尾页</a>
  <?}else{?>尾页 <?};?>  页次： <font color=red><?echo $page;?></font>/<?echo $pagecount;?>页
  共<?echo $count ;?>条记录 斑竹：<a href="mailto:<? echo $masteremail?>"><? echo $boardmaster;?></a>
  版 面 ：<a href = "manageboard.php?boardid = <? echo $boardid?>&page = <?echo $page;?>&username=<?echo $username;?>&password=<?echo $password;?>"><? echo $boardname;?></a>
    <a href="#top"><img src="images/arrow-up.gif" width="9" height="19" border="0"></a><a name="bottom"></a></td>
</tr>
<tr bgcolor="#ECFFEC">
  <td>
    <form method="post" action="manageboard.php" name="goto">
      转到
      <input type="text" name="page" maxlength="6" size="3">
      页
      <input type="submit" name="Submit" value="GOTO">
    </form>
  </td>
</tr>
</table>
</body>
</html>
```

版主版面显示程序（manageboard.php**）包括两个程序** boardchecklogin.php **和** mastershow.php**。**boardchecklogin.php **是用来检测** session **的，也就是检查用户是否有版主身份，**mastershow.php **和前面的普通用户显示文章程序** showplan **差不多，主要用于显示文章标题。它们的源程序如下：**

boardchecklogin.php

```
<?
session_start();
if( $loginok=true && $masterusername!="" && $masterpassword!="" && $boardid!=""){
    header("location: manageboard.php?boardid=".$boardid);
}else{
    $masterusername="";
    $masterpassword="";
    $masterok=false;
    header("location: mastermanage.php");
};
?>
```

Green Apple Data Center

mastershow.php

```
<?
function showplan($text,$username,$password,$boardid){
    $qconn = mysql_connect('localhost:3306','root','');
    mysql_select_db('mysql');
    $rs=mysql_query ($text,$qconn);
    $i= 0 ;
    echo "<ul>";
    while ($rowt=mysql_fetch_array ($rs) ){
        if ( $rowt["planid"]!=$rowt["parentid"]) {
            if ($rowt["renum"]>=1 ){
                    if ($parentid[$i]!=$rowt["parentid"]){
                        for($j=$i;$j>=1;$j--){
                            //echo "fdafdasfdasfdsafdafdsa";
                            if ($parentid[$j]!=$rowt["parentid"]){
                                echo "</ul>";//.$rowt["parentid"];
                                $i--;
                            }else{
                              // echo "</ul>" ;
                                $j=0;
                            };
                        };
                    }else{
                        $parentid[$i]=$rowt["parentid"];
                    };
                $parentid[$i+1] = $rowt["planid"];
                $i++;
                echo "<li>";//.$rowt["parentid"]."fdafdas";
                echo "<img src='../images/mod" ;
                echo $rowt["emote"] ;
                echo ".gif'>" ;
              echo "星级：";
        if ($rowt["hote"]==0){echo "0 级";};for($jm=$rowt["hote"];$jm>=1;$jm--){echo "*";};
          echo "[<B>ID:".$rowt["planid"]."</B>]"."<a href='replan.php?boarded ="$rowt ["boardid"]. "&planid=".$rowt["planid"]."' target='_bland'>".$rowt["title"]."</a>"." [ 作 者 :"."<a href='mailto:".$rowt ["email"].">".$rowt["username"]."</a> 大小:".strlen($rowt["body"])."bytes]"." [人气:".$rowt["clicknum"]." 回应:".$rowt["renum"]."]"."<i>(".$rowt["addtime"].")</i> ";
                $ctime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
                echo "<a href='delplan.php?planid=".$rowt["planid"]."&username=".$username."&password=".$password."&boardid=".$boardid."' target='_blank'>删除此帖</a>";
                echo "  <a href='setplan.php?planid=".$rowt["planid"]."&username=".$username."&password=".$password."&boardid=".$boardid."' target='_blank'>评定星级</a>";
                echo "  <a href='addtoglod.php?planid=".$rowt["planid"]."&username=".$username."&password=".$password."&boardid=".$boardid."' target='_blank'>加入精华</a>";
                 echo "</li><ul>";
            }else{
                        if ($parentid[$i]!=$rowt["parentid"]){
```

Green Apple Data Center

```
                    for($j=$i;$j>=1;$j--){
                        //echo "fdafdasfdasfdsafdafdsa";
                      if ($parentid[$j]!=$rowt["parentid"]){
                        echo "</ul>";//.$rowt["parentid"];
                        $i--;
                      }else{
                       // echo "</ul>" ;
                        $j=0;
                      };
                    };
                }else{
                  $parentid[$i]=$rowt["parentid"];
                };
                echo "<li>";
                echo "<img src='../images/mod" ;
                echo $rowt["emote"] ;
                echo ".gif'>" ;
            echo "星级：";
                if ($rowt["hote"]==0){echo "0 级";};for($jm=$rowt["hote"];$jm>=1;$jm--){echo "*";};
              echo"[<B>ID:".$rowt["planid"]."</B>]"."<ahref='replan.php?boardid=".$rowt["boardid"]."&planid
    ="."$rowt["planid"]."'target='_bland'>".$rowt["title"]."</a>"."  [ 作 者 :"."<a  href='mailto:".$rowt["email"]."'>".$rowt["username"].
"</a>  大小:".strlen($rowt["body"])."bytes]"." [人气:".$rowt["clicknum"]."  回应:".$rowt["renum"]."]"."<i>(".$rowt["addtime"].")</i>  ";
                $ctime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
              echo "<a href='delplan.php?planid=".$rowt["planid"]."&username=".$username."&password=".$password."&boardid=
".$boardid."' target='_blank'>删除此帖</a>";
              echo "  <a href='setplan.php?planid=".$rowt["planid"]."&username=".$username."&password=
".$password."&boardid=".$boardid."' target='_blank'>评定星级</a>";
              echo "  <a href='addtoglod.php?planid=".$rowt["planid"]."&username=".$username."&password=
".$password."&boardid=".$boardid."' target='_blank'>加入精华</a>";
            echo "</li>";
            };
          }else{
            if ($i==1){
              echo "</ul>";
              $i=0;
            }else{
                    for($j=$i;$j>=1;$j--){
                        //echo "fdafdasfdasfdsafdafdsa";
                      if ($parentid[$j]!=$rowt["parentid"]){
                        echo "</ul>";//.$rowt["parentid"];
                        $i--;
                      }else{
                       // echo "</ul>" ;
                        $j=0;
                      };
                    };
              $i=0;
```

Green Apple Data Center

```
                };
                echo "<li>";
                echo "<img src='../images/mod" ;
                echo $rowt["emote"] ;
                echo ".gif'>" ;
                echo "星级：";
                if ($rowt["hote"]==0){echo "0 级";};for($jm=$rowt["hote"];$jm>=1;$jm--){echo "*";};
            echo  "[<B>ID:".$rowt["planid"]."</B>]"."<a  href='replan.php?boardid=".$rowt["boardid"]."&planid=".$rowt["planid"]."'
target='_bland'>".$rowt["title"]."</a>"."  [作者:"."<a  href='mailto:".$rowt["email"]."'>".$rowt["username"]."</a>  大小:".strlen($rowt
["body"])."bytes]"." [人气:".$rowt["clicknum"]."  回应:".$rowt["renum"]."]"."<i>(".$rowt["addtime"].")</i>  ";
                $ctime=date("Y")."-".date("m")."-".date("d")." ".date("H").":".date("i").":".date("s");
                    echo "<a href='delplan.php?planid=".$rowt["planid"]."&username=".$username."&password=".$password."&boardid=
".$boardid."' target='_blank'>删除此帖</a>";
                    echo  "  <a  href='setplan.php?planid=".$rowt["planid"]."&username=".$username."&password=
".$password."&boardid=".$boardid."' target='_blank'>评定星级</a>";
                    echo  "  <a  href='addtoglod.php?planid=".$rowt["planid"]."&username=".$username."&password=
".$password."&boardid=".$boardid."' target='_blank'>加入精华</a>";
            echo "</li>";
                if ($rowt["renum"]>=1 ){
                    $parentid[$i+1] = $rowt["planid"];
                    $i++;
                    echo "<ul>";
                };
                //$i=$i+1;
            };
        };
       // $i=$i+($i+1);
        for($j=0;$j<=$i;$j++){
            echo "</ul>";
        };
    };
    ?>
```

其中版主显示文章内容程序和普通用户显示文章内容程序一样，都是 replan.php（程序源码见上一节），**不同的只是多了删除文章程序（**delplan.php**）、评定星级程序（**setplan.php**）和把文章加入精华区程序（**addtoglod.php**），它们的源程序分别如下：**

delplan.php

```
<?include "boardchecklogin.php"?>
<?include "plansave.php"?>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#FFFFFF">
```

Green Apple Data Center

```
<?
$planid=htmlspecialchars($planid);
$username=htmlspecialchars($username);
$password=htmlspecialchars($password);
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
$text="select*from plan where planid='$planid'";
$rs=mysql_query($text,$conn);
$board=mysql_fetch_row($rs);
$tit="有以下错误以至操作不能进行：<br><ul>";
$comeok=true;
$err=mysql_error();
echo $err;
if (! $board){
        $tit=$tit."此帖子不存在；<br>";
        $comeok=false;
}else{
    $parentid=$board[17];
    $decusername=$board[6];
};
if ($comeok==true){
$login=true;
//session_register($login);
session_name($login);
session_name($boardid);
session_name($username);
session_name($boardname);
session_name($password);
//echo session_name($login);
};
if ($comeok==false){
?>
<?echo $tit;?>
        <input type="button" name="comeback" onclick="document.location='mastermanage.php'" value="&lt;&lt;返回">
<?}else{?>
<?
$text="delete from plan where planid='$planid'";
mysql_query($text,$conn);
decplan($parentid);
decusernum($decusername);
?>
<br>
已经删除此帖！！
<br>
<br>
<br>
<a href="javascript:{window.close()}">关闭窗口</a>
```

Green Apple Data Center

```
<?};?>
</body>
</html>

setplan.php
<?include "boardchecklogin.php"?>
<html>
<head>
<title>Untitled Document</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#ECFFEC">
<?

