

教育部顧問室「製商整合科技教育改進」計畫
九十年年度執行成果－教材與教案

計畫名稱：淡江製商整合科技產學合作教育改進計畫

教材名稱：

軟體技術

指導老師：魏世杰 博士

教材編撰：淡江大學資訊管理所

編撰日期：中華民國九十二年一月

單元一 前言	8
第一章 楔子	8
1.1 發展背景	8
單元二 PHP 簡介	11
第二章 PHP 介紹	11
2.1 變數	11
2.1.1 變數的注意事項	11
2.2 動態變數	11
2.2.3 參照變數	12
2.2.4 區域變數與全域變數	12
2.2.5 常數	12
2.2.6 預設常數	13
2.3 資料型別	13
2.3.1 布林	13
2.3.2 整數	13
2.3.3 浮點數	14
2.3.4 字串	14
2.4 陣列	14
2.4.1 建立陣列：	15

2.4.1	陣列的函數	15
2.5	物件	17
2.5.1	建立物件：	17
2.5.2	物件屬性：	17
2.6	NULL	18
2.7	型別轉換	18
2.7.1	使用 Settype()改變資料型態	18
2.7.2	指定資料型態	18
2.8	運算子	18
2.8.1	指定運算子	18
2.8.2	算數運算子	18
2.8.3	連鎖運算子	19
2.8.4	組合指定運算子	19
2.8.5	比較運算子	19
2.8.6	邏輯運算子	20
第三章	PHP 保留字與函數介紹.....	21
3.1	保留字	21
3.1.1	PHP 預設環境常數保留字.....	21
3.2	條件流程控制	21
3.2.1	if 敘述句	21

3.3	switch 敘述句	22
3.4	迴圈流程控制	23
3.4.1	while 迴圈	23
3.4.2	for 迴圈	24
3.5	break 和 continue 敘述	24
3.6	自訂函數	25
3.6.1	字串函數	26
3.7	正規式函數	28
3.7.1	基本的樣版比對	28
3.7.2	函數	29
3.8	cookie / session 變數	30
3.9	日期時間函數	33
3.10	檔案目錄函數	35
3.11	動態繪圖函數	43
3.12	使用 MySQL 資料庫函數	49
單元三	MySQL 簡介	54
第四章	MySQL 介紹	54
4.1	簡介	54
4.2	MySQL 使用者名稱和密碼	54

4.3	MySQL 提供的權限.....	55
4.4	權限的工作原理.....	56
4.4.1	階段一：連接證實.....	58
4.4.2	階段二：請求證實.....	59
4.5	建立初始的 MySQL 權限.....	60
4.6	新增使用者.....	61
4.7	如何設定密碼.....	62
4.8	資料的匯入和匯出.....	63
4.9	備份和復原資料庫.....	65
4.10	建立資料庫和資料表和索引.....	67
4.10.1	建立資料庫.....	67
4.10.2	建立資料表.....	67
	建立索引.....	68
4.12	保留字：.....	69
4.13	欄位型別：.....	71
4.14	SQL 語法.....	74
4.14.1	DCL(Data Control Language).....	74
4.14.2	DDL(Data Design Language).....	76
4.14.3	DML(Data Manipulation Language).....	77
4.15	函數.....	78
	第五章 計數器.....	81

5.1	計數器簡介	81
5.2	資料庫說明	81
5.3	程式碼說明	82
5.4	計數器之範例	84
5.5	會員密碼認證簡介	84
5.6	密碼認證的方法	85
5.7	輸入認證資料介面	85
5.7.1	WWW Authentication 方式	85
5.7.2	利用 form 搭配資料庫的方式	86
5.8	WWW Authentication 及 Form 認證方式	86
5.8.1	WWW Authentication	86
5.8.2	利用 form 搭配資料庫	88
5.9	多網頁間的認證	93
5.9.1	多網頁間的認證—使用 cookie 方式	94
5.9.2	多網頁間的認證—使用 session 方式	97
第六章 基本資料檔維護(新增/刪除/修改/不同過濾條件的查詢)		103
6.1	主題	103
6.2	目的	103
6.3	功能	103
6.4	資料庫(資料表欄位)設計	103

6.5	操作流程	104
6.6	執行畫面範例	105
6.7	程式碼簡介	110
6.8	加簡要註解的完整 PHP 程式碼列表	112
第七章	留言板,群組討論板,線上即時聊天室	121
7.1	留言板	121
7.1.1	整體架構:	121
7.1.2	輸入留言:	121
7.1.3	觀看留言	124
7.1.4	板主管理	125
7.2	群組討論版	127
7.2.1	簡介	127
7.2.3	程式流程	131
7.2.4	程式解說	131
第八章	搜尋引擎	141
8.3	全文檢索	141
	145
第九章	檔案上傳,線上投票,猜謎	149
9.1	檔案上傳	149
9.2	線上投票	154

9.2.1 簡介	154
9.2.2 程式架構	154
9.2.3 程式流程	154
9.2.4 表單規格	155
9.2.5 程式解說	156
9.3 猜謎	166
9.3.1 簡介	166
9.3.2 程式架構	166
9.3.3 程式流程	166
9.3.4 表單規格	168
第十章 購物車	176
10.1 基本商品資料	176
10.2 花束資料的建檔、新增	179
附錄	182

單元一 前言

第一章 楔子

對於網站管理者來說，後端平台的開發介面及應用是整個網站能否達到其應有效能表現的重要關鍵，一個開發工具—PHP 正以驚人的速度在成長。Structure Query Language (SQL，結構化查詢語言) 是當今世界上最流行的和標準化的資料庫語言，它使得儲存、更新和存取資訊更容易。而 MySQL 是一個客戶端/伺服器單結構的實現，更是一個快速、多執行緒、多使用者的 SQL 資料庫伺服器。兩個強大系統的結合，無論是功能、執行的效率、使用的人口，都是目前及未來的主流。PHP、MySQL 與 Apache 三者完美的結合，提供了電子商務一個經濟、效能強大的平台。

1.1 發展背景

PHP 最初是在西元 1994 年 Rasmus Lerdorf 開始計畫發展。在 1995 年以 Personal Home Page Tools (PHP Tools) 開始對外發表第一個版本。在這早期的版本中，提供了訪客留言本、訪客計數器等簡單的功能。隨後在新的成員加入開發行列之後，在 1995 年中，第二版的 PHP 問市。第二版定名為 PHP/FI (Form Interpreter)。PHP/FI 並加入了 MySQL 的支援，自此奠定了 PHP 在動態網頁開發上的影響力。在 1996 年底，有一萬五千個 web 站台使用 PHP/FI；在 1997 年中，使用 PHP/FI 的 web 站台成長到超過五萬個。而在 1997 年中，開始了第三版的開發計劃，開發小組加入了 Zeev Suraski 及 Andi Gutmans，而第三版就定名為 PHP3。

PHP3 跟 Apache 伺服器緊密結合的特性；加上它不斷的更新及加入新的功能；並且它幾乎支援所有主流與非主流資料庫；再以它能高速的執行效率，使得 PHP 在 1999 年中的使用站台超過了十五萬。它的原始碼完全公開，在 Open Source 意識抬頭的今天，它更是這方面的中流砥柱。不斷地有新的函式庫加入，以及不停地更新的活力，使得 PHP 無論在 UNIX 或是 win32 的平台上都可以有更多新的功能。它提供豐富的函式，使得在程式設計方面有著更好的支援。

PHP 是屬於公開式程式 (OPEN SOURCE)，歡迎用於商業或非商業性質用途上，而且是完全免費的。它是一種伺服器端 (server-side)，跨平台 (cross-platform) 的 HTML 內嵌式語言 (類似 IIS 上的 ASP)。獨特的語法混

合了 C、Java、Perl 以及 PHP 式的新語法，並以模組(module)的形式和 Apache 伺服器結合，提供多種連結資料庫的介面，如 MySQL，Microsoft SQL Server，PostgreSQL，Sybase，Informix，InterBase 等。PHP 的表現並不遜色於其他的同類伺服器端介面語言(iHTML、Cold Fusion、locomotive、JSP、ASP、...)，執行效率和開發速率也比 Perl、CGI 等快很多。

PHP 快速成長的原因整理為以下幾點：

1. 支援多種作業平台，無論是個人使用的 Windows95 或者是高負載的 UNIX 叢集，它都可以順利的運作。
2. 自由軟體的特性，使用時不會有道德及法律上的困擾。公開原始碼的方式使用者可自行加入所需的功能。
3. 輕易的和網頁整合，提供互動式的交談功能，讓視覺美工及程式開發人員更能各司其職。
4. 豐富的功能，從結構化的特性、物件式的理想、資料庫處理、網路介面使用、到安全編碼機制...等等，幾乎完整地囊括所有網站所需的功能。
5. 系統轉移速度快，只需少許修正，就可以將整個網站從現有平台移轉。

MySQL 是一個結構化查詢語言資料庫伺服器軟體，它是一種主從式的應用程式，而且可由一部伺服器主機，及許多的用戶主機一同運作。

約一九八一年時，IBM 推出了 SQL，從那個時候開始，這個 SQL 就在關聯式資料庫的發展上，持續扮演著重要角色。IBM 曾經提出一種版本的 SQL 給美國國家標準組織 (ANSI)，後來被通過成為標準，自此，SQL 就廣泛的被使用在一般的關聯式資料庫中。在當時，DB2 是該類型資料庫中，最受歡迎的一種，它是在 1983 年被設計出來，大部份用在大型主機環境。

在 GNU 的世界裡，MySQL 是 Linux 環境中，最常被引用來介紹關聯式資料庫的程式之一。這個應用程式，通常是不會包含在各式安裝套件中，因為它不是遵照 GNU 的軟體使用許可發行的；如果是使用於商業場合，或在其它應用程式中採用它，就需要取得許可證照才可以。

不管資料量的大小，這套資料庫伺服器軟體，都被認為（儘管已在 MySQL 的文件裡載明）是速度快速、系統強固的一個（當然是在同一等級中，與其它的資料庫相比）。MySQL 的原始檔，以及可執行檔，可以用在：Linux 2.0+，SCO，Solaris 2.5，2.6，SUNOS 4.x，BSDI 2.x，3.0，SGI IRIX 6.x，AIX 4.x，DEC UNIX 4.x，HPUX 10.20，Windows，幾乎所有常見的作業系統都支援了。

MySQL 的優點：

1. 便宜（通常是免費）。
2. 網路承載比較少。
3. 高度最佳化（Highly Optimized）。
4. 應用程式透過它做起備份來比較簡單。
5. 為各種不同的資料格式提供提供彈性的介面。
6. 好學，且操作簡單。

單元二 PHP 簡介

第二章 PHP 介紹

2.1 變數

變數是一種可以讓你暫時存放指定的資料的地方。同時，變數可以存放各種不同型別的資料。包括了數字、字串、物件、陣列或是布林型態的資料。這些資料可以依需要而做改變。

變數的表示是在字串的前面加上”\$”，例如：

```
$name;  
$name="Kevin"
```

上面所表示的是將”Kevin”的字串存放到變數 name 裡面。

在 PHP 中與其他的程式語言所不同的是，變數的型別不需要事先宣告，依指定的資料自行決定型別。

2.1.1 變數的注意事項

變數的名稱可以是字母、數字或是底線，但是不能包含空白或是非英文字母。以下是合法的變數：

```
$a;  
$a12;  
$I_Love_You;
```

上面的分號用來結束 PHP 的敘述。

變數可以放置的資料型態包含了：

數字、常數、物件、陣列、或是布林等資料型態。

2.2 動態變數

我們知道變數由\$為開頭，緊接後面加上變數名稱。然而變數也可以存放變數，如下面的例子所示：

```
$user="kevin";  
等於  
$holder="user";
```

```
$$holder="kevin";
```

\$holder 變數放了"user"的字串，所以可以將\$\$holder 中的\$holder 換成 user，那\$\$holder 就等於 user。

可以使用字串的常數來定義一個動態變數，這時候就以大括弧來表示"{}"

```
$user="kevin";
```

```
$holder="user";
```

```
print "$$holder";
```

只能顯示出 "user"

現在只要加上{}

```
$user="kevin";
```

```
$holder="user";
```

```
print "${$holder}";
```

就會顯示出 user

2.2.3 參照變數

我們可以將一個變數的值指定到另一個變數的，例如：我們將變數\$a 的值指定給\$b 變數，\$a 會將值複製到\$b，之後不論\$a 改變任何值都不會影響到\$b 的值，範例程式如下

指定變數的值：

```
$a=42;
```

```
$b=$a;
```

```
print "b=$b"; //output b=42
```

另一種指定的方式參照方式指定，可以將變數\$a 的參照位置指定給變數\$b，因此變數\$b 的值會根據變數\$a 的改變而改變，其範例程式如下

以參照方式指定值：

```
$a=55;
```

```
$b=&$a;
```

```
print "b=$b"; //output b=55
```

```
$a=32;
```

```
print "b= $b" //output b=32
```

2.2.4 區域變數與全域變數

當我們寫一個函數的程式，所用到的變數只適合此函數內部使用，而不會影像到其他的函數

2.2.5 常數

變數提供了一個空間可以彈性的將值存放進去，裡面的值可以依不同的規定與方式來做改變，然而常數卻與之相反，如果我們程式中在整個執行過程裡，需要用到固定的一個值，那我們可以用定義的方式來定義此值，於是我們可以使用

`define()` 函式來定義常數

`define(常數名稱, 常數值)`

一般來講，常數名稱大部分為大寫英文字母。前面不加 `$`

常數值可以是字串也可以是數值

例：`define("$cont_name", "Jordan")`

2.2.6 預設常數

PHP 提供了一些內建的常數，如下面兩個

`_FILE_`：傳回目前解譯器正在讀取的檔案名稱。

`_LINE_`：傳回檔案的列數，對於錯誤訊息的產生很有用。

2.3 資料型別

2.3.1 布林

布林型別在程式中可以用來作為判斷，例如：

```
if(a)
```

```
    printf 55;
```

```
elseif
```

```
    printf 66;
```

如果 `a=true` 則輸出 55，否則輸出 66。

下面個型別的值被認為 `FALSE`。

<i>Boolean</i>	<i>FALSE</i>
<i>Integer</i>	<i>0 (zero)</i>
<i>Float</i>	<i>0.0 (zero)</i>
<i>String</i>	<i>empty string, and the string "0"</i>
<i>Array</i>	<i>Array with zero elements</i>
<i>Object</i>	<i>Object with zero elements</i>
<i>NULL</i>	<i>NULL</i>

2.3.2 整數

整數的表示

`$a = 1234;` # 十進位

`$a = -123; # 負數`
`$a = 0123; # 八進位 數字前面加 0 (equivalent to 83 decimal)`
`$a = 0x1A; # 十六進位 數字前面加 0x (equivalent to 26 decimal)`
 整數由 32 bits 表示，當正整數發生溢位，會以倍精準表示這個數字
 但負整數發生溢位可能會有問題。
 例如：`- 50000000000` 最後會表示出 `-429496728`。

2.3.3 浮點數

浮點數的表示

`$a = 1.234; $a = 1.2e3; $a = 7E-10;`

浮點數表示以 64 bits IEEE 格式表示。

2.3.4 字串

- 單一敘述
單行的敘述，不能表現出“\” “\$”或是“enter”等特殊表示
- 雙重敘述
可表示特殊符號的形式，在特殊符號前面加個“\”

Table 6-1. Escaped characters

<i>sequence</i>	<i>meaning</i>
<code>\n</code>	<i>linefeed (LF or 0x0A (10) in ASCII)</i>
<code>\r</code>	<i>carriage return (CR or 0x0D (13) in ASCII)</i>
<code>\t</code>	<i>horizontal tab (HT or 0x09 (9) in ASCII)</i>
<code>\\</code>	<i>backslash</i>
<code>\\$</code>	<i>dollar sign</i>
<code>\"</code>	<i>double-quote</i>
<code>\[0-7]{1,3}</code>	<i>the sequence of characters matching the regular expression is a character in octal notation</i>
<code>\x[0-9A-Fa-f]{1,2}</code>	<i>the sequence of characters matching the regular expression is a character in hexadecimal notation</i>

2.4 陣列

陣列是以變數配合索引來存取資料的資料結構。我們可以事前定義一個連續的空間，以作為資料的存放，同時附上索引值以最為修改資料的依據。

2.4.1 建立陣列：

- 使用 `array()` 來定義

例如：

```
$name=array("kevin","bob","amy","jack");
```

同義於：

```
$name[0]="kevin";    $name[1]="bob";
```

```
$name[2]="amy";     $name[3]="jack";
```

注意：陣列的放置從 0 開始放置

- 使用陣列符號定義陣列或新增元素

```
$name[]="kevin";
```

```
$name[]="bob";
```

```
$name[]="amy";
```

```
$name[]="jack";
```

PHP 會自動將 `index` 放上去，所以同義於

```
$name[0]="kevin";
```

```
$name[1]="bob";
```

```
$name[2]="amy";
```

```
$name[3]="jack";
```

注意：可以直接可以使用 `index` 來存放資料

但 `index` 必須要有順序的，例如：`$a[0]=33 $a[1]=42` 將資料依序存放。

切不可隨意存放，例如：`$a[0]=33;$a[20]=42`

2.4.1 陣列的函數

`array_change_key_case` – 返回字串鍵名全為小寫或大寫的陣列

`array_chunk` – 將一個陣列分割成多個

`array_count_values` – 統計陣列中所有的值出現的次數

`array_diff` – 計算陣列的差集

`array_fill` – 用給定的值填充陣列

`array_filter` – 用回調函數過濾陣列中的單元

`array_flip` – 反轉陣列中的鍵和值

`array_intersect` – 計算陣列的交集

`array_key_exists` – 檢查給定的鍵名或索引是否存在於陣列中

`array_keys` – 返回陣列中所有的鍵名
`array_map` – 將回調函數作用到給定陣列的單元上
`array_merge_recursive` – 遞迴地合併兩個或多個陣列
`array_merge` – 合併兩個或多個陣列
`array_multisort` – 對多個陣列或多維陣列進行排序
`array_pad` – 用值將陣列填補到指定長度
`array_pop` – 將陣列最後一個單元彈出（出棧）
`array_push` – 將一個或多個單元壓入陣列的末尾（入棧）
`array_rand` – 從陣列中隨機取出一個或多個單元
`array_reduce` – 用回調函數疊代地將陣列簡化為單一的值
`array_reverse` – 返回一個單元順序相反的陣列
`array_search` – 在陣列中搜索給定的值，如果成功則返回相應的鍵名
`array_shift` – 將陣列開頭的單元移出陣列
`array_slice` – 從陣列中取出一段
`array_splice` – 把陣列中的一部分去掉並用其他值取代
`array_sum` – 計算陣列中所有值的和
`array_unique` – 移除數組中重復的值
`array_unshift` – 在陣列開頭插入一個或多個單元
`array_values` – 返回陣列中所有的值
`array_walk` – 對陣列中的每個成員應用用戶函數
`array` – 新建一個陣列
`arsort` – 對陣列進行逆向排序並保持索引關係
`asort` – 對陣列進行排序並保持索引關係
`compact` – 建立一個陣列，包括變數名和它們的值
`count` – 統計變數中的單元數目
`current` – 返回陣列中的當前單元
`each` – 返回陣列中當前的鍵／值對並將陣列指標向前移動一步
`end` – 將陣列的內部指標指向最後一個單元
`extract` – 從陣列中將變數導入到當前的符號表
`in_array` – 如果陣列中存在該值則返回 `TRUE`
`key` – 從結合陣列中取得鍵名
`krsort` – 對陣列按照鍵名逆向排序
`ksort` – 對陣列按照鍵名排序
`list` – 把陣列中的值賦給一些變數
`natcasesort` – 用“自然排序”演算法對陣列進行不區分大小寫字母的排序
`natsort` – 用“自然排序”演算法對陣列排序
`next` – 將陣列中的內部指標向前移動一位
`pos` – 得到陣列當前的單元

- prev – 將陣列的內部指標倒回一位
- range – 建立一個包含指定範圍單元的陣列
- reset – 將陣列的內部指標指向第一個單元
- rsort – 對陣列逆向排序
- shuffle – 將陣列打亂
- sizeof – 統計變數中的單元數目
- sort – 對陣列排序
- uasort – 使用用戶自定義的比較函數對陣列中的值進行排序並保持索引關聯
- uksort – 使用用戶自定義的比較函數對陣列中的鍵名進行排序
- usort – 使用用戶自定義的比較函數對陣列中的值進行排序

2.5 物件

物件是將變數和函式包裝起來形成一種樣板，稱為類別。

2.5.1 建立物件：

用 `class` 宣告一個類別：

```
class data
{
    //最小的 class;
}
```

宣告後要建立一個類別的樣板，使用 `new` 來敘述

```
$obj=new data();
print "Data type is".gettype($obj);// 回傳你的類別型態為
object; output Data type is object
```

2.5.2 物件屬性：

物件可以存取特殊的變數稱為屬性

```
class data
{
    var $name="Jordan";
}
```

`data` 的類別中包含了一個屬性 `name`，其值 `Jordan`

2.6 NULL

一個變數不包含任何的值，以 NULL 來表示。

2.7 型別轉換

2.7.1 使用 Settype()改變資料型態

Settype(變數, 型別)

例：

```
$pi=3.14;  
print gettype($pi);//double  
settype($pi, integer);  
print gettype($pi);//integer
```

settype()是強制轉換，轉換後即為轉換型別

2.7.2 指定資料型態

例：

```
$pi=3.14;  
$temp=$pi;  
print gettype($temp);//double  
$temp=(integer) $pi;  
print gettype($temp);//integer  
$temp=(string) $pi;  
print gettype($temp);//string
```

我們可以發現，\$pi 並沒有被改變型態

而是將\$pi 的值複製到\$temp，所以原來\$pi 的資料型態沒有被改變。

2.8 運算子

2.8.1 指定運算子

等號 '='

等號的右邊指定值給等號的左邊

2.8.2 算數運算子

即一般的四則運算

運算子	名稱
+	加號
-	減號
*	乘號
/	除號
%	餘數

2.8.3 連鎖運算子

可作為字串與字串的連結

“.”

“hello”.”world”

等於

helloworld

要注意的是,空格也算一個字元

例如: “hello”.” world”

等於

hello word

2.8.4 組合指定運算子

運算子	範例	原表示式
+=	$a += 2$	$a = a + 2$
-=	$a -= 2$	$a = a - 2$
/=	$a /= 2$	$a = a / 2$
*=	$a *= 2$	$a = a * 2$
%=	$a \% = 2$	$a = a \% 2$
.=	$a .= "book"$	$a = a . "book"$

2.8.5 比較運算子

假設 $a=4$

運算子	名稱	範例	如果 TRUE	結果
-----	----	----	---------	----

==	等於	$\$a==5$	$\$a$ 等於 4	傳回 FALSE
!=	不等於	$\$a!=5$	$\$a$ 不等於 5	傳回 TRUE
===	相同	$\$a===5$	$\$a$ 相同於 4 且 資料型態一樣	傳回 FALSE
>	大於	$\$a>4$	$\$a$ 大於 4	傳回 FALSE
>=	大於等於	$\$a>=4$	$\$a$ 大於等於 4	傳回 TRUE
<	小於	$\$a<5$	$\$a$ 小於 5	傳回 TRUE
<=	小於等於	$\$a<=3$	$\$a$ 小於等於 3	傳回 FALSE

2.8.6 邏輯運算子

邏輯運算子可以作為邏輯運算

運算子	名稱	範例	傳回 TRUE	傳回 FALSE
// or	或運算	$\$a$ or $\$b$	$\$a$ % $\$b$ 其中 一個為 true 或 兩個為 true	兩個都為 FALSE
xor	互斥運算	$\$a$ xor $\$b$	$\$a$ 和 $\$b$ 其中一 個必為 true 另 一個必為 false	兩個都為 FALSE 或是兩個都為 TRUE
&& and	和運算	$\$a$ and $\$b$	$\$a$ 和 $\$b$ 都必 為 true	$\$a$ % $\$b$ 其中一 個為 false 或兩個 為 false
!	反運算	! $\$a$	$\$a$ 為 false	$\$a$ 為 true

第三章 PHP 保留字與函數介紹

3.1 保留字

and、or、not	邏輯運算保留字
list	將陣列變數一一對應到指定的變數中
empty()	判斷某個變數是否已指定值
global、static	變數的修飾詞
if、else、elseif、endif	條件流程控制保留字
switch、case、default、endswitch	條件流程控制保留字
for、foreach、endfor	迴圈流程控制保留字
do、while、endwhile	迴圈流程控制保留字
break	跳出迴圈
continue	繼續迴圈
function	自訂函數保留字 - 宣告函數
class	類別資料型態 - 宣告類別
return	自訂函數保留字 - 傳回值
require	用 <code>require</code> 方法引用外部檔案
include	用 <code>include</code> 方法引用外部檔案

3.1.1 PHP 預設環境常數保留字

PHP_OS	顯示 PHP 執行的作業系統平台
PHP_VERSION	顯示 PHP 模組的版本
E_ERROR	顯示除了解析語法錯誤外，其他程式上的錯誤
E_WARNING	當 PHP 解譯程式碼時，發現錯誤但仍會繼續執行下去，此常數顯示此種錯誤訊息
E_PARSE	顯示解譯語法出現的錯誤
E_NOTICE	有可能會產生錯誤的程式碼，程式仍會執行下去，此常數顯示此種錯誤訊息
E_ALL	將所有 E_ 開頭的常數訊息合在一起

3.2 條件流程控制

條件敘述句讓我們設立條件和一段程式碼，當條件成立時便執行。PHP 提供兩種條件結構，`if... elseif ...else` 和 `switch ...case`。

3.2.1 if 敘述句

if 敘述句的最基本型態為：

```
if (條件式)
```

```
敘述;
```

例如：

```
    if ($a > $b)
        print "a is bigger than b";
```

我們可以加上 **else** 來作兩段程式碼的選擇：

```
    if (條件式)
        敘述 1
    else
        敘述 2
```

例如：

```
    if ($a > $b)
    { print "a is bigger than b"; }
    else
    { print "a is NOT bigger than b"; }
```

接著我們可以作更多程式碼的控制：

```
    if (條件式 1)
        敘述 1
    elseif(條件式 2)
        敘述 2
    else
        敘述 3
```

例如：

```
    if ($a > $b)
    { print "a is bigger than b"; }
    elseif ($a == $b)
    { print "a is equal to b"; }
    else
    { print "a is smaller than b"; }
```

3.3 switch 敘述句

switch 敘述句是一種判斷條件的方式，在多條件的情況下，它比 if...elseif 更有效率，格式如下：

```
switch(變數) {
    case 值 1:
        敘述 1;
        break;
```

```
    case 值 2:
敘述 2;
break;
default:
    其他敘述;
}
```

例如：

```
switch ($i) {
    case 0: print "i equals 0"; break;
    case 1: print "i equals 1"; break;
    case 2: print "i equals 2"; break;
    default: print "i is not equal to 0, 1 or 2";
}
```

3.4 迴圈流程控制

迴圈可以指定次數或在某種條件下，重複執行某段程式碼，PHP 提供兩種迴圈，while 和 for。

3.4.1 while 迴圈

執行 while 迴圈時，PHP 會檢查 while 迴圈設立的條件是否成立，若成立則執行迴圈內的程式，執行之後再檢查一次條件是否成立。格式為：

```
while (expr) {
statements;
}
```

例如：

```
$i = 1;
while ($i <= 10) {
    print $i++;
}
```

或者 PHP 也提供 while... endwhile 的寫法，作用和 while 完全一樣：

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

例如：

```
$i = 1;
while ($i <= 10):
```

```
print $i;
    $i++;
endwhile;
```

另外 PHP 也有 `do...while` 的寫法，跟 `while` 的不同處在於，`while` 在執行為迴圈內容前會先檢查條件是否符合，`do...while` 則在第一次執行時不會檢查，迴圈內容執行完後才會檢查條件，例如：

```
$i = 0;
do {
print $i;
}while ($i>0);
```

3.4.2 for 迴圈

`for` 的功能是，在撰寫迴圈時可以先將迴圈起始敘述定義在第一個項目內，之後用分號隔開，第二個項目表示迴圈繼續執行的條件，第三個項目表示迴圈每執行一次所要執行的指令，格式如下：

```
for (expr1; expr2; expr3) {
statements;
}
```

例如：

```
for ($i = 1; $i <= 10; $i++) {
print $i;
}
```

實際上，`for` 迴圈和 `while` 迴圈能做到的事情是一樣的，差別只是在於寫程式的習慣和簡便罷了，例如以上的 `for` 範例可以改寫成：

```
$i = 1;
while($i <= 10) {
print $i;
$i++;
}
```

3.5 break 和 continue 敘述

`break` 和 `continue` 都用於迴圈之中。`break` 表示迴圈進行到 `break` 敘述時，無條件跳出迴圈，不再執行之後的程式，也不重新檢查迴圈條件，例如：

```
for ($i = 1;;$i++) {
if ($i > 10) {
```

```

break;                // 若$i>10，則無條件跳出迴圈
}
print $i;
}

```

`continue` 表示迴圈內執行到 `continue` 敘述時，自動跳過之後的程式碼，重新檢查迴圈條件。例如：

```

while (list ($key, $value) = each ($arr)) {
  if (!(($key % 2)) {
    continue;          // 若$key 為偶數，則跳過之後的
                      //print 敘述，重頭執行迴圈
  }
  print $value;      // 這裡只會印出奇數
}

```

3.6 自訂函數

函數是一段程式碼，只要在某處定義一次，就能從其他地方使用它。一般來說，函數會接受參數，處理之後，將結果傳回呼叫函數的地方。例如：

```

//宣告並定義函數
function sum($num1, $num2) {
  $answer = $num1+$num2;
  return $answer;      //傳回$num1 + $num2 的和
}
echo sum(2, 3);        //呼叫函數，輸出 5

```

函數中宣告的變數，使用範圍僅限於函數中，在此函數外無法使用此函數，即使在函數外用相同名稱的變數，也不會互相影響例如：

```

function change_pos() {
  $pos = "b";
}
$pos = "a";
change_pos();
echo $pos;            //輸出"a"

```

但可以使用 `global` 修飾字，宣告此變數為全域變數，如此不管在函數的內外都可以使用相同的變數。例如：

```

function change_pos() {
  global $pos;
  $pos = "b";
}

```

```
$pos = "a";
change_pos();
echo $pos;          //輸出"b"
```

呼叫函數預設所傳的參數為傳值(call by value)，將某個變數當參數傳遞時，此變數原本的值並不會改變，因為它是將值傳入函數中，相對的，若是傳的參數為位址(call by address)，那個就是將此變數的位址傳入函數，在函數內改變參數值時就會改變原本變數的內容，宣告函數時，在參數前加上 & 符號，就表示此參數傳遞的是位址。例如：

```
function raise($per, &$salary) {
    $salary = $salary + $salary * $per/100 ;
}
$sal = 1000;
echo "調整前：$sal";    //輸出 1000
raise(4, $sal);
echo "調整後： $sal";    //輸出 4100
```

3.6.1 字串函數

```
string substr(string source, int begin, int [length]) ;
```

可傳回字串的一部份。

引數一 -- 「來源字串」

引數二 -- 「開始傳回字串的位置」 (從 0 算起)

引數三 -- 「欲傳回的字串長度」 (可省略)

範例：

```
echo (substr("Christopher", 1)) ;    //輸出 hristopher
echo (substr("Christopher", -3)) ;    //輸出 her
echo (substr("Christopher", -5, 3)) ; //輸出 oph
echo (substr("Christopher", 2, 3)) ;  //輸出 ris
echo (substr("Christopher", 2, -3)) ; //輸出 ristop
echo (substr("Christopher", -6, -3)) ; //輸出 top
echo (substr("Christopher", 7, -8)) ; //輸出 ' '
```

```
string trim ( string str [, string charlist])
```

```
string ltrim ( string str [, string charlist])
```

```
string chop ( string str [, string charlist])
```

移除字串開頭與尾端的空白字元 (空格、定位 TAB)