$planid=htmlspecialchars($planid);
$boardid=htmlspecialchars($boardid);
$username=htmlspecialchars($username);
$boardmaster=$username;
$password=htmlspecialchars($password);
$masterpassword=$password;
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
$text="select*from plan where planid='$planid'";
$rs=mysql_query($text,$conn);
$srom=mysql_fetch_row($rs);
$tit="有以下错误以至操作不能进行：<br><ul>";
$comeok=true;
$err=mysql_error();
echo $err;
if (! $srom){
        $tit=$tit."此帖子不存在；<br>";
        $comeok=false;
}else{
    $boardid=$srom[1];
    $username=$srom[6];
    $title=$srom[4];
```

Green Apple Data Center

```
        $body=$srom[5];
        $addtime=$srom[16];
        $link=$srom[8];
        $email=$srom[7];
        $linktitle=$srom[13];
        $picturelink=$srom[9];
        $clicknum=$srom[11];
        $writename=$srom[12];
        //echo $clicknum."fdasfdsa";
        //addclicknum($planid,$clicknum+1);
        $rootid=$srom[2];
        $layer=$srom[3];
        $hote=$srom[19];
    };

    $text="select*from board where boardid='$boardid'";
    $rs=mysql_query($text,$conn);
    $board=mysql_fetch_row($rs);
    if (!$board){
        $tit=$tit."此版面不存在！；<br>";
        $comeok=false;
    }else{
        $boardname=$board[1];
    };
    if ($comeok==true){
    $login=true;
    //session_register($login);
    session_name($login);
    session_name($boardid);
    session_name($username);
    session_name($boardname);
    session_name($password);
    //echo session_name($login);
    };
    if ($comeok==false){
    ?> <?echo $tit;?>
    <input type="button" name="comeback" onClick="document.location='mastermanage.php'" value="&lt;&lt;返回">
    <?}else{?> <?
    ?>
    <table width="100%" border="1" cellspacing="0" cellpadding="2" bgcolor="#FFFFFF" bordercolorlight="#00CC00"
bordercolordark="#FFFFFF">
      <tr bgcolor="#99FF99">
        <td> 评定星级： <b> </b></td>
      </tr>
      <tr bgcolor="#ECFFEC" valign="top">
        <td>
          <form action="savehote.php" method="POST" name="planform" >
```

Green Apple Data Center

```
                <input type="hidden" name="planid" value="<?echo $planid;?>">
                <span class=smallFont><a name="reply">版 面</a>： <? echo $boardname; ?>
                </span>
                <input type="hidden" name="username" value="<?echo $boardmaster;?>">
                <input type="hidden" name="password" value="<?echo $masterpassword;?>">
                <input type="hidden" name="boardid" value="<?echo $boardid;?>">
                <br>
                <span class=smallFont>主 题：</span> <?echo $title;?> 星级：<?if ($hote==0){echo "0 级";};for($jm=$hote;
$jm>=1;$jm--){echo "*";};?>
                <p><? if ($body==""){
          echo "无内容";}
      else{
      echo "内容（".strlen($body)."bytes）:<BR><ul>";
      $Tbody=str_replace("\n","<br>",$body);
      echo $Tbody;
      echo "</ul>";
      }?></p>
                <p>请选择星级：
                  <input type="radio" name="hote" value="1">
                  一级
                  <input type="radio" name="hote" value="2">
                  二级
                  <input type="radio" name="hote" value="3">
                  三级
                  <input type="radio" name="hote" value="4">
                  四级
                  <input type="radio" name="hote" value="5">
                  五级</p>
                <p> </p>
                <p>
                  <input type="submit" name="Submit" value="提交评定">
                  <input type="reset" name="Submit2" value="重新选择">
                  <br>
                </p>
              </form>
            </td>
          </tr>
      </table>
      <?};?>
      </body>
      </html>

      addtoglod.php
      <?include "boardchecklogin.php";?>
      <?include "plansave.php";?>
      <html>
      <head>
```

Green Apple Data Center

```
<title>Untitled Document</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#FFFFFF">
<?

$planid=htmlspecialchars($planid);
$username=htmlspecialchars($username);
$password=htmlspecialchars($password);
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
$text="select*from plan where planid='$planid'";
$rs=mysql_query($text,$conn);
$board=mysql_fetch_row($rs);
$tit="有以下错误以至操作不能进行：<br><ul>";
$comeok=true;
$err=mysql_error();
echo $err;
if (! $board){
        $tit=$tit."此帖子不存在；<br>";
        $comeok=false;
}else{
    $title=$board[4];
    $body=$board[5];
    $username=$board[6];
};
if ($comeok==true){
$login=true;
//session_register($login);
session_name($login);
session_name($boardid);
session_name($username);
session_name($boardname);
session_name($password);
//echo session_name($login);
};
```

Green Apple Data Center

```
if ($comeok==false){
?> <?echo $tit;?>
<p>
   <input type="button" name="comeback" onClick="document.location='mastermanage.php'" value="&lt;&lt;返回">
   <?}else{?> <?
addglod($username);
$text="update plan set glod='是' where planid='$planid'";
mysql_query($text,$conn);
$err=mysql_error();
echo $err;
?> <br>
</p>
<p>已经把此帖子加入精华区域！<br>
   <br>
   <br>
   <a href="javascript:{window.close()}">关闭窗口</a> <?};?> </p>
</body>
</html>
```

7.5.2 站长管理程序

一个论坛一般都需要一个或者几个站长来管理本站的事务，具有该站点的最高权限，比如维护用户帐号、负责更换版主、管理版面等。由于站长人数可能非常少，一般只有一个人，所以本论坛程序只是内置了一个 administrator **帐号作为站长，有兴趣的读者可以修改这个程序是论坛可以具有多个站长。**

站长管理程序包括管理入口程序（master.php**）、入口处理程序**(login.php)**以及站长版面显示程序**(manager.php)**。**

管理入口程序（master.php**）十分简单，就是用于输入站长密码，因为本论坛的站长帐号只有一个并且事先在数据库中内置了一个** administrator **帐号。源程序如下：**

master.php

```
<html>
<head>
<title>Untitled Document</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>

<body bgcolor="#ECFFEC">
```

Green Apple Data Center

```
<form method="post" action="login.php" name="master">
  <table width="80%" border="1" cellspacing="0" cellpadding="3" bordercolorlight="#FFFFFF" bordercolordark="#00B000" align="center">
    <tr bgcolor="#99FF99">
      <td colspan="2">站长管理 需要输入密码</td>
    </tr>
    <tr>
      <td width="10%" nowrap>请输入密码:</td>
      <td>
        <input type="password" name="password" maxlength="50" size="30">
      </td>
    </tr>
    <tr>
      <td colspan="2">
        <input type="submit" name="Submit" value="登录&gt;&gt;">
        <input type="reset" name="Submit2" value=" 重写 ">
      </td>
    </tr>
  </table>
</form>
</body>
</html>
```

入口处理程序(login.php)**主要是通过查询数据库检验入口程序界面里输入站长密码是否正确，特别注意的是其中也是用到了** session **技术，和前面版主管理程序相似。源程序如下：**

login.php

```
<?
session_start();
$loginusername="administrator";
$loginpassword=htmlspecialchars($password);
$conn=mysql_connect('localhost:3306','root','');
mysql_select_db("mysql");
//echo $loginusername;
//echo $loginpassword;
$text="select * from boarduser where username='$loginusername' or password='$loginpassword'";
$check=mysql_query($text,$conn);
$recheck=mysql_fetch_row($check);
if (! $recheck){
    $loginok=false;
    //session_register("loginusername");
    //session_register("loginpassword");
    session_register("loginok");
    //echo "fda";
    header("location: master.php");
}else{
    $loginok=true;
```

Green Apple Data Center

```
        session_register("loginusername");
        session_register("loginpassword");
        session_register("loginok");
        //echo $loginpassword;
        //echo $loginok;
        header("location: manager.php");
        };
?>
```

站长版面显示程序(manager.php)**用于显示站长的管理界面，如图** 7-15 **所示，本论坛站长管理只包括删除用户帐号和版面管理，有兴趣的读者可以增加更多的功能，比如添加用户帐号等。**

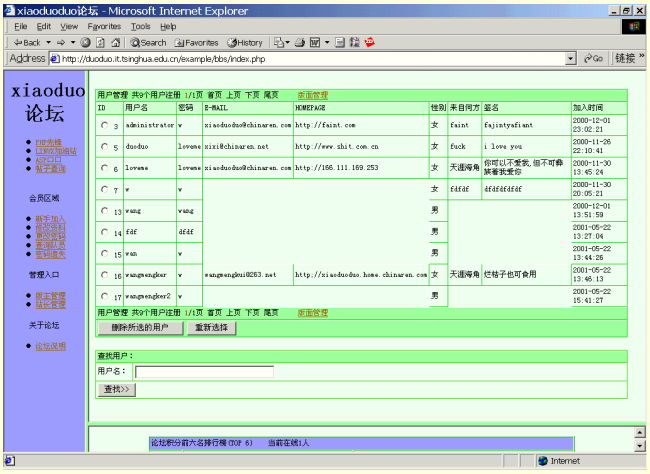


图 7-15 **站长管理显示界面**

其源程序如下：

manager.php

```
<? include "checklogin.php"?>
<html>
<head>
<title>Untitled Document</title>
<style type="text/css">
<!--
a:active {    color: #666600; text-decoration: underline}
a:hover {    color: #FF0000; text-decoration: none}
a:link {    color: #CC6633; text-decoration: underline}
a:visited {    color: #996600; text-decoration: underline}
body {    font-size: 12px}
td {    font-size: 12px}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
</head>
<body bgcolor="#ECFFEC">
<?
$conn=mysql_connect('localhost:3306','root','');
```

Green Apple Data Center

```
mysql_select_db("mysql");
?>
<?
$page=0+$page;
$pagesize=10;
$text="select count(*) as id from boarduser order by id";
$rs=mysql_query($text);
$count=mysql_result($rs,"0","id");
$pagecount=ceil($count/$pagesize);
if ($page>$pagecount) {
        $page=$pagecount;
};
if ($page<=0){
        $page=1;
};
$start=($page-1)*$pagesize;
$text="select*from boarduser order by id limit $start,$pagesize";
$rs=mysql_query($text);
?>
<script language="JavaScript">
function check(){
var delok=true;
if (confirm("真的要删除吗？")==false){
     delok=false
};
return delok;
};
</script>
<br>
<form name="deluser" action="deluser.php" onsubmit="return check()" method="post">
   <table width="100%" border="1" cellspacing="0" cellpadding="2" bordercolorlight="#00CC00" bordercolordark="#FFFFFF">
   <tr bgcolor="#99FF99">
        <td colspan="9">用户管理 共<?echo $count;?>个用户注册 <font color=red><?echo $page;?></font>/<?echo $pagecount;?>页
          <?if ($page>1){?><a href="manager.php?page=1">首页</a> <?}else{?>首页 <?};?><?if ($page>1){?><a href="manager.php?page=<?echo $page-1;?>">上页</a>
          <?}else{?>上页 <?};?><?if ($page<$pagecount){?><a href="manager.php?page=<?echo $page+1;?>">下页</a>
          <?}else{?>下页 <?};?><?if ($page<$pagecount){?><a href="manager.php?page=<?echo $pagecount;?>">尾页</a>
          <?}else{?>尾页<?};?>     <a href="boardmanger.php">版面管理</a></td>
   </tr>
   <tr bgcolor="#CCFFCC">
      <td nowrap>ID</td>
      <td nowrap>用户名</td>
      <td nowrap>密码</td>
      <td nowrap>E-MAIL</td>
      <td nowrap>HOMEPAGE</td>
      <td nowrap>性别</td>
```

Green Apple Data Center

```
        <td nowrap>来自何方</td>
        <td nowrap>签名</td>
        <td nowrap>加入时间</td>
      </tr>
      <?while($rowt=mysql_fetch_array($rs)){?>
      <tr>
        <td nowrap>
          <input type="radio" value="<?echo $rowt["id"];?>" name="id">
          <?echo $rowt["id"];?></td>
        <td><?echo $rowt["username"];?></td>
        <td><?echo $rowt["password"];?></td>
        <td><?echo $rowt["email"];?></td>
        <td><?echo $rowt["homepage"];?></td>
        <td><?echo $rowt["sex"];?></td>
        <td><?echo $rowt["comefrom"];?></td>
        <td><?echo $rowt["writename"];?></td>
        <td><?echo $rowt["addtime"];?></td>
      </tr>
      <?};?>
      <tr>
          <td colspan="9" bgcolor="#99FF99">用户管理 共<?echo $count;?>个用户注册 <font color=red><?echo $page;?></font>
/<?echo $pagecount;?>页
            <?if ($page>1){?><a href="manager.php?page=1">首 页 </a> <?}else{?> 首 页 <?};?><?if ($page>1){?><a href=
"manager.php?page=<?echo $page-1;?>">上页</a>
            <?}else{?>上页 <?};?><?if ($page<$pagecount){?><a href="manager.php?page=<?echo $page+1;?>">下页</a>
            <?}else{?>下页 <?};?><?if ($page<$pagecount){?><a href="manager.php?page=<?echo $pagecount;?>">尾页</a>
            <?}else{?>尾页<?};?>     <a href="boardmanger.php">版面管理</a></td>
      </tr>
    <tr>
          <td colspan="9" bgcolor="#99FF99">
            <input type="submit" name="Submit" value="删除所选的用户">
            <input type="reset" name="Submit2" value="重新选择">
          </td>
        </tr>
    </table>
    </form>
    <form method="post" action="mastersearch.php" name="search" target="_blank">
      <table width="100%" border="1" cellspacing="0" cellpadding="2" bordercolorlight="#33CC00" bordercolordark="#FFFFFF">
        <tr bgcolor="#99FF99">
          <td colspan="2">查找用户：</td>
        </tr>
        <tr>
          <td nowrap width="7%">用户名：</td>
          <td nowrap width="93%">
            <input type="text" name="username" maxlength="50" size="30">
          </td>
        </tr>
```

Green Apple Data Center

```
        <tr>
          <td colspan="2">
            <input type="submit" name="Submit3" value="查找&gt;&gt;">
          </td>
        </tr>
      </table>
</form>
</body>
</html>
```

此程序包括一个 session **检测程序（**checklogin.php**），用于检查站长身份，其源程序如下：**

checklogin.php