移除字串前的空白

移除字串尾端的空白

範例：

```
    echo (trim("    sample string  ")) ;    //輸出'sample
string'
    echo (ltrim("    sample string  ")) ;    //    輸
出'sample string '
    echo (chop("    sample string  ")) ;    // 輸 出 '
sample string '
```

string chr (int ascii)

接受一個 ASCII 碼，並傳回相對應值

範例：

```
    echo (chr(34)) ;                //輸出 “
```

int ord (string string)

chr()的互補函數，接受一個字元並傳回對應的 ASCII 碼整數

範例：

```
    if ($c != ord(9) && $c != ord(13)) {
        $my_string .= $c;
    }
    //當$c 不為 TAB 或 ENTER 時，才將它附加至 $my_string
    echo (ord('@'));                //輸出 64
```

int strlen (string str)

傳回字串所包含的字元數目

範例：

```
    echo (strlen("Christopher")) ;    //輸出 11
    if (strlen($userinput) >30) {
        echo ( "請將字元控制在 30 個字元以內。" ) ;
    }
    //實際應用上，可配合欄位長度的限制。
```

void printf (string format [, mixed args])

void sprintf (string format [, mixed args])

根據指令而輸出格式化(format)的字串

前者將字串輸出至瀏覽器而無傳回值

後者適用於將值傳回變數或資料庫

範例：

```
    $dollar = "20 sollars" ;
```

```
printf (“%d”, $dollars) ;           //輸出 20
//格式字串%d 會使字串被輸出成十進位值
```

```
string number_format ( float number [, int decimals [,
string dec_point [, string thousands_sep]]])
```

簡單的數字格式化功能，只可採用 1、2 或 4 個引數，3 個會造成錯誤
範例：

```
$num = 123456789.1234567 ;
echo (number_format ($num)) ;           //      輸      出
123,456,789
```

//只採用第一個引數，會用逗號分隔千分位

```
$num = 123456789.1234567 ;
echo (number_format($num, 4)) ;           //      輸      出
123,456,789.1235
```

//第二個引數表示小數點後的精確度

```
$num = 123456789.1234567 ;
echo (number_format($num, 7, chr(44), “ ”)) ; //輸出
123 456 789,1234567
```

//第三個引數可變更小數點字元，chr(44)為逗號

//第四個引數為整數部分千分位的分隔字元

3.7 正規式函數

正規表達式本質上是一個樣板(pattern)，是一組「用來描述字串特徵」的字元可以只表示普通字串，也可以表示字元的連續範圍、多重出現、搜尋特定的上下文關係。

3.7.1 基本的樣版比對

比對以 This 開頭的字串：`^This`

比對以 le 結尾的字串：`^le$`

比對 Th 字串：`^Th$`

`[a-z]` //比對任何小寫字母

`[A-Z]` //比對任何大寫字母

`[a-zA-Z]` //比對任何字母

`[0-9]` //比對任何數字

```

[0-9\.] //比對任何數字、點
[ \f\r\t\n] //比對任何空白字元
[[:alpha:]] //任何字母
[[:digit:]] //任何數字
[[:alnum:]] //任何字母或數字
[[:space:]] //任何空白
[[:upper:]] //任何大寫字母
[[:lower:]] //任何小寫字母
[[:punct:]] //任何標點符號
[[:xdigit:]] //任何 16 進位制數字
^a$ //比對 a
^a{4}$ //比對 aaaa
^a{2,4}$ //比對 aa 或 aaa 或 aaaa
^a{2,}$ //比對”含 2 個 a 以上”的字串

```

3.7.2 函數

```

int ereg(string pattern, string source, array[regs])
int eregi(string pattern, string source, array[regs])

```

用來搜尋與比對

範例：

```

if (ereg( “^(.+)\.(.+)$”, $email, $arr)){
    echo (“正確的 Email:$arr[0]\n”.
        “帳號:$arr[1]\n”.
        “網域名稱:$arr[2]\n”.
        “頂層網域名稱:$arr[3]\n” );
} else {
    echo (“不正確的 Email” );
}

```

```

string ereg(string pattern, string repl, string string)
string eregi_replace(string pattern, string repl, string
string)

```

用來搜尋並取代

範例：

```

$str=“This is a apple”;
$pat=“apple”;
$repl=“APPLE”;
echo (ereg_replace($pat, $repl, $str));

```

```
array split(string pattern, string string, int [limit])
```

分割字串

範例：

```
$str="This is a apple";  
$pat=" ";  
$arr1=split($pat, $str);  
$arr2=split($pat, $str, 3);  
echo ( "$arr1[0]; $arr1[1]; $arr1[2]; $arr1[3]\n" );  
echo ( "$arr2[0]; $arr2[1]; $arr2[2]\n" );
```

```
array preg_grep ( string pattern, array input)
```

以陣列形式傳回 input 陣列裡符合 pattern 的資料項

```
int preg_match_all ( string pattern, string subject, array  
matches [, int flags])
```

進行全域的正規表達式比對

```
int preg_match ( string pattern, string subject [, array  
matches [, int flags]])
```

進行正規表達式比對

```
string preg_quote ( string str [, string delimiter])
```

脫逸特殊的正規表達式字元

3.8 cookie / session 變數

cookie 為存在使用者端的純文字字串，儲存格式為 name = value，在 PHP 中，cookie 的使用有幾個限制：一個瀏覽器最多 300 個 cookie、一個瀏覽器對伺服器最多 20 個 cookie、瀏覽器可關閉 cookie 功能。Cookie 函數的用法有下列幾種：

最基本的設定

```
setcookie("cookienam", "value");
```

多值 Cookie

```
setcookie("cookies", $visitor);
```

```
setcookie("cookies[1]", $password);
```

設定期限

```
setcookie("cookiename", "value", $lifetime);
```

限制 Cookie 範圍

```
setcookie("c_name", "value", $lifetime, "/mydir/");  
setcookie("c_name", "value", $lifetime, "/mydir/",  
".server.net");
```

安全的 HTTP 連線 (HTTPS)

```
setcookie("c_name", "value", $lifetime, "", "", 1);
```

刪除 Cookie

```
setcookie("c_name");
```

存取 cookie 時，可以直接存取全域變數，例如：

```
setcookie("MyName", "SEAMAN");  
echo $MyName;
```

在 PHP4.2 版之後，預設已經不允許直接存取全域變數，必須用全域變數陣列的方式，存取 cookie 的值，例如：

```
Setcookie( "MyName", "SEAMAN" );  
echo $HTTP_COOKIE_VARS[ "MyName" ];  
echo $_COOKIE[ "MyName" ];
```

`session` 為儲存在伺服器上的純文字檔，儲存格式為 檔名 = `session_id`，`session` 是在 PHP4.0 之後的版本才有支援，並且在使用上要小心儲存太多的 `session` 可能會造成伺服器的負擔。要使用 `session` 變數必須向伺服器註冊，註冊的方法為：

```
//開始使用 session  
session_start();  
//註冊 session 變數  
session_register("MyName");  
使用時，和 cookie 的方式類似，可以直接存取全域變數：  
$MyName = "SEAMAN" ;
```

同樣的，在 PHP4.2 版之後，因為安全性考量，必須用全域變數陣列的方式存取：

```
$_SESSION[ "MyName" ] = "SEAMAN" ;
```

PHP 提供了一些 `session` 的函數，可以對 `session` 變數作一些控制：

`session_start()`

初始 session

`session_destroy()`

結束 session

`string session_name(string name)`

存取目前 session 名稱

`string session_module_name(string module)`

存取目前 session 模組

`string session_save_path(string path)`

存取目前 session 路徑

`string session_id(string id)`

存取目前 session 代號

`boolean session_register(string name)`

註冊新的變數

`boolean session_unregister(string name)`

刪除已註冊變數

`boolean session_is_registered(string name)`

檢查變數是否註冊

`boolean session_decode(string data)`

Session 資料解碼。

`boolean session_encode(void)`

Session 資料編碼。

在使用 `cookie` 和 `session` 時，常見的錯誤為在 設定 `cookie` 或 `session` 之前在網頁上已經有輸出，此時瀏覽器會顯示錯誤訊息：

Cannot add header information - headers already sent ...

所以，使用上必須小心在設定 `cookie` 和 `session` 變數之前，不可以在網頁上有任何輸出，即使是輸出空白也不允許。

3.9 日期時間函數

`int checkdate(int month, int day, int year)`

驗證日期的正確性。

`string date(string format, int [timestamp])`

將伺服器的時間格式化。

範例：

```
print(date( "l ds of F Y h:i:s A" ));  
print("July 1, 2000 is on a " . date("l",  
mktime(0,0,0,7,1,2000)));
```

```
$tomorrow =  
mktime(0,0,0,date("m") ,date("d")+1,date("Y"));  
$lastmonth = mktime(0,0,0,date("m")-1,date("d"),  
date("Y"));  
$nextyear = mktime(0,0,0,date("m"), date("d",  
date("Y")+1);
```

`string strftime(string format, int [timestamp])`

將伺服器的時間本地格式化。

範例：

```
setlocale ("LC_TIME", "C");  
print(strftime("%A in Finnish is "));  
setlocale ("LC_TIME", "fi");  
print(strftime("%A, in French "));  
setlocale ("LC_TIME", "fr");  
print(strftime("%A and in German "));  
setlocale ("LC_TIME", "de");  
print(strftime("%A.\n"));
```

`array getdate(int timestamp)`

獲得時間及日期資訊。

`array gettimeofday(void)`

取得目前時間。

`string gmdate(string format, int timestamp)`

取得目前與 GMT 差後的時間。

範例：

```
echo date( "M d Y H:i:s",mktime(0,0,0,1,1,1998) );  
echo gmtime( "M d Y H:i:s",mktime(0,0,0,1,1,1998) );
```

`int easter_date(int [year])`

計算復活節日期。

範例：

```
echo date("M-d-Y", easter_date(1999)); // 輸出  
Apr-04-1999  
echo date("M-d-Y", easter_date(2000)); // 輸出  
Apr-23-2000  
echo date("M-d-Y", easter_date(2001)); // 輸出  
Apr-15-2001
```

`int easter_days(int [year])`

計算復活節與三月廿一日之間日期數。

範例：

```
echo easter_days(1999); //輸出 14 (4/4)  
echo easter_days(1492); //輸出 32 (4/22)  
echo easter_days(1913); //輸出 2 (3/23)
```

`int mktime(int hour, int minute, int second, int month, int day, int year)`

取得 UNIX 時間戳記。

範例：

```
echo date( "M-d-Y", mktime(0,0,0,12,32,1997) );  
echo date( "M-d-Y", mktime(0,0,0,13,1,1997) );  
echo date( "M-d-Y", mktime(0,0,0,1,1,1998) );
```

`int gmmktime(int hour, int minute, int second, int month, int day, int year)`

取得 UNIX 時間戳記的格林威治時間。

`int time(void)`

取得目前時間的 UNIX 時間戳記。

`string microtime(void)`

取得目前時間的 UNIX 時間戳記的百萬分之一秒值。

3.10 檔案目錄函數

string basename(string path)

將含路徑及檔案字串中的路徑拿掉，傳回只含檔名的字串。在 Windows 系列的作業系統中，路徑可以是斜線 (/) 或反斜線 (\)；在 UNIX 系列的作業系統，路徑為斜線 (/)。

範例：

```
$path = "/home/httpd/html/index.php";  
$file = basename($path);
```

int chgrp(string filename, mixed group)

用來改變檔案所屬的群組，僅系統管理員權限可任意改變檔案所屬群組，其他使用者只能改變本身擁有的檔案。成功傳回 **true**，失敗傳回 **false**；而在 windows 作業系統中，此函數什麼也不做就傳回 **true**。

int chmod(string filename, int mode)

用來改變檔案的屬性。成功則傳回 **true**、失敗傳回 **false**。在 **mode** 值中，系統不會自動將它加 0 以獲得正確的八進位演算，若要正確使用，應用下列第二行的方式較好。

範例：

```
chmod("/mydir/myfile", 755 ); // mode 為十進位，可能有  
潛在錯誤  
chmod("/mydir/myfile", 0755 ); // mode 為八進位，確定的  
屬性值
```

int chown(string filename, mixed user)

用來改變檔案的擁有者，只有系統管理員權限可以任意改變檔案所屬群組，其他使用者都只能改變本身擁有的檔案。成功傳回 **true**，失敗傳回 **false**；在 windows 作業系統中，此函數什麼也不做就傳回 **true**。

void clearstatcache(void)

對作業系統而言，使用 **stat()** 或 **lstat()** 二個函數很耗資源，尤其當每次要重新呼叫 **stat()** 及 **lstat()** 二個函數時，更是耗資源的方式。因此為節省資源，PHP 系統會將檔案狀態放在快取記憶體中，以便隨時使用。而此函數就是提供清除快取記憶體中檔案資訊的函數。

int copy(string source, string dest)

用來複製檔案。成功則傳回 `true`、失敗傳回 `false`。

範例：

```
if (!copy($file, $file.'.bak')) {  
    print("複製檔案 $file 失敗...<br>\n");  
}
```

string dirname(string path)

傳回 `path` 中的目錄名稱。在 `win32` 系統中，用斜線 `/` 或者反斜線 `\` 都可以；但是其它的作業系統的路徑都是斜線 `/`。

範例：

```
$path = "/etc/hostname";  
$file = dirname($path);
```

float diskfreespace(string directory)

用來取得目錄所在儲存裝置剩餘空間，單位為位元組 (`byte`)。

範例：

```
$df = diskfreespace("/");
```

int fclose(int fp)

用來關閉已經開啓的檔案的指標 `fp`。成功傳回 `true`，失敗則傳回 `false`。其中檔案指標必須是有效的，必須是已用 `fopen()` 或 `fsockopen()` 成功開檔的指標。

int feof(int fp)

測試檔案的指標是否指到檔尾 (`End of File, EOF`)。若是成功或發生錯誤會傳回 `true` 值，其它情形則傳回 `false` 值。其中檔案指標必須是有效的，必須是已用 `fopen()`、`popen()` 或 `fsockopen()` 成功開檔的指標。

string fgetc(int fp)

取得檔案指標所指的字元，傳回字串型態的字元。指標若在 `EOF` 則傳回 `false`。其中檔案指標必須是有效的，必須是已用 `fopen()`、`popen()` 或 `fsockopen()` 成功開檔的指標。

array fgetcsv(int fp, int length, string [delimiter])

此函數與 `fgets()` 的功用相似，不同處在於此函數用來解析讀取行的 `CSV` 欄位資料，並放入陣列變數之中。其中第三個參數 `delimiter` 若未指定，則使用內定值：逗號。其中檔案指標 `fp` 必須是有效的，必須是已用 `fopen()`、`popen()` 或 `fsockopen()` 成功開檔的指標。而第二個參數 `length` 的值必須

要比 CSV 檔案最長一行的字元數還大。

範例：

```
$row=1;
$fp = fopen("mycsv.csv","r");
while ($data = fgetcsv($fp,1000, ",")) {
    $num = count($data);
    print "<p>欄位 $num 在第 $row 行: <br>";
    $row++;
    for ( $c=0; $c<$num; $c++ ) print $data[$c] . "<br>";
}
fclose($fp);
```

string fgets(int fp, int length)

取得檔案指標所指的行，傳回字串長度為行的長度減一。若發生錯誤傳回 `false`。其中檔案指標必須是有效的，必須是已用 `fopen()`、`popen()` 或 `fsockopen()` 成功開檔的指標。

範例：

```
$fd = fopen("/tmp/myfile.txt", "r");
while ($buffer = fgets($fd, 4096)) {
    echo $buffer;
}
fclose($fd);
```

string fgetss(int fp, int length)

除了 `fgets()` 的功能，並同時去掉取回字串中的 HTML 語法及 PHP 語法的標記字串。

array file(string filename)

將檔案全部讀出，並輸出到陣列變數中，每行都是單獨的陣列元素。

int file_exists(string filename)

用來測試檢查檔案是否存在，傳回 `true` 表示檔案存在。

int fileatime(string filename)

傳回指定檔案 `filename` 的最後存取時間。傳回 `false` 表示發生錯誤。

int filectime(string filename)

傳回指定檔案 `filename` 的 `inode` 最後改變時間。傳回 `false` 表示發生

錯誤。

`int filegroup(string filename)`

傳回指定檔案的群組使用者 `GID` 值。傳回 `false` 表示發生錯誤。

`int fileinode(string filename)`

傳回指定檔案的 `inode` 值。傳回 `false` 表示發生錯誤。

`int filemtime(string filename)`

傳回指定檔案 `filename` 的最後修改時間。傳回 `false` 表示發生錯誤。

`int fileowner(string filename)`

傳回指定檔案擁有者的 `UID` 值。傳回 `false` 表示發生錯誤。

`int fileperms(string filename)`

傳回指定檔案的權限設定值。傳回 `false` 表示發生錯誤。

`int filesize(string filename)`

傳回指定檔案的檔案大小。傳回 `false` 表示發生錯誤。

`string filetype(string filename)`

傳回指定檔案的檔案型態。可能的傳回型態有 `fifo`、`char`、`dir`、`block`、`link`、`file` 及 `unknown` 等。傳回 `false` 表示發生錯誤。

`boolean flock(int fp, int operation)`

用來鎖住檔案，使別的行程無法存取。傳入的參數 `fp` 為檔案指標。參數 `operation` 的值為下列的數字之一：

- 1：表示設定鎖住檔案可以允許別的行程讀取；
- 2：表示只有該行程可以寫入檔案；
- 3：表示讀寫均鎖住；
- 4：則不鎖住區塊 (`block`)。

執行成功則傳回 `true` 值，否則傳回 `false` 值。

`int fopen(string filename, string mode)`

可用來開啓本地或遠端檔案。若開檔失敗，則傳回 `false` 值。字串參數 `mode` 可以是下列的情形：

- `r`：開檔方式為唯讀，檔案指標指到開始處。
- `r+`：開檔方式為可讀寫，檔案指標指到開始處。
- `w`：開檔方式為寫入，檔案指標指到開始處，並將原檔的長度設為 `0`。若檔案

不存在，則建立新檔。

w+：開檔方式為可讀寫，檔案指標指到開始處，並將原檔的長度設為 **0**。若檔案不存在，則建立新檔。

a：開檔方式為寫入，檔案指標指到檔案最後。若檔案不存在，則建立新檔。

a+：開檔方式為可讀寫，檔案指標指到檔案最後。若檔案不存在，則建立新檔。

b：若作業系統的文字及二進位檔不同，則可以用此參數，**UNIX** 系統不需要使用本參數。

```
int fpassthru(int fp)
```

讀取檔案 **fp** 直到檔尾 (End of file, EOF)，將資料輸出到標準輸出 (standard output)。若有錯誤發生，則傳回 **false** 值。

```
int fputs(int fp, string str, int [length])
```

即 **fwrite()**，僅是一個別名。會將字串 **str** 寫到檔案指標。

```
string fread(int fp, int length)
```

讀到指定長度的位元組或到檔尾 EOF。

範例：

```
$filename = "/usr/local/myfile.txt";  
$fd = fopen( $filename, "r" );  
$contents = fread($fd, filesize($filename));  
fclose( $fd );
```

```
int fseek(int fp, int offset)
```

將檔案 **fp** 的指標移到指定的偏移位元 (**offset**) 上。成功則傳回 **0**，失敗則傳回 **-1** 值。當 **fp** 由 **fopen()** 開啓 "**http://....**" 或是 "**ftp://....**" 等 URL 檔案時，本函數無法作用。

```
int ftell(int fp)
```

傳回檔案 **fp** 的指標偏移位元 (**offset**) 值。當發生錯誤時，傳回 **false** 值。檔案指標 **fp** 必須是有效的，且使用 **fopen()** 或者 **popen()** 二個函數開啓方可作用。

```
int fwrite(int fp, string string, int [length])
```

將字串 **string** 寫入檔案資料流的指標 **fp** 上。若有指定長度 **length**，則會寫入指定長度字串，或寫到字串結束。

```
int set_file_buffer(int fp, int buffer)
```

設定檔案緩衝區大小供寫入檔案使用。若 `buffer` 值為 0，則在寫入操作時無緩衝區。成功則傳回 0 值。

`boolean is_dir(string filename)`

傳回 `true` 值表示指定的 `filename` 為目錄名稱。

`boolean is_executable(string filename)`

傳回 `true` 值表示指定的 `filename` 存在並為可執行檔。

`boolean is_file(string filename)`

傳回 `true` 值表示指定的 `filename` 存在並為正常的檔案。

`boolean is_link(string filename)`

傳回 `true` 值表示指定的 `filename` 存在並為符號連結檔 (symbolic link)。

`boolean is_readable(string filename)`

傳回 `true` 值表示指定的 `filename` 存在並且可讀取。

`boolean is_writable(string filename)`

傳回 `true` 值表示指定的 `filename` 存在並且可讀取。參數 `filename` 也可以是目錄名稱，用來測試該目錄是否可寫入。

`int link(string target, string link)`

建立硬式連結，用來將參數 `link` 連結至 `target`。

`int linkinfo(string path)`

使用系統呼叫 `lstat`，傳回 UNIX 上 C 語言 `stat` 結構型態的 `st_dev` 欄位。傳回 0 或 `false` 表示發生錯誤。

`int mkdir(string pathname, int mode)`

建立參數 `pathname` 目錄，參數 `mode` 以八進位方式指定（在參數前加 0）比較不會出錯。傳回 `true` 表示成功，失敗則傳回 `false`。

`int pclose(int fp)`

用來關閉以 `popen()` 開啓的檔案。而參數 `fp` 必需是有效的，且使用 `popen()` 開啓的檔案。會傳回結處理檔案的結束狀態。

`int popen(string command, string mode)`

執行指令開檔，而該檔案是用管道方式處理的檔案。用本函數開啓檔案只能單向(只能讀或只能寫)，且一定要用 `pclose()` 關閉。在檔案操作上可使用 `fgets()`、`fgetss()` 與 `fputs()`。若是開檔發生錯誤，傳回 `false` 值。

`int readfile(string filename)`

讀取檔案，並將檔案內容寫入標準輸出設備中。傳回值是讀取的檔案大小。若有錯誤傳回 `false` 值。

`string readlink(string path)`

傳回符號連結的檔案。發生錯誤則傳回 `0`。

`int rename(string oldname, string newname)`

可將舊檔名 `oldname` 改成新檔名 `newname`。成功則傳回 `true`、失敗則傳回 `false`。

`int rewind(int fp)`

重置檔案的讀寫位置指標到標案的開頭處。發生錯誤則傳回 `0`。檔案 `fp` 必須有效且用 `fopen()` 開啓的檔案。

`int rmdir(string dirname)`

用來刪除目錄。欲刪除的路徑必須是空目錄，且權限必須允許。發生錯誤則傳回 `0`。

`array stat(string filename)`

可蒐集檔案的相關資訊，並傳回陣列。陣列的元素如下：

7. 裝置名稱 (device)
8. inode 值
9. inode 檔案模式
10. 參照到該檔的名稱總數，含 link 檔。
11. 擁有者的 UID
12. 擁有者的 GID
13. 若為字元或裝置設備則為裝置型態 (限系統支援 `st_blksize` 方有效，Windows 系列傳回 `-1`)
14. 位元組為單位的檔案長度
15. 最後存取時間
16. 最後修改時間
17. 最後更動時間

18. 檔案系統的區塊大小 (限系統支援 `st_blksize` 方有效, Windows 系列傳回 -1)
19. 匹配的區塊數

`array lstat(string filename)`

可蒐集連結檔案的相關資訊, 並傳回陣列。陣列的元素如下:

1. 裝置名稱 (device)
2. inode 值
3. 連結 (link) 到該檔的總數。
4. 擁有者的 UID
5. 擁有者的 GID
6. 若為字元或裝置設備則為裝置型態 (限系統支援 `st_blksize` 方有效, Windows 系列傳回 -1)
7. 位元組為單位的檔案長度
8. 最後存取時間
9. 最後修改時間
10. 最後更動時間
11. 檔案系統的區塊大小 (限系統支援 `st_blksize` 方有效, Windows 系列傳回 -1)
12. 匹配的區塊數

`int symlink(string target, string link)`

可用來建立符號連結。從參數 `target` 連結到 `link` 處。

`string tempnam(string dir, string prefix)`

可建立一個臨時檔, 檔名不會與同目錄檔案重覆。所傳回字串就是臨時檔名。若發生錯誤則傳回空字串。

範例:

```
$tmpfname = tempnam( "/tmp", "HAHA" );
```

`int touch(string filename, int [time])`

用來設定最後修改時間。若有指定參數 `time`, 則依指定; 若無指定, 則為伺服器時間。

`int umask(int [mask])`

用來設定 PHP 的 `umask` 值。此值是 `mask` 與 `0777` 運算的值, 會傳回舊的 `umask` 值。當參數 `mask` 不存在時, 本函數則傳回目前的 `umask` 值。

`int unlink(string filename)`

刪除檔案 `filename`。當發生錯誤則傳回 `0` 或 `false`。

3.11 動態繪圖函數

PHP 著重於處理並提可程式化圖形的解決方案，擁有能產生規則的幾何圖形、修改圖形的函數，能巧妙處理文字、字型及色彩甚至可控制每一個像素。我們經常需要根據使用者的輸入值而動態顯示、修改圖形，PHP 提供可程式化的架構來達成此動態圖形目的。

`array getimagesize(string filename, array [imageinfo])`

本函數可用來取得 GIF、JPEG 及 PNG 三種 WWW 上圖片的高與寬，不需要安裝 GD library 就可以使用本函數。傳回的陣列有四個元素。傳回陣列的第一個元素（索引值 0）是圖片的高度，單位是像素（pixel）。第二個元素（索引值 1）是圖片的寬度。第三個元素（索引值 2）是圖片的檔案格式，其值 1 為 GIF 格式、2 為 JPEG/JPG 格式、3 為 PNG 格式。第四個元素（索引值 3）為圖片的高與寬字串，`height=xxx width=yyy`。可省略的參數 `imageinfo` 用來取得一些圖片的相關訊息。

範例：

```
function MyImg($imgfile) {
    $size = GetImageSize($imgfile);
    echo "<img src=\"\$imgfile\" $size[3]>";
}
MyImg("img/img1.gif");
MyImg("img/img2.png");
```

`int imagearc(int im, int cx, int cy, int w, int h, int s, int e, int col)`

用來畫弧線。`cx`、`cy` 為橢圓心坐標，`w` 為水平軸長，`h` 為垂直軸長，`s` 及 `e` 分別為起始角與結束角，`col` 為弧線顏色。`im` 表示圖形 handle。

`int imagechar(int im, int font, int x, int y, string c, int col)`

用來書寫橫向的字元。`font` 表示字體的大小，從最小的 1 起，`x`、`y` 是寫入字元的坐標值，`c` 為欲寫出的字元，`col` 為字的顏色。`im` 表示圖形的 handle。

`int imagecharup(int im, int font, int x, int y, string c, int col)`

用來書寫直式的字元。參數定義同上一個函數。

```
int imagecolorallocate(int im, int red, int green, int blue)
```

用來指定圖形的顏色，供其它繪圖函數使用。im 表示圖形 handle。red、green、blue 是色彩三原色，其值從 0 至 255。

範例：

```
$white = ImageColorAllocate($im, 255,255,255);  
$black = ImageColorAllocate($im, 0,0,0);
```

```
int imagecolortransparent(int im, int [col])
```

用來指定某色為透明背景。im 為使用 imagecreate() 打開圖形 handle。col 為 ImageColorAllocate() 所指定的顏色。傳回值為新的透明背景色。

```
int imagecopyresized(int dst_im, int src_im, int dstX, int dstY, int srcX, int srcY, int dstW, int dstH, int srcW, int srcH)
```

可複製新圖，並重新調整圖片尺寸大小。dst_im 及 src_im 為圖片 handle。dstX、dstY、srcX、srcY 分別為目的及來源坐標。dstW、dstH、srcW、srcH 分別為來源及目的寬及高，若欲調整新圖的尺寸利用此設定。

```
int imagecreate(int x_size, int y_size)
```

用來建立一張空圖形。x_size、y_size 為圖形的尺寸，單位為像素(pixel)。

```
int imagedashedline(int im, int x1, int y1, int x2, int y2, int col)
```

在圖形上畫出一條虛線。從 x1、y1 連接到 x2、y2，參數 col 為虛線的顏色。

```
int imagedestroy(int im)
```

將圖片 handle 解構，釋於記憶體空間。im 為 ImageCreate() 所建立的圖片 handle。

```
int imagefill(int im, int x, int y, int col)
```

將圖片坐標 (x,y) 所在區域著色。col 表示欲塗上的顏色。

```
int imagefilledpolygon(int im, array points, int num_points, int col)
```

將圖片封閉多邊形區域著色。`points` 為陣列，代表多邊形區域，陣列的偶數元素為 X 坐標，奇數元素為 Y 坐標，如 `points[0] = x0`、`points[1] = y0`、`points[2] = x1`、`points[3] = y1`。`num_points` 為多邊形的轉折點數目。`col` 表示欲塗上的顏色。

```
int imagefilledrectangle(int im, int x1, int y1, int x2,
int y2, int col)
```

將圖片的封閉長方形區域著色。`x1`、`y1` 及 `x2`、`y2` 分別為矩形對角線的坐標。`col` 表示欲塗上的顏色。

```
int imagefilltoborder(int im, int x, int y, int border, int
col)
```

將圖片中指定的顏色做為邊界，著色在其中封閉區域中。`x`、`y` 為著色區內坐標，`border` 為顏色值，表示填入顏色的邊界範圍。`col` 表示欲塗上的顏色。

```
int imagefontheight(int font)
```

用來取得指定字型的高度，單位為像素 (pixel)。

```
int imagefontwidth(int font)
```

用來取得指定字型的寬度，單位為像素 (pixel)。

```
int imageinterlace(int im, int [interlace])
```

用來決定圖片是否使用交錯式顯示 (`interlace`)。欲使用交錯式顯示，則將 `interlace` 值設為 1。否則設為 0。傳回值 1 表示已設定成交錯式、0 表示已設成非交錯式。

```
int imageline(int im, int x1, int y1, int x2, int y2, int
col)
```

在圖形上畫出一條實線。從 `x1`、`y1` 連接到 `x2`、`y2`，參數 `col` 為實線的顏色。

```
int imageloadfont(string file)
```

用來載入使用者自訂的點陣字型。傳回值為字型代號，為大於 5 的整數。

```
int imagepolygon(int im, array points, int num_points, int
col)
```

可在圖片上繪出多邊形。`points` 為陣列，代表多邊形區域，陣列的偶數元素為 X 坐標，奇數元素為 Y 坐標，例如 `points[0] = x0`、`points[1] = y0`、

`points[2] = x1`、`points[3] = y1`。 `num_points` 為多邊形的轉折點數目。
`col` 表示多邊形線的顏色。

```
int imagerectangle(int im, int x1, int y1, int x2, int y2,  
int col)
```

可在圖片上繪出長方形。參數 `x1`、`y1` 及 `x2`、`y2` 分別為矩形對角線的坐標。
`col` 表示矩形邊框的顏色。

```
int imagesetpixel(int im, int x, int y, int col)
```

可在圖片上繪出一點。`x`、`y` 為欲繪點的坐標，`col` 表示該點的顏色。

```
int imagestring(int im, int font, int x, int y, string s,  
int col)
```

在圖片上繪出水平的橫式字串。`font` 為字形，設為 1 到 5 表示使用內定
字形。`x`、`y` 為字串起點坐標。字串的內容放在 `s` 上。`col` 表示字串的顏色。

```
int imagestringup(int im, int font, int x, int y, string  
s, int col)
```

在圖片上繪出鉛直的直式字串。變數設定與上一個函數同。

```
int imagesx(int im)
```

取得圖片的寬。

```
int imagesy(int im)
```

取得圖片的高。

```
array ImageTTFBBox(int size, int angle, string fontfile,  
string text)
```

用來計算並傳回 TTF 文字區域框 (bounding box) 大小。`size` 為字形的
尺寸；`angle` 為字型的角度；`fontfile` 為字型檔名稱，亦可是遠端的檔案；
`text` 是字串內容。傳回值為陣列，包括了八個元素，分別為左下的 `x`、`y` 坐標，
右下角的 `x`、`y` 坐標，以及右上及左上的 `x`、`y` 坐標。

```
array ImageTTFText(int im, int size, int angle, int x, int  
y, int col, string fontfile, string text)
```

將 TTF (TrueType Fonts) 字型文字寫入圖片。`size` 為字形的尺寸；
`angle` 為字型的角度，順時針計算；`x`、`y` 為文字的坐標值 (原點為左上角)；`col`
為字的顏色；`fontfile` 為字型檔名稱，亦可是遠端的檔案；`text` 是字串內容。

傳回值為陣列，包括了八個元素，分別為左下的 x 、 y 坐標，右下角的 x 、 y 坐標，以及右上及左上的 x 、 y 坐標。

int imagecolorat(int im, int x, int y)

可取得圖形中某指定點的顏色索引值 (index)。

int imagecolorclosest(int im, int red, int green, int blue)

用來計算調色盤中與指定 RGB 顏色最接近的顏色。red、green、blue 分別為紅、綠、藍三原色，也就是 RGB 值。

int imagecolorexact(int im, int red, int green, int blue)

用來計算調色盤中指定 RGB 顏色的色素索引值。red、green、blue 分別為紅、綠、藍三原色，也就是所謂的 RGB 值。若調色盤中沒有指定的顏色，則傳回 -1。

int imagecolorresolve(int im, int red, int green, int blue)

用來計算調色盤中指定 RGB 顏色的色素索引值。red、green、blue 分別為紅、綠、藍三原色，也就是所謂的 RGB 值。若無指定顏色則取得最接近顏色。此函數必定會有索引值傳回。

boolean imagecolorset(int im, int index, int red, int green, int blue)

用來設定調色盤上指定索引的顏色值。index 為索引值。red、green、blue 分別為紅、綠、藍三原色，也就是所謂的 RGB 值。此函數適合將圖片上某顏色改掉，即利用索引的顏色更改圖片顏色。

array imagecolorsforindex(int im, int index)

用來取得調色盤上指定索引的顏色值。index 為索引值。陣列傳回值分別為紅、綠、藍三原色的值。

int imagecolorstotal(int im)

用來計算圖片的調色盤有幾種不同的顏色。

int imagepsloadfont(string filename)

用來載入 PostScript Type1 字型。參數 filename 為 PostScript Type1 字型的名稱。

void imagepsfreefont(int fontindex)

將已載入 **PostScript Type1** 字型卸下。參數 **fontindex** 為載入 **PostScript Type1** 字型的 **handle**。

```
int imagepsencodefont(string encodingfile)
```

將 **PostScript Type1** 字型轉換成向量字。

```
array imagepstext(int image, string text, int font, int size,  
int foreground, int background, int x, int y, int space, int  
tightness, float angle, int antialias_steps)
```

將 **PostScript Type1** 字型文字寫入圖片。**image** 為圖形。**text** 為要寫入的字串。**font** 為指定的字型。**size** 為字的大小。**foreground** 及 **background** 分別為前景及背景色。**x**、**y** 為文字坐標，以左下角開始計算。**space** 為文字間的空隔大小。**tightness** 為單字間的緊密度。**angle** 為角度。**antialias_steps** 為邊緣鋸齒狀修正等級，範圍為 4 至 16。傳回陣列有四個元素，分別為左下角的 **x**、**y** 值，右上角的 **x**、**y** 值。

```
array imagepsbbox(string text, int font, int size, int space,  
int width, float angle)
```

用來計算並傳回 **PostScript** 文字區域框 (**bounding box**) 大小。**text** 為要寫入的字串。**font** 則為指定的字型。**size** 為字的大小。**width** 為寬度。**angle** 為角度。傳回陣列有四個元素，分別為左下角的 **x**、**y** 值，右上角的 **x**、**y** 值。

```
int imagecreatefrompng(string filename);
```

用來取出一張 **PNG** 圖形，取出當背景或者基本的畫布使用。**filename** 可以是本地端的檔案，也可以是 **URL** 位址。傳回值為 **PNG** 檔案代碼，可供其它函數使用。本函數在 **PHP 3.0.13** 版之後才支援。

```
int imagepng(int im, string filename)
```

用來建立一張 **PNG** 圖形。**im** 為使用 **ImageCreate()** 所建立的圖片代碼。**filename** 可省略，若無 **filename**，則會將圖片送到瀏覽器端，送出圖片之前要先送出使用 **Content-type: image/png** 的標頭字串 (**header**) 到瀏覽器端，以順利傳輸圖片。本函數在 **PHP 3.0.13** 版之後才支援。

```
int imagecreatefromgif(string filename)
```

用來取出一張 **GIF** 圖形，取出當背景或者基本的畫布使用。**filename** 可以是本地端的檔案，也可以是 **URL** 位址。傳回值為 **GIF** 的檔案代碼，可供其它函數使用。本函數產生的 **GIF** 圖，有版權的問題，在商業上的使用要多加考量。

```
int imagegif(int im, string [filename])
```

用來建立一張 GIF 圖形。im 為使用 ImageCreate() 所建立的圖片代碼。filename 可省略，若無本參數 filename，則會將圖片送到瀏覽器端，在送出圖片之前要先送出使用 Content-type: image/gif 的標頭字串 (header) 到瀏覽器端，以順利傳輸圖片。若要使用透明背景的 GIF 圖，也就是 GIF89a 格式，要先使用 ImageColorTransparent() 設定透明背景。本函數產生的 GIF 圖，有版權的問題，在商業上的使用要多加考量。

3.12 使用 MySQL 資料庫函數

```
int mysql_connect(string hostname :port, string username, string password)
```

本函數建立與 MySQL 伺服器的連線。其中所有的參數都可省略。當使用本函數卻不加任何參數時，參數 hostname 的內定值為 localhost、參數 username 的內定值為 PHP 執行行程的擁有者、參數 password 則為空字串 (即沒有密碼)。而參數 hostname 後面可以加冒號與埠號，代表使用那個埠與 MySQL 連接。

```
int mysql_close(int [link_identifier])
```

關閉與 MySQL 資料庫伺服器的連線。若無指定參數 link_identifier 則會關閉最後一筆。用 mysql_pconnect() 連線則無法使用本函數關閉。成功傳回 true、失敗傳回 false 值。

```
int mysql_affected_rows(int [link_identifier])
```

可得到 MySQL 最後查詢操作 INSERT、UPDATE 或 DELETE 所影響的列 (row) 數目。若最後的查詢 (query) 是使用 DELETE 而且沒有使用 WHERE 命令，則會刪除全部資料，本函數將傳回 0。

```
int mysql_create_db(string database name, int [link_identifier])
```

用來建立新的資料庫 (database)。建立前必須先與伺服器連線。

```
int mysql_data_seek(int result_identifier, int row_number)
```

可移動內部傳回的列指標到指定 row_number 去。之後若使用 mysql_fetch_row() 可以傳回新列的值。成功傳回 true、失敗則傳回 false。

```
int mysql_db_query(string database, string query, int
```

`[link_identifier]`)

用來送出查詢字串 (query) 到後端的 MySQL 資料庫中。可省略的 `link_identifier` 若不存在，程式會自動尋找其它 `mysql_connect()` 連線後的連線代碼。發生錯誤時會傳回 `false`，其它沒錯誤時則傳回它的傳回代碼。

```
int mysql_drop_db(string database_name, int  
[link_identifier])
```

移除已存在的資料庫。成功傳回 `true`、失敗則傳回 `false`。

```
int mysql_errno(int [link_identifier])
```

可得到 MySQL 資料庫伺服器的錯誤代碼。作為 PHP 與 MySQL 的除錯用。

範例：

```
mysql_connect("marliesle");  
echo mysql_errno().": ".mysql_error()."<BR>";  
mysql_select_db("nonexistentdb");  
echo mysql_errno().": ".mysql_error()."<BR>";  
$conn = mysql_query("SELECT * FROM nonexistenttable");  
echo mysql_errno().": ".mysql_error()."<BR>";
```

```
string mysql_error(int [link_identifier])
```

可得到 MySQL 資料庫伺服器的錯誤訊息。可與 `mysql_errno()` 一起作為 PHP 與 MySQL 的除錯用。

```
array mysql_fetch_array(int result, int [result_typ])
```

用來將查詢結果 `result` 拆到陣列變數中。若 `result` 沒有資料，則傳回 `false` 值。

```
object mysql_fetch_field(int result, int [field_offset])
```

傳回的物件資料為 `result` 的欄位 (Column) 資訊。傳回物件的屬性如下：

- `name`：欄位名稱
- `table`：欄位所在表格的資料表名稱
- `max_length`：欄位的最大長度
- `not_null`：若為 1 表示本欄位不能是空的 (null)
- `primary_key`：若為 1 表示本欄位是主要鍵 (primary key)
- `unique_key`：若為 1 表示本欄位為不可重覆鍵 (unique key)
- `multiple_key`：若為 1 表示本欄位為可重覆鍵 (non-unique key)
- `numeric`：若為 1 表示本欄位為數字型態 (numeric)

- `blob`：若為 1 表示本欄位為位元型態 (BLOB)
- `type`：欄位型態
- `unsigned`：若為 1 表示本欄位為無記號 (unsigned)
- `zerofill`：若為 1 表示本欄位為被零填滿 (zero-filled)

`array mysql_fetch_lengths(int result)`

將 `mysql_fetch_row()` 處理過的最後一列資料的各欄位資料的最大長度放在陣列變數中。若執行失敗則傳回 `false` 值。傳回陣列第一筆資料索引值是 0。

`object mysql_fetch_object(int result, int [result_typ])`

用來將查詢結果 `result` 拆到物件變數中。使用方法和 `mysql_fetch_array()` 幾乎相同，不同地方在於本函數傳回資料是物件不是陣列。若 `result` 沒有資料，則傳回 `false` 值。另外，取回物件資料索引只能是文字而不能數字。

`array mysql_fetch_row(int result)`

用來將查詢結果 `result` 之單列拆到陣列變數中。陣列的索引是數字索引，第一個的索引值是 0。若 `result` 沒有資料，則傳回 `false` 值。

`string mysql_field_name(int result, int field_index)`

用來取得指定欄位的名稱。

`int mysql_field_seek(int result, int field_offset)`

設定目前的指標到傳回值特定欄位中。

`string mysql_field_table(int result, int field_offset)`

可以得到目前所在欄位的資料表名。

`string mysql_field_type(int result, int field_offset)`

可以得到目前所在欄位的型態格式。

範例：

```
mysql_connect("localhost:3306");
mysql_select_db("wisconsin");
$result = mysql_query("SELECT * FROM onek");
$fields = mysql_num_fields($result);
$rows = mysql_num_rows($result);
$i = 0;
$table = mysql_field_table($result, $i);
```

```

    echo "資料表 '$table.'" 有 ".$fields." 欄及 ".$rows."
    列。<br>";
    echo "本資料表的欄位如下<br>";
    while ($i < $fields) {
        $type = mysql_field_type ($result, $i);
        $name = mysql_field_name ($result, $i);
        $len = mysql_field_len ($result, $i);
        $flags = mysql_field_flags ($result, $i);
        echo $type." ".$name." ".$len." ".$flags."<br>";
        $i++;
    }
    mysql_close();

```

```
string mysql_field_flags(int result, int field_offset);
```

可以得到目前所在欄位的旗標。若一個欄位有數種屬性旗標，則傳回的旗標為這些屬性連起來的字串，每個屬性用空格隔開。

```
int mysql_field_len(int result, int field_offset)
```

可以得到目前所在欄位長度。

```
boolean mysql_free_result(int result)
```

可以釋放目前 MySQL 資料庫 query 傳回所占用的記憶體。

```
int mysql_insert_id(int [link_identifier])
```

可以得到最後一次使用 INSERT 到 MySQL 資料庫的執行 ID。

```
int mysql_list_fields(string database_name, string
table_name, int [link_identifier])
```

可得到指定資料表所有欄位。欄位資訊給 mysql_field_flags()、mysql_field_len()、mysql_field_name() 及 mysql_field_type() 等函數使用。若有錯誤則傳回 -1。

```
int mysql_list_dbs(int [link_identifier])
```

可得到 MySQL 伺服器可用資料庫。

```
int mysql_list_tables(string database, int
[link_identifier])
```

可得到指定資料庫中的所有資料表名稱。

```
int mysql_num_fields(int result)
```

可得到傳回欄位的數目。

```
int mysql_num_rows(int result)
```

可得到傳回列的數目。

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

和 `mysql_connect()` 雷同。不同地方在於本函數開啓資料庫時，程式會先尋找是否曾經執行過本函數，若執行過則傳回先前執行的 ID。本函數無法使用 `mysql_close()` 關閉資料庫。

```
int mysql_query(string query, int [link_identifier])
```

送出 `query` 字串供 MySQL 做相關的處理或者執行。若沒有指定 `link_identifier`，程式會自動尋找最近開啓的 ID。當 `query` 字串是 UPDATE、INSERT 及 DELETE 時，傳回的可能是 `true` 或者 `false`；查詢的字串是 SELECT 則傳回新的 ID 值。

```
int mysql_result(int result, int row, mixed field)
```

取得一格 `query` 的結果。`field` 可以是欄位名稱、順序或者是 `FieldName.TableName` 的格式。傳回資料量少時，可使用本函數來處理。

```
int mysql_select_db(string database_name, int  
[link_identifier])
```

選擇 MySQL 伺服器中的資料庫，供之後的資料查詢作業 (`query`) 處理。成功傳回 `true`，失敗則傳回 `false`。

```
string mysql_tablename(int result, int i)
```

可取得資料表名稱字串，一般配合 `mysql_list_tables()` 使用，取得該函數傳回的數字名稱字串。

單元三 MySQL 簡介

第四章 MySQL 介紹

4.1 簡介

MySQL 是一個快速、多執行緒、多使用者和強壯的 SQL 資料庫伺服器。對 Unix 和 OS/2 平台，MySQL 基本上是免費的；但對微軟平台，你在 30 天的試用期後必須獲得一個 MySQL 使用授權。

4.2 MySQL 使用者名稱和密碼

由於 MySQL 使用者名稱和密碼的方法與 Unix 或 windows 使用的方式有很多不同之處：

- MySQL 使用於認證目的的使用者名稱，與 Unix 使用者名稱(登錄名字)或 Windows 使用者名稱無關。預設地，大多數 MySQL 客戶嘗試使用當前 Unix 使用者名稱作為 MySQL 使用者名稱登錄，但是這僅僅為了方便。客戶程式允許用 `-u` 或 `--user` 選項指定一個不同的名字，這意味著無論如何你不能使得一個資料庫更安全，除非所有的 MySQL 使用者名稱都有密碼。任何人可以試圖用任何名字連接伺服器，而且如果他們指定了沒有密碼的任何名字，他們將成功。
- MySQL 使用者名稱最長可以是 16 個字元；典型地，Unix 使用者名稱限制為 8 個字元。
- MySQL 密碼與 Unix 密碼沒關係。在你使用登錄到一台 Unix 機器密碼和你使用在那台機器上存取一個資料庫的密碼之間沒有必要有關聯。
- MySQL 加密密碼使用了一個 Unix 登錄期間所用的不同算法。

當你想要存取一個 MySQL 伺服器時，MySQL 客戶程式一般要求你指定連接參數：你想要連接的主機、你的使用者名稱和你的密碼。例如，`mysql` 客戶可以像這樣啟動(可選的參數被包括在 “[” 和 “]” 之間)：

```
shell> mysql [-h host_name] [-u user_name] [-p your_pass ]  
-h, -u 和 -p 選項的另一種形式是 --host=host_name、  
--user=user_name 和 --password=your_pass。注意在 -p 或 --password=  
與跟隨它後面的密碼之間沒有空格。
```

對於指令列沒有的連接參數，`mysql` 使用預設值：
預設主機名是 `localhost`。

預設使用者名稱是你的 **Unix** 登錄名。

如果沒有 **-p**，則沒有提供密碼。所以，對一個 **Unix** 用戶 **joe**，下列指令是相同的：

```
shell>mysql -h localhost -u joe
shell>mysql -h localhost
shell>mysql -u joe
shell>mysql
```

在指令列上指定一個密碼是不安全的！使用一個在指令列上 **-p***your_pass* 或 **--password=your_pass** 的選項。這很方便但是不安全，因為你的密碼對作業系統來說(例如使用 **ps** 指令)變得可見，它可以被其他的用戶調用來顯示指令列。使用一個 **-p** 或 **--password** 選項(沒有指定 *your_pass* 值)。在這種情況下，客戶程式請求來自終端的密碼：

```
shell>mysql -u user_name -p
Enter password: *****
```

客戶回應 “*” 字元到作為輸入你的密碼的終端使得旁觀者不能看見它。因為它對其他用戶不可見，與在指令列上指定它相比，這樣進入你的密碼更安全。

4.3 MySQL 提供的權限

權限	列	上下文
<i>select</i>	<i>Select_priv</i>	資料表
<i>insert</i>	<i>Insert_priv</i>	資料表
<i>update</i>	<i>Update_priv</i>	資料表
<i>delete</i>	<i>Delete_priv</i>	資料表
<i>index</i>	<i>Index_priv</i>	資料表
<i>alter</i>	<i>Alter_priv</i>	資料表
<i>create</i>	<i>Create_priv</i>	資料庫、資料表或索引
<i>drop</i>	<i>Drop_priv</i>	資料庫或資料表
<i>grant</i>	<i>Grant_priv</i>	資料庫或資料表
<i>references</i>	<i>References_priv</i>	資料庫或資料表
<i>reload</i>	<i>Reload_priv</i>	伺服器管理
<i>shutdown</i>	<i>Shutdown_priv</i>	伺服器管理
<i>process</i>	<i>Process_priv</i>	伺服器管理
<i>file</i>	<i>File_priv</i>	在伺服器上的文件存取

select、**insert**、**update** 和 **delete** 權限允許你在一個資料庫現有的資料表上執行操作。

SELECT 語句只有在他們真正從一個資料表中檢索行是才需要 **select** 權限，你可以執行某個 **SELECT** 語句，甚至沒有任何到伺服器上的資料庫裡的存取任何東西的許可。例如，你可使用 **mysql** 客戶作為一個簡單的計算器：

```
mysql> SELECT 1+1;
mysql> SELECT PI()*2;
```

index 權限允許你建立或拋棄(刪除)索引。

alter 權限允許你使用 **ALTER TABLE**。

create 和 **drop** 權限允許你建立新的資料庫和資料表，或拋棄(刪除)現存的資料庫和資料表。

file 權限給予你用 **LOAD DATA INFILE** 和 **SELECT ... INTO OUTFILE** 語句讀和寫伺服器上的文件，任何被授予這個權限的用戶都能讀或寫 **MySQL** 伺服器能讀或寫的任何文件。

其餘的權限用於管理性操作，它使用 **mysqladmin** 程式執行。

權限	權限擁有者允許執行的指令
<i>reload</i>	<i>reload, refresh, flush-privileges, flush-hosts, flush-logs, flush-tables</i>
<i>shutdown</i>	<i>shutdown</i>
<i>process</i>	<i>processlist, kill</i>

reload 指令告訴伺服器再讀入授權表，**refresh** 指令清洗所有授權表並打開和關閉記錄文件，**flush-privileges** 是 **reload** 的一個同義詞，其它 **flush-***指令執行類似 **refresh** 的功能，但是範圍更有限，並且在某些情況下可能更好用。例如，如果你只是想清洗記錄文件，**flush-logs** 比 **refresh** 是更好的選擇。

shutdown 指令關掉伺服器。

processlist 指令顯示在伺服器內執行的執行緒的資訊。**kill** 指令殺死伺服器執行緒。你總是能顯示或殺死你自己的執行緒，但是你需要 **process** 權限來顯示或殺死其他用戶啟動的執行緒。

4.4 權限的工作原理

MySQL 權限系統保證所有的用戶可以嚴格地做他們假定被允許做的事情。當你連接一個 **MySQL** 伺服器時，你的身份由你從那連接的主機和你指定的用戶名來決定，系統根據你的身份和你想做什麼來授予權限。

MySQL 在認定身份中考慮你的主機名和用戶名字，是因為有很小的原因假定一個給定的用戶在網際網路上屬於同一個人。例如，用戶從 **whitehouse.gov** 連接的 **bill** 不必和從 **microsoft.com** 連接 **bill** 是同一個人。**MySQL** 通過允許你區分在不同的主機上碰巧有同樣名字用戶來處理它：你可以對從

whitehouse.gov 連接授與 bill 一個權限集，而為從 microsoft.com 的連接授予一個不同的權限集。

MySQL 存取控制包含 2 個階段：

- 階段 1：伺服器檢查你是否允許連接。
- 階段 2：假定你能連接，伺服器檢查你發出的每個請求。看是否有足夠的權限執行它。例如，如果你從資料庫中一個資料表選取(select)行或從資料庫拋棄一個資料表，伺服器確定你對資料表有 select 權限或對資料庫有 drop 權限。

伺服器在存取控制的兩個階段使用在 mysql 的資料庫中的 user、db 和 host 資料表，在這些授權表中段落如下：

資料表名稱	<i>user</i>	<i>db</i>	<i>host</i>
範圍段落	<i>Host</i>	<i>Host</i>	<i>Host</i>
	<i>User</i>	<i>Db</i>	<i>Db</i>
	<i>Password</i>	<i>User</i>	
權限段落	<i>Select_priv</i>	<i>Select_priv</i>	<i>Select_priv</i>
	<i>Insert_priv</i>	<i>Insert_priv</i>	<i>Insert_priv</i>
	<i>Update_priv</i>	<i>Update_priv</i>	<i>Update_priv</i>
	<i>Delete_priv</i>	<i>Delete_priv</i>	<i>Delete_priv</i>
	<i>Index_priv</i>	<i>Index_priv</i>	<i>Index_priv</i>
	<i>Alter_priv</i>	<i>Alter_priv</i>	<i>Alter_priv</i>
	<i>Create_priv</i>	<i>Create_priv</i>	<i>Create_priv</i>
	<i>Drop_priv</i>	<i>Drop_priv</i>	<i>Drop_priv</i>
	<i>Grant_priv</i>	<i>Grant_priv</i>	<i>Grant_priv</i>
	<i>Reload_priv</i>		
	<i>Shutdown_priv</i>		
	<i>Process_priv</i>		
	<i>File_priv</i>		

對存取控制的第二階段(請求證實)，如果請求涉及資料表，伺服器可以另外參考 *tables_priv* 和 *columns_priv* 資料表。這些資料表的段落如下：

資料表名稱	<i>tables_priv</i>	<i>columns_priv</i>
範圍段落	<i>Host</i>	<i>Host</i>
	<i>Db</i>	<i>Db</i>
	<i>User</i>	<i>User</i>

	<i>Table_name</i>	<i>Table_name</i>
		<i>Column_name</i>
權限段落	<i>Table_priv</i>	<i>Column_priv</i>
	<i>Column_priv</i>	
其他段落	<i>Timestamp</i>	<i>Timestamp</i>
	<i>Grantor</i>	

每個授權表包含範圍段落和權限段落。

範圍段落決定資料表中每筆紀錄的範圍，即，紀錄適用的上下文。例如，一個 `user` 資料表紀錄的 `Host` 和 `User` 值為 `'thomas.loc.gov'` 和 `'bob'` 將被用於證實來自主機 `thomas.loc.gov` 的 `bob` 對伺服器的連接。同樣，一個 `db` 資料表條目的 `Host`、`User` 和 `Db` 段落的值是 `'thomas.loc.gov'`、`'bob'` 和 `'reports'` 將用在 `bob` 從主機聯接 `thomas.loc.gov` 存取 `reports` 資料庫的時候。`tables_priv` 和 `columns_priv` 資料表包含範圍段落，指出每個條目適用的資料表或資料表／列的組合。

對於檢查存取的目的，比較 `Host` 值是忽略大小寫的。`User`、`Password`、`Db` 和 `Table_name` 值是區分大小寫的。`Column_name` 值在 MySQL 3.22.12 或以後版本是忽略大小寫的。

4.4.1 階段一：連接證實

當你試圖聯接一個 MySQL 伺服器時，伺服器基於你的身份和你是否能通過供應正確的密碼驗證身份來接受或拒絕連接。如果不是，伺服器完全具結你的存取，否則，伺服器接受連接，然後進入階段 2 並且等待請求。

你的身份基於 2 個資訊：

- 你從那個主機連接
- 你的 MySQL 用戶名

身份檢查使用 3 個 `user` 資料表 (`Host`、`User` 和 `Password`) 範圍段落執行。伺服器只有在一個 `user` 資料表條目代表你的主機名和用戶名並且你提供了正確的密碼時才接受連接。

在 `user` 資料表範圍段落可以如下被指定：

- 一個 `Host` 值可以是主機名或一個 IP 數字，或 `'localhost'` 指出本地主機。
- 你可以在 `Host` 段落裡使用通用字元“%”和“_”。
- 一個 `Host` 值 '%' 代表任何主機名，一個空白 `Host` 值等價於 '%'。注意這些值代表能建立一個連接到你的伺服器的任何主機！

- 通用字元在 `User` 段落中不允許，但是你能指定空白的值，它代表任何名字。如果 `user` 資料表代表到來的連接的條目有一個空白的用戶名，用戶被認為是匿名用戶(沒有名字的用戶)，而非客戶實際指定的名字。這意味著一個空白的用戶名被用於在連接期間的進一步的存取檢查(即，在階段 2 期間)。
- `Password` 段落可以是空白的。這不意味著代表任何密碼，它意味著用戶必須不指定一個密碼進行連接。

非空白 `Password` 值代表加密的密碼。MySQL 不以任何人可以看純文本格式儲存密碼，相反，正在試圖聯接的一個用戶提供的密碼被加密(使用 `PASSWORD()` 函數)，並且與儲存了 `user` 資料表中的已經加密的版本比較。如果他們相同，密碼是正確的。

`user` 資料表的前 2 的欄位看起來像這樣：

```
+-----+-----+
| Host   | User   | ...
+-----+-----+
| %      | root   | ...
| %      | jeffrey | ...
| localhost | root   | ...
| localhost |       | ...
+-----+-----+
```

4.4.2 階段二：請求證實

一旦你建立了一個連接，伺服器進入階段 2。對在此連接上進來的每個請求，伺服器檢查你是否有足夠的權限來執行它，它基於你希望執行的操作類型。這正是在授權表中的權限段落發揮作用的地方。這些權限可以來自 `user`、`db`、`host`、`tables_priv` 或 `columns_priv` 資料表的任何一個。授權表用 `GRANT` 和 `REVOKE` 指令操作。

`user` 資料表在一個全局基礎上授予賦予你的權限，該權限不管當前的資料庫是什麼均適用。例如，如果 `user` 資料表授予你 `delete` 權限，你可以刪除在伺服器主機上從任何資料庫刪除行！換句話說，`user` 資料表權限是超級用戶權限。只把 `user` 資料表的權限授予超級用戶如伺服器或資料庫主管是明智的。對其他用戶，你應該把在 `user` 資料表中的權限設成 'N' 並且僅在一個特定資料庫的基礎上授權，使用 `db` 和 `host` 資料表。

`db` 和 `host` 資料表授予資料庫特定的權限。在範圍段落的值可以如下被指定：

- 通用字元“%”和“_”可被用於兩個資料表的 `Host` 和 `Db` 段落。
- 在 `db` 資料表的 '%Host' 值意味著“任何主機”，在 `db` 資料表中一個空白 `Host` 值

意味著“對進一步的資訊諮詢 host 資料表”。

- 在 host 資料表的一個 '%' 或空白 Host 值意味著“任何主機”。
- 在兩個資料表中的一個 '%' 或空白 Db 值意味著“任何資料庫”。
- 在兩個資料表中的一個空白 User 值相當匿名用戶。

db 和 host 資料表在伺服器啟動時被讀取和排序(同時它讀 user 資料表)。db 資料表在 Host、Db 和 User 範圍段落上排序，並且 host 資料表在 Host 和 Db 範圍段落上排序。對於 user 資料表，排序首先放置最特定的值然後最後最不特定的值，並且當伺服器尋找相同的紀錄時，它使用它找到的第一個相當。

tables_priv 和 columns_priv 資料表授予資料表和紀錄列特定的權限。在範圍段落值可以如下被指定：

- 通用字元“%”和“_”可用在使用在兩個資料表的 Host 段落。
- 在兩個資料表中的一個 '%' 或空白 Host 意味著“任何主機”。
- 在兩個資料表中的 Db、Table_name 和 Column_name 段落不能包含通用字元或空白。

tables_priv 和 columns_priv 資料表在 Host、Db 和 User 段落上被排序。這類似於 db 資料表的排序，儘管因為只有 Host 段落可以包含通用字元，但排序更簡單。

4.5 建立初始的 MySQL 權限

在安裝 MySQL 後，你可先執行 `scripts/mysql_install_db` 安裝初始的存取權限。`scripts/mysql_install_db` 啟動 `mysqld` 伺服器，然後初始化授權表，包含下列權限集合：

- MySQL root 用戶作為可做任何事情的一個超級用戶被創造。連接必須由本地主機發出。注意：此 Script 產生的 root 密碼是空的，因此任何人能以 root 而沒有一個密碼進行連接並且被授予所有權限。
- 一個匿名用戶被創造，他可對有一個 'test' 或以 'test_' 開始的名字的資料庫做任何時期事情，連接必須由本地主機發出。這意味著任何本地用戶能連接並且視為匿名用戶。
- 其他權限被拒絕。例如，一般用戶不能使用 `mysqladmin shutdown` 或 `mysqladmin processlist`。

注意：對 win32 的初始權限是不同的。

既然你的安裝初始時廣開大門，你首先應該做的事情之一是為 MySQL root 用戶指定一個密碼。你可以做如下(注意，你使用 `PASSWORD()` 函數指定密碼)：

```
shell> mysql -u root mysql
mysql> UPDATE user SET Password=PASSWORD('new_password')
WHERE user='root';
mysql> FLUSH PRIVILEGES;
```

在 MySQL 3.22 和以上版本中，你可以使用 SET PASSWORD 語句：

```
shell> mysql -u root mysql
mysql> SET PASSWORD FOR root=PASSWORD('new_password');
```

設置密碼的另一種方法是使用 mysqladmin 指令：

```
shell> mysqladmin -u root password new_password
```

注意：如果你使用第一種方法在 user 資料表裡直接更新密碼，你必須告訴伺服器再次讀入授權表(用 FLUSH PRIVILEGES)，因為否則改變將不被注意到。

一旦 root 密碼被設置，此後當你作為 root 與伺服器連接時，你必須供應那個密碼。

你可能希望讓 root 密碼為空白以便當你施行附加的安裝時，你不需要指定它或測試，但是請注意在任何真實的、正式的環境中使用 MySQL 之前，請設定 root 密碼。

看看 scripts/mysql_install_db 的內容，看它如何安裝預設的權限。你可用它作為一個研究如何增加其他用戶的基礎。

如果你想要初始的權限不同於上面描述的那些，在你執行 mysql_install_db 之前，你可以修改它。

4.6 新增使用者

你可以有 2 個不同的方法增加用戶：通過使用 GRANT 語句或通過直接操作 MySQL 授權表。比較好的方法是使用 GRANT 語句，因為他們是更簡明並且好像錯誤少些。

下面的例子顯示出如何使用 mysql 客戶安裝新用戶。這些例子假定權限根據以前的章節描述的預設被安裝。這意味著為了改變，你必須在 mysqld 正在運行同一台機器上，你必須作為 MySQL root 用戶連接，並且 root 用戶必須對 mysql 資料庫有 insert 權限和 reload 管理權限。另外，如果你改變了 root 用戶密碼，你必須於 mysql 指令中加上 -p 指定它。