```
<?
session_start();
//echo $loginusername;
//echo $loginpassword;
//echo $loginok;
if( $loginok=true or $loginusername=="administrator" ){
        ;
}else{
        $loginusername="";
        $loginpassword="";
        $loginok=false;
        header("location: master.php");
};
?>
```

本论坛站长版面管理程序和版主版面管理程序是一样的，这里就不在赘述。到此为止，这个功能强大的论坛基本上已经建立完毕，读者在学习的过程中也体会到商业网站的建立过程，这是在其他一般基础书籍上不能体会到的。当然论坛程序还有包括文章查询、显示在线用户等功能，这些功能实现起来相对简单，为了节约篇幅，本书就不在赘述了。有兴趣的读者可以参考本书后面附带的光盘中的全部论坛源程序。

Green Apple Data Center

附　录

附录 A　PHP 函数列表

下面是从 PHP 按字母顺序产生的函数清单。学习 PHP 时，经常想要一个特定的函数，但却不知道它属于下面的哪一组，下面的这个列表可以节约时间。

函　数	组　名	描　述
Abs	MATH	取绝对值
Acos	MATH	取反余弦值
ada_afetch	ADABAS	取得数据库的返回行
ada_autocommit	ADABAS	开关自动切换功能
ada_close	ADABAS	关闭与 AdabasD 服务器的链接
ada_commit	ADABAS	更动 AdabsdD 数据库
ada_connect	ADABAS	链接 AdabasD 数据源
ada_exec	ADABAS	准备并执行一个 SQL 语句
ada_fetchrow	ADABAS	从结果中取得一行
ada_fieldname	ADABAS	取得一栏的名字
ada_fieldnum	ADABAS	取得栏号
ada_fieldtype	ADABAS	取得一个字段的数据类型
ada_freeresult	ADABAS	释放返回数据的内存
ada_numfields	ADABAS	获得结果栏的数目
ada_numrows	ADABAS	获得结果行的数目
ada_result	ADABAS	从结果中取得数据
ada_resultall	ADABAS	以 HTML 表格方式打印结果
ada_rollback	ADABAS	取消一次事务
AddSlashes	STRINGS	使用斜线引住字符串
apache_lookup_uri	APACHE	为指定 URI 执行部分请求有关的所有信息
apache_note	APACHE	取得和设置 apache 请求注解
Array	ARRAY	创建一个数组
array_walk	ARRAY	对每个数组的成员应用一个用户函数
Arsort	ARRAY	反向排序一个数组并维护索引
Asin	MATH	反正弦
Asort	ARRAY	排序一个数组并维护索引
ASPELL_check	ASPELL	检查单词
ASPELL_check_raw	ASPELL	不改变它的大小写以检查单词，或试图删除其中的空格
ASPELL_new	ASPELL	载入一个新字典
ASPELL_suggest	ASPELL	单词的建议拼写
Atan	MATH	反正切
Atan2	MATH	两个变量的反正切
base64_decode	URL	解开 base64 编码的数据
base64_encode	URL	使用 base64 对数据进行编码
basename	FILESYSTEM	返回路线的组件
base_convert	MATH	改变数字的进制表示
bcadd	BC	将两个高精度数字相加
bccomp	BC	比较两个高精度的数字

Green Apple Data Center

续 表

函 数	组 名	描 述
Bcdiv	BC	两个高精度的数字相除
Bcmod	BC	获得两个任意精度数字取模的余数
Bcmul	BC	两个任意精度数字相乘
Bcpow	BC	取一个任意数字的另一个数字的幂
bcscale	BC	设置所有 bc 数学计算的缺省精度
Bcsqrt	BC	取得任意精度数字的平方根
Bcsub	BC	两个任意精度数字相减
Bin2hex	STRINGS	转变二进制数据为十六进制的表示方式
BinDec	MATH	二进制到十进制转换
Ceil	MATH	取整
Chdir	DIR	改变目录
checkdate	DATETIME	验证日期/时间
checkdnsrr	NETWORK	检查给定名字或 IP 地址的 DNS 记录
Chgrp	FILESYSTEM	改变文件的属组
Chmod	FILESYSTEM	改变文件的属性
Chop	STRINGS	删除结尾空白字符
Chown	FILESYSTEM	改变文件属主
Chr	STRINGS	返回指定字符
Chunk_split	STRINGS	分割字符串
clearstatcache	FILESYSTEM	清除文件 stat 的缓冲
closedir	DIR	关闭目录句柄
closelog	NETWORK	关闭与系统日志的链接
connection_aborted	MISC	如果客户断开了，则返回真
connection_status	MISC	返回链接状态的位字段
connection_timeout	MISC	如果超时则返回真
convert_cyr_string	STRINGS	TRINGS 从一个 Cyrillic 字符集转换为另一个
Copy	FILESYSTEM	复制文件
Cos	MATH	余弦值
count	ARRAY	对变量中的元素进行记数
cpdf_add_annotation	CPDF	添加注解
cpdf_add_outline	CPDF	为当前页面添加书签
cpdf_arc	CPDF	画弧
cpdf_begin_text	CPDF	开始一个文本部分
cpdf_circle	CPDF	画圆
cpdf_clip	CPDF	剪切到当前路线
cpdf_close	CPDF	关闭 pdf 文档
cpdf_closepath	CPDF	关闭路线
cpdf_closepath_fill_stroke	CPDF	关闭、并在当前路线填充并画线
cpdf_closepath_stroke	CPDF	关闭路线并沿路线画线
cpdf_continue_text	CPDF	在下一行输出文本
cpdf_curveto	CPDF	画曲线
cpdf_end_text	CPDF	开始文本部分
cpdf_fill	CPDF	填充当前路线
cpdf_fill_stroke	CPDF	填充并画线
cpdf_finalize	CPDF	终结文档
cpdf_finalize_page	CPDF	终结页面
cpdf_import_jpeg	CPDF	打开 JPEG 图像

Green Apple Data Center

续 表

函 数	组 名	描 述
cpdf_lineto	CPDF	画线
cpdf_moveto	CPDF	设置当前点
cpdf_open	CPDF	打开一个新 pdf 文档
cpdf_output_buffer	CPDF	在内存缓冲中输出 pdf 文档
cpdf_page_init	CPDF	开始一个新页面
cpdf_page_inline_image	CPDF	在一个页面上放一个图像
cpdf_rect	CPDF	画矩形
cpdf_restore	CPDF	恢复以前保存的环境
cpdf_rlineto	CPDF	画线
cpdf_rmoveto	CPDF	设置当前点
cpdf_rotate	CPDF	设置旋转
cpdf_save	CPDF	保存当前环境
cpdf_save_to_file	CPDF	将 pdf 文档写入文件中
cpdf_scale	CPDF	设置比例
cpdf_setdash	CPDF	设置虚线模式
cpdf_setflat	CPDF	设置为平滑线条
cpdf_setgray	CPDF	设置画和填充的灰度值
cpdf_setgray_fill	CPDF	设置填充的灰度值
cpdf_setgray_stroke	CPDF	设置画线的灰度值
cpdf_setlinecap	CPDF	设置 linecap 参数
cpdf_setlinejoin	CPDF	设置 linejoin 参数
cpdf_setlinewidth	CPDF	设置线宽
cpdf_setmiterlimit	CPDF	设置斜线宽
cpdf_setrgbcolor	CPDF	设置画线和填充颜色的 RGB
cpdf_setrgbcolor_fill	CPDF	设置填充颜色的 rgb 值
cpdf_setrgbcolor_stroke	CPDF	设置画线颜色的 rgb 值
cpdf_set_char_spacing	CPDF	设置字符间隔
cpdf_set_creator	CPDF	设置 pdf 文档的创建者域
cpdf_set_current_page	CPDF	设置当前页
cpdf_set_font	CPDF	设置当前字体外观和大小
cpdf_set_horiz_scaling	CPDF	设置水平比例
cpdf_set_keywords	CPDF	设置 pdf 文档的关键词域
cpdf_set_leading	CPDF	
cpdf_set_page_animation	CPDF	设置页面之间的持续时间
cpdf_set_subject	CPDF	设置 pdf 文档的主题域
cpdf_set_text_matrix	CPDF	设置文本矩阵
cpdf_set_text_pos	CPDF	设置文本位置
cpdf_set_text_rendering	CPDF	设置文本表现的方式
cpdf_set_text_rise	CPDF	设置文本增大
cpdf_set_title	CPDF	设置 pdf 文档的标题域
cpdf_set_word_spacing	CPDF	设置词之间的空格
cpdf_show	CPDF	在当前位置输出文本
cpdf_show_xy	CPDF	在 xy 位置输出文本
cpdf_stringwidth	CPDF	以当前字体返回文本的宽度
cpdf_stroke	CPDF	沿路线画线
cpdf_text	CPDF	使用参数输出文本
cpdf_translate	CPDF	设置坐标原点

Green Apple Data Center

续 表

函 数	组 名	描 述
crypt	STRINGS	对字符串进行 DES 加密
current	ARRAY	返回一个数组的当前元素
date	DATETIME	格式化本地时间/日期
dbase_add_record	DBASE	向 dBase 数据库添加数据
dbase_close	DBASE	关闭 dBase 数据库
dbase_create	DBASE	创建 dBase 数据库
dbase_delete_record	DBASE	从 dBase 数据库中删除记录
dbase_get_record	DBASE	从 dBase 数据库中取记录
dbase_numfields	DBASE	找出 dBase 数据库中有多少字段
dbase_numrecords	DBASE	找出 dBase 数据库中有多少记录
dbase_open	DBASE	打开一个 dBase 数据库
dbase_pack	DBASE	打包一个 dBase 数据库
dbase_replace_record	DBASE	替换 dBase 数据库中的一条记录
dba_close	DBA	关闭数据库
dba_delete	DBA	删除键指引的条目
dba_exists	DBA	检查键是否存在
dba_fetch	DBA	获取键指引的数据
dba_firstkey	DBA	获取第一个键
dba_insert	DBA	添加一个条目
dba_nextkey	DBA	获取下一个键
dba_open	DBA	打开数据库
dba_optimize	DBA	优化数据库
dba_popen	DBA	打开一个数据库永久链接
dba_replace	DBA	替换或插入一个条目
dba_sync	DBA	同步数据库
dblist	DBM	描述正在使用的 dbm 兼容库
dbmclose	DBM	关闭库
dbmdelete	DBM	删除一个条目
dbmexists	DBM	判断是否一个键对应的值在 dbm 数据库中
dbmfetch	DBM	从 dbm 数据库中用一个键获取一个值
dbmfirstkey	DBM	取得第一个键
dbminsert	DBM	为一个键添加一个值
dbmnextkey	DBM	取得下一个键
dbmopen	DBM	打开一个 dbm 数据库
dbmreplace	DBM	在 dbm 数据库中替换一个键的值
debugger_off	NETWORK	屏蔽 PHP 内部调试器
debugger_on	NETWORK	打开 PHP 内部调试器
DecBin	MATH	十进制转换为二进制
DecHex	MATH	十进制转换为十六进制
DecOct	MATH	十进制转换为八进制
delete	FILESYSTEM	无手册条目
die	MISC	输出信息并终止程序
dir	DIR	目录类
dirname	FILESYSTEM	返回路径的目录名
diskfreespace	FILESYSTEM	返回目录下的可用空间
dl	DL	在运行时刻载入 PHP 扩展
doubleval	VAR	取得变量值的两倍

Green Apple Data Center

续 表

函 数	组 名	描 述
each	ARRAY	返回下一个键/值对
easter_date	CALENDAR	获取给定年复活节半夜的 Unix 时间
easter_days	CALENDAR	获取在 3 月 21 日后到给定年复活节的天数
echo	STRINGS	输出一个或多个字符串
empty	VAR	决定一个变量是否被设置了
end	ARRAY	设置数组的内部指针到其最后一个元素
ereg	REGEX	正则表达式匹配
eregi	REGEX	忽略大小写的正则表达式匹配
eregi_replace	REGEX	忽略大小写的正则表达式匹配并替换
ereg_replace	REGEX	正则表达式匹配并替换
error_log	INFO	发送一条错误日志
error_reporting	INFO	设置要报告的 PHP 错误
escapeshellcmd	EXEC	将 shell 转义字符转回其字面意义
eval	MISC	将值代入字符串中
exec	EXEC	执行一个外部程序
exit	MISC	终止当前脚本
Exp	MATH	e 的幂次
explode	STRINGS	分割字符串
extension_loaded	INFO	找出是否载入了扩展
extract	MISC	导入变量到一个符号表中
fclose	FILESYSTEM	关闭文件指针
fdf_close	FDF	关闭 FDF 指针
fdf_create	FDF	创建新的 FDF 文档