你可以使用 GRANT 語句增加新用戶：

```
shell> mysql --user=root mysql
mysql> GRANT ALL PRIVILEGES ON *.* TO monty@localhost
      IDENTIFIED BY 'something' WITH GRANT OPTION;
mysql> GRANT ALL PRIVILEGES ON *.* TO monty%"
      IDENTIFIED BY 'something' WITH GRANT OPTION;
mysql> GRANT RELOAD,PROCESS ON *.* TO admin@localhost;
mysql> GRANT USAGE ON *.* TO dummy@localhost;
上面這些 GRANT 語句安裝 3 個新用戶：monty、admin、dummy
```

monty

可以從任何地方連接伺服器的一個完全的超級用戶，但是必須使用一個密碼 'something'。注意，我們必須對 `monty@localhost` 和 `monty@%"` 發出 `GRANT` 語句。如果我們增加 `localhost` 條目，對 `localhost` 的匿名用戶條目在我們從本地主機連接時由 `mysql_install_db` 建立的紀錄將優先考慮，因為它有更特定的 `Host` 段落值，所以以 `user` 資料表排列順序看更早到來。

admin

可以從 `localhost` 沒有一個密碼進行連接並且被授予 `reload` 和 `process` 管理權限的用戶。這允許用戶執行 `mysqladmin reload`、`mysqladmin refresh` 和 `mysqladmin flush-*` 指令，還有 `mysqladmin processlist`。沒有授予資料庫有關的權限。他們能在以後通過發出另一個 `GRANT` 語句授權。

dummy

可以不用一個密碼連接的一個用戶，但是只能從本地主機。全部權限被設置為 'N' --USAGE 權限類型允許你無需權限就可設置一個用戶。它假定你將在以後授予資料庫相關的權限。

你也可以直接通過發出 `INSERT` 語句增加同樣的用戶存取資訊，然後告訴伺服器再次裝入授權表：

```
shell> mysql --user=root mysql
mysql>          INSERT          INTO          user
VALUES('localhost', 'monty', PASSWORD('something'),
'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y')
mysql> INSERT INTO user VALUES('%','monty',PASSWORD('something'),
'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y')
mysql> INSERT INTO user SET Host='localhost', User='admin',
Reload_priv='Y', Process_priv='Y';
mysql> INSERT INTO user (Host,User,Password)
VALUES('localhost','dummy','');
mysql> FLUSH PRIVILEGES;
```

4.7 如何設定密碼

當你使用 `INSERT` 或 `UPDATE` 語句儲存一個非空的密碼時，你必須使用 `PASSWORD()` 函數加密它。這是因為在 `user` 資料表中以加密形式儲存密碼，而不是純文字。如果你忘記這個事實，你可能像這樣試圖設置密碼：

```
shell> mysql -u root mysql
mysql>          INSERT          INTO          user          (Host,User,Password)
```

```
VALUES('%','jeffrey','biscuit');
mysql> FLUSH PRIVILEGES
```

結果是純文字值 'biscuit' 作為密碼被儲存在 `user` 資料表中。在用戶 `jeffrey` 試圖用這個密碼連接伺服器時，`mysql` 客戶用 `PASSWORD()` 加密它並且將結果送給伺服器，伺服器比較在 `user` 資料表中的值（它是純文字值 'biscuit'）和加密的密碼（而不是 'biscuit'），比較失敗並且伺服器拒絕連接：

```
shell> mysql -u jeffrey -pbiscuit test
Access denied
```

因為當他們被插入 `user` 資料表時，密碼必須被加密，相反，`INSERT` 語句應該像這樣被指定：

```
mysql> INSERT INTO user (Host,User,Password)
VALUES('%','jeffrey',PASSWORD('biscuit'));
```

當你使用 `SET PASSWORD` 語句時，你也必須使用 `PASSWORD()` 函數：

```
mysql> SET PASSWORD FOR jeffrey@"%" = PASSWORD('biscuit');
```

如果你使用 `GRANT ... IDENTIFIED BY` 語句或 `mysqladmin password` 指令設置密碼，`PASSWORD()` 函數是不必要的。他們都考慮到為你加密密碼，多以你可像這樣指定一個密碼 'biscuit'：

```
mysql> GRANT USAGE ON *.* TO jeffrey@"%" IDENTIFIED BY
'biscuit';
```

或

```
shell> mysqladmin -u jeffrey password biscuit
```

注意：`PASSWORD()` 不是以在 `Unix` 密碼加密的同樣方法施行密碼加密。你不應該假定如果你的 `Unix` 密碼和你的 `MySQL` 密碼是一樣的，`PASSWORD()` 將導致與在 `Unix` 密碼文件被儲存的同樣的加密值。

4.8 資料的匯入和匯出

資料匯出

由 `mysqldump` 引申而來的話題，當我們不想把資料表的記錄用 `SQL` 的語法備份時，我們可以用資料匯出的方式來備份記錄，這種做法的好處有很多，其中不乏可供其他的 `DBMS` 來轉換資料。如此一來，

資料表內容之 `portability` 大增。一般用法如下：

```
select * into outfile "/tmp/stud2.txt"
fields terminated by ',' enclosed by ''
```

```
lines terminated by '\n' from student ;
```

語法說明

基本的概念為：將 `inventory` 資料表內所有的記錄都匯出到 `/tmp/stud2.txt` 檔案中，而每個欄位值用雙引號括起來，並以逗號做為分隔記號，每一行，則以 `\n` 為分行符號。參考內容如下：

```
"5851001","張三","台北市中山南路","1979-01-12 00:00:00","F"
"5851002","李四","台北市復興北路","1980-12-24 00:00:00","M"
"5851003","王五","台北縣新莊市中正路","1981-08-15
00:00:00","M"
```

.....

其實 `into outfile` 是附加在 `select syntax` 語法中，所以，可以在 `from` 後面，再接上一些 `where` 條件子句。如：

```
select * into outfile "/tmp/stud2.txt"
fields terminated by ',' enclosed by '"'
lines terminated by '\n' from student
where Address like '台北市%';
"5851001","張三","台北市中山南路","1979-01-12 00:00:00","F"
"5851002","李四","台北市復興北路","1980-12-24 00:00:00","M"
```

.....

如此，使可以將住在台北市的資料匯出來了。

資料匯入

談完資料匯出之後，緊接著就介紹資料匯入的部份。在這裡我們所提到的匯入資料是指一般的純文字檔，像 `SQL Server` 或 `Access` 甚至於 `Excel` 所匯出的資料等。

執行匯入的 `SQL` 語法如下：

```
load data infile "/tmp/Inventory.txt"
into table inventory
fields terminated by ',' enclosed by '"'
lines terminated by '\n' ;
```

語法說明

上述的 `SQL` 語法大致上跟資料匯出的語法相同，便不再多做解釋。另一種資料匯入的方式，是採用 `command-line` 的 `MySQL` 相關指令 - `mysqlimport`。用法如下：

```
mysqlimport -u root \
--fields-terminated-by=',' \
--fields-enclosed-by='"' \
--lines-terminated-by='\n'
```

`mis student.txt`

大致上類似於 `LOAD DATA INFILE...`，不同的是，它是在指令列執行，也就是 `shell` 模式下。

4.9 備份和復原資料庫

`mysqldump`

`mysqldump` 是 MySQL 備份的主要工具，是在 `command-line` 下的一個可執行的指令，它的參數設定提供了強大的功能，但對於一般的備份，甚至連特殊參數都不用設定即可備份。主要的使用方式如下：

Usage: `mysqldump [OPTIONS] database [tables]`

OR `mysqldump [OPTIONS] --databases [OPTIONS] DB1 [DB2 DB3...]`

OR `mysqldump [OPTIONS] --all-databases [OPTIONS]`

指令說明：

第一種方式是指，在 `mysqldump` 後面，加上欲備份之資料庫。中括號的部份則是選擇性項目，可加可不加入的部份。如：

`mysqldump mis student > dump.file`

則是將 `mis` 資料庫的 `student` 資料表備份到 `dump.file` 檔案裡。

第二種方式為指定“哪些”欲備份的資料庫。如：

`mysqldump -B mis test > dump.file`

注意上面的 `-B`，則代表 `--databases`，而 `mis` 與 `test` 則是欲備份的資料庫名稱。

第三種方式則是將所有的資料庫都備份。如：

`mysqldump -A > dump.file`

同樣地，`-A` 代表 `--all-databases`，連 MySQL 基本的權限資料庫(`mysql`)也都會備份進去。

以上是粗淺的介紹使用方式，現在則將說明一般常用的備份方法。一般說來，下 `mysqldump` 的語法及參數如下：

`mysqldump --opt -A > dump.file`

其中 `--opt` 其實就包含了五個參數，分別如下：

`--quick`。立刻執行備份的動作。

`--add-drop-table`。在建立資料表 SQL 語法前，先用 `DROP TABLE` 語法將已經存在的資料表刪除。

`--add-locks`。在每個資料表 `dump` 前加上 `LOCK TABLES` 語法及在資料

表 `dump` 完後加上 `UNLOCK TABLE`。

`--extended-insert`。將多行的 `INSERT INTO` 濃縮成一行。

`--lock-tables`。在開始 `dump` 前，將資料表鎖住，最主要的用意是防止，`READ` 跟 `INSERT` 同時發生，造成資料的不一致。

看完 `--opt` 所代表的含意之後，就來檢視一下 `dump.file` 檔案的內容。發現到，原來 `mysqldump` 是將資料庫裡所有的內容資訊包括資料庫所擁有的資料表、資料表的結構、資料表內的所有記錄，都 `dump` 成 `SQL` 語法。以便日後直接用批次執行方試還原。就其內容稍做說明如下：

```
CREATE DATABASE /*!32312 IF NOT EXISTS*/ mis;
```

如果資料庫 `mis`，不存在的話，就建立一個新資料庫，這一行 `SQL` 語法並不是每一種 `mysqldump` 方式都會加進來。舉例如下：

```
mysqldump --opt -B mis > dump.file (會加入)
```

```
mysqldump --opt mis > dump.file (不會加入)
```

如果 `dump.file` 沒有加入上述那行 `SQL` 語法時，則必須自己建立 `mis` 資料庫，否則在“批次執行”的時候，會出現錯誤。

```
DROP TABLE IF EXISTS inventory;
```

這一行是由於下了 `--add-drop-table` 參數的關係。

```
LOCK TABLES inventory WRITE;
```

```
UNLOCK TABLES;
```

這兩行是加入 `--add-locks` 參數後的結果。

`CREATE TABLE`，則為建立原本資料表結構的語法。

```
INSERT INTO student VALUES ('5851001','張三','台北市中山南路','1979-01-12 00:00:00','F'),('5851002','李四','台北市復興北路','1980-12-24 00:00:00','M'),...
```

這一行，長長的 `INSERT INTO` 為加入 `--extended-insert` 後的結果，也就是之前說的，將每個 `insert into` 縮成一行。

從上面這些說明可以知道，`dump.file` 主要包含了資料表建立的 `SQL` 語法及新增資料表資料的語法。那進一步的來說，可以讓 `dump.file` 只保有資料表結構或資料表的記錄。做法分別如下：

保有資料表結構：

```
mysqldump --no-data mis > dump.file
```

保有資料表記錄：

```
mysqldump --no-create-info mis > dump.file
```

至於回復資料庫的方式相當的簡單，使用批次執行 `mysql` 指令的方式就可以完成了，方法如下：

```
mysql -u root < dump.file
```

4.10 建立資料庫和資料表和索引

4.10.1 建立資料庫

語法：CREATE DATABASE db_name

CREATE DATABASE 用給定的名字建立一個資料庫。如果資料庫已經存在，發生一個錯誤。在 MySQL 中的資料庫實現成包含對應資料庫中表的文件的目錄。因為資料庫在初始建立時沒有任何表，CREATE DATABASE 語句只是在 MySQL 數據目錄下面建立一個目錄。

使用 SHOW 語句找出在伺服器上當前存在什麼資料庫：

```
mysql> SHOW DATABASES;
```

```
+-----+
| Database |
+-----+
| mysql   |
| test    |
| tmp     |
+-----+
```

4.10.2 建立資料表

語法：CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
[(create_definition,...)]
[table_options] [select_statement]

建立資料庫是容易的部分，但是此時資料庫是空的，SHOW TABLES 將告訴你：

```
mysql> SHOW TABLES;
Empty set (0.00 sec)
```

使用一個 CREATE TABLE 語句建立你的資料表：

```
mysql> CREATE TABLE pet (name VARCHAR(20), owner
VARCHAR(20),
-> species VARCHAR(20), sex CHAR(1), birth DATE, death
```

DATE);

既然你建立了一個表，SHOW TABLES 應該產生一些輸出：

```
mysql> SHOW TABLES;
+-----+
| Tables in menagerie |
+-----+
| pet          |
+-----+
```

爲了驗證資料表是按你期望的方式被建立，使用一個 DESCRIBE 語句：

```
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    |      |
+-----+-----+-----+-----+-----+
--+-----+
```

你能隨時 DESCRIBE，例如，如果你忘記在你表中的列的名字或他們是什麼類型。

建立索引

語法：CREATE [UNIQUE] INDEX index_name ON tbl_name (col_name[(length)],...)

CREATE INDEX 語句在 MySQL 版本 3.22 以前不做任何事情。在 3.22 或以後版本中，CREATE INDEX 被映射到一個 ALTER TABLE 語句來建立索引。通常，你在用 CREATE TABLE 建立資料表本身時建立表的所有索引，CREATE INDEX 允許你把索引加到現有表中。

一個(`col1,col2,...`)形式的列表創造一個多列索引。索引值有給定列的值串聯而成。對於 **CHAR** 和 **VARCHAR** 列，索引可以只用一個列的部分來建立，使用 `col_name(length)`句法。(在 **BLOB** 和 **TEXT** 列上需要長度)。下面顯示的語句使用 `name` 列的頭 10 個字符建立一個索引：

```
mysql> CREATE INDEX part_of_name ON customer (name(10));
```

因為大多數名字通常在頭 10 個字元不同，這個索引應該不比從整個 `name` 列的建立的索引慢多少。另外，在索引使用部分的列能使這個索引成為更小的文件大部分，它能保存很多磁碟空格並且可能也加快 **INSERT** 操作！

4.12 保留字：

下列詞明確地在 **MySQL** 中被保留。他們的大多數被 **ANSI SQL92** 禁止作為列(`row`)或表(`table`)名(例如，`group`)。一些被保留因為 **MySQL** 需要他們並且正在(當前)使用：

<i>action</i>	<i>add</i>	<i>aggregate</i>	<i>all</i>
<i>alter</i>	<i>after</i>	<i>and</i>	<i>as</i>
<i>asc</i>	<i>avg</i>	<i>avg_row_length</i>	<i>auto_increment</i>
<i>between</i>	<i>bigint</i>	<i>bit</i>	<i>binary</i>
<i>blob</i>	<i>bool</i>	<i>both</i>	<i>by</i>
<i>cascade</i>	<i>case</i>	<i>char</i>	<i>character</i>
<i>change</i>	<i>check</i>	<i>checksum</i>	<i>column</i>
<i>columns</i>	<i>comment</i>	<i>constraint</i>	<i>create</i>
<i>cross</i>	<i>current_date</i>	<i>current_time</i>	<i>current_timestamp</i>
<i>data</i>	<i>database</i>	<i>databases</i>	<i>date</i>
<i>datetime</i>	<i>day</i>	<i>day_hour</i>	<i>day_minute</i>
<i>day_second</i>	<i>dayofmonth</i>	<i>dayofweek</i>	<i>dayofyear</i>
<i>dec</i>	<i>decimal</i>	<i>default</i>	<i>delayed</i>
<i>delay_key_write</i>	<i>delete</i>	<i>desc</i>	<i>describe</i>
<i>distinct</i>	<i>distinctrow</i>	<i>double</i>	<i>drop</i>
<i>end</i>	<i>else</i>	<i>escape</i>	<i>escaped</i>
<i>enclosed</i>	<i>enum</i>	<i>explain</i>	<i>exists</i>
<i>fields</i>	<i>file</i>	<i>first</i>	<i>float</i>
<i>float4</i>	<i>float8</i>	<i>flush</i>	<i>foreign</i>

<i>from</i>	<i>for</i>	<i>full</i>	<i>function</i>
<i>global</i>	<i>grant</i>	<i>grants</i>	<i>group</i>
<i>having</i>	<i>heap</i>	<i>high_priority</i>	<i>hour</i>
<i>hour_minute</i>	<i>hour_second</i>	<i>hosts</i>	<i>identified</i>
<i>ignore</i>	<i>in</i>	<i>index</i>	<i>infile</i>
<i>inner</i>	<i>insert</i>	<i>insert_id</i>	<i>int</i>
<i>integer</i>	<i>interval</i>	<i>int1</i>	<i>int2</i>
<i>int3</i>	<i>int4</i>	<i>int8</i>	<i>into</i>
<i>if</i>	<i>is</i>	<i>isam</i>	<i>join</i>
<i>key</i>	<i>keys</i>	<i>kill</i>	<i>last_insert_id</i>
<i>leading</i>	<i>left</i>	<i>length</i>	<i>like</i>
<i>lines</i>	<i>limit</i>	<i>load</i>	<i>local</i>
<i>lock</i>	<i>logs</i>	<i>long</i>	<i>longblob</i>
<i>longtext</i>	<i>low_priority</i>	<i>max</i>	<i>max_rows</i>
<i>match</i>	<i>mediumblob</i>	<i>mediumtext</i>	<i>mediumint</i>
<i>middleint</i>	<i>min_rows</i>	<i>minute</i>	<i>minute_second</i>
<i>modify</i>	<i>month</i>	<i>monthname</i>	<i>myisam</i>
<i>natural</i>	<i>numeric</i>	<i>no</i>	<i>not</i>
<i>null</i>	<i>on</i>	<i>optimize</i>	<i>option</i>
<i>optionally</i>	<i>or</i>	<i>order</i>	<i>outer</i>
<i>outfile</i>	<i>pack_keys</i>	<i>partial</i>	<i>password</i>
<i>precision</i>	<i>primary</i>	<i>procedure</i>	<i>process</i>
<i>processlist</i>	<i>privileges</i>	<i>read</i>	<i>real</i>
<i>references</i>	<i>reload</i>	<i>regexp</i>	<i>rename</i>
<i>replace</i>	<i>restrict</i>	<i>returns</i>	<i>revoke</i>
<i>rlike</i>	<i>row</i>	<i>rows</i>	<i>second</i>
<i>select</i>	<i>set</i>	<i>show</i>	<i>shutdown</i>
<i>smallint</i>	<i>soname</i>	<i>sql_big_tables</i>	<i>sql_big_selects</i>
<i>sql_low_priority_updates</i>	<i>sql_log_off</i>	<i>sql_log_update</i>	<i>sql_select_limit</i>
<i>sql_small_result</i>	<i>sql_big_result</i>	<i>sql_warnings</i>	<i>straight_join</i>
<i>starting</i>	<i>status</i>	<i>string</i>	<i>table</i>

<i>tables</i>	<i>temporary</i>	<i>terminated</i>	<i>text</i>
<i>then</i>	<i>time</i>	<i>timestamp</i>	<i>tinyblob</i>
<i>tinytext</i>	<i>tinyint</i>	<i>trailing</i>	<i>to</i>
<i>type</i>	<i>use</i>	<i>using</i>	<i>unique</i>
<i>unlock</i>	<i>unsigned</i>	<i>update</i>	<i>usage</i>
<i>values</i>	<i>varchar</i>	<i>variables</i>	<i>varying</i>
<i>varbinary</i>	<i>with</i>	<i>write</i>	<i>when</i>
<i>where</i>	<i>year</i>	<i>year_month</i>	<i>zerofill</i>

下列符號(來自上表)被 ANSI SQL 禁止但是被 MySQL 允許作為列(row)/表(table)名。這是因為這些名字的一些是很自然的名字並且很多人已經使用了他們。

- ACTION
- BIT
- DATE
- ENUM
- NO
- TEXT
- TIME
- TIMESTAMP

4.13 欄位型別：

MySQL 支援大量的欄位型別，它可以被分為 3 類：數字類型、日期和時間類型以及字串(字元)類型。本節介紹可用類型的概述，並且總結每個欄位型別的儲存需求。

由 MySQL 支援的列類型列在下面。下列代碼字母用於描述中：

M

指出最大的顯示尺寸。最大的合法的顯示尺寸是 255 。

D

適用於浮點類型並且指出跟隨在十進制小數點後的數碼的數量。最大可能的值是 30，但是應該不大於 M-2。

方括號(“[”和“]”)指出可選的類型修飾的部分。

注意，如果你指定一個了為 ZEROFILL，MySQL 將為該列自動地增加 UNSIGNED 屬性。

TINYINT[(M)] [UNSIGNED] [ZEROFILL]

一個很小的整數。有號的範圍是-128 到 127，無號的範圍是 0 到 255。

SMALLINT[(M)] [UNSIGNED] [ZEROFILL]

一個小整數。有號的範圍是-32768 到 32767，無號的範圍是 0 到 65535。

MEDIUMINT[(M)] [UNSIGNED] [ZEROFILL]

一個中等大小整數。有號的範圍是-8388608 到 8388607，無號的範圍是 0 到 16777215。

INT[(M)] [UNSIGNED] [ZEROFILL]

一個正常大小整數。有號的範圍是-2147483648 到 2147483647，無號的範圍是 0 到 4294967295。

INTEGER[(M)] [UNSIGNED] [ZEROFILL]

這是 INT 的一個同義詞。

BIGINT[(M)] [UNSIGNED] [ZEROFILL]

一個大整數。有號的範圍是 -9223372036854775808 到 9223372036854775807，無號的範圍是 0 到 18446744073709551615。注意，所有算術運算用有符號的 **BIGINT** 或 **DOUBLE** 值完成，因此你不應該使用大於 9223372036854775807 (63 位) 的有符號大整數，除了位函數！注意，當兩個參數是 **INTEGER** 值時，-、+和*將使用 **BIGINT** 運算！這意味著如果你乘 2 個大整數(或來自於回傳整數的函數)，如果結果大於 9223372036854775807，你可以得到意外的結果。一個浮點數字，不能是無符號的，對一個單精度浮點數，其精度可以是 ≤ 24 ，對一個雙精度浮點數，是在 25 和 53 之間，這些類型如 **FLOAT** 和 **DOUBLE** 類型馬上在下面描述。**FLOAT(X)** 有對應的 **FLOAT** 和 **DOUBLE** 相同的範圍，但是顯示尺寸和小數位數是未定義的。在 MySQL 3.23 中，這是一個真正的浮點值。在更早的 MySQL 版本中，**FLOAT(precision)** 總是有 2 位小數。該方法爲了 ODBC 兼容性而提供。

FLOAT[(M,D)] [ZEROFILL]

一個小(單精密)浮點數字。不能無符號。允許的值是-3.402823466E+38 到-1.175494351E-38，0 和 1.175494351E-38 到 3.402823466E+38。M 是顯示寬度而 D 是小數的位數。沒有參數的 **FLOAT** 或有 <24 的一個參數表示一個單精密浮點數字。

DOUBLE[(M,D)] [ZEROFILL]

一個正常大小(雙精密)浮點數字。不能無符號。允許的值是-1.7976931348623157E+308 到-2.2250738585072014E-308、0 和 2.2250738585072014E-308 到 1.7976931348623157E+308。M 是顯示寬度而 D 是小數位數。沒有一個參數的 **DOUBLE** 或 **FLOAT(X)** ($25 \leq X \leq 53$) 代表一個雙精密浮點數字。

DOUBLE PRECISION[(M,D)] [ZEROFILL]

REAL[(M,D)] [ZEROFILL]

這些是 **DOUBLE** 同義詞。

DECIMAL[(M[,D])] [ZEROFILL]

一個未壓縮(**unpack**)的浮點數字。不能無符號。行為如同一個 **CHAR** 列：“未壓縮”意味著數字作為一個字符串被儲存，值的每一位使用一個字符。小數點，並且對於負數，“-”符號不在 **M** 中計算。如果 **D** 是 **0**，值將沒有小數點或小數部分。**DECIMAL** 值的最大範圍與 **DOUBLE** 相同，但是對一個給定的 **DECIMAL** 列，實際的範圍可以通過 **M** 和 **D** 的選擇被限制。如果 **D** 被省略，它被設置為 **0**。如果 **M** 被省掉，它被設置為 **10**。注意，在 **MySQL3.22** 裡，**M** 參數包括符號和小數點。

NUMERIC(M,D) [ZEROFILL]

這是 **DECIMAL** 的一個同義詞。

DATE

一個日期。支援的範圍是 '1000-01-01' 到 '9999-12-31'。MySQL 以 'YYYY-MM-DD' 格式來顯示 **DATE** 值，但是允許你使用字符串或數字把值賦給 **DATE** 列。

DATETIME

一個日期和時間組合。支援的範圍是 '1000-01-01 00:00:00' 到 '9999-12-31 23:59:59'。MySQL 以 'YYYY-MM-DD HH:MM:SS' 格式來顯示 **DATETIME** 值，但是允許你使用字符串或數字把值賦給 **DATETIME** 的列。

TIMESTAMP[(M)]

一個時間戳記。範圍是 '1970-01-01 00:00:00' 到 2037 年的某時。MySQL 以 **YYYYMMDDHHMMSS**、**YYMMDDHHMMSS**、**YYYYMMDD** 或 **YYMMDD** 格式來顯示 **TIMESTAMP** 值，取決於是否 **M** 是 **14**（或省略）、**12**、**8** 或 **6**，但是允許你使用字符串或數字把值賦給 **TIMESTAMP** 列。一個 **TIMESTAMP** 列對於記錄一個 **INSERT** 或 **UPDATE** 操作的日期和時間是有用的，因為如果你不自己給它賦值，它自動地被設置為最近操作的日期和時間。你可以通過賦給它一個 **NULL** 值設置它為當前的日期和時間。

TIME

一個時間。範圍是 '-838:59:59' 到 '838:59:59'。MySQL 以 'HH:MM:SS' 格式來顯示 **TIME** 值，但是允許你使用字符串或數字把值賦給 **TIME** 列。

YEAR[(2|4)]

一個 2 或 4 位數字格式的年（內定是 4 位）。允許的值是 1901 到 2155，和 0000（4 位年格式），如果你使用 2 位，1970-2069（70-69）。MySQL 以 **YYYY** 格式來顯示 **YEAR** 值，但是允許你把使用字符串或數字值賦給 **YEAR** 列。（**YEAR** 類型在 **MySQL3.22** 中是新類型。）

CHAR(M) [BINARY]

一個定長字符串，當儲存時，總是是用空格填滿右邊到指定的長度。**M** 的範圍是 1 ~ 255 個字符。當值被檢索時，空格尾部被刪除。**CHAR** 值根據內定字符集以大小寫不區分的方式排序和比較，除非給出 **BINARY** 關鍵詞。**NATIONAL CHAR**（短形式 **NCHAR**）是 **ANSI SQL** 的方式來定義 **CHAR** 列應該使用內定字符集。這是 **MySQL** 的內定。**CHAR** 是 **CHARACTER** 的一個縮寫。

[NATIONAL] VARCHAR(M) [BINARY]

一個變長字符串。注意：當值被儲存時，尾部的空格被刪除（這不同於 ANSI SQL 規範）。M 的範圍是 1 ~ 255 個字符。VARCHAR 值根據內定字符集以大小寫不區分的方式排序和比較，除非給出 BINARY 關鍵詞值。VARCHAR 是 CHARACTER VARYING 一個縮寫。

TINYTEXT

一個 BLOB 或 TEXT 列，最大長度為 255(2⁸-1)個字元。

TEXT

一個 BLOB 或 TEXT 列，最大長度為 65535(2¹⁶-1)個字元。

MEDIUMTEXT

一個 BLOB 或 TEXT 列，最大長度為 16777215(2²⁴-1)個字元。

LONGTEXT

一個 BLOB 或 TEXT 列，最大長度為 4294967295(2³²-1)個字元。

ENUM('value1', 'value2', ...)

枚舉。一個僅有一個值的字符串對像，這個值式選自與值列表 'value1'、'value2'，...，或 NULL。一個 ENUM 最多能有 65535 不同的值。

SET('value1', 'value2', ...)

一個集合。能有零個或多個值的一個字符串對像，其中每一個必須從值列表 'value1'， 'value2'， ... 選出。一個 SET 最多能有 64 個成員。

4.14 SQL 語法

4.14.1 DCL(Data Control Language)

mysqladmin 是在 MySQL 下管理資料庫的主要工具。這工具程式可讓你建立、銷毀、和監控伺服器與它所支援的資料庫。MySQL 在 mysqladmin 工具程式中提供相當齊全的命令集，可以監控 MySQL 伺服器。用 mysqladmin 有一些命令選項，其通用格式為：

```
mysqladmin OPTIONS COMMAND1 COMMAND2 .....COMMANDn
```

以下列出幾個屬於 DCL 的指令：

drop DATABASENAME

丟棄名為 DATABASENAME 的資料庫。

Extended-status

提供伺服器的詳細狀態資訊。

Flush-hosts

出清所有快取區內的主機。

Flush-logs

出清所有日誌檔。

Flush-tables

出清所有表格。

Kill ID1, ID2, ..., IDn

終止所指定的執行緒。

Password NEWPASSWORD

將密碼變更為 NEWPASSWORD。

Ping

檢查 mysql 是否仍在運作。

Processlist

列出仍在活動中的執行緒清單。

Reload

重新載入所有授權表(**grant table**)

refresh

出清所有表格、關閉所有日誌檔、再重新開啓所有日誌檔。

Shutdown

關閉伺服器。

Status

提供簡短的伺服器狀態訊息。

Variables

顯示伺服器的版本資訊。

除了上述命令外，**mysqladmin** 還支援以下選項：

#LOG

輸出除錯日誌檔。

-f

不經確認就直接刪除表格，而且不管動作是否成功，都繼續執行下一道命令。

-? 或 -help

顯示 **mysqladmin** 的說明。

-C

在主行式協定使用壓縮。

-h HOST

連接至所指定的主機。

-p [PASSWORD]

使用這密碼來驗證使用者的身份。如果使用這選項時沒有指定 **PASSWORD**，則 **mysqladmin** 會提示使用者輸入一個新密碼。

-P PORT

以指定的 **port** 來連線。

-I SECONDS

以指定的間隔秒數反覆執行命令。

-S

若無法與伺服器建立連線，就安靜的結束。

-S SOCKET

當成 UNIX socket 的檔案。

-t TIMEOUT

連線的逾時時間。

-u USER

若不是目前的使用者，則以指定的 **USER** 登入。

-V

顯示版本資訊並退出。

-w COUNT

若伺服器尚未改動，則等候並重試 **COUNT** 次。

4.14.2 DDL(Data Design Language)

ADD [COLUMN] create_clause

對現存的表格中加入一個新欄。**create_clause** 的語法與 **CREATE** 敘述相同。新欄的名稱不能與表格舊有的任何欄名相同。**COLUMN** 關鍵字是可有可無的，它沒有任何作用。

ADD INDEX [name]{column, ...}

為 **column** 建立索引，一個索引最多可以包含 15 個欄。索引的 **name** 是可有可無的，若不指定，會以第一欄的名稱來命名。

ADD UNIQUE[name] {column, ...}

這與 **ADD INDEX** 幾乎完全相同，差別在於被索引的欄值保證一定不會互相重複，也就是說，若使用者嘗試新增一個值，而此值已經存在於索引中，此時將會出現錯誤。

DROP [COLUMN] column 從表格中刪除指定的 **column**。被刪除的欄與其資料將永遠從表格中消失，除非你預先備份資料，否則無法挽回被刪掉的資料。此外，在索引中參考到 **column** 的指標也會被刪除。

DROP INDEX key

從表格中移除一個索引，但是不會刪除或變更表格本身的資料，只有索引資料會被刪除，因此，你可以使用 **ALTER TABLE ... AND INDEX** 敘述來重建以這種方式刪除的索引。

DROP PRIMARY KEY

用法與用途與 **DROP INDEX** 完全相同，差別在於 **DROP PRIMARY KEY** 會去尋找主鍵(Primary Key)這種特殊的索引來刪除，如果表格中沒有主鍵，則第一個 **unique key** 會被刪除。

DROP INDEX key

從表格中移除一個索引，但是不會刪除或變更表格本身的資料，只有索引資料會被刪除，因此，你可以使用 **ALTER TABLE ... AND INDEX** 敘述來重建以這種方式刪除的索引。

DROP PRIMARY KEY

用法與用途與 **DROP INDEX** 相同，差別在於 **DROP PRIMARY KEY** 會去尋找主鍵(Primary Key)這種特殊的索引來刪除，如果表格中沒有主鍵，則第一個 **unique key** 會被刪除。

```
CREATE DATABASE dbname
CREATE Table name (field_name field_type, ...)
CREATE INDEX name ON table(column, ...)
CREATEFUNCTION name RETURNS values SONAME library
```

建立新的資料庫元件(或全新資料庫)。**CREATE** 敘述用來建立資料庫、表格、或使用者定義的函式(User Defined Functions , UDF)。

4.14.3 DML(Data Manipulation Language)

```
ALTER [COLUMN] column SET DEFAULT value
ALTER [COLUMN] column DROP DEFAULT
```

建立、修改、或刪除 **column** 的內定值。當使用 **SET DEFAULT** 時，**column** 的內定值(不管原來是否有內定值存在)將被設定成 **value**。當使用 **DROP DEFAULT** 時，任何既有的內定值將被刪除。在刪除內定值之後，原本以內定值建立的列也不會受到影響。**CLOUMN** 關鍵字是可有可無的，它沒有任何作用。

```
SELECT [STRAIGHT_JOIN] [SQL_SMALL_RESULT] [SQL_BIG_RESULT]
[HIGH_PRIORITY] [DISTINCT | DISTINCTROW | ALL]
select_expression, ...
[INTO {OUTFILE | DUMPFILE} 'file_name' export_options]
[FROM table_references
    [WHERE where_definition]
    [GROUP BY col_name, ...]
    [HAVING where_definition]
    [ORDER BY {unsigned_integer | col_name | formula}
[ASC | DESC] , ...][LIMIT [offset,] rows][PROCEDURE
procedure_name] ]
```

SELECT 被用來檢索從一個或多個表中選取的行。**select_expression** 指出你想要檢索的列。**SELECT** 也可以用來檢索不引用任何表的計算行。例如：

```
mysql> SELECT 1 + 1;
-> 2
```

所有使用的關鍵詞必須精確地以上面的順序給出。例如，一個 **HAVING** 子句必須跟在 **GROUP BY** 子句之後和 **ORDER BY** 子句之前。

一個 **SELECT** 表達式可以用一個 **AS** 給定一個別名，別名被用作表達式的列名並且能使用在 **ORDER BY** 或 **HAVING** 子句中。例如：

```
mysql> select concat(last_name, ', ', ',first_name) AS
```

full_name

```
from mytable ORDER BY full_name;
```

FROM table_references 子句指出從哪個表中檢索行。如果你命名多個表，你正在執行一個聯結(join)。對於聯結的句法資訊。

你可以引用一個列為 col_name、tbl_name.col_name 或 db_name.tbl_name.col_name，你不必在一個 SELECT 語句中指定一個 tbl_name 或 db_name.tbl_name 是一個列引用的前綴，除非引用有二義性。對於二義性的例子要求更加顯式的列引用格式。

一個表引用可以使用 tbl_name [AS] alias_name 起一個別名。

```
mysql> select t1.name, t2.salary from employee AS t1, info AS t2
```

```
where t1.name = t2.name;
```

```
mysql> select t1.name, t2.salary from employee t1, info t2
where t1.name = t2.name;
```

選取輸出的列可以用列名、列別名或列位置在 ORDER BY 和 GROUP BY 子句引用，列位置從 1 開始。

```
mysql> select college, region, seed from tournament
ORDER BY region, seed;
```

```
mysql> select college, region AS r, seed AS s from tournament
ORDER BY r, s;
```

```
mysql> select college, region, seed from tournament
ORDER BY 2, 3;
```

為了以降序排列，把 DESC (下降) 關鍵詞加到 ORDER BY 子句中你要排序的列名前。內定是升序；這也可以用 ASC 關鍵詞明確指定。

```
INSERT [LOW_PRIORITY | DELAYED] [IGNORE]
[INTO] tbl_name [(col_name,...)]
VALUES (expression,...),(...),...
```

或 INSERT [LOW_PRIORITY | DELAYED] [IGNORE]
[INTO] tbl_name [(col_name,...)]
SELECT ...

或 INSERT [LOW_PRIORITY | DELAYED] [IGNORE]
[INTO] tbl_name
SET col_name=expression, col_name=expression, ...

INSERT 把新行插入到一個存在的表中，INSERT ... VALUES 形式的語句基於明確指定的值插入行，INSERT ... SELECT 形式插入從其他表選擇的行，有多個值表的 INSERT ... VALUES 的形式在 MySQL 3.22.5 或以後版本中支援，col_name=expression 語法在 MySQL 3.22.10 或以後版本中支援。

4.15 函數

<code>mysql_affected_rows()</code>	返回被最新的 UPDATE, DELETE 或 INSERT 查詢影響的行數。
<code>mysql_close()</code>	關閉一個伺服器連接。
<code>mysql_connect()</code>	連接一個 MySQL 伺服器。該函數不推薦；使用 <code>mysql_real_connect()</code> 代替。
<code>mysql_change_user()</code>	改變在一個打開的連接上的用戶和資料庫。
<code>mysql_create_db()</code>	創建一個資料庫。該函數不推薦；而使用 SQL 命令 CREATE DATABASE。
<code>mysql_data_seek()</code>	在一個查詢結果集中搜尋一任意行。
<code>mysql_debug()</code>	用給定字符串做一個 DEBUG_PUSH。
<code>mysql_drop_db()</code>	拋棄一個資料庫。該函數不推薦；而使用 SQL 命令 DROP DATABASE。
<code>mysql_dump_debug_info()</code>	讓伺服器將調試資訊寫入日誌文件。
<code>mysql_eof()</code>	確定是否已經讀到一個結果集合的最後一行。這功能被反對； <code>mysql_errno()</code> 或 <code>mysql_error()</code> 可以相反被使用。
<code>mysql_errno()</code>	返回最近被調用的 MySQL 函數的出錯編號。
<code>mysql_error()</code>	返回最近被調用的 MySQL 函數的出錯消息。
<code>mysql_escape_string()</code>	用在 SQL 語句中的字符串的轉義特殊字符。
<code>mysql_fetch_field()</code>	返回下一個表字段的類型。
<code>mysql_fetch_field_direct()</code>	返回一個表字段的類型，給出一個字段編號。
<code>mysql_fetch_fields()</code>	返回一個所有字段結構的數組。
<code>mysql_fetch_lengths()</code>	返回當前行中所有列的長度。
<code>mysql_fetch_row()</code>	從結果集中取得下一行。
<code>mysql_field_seek()</code>	把列光標放在一個指定的列上。
<code>mysql_field_count()</code>	返回最近查詢的結果列的數量。
<code>mysql_field_tell()</code>	返回用於最後一個 <code>mysql_fetch_field()</code> 的字段光標的位置。
<code>mysql_free_result()</code>	釋放一個結果集合使用的內存。
<code>mysql_get_client_info()</code>	返回客戶版本資訊。
<code>mysql_get_host_info()</code>	返回一個描述連接的字符串。
<code>mysql_get_proto_info()</code>	返回連接使用的協議版本。
<code>mysql_get_server_info()</code>	返回伺服器版本號。
<code>mysql_info()</code>	返回關於最近執行得查詢的資訊。

<code>mysql_init()</code>	獲得或初始化一個 MySQL 結構。
<code>mysql_insert_id()</code>	返回有前一個查詢為一個 <code>AUTO_INCREMENT</code> 列產生的 ID。
<code>mysql_kill()</code>	殺死一個給定的執行緒。
<code>mysql_list_dbs()</code>	返回匹配一個簡單的正則表達式的資料庫名。
<code>mysql_list_fields()</code>	返回匹配一個簡單的正則表達式的列名。
<code>mysql_list_processes()</code>	返回當前伺服器執行緒的一張表。
<code>mysql_list_tables()</code>	返回匹配一個簡單的正則表達式的表名。
<code>mysql_num_fields()</code>	返回一個結果集合重的列的數量。
<code>mysql_num_rows()</code>	返回一個結果集合中的行的數量。
<code>mysql_options()</code>	設置對 <code>mysql_connect()</code> 的連接選項。
<code>mysql_ping()</code>	檢查對伺服器的連接是否正在工作，必要時重新連接。
<code>mysql_query()</code>	執行指定為一個空結尾的字符串的 SQL 查詢。
<code>mysql_real_connect()</code>	連接一個 MySQL 伺服器。
<code>mysql_real_query()</code>	執行指定為帶計數的字符串的 SQL 查詢。
<code>mysql_reload()</code>	告訴伺服器重裝授權表。
<code>mysql_row_seek()</code>	搜索在結果集合中的行，使用從 <code>mysql_row_tell()</code> 返回的值。
<code>mysql_row_tell()</code>	返回行光標位置。
<code>mysql_select_db()</code>	連接一個資料庫。
<code>mysql_shutdown()</code>	關掉資料庫伺服器。
<code>mysql_stat()</code>	返回作為字符串的伺服器狀態。
<code>mysql_store_result()</code>	檢索一個完整的結果集合給客戶。
<code>mysql_thread_id()</code>	返回當前執行緒的 ID。
<code>mysql_use_result()</code>	初始化一個一行一行地結果集合的檢索。

第五章 計數器

5.1 計數器簡介

計數器是網頁中不可少的配件之一。

目的是要了解訪客確實人數，通常訪客計數器是讓 web 訪客知道該網頁或者網站的人氣指數最直接的方法。

以下我們著手設計一個資料庫形式的計數器。

5.2 資料庫說明

1. 只紀錄到訪次數的 table :

```
create table counter ( visited INT unsigned );
```

欄位	型態	Null	預設值
visited	int(10)	UNSIGNED	是
lastaccu	int(10)	UNSIGNED	是

```
alter table counter add column
lastaccu INT unsigned after visited;
```

2. 紀錄每天訪客數用的 table :

```
create table daycount(
yearid tinyINT unsigned not null,
dayid smallINT unsigned,
daycount smallINT unsigned,
index yid (yearid)
);
```

欄位	型態	屬性	Null	預設值
yearid	tinyint(3)	UNSIGNED	否	0
dayid	smallint(5)	UNSIGNED	是	NULL
daycount	smallint(5)	UNSIGNED	是	NULL

```
insert into
daycount(yearid,dayid,daycount)
select
date_format(now(),'%y'),dayofyear(
now()),
visited
from
counter;
```

```
update counter set lastaccu = visited;
```

5.3 程式碼說明

先做好 0 到 9 十個 gif 檔再用 PHP 的 image function 產生圖形

//5-1 訪客數字的顯示

<?

// Counter Image Generation

// 首先傳 Header 資料

Header("Content-type: image/gif");

// 定義圖形

\$img = ImageCreate(60,20);

// 連線到資料庫

\$linkptr = mysql_pconnect("localhost","root","");

mysql_select_db("dbname",\$linkptr);

// 訪客數加一並讀出

mysql_query("update counter set visited=visited+1",\$linkptr);

\$result = mysql_query("select visited from counter",\$linkptr);

list (\$counter) = mysql_fetch_row(\$result);

\$counter = sprintf("%05d",\$counter); // 補到五位數

// 辨識數字並讀入相對圖形貼上

for (\$i=0;\$i<5;\$i++) {

 \$tmpstr = "../bookfigures/" . substr(\$counter,\$i,1) . ".gif";

 \$tmpimg = ImageCreateFromGif(\$tmpstr);

 ImageCopyResized(\$img,\$tmpimg,\$i*12,0,0,12,20,12,20);

 ImageDestroy(\$tmpimg);}

// 輸出成 GIF 等格式

ImageGIF(\$img);

// 釋放記憶體

ImageDestroy(\$img);

?>

//5-2 爲方便統計,我們還提供一個工具,利用計數器紀錄的資料,做進一步分析,畫出一隨日期變化的到訪人數曲線圖

<?

```
// Show visited number curve
```

```
// 首先傳 Header 資料
```

```
Header("Content-type: image/gif" );
```

```
// 定義圖形，先做好一張 400 x 240
```

```
$img = ImageCreateFromGif("../bookfigures/visitedcurve.gif" );
```

```
// 連線到資料庫
```

```
$linkptr = mysql_pconnect("localhost","root","");
```

```
mysql_select_db("dbname",$linkptr);
```

```
// 讀出當年份資料並存入 array
```

```
$querystring = "select dayid,daycount from daycount
```

```
    where yearid = date_format(now(),'%y')
```

```
    order by dayid asc" ;
```

```
$result = mysql_query($querystring,$linkptr);
```

```
while ( list($dayid,$daycount) = mysql_fetch_row($result) ) {
```

```
    $dayvisited[$dayid] = $daycount ; }
```

```
$totalday = mysql_num_rows($result);
```

```
mysql_free_result($result);
```

```
// 找出最大值
```

```
$querystring = "select max(daycount) from daycount where yearid = date_format(now(),'%y')" ;
```

```
$result = mysql_query($querystring,$linkptr);
```

```
list( $maxcount ) = mysql_fetch_row($result);
```

```
// 定義顏色
```

```
$black = ImageColorAllocate($img,0,0,0);
```

```
$red = ImageColorAllocate($img,255,0,0);
```

```
// 畫 y 軸刻度
```

```
for ($i=0;$i<5;$i++) {
```

```
    imagestring($img,5,0,44*$i,intval($maxcount*1.1*(5-$i)/5),$black); }
```

```
// 每天畫一個矩形
```

```
$ratio = 220 / $maxcount / 1.1 ;
```

```

for ($i=0;$i<$totalday;$i++) {
    $tmpx = $i + 34 ;
    $tmpy = 220 - $dayvisited[$i] * $ratio ;
    ImageFilledRectangle($img,$tmpx-1,$tmpy-1,$tmpx+1,$tmpy+1,$red) ; }

// 輸出成 GIF 等格式
ImageGIF($img) ;

// 釋放記憶體
ImageDestroy($img) ;
?>

<html>
<head>
    <title>5-3</title>
</head>

<body bgcolor="#FFFFFF">
<table width="100%"><tr>
<td valign="TOP" align="CENTER">
<p>
<!-- 計數器 -->
拜訪人數</a><br>
</tr></table>
</body>
</html>

```

5.4 計數器之範例

拜訪人數：

5.5 會員密碼認證簡介

本章主要是要來介紹 如何來建立一個 PHP 安全登入的認證

為什麼要設登入認證的動作？

1. 網頁限制：因為藉著使用者的登入的動作，可以使得網頁的資料，只有經過認證過的使用者才可以使用。
2. 權限控管：另外，透過登入的機制，可以方便管理者作權限管理的動作。例如可以作到權限分級的功能，依登入者的身份不同而給予對網站不同的權限。

5.6 密碼認證的方法

1. 利用密碼搭配使用者名稱、身份證字號、e-mail 等等資料驗證。
2. 採用認證金鑰之類的方法加密

密碼認證的方式大致分爲

1. 輸入認證資料介面
2. 密碼存放
3. 多網頁間認證資格的傳遞

5.7 輸入認證資料介面

輸入認證資料的介面大致上可以分爲二種方式，一種是 **WWW Authentication** 的方式，另一種是利用 **form** 搭配資料庫的方式。當然也可以使用 **WWW Authentication** 搭配資料庫的方式，本章將以介紹前面所述的這二種方式爲主。

5.7.1 WWW Authentication 方式

這種方式的認證，是透過 **pop up** 出一個視窗來給使用者登入的介面。不過值得注意的是這種方式，一般只用來輸入 **Username/Password**。

```
<?
  Header("WWW-authenticate: basic realm=\"保密範圍\"");
  Header("HTTP/1.0 401 Unauthorized");
?>
```



5.7.2 利用 form 搭配資料庫的方式

由於利用 **WWW Authentication** 的方式，將被侷限於只能輸入 **Username / Password**，因此如果需要多一個身份證字號的欄位的話，則要用另一種方式即利用 **form** 的方式。

```
<html>
請輸入帳號:
<input type "text" size="10" name="user">
<br>
請輸入密碼:
<input type ="password" size="10" name="pass">
<input type="submit" name="submit" value="登入">
</html>
```

5.8 *WWW Authentication* 及 *Form* 認證方式

5.8.1 *WWW Authentication*

這種方式的認證，是透過 **pop up** 出一個視窗來給使用者登入的介面，利用 **PHP** 的 **header()** 函數搭配 **\$PHP_AUTH_USER** 和 **\$PHP_AUTH_PW** 這兩個全域變數 (**global variables**) 來製作出基本的使用者認證機制。

下面是正常的伺服器端密碼保護機制的運作方式：

1. 使用者想要瀏覽伺服器上面的某個檔案，如果這個檔案受到密碼保護，那麼伺服器便會回應一個 401 (未認證的使用者) 表頭給瀏覽器。
2. 瀏覽器接收到這個表頭資訊，便會跳出一個對話方塊，要求使用者輸入使用者名稱與密碼。

3. 使用者在對話方塊中輸入使用者名稱與密碼，然後按下 OK 按鈕，瀏覽器便會將帳號密碼傳送給伺服器以供認證之用。
4. 如果使用者輸入的帳號密碼是有效的，那麼伺服器便會將該使用者之前要求瀏覽的檔案傳送給使用者。反之如果帳號密碼不正確，伺服器會再度傳送 401 HTTP 表頭給瀏覽器，瀏覽器接收到以後又會再度跳出請使用者輸入帳號密碼的對話方塊，重新按照上面的步驟進行認證，直到帳號密碼正確，或者使用者在對話方塊中按下取消按鈕為止。當使用者通過認證以後，這個認證身分在同一個受保護的區域中都會保持有效。

程式碼如下頁（程式檔名：`www_Auth_login.php`）：

範例說明：

透過 PHP 的 `header()` 函數輸出 HTTP 表頭來模擬上面提到的 HTTP 使用者認證機制。PHP 會自動把使用者在對話方塊中所輸入的使用者名稱與密碼分別儲存在 `$PHP_AUTH_USER` 與 `$PHP_AUTH_PW` 這兩個全域變數裡面。透過這兩個變數，我們就可以檢查使用者輸入的帳號密碼是否有效。有效的帳號密碼可以儲存在文字檔或資料庫裡面，或者其他任何我們覺得適當的地方都可以。

這個範例程式，是將有效的使用者帳號與密碼寫死在程式裡面的。當管理者只有一個的時候或只對少數人作控管時，這樣的方式是比較有效率的。相反的，如果面對較多的使用者，則還是需要作會員管理及資料庫的設計。

(程式檔名：www_Auth_login.php)

```
<?
$user="username";      //利用變數存放使用者資料
$pass="userpassword";

/* 檢查$PHP_AUTH_USER 與$PHP_AUTH_PW 這兩個變數的值 */
if (!isset($PHP_AUTH_USER) || !isset($PHP_AUTH_PW)) :
/* 沒有值：表示使用者尚未通過驗證，送出 HTTP 表頭資訊給瀏覽器，讓瀏覽器
顯示認證對話方塊 */
    Header("WWW-authenticate: basic realm=\"保密範圍\"");
    Header("HTTP/1.0 401 Unauthorized");
    echo "抱歉!您沒有瀏覽這個網頁的權限!";
    exit();
elseif(($PHP_AUTH_USER==$user) && ($PHP_AUTH_PW==$pass)) :
    echo "您已經通過認證了!";
else :
/* 如果帳號或密碼其中之一是無效的，就再度傳送 HTTP 表頭資訊給瀏覽器，讓
瀏覽器顯示認證對話方塊 */
    Header("WWW-authenticate: basic realm=\"保密範圍\"");
    Header("HTTP/1.0 401 Unauthorized");
    echo "您必須通過使用者認證。";
    exit();
endif; ?>
```

注意：當你使用以檔案為基礎的保護措施的時候，並非該目錄下的所有檔案都會自動受到保護。但是如果你以為在瀏覽器端跳出認證對話方塊就等於保護某一目錄下的所有檔案的話，那麼的確應該好好想想你的思考模式了。

5.8.2 利用 form 搭配資料庫

利用 **form** 搭配資料庫的方式，大致上需要經過下列基本的程序。

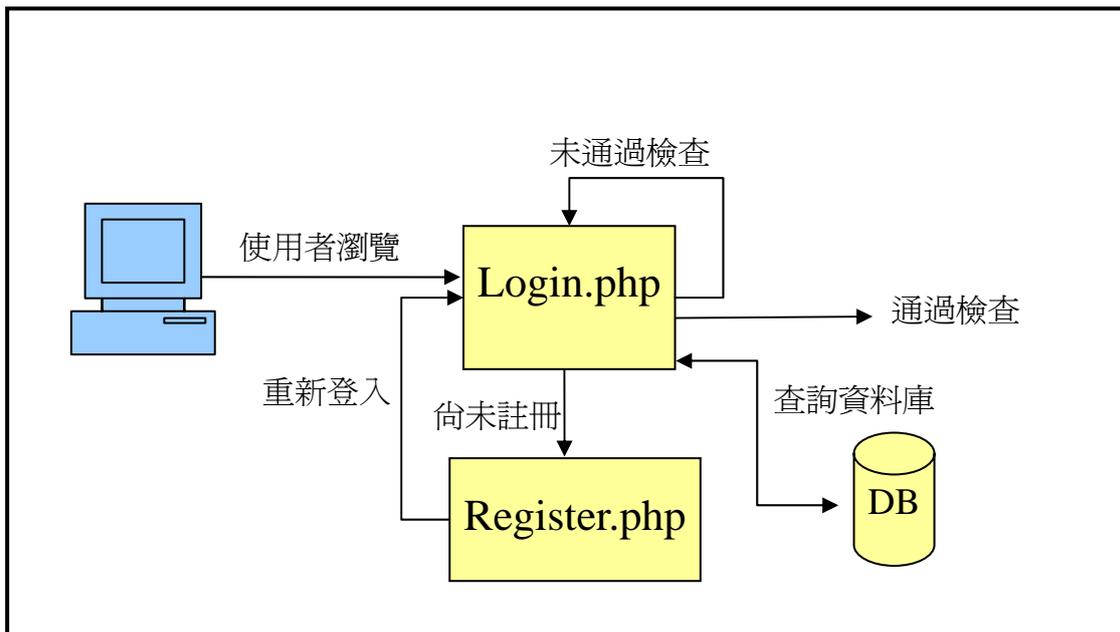
第一個步驟：建立資料庫

第二個步驟：設計會員註冊表單

第三個步驟：設計會員登入表單

第四個步驟：登入帳號與密碼檢查

整個認證運作流程如下所示：



那我們就先來將資料庫建立起來吧！

第一個步驟：建立資料庫

首先以 **MySQL superuser** 建立一個資料庫 **memberdb**：

```
SHEEL>mysqladmin -u root -p create memberdb
```

並給定權限

```
SHEEL >mysql -u root -p mysql
mysql > grant all privileges on memberdb.*
        to test@localhost identified by 'test';
mysql> quit
```

接下來規劃一個 **table**，用來儲存會員資料，現在我先暫時考慮姓名、性別、使用者名稱、密碼就好了，另外再加上一個獨立的 **id** 當作索引：

```
create table members(  
    id int not null auto_increment,  
    name char(10),  
    sex enum('0','1'),  
    loginusr char(10),  
    loginpsd char(10),  
    primary key(id) );
```

設定 **id** 為自動累加，當作一個 **key**。下一步我們將設計一個簡單的會員註冊畫面，然後把資料寫到資料庫裡。

第二個步驟：設計會員註冊表單

會員註冊這個部份可以參考基本資料維護檔，這裡將以簡單的會員註冊程式，以使得本案例可以更完整的呈現。其主要過程，不過是將 **form** 所讀進來的資料，加上流程控制，並將其寫入資料庫。

(程式檔名：**register.php**)

```
<html><head> <title>會員註冊</title></head>  
<body>  
<?  
    if (empty($_HTTP_POST_VARS)):  
?>  
<h3>會員註冊</h3>  
<form action="register.php" method="post">  
<table>  
    <tr> <td align="right">您的姓名</td>  
    <td> <input type="text" name="username">  
    </td></tr>  
    <tr> <td align="right">性別 </td>  
    <td> <select name="sex">  
        <option value="0">女生  
        <option value="1">男生  
    </select> </td></tr>  
    <tr> <td align="right">登入名稱</td>  
    <td> <input type="text" name="logname">
```

```

        </td></tr>
        <tr> <td align="right">登入密碼</td>
        <td> <input type="text" name="logpass">
        </td></tr>
        <tr> <td align="center" colspan="2">
        <input type="submit" name="submit" value="註冊">
        <td> </tr>
</table>
</form>

<? else:
//先讀出變數
$username= $HTTP_POST_VARS["username"];
$sex= $HTTP_POST_VARS["sex"];
$logname= $HTTP_POST_VARS["logname"];
$logpass= $HTTP_POST_VARS["logpass"];
//建立資料庫的連線
$link = @mysql_pconnect("localhost","test","test")
        or die("連線到資料庫失敗!");
mysql_select_db("memberdb",$link);

//寫出 query string
$strSQL="insert into members
        (name,sex,loginusr,loginpsd) values
        ('$username','$sex','$logname','$logpass')";
//執行 query
mysql_query($strSQL);

//假如沒有該筆資料，則印出"新增失敗"，然後結束程式
if(mysql_affected_rows()!=1)
    die("新增失敗!");
echo "謝謝您的註冊,請用 $logname / $logpass 登入!";
echo '<a href="login.php">進入登入頁面 </a>';
endif;
?>
</body></html>

```

第三個步驟：設計會員登入表單及登入的帳號與密碼檢查

現在使用者的登入資料已經註冊好在資料庫中了，因此接下來就是提供使用者一個登入的畫面。下面這隻程式主要是提供使用者登入的介面以及判斷是否正確的通過認證

(程式檔名：`login.php`)

```
//執行 query
$result= mysql_query($strSQL);

//假如沒有該筆資料，則印出"密碼錯誤!請重新登入"，然後結束程式
if (mysql_num_rows($result)!=1) :
die("密碼錯誤!請重新登入");
else:
echo "您已經通過認證了!";
endif;
?>
```

範例說明：

檢查使用者帳號與密碼的機制主要完成於 `if` 判斷式中，判斷 `loginuser` 是否有傳值進來 `if(empty($_HTTP_POST_VARS["loginuser"]))` 如果有值，則連結資料庫，查詢該筆資料，判斷是否有查到該筆帳號及密碼 `if (mysql_num_rows($result)!=1)`

```

<?
if(empty($_HTTP_POST_VARS["loginuser"])):
//顯示登入畫面
?>

<form action="login.php" method="post">
登入名稱:<input type="text" name="loginuser"> <br>
登入密碼:<input type="password" name="loginpass"> <br>
<input type="submit" name="submit" value="登入">
</form>
<br><h>
<a href="register.php"> 我要加入會員 </a>

<?
else :
    $loginuser=$_HTTP_POST_VARS["loginuser"];
    $loginpass=$_HTTP_POST_VARS["loginpass"];

//建立資料庫連線
    $lnk = @mysql_pconnect("localhost","test","test")
        or die("資料庫連線失敗!");
    mysql_select_db("memberdb",$lnk);

//寫出 query string
    $strSQL="select id,name,sex from members where
        loginusr='$loginuser' and loginpsd='$loginpass'";

```

5.9 多網頁間的認證

從根本上來說，**www** 網路是一個無狀態的環境，也就是說 **WEB** 伺服器無法識別不同登入的使用者。因為當某個使用者發出頁面請求時，**WEB** 伺服器只是簡單的進行回應，然後就關閉與該使用者的連接。因此，**WEB** 伺服器無法記錄同一個使用者發出的不同請求資訊。雖然，**WEB** 伺服器有此先天不足的缺點，但是現在卻可以利用 **cookie** 來保存某一使用者的識別資訊，從而實現有態連接。除非 **cookie** 超過有效期限，否則使用者每次訪問站點時，**WEB** 伺服器都可以獲取先前保存該用戶 **cookie** 中的資訊。

5.9.1 多網頁間的認證－使用 cookie 方式

下面，我們就來實際看一下如何具體應用 **cookie**。

首先，我們使用 **cookie** 為登入網站的使用者設置一個隨機生成的用戶 **ID**，以方便以後根據 **ID** 號碼來識別該使用者。程式碼如下：

```
<?php
if (!isset($id)) {
    srand((double)microtime()*1000000);
    $randvalue = rand(); setcookie("id",$randvalue,time()+14400,"/",".domain.name",0); }
?>
```

程式碼說明：

這裏，我們設置了一個名為 **id** 的 **cookie**，並將 **\$randval** 變數的值賦於該 **cookie**。同時，規定該 **cookie** 在 4 小時之後將被自動清除（現在時間加 14400 秒），並且在指定功能變數名稱下的所有頁面內有效。這樣，當使用者下次瀏覽網站頁面時，我們就可以直接使用 **\$HTTP_COOKIE_VARS** 環境陣列變數獲取先前設定的 **cookie** 值，即：

```
$id = $HTTP_COOKIE_VARS["id"]
```

現在我們知道可以利用 **cookie** 及加上一個不會跟別人重覆的 **ID** 號碼後，我們開始來練習一個例子吧！首先得在資料庫加入一個 **loginid** 的欄位來儲存 **ID** 號碼。

```
alter table members add column loginid CHAR(32) not null
    default '-1' after loginpsd;
alter table members add index(loginid);
```

接下來這隻程式是利用 **cookie** 來作認證的範例

(程式檔名：login_cookie.php)

```
<?
// 連線到資料庫
$linkptr = mysql_pconnect("localhost","ping","ping");
mysql_select_db("stech",$linkptr);

// 檢查 username, password
if (isset($user)):
$querystring = "select name from members where loginusr='$user' and loginpsd='$pass'";
    $result = mysql_query($querystring,$linkptr);

if (mysql_num_rows($result)!=1):
    // 如果沒有符合的資料，則印出提示訊息
    die("Username / Password does not match !!");
else:
    list($name) = mysql_fetch_row($result);
```

```

mysql_free_result($result);
mt_srand((double)microtime()*1000000);
//產生一個唯一的 ID 號碼，並用 md5 作加密
$loginid = md5(uniqid(mt_rand()));
//將產生出來的 ID 碼存入使用者的 cookie 中
setcookie("loginid",$loginid,time()+3600);
// 將產生出來的 ID 碼存入資料庫
$querystring = "update members set loginid='$loginid' where loginusr='$user' and
loginpsd='$pass'";
mysql_query($querystring,$linkptr);
endif;
elseif (isset($loginid)) :
// 已經有 loginid，檢查看看
$querystring = "select name from members where loginid='$loginid'";
$result = mysql_query($querystring,$linkptr);
if (mysql_num_rows($result)!=1) :
// 如果資料庫查不到該筆資料則將 loginid 變數刪除
unset($loginid);
else :
list($name) = mysql_fetch_row($result);
setcookie("loginid",$loginid,time()+3600);
endif;
endif ;?>

<html><head> <title>多網頁間的認證－使用 cookie 方式</title></head>
<body bgcolor="#FFFFFF">
<? //如果還沒有登入過，則顯示登入畫面
if (empty($user)&&empty($loginid)) :?>
<form action="login_cookie.php" method="POST">
UserName : <input type="Text" name="user"><br>
PassWord : <input type="Password" name="pass"><br>
<input type="Submit" name="submit" value="Send">
</form>
<? else :
echo "Hello ! $name ! Welcome to DW !!";
endif ?>
</body>
</html>

```

範例說明：

一開始使用者尚未登錄過網站，因此會先給使用者一個登入的畫面 `if(isset($user))` 即是用來判斷是否有經過登入畫面，如果有經過登入畫面，則自然這個變數 `$user` 就會有值，如果沒有值，則表示使用者可能已經登入過，所以將會直接去比對其 `loginid` 是否存在 即程式碼 `elseif(isset($loginid))`。