fdf_get_file	FDF	获得/F 键的值
fdf_get_status	FDF	获得/STATUS 键的值
fdf_get_value	FDF	获得字段的值
fdf_next_field_name	FDF	获得下一个字段的名字
fdf_open	FDF	打开 FDF 文档
fdf_save	FDF	保存 FDF 文档
fdf_set_ap	FDF	设置字段的外观
fdf_set_file	FDF	设置/F 键的值
fdf_set_status	FDF	设置/STATUS 键的值
fdf_set_value	FDF	设置字段的值
feof	FILESYSTEM	测试文件结束标志
fgetc	FILESYSTEM	从文件指针中获取一个字符
fgetcsv	FILESYSTEM	从文件指针获取一行，并分析 CSV 字段
fgets	FILESYSTEM	从文件指针获取一行
fgetss	FILESYSTEM	从文件指针获取一行，并删除 HTML 标签
file	FILESYSTEM	将所有文件读入一个数组中
fileatime	FILESYSTEM	获得文件的最后访问时间
filectime	FILESYSTEM	获取文件的 I 节点的最后修改时间
filegroup	FILESYSTEM	获得文件的属组
fileinode	FILESYSTEM	获得文件的 i 节点
filemtime	FILESYSTEM	获得文件的修改时间
fileowner	FILESYSTEM	获得文件的属主
fileperms	FILESYSTEM	获得文件的许可权设置
filepro	FILEPRO	读并验证映像文件

Green Apple Data Center

续 表

函 数	组 名	描 述
filepro_fieldcount	FILEPRO	找出 filePro 数据库中有多少字段
filepro_fieldname	FILEPRO	获得一个字段的名字
filepro_fieldtype	FILEPRO	获得一个字段的类型
filepro_fieldwidth	FILEPRO	获得一个字段的宽度
filepro_retrieve	FILEPRO	从 filePro 数据库中获取数据
filepro_rowcount	FILEPRO	找出 filePro 数据库中有多少行
filesize	FILESYSTEM	获得文件大小
filetype	FILESYSTEM	获得文件类型
file_exists	FILESYSTEM	检查文件是否存在
flock	FILESYSTEM	可移植的文件加锁
Floor	MATH	去尾保留
flush	STRINGS	刷新输出缓冲区
fopen	FILESYSTEM	打开文件或 URL
fpassthru	FILESYSTEM	输出文件指针中的所有剩余数据
fputs	FILESYSTEM	向文件指针中写
fread	FILESYSTEM	二进制安全的文件读
FrenchToJD	CALENDAR	从法兰西共和历到罗马儒略历进行日期转换
fseek	FILESYSTEM	搜索文件指针
fsockopen	NETWORK	打开一个 Internet 或 UNIXdomain 的套接字链接
ftell	FILESYSTEM	返回文件指针的读写位置
function_exists	MISC	如果给定的函数被定义则返回真
fwrite	FILESYSTEM	二进制安全的文件写入
getallheaders	APACHE	取得所有 HTTP 请求 Header
getdate	DATETIME	获得日期/时间信息
getenv	INFO	获得环境变量的值
gethostbyaddr	NETWORK	从给定的 IP 地址获得 Internet 的主机名
gethostbyname	NETWORK	从给定的 Internet 的主机名获得其 IP 地址
gethostbynamel	NETWORK	获得对应一个 Internet 主机名的一个 IP 地址列表
GetLmageSize	IMAGE	获得 GIF、JPG 或 PNG 图像的大小
getlastmod	INFO	获得最后页面修改时间
getmxrr	NETWORK	获得给定 Internet 主机名的 MX 记录
getmyinode	INFO	获得当前脚本的 I 节点
getmypid	INFO	获得 PHP 进程的 ID
getmyuid	INFO	获得 PHP 脚本的属主 UID
getrandmax	MATH	显示最大可能的随机数
getrusage	INFO	获得当前资源用法
gettimeofday	DATETIME	获得当前时间
gettype	VAR	获得变量类型
get_cfg_var	INFO	获得 PHP 配置选项的值
get_current_user	INFO	获得当前 PHP 脚本的属主名字
get_magic_quotes_gpc	INFO	获得当前配置中有效的 magic_quotes_gpc 设置
get_magic_quotes_runtime	INFO	获得当前配置中有效的 magic_quotes_runtime 设置
get_meta_tags	STRINGS	从文件或返回的数组中展开所有的转换标签内容
gmdate	DATETIME	格式化 GMT/CUT 日期/时间
gmmktime	DATETIME	获得对应 GMT 日期的 Unix 时戳
GregorianToJD	CALENDAR	转换格里高利历到罗马儒略历
gzclose	ZLIB	关闭一个打开的 gz 文件

Green Apple Data Center

续 表

函 数	组 名	描 述
gzeof	ZLIB	测试 gz 文件的文件结束标志
gzfile	ZLIB	将所有 gz 文件读入数组中
gzgetc	ZLIB	从 gz 文件指针中读入字符
gzgets	ZLIB	从 gz 文件指针中读入一行
gzgetss	ZLIB	从 gz 文件指针中读入一行，并删除其中的 HTML 标签
gzopen	ZLIB	打开一个 gz 文件
gzpassthru	ZLIB	输出 gz 文件指针中的所有剩余数据
gzputs	ZLIB	向一个 gz 文件指针中写
gzread	ZLIB	从一个 gz 文件指针中读
gzrewind	ZLIB	重绕一个 gz 文件指针的位置
gzseek	ZLIB	搜索一个 gz 文件指针
gztell	ZLIB	返回一个 gz 文件的读写位置
gzwrite	ZLIB	二进制安全的 gz 文件写
header	HTTP	发送一个未加工的 HTTP 标题
HexDec	MATH	二进制到十进制转换
htmlentities	STRINGS	转换所有的必要字符为 HTML 实体
htmlspecialchars	STRINGS	转换指定的字符为 HTML 实体
hw_Children	HW	子对象的 ID
hw_ChildrenObj	HW	子对象的记录
hw_Close	HW	关闭链接
hw_Connect	HW	打开链接
hw_Cp	HW	复制对象
hw_Deleteobject	HW	删除对象
hw_DocByAnchor	HW	属于另一个标识的 ID 对象
hw_DocByAnchorObj	HW	属于另一个标识的记录对象
hw_DocumentAttributes	HW	hw_document 的记录对象
hw_DocumentBodyTag	HW	hw_document 的 body 标签
hw_DocumentContent	HW	返回 hw_document 的内容
hw_DocumentSetContent	HW	设置/替换 hw_document 的内容
hw_DocumentSize	HW	hw_document 的大小
hw_EditText	HW	获取 text 文档
hw_Error	HW	错误号
hw_ErrorMsg	HW	返回错误信息
hw_Free_Document	HW	释放 hw_document
hw_GetAnchors	HW	获取文档标识的 HTML 标记
hw_GetAnchorsObj	HW	获取文档记录对象
hw_GetAndLock	HW	返回记录对象和锁对象
hw_GetChildColl	HW	获取子集标识
hw_GetChildCollObj	HW	获取子集记录对象
hw_GetChildDocColl	HW	获取子文档
hw_GetChildDocCollObj	HW	获取子文档记录对象
hw_GetObject	HW	对象记录
hw_GetObjectByQuery	HW	搜寻对象
hw_GetObjectByQueryColl	HW	数集中的搜寻对象
hw_GetObjectByQueryCollObj	HW	数集中的搜寻对象
hw_GetObjectByQueryObj	HW	搜寻对象

Green Apple Data Center

续 表

函 数	组 名	描 述
hw_GetParents	HW	父对象标识
hw_GetParentsObj	HW	父记录对象
hw_GetRemote	HW	获取远程文档
hw_GetRemoteChildren	HW	获得远程子文档
hw_GetSrcByDestObj	HW	返回对象的 HTML 标记
hw_GetText	HW	获取文本文档
hw_Identify	HW	标识用户
hw_InCollections	HW	检查对象标识是否在数集中
hw_Info	HW	有关链接的信息
hw_InsColl	HW	插入数集
hw_InsDoc	HW	插入文档
hw_InsertDocument	HW	上载任一文档
hw_InsertObject	HW	插入对象记录
hw_Modifyobject	HW	修改对象记录
hw_Mv	HW	移动对象
hw_New_Document	HW	创建新文档
hw_Objrec2Array	HW	从对象记录到对象数组转换属性
hw_OutputDocument	HW	打印 hw_document
hw_pConnect	HW	建立永久数据库链接
hw_PipeDocument	HW	获取任一文档
hw_Root	HW	根对象标识
hw_Unlock	HW	解锁对象
hw_Username	HW	当前登录用户的名字
hw_Who	HW	当前登录用户的清单
ibase_close	IBASE	关闭链接
ibase_connect	IBASE	打开链接
ibase_execute	IBASE	执行链接
ibase_fetch_row	IBASE	返回字段
ibase_prepare	IBASE	分析 SQL 语法
ifxus_close_slob	IFX	删除 slob 对象
ifxus_create_slob	IFX	创建 slob 对象
ifxus_open_slob	IFX	打开 slob 对象
ifxus_read_slob	IFX	读取 slob 对象的 n 字节
ifxus_seek_slob	IFX	设置当前或搜寻的位置
ifxus_tell_slob	IFX	返回当前或搜寻的位置
ifxus_write_slob	IFX	将一个字符串写入 slob 对象
ifx_affected_rows	IFX	获得被查询影响的行
ifx_blobinfile_mode	IFX	设置 select 查询的缺省 blob 模式
ifx_byteasvarchar	IFX	设置缺省字节模式
ifx_close	IFX	关闭 Informix 链接
ifx_connect	IFX	打开 Informix 服务器链接
ifx_copy_blob	IFX	复制给定的 blob
ifx_create_blob	IFX	创建新的 blob 对象
ifx_create_char	IFX	创建新的字符对象
ifx_do	IFX	执行预准备的 SQL 语句
ifx_error	IFX	返回上次 Informix 调用的错误代码
ifx_errormsg	IFX	返回上次 Informix 调用的错误信息
ifx_fetch_row	IFX	返回字段

Green Apple Data Center

续 表

函 数	组 名	描 述
ifx_fieldproperties	IFX	列出 SQL 字段属性
ifx_fieldtypes	IFX	列出 Informix 的 SQL 字段
ifx_free_blob	IFX	删除 blob 对象
ifx_free_char	IFX	删除字符对象
ifx_free_result	IFX	释放查询资源
ifx_free_slob	IFX	删除 slob 对象
ifx_getsqlca	IFX	查询之后，获得 sqlca.sqlerrd[0..5]的链接
ifx_get_blob	IFX	返回 blob 对象的内容
ifx_get_char	IFX	返回字符对象的内容
ifx_htmltbl_result	IFX	格式化所有的查询行为一个 HTML 表格
ifx_nullformat	IFX	设置取回行时的缺省返回值
ifx_num_fields	IFX	返回查询中列的数目
ifx_num_rows	IFX	返回查询已经取到的行的数目
ifx_pconnect	IFX	打开一个永久链接
ifx_prepare	IFX	准备 SQL 语句以便执行
ifx_query	IFX	发送 Informix 查询
ifx_textasvarchar	IFX	设置缺省文本模式
ifx_update_blob	IFX	更新 blob 对象类型
ifx_update_char	IFX	更新字符对象类型
ignore_user_abort	MISC	设置一个客户在脚本执行完毕之后是否断开
ImageArc	IMAGE	画弧线
ImageChar	IMAGE	水平画字
ImageCharUp	IMAGE	垂直画字
ImageColorAllocate	IMAGE	为一个图像分配颜色
ImageColorAt	IMAGE	获得一个像素的颜色索引
ImageColorClosest	IMAGE	获得指定颜色的最相近颜色索引
ImageColorExact	IMAGE	获得指定颜色的索引
ImageColorResolve	IMAGE	获得指定颜色的索引或最接近的值
ImageColorSet	IMAGE	设置指定调色板索引的颜色
ImageColorsForIndex	IMAGE	获得索引的颜色
ImageColorsTotal	IMAGE	找出一个图像调色板的颜色数
ImageColorTransparent	IMAGE	定义一种颜色为透明色
ImageCopyResized	IMAGE	复制并缩放图像
ImageCreate	IMAGE	创建图像
ImageCreateFromGif	IMAGE	从文件或 URL 中创建图像
ImageDashedLine	IMAGE	画虚线
ImageDestroy	IMAGE	删除一个图像
ImageFill	IMAGE	填充
ImageFilledPolygon	IMAGE	画一个填充的多边形