如果使用者登入成功，則會在將使用者的 `loginid` 儲存在資料庫及使用者電腦的 `cookie` 中，即程式碼 `setcookie("loginid",$loginid,time()+3600)` 及 `$querystring="update members set loginid='$loginid' where loginusr='$user' and loginpsd='$pass' "`；。接下來，使用者的 `loginid` 已經在他自己電腦上的 `cookie` 中了，所以在作認證時，只要將其 `cookie` 中的 `loginid` 拿出來與資料庫中的 `loginid` 作比對就可以了，如果比對成功，則通過認證。但是如果使用者的 `cookie` 過期了，則會要求使用者重新登入。

5.9.2 多網頁間的認證—使用 *session* 方式

由於，使用 `cookie` 的方式，需要使用者端將 `cookie` 開啓，才能將 `cookie` 存在使用者端。因此，我們要如何在使用者不提供 `cookie` 的情況下，來記錄使用者的狀態呢？那就是使用 `session` 的方式。接下來，我們將利用 `PHP4` 所提供的新功能 `session` 來作使用者的認證。

(程式檔名：`login_session.php`)

```
<?
function auth(){
    global $PHP_SELF, $sent, $mysession, $login, $passwd, $bye;
    //假如$bye 為 true，則表示使用者將登出，
    //那就要將 mysession 這個變數從 session 中刪除掉
    if ($bye) {
        session_start();
        session_unregister("mysession");
        session_destroy();
        return 0;
    }
}
```

```

if(isset($sent)):
    $login_ok = 0;
    if (isset($login) AND isset($passwd)):
        // 連線到資料庫
        $linkptr = mysql_pconnect("localhost","ping","ping") ;
        mysql_select_db("stech",$linkptr) ;
        // 檢查 username, password
    $querystring = "select name from members where loginusr='$login' and
loginpsd='$passwd'";
        $result = mysql_query($querystring,$linkptr) ;
        list($user)=mysql_fetch_row($result) ;
        if (mysql_num_rows($result)==1) :
            session_start();
            // create the session array
    $mysession = array ("login" => $login, "passwd" => $passwd, "ID" => session_id(), "valid"
=> 1);

            session_register("mysession");
            return 1;          // 認證成功,傳回"1"
            $login_ok = 1;
            break;
        endif;
    endif;

    if(!$login_ok):
        return 0;          //拒絕存取,傳回"0"
    endif;
else:          //使用 session 中的變數,來作認證
    $login_ok = 0;
    session_start();
//取出 session 中的陣列
foreach($GLOBALS["mysession"] as $elem):
    $ses_tmp[] = $elem;
        endforeach;
        $login = $ses_tmp[0];    //取出登入帳號放在 login 中
    $passwd = $ses_tmp[1];      //取出登入密碼放在 password 中

```

```

        // 連線到資料庫
        $linkptr = mysql_pconnect("localhost","ping","ping");
        mysql_select_db("stech",$linkptr);
        // 檢查 username, password
        $querystring = "select name from members where loginusr='$login' and
        loginpsd='$passwd'";
        $result = mysql_query($querystring,$linkptr);
        list($user)=mysql_fetch_row($result);
        if (mysql_num_rows($result)==1):
            session_start();
            // 建立 session 的陣列
        $mysession = array ("login" => $login, "passwd" => $passwd, "ID" => session_id(), "valid"
        => 1);

            session_register("mysession");
            return 1;
            // 認證成功
            $login_ok = 1;
        endif;

        if(!$login_ok):
            return 0;           // 拒絕存取
        endif;
    endif;
}

//登入畫面
function LoginForm(){
    global $PHP_SELF;
    ?>
    <html>
    <head>
        <title>login form</title>
    </head>
    <body>

```

```
<form method="post" action="<?php echo basename($PHP_SELF); ?>" name="loginform">
  <table frame=void rules=none WIDTH="300">
    <tr>
      <td align="left"> login: </td>
    <td align="left"> <input type="text" name="login" maxlength=50 size=15> </td>
    <tr>
      <td align="left"> password: </td>
    <td align="left"> <input type="password" name="passwd" maxlength=50 size=15> </td>
    <tr>
      <td align="left"> &nbsp; </td>
    <td align="left"> <input type=submit value="login" name="sent"> </td>
  </table>
</form>
<script type="text/javascript">
<!--
if (document.loginform) {
  document.loginform.login.focus();
}
// -->
</script>
</body>
</html>
<?php
}
```

```

//主程式
//初始 session 變數,mysession 爲一陣列
$mysession = array ("login"=>FALSE, "passwd"=>FALSE, "ID"=>FALSE,
"valid"=>FALSE);
    $uri = basename($PHP_SELF);
    $stamp = md5(srand(5));

if(!auth()):
    LoginForm();
else:
    $tmp = session_id();
    if (isset($sent)):
        echo "logged in from <b>login form</b><br><br>";
        echo "Your login name: <b>".$mysession["login"]."</b><br>";
        echo "Your password: <b>".$mysession["passwd"]."</b><br>";
        echo "<br><br><br><a href='$uri?SID=$tmp'> load </a>";
        echo "<br><a href='$uri?bye=true'>logout</a>";
    else:
        echo "logged in from <b>session:</b> $tmp<br>";
        echo "changing <b>stamp:</b> $stamp<br><br>";
        echo "Your login name: <b>".$mysession["login"]."</b><br>";
        echo "Your password: <b>".$mysession["passwd"]."</b><br>";
        echo "<br><br><br><a href='$uri?SID=$tmp&stamp=$stamp'>load</a>";
        echo "<br><a href='$uri?bye=true'>logout</a>";
    endif;
endif;
?>

```

範例說明：

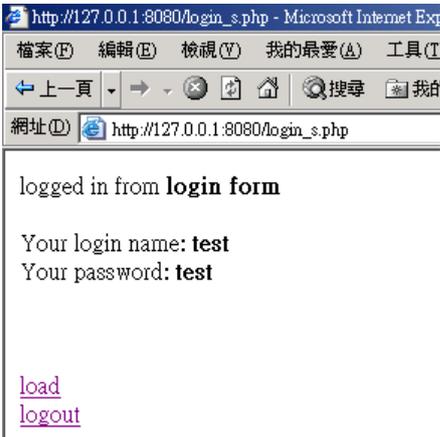
程式主要有三個部份，第一個部份即在作認證的部份—**function auth()**，第二個部份爲顯示登入的畫面—**function LoginForm()**，第三個部份則是主程式的部份。在 **function auth()** 中，一開始將初始化變數，**\$sent** 用來判斷是否有經過登入畫面，**\$mysession** 用來儲存 **session** 資訊的陣列，**\$login** 用來儲存使用者帳號，**\$passwd** 用來儲存使用者密碼，**\$bye** 用來判斷是否要登出的旗標。當第一次登入時，會判斷是使用者帳號及密碼是否正確，如果正確的話，則會將使用者的帳號、密碼、**session_id** 及 **valid** 儲存在 **session** 中。接下來使用者的 **session_id** 已經儲存在 **session** 中，在沒有登出前，將會保有該 **session_id**，直到登出爲止。

登入畫面

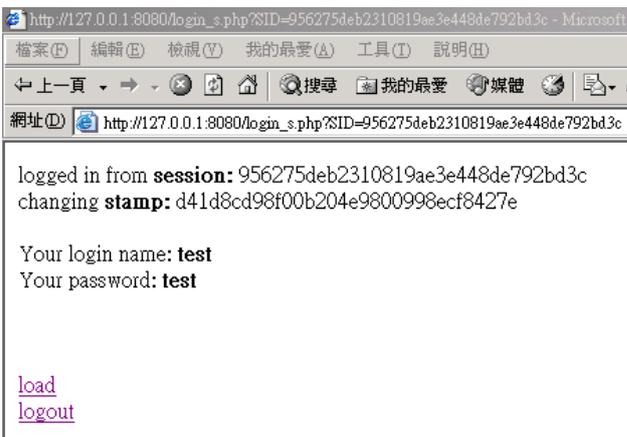
login:

password:

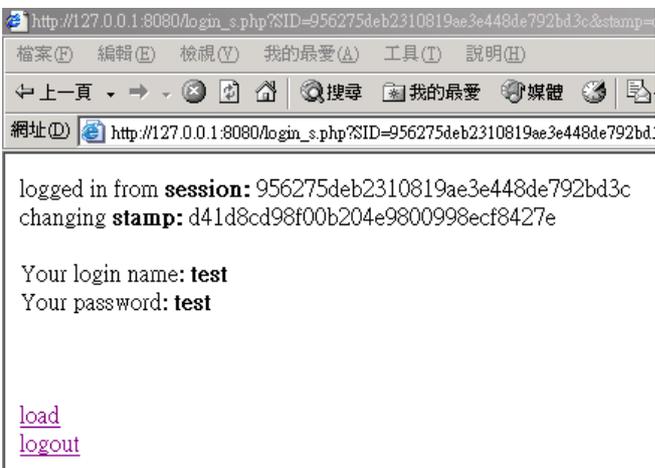
第一次登入畫面



第二次 **load**，即表示當從一個網頁到另一個網頁的情況。可以從畫面看到 **session_id** 的值。



當第三次 **load** 時，可以發現 **session_id** 是沒有變的，除非是登出。



第六章 基本資料檔維護(新增/刪除/修改/不同過濾條件的查詢)

6.1 主題

這一章我們將建立整個電子商務網站的主體，花束的資料庫。

6.2 目的

因為我們作的是一個花店的電子商務網站，所以我們必須將花店內所有的花束的資料，包括種類、名稱.....等，都放入資料庫中，以便於內部管理與對外的營業。

6.3 功能

在這一章節裡，我們要讓我們的網站擁有新增、刪除、修改，以及透過不同的過濾條件，來查詢我們所需要的資料。我們將一步一步的建立這些功能。

6.4 資料庫(資料表欄位)設計

假設我們要將花店內的花作分類，這個分類不是學理上的分類，而是我們對這種花的感覺的分類，假設我們有「情人、祝賀、結婚、生日」這四類，所以我們需要一個 **table** 來記錄種類，**SQL** 語法如下。

```
CREATE TABLE style (  
  styleid tinyint(4) NOT NULL auto_increment,  
  stylename char(20) NOT NULL default "",  
  PRIMARY KEY (styleid)  
);
```

其中的 **styleid** 為 **primary key**，則是為了搜尋與外鍵使用時增進效率，此外，我們的花束資料還需要「類別 **id**、花名、花材、價位、圖片檔名」等，**SQL** 語法如下。

```
CREATE TABLE flowers (  
  flowerid int(11) NOT NULL auto_increment,  
  styleid tinyint(4) NOT NULL default '0',  
  flowername varchar(20) default NULL,  
  material varchar(60) default NULL,  
  price smallint(6) default NULL,  
  picture varchar(20) default NULL,  
  PRIMARY KEY (flowerid),  
  KEY styleid (styleid));
```

其中的 flowerid 做爲花束編號，爲 primary key。另外在花材 material 的部分採用 varchar，則是爲了節省空間。

6.5 操作流程

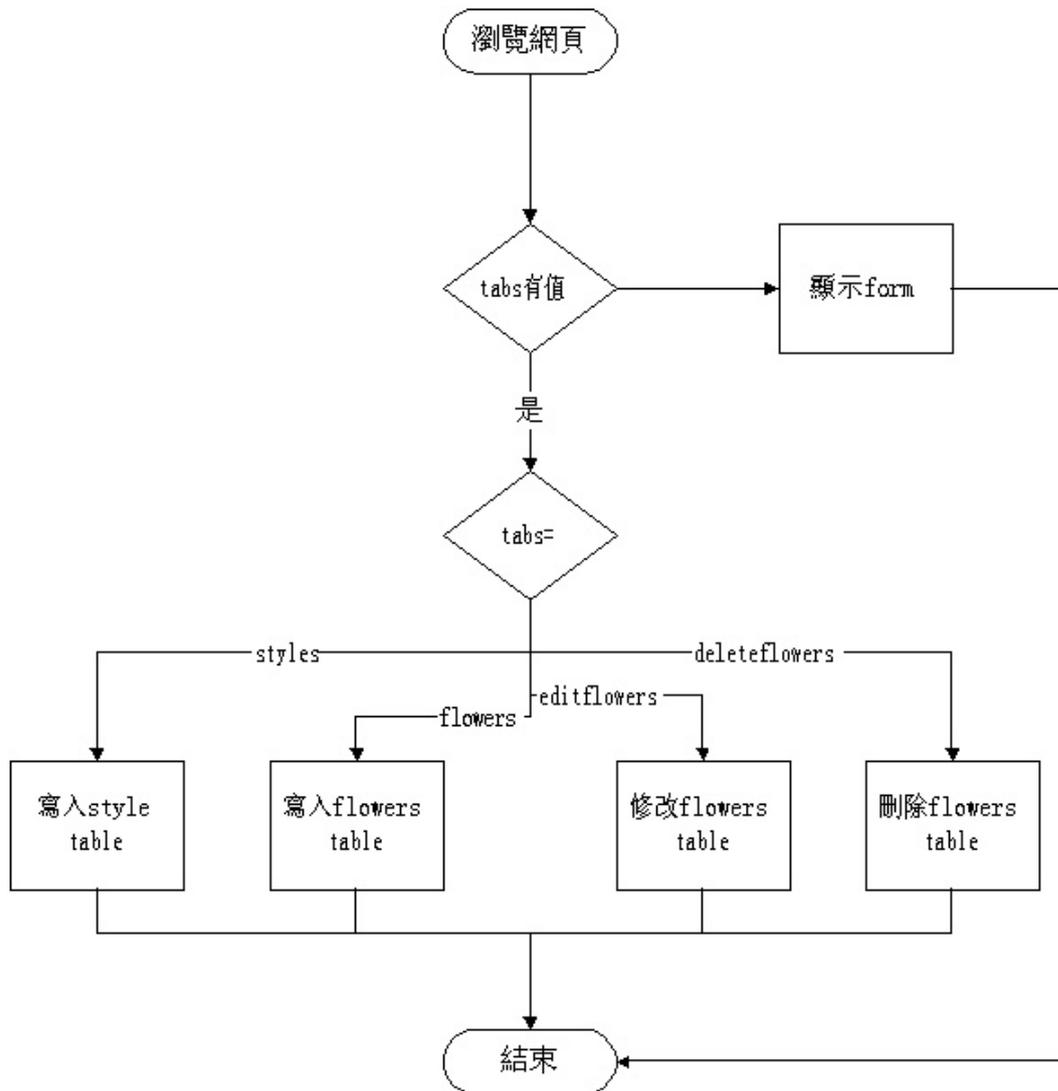


圖 6-1

6.6 執行畫面範例

花束類別、資料新增畫面



The screenshot shows a Microsoft Internet Explorer browser window titled "花束資料輸入 - Microsoft Internet Explorer". The address bar displays "http://localhost/ch06/DEMO6-1.php". The main content area contains a form with the following fields and buttons:

- "請輸入類別:" followed by a text input field and a "新增類別" button.
- "請輸入花名:" followed by a text input field.
- "請選擇種類:" followed by a dropdown menu with "愛情" selected.
- "請輸入花材:" followed by a text input field.
- "請輸入價位:" followed by a text input field.
- "請輸入圖片名稱:" followed by a text input field and a "瀏覽..." button.
- At the bottom of the form, there are two buttons: "新增花束資料" and "花束資料列表".

The browser's status bar at the bottom shows "完成" and "近端內部網路".

按下新增類別，可新增花束的種類到 style table。

按下新增花束，可將花名、種類、花材.....等資料新增到 flowers table。

按下花束資料列表，可以連結到圖 6-2。

圖 6-2

花束列表管理畫面



按下修改圖示，可連結到圖 6-3 作資料修改。

按下刪除圖示，可直接將該筆資料刪除。

按下新增花束資料，可連結到圖 6-1 作資料新增。

按下搜尋花束資料，可以連結到圖 6-4。

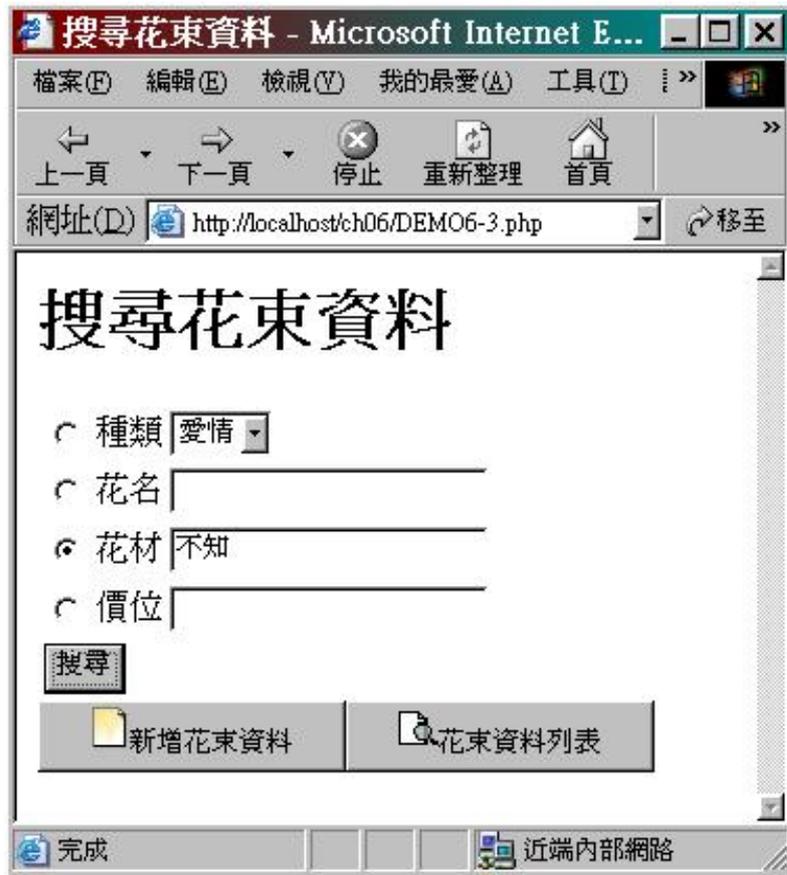
圖 6-3

花束資料修改畫面



圖 6-3

搜尋花束資料畫面



點選所要搜尋的項目，並選擇或填入搜尋的內容，按下搜尋按鈕，可連結到圖 6-5 作資料搜尋。按下花束資料列表，可以連結到圖 6-2。

圖 6-4

搜尋花束資料結果畫面

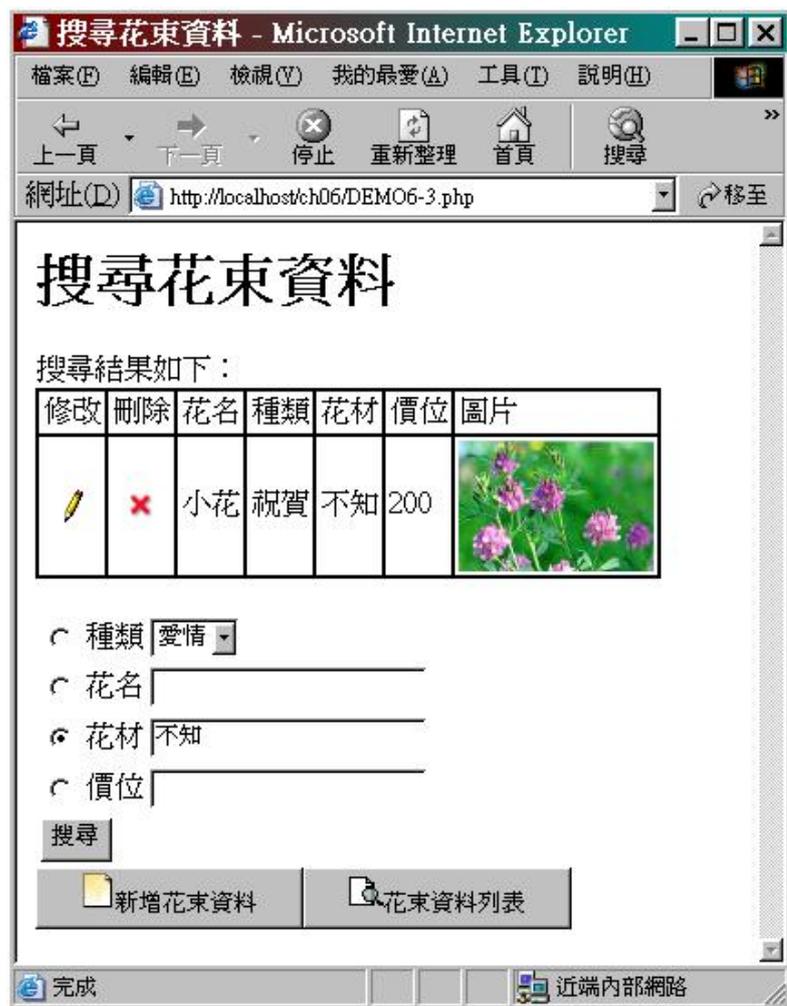


圖 6-5

依照圖 6-4 所作輸入的條件顯示搜尋結果，並將所選擇的條件顯示在下方的資料輸入列中。按下修改圖示，可連結到圖 6-3 作資料修改。按下刪除圖示，可直接將該筆資料刪除。按下新增花束資料，可連結到圖 6-1 作資料新增。按下花束資料列表，可以連結到圖 6-2。

6.7 程式碼簡介

DEMO6-1.php

新增花束類別的部分。

```
$querystring = "insert into style (stylename) values ('$fstyle')";  
mysql_query($querystring,$linkptr);
```

新增花束資料的部分，其中 copy 函數為將 client 端的圖片複製到 server 端。

```
$querystring = "insert into flowers (styleid,flowername,material,price,picture)  
values ('$fstyle','$fname','$material','$price','$pict')";  
mysql_query($querystring,$linkptr);  
copy($_FILES["userfile"]["tmp_name"],$uploadpath.$flowerid.strchr($_FILES["userfile"]["name"],  
."));
```

修改花束資料的部分，如果當使用者沒有上傳圖片時，不要更新 picture 欄位。

```
$querystring = "update flowers set styleid = $fstyle , flowername = '$fname' , material =  
'$material' , price = $price ";  
if(!empty($_FILES["userfile"]["name"])){  
$querystring .= " , picture = '$pict'";  
}  
$querystring .= " where flowerid = '$flowerid' ";  
mysql_query($querystring,$linkptr);
```

刪除花束資料的部分，刪除該筆花束資料，當然也要把圖片也一起刪除，避免浪費硬碟空間。

```
unlink($uploadpath.$pict); //將圖片從資料夾刪除  
$querystring = "delete from flowers where flowerid = '$flowerid' ";  
mysql_query($querystring,$linkptr);
```

DEMO6-2.php

此檔案主要是將資料用表格的方式呈現給使用者。

```
$linkptr = mysql_pconnect("localhost","florist","123456");  
mysql_select_db("florist",$linkptr);  
$querystring = "select flowerid , stylename , flowername , material , price , picture from flowers A ,  
style B where A.styleid = B.styleid order by flowerid";  
$result = mysql_query($querystring);  
while(list($flowerid,$stylename,$flowername,$material,$price,$picture) =  
mysql_fetch_row($result)){
```

```

echo "<tr>\n";
echo "<td align='center'><img src='image/edit.gif"
onclick=\"location.href='DEMO6-1.php?edit=1&flowerid=$flowerid'\"></td>\n";
echo "<td align='center'><img src='image/delete.gif"
onclick=\"location.href='DEMO6-1.php?tabs=deleteflowers&delete=1&flowerid=$flowerid'\"></td>\n";
echo "<td>$flowername</td>\n";
echo "<td>$stylename</td>\n";
echo "<td>$material</td>\n";
echo "<td>$price</td>\n";
echo "<td align='center'><img src='uploadimage/$picture'></td>\n";
echo "</tr>\n";
}

```

DEMO6-3.php

此檔案主要是提供使用者輸入搜尋條件尋找符合的資料。

判斷所要採用的搜尋語法

```

if($search=='fstyle'){ $qstr="B.styleid='$fstyle'"; } //以種類為條件
if($search=='fname'){ $qstr="A.flowername like '$fname'"; } //以花名為條件
if($search=='material'){ $qstr="A.material like '$fmaterial'"; } //以花材為條件
if($search=='price'){ $qstr="A.price='$fprice'"; } //以價位為條件
$querystring = "select flowerid , stylename , flowername , material , price , picture from flowers A ,
style B where A.styleid = B.styleid && ".$qstr." order by flowerid";
$result = mysql_query($querystring);

```

選擇及輸入搜尋條件，搜尋完畢後，會將結果及所採用的條件顯示出來。

```

<form name="frmdata1" action="DEMO6-3.php" method="POST">
<tr><td><input type="radio" name="search" value="fstyle"
<?if($search=="fstyle"){?>checked<?}?>></td><td>種類</td>
<td><select name="fstyle" <?if($search=="fstyle"){?>value="<?=$fstyle?>"<?}?>>
    <?
        $querystring = "select * from style order by styleid" ;
        $result = mysql_query($querystring);
        while(list($styleid,$stylename) = mysql_fetch_row($result)){
            if($styleid == $fstyle) $sel = " selected ";
            else $sel = "";
            echo "<option value=\"\$styleid\"$sel>$stylename</option>\n";
        }
    }

```

```

        ?>
        </select></td>
</tr>
<tr><td><input type="radio" name="search" value="fname"
<?if($search=="fname"){?>checked<?}?>></td><td>花名</td>
    <td><input type="text" name="fname"
<?if($search=="fname"){?>value="<?=$fname?>"<?}?>></td>
</tr>
<tr><td><input type="radio" name="search" value="material"
<?if($search=="material"){?>checked<?}?>></td><td>花材</td>
    <td><input type="text" name="fmaterial"
<?if($search=="material"){?>value="<?=$fmaterial?>"<?}?>></td>
</tr>
<tr><td><input type="radio" name="search" value="price"
<?if($search=="price"){?>checked<?}?>></td><td>價位</td>
    <td><input type="text" name="fprice"
<?if($search=="price"){?>value="<?=$fprice?>"<?}?>></td>
</tr>
<tr><td colspan="3"><input type="submit" value="搜尋"></td></tr>
</form>

```

6.8 加簡要註解的完整 PHP 程式碼列表

```

DEMO6-1.php
<html>
<head>
    <title>花束資料輸入</title>
</head>
<body>
<?
// 連線到資料庫
$linkptr = mysql_pconnect("localhost","florist","123456");
mysql_select_db("florist",$linkptr);

$tabs = $_REQUEST["tabs"];
$edit = $_REQUEST["edit"];
$delete = $_REQUEST["delete"];
$flowerid = $_REQUEST["flowerid"];
$fstyle = $_REQUEST["fstyle"];

```

```

$name = $_REQUEST["fname"];
$material = $_REQUEST["material"];
$price = $_REQUEST["price"];
$pict = $_FILES["userfile"]["name"];

// 判斷有沒有資料輸入
if (empty($tabs)) :?>
<!-- 這裡是輸入畫面的程式片段 -->
<!-- 第一個 form，輸入類別 -->
<? if (empty($edit)) :?>
<form name="frmdata1" action="DEMO6-1.php" method="POST">
    <!-- 告訴後面這是要到 style 的 -->
    <input type="Hidden" name="tabs" value="styles">
    請輸入類別： <input type="Text" name="fstyle">
    <button onClick="frmdata1.submit();" >新增類別</button>
</form>
<hr>
<?endif;?>

<!-- 第二個 form，輸入花束資料 -->
<form name="frmdata2" action="DEMO6-1.php" method="POST" enctype="multipart/form-data">

    <!-- 告訴後面這是要到 flowers 的 -->
    <? if (empty($edit)) :?>
    <input type="Hidden" name="tabs" value="flowers">
    <? else :
        $querystring = "select styleid , flowername , material , price , picture from flowers
where flowerid = '$flowerid'";
        $result = mysql_query($querystring);
        list($fstyle,$fname,$material,$price,$pict) = mysql_fetch_row($result);
    ?>
    <input type="Hidden" name="tabs" value="editflowers">
    <input type="Hidden" name="flowerid" value="<?echo $flowerid;?>">
    <? endif;?>
    請輸入花名： <input type="Text" name="fname" value="<?if ($edit=="1") echo
$name;?>"><br>
    請選擇種類：
    <select name="fstyle">
    <?

```

```

$querystring = "select * from style order by styleid" ;
$result = mysql_query($querystring);
while(list($styleid,$stylename) = mysql_fetch_row($result)){
    if($styleid == $fstyle) $sel = " selected ";
    else $sel = "";
    echo "<option value=\"\$styleid\"$sel>$stylename</option>\n";
}
?>
</select><br>
請輸入花材： <input type="Text" name="material" value="<?if ($edit=="1") echo
$material;?>"><br>
請輸入價位： <input type="Text" name="price" value="<?if ($edit=="1") echo
$price;?>"><br>
請輸入圖片名稱： <input type="file" name="userfile"><br>
<button onClick="frmdata2.submit();">"><?if ($edit=="1") echo "修改"; else echo "新增"?>花束資料</button>
<button name="b2" onclick="location.href='DEMO6-2.php'">花束資料列表</button>
</form>

<?
else :
$patharr = explode("\\",$_SERVER['PATH_TRANSLATED']);
$uploadpath = "";
for($i = 0 ; $i < sizeof($patharr)-1 ; $i++){
    $uploadpath .= $patharr[$i]."\\";
}
$uploadpath .= "uploadimage\\"; //取得上傳檔案的路徑
$pict = $flowerid.strchr($_FILES["userfile"]["name"],"."); //上傳之後的檔名
if (!is_numeric($price)) $price = 0; //如果價錢輸入的不是數字，就將價錢歸 0
switch ($tabs) :
    case "styles" :
        // 這是新增類別的輸入部分
        $querystring = "insert into style (stylename) values ('$fstyle')";
        mysql_query($querystring,$linkptr) ;
        echo "<script>alert(\".$fstyle.\"類別新增成功
');location.href='DEMO6-1.php';</script>" ;
        break ;

```

```

case "flowers" :
    // 這是新增花束的輸入部分

    $querystring = "insert into flowers (styleid,flowername,material,price,picture)
        values ($fstyle,'$fname','$material',$price,'$pict') " ;
    mysql_query($querystring,$linkptr) ;
    $result = mysql_query("select max(flowerid) from flowers");
    list($flowerid) = mysql_fetch_row($result); //因為新增時 flowerid 還不知道，
    所以先新增完後再將圖檔欄位更新
    $querystring = "update flowers set picture = '$flowerid$pict' where flowerid =
    $flowerid";
    mysql_query($querystring);

    if(!empty($_FILES["userfile"]["name"])){ //如果沒有選擇圖片，則不作上傳的
    動作

        copy($_FILES["userfile"]["tmp_name"],$uploadpath.$flowerid.strchr($_FILES["userfile"]["na
        me"],"."));

            //將圖片上傳到資料夾
        }
        echo "<script>alert('花束資料新增成功
        ');location.href='DEMO6-1.php';</script> " ;
        break;

    case "editflowers" :
        // 這是修改花束資料的部分

        $querystring = "update flowers set styleid = $fstyle , flowername = '$fname' ,
        material = '$material' , price = $price ";
        if(!empty($_FILES["userfile"]["name"])){
            $querystring .= " , picture = '$pict'";
        }
        $querystring .= " where flowerid = '$flowerid' ";

        mysql_query($querystring,$linkptr) ;
        if(!empty($_FILES["userfile"]["name"])){

            copy($_FILES["userfile"]["tmp_name"],$uploadpath.$flowerid.strchr($_FILES["userfile"]["na
            me"],"."));

```

```

        //將圖片上傳到資料夾
    }
    echo "<script>alert('花束資料修改成功
');location.href='DEMO6-2.php';</script>";
    break;
case "deleteflowers" :
    // 這是刪除花束資料的部分

    $querystring = "select picture from flowers where flowerid = '$flowerid' ";

    $result = mysql_query($querystring,$linkptr);
    list($pict) = mysql_fetch_row($result);
    unlink($uploadpath.$pict); //將圖片從資料夾刪除

    $querystring = "delete from flowers where flowerid = '$flowerid' ";
    mysql_query($querystring,$linkptr);

    echo "<script>alert('花束資料刪除成功
');location.href='DEMO6-2.php';</script>";
    break;
endswitch ;
endif ;
?>
</body>
</html>

```

DEMO6-2.php

```

<html>
<head>
    <title>花束資料管理列表</title>
</head>

<body bgcolor="#FFFFFF">
<h1>花束資料列表</h1>

<table border=1 cellspacing="0" cellpadding="2" bordercolor="#000000" >
    <tr>
        <td>修改</td>

```

```

        <td>刪除</td>
        <td>花名</td>
        <td>種類</td>
        <td>花材</td>
        <td>價位</td>
        <td>圖片</td>
    </tr>
<?
    $linkptr = mysql_pconnect("localhost","florist","123456") ;
    mysql_select_db("florist",$linkptr) ;
    $querystring = "select flowerid , stylename , flowername , material , price , picture from
flowers A , style B where A.styleid = B.styleid order by flowerid";
    $result = mysql_query($querystring);
    while(list($flowerid,$stylename,$flowername,$material,$price,$picture) =
mysql_fetch_row($result)){
        echo "<tr>\n";
        echo "<td align='center'><img src='image/edit.gif"
onclick=\"location.href='DEMO6-1.php?edit=1&flowerid=$flowerid'\"></td>\n";
        echo "<td align='center'><img src='image/delete.gif"
onclick=\"location.href='DEMO6-1.php?tabs=deleteflowers&delete=1&flowerid=$flowerid'\"></td>\n";
        echo "<td>$flowername</td>\n";
        echo "<td>$stylename</td>\n";
        echo "<td>$material</td>\n";
        echo "<td>$price</td>\n";
        echo "<td align='center'><img src='uploadimage/$picture'></td>\n";
        echo "</tr>\n";
    }
?>
</table>
<button name="b1" onclick="location.href='DEMO6-1.php'">新增花束資料</button>
<button name="b2" onclick="location.href='DEMO6-3.php'">搜尋花束資料</button>
</body>
</html>
DEMO6-3.php
<html>
<head>

```

```

        <title>搜尋花束資料</title>
</head>
<body bgcolor="#FFFFFF">
<h1>搜尋花束資料</h1>
<?
// 連線到資料庫
    $linkptr = mysql_pconnect("local","florist","123456");
    mysql_select_db("florist",$linkptr);

// 判斷有沒有資料輸入
if (!empty($search)){?>
搜尋結果如下：
<table border=1 cellspacing="0" cellpadding="2" bordercolor="#000000" >
    <tr>
        <td>修改</td>
        <td>刪除</td>
        <td>花名</td>
        <td>種類</td>
        <td>花材</td>
        <td>價位</td>
        <td>圖片</td>
    </tr>
<?
    if($search=='fstyle'){ $qstr="B.styleid='$fstyle'";} //以種類為條件
    if($search=='fname'){ $qstr="A.flowername like '$fname'";} //以花名為條件
    if($search=='material'){ $qstr="A.material like '$fmaterial'";} //以花材為條件
    if($search=='price'){ $qstr="A.price='$fprice'";} //以價位為條件
    $querystring = "select flowerid , stylename , flowername , material , price , picture from
flowers A , style B where A.styleid = B.styleid && ".$qstr." order by flowerid";
    $result = mysql_query($querystring);
    while(list($flowerid,$stylename,$flowername,$material,$price,$picture) =
mysql_fetch_row($result)){

        echo "<tr>\n";
        echo "<td align='center'><img src='image/edit.gif'
onclick=\"location.href='DEMO6-1.php?edit=1&flowerid=$flowerid'\"></td>\n";
        echo "<td align='center'><img src='image/delete.gif'
onclick=\"location.href='DEMO6-1.php?tabs=deleteflowers&delete=1&flowerid=$flowerid'\"></td>\n";
    }
}

```

```

        echo "<td>${flowername}</td>\n";
        echo "<td>${stylename}</td>\n";
        echo "<td>${material}</td>\n";
        echo "<td>${price}</td>\n";
        echo "<td align='center'><img src='uploadimage/${picture}'></td>\n";
        echo "</tr>\n";
    }
?>
</table>
<?}?>
<p> </p>
<table border="0">
<!-- 這裡是輸入畫面的程式片段 -->
<!-- 輸入搜尋條件 -->
<form name="frmdata1" action="DEMO6-3.php" method="POST">
    <tr><td><input type="radio" name="search" value="fstyle"
<?if($search=="fstyle"){?>checked<?}?>></td><td>種類</td><td>
        <select name="fstyle" <?if($search=="fstyle"){?>value="<?=$fstyle?>"<?}?>>
        <?
            $querystring = "select * from style order by styleid" ;
            $result = mysql_query($querystring);
            while(list($styleid,$stylename) = mysql_fetch_row($result)){
                if($styleid == $fstyle) $sel = " selected ";
                else $sel = "";
                echo "<option value=\"\$styleid\"$sel>${stylename}</option>\n";
            }
        ?>
        </select></td>
    </tr>
    <tr><td><input type="radio" name="search" value="fname"
<?if($search=="fname"){?>checked<?}?>></td><td>花名</td>
        <td><input type="text" name="fname"
<?if($search=="fname"){?>value="<?=$fname?>"<?}?>></td>
    </tr>
    <tr><td><input type="radio" name="search" value="material"
<?if($search=="material"){?>checked<?}?>></td><td>花材</td>
        <td><input type="text" name="fmaterial"
<?if($search=="material"){?>value="<?=$fmaterial?>"<?}?>></td>
    </tr>

```

```

        <tr><td><input type="radio" name="search" value="price"
<?if($search=="price"){?>checked<?}>></td><td>價位</td>
        <td><input type="text" name="fprice"
<?if($search=="price"){?>value="<?=$fprice?>"<?}>></td>
        </tr>
        <tr><td colspan="3"><input type="submit" value="搜尋"></td></tr>
</form>
</table>
<button name="b1" onclick="location.href='DEMO6-1.php'">新增花束資料</button>
<button name="b2" onclick="location.href='DEMO6-2.php'">花束資料列表</button>
</body>
</html>

```

第七章 留言板, 群組討論板, 線上即時聊天室

7.1 留言板

7.1.1 整體架構：

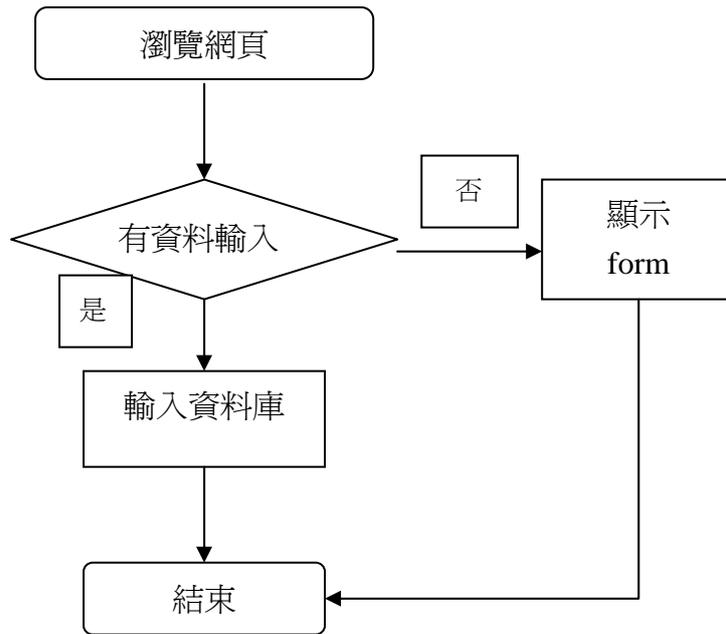
留言板的基本功能不外是記錄留言人資料、留言內容。資料庫架構可以這麼寫：

```
create table msqboard (  
    msqid smallINT unsigned not null auto_increment,  
    authname varCHAR(20),  
    authemail varCHAR(50),  
    totem varCHAR(20),  
    posttime DATETIME,  
    msq tinyBLOB,  
    primary key (msqid)  
);
```

msqid 作 primary key 方便後面直接抓取；另外記錄留言者的名字和 email、留言時間與內容。totem 就是可以代表心情的小圖案。

7.1.2 輸入留言：

規劃一下流程，用一個 form 輸入留言，輸入後記錄在資料庫中。



程式碼：

```

<html>
<head>
  <title>Demo 11-1</title>
</head>
<body bgcolor="#FFFFFF">
<?if (empty($submit)) :?>
<!-- Show form -->
<form action="demo11-1.phtml" method="POST">
<table>
  <tr><td colspan="3">最接近您現在心情的圖樣？</td></tr>
  <tr><td><input type="Radio" name="totem" value="totem1.gif" checked></td>
  <td><input type="Radio" name="totem" value="totem2.gif"></td>
  <td><input type="Radio" name="totem" value="totem3.gif"></td></tr>
</table>
<table>
  <tr><td>怎麼稱呼您？</td><td><input type="Text" name="poster"></td></tr>
  <tr><td>留個 EMail ？</td><td><input type="Text" name="email"></td></tr>
</table>
<table>

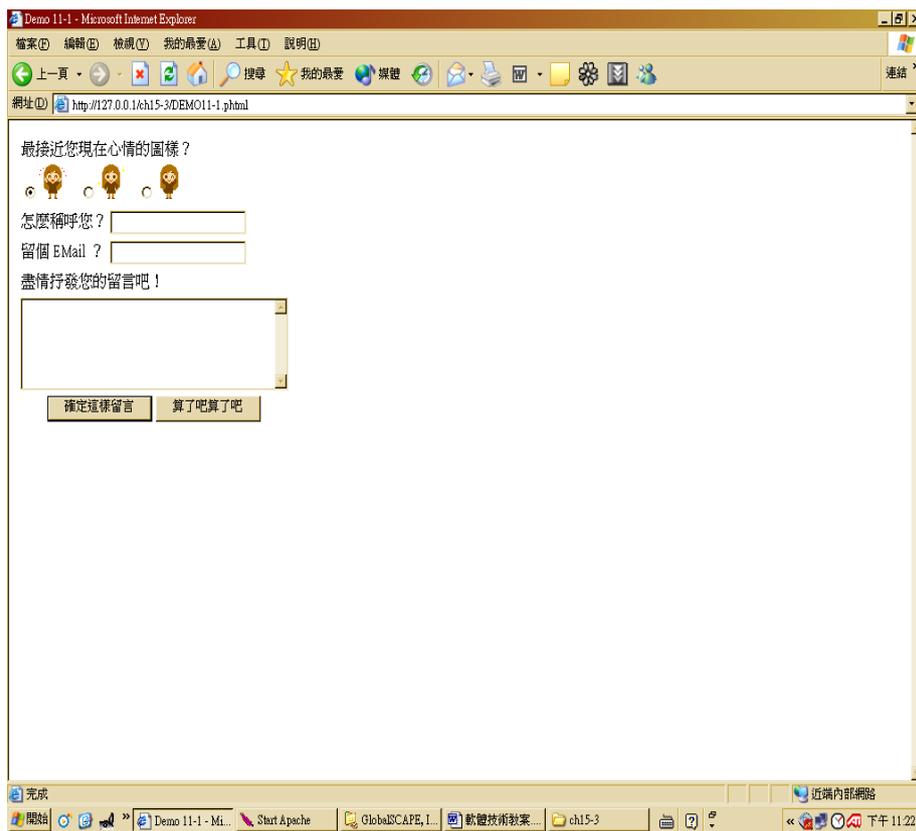
```

```

<tr><td>盡情抒發您的留言吧！</td></tr>
<tr><td><textarea name="msg" cols="40" rows="5"
wrap="VIRTUAL"></textarea></td></tr>
<tr><td align="CENTER"><input type="Submit" name="submit" value="確定這樣留
言">&nbsp;<input type="Reset" value="算了吧算了吧"></td></tr>
</table>
</form>
<?else :
    // 輸入資料庫
    // 連線到資料庫
    $linkptr = mysql_pconnect("localhost","florist","123456");
    mysql_select_db("florist",$linkptr);
    // 做一些基本的篩檢
    if (empty($poster)) $poster="無名氏";
    if (!strpos($email,"@")) $email="";
    if (empty($msg)) $msg="沒有留言";
    $msg = addslashes(nl2br($msg));
    // 寫入資料庫
    $querystring = "insert into msgboard (totem,authname,authemail,posttime,msg) values
('$stotem','$poster','$email',NOW(),'$msg')";
    mysql_query($querystring,$linkptr);
    // 輸出一些訊息
    echo "$poster ，謝謝您的留言！";
    echo $querystring;
endif ;?>
</body>
</html>

```

輸出結果：



當訪客輸入資料後，接下來會連線到資料庫，做些基本的篩檢，決定是否將此筆留言寫入資料庫 msgboard 和做預設的回應。

7.1.3 觀看留言

程式碼：

```
<html>
<head>
  <title>Demo 11-2</title>
</head>
<body bgcolor="#FFFFFF">
<?
// 連線到資料庫
$linkptr = mysql_pconnect("localhost","florist","123456");
mysql_select_db("florist",$linkptr);

$querystring = "select totem,authname,authemail,posttime,msg from msgboard order by
```

```

msgid desc" ;
$result = mysql_query($querystring,$linkptr) ;

while ( list($totem,$authname,$authemail,$posttime,$msg) = mysql_fetch_row($result) ) :
    $totem = "../bookfigures/" . $totem ;
    $imgsize = getimagesize($totem) ;
    echo "<table>\n<tr><td><img src=\"\$totem\" \" . $imgsize[3] . \" border=0>" ;
    if ($authemail!="") { echo "<a href=\"mailto:$authemail\">$authname</a> 在
$posttime 寫下 : </td></tr>\n" ; }
    else { echo "$authname 在 $posttime 寫下 : </td></tr>\n" ; }
    echo "<tr><td>$msg</td></tr></table>\n" ;
endwhile ;
?>
</body>
</html>

```

輸出結果：

若在上圖「怎麼稱呼您」中輸入：JONN
 留言的部份輸入：ABC
 然後按下「確定這樣留言」
 則會顯示出： **JOHN** ，謝謝您的留言！ABC

7.1.4 板主管理

板主管理的任務主要處理留言，尤其是刪除留言更為常見的執行動作。以下就舉板主處理刪除留言的部份，配合以上的例子做說明。

程式碼：

```

<html>
<head>
    <title>Demo11-5</title>
</head>
<body>
<? if (empty($pass)) :?>
<form action="demo11-5.phtml" method="POST">
留言版管理<br>

```

```

PassWord : <input type="Password" name="pass"><br>
<input type="Submit" name="submit" value="Send">
</form>
<?else :
    if ($pass!="password1") :
        echo "密碼錯誤。";
        exit();
    endif ;

    // 已通過認證
    // 連線到資料庫
    $linkptr = mysql_pconnect("localhost","florist","123456");
    mysql_select_db("florist",$linkptr);

    if (empty($msgid)) :
        // 顯示選擇畫面
        $querystring = "select msgid,authname,posttime,left(msg,20) from msgboard order
by msgid desc";
        $result = mysql_query($querystring,$linkptr);

        while (list($msgid,$authname,$posttime,$msg)=mysql_fetch_row($result)) :
            echo "$authname 在 $posttime 所說的 : <br>" . htmlspecialchars($msg) .
" .....<br>\n" ;?>
<form action="demo11-5.phtml" method="post">
<input type="Hidden" name="pass" value="<?echo $pass ;?>">
<input type="Hidden" name="msgid" value="<?echo $msgid ;?>">
<input type="Submit" name="submit" value="刪除！">
</form><hr width="300" align="left"><?
    endwhile ;
    else :
        // 刪除特定資料
        $querystring = "delete from msgboard where msgid=$msgid";
        mysql_query($querystring,$linkptr);
        echo "該筆資料已刪除。";

    endif ; // end msgid check

endif ; // end password check

```

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

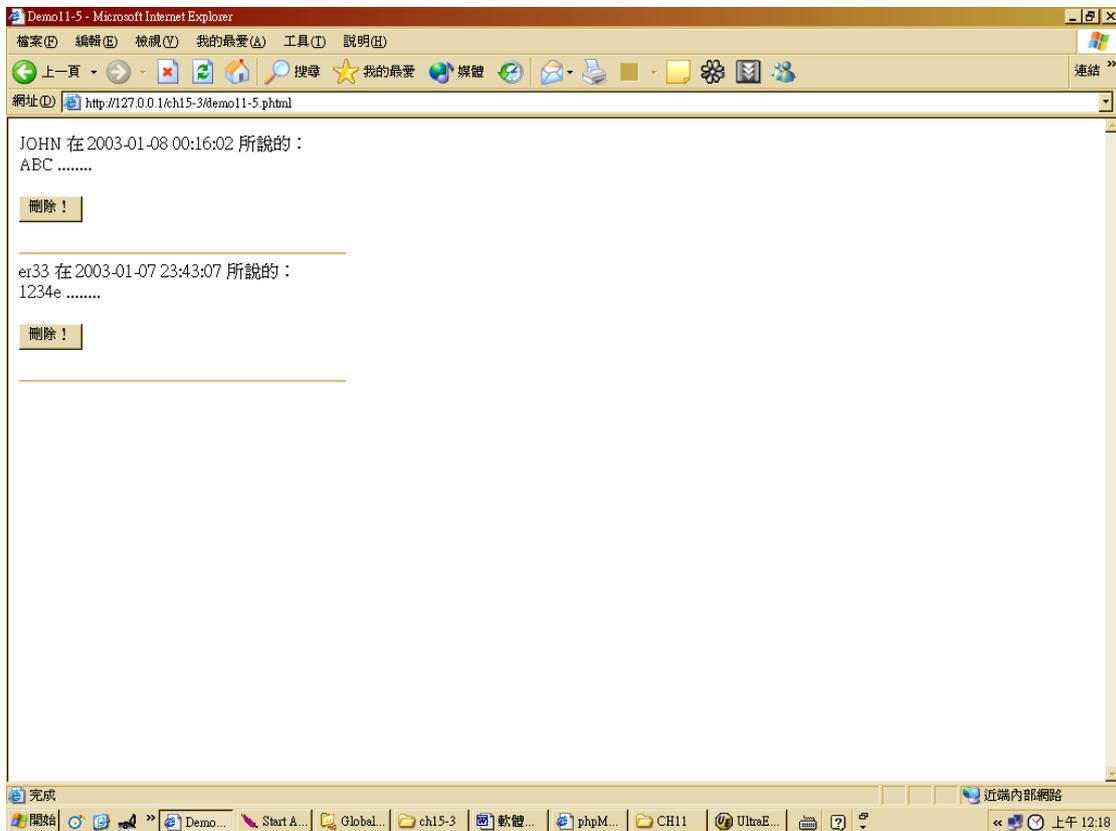
輸出結果：

先會出現認證板主的畫面：

留言板管理

PassWord :

再依據程式碼設定的密碼：password1 輸入即會出現以下的畫面



JOHN 輸入留言的例子就出現在上述畫面中。若刪除 JOHN 的留言，則出現：“該筆資料已刪除”的字樣。

7.2 群組討論版

7.2.1 簡介

「群組討論版」即為一般常見之「討論區」，與留言版之最大不同應在於討論區能依各討論主題作留言的區隔；而留言版就只有一個版面，依留言時間先後順序呈現在瀏覽的網頁上，時間一久，留言越多，則版面感覺越混亂；所以並不適合同時間內有數個主題交錯討論的情形。本節僅簡單做出討論區的雛形以供參考。

7.2.2 程式架構

- 資料庫 *florist* - 資料表 *forumreply*

欄位	型態	屬性	Null	預設值	附加
<u>ReplyID</u>	int(10)	UNSIGNED	否		auto_increment
Subject	varchar(200)		否		
Message	mediumtext		否		
PostTime	timestamp(14)		是	NULL	
Email	varchar(200)		否		
Author	varchar(200)		否		
TopicID	int(10)	UNSIGNED	否	0	

ReplyID: 將每篇回覆文章編號。 Subject: 回覆文章的主題。

Message: 回覆的訊息(文章)。 PostTime: 發表文章的時間。

Email: 發文者電子郵件。 Author: 發文者。

TopicID: 回覆文章所屬討論主題的編號。

- 資料庫 *florist* - 資料表 *forumtopic*

欄位	型態	屬性	Null	預設值	附加
<u>TopicID</u>	int(10)	UNSIGNED	否		auto_increment
Subject	varchar(200)		否		
Author	varchar(200)		否	Guest	
LastPostTime	timestamp(14)		是	NULL	
ReplyNum	int(10)	UNSIGNED	否	0	

TopicID: 討論主題的編號。

Subject: 討論主題(標題)。

Author: 發文者。

LastPostTime: 討論串做後一篇回應時間。

ReplyNum: 已回覆文章的數目。

- ForumTopic_Query.php

展示討論區所有主題。



- ForumTopic_Form.php

發表文章用的表

單

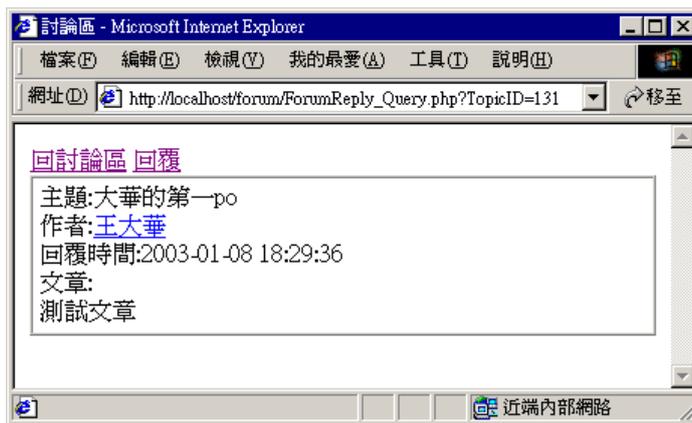


- ForumTopic_Modify.php

將 ForumTopic_Form.php 表單上輸入的資料：作者、e-mail、主題、文章，存入資料庫。

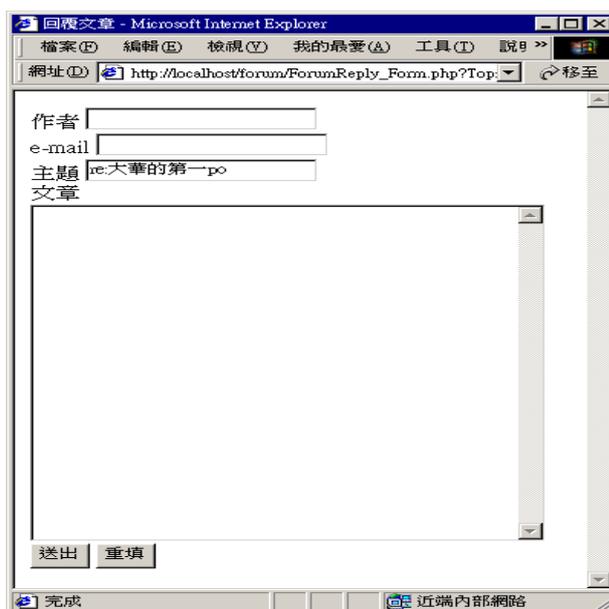
- ForumReply_Query.php

顯示個別主題下的討論串。



- ForumReply_Form.php

回覆某主題下的討論留言。

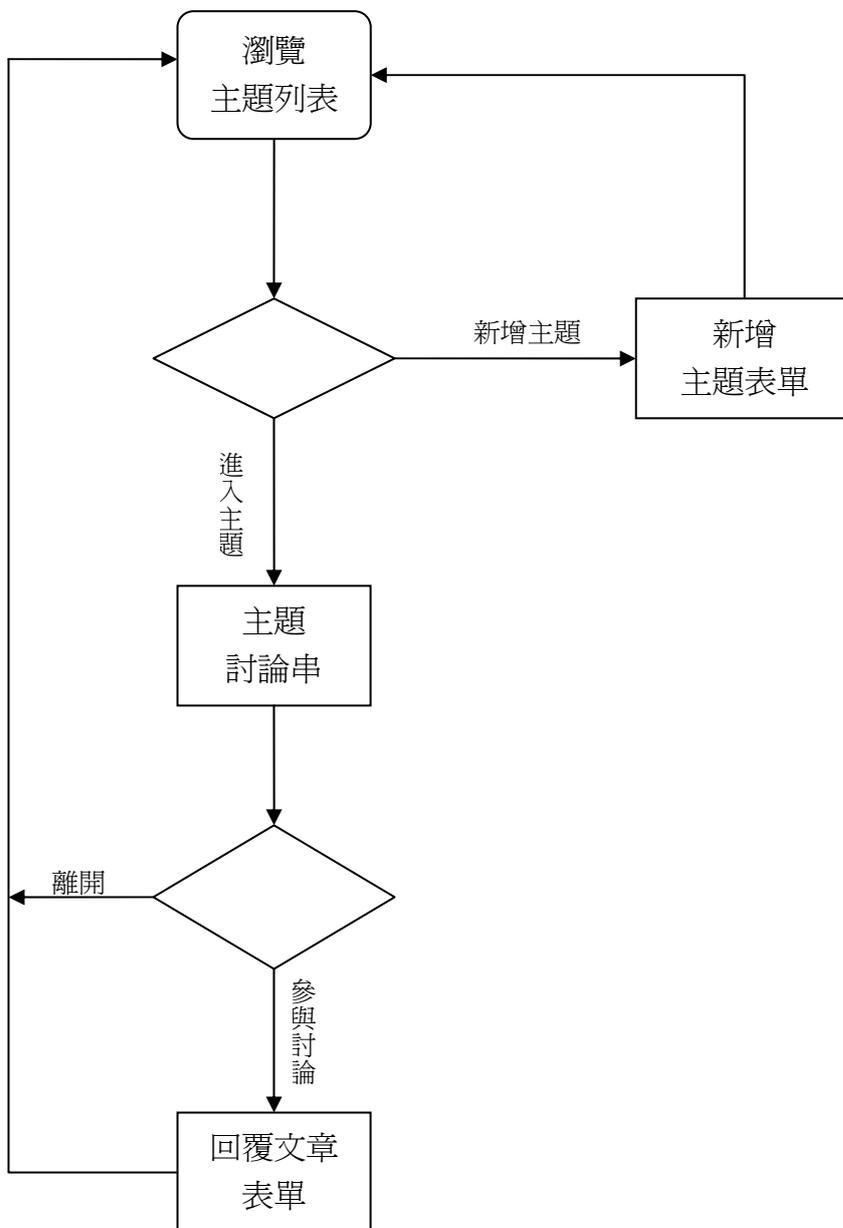


- ForumReply_Modify.php

功能同 ForumTopic_Modify.php，將回覆留言下所輸入的資料傳回資料庫。

- gConfig.php
資料庫設定檔。

7.2.3 程式流程



7.2.4 程式解說

- ForumTopic_Query.php

```

<?php
include("gConfig.php");
?>
<html>
<head>
<title>討論區</title>
</head>
<body>
<a href="ForumTopic_Form.php">新增主題</a>
<table border="1" cellspacing="0" cellpadding="4">
  <tr>
    <td>主題</td>
    <td>作者</td>
    <td>回覆篇數</td>
    <td>最後發表時間</td>
  </tr>
</table>
<?php
// 連線到資料庫
$mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],
    $GLOBALS["GSTR_MYSQL_USER"],
    $GLOBALS["GSTR_MYSQL_PWD"]);
// 選擇資料庫
mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);
// 選取 ForumTopic 全有資料
$mstrSQL = "SELECT TopicID, Subject, Author, " .
    " DATE_FORMAT>LastPostTime,\"% Y-%m-%d %H:%i:%s\"), ReplyNum
FROM ForumTopic " .
    " ORDER BY LastPostTime DESC ";
// 執行 SQL Statement
$mrsResult = mysql_query($mstrSQL,$mrsConn);
// 讀出資料
while(list($TopicID,$Subject,$Author,$LastPostTime,$ReplyNum) =
mysql_fetch_row($mrsResult))
{
    // 印出 Topic 內容

```

```

        echo "<tr>\n";
        echo "<td>";
        echo "<a href=\"ForumReply_Query.php?TopicID=" . $TopicID . "\"> .
$Subject . "</a>";
        echo "</td>";
        echo "<td>" . $Author . "</td>\n";
        echo "<td>" . $ReplyNum . "</td>\n";
        echo "<td>" . $LastPostTime . "</td>\n";
        echo "</tr>\n";
    }
    // 釋放資源
    mysql_free_result($mrsResult);
    // 關閉連線
    mysql_close($mrsConn);
?>
</table>
</body>
</html>

```

- ForumTopic_Form.php

```

<html>
<head>
<title>發表文章</title>
</head>
<body>
<form name="frmData" id="frmData" action="ForumTopic_Modify.php"
method="post" >
    作者
    <input type="text" name="Author" id="Author" size="20" maxlength="200"><br
/>
    e-mail
    <input type="text" name="Email" id="Email" size="20" maxlength="200"><br />
    主題
    <input type="text" name="Subject" id="Subject" size="20" maxlength="200"><br
/>
    文章<br />
    <textarea name="Message" id="Message" cols="45" rows="20" ></textarea>
    <br />

```

```

        <input type="submit" name="btnSubmit" id="btnSubmit" value="送出" >
        <input type="reset" name="btnReset" id="btnReset" value="重填">
    </form>
</body>
</html>

```

- ForumTopic_Modify.php

```

<?php
include("gConfig.php");
/**
 * 1.新增 ForumTopic
 * 2.選取 ForumTopic 中 TopicID 最大的一筆資料,即選取剛新增的 Topic
 * 3.新增資料 ForumReply
 */
// 取得前頁值
$Author = $HTTP_POST_VARS["Author"]; // 作者
$email = $HTTP_POST_VARS["Email"]; // e-mail
$Subject = $HTTP_POST_VARS["Subject"]; // 主題
$message = $HTTP_POST_VARS["Message"]; // 文章
// 連線到資料庫
$mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],
                        $GLOBALS["GSTR_MYSQL_USER"],
                        $GLOBALS["GSTR_MYSQL_PWD"]);
// 選擇資料庫
mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);
// 新增 ForumTopic
echo $mstrSQL = "INSERT INTO ForumTopic (Subject, Author, LastPostTime,
ReplyNum) " .
                " VALUES (" . $Subject . ", " . $Author . ", NOW(NULL), 0)";
// 執行新增 ForumTopic
mysql_query($mstrSQL,$mrsConn);
// 選取 ForumTopic 中 TopicID 最大的一筆資料,即選取剛新增的 Topic
$mstrSQL = "SELECT TopicID FROM ForumTopic ORDER BY TopicID DESC
LIMIT 1";
// 執行 SQL Statement
$mrsResult = mysql_query($mstrSQL,$mrsConn);
// 讀出 TopicID
list($TopicID) = mysql_fetch_row($mrsResult);

```

```

// 新增資料 ForumReply
$strSQL = "INSERT INTO ForumReply (Author,Email,Subject,Message,TopicID) " .
        " VALUES (" . $Author . ", " . $Email . ", " .
        " " . $Subject . ", " . $Message . ", " . $TopicID . ")";
// 執行 SQL Statement
mysql_query($strSQL,$mrsConn);
// 關閉連線
mysql_close($mrsConn);
?>
<script language="javascript" type="text/javascript">
<!--
        location.replace("ForumTopic_Query.php");

//-->
</script>

```

- ForumReply_Query.php

```

<?php
include("gConfig.php");

// 取得前頁值
$TopicID = $_HTTP_GET_VARS["TopicID"]; // TopicID

?>
<html>
<head>
<title>討論區</title>
</head>

<body>

<a href="ForumTopic_Query.php">回討論區</a>
<a href="ForumReply_Form.php?TopicID=<?=$TopicID?>">回覆</a>
<?php

// 連線到資料庫
$mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],