ImageFilledRectangle	IMAGE	画一个填充的矩形
ImageFillToBorder	IMAGE	填充一个指定的颜色
ImageFontHeight	IMAGE	获得字体的高度
ImageFontWidth	IMAGE	获得字体的宽度
ImageGif	IMAGE	向浏览器或文件输出图像
ImageInterlace	IMAGE	许可或屏蔽交错显示
ImageLine	IMAGE	画一条线
ImageLoadFont	IMAGE	载入一个新字体

Green Apple Data Center

续 表

函 数	组 名	描 述
ImagePolygon	IMAGE	画一个多边形
ImagePSBBox	IMAGE	给出使用 PostScriptType1 字体的文本边界
ImagePSCopyFont	IMAGE	复制一个已经载入的字体以便进一步修改
ImagePSEncodeFont	IMAGE	改变一个字体向量的字符编码
ImagePSFreeFont	IMAGE	释放 PostScriptType1 字体占用的内容
ImagePSLoadFont	IMAGE	从文件中载入 PostScriptType1
ImagePSText	IMAGE	使用 PostScriptType1 字体画一串字符串
ImageRectangle	IMAGE	画矩形
ImageSetPixel	IMAGE	画一个像素
ImageString	IMAGE	水平画一个字符串
ImageStringUp	IMAGE	垂直画一个字符串
ImageSX	IMAGE	获得图像宽度
ImageSY	IMAGE	获得图像高度
ImageTTFBBox	IMAGE	给定使用 TrueType 字体的文本的边界
ImageTTFText	IMAGE	使用 TrueType 字体将文本写成图像
imap_8bit	IMAP	转变 8 位字符串为 quoted-printable 字符串
imap_append	IMAP	在一个指定的邮箱中添加一个字符串信息
imap_base64	IMAP	解开 base64 编码的文本
imap_binary	IMAP	转变 8 位字符串为 base64 编码的字符串
imap_body	IMAP	读取信息正文
imap_check	IMAP	查看当前邮箱文件夹
imap_clearflag_full	IMAP	清除信息标志
imap_close	IMAP	关闭 IMAP 流
imap_createmailbox	IMAP	创建新邮箱文件夹
imap_delete	IMAP	在当前邮箱文件夹中标记某个信息为删除
imap_deletemailbox	IMAP	删除一个邮箱文件夹
imap_expunge	IMAP	删除所有标记为删除的信息
imap_fetchbody	IMAP	取回一个信息正文的特定部分
imap_fetchstructure	IMAP	读取特定信息的结构
imap_header	IMAP	读取信息的题头
imap_headers	IMAP	读取邮箱中所有信息的题头
imap_listmailbox	IMAP	列出所有邮箱文件夹列表
imap_listsubscribed	IMAP	列出所有预订的邮箱文件夹
imap_mailboxmsginfo	IMAP	获取当前邮箱文件夹的信息
imap_mail_copy	IMAP	复制指定的信息到一个邮箱文件夹
imap_mail_move	IMAP	移动指定的信息到一个邮箱文件夹
imap_num_msg	IMAP	给出当前邮箱文件夹中的信息数量
imap_num_recent	IMAP	给出当前邮箱文件夹中的新邮件数量
imap_open	IMAP	打开一个 IMAP 链接
imap_ping	IMAP	测试 IMAP 链接是否正常
imap_qprint	IMAP	转变 quoted-printable 编码的字符串到 8 位字符串
imap_renamemailbox	IMAP	更改邮箱文件夹的名字
imap_reopen	IMAP	为一个新的邮箱文件夹重新打开一个 IMAP 链接
imap_rfc822_parse_adrlist	IMAP	分析地址字符串
imap_rfc822_write_address	IMAP	给定主机、人员和邮箱文件夹信息，返回正确格式的邮件地址
imap_scanmailbox	IMAP	读取邮箱文件夹的列表，在文件夹的内容中搜寻一个给定字符串的内容

Green Apple Data Center

续 表

函 数	组 名	描 述
imap_setflag_full	IMAP	设置信息的标志
imap_sort	IMAP	返回按给定参数排序的一个信息号数组
imap_subscribe	IMAP	预订一个邮箱文件夹
imap_uid	IMAP	返回给定信息序列号的 UID
imap_undelete	IMAP	对一个标记为删除的信息，取消其删除标记
imap_unsubscribe	IMAP	取消一个预订的邮箱文件夹
implode	STRINGS	链接一个数组中的元素
intval	VAR	获得变量的整数值
iptcparse	MISC	分析一个二进制的 IPTChttp://www.xe.net/iptc/块为单个标签
isset	VAR	判断变量是否被设置
is_array	VAR	判断变量是否是数组
is_dir	FILESYSTEM	判断文件名是否是目录
is_double	VAR	判断变量是否为双精度数
is_executable	FILESYSTEM	判断文件是否可执行
is_file	FILESYSTEM	判断给定的名字是否为正常的文件
is_float	VAR	判断变量是否为浮点数
is_int	VAR	判断变量是否为整数
is_integer	VAR	判断变量是否为长整数
is_link	FILESYSTEM	判断文件名是否为符号链接
is_long	VAR	判断变量是否为整数
is_object	VAR	判断变量是否为对象
is_readable	FILESYSTEM	判断文件是否可读
is_real	VAR	判断变量是否为实数
is_string	VAR	判断变量是否为字符串
is_writeable	FILESYSTEM	判断变量是否可写
JDDayOfWeek	CALENDAR	返回星期几
JDMonthName	CALENDAR	返回月份的名字
JDToFrench	CALENDAR	罗马儒略历转变为法兰西共和历
JDToGregorian	CALENDAR	罗马儒略历转变为格里高利历
JDToJewish	CALENDAR	罗马儒略历转变为犹太历
JDToJulian	CALENDAR	罗马儒略历的日期转变为天数
JewishToJD	CALENDAR	犹太历转变为罗马儒略历
join	STRINGS	链接数组中的元素
JulianToJD	CALENDAR	罗马儒略历的日期转变为天数
key	ARRAY	获得与一个值相关联的下标
ksort	ARRAY	通过下标排序一个数组
ldap_add	LDAP	向 LDAP 目录中添加一个实体
ldap_bind	LDAP	绑定在 LDAP 目录上
ldap_close	LDAP	关闭与 LDAP 服务器的链接
ldap_connect	LDAP	链接 LDAP 服务器
ldap_count_entries	LDAP	返回搜寻到的实体数量
ldap_delete	LDAP	从目录中删除一个实体
ldap_dn2ufn	LDAP	转变 DN 为用户友好的名字
ldap_explode_dn	LDAP	将 DN 分割为其各个组成部分
ldap_first_attribute	LDAP	返回第一个属性
ldap_first_entry	LDAP	返回第一个实体

Green Apple Data Center

续 表

函 数	组 名	描 述
ldap_free_result	LDAP	释放结果内存
ldap_get_attributes	LDAP	获取一个搜寻结果实体的属性
ldap_get_dn	LDAP	获得结果实体的 DN
ldap_get_entries	LDAP	获得所有的结果实体
ldap_get_values	LDAP	获得一个结果实体的所有值
ldap_list	LDAP	单级搜寻
ldap_modify	LDAP	修改 LDAP 实体
ldap_mod_add	LDAP	添加当前属性的属性值
ldap_mod_del	LDAP	删除当前属性的属性值
ldap_mod_replace	LDAP	使用新值替换属性值
ldap_next_attribute	LDAP	获得结果中的下一个属性
ldap_next_entry	LDAP	获得结果中的下一个实体
ldap_read	LDAP	读取一个实体属性
ldap_search	LDAP	搜寻 LDAP 目录
ldap_unbind	LDAP	解开与 LDAP 目录的绑定
leak	MISC	泄漏内存
link	FILESYSTEM	创建一个文件的硬链接
linkinfo	FILESYSTEM	获得一个链接的信息
list	ARRAY	以数组的方式给变量赋值
Log	MATH	自然对数
Log10	MATH	以 10 为底的对数
lstat	FILESYSTEM	获得一个文件或符号链接的信息
ltrim	STRINGS	从字符串的开头开始删除空白字符
mail	MAIL	发送邮件
max	MATH	找出最大值
mcrypt_cbc	MCRYPT	对数据使用 CBC 模式进行加密/解密
mcrypt_cfb	MCRYPT	对数据使用 CFB 模式进行加密/解密
mcrypt_create_iv	MCRYPT	随机地创建一个初始向量
mcrypt_ecb	MCRYPT	对数据使用 ECB 模式进行加密/解密
mcrypt_get_block_size	MCRYPT	获得指定加密算法的块长度
mcrypt_get_cipher_name	MCRYPT	获得指定加密算法的名字
mcrypt_get_key_size	MCRYPT	获得指定加密算法的密钥长度
mcrypt_ofb	MCRYPT	对数据使用 OFB 模式进行加密/解密
md5	STRINGS	计算一个字符串的 md5 哈希值
mhash	MHASH	计算哈希值
mhash_count	MHASH	获得最大的哈希 ID
mhash_get_block_size	MHASH	获得指定哈希的块大小
mhash_get_hash_name	MHASH	获得指定哈希的名字
microtime	DATETIME	使用毫秒返回当前 Unix 时戳
min	MATH	找出最小值
mkdir	FILESYSTEM	建立目录
mktime	DATATIME	从一个日期中获得 Unix 时戳
msql	MSQL	发送 msql 查询
msql_affected_rows	MSQL	返回受影响行的数量
msql_close	MSQL	关闭 msql 链接
msql_connect	MSQL	打开 msqlL 链接
msql_createdb	MSQL	创建 msql 数据库

Green Apple Data Center

续 表

函 数	组 名	描 述
msql_create_db	MSQL	创建 msql 数据库
msql_data_seek	MSQL	移动内部行指针
msql_dbname	MSQL	获得 msql 的内部数据库名字
msql_dropdb	MSQL	删除 msql 数据库
msql_drop_db	MSQL	删除 msql 数据库
msql_error	MSQL	返回最后一个 msql 调用的错误信息
msql_fetch_array	MSQL	将行作为数组取回
msql_fetch_field	MSQL	获取字段信息
msql_fetch_object	MSQL	将行作为对象取回
msql_fetch_row	MSQL	将行作为枚举数组取回
msql_fieldflags	MSQL	获得字段标志
msql_fieldlen	MSQL	获得字段长度
msql_fieldname	MSQL	获得字段名字
msql_fieldtable	MSQL	获得字段的表名字
msql_fieldtype	MSQL	获得字段类型
msql_field_seek	MSQL	设置字段偏移
msql_freeresult	MSQL	释放结果内存
msql_free_result	MSQL	释放结果内存
msql_listdbs	MSQL	列出服务器上的 MSQL 数据库
msql_listfields	MSQL	列出结果字段
msql_listtables	MSQL	列出一个 MSQL 数据库中的表
msql_list_dbs	MSQL	列出服务器上的 MSQL 数据库
msql_list_fields	MSQL	列出结果字段
msql_list_tables	MSQL	列出一个 MSQL 数据库中的表
msql_numfields	MSQL	获得结果中字段的数量
msql_numrows	MSQL	获得结果中行的数量
msql_num_fields	MSQL	获得结果中字段的数量
msql_num_rows	MSQL	获得结果中行的数量
msql_pconnect	MSQL	设置永久链接
msql_query	MSQL	发送 MSQL 查询
msql_regcase	MSQL	构成大小写不敏感的正则表达式
msql_result	MSQL	获得结果数据
msql_selectdb	MSQL	选择 MSQL 数据库
msql_select_db	MSQL	选择 MSQL 数据库
msql_tablename	MSQL	获得字段的表名字
mssql_affected_rows	MSSQL	获得最后一次查询中受影响的行
mssql_close	MSSQL	关闭与 MSSQL 服务器的链接
mssql_connect	MSSQL	打开与 MSSQL 服务器的链接
mssql_data_seek	MSSQL	移动内部行指针
mssql_fetch_array	MSSQL	将行作为数组取回
mssql_fetch_field	MSSQL	获取字段信息
mssql_fetch_object	MSSQL	将行作为对象取回
mssql_fetch_row	MSSQL	将行作为枚举数组取回
mssql_field_seek	MSSQL	设置字段偏移
mssql_free_result	MSSQL	释放结果内存
mssql_num_fields	MSSQL	获得结果中字段的数量
mssql_num_rows	MSSQL	获得结果中行的数量

Green Apple Data Center

续 表

函 数	组 名	描 述
mssql_pconnect	MSSQL	设置永久链接
mssql_query	MSSQL	发送 MSSQL 查询
mssql_result	MSSQL	获得结果数据
mssql_select_db	MSSQL	选择 MSSQL 数据库
mt_getrandmax	MATH	显示最大可能的随机值
mt_rand	MATH	产生一个更好的随机值
mt_srand	MATH	使用种子初始化随机数
mysql_affected_rows	MYSQL	返回受影响行的数量
mysql_close	MYSQL	关闭 MySQL 链接
mysql_connect	MYSQL	打开 MySQL 链接
mysql_create_db	MYSQL	创建 MySQL 数据库
mysql_data_seek	MYSQL	移动内部行指针
mysql_dbname	MYSQL	获得 MySQL 的内部数据库名字
mysql_db_query	MYSQL	向 MySQL 数据库发送 MySQL 请求
mysql_drop_db	MYSQL	删除 MySQL 数据库
mysql_errno	MYSQL	返回最后一个 MySQL 调用产生的错误信息号
mysql_error	MYSQL	返回最后一个 MySQL 调用的错误信息文本
mysql_fetch_array	MYSQL	将行作为数组取回
mysql_fetch_field	MYSQL	获取字段信息
mysql_fetch_lengths	MYSQL	获得结果中输出数据的最大长度
mysql_fetch_object	MYSQL	将行作为对象取回
mysql_fetch_row	MYSQL	将行作为枚举数组取回
mysql_field_flags	MYSQL	获得字段标志
mysql_field_len	MYSQL	获得字段长度
mysql_field_name	MYSQL	获得字段名字
mysql_field_seek	MYSQL	设置字段偏移
mysql_field_table	MYSQL	获得字段的表名字
mysql_field_type	MYSQL	获得字段类型
mysql_free_result	MYSQL	释放结果内存
mysql_insert_id	MYSQL	获得前一次 INSERT 操作产生的标识
mysql_list_dbs	MYSQL	列出服务器上的 MySQL 数据库
mysql_list_fields	MYSQL	列出结果字段
mysql_list_tables	MYSQL	列出一个 MySQL 数据库中的表
mysql_num_fields	MYSQL	获得结果中字段的数量
mysql_num_rows	MYSQL	获得结果中行的数量
mysql_pconnect	MYSQL	设置永久链接
mysql_query	MYSQL	发送 MySQL 查询
mysql_result	MYSQL	获得结果数据
mysql_select_db	MYSQL	选择 MySQL 数据库
mysql_tablename	MYSQL	获得字段的表名字
next	ARRAY	向前移动数组的内部指针
nl2br	STRINGS	转变换行符为 HTML 换行标签
number_format	MATH	用三位（每千）为一组格式化数字
OCIBindByName	OCI8	将 PHP 变量与 Oracle 占位符绑定在一起
OCIColumnIsNULL	OCI8	测试一结果列是否为空值
OCIColumnSize	OCI8	返回列的大小
OCICommit	OCI8	提交未确认的事务

Green Apple Data Center

续 表

函 数	组 名	描 述
OCIDefineByName	OCI8	在 SELECT 中使用 PHP 变量作为定义步骤
OCIExecute	OCI8	执行一个语句
OCIFetch	OCI8	获得下一行放入结果缓冲中
OCIFetchInfo	OCI8	获得下一行放入结果数组中
OCILogOff	OCI8	断开与 Oracle 的链接
OCILogon	OCI8	设立与 Oracle 的链接
OCINumRows	OCI8	获得受影响的行
OCIResult	OCI8	获得取回的行的列值
OCIRollback	OCI8	取消未提交的事务
OctDec	MATH	八进制到十进制转换
odbc_autocommitUOD	BC	确认自动提交模式
odbc_binmodeUOD	BC	使用二进制方式处理列数据
odbc_closeUOD	BC	关闭 ODBC 链接
odbc_close_allUOD	BC	关闭所有的 ODBC 链接
odbc_commitUOD	BC	提交一个 ODBC 事务
odbc_connectUOD	BC	打开 ODBC 链接
odbc_cursorUOD	BC	获得游标名字
odbc_doUOD	BC	odbc_exec 的同义词
odbc_execUOD	BC	准备并执行一次 SQL 语句
odbc_executeUOD	BC	执行一个准备好的语句
odbc_fetch_intoUOD	BC	将一个结果行取到数组中
odbc_fetch_rowUOD	BC	取回一行
odbc_field_lenUOD	BC	获得字段长度
odbc_field_nameUOD	BC	获得字段名字
odbc_field_numUOD	BC	获得列的数量
odbc_field_typeUOD	BC	获得字段类型
odbc_free_resultUOD	BC	释放结果内存
odbc_longreadlenUOD	BC	处理 LONG 列
odbc_num_fieldsUOD	BC	结果中列的数量
odbc_num_rowsUOD	BC	获得结果中行的数量
odbc_pconnectUOD	BC	设置永久链接
odbc_prepareUOD	BC	准备一个语句以便执行
odbc_resultUOD	BC	获得结果数据
odbc_result_allUOD	BC	以 HTML 表单的形式打印结果数据
odbc_rollbackUOD	BC	取消一次事务
odbc_setoptionUOD	BC	调整 ODBC 设置，如果发生错误则返回假，否则返回真
opendir	DIR	打开一个目录句柄
openlog	NETWORK	打开与系统日志的链接
Ora_Bind	ORACLE	将 PHP 变量与 Oracle 参数绑定在一起
Ora_Close	ORACLE	关闭一个 Oracle 游标
Ora_ColumnName	ORACLE	获得 Oracle 结果列的名字
Ora_ColumnType	ORACLE	获得 Oracle 结果列的类型
Ora_Commit	ORACLE	提交一个 Oracle 事务
Ora_CommitOff	ORACLE	屏蔽自动提交
Ora_CommitOn	ORACLE	打开自动提交
Ora_Error	ORACLE	获得 Oracle 错误信息

Green Apple Data Center

续 表

函 数	组 名	描 述
Ora_ErrorCode	ORACLE	获得 Oracle 错误代码
Ora_Exec	ORACLE	在一个 Oracle 游标上执行一个分析过的语句
Ora_Fetch	ORACLE	从一个游标中取回数据的一行
Ora_GetColumn	ORACLE	从一个取回的行中获取数据
Ora_Logoff	ORACLE	关闭 Oracle 链接
Ora_Logon	ORACLE	打开一个 Oracle 链接
Ora_Open	ORACLE	打开一个 Oracle 游标
Ora_Parse	ORACLE	分析一个 Oracle 语句
Ora_Rollback	ORACLE	取消一次事务
Ord	STRINGS	返回字符的 ASCII 值
pack	MISC	将数据压紧进二进制字符串中
parse_str	STRINGS	分析字符串
parse_url	URL	分析 URL 并返回组件
passthru	EXEC	执行外部程序并显示行输出
PatternOptions	PCRE	描述可能的 regex 模式选项
PatternSyntax	PCRE	描述 PCREregex 语法
pclose	FILESYSTEM	关闭进程文件指针
pdf_add_annotation	PDF	增加一个记号
PDF_add_outline	PDF	为当前页增加一个标签
PDF_arc	PDF	画弧
PDF_begin_page	PDF	开始一个新页
PDF_circle	PDF	画圆
PDF_clip	PDF	剪切当前路线
PDF_close	PDF	关闭 pdf 文档
PDF_closepath	PDF	闭合路线
PDF_closepath_fill_stroke	PDF	闭合、填充并画出当前路线
PDF_closepath_stroke	PDF	闭合并画出当前路线
PDF_close_image	PDF	关闭一个图像
PDF_continue_text	PDF	在下一行输出文本
PDF_curveto	PDF	画一条曲线
PDF_endpath	PDF	终止当前路线
PDF_end_page	PDF	结束一页
PDF_execute_image	PDF	在一页中放置一个保存好的图像
PDF_fill	PDF	填充当前路线
PDF_fill_stroke	PDF	填充并画出当前路线
PDF_get_info	PDF	获得 pdf 文档的缺省信息
PDF_lineto	PDF	画线
PDF_moveto	PDF	移动当前位置
PDF_open	PDF	打开 pdf 文档
PDF_open_gif	PDF	打开 GIF 图像
PDF_open_jpeg	PDF	打开 JPG 图像
PDF_open_memory_image	PDF	使用 PHP 的图像函数创建图像
PDF_place_image	PDF	在页面上放置图像
PDF_put_image	PDF	在文档中保存图像以便以后使用
PDF_rect	PDF	画矩形
PDF_restore	PDF	恢复从前保存的环境
PDF_rotate	PDF	设置旋转

Green Apple Data Center

续 表

函 数	组 名	描 述
PDF_save	PDF	保存当前环境设置
PDF_scale	PDF	设置比例尺
PDF_setdash	PDF	设置虚线的模式
PDF_setflat	PDF	设置为平滑线条
PDF_setgray	PDF	设置画图和填充颜色的灰度值
PDF_setgray_fill	PDF	设置填充颜色的灰度值
PDF_setgray_stroke	PDF	设置画图颜色的灰度值
PDF_setlinecap	PDF	设置 linecap 参数
PDF_setlinejoin	PDF	设置 linejoin 参数
PDF_setlinewidth	PDF	设置线宽
PDF_setmiterlimit	PDF	设置斜线宽
PDF_setrgbcolor	PDF	设置画线和填充颜色的 rgb 值
PDF_setrgbcolor_fill	PDF	设置填充颜色的 rgb 值
PDF_setrgbcolor_stroke	PDF	设置画线颜色的 rgb 值
PDF_set_char_spacing	PDF	设置字符间隔
PDF_set_duration	PDF	设置页面之间的持续时间
PDF_set_font	PDF	设置当前字体的外观和大小
PDF_set_horiz_scaling	PDF	设置水平比例
PDF_set_info_author	PDF	设置信息结构中的作者域
PDF_set_info_creator	PDF	设置信息结构中的创建者域
PDF_set_info_keywords	PDF	设置信息结构中的关键词域
PDF_set_info_subject	PDF	设置信息结构中的主题域
PDF_set_info_title	PDF	设置信息结构中的标题域
PDF_set_leading	PDF	设置行距
PDF_set_text_matrix	PDF	设置文本矩阵
PDF_set_text_pos	PDF	设置文本位置
PDF_set_text_rendering	PDF	设置文本表现的方式
PDF_set_text_rise	PDF	设置文本高度
PDF_set_transition	PDF	设置页面之间的转换
PDF_set_word_spacing	PDF	设置词之间的空间
PDF_show	PDF	在当前位置上输出文本
PDF_show_xy	PDF	在 xy 位置输出文本
PDF_stringwidth	PDF	以当前字体返回文本的宽度
PDF_stroke	PDF	沿路线画线
PDF_translate	PDF	设置坐标原点
pfsockopen	NETWORK	打开一个永久 Internet 或 Unix 域套接字链接
pg_Close	PGSQL	关闭 PostgreSQL 链接
pg_cmdTuples	PGSQL	返回受影响元组的数量
pg_Connect	PGSQL	打开链接
pg_DBname	PGSQL	数据库名字
pg_ErrorMessage	PGSQL	错误信息
pg_Exec	PGSQL	执行一次查询
pg_Fetch_Array	PGSQL	以数组方式取回行
pg_Fetch_Object	PGSQL	以对象方式取回行
pg_Fetch_Row	PGSQL	以枚举数组方式获取行
pg_FieldIsNull	PGSQL	测试字段是否为空
pg_FieldName	PGSQL	返回字段名

Green Apple Data Center

续 表

函 数	组 名	描 述
pg_FieldNum	PGSQL	返回字段号
pg_FieldPrtLen	PGSQL	返回字段中可打印的长度
pg_FieldSize	PGSQL	返回命名字段中的内部存储的大小
pg_FieldType	PGSQL	返回相应字段号的类型名
pg_FreeResult	PGSQL	释放内存
pg_GetLastOid	PGSQL	返回最后一个对象标识符
pg_Host	PGSQL	返回主机名
pg_loclose	PGSQL	关闭一个大对象
pg_locreate	PGSQL	创建一个大对象
pg_loopen	PGSQL	打开一个大对象
pg_loread	PGSQL	读取一个大对象
pg_loreadall	PGSQL	读取一个完整的大对象
pg_lounlink	PGSQL	删除一个大对象
pg_lowrite	PGSQL	保存一个大对象
pg_NumFields	PGSQL	返回字段号
pg_NumRows	PGSQL	返回行号
pg_Options	PGSQL	返回一个选项
pg_pConnect	PGSQL	创建一个永久的数据库链接
pg_Port	PGSQL	返回端口号
pg_Result	PGSQL	返回结果标识符的值
pg_tty	PGSQL	返回 tty 名字
phpinfo	INFO	输出 PHP 信息
phpversion	INFO	输出 PHP 版本
pi	MATH	圆周率值
popen	FILESYSTEM	打开一个进程文件描述符
pos	ARRAY	从一个数组中获取当前元素
pow	MATH	幂运算
preg_match	PCRE	执行正则表达匹配
preg_match_all	PCRE	执行全局正则表达匹配
preg_replace	PCRE	执行正则搜索并替换
preg_split	PCRE	使用正则表达式分割字符串
prev	ARRAY	回卷内部数组指针
print	STRINGS	输出字符串
printf	STRINGS	输出格式化的字符串
putenv	INFO	设置环境变量的值
quoted_printable_decode	STRINGS	转变一个 quoted-printable 字符串到 8 位字符串
rand	MATH	产生一个随机值
range	ARRAY	创建一个包含一定范围整数的数组
rawurldecode	STRINGS	解开 URL 编码的字符串
rawurlencode	STRINGS	按照 RFC1738 对 URL 进行编码
readdir	DIR	从目录句柄中读取一个入口
readfile	FILESYSTEM	输出一个文件
readgzfile	ZLIB	输出一个 gz 文件
readlink	FILESYSTEM	返回一个符号链接的目标
register_shutdown_function	MISC	登记在停止时执行的函数
rename	FILESYSTEM	更改一个文件的名字
reset	ARRAY	设置一个数组的内部指针为其第一个元素

Green Apple Data Center

续 表

函 数	组 名	描 述
rewind	FILESYSTEM	回卷文件指针位置
rewinddir	DIR	回卷目录句柄
rmdir	FILESYSTEM	删除目录
round	MATH	对浮点数进行四舍五入操作
rsort	ARRAY	以反序排列一个数组
sem_acquire	SEM	要求一个信号灯(semaphore)
sem_get	SEM	获得一个信号灯的标识符
sem_release	SEM	释放一个信号灯
serialize	MISC	产生一个值的可存储表示方式
setcookie	HTTP	发送一个 cookie
setlocale	STRINGS	设置场所信息
settype	VAR	设置变量的类型
set_file_buffer	FILESYSTEM	设置规定文件指针的缓冲区
set_magic_quotes_runtime	INFO	设置 magic_quotes_runtime 的当前设置
set_socket_blocking	NETWORK	设置一个套接字的阻塞/非阻塞模式
set_time_limit	INFO	设置最大执行时间
shm_attach	SEM	创建或打开一个共享内存段
shm_detach	SEM	从一个共享内存段断开
shm_get_var	SEM	从共享内存中返回一个值
shm_put_var	SEM	在共享内存中设置或更新一个值
shm_remove	SEM	从 Unix 系统中移去共享内存
shm_remove_var	SEM	从共享内存中移去变量
shuffle	ARRAY	遍历一个数组
similar_text	STRINGS	在两个字符串中计算相似性
Sin	MATH	正弦
sizeof	ARRAY	返回数组元素的大小
sleep	MISC	延迟执行
snmpget	SNMP	取回一个 SNMP 对象
snmpwalk	SNMP	从代理中取回所有 SNMP 对象
snmpwalkoid	SNMP	查询一个网络实体的信息树
snmp_get_quick_print	SNMP	取回 UCD 库 quick_print 设置的当前值
snmp_set_quick_print	SNMP	设置 UCD 库 quick_print 的当前值
solid_close	SOLID	关闭 Solid 链接
solid_connect	SOLID	链接一个 Solid 数据源
solid_exec	SOLID	执行一个 Solid 查询
solid_fetchrow	SOLID	从 Solid 查询中取回数据行
solid_fieldname	SOLID	从 Solid 查询中取回列名
solid_fieldnum	SOLID	从 Solid 查询中取回索引列
solid_freeresult	SOLID	释放内存
solid_numfields	SOLID	获取查询结果字段的数量
solid_numrows	SOLID	获取查询结果行的数量
solid_result	SOLID	从查询结果中获取数据
sort	ARRAY	排序一个数组
soundex	STRINGS	计算字符串的 soundex 键值
split	REGEX	用正则表达式分割字符串为数组
sprintf	STRINGS	返回格式化的字符串
sql_regcase	REGEX	创建正则表达式，以进行大小写不敏感的匹配

Green Apple Data Center

续 表

函 数	组 名	描 述
Sqrt	MATH	平方根
srand	MATH	初始化随机数产生器
stat	FILESYSTEM	取回一个文件信息
strchr	STRINGS	找到参数字符第一次出现的位置
strcmp	STRINGS	比较两个字符串
strcspn	STRINGS	找出不匹配掩码的初始段的长度
strftime	DATETIME	按照场所设置格式化本地时间/日期
StripSlashes	STRINGS	使用附加的斜线使括住的字符串不用使用括号
strip_tags	STRINGS	从字符串中去除 HTML 和 PHP 标签
strlen	STRINGS	返回字符串长度
strpos	STRINGS	找出字符第一次出现的位置
strrchr	STRINGS	找出一个字符串中最后一次出现的字符
strrev	STRINGS	反转一个字符串
strrpos	STRINGS	找出一个字符串中一个字符最后一次出现的位置
strspn	STRINGS	找出匹配掩码的初始段的长度
strstr	STRINGS	找出一个字符串的第一个出现
strtok	STRINGS	将字符串转变为记号
strtolower	STRINGS	使字符串转变为小写
strtoupper	STRINGS	使字符串转变为大写
strtr	STRINGS	翻译某些字符
strval	VAR	获得一个变量的字符串值
str_replace	STRINGS	字符串替换
substr	STRINGS	返回字符串的一部分
sybase_affected_rows	SYBASE	返回受影响行的数量
sybase_close	SYBASE	关闭 Sybase 链接
sybase_connect	SYBASE	打开 Sybase 链接
sybase_data_seek	SYBASE	移动内部行指针
sybase_fetch_array	SYBASE	将行作为数组取回
sybase_fetch_field	SYBASE	获取字段信息
sybase_fetch_object	SYBASE	将行作为对象取回
sybase_fetch_row	SYBASE	将行作为枚举数组取回
sybase_field_seek	SYBASE	设置字段偏移
sybase_free_result	SYBASE	释放结果内存
sybase_num_fields	SYBASE	获得结果中字段的数量
sybase_num_rows	SYBASE	获得结果中行的数量
sybase_pconnect	SYBASE	设置永久链接
sybase_query	SYBASE	发送 sybase 查询
sybase_result	SYBASE	获得结果数据
sybase_select_db	SYBASE	选择 sybase 数据库
symlink	FILESYSTEM	创建符号链接
syslog	NETWORK	产生系统日志信息
system	EXEC	执行一个外部程序并显示输出
Tan	MATH	正切
tempnam	FILESYSTEM	创建一个唯一的文件名
time	DATETIME	返回当前 Unix 时戳
touch	FILESYSTEM	设置文件的修改时间
trim	STRINGS	从字符串开始到尾部，删除其中的空白字符

Green Apple Data Center

续 表

函 数	组 名	描 述
uasort	ARRAY	使用自定义的比较函数对数组排序，并维护相关索引
ucfirst	STRINGS	使字符串的首字符大写
ucwords	STRINGS	使字符串每个单词的首字符大写
uksort	ARRAY	对于键，使用自定义的函数来排序数组
umask	FILESYSTEM	改变当前掩码
uniqid	MISC	产生一个唯一的标识符
unlink	FILESYSTEM	删除一个文件
unpack	MISC	从二进制字符串中解开数据
unserialize	MISC	从存储的表示方式中创建 PHP 变量
unset	VAR	取消一个变量的定义
urldecode	URL	解开一个 URL 编码的字符串
urlencode	URL	URL 编码的字符串
usleep	MISC	以毫秒为单位延迟执行
usort	ARRAY	对于值，使用自定义的函数来排序数组
utf8_decode	XML	转变 UTF-8 编码的字符串为 ISO-8859-1
utf8_encode	XML	编码 ISO-8859-1 字符串为 UTF-8
virtual	APACHE	执行 Apache 子请求
vm_addaliasV	MAIL	MGR 对一个虚拟用户增加一个别名
vm_adduserV	MAIL	MGR 增加一个带口令的新虚拟用户
vm_delaliasV	MAIL	MGR 删除一个别名
vm_deluserV	MAIL	MGR 删除一个虚拟用户
vm_passwdV	MAIL	MGR 改变虚拟用户的口令
wddx_add_vars	WDDX	使用指定的 ID 终止一个 WDDX 包
wddx_deserialize	WDDX	描述一个 WDDX 包
wddx_packet_end	WDDX	使用指定的 ID 终止一个 WDDX 包
wddx_packet_start	WDDX	开始一个新的 WDDX 包
wddx_serialize_value	WDDX	连续化一个单值为一个 WDDX 包
wddx_serialize_vars	WDDX	连续化变量为一个 WDDX 包
xml_error_string	XML	获得 XML 分析器错误字符串
xml_get_current_byte_index	XML	获得 XML 分析器的当前字节索引
xml_get_current_column_number	XML	获得 XML 分析器的当前列号
xml_get_current_liner_numbe	XML	获得 XML 分析器的当前行号
xml_get_error_code	XML	获得 XML 分析器错误码
xml_parse	XML	开始分析一个 XML 文档
xml_parser_create	XML	创建一个 XML 分析器
xml_parser_free	XML	释放 XML 分析器
xml_parser_get_option	XML	获得 XML 分析器的选项
xml_parser_set_option	XML	设置 XML 分析器的选项
xml_set_character_data_handler	XML	设置字符串数据处理程序
xml_set_default_handler	XML	设置缺省的处理程序
xml_set_element_handler	XML	设置开始和终止元素处理程序
xml_set_external_entity_ref_handler	XML	设置外部实体引用处理程序
xml_set_notation_decl_handler	XML	设置注释声明处理程序
xml_set_processing_instruction_handler	XML	设置进程指令处理程序

Green Apple Data Center

续 表

函 数	组 名	描 述
xml_set_unparsed_entity_decl_handler	XML	设置未分析实体声明处理程序
yp_errno	NIS	返回前一步操作的错误码
yp_err_string	NIS	返回与前一步操作相联系的错误字符串
yp_first	NIS	返回命名映射的第一个键/值对
yp_get_default_domain	NIS	获得主机的缺省 NIS 域
yp_master	NIS	从映射中返回主 NIS 服务器的机器名
yp_match	NIS	返回匹配行
yp_next	NIS	返回命名映射的下一个键/值对
yp_order	NIS	返回映射的序号

Green Apple Data Center

附录 B Internet 资源

读完本书之后，读者已全面学习了 PHP，这里介绍一些国内外有用的资源，帮读者进一步提高，知道在哪里能找得到相关的信息以及 PHP 脚本，以便改动之后适应自己的需要。

国外资源

（1）http://www.php.net/

这是 PHP 的主页，这里有关于 PHP 的新闻、文档以及可下载的 PHP 最新版本。

（2）https://www.tcx.se/

这是 MySQL 的主页，这里有关于 MySQL 的新闻、文档以及可下载的 MySQL 最新版本。能在 http://www.buoy.com/mysql/下找到它的很多镜像站点。

（3）PHPBuilder.com

除了 PHP 的主页之外，这个站点对于新 PHP 程序员恐怕是最有价值的站点了。它的价值是有很多文章可供阅读，这些文章包括“看例子，学用正则表达式”、“安装 Apache”和“使用 PHP 创建图像”。

（4）http://www.htmlwizard.net/

这是 phpMyAdmin 应用程序的主页，phpMyAdmin 是一个极好的 MySQL 管理工具。

（5）http://px.sklar.com

（6）http://www.jcc.com/SQLPages/jccs_sql.htm

这个页面设计为有关 SQL 标准进展和它当前状态信息的中心来源，它包含对大量涉及 SQL 的其他资源的链接。

国内资源

（1）PHP 之星

http://www.phpstar.com，一个包含 php 教程、源代码教学、源程序下载、mysql 数据库编程的 php 网站。

（2）中国 PHP 联盟

http://www.phpexe.com，中国 PHP 联盟站，有 PHP 新闻、问答、文章、论坛及 PHP 资源。

（3）PHP 中文用户

http://www.phpuser.com，新手上路，教程与文章，程序与代码，技巧与提示，在线论坛，邮件列表，资源与链接，常见问题解答，相关下载。

（4）中国 PHP 网

http://chinaphp.363.net，中国 PHP 网，提供 PHP 简介，安装，技术文章，例子，论坛及资源。

（5）PHP 资源网

http://php.silversand.net，PHP 资源网，PHP 中文参考手册，PHP 应用源码，PHP 技术论坛等等。

Green Apple Data Center