```

```

$GLOBALS["GSTR_MYSQL_USER"],
$GLOBALS["GSTR_MYSQL_PWD"]);

// 選擇資料庫
mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);

// 選取 ForumTopic 符合 TopicID 所有資料
$mstrSQL = "SELECT Subject, Message, "
    " DATE_FORMAT(PostTime, \"%Y-%m-%d %H:%i:%s\"), Email, Author
FROM ForumReply "
    " WHERE TopicID = " . $TopicID . " ORDER BY PostTime DESC ";

// 執行 SQL Statement
$mrsResult = mysql_query($mstrSQL,$mrsConn);

// 讀出資料
while(list($Subject,$Message,$PostTime,$Email,$Author) =
mysql_fetch_row($mrsResult))
{
    // 印出 Reply 內容
    echo "<table border=\"1\" cellspacing=\"0\" cellpadding=\"4\"
width=\"100%\">\n";

    echo "<tr><td>\n";
    echo "主題:" . $Subject . "<br />\n";

    echo "作者:<a href=\"mailto:\" . $Email . "\"> . $Author . "</a> <br />\n";
    echo "回覆時間:" . $PostTime . "<br />\n";
    echo "文章:<br />\n" . nl2br($Message) . "<br />\n";

    echo "</td>\n</tr>\n";

    echo "</table><br />\n";
}

// 釋放資源
mysql_free_result($mrsResult);

```

```
// 關閉連線
mysql_close($mrsConn);
?>
```

```
</body>
</html>
```

- ForumReply_Form.php

```
<?php
include("gConfig.php");

// 取得前頁值
$TopicID = $HTTP_GET_VARS["TopicID"]; // TopicID

// 連線到資料庫
$mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],
                        $GLOBALS["GSTR_MYSQL_USER"],
                        $GLOBALS["GSTR_MYSQL_PWD"]);

// 選擇資料庫
mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);

$mstrSQL = "SELECT Subject FROM forumtopic " .
           " WHERE TopicID=" . $TopicID . " ORDER BY LastPostTime DESC ";

// 執行 SQL Statement
$mrsResult = mysql_query($mstrSQL,$mrsConn);

// 讀出資料
list($Subject) = mysql_fetch_row($mrsResult);

// 釋放資源
mysql_free_result($mrsResult);

// 關閉連線
mysql_close($mrsConn);
```

?>

<html>

<head>

<title>回覆文章</title>

</head>

<body>

<form name="frmData" id="frmData" action="ForumReply_Modify.php"
method="post" >

作者

<input type="text" name="Author" id="Author" size="20" maxlength="200"><br
</>

e-mail

<input type="text" name="Email" id="Email" size="20" maxlength="200">

主題

<input type="text" name="Subject" id="Subject" size="20" maxlength="200"
value="re:<?=\$Subject?>">

文章

<textarea name="Message" id="Message" cols="45" rows="20" ></textarea>

<input type="hidden" name="TopicID" id="TopicID" value="<?=\$TopicID?>">

<input type="submit" name="btnSubmit" id="btnSubmit" value="送出" >

<input type="reset" name="btnReset" id="btnReset" value="重填">

</form>

</body>

</html>

- ForumReply_Modify.php

```

<?php
include("gConfig.php");

/**
 * 1.新增 ForumTopic
 * 2.選取 ForumTopic 中 TopicID 最大的一筆資料,即選取剛新增的 Topic
 * 3.新增資料 ForumReply
 */

// 取得前頁值
$Author = $HTTP_POST_VARS["Author"]; // 作者
$email = $HTTP_POST_VARS["Email"]; // e-mail
$Subject = $HTTP_POST_VARS["Subject"]; // 主題
$message = $HTTP_POST_VARS["Message"]; // 文章
$TopicID = $HTTP_POST_VARS["TopicID"]; // TopicID

// 連線到資料庫
$mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],
                        $GLOBALS["GSTR_MYSQL_USER"],
                        $GLOBALS["GSTR_MYSQL_PWD"]);

// 選擇資料庫
mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);

// 更新 ForumTopic
$mstrSQL = "UPDATE ForumTopic SET LastPostTime=now(),
ReplyNum=ReplyNum+1 " .
           " WHERE TopicID=" . $TopicID;

// 執行更新 ForumTopic
mysql_query($mstrSQL,$mrsConn);

// 新增資料 ForumReply
$mstrSQL = "INSERT INTO ForumReply (Author,Email,Subject,Message,TopicID) " .
           " VALUES (" . $Author . ", " . $Email . ", " .
           " " . $Subject . ", " . $Message . ", " . $TopicID . ")";

```

```
// 執行 SQL Statement
mysql_query($mstrSQL,$mrsConn);

// 關閉連線
mysql_close($mrsConn);

?>
<script language="javascript" type="text/javascript">
<!--

        location.replace("ForumReply_Query.php?TopicID=<?=$TopicID?>");

//-->
</script>
```

- gConfig.php

```
<?php

// MySQL 的主機名稱
$GSTR_MYSQL_HOST = "localhost";

// MySQL 的使用者帳號
$GSTR_MYSQL_USER = "root";

// MySQL 的使用者密碼
$GSTR_MYSQL_PWD = "";

// 花店資料庫
$GSTR_DB_FLORIST = "florist";

?
```

第八章 搜尋引擎

8.3 全文檢索

MySQL 到了 3.23.23 版時開始支援全文檢索及索引，而在系統中全文索引所使用的資料型別為 FULLTEXT。FULLTEXT 可由型別為 VARCHAR 或 TEXT 的欄位在建立表格的時候設定或是在完成之後使用 ALTER TABLE 或 CREATE INDEX 修改加入。注意當資料量大的時候使用 FULLTEXT index 會使用系統從表格中讀取資料的速度減慢。

我們使用 MATCH() 的函數來做全文檢索的動作。

```
mysql> CREATE TABLE articles (  
-> id INT UNSIGNED AUTO_INCREMENT NOT NULL PRIMARY KEY,  
-> title VARCHAR(200),  
-> body TEXT,  
-> FULLTEXT (title,body)  
-> );  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> INSERT INTO articles VALUES  
-> (0,'MySQL Tutorial', 'DBMS stands for DataBase ...'),  
-> (0,'How To Use MySQL Efficiently', 'After you went through a ...'),  
-> (0,'Optimising MySQL','In this tutorial we will show ...'),  
-> (0,'1001 MySQL Trick','1. Never run mysqld as root. 2. ...'),  
-> (0,'MySQL vs. YourSQL', 'In the following database comparison ...'),  
-> (0,'MySQL Security', 'When configured properly, MySQL ...');  
Query OK, 6 rows affected (0.00 sec)  
Records: 6 Duplicates: 0 Warnings: 0  
  
mysql> SELECT * FROM articles  
-> WHERE MATCH (title,body) AGAINST ('database');
```

```
+----+-----+-----+  
| id | title                | body                |  
+----+-----+-----+  
|  5 | MySQL vs. YourSQL    | In the following database comparison ... |  
|  1 | MySQL Tutorial       | DBMS stands for DataBase ...           |  
+----+-----+-----+  
2 rows in set (0.00 sec)
```

MATCH() 函數提供了自然語言搜尋比對的功能，系統會根據 AGAINST() 所提供的參數去比對資料庫中的每一筆資料並計算出其相似度給予一個值，這部份可以看之後的範例。而當我們把 MATCH() 函數放在 WHERE 之後使用時，結果系統會自動回報具有最高相關性的資料如上例。

相關值為一非負的數字，當其為零時表示兩者毫不相關。而相關值是依據總字數、不同字的數目及其包括的欲核對字所計算出來的。

系統也可以採用布林模式的搜尋，這個部份會在稍後作解釋。

之前的例子我們可以看到把 MATCH()放在 WHERE 之後的執行結果，而以下的例子則是系統把所有的相關值均計算出來，其結果當未排序。

```
mysql> SELECT id, MATCH (title,body) AGAINST ('Tutorial') FROM articles;
```

```
+----+-----+
| id  | MATCH (title,body) AGAINST ('Tutorial') |
+----+-----+
| 1   | 0.64840710366884                        |
| 2   | 0                                          |
| 3   | 0.66266459031789                        |
| 4   | 0                                          |
| 5   | 0                                          |
| 6   | 0                                          |
+----+-----+
6 rows in set (0.00 sec)
```

以下的例子更複雜了，我們查詢資料庫中資料的相似度，並依相似度值將其由大至小排序。在下面的查詢中我們用了兩次 MATCH()函數。但實際運作時並不會增加額外的負擔，因為系統本身會自行作最佳化的動作將兩者合併，一次執行完畢。

```
mysql> SELECT id, body, MATCH (title,body) AGAINST
-> ('Security implications of running MySQL as root') AS score
-> FROM articles WHERE MATCH (title,body) AGAINST
-> ('Security implications of running MySQL as root');
```

```
+----+-----+-----+
| id  | body                                     | score |
+----+-----+-----+
| 4   | 1. Never run mysqld as root. 2. ...     | 1.5055546709332 |
| 6   | When configured properly, MySQL ...    | 1.31140957288   |
+----+-----+-----+
```

2 rows in set (0.00 sec)

MySQL 使用非常簡單的方法把文章分割成字。”字”是由任何字母、數字、” ’ ”或”_”所組成的，其中系統會把 STOPWORD 清單及太短(三個以下的字母組成)的字排除掉。

每個合理的字都可以根據其和查詢字組的相似度計算出其權重。但如果一個字出現在很多文章中，那可能其在文章所包含的語義價值就會較低，故系統便會降低其權重，甚至到零。換句話說，假如那個字較少出現，就會有較高的權重。當有了字的權重之後，系統便可以依據這些資料計算出整筆資料的相關性。

這種方式相當適合於大量的資料。如果資料量大小的話，這種方法就無法正確的得到字的語義價值，故常常會出現異常的結果

```
mysql> SELECT * FROM articles WHERE MATCH (title,body) AGAINST ('MySQL');
Empty set (0.00 sec)
```

上述的例子，並未得到任何結果，其最主要的原因是由於”MySQL”這個字出現在

超過一半以的資料，故其權重幾乎為零。另外如”e_ectively”這個字系統也因為其幾乎沒有任何意義而把它加入 stopword 清單之中。當有半數以上的資料和一個字相關時，那這個字就無法有效判別那些是相關的資料，實際上這樣會很容易找到一堆無關緊要的資料。就如同我們常用的網際網路搜尋引擎它為了要避免這樣的情形也都自行建立其低語意的資料集，以提供更好的搜尋結果。

在 4.0.1 版之後，MySQL 也提供了布林型態的全文檢索，是使用 IN BOOLEAN MODE 的參數來設定，如下：

```

=====
mysql> SELECT * FROM articles WHERE MATCH (title,body)
-> AGAINST ('+MySQL -YourSQL' IN BOOLEAN MODE);
+----+-----+-----+
| id  | title                                     | body                                     |
+----+-----+-----+
| 1   | MySQL Tutorial                           | DBMS stands for DataBase ...           |
| 2   | How To Use MySQL Efficiently             | After you went through a ...           |
| 3   | Optimising MySQL                         | In this tutorial we will show ...      |
| 4   | 1001 MySQL Trick                         | 1. Never run mysqld as root. 2. ...    |
| 6   | MySQL Security                           | When configured properly, MySQL ...    |
+----+-----+-----+
=====

```

上例的查詢是找出資料中包含”MySQL”且不包含”YourSQL“(在這裡並未採用 50%的門檻)。布林型式的搜尋系統並不會把結果依相關性排序輸出。可以參考上例輸出的結果 - (其中相關性最高包含兩個”MySQL”的那一筆卻列在最後。布林型式的全文檢索可以在沒有定義為 FULLTEXT index 的欄位資料上作用，但是運作會比較慢。

以下我們介紹了一些和布林全文檢索相關的符號：

- + 前加號表示傳回有找到相關字的資料
- 前減號表示傳回沒有找到相關字的資料
- 預設值：當字之前並未包含加號或減號時，表示這個字是選擇性的，可有可不能有，但是如包含此字的話，這筆資料的相關值會增加。
- <> 這兩個運算子可以改變”字”對整個相關值的貢獻度。運算子”<”會減少貢獻度和運算子”>”會增加貢獻度，可以參考之後的例子。
- () 括號是用來定義子集合
- *
- " 表示引號內的文字必須同時連續的出現才符合條件。

以下是相關的例子：

apple banana

... 找出至少包含上述兩字之一的資料

+apple +juice

... 同時包含兩個字

+apple macintosh

... 包含“apple”，且當其也包含“macintosh”，其等級就較高

+apple -macintosh

... 包含 “apple”，且不包含 “macintosh”

+apple +(>pie <strudel)

... 包含“apple”和“pie”或是“apple”和“strudel”(任何順序皆可)，但是“apple pie”的等級比“apple strudel”來得高

apple*

... “apple”、“apples”、“applesauce”和“applet”皆可

“some words”

... 包括“some words of wisdom”,不包括“some noise words”.

全文檢索的限制

- MATCH()函數所採用的所有參數都必須在同一表格中有定義為 FULLTEXT index 上才能作用，除非是在布林模式下使用。
- MATCH() 函數所使用的欄位必須都是表格中定義為 FULLTEXT index 的欄位，除非是在布林模式下使用。
- AGAINST()是必須使用固定的字串



全文檢索也是一個利用資料庫的好地方。單純的想法，巷把所有字放在資料庫，一尋就好了。問題在，第幾頁、第幾行怎麼記錄呢??總不能回答”找到”就算數吧！

那我把一整頁當成一筆資料總行吧！這樣可以知道找到的字串是在第幾頁了。還有問題，剛好在分頁的地方的字串怎麼辦？

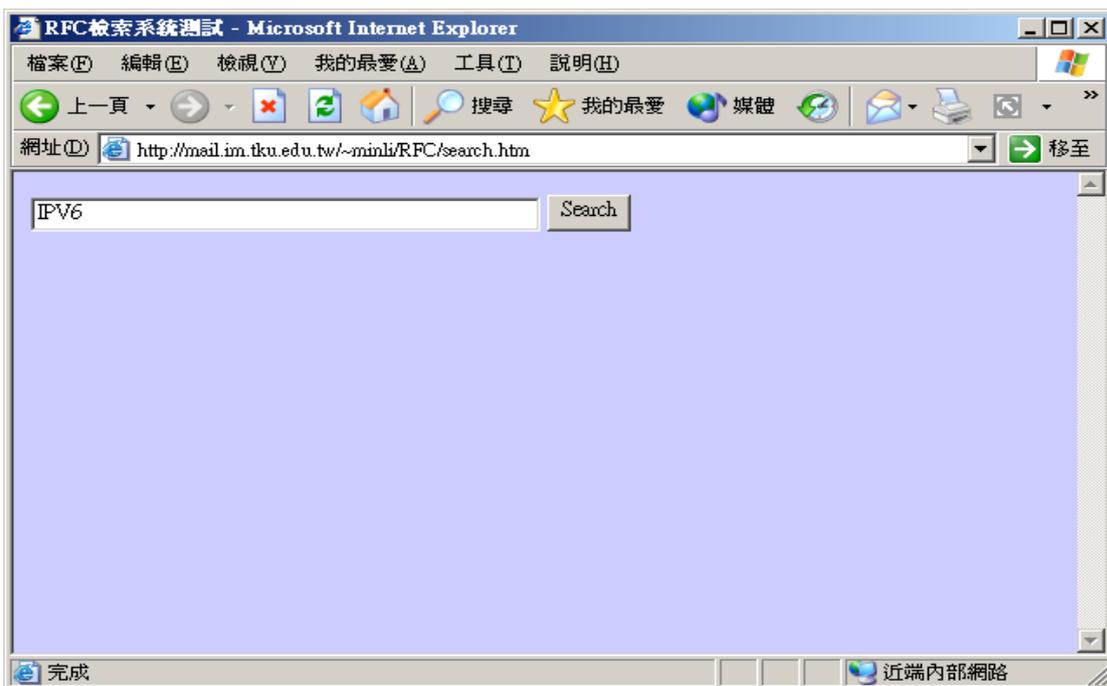
因此您要設計全文檢索，筆者膚淺的認應該考慮的就是要做到什麼程度。比方論文來索，也許找到在哪一篇論文中有了就可以了，而一本書的檢索，至少應該到某一章節。

有兩種方式可以做，跟計數器一樣，您可以採用檔案形式，或者資料庫形式。您可以自訂一種規則，讀到某個特殊字元／字串代表換行／換頁等等，然後把檔案一行一行讀進來。在這裡試著使用資料庫方式來簡略說明。

編號	章	節	段	內容
Id	Chapter	Section	Parag	Content
1	1	1	1	這裡為第一段的內容
1	1	1	2	這裡為第二段的內容

這樣只要一個 `select ... where content like '&$key%'` 就可以了。給你參考。

接下來為使用檔案方式來檢索尋找哪個檔案包所要找的 key point。(以 RFC 文件檢索為主)：



```

<?php
//此程式利用遞迴呼叫去找尋目錄下所有的 file 找出所要找的 key point.
echo "<p align=\"center\">";
echo "您尋找的關鍵字爲\"".$key."\"，以下是符合檢索的結果";
echo "</p><hr>";
if (function_exists("set_time_limit") && !get_cfg_var('safe_mode')){
    set_time_limit(600);
}

function get_msg($path) {
    global $key, $i;
    $handle = opendir($path);
    while ($filename = readdir($handle)) {
        //echo $path."/".$filename."<br>";
        $newpath = $path."/".$filename;
        if (is_file($newpath)) {
            $fp = fopen($newpath, "r");
            $msg = fread($fp, filesize($newpath));
            fclose($fp);
            match_show($key, $msg, $newpath, $filename);
        }
        if (is_dir($path."/".$filename) && ($filename != ".") && ($filename != ".."))
        {
            //echo "<BR><BR><BR>".$newpath."<BR><BR><BR>";
            get_msg($path."/".$filename);
        }
    }
    closedir($handle);
    return $i;
}

function match_show($key, $msg, $newpath, $filename) {
    global $i;
    $key = chop($key);
    if($key) {
        $check_type = preg_match("/\.\html?$/", $filename);
        If($check_type) {
            $title = getHtmlTitle($msg);
        }
    }
}

```

```

        $msg = preg_replace("/<style>.+</style>/is", "", $msg);
        $msg = preg_replace("/<[^>]+>/", "", $msg);
        $m = $filename;
        $i++;
        $link = $newpath;
        echo "$i.<a href=\"\$link\">$m</a><BR><BR>";
    }
}

function getHtmlTitle($msg) {

    /* Locate where <title> is located in html file. */
    $lBound = strpos($msg, '<title>') + 7;
    //7 is the length of <title>.

    if ($lBound < 1) return;

    /* Locate where </TITLE> is located in html file. */
    $uBound = strpos($msg, '</title>', $lBound);

    if ($uBound < $lBound) return;

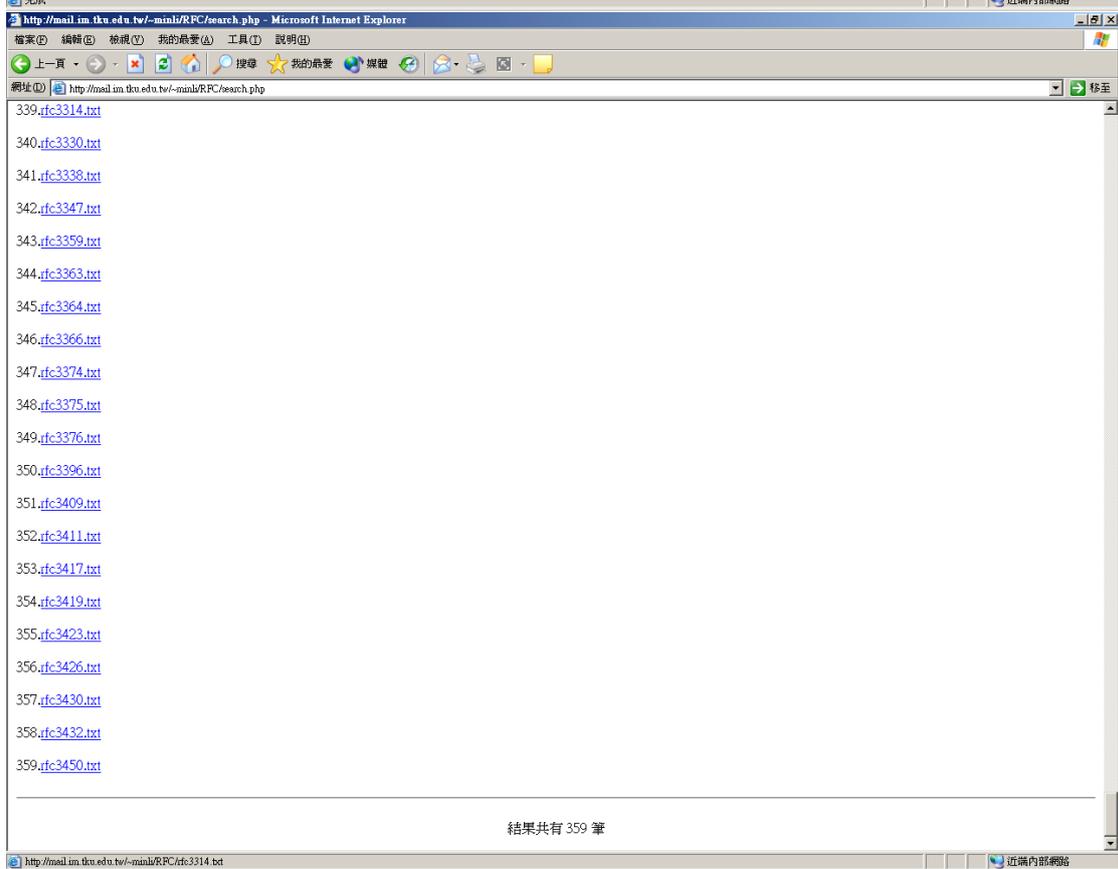
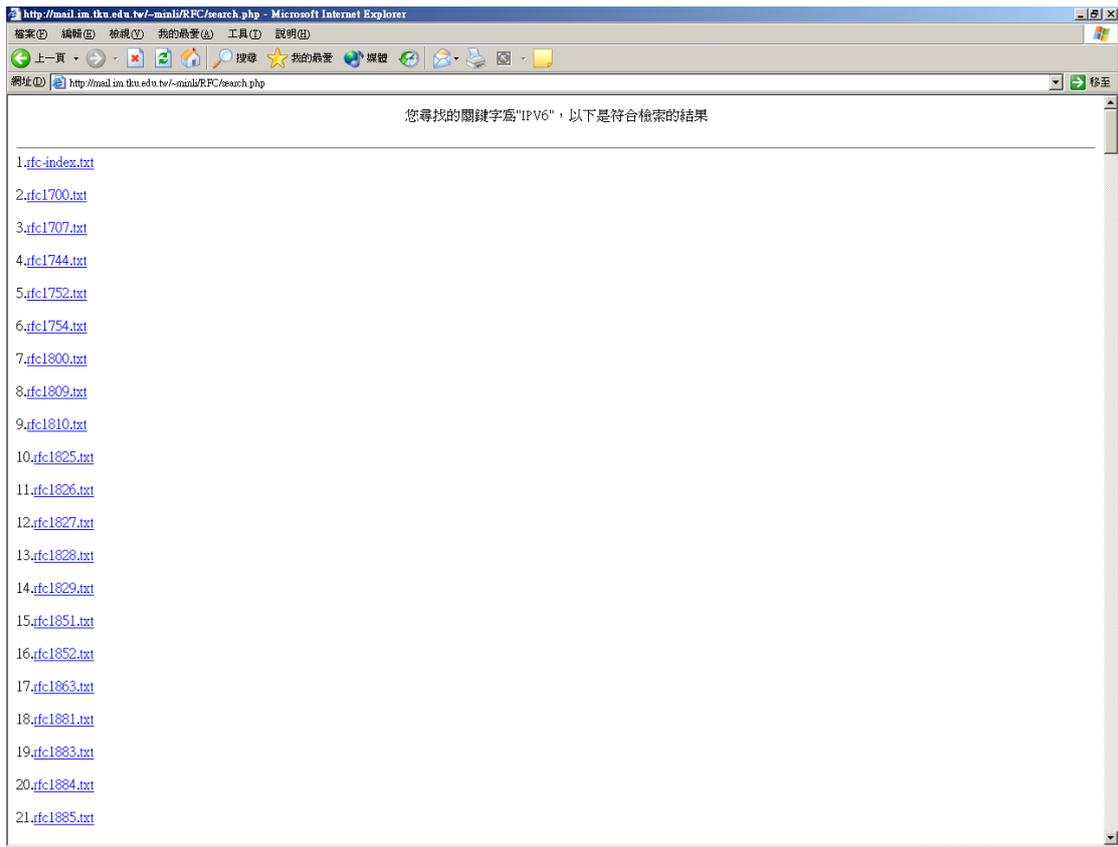
    /* Clean HTML and PHP tags out of $title with the madness below. */
    $title = ereg_replace("[\t\n\r]", "", substr($msg, $lBound, $uBound - $lBound));
    $title = trim(strip_tags($title));

    if (strlen($title) < 1) return; //A blank title is worthless.

    return $title;
}

$i = get_msg(".");
echo "<hr><p align=\"center\">";
echo "結果共有 $i 筆";
?>

```

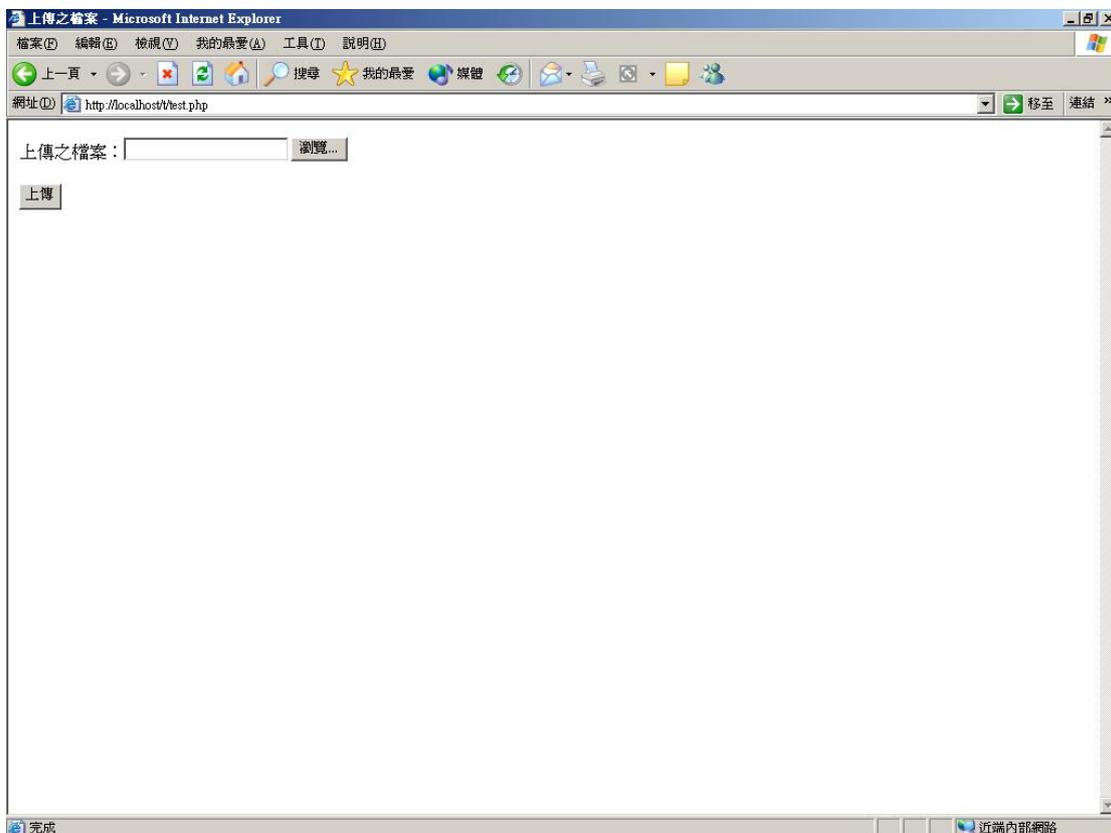


第九章 檔案上傳, 線上投票, 猜謎

9.1 檔案上傳

這個部份應該可以算是 PHP 世界中的基本功夫之一，只要經過些許的講解、說明，相信大家一定是很快就能學得會的。這個部份可以分成兩種模式來說：一是單一檔案的上傳，二是多個檔案的上傳。

首先，我們必須要動手設計一個表單（form），我所設計之一個簡單的表單畫面如下圖：



圖一 檔案上傳之表單畫面

當然，這個畫面的功能就只是單純的上傳檔案而已，此畫面是可以依個人的需求來做設計的。

如果將此畫面的功能再搭配上「認證」系統的話，那麼您就可以針對特定人士的帳號來進行控管了。如此一來的話，管理者就能較為輕鬆地來管理整個網站了，對於系統整體上的安全來說，也能更加地有保障。

接下來我們先來看看這個畫面的原始程式碼，程式碼如下：(單一檔案的上傳)

```

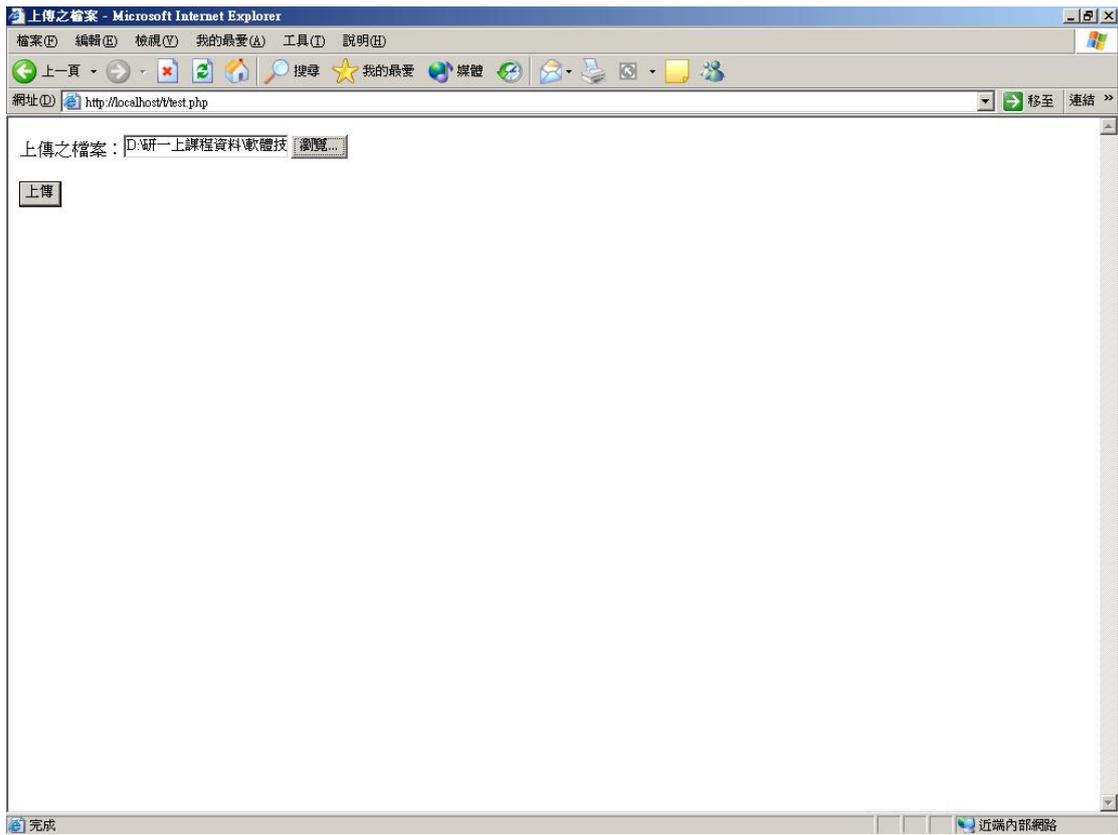
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=big5">
<title>上傳之檔案</title>
</head>
<body>
<form action="test.php" method="post" enctype="multipart/form-data">
    上傳之檔案：<input type="file" name=userfile size=20><p>
        <input type=submit value=上傳>
        <input type=hidden name=MAX_FILE_SIZE value=40960>
    </p>
</form>
<?php
    if($userfile!=null){
        copy($userfile,"D:\\$userfile_name");
        unlink($userfile);
        echo '上傳成功！';
    }
?>
</body>
</html>

```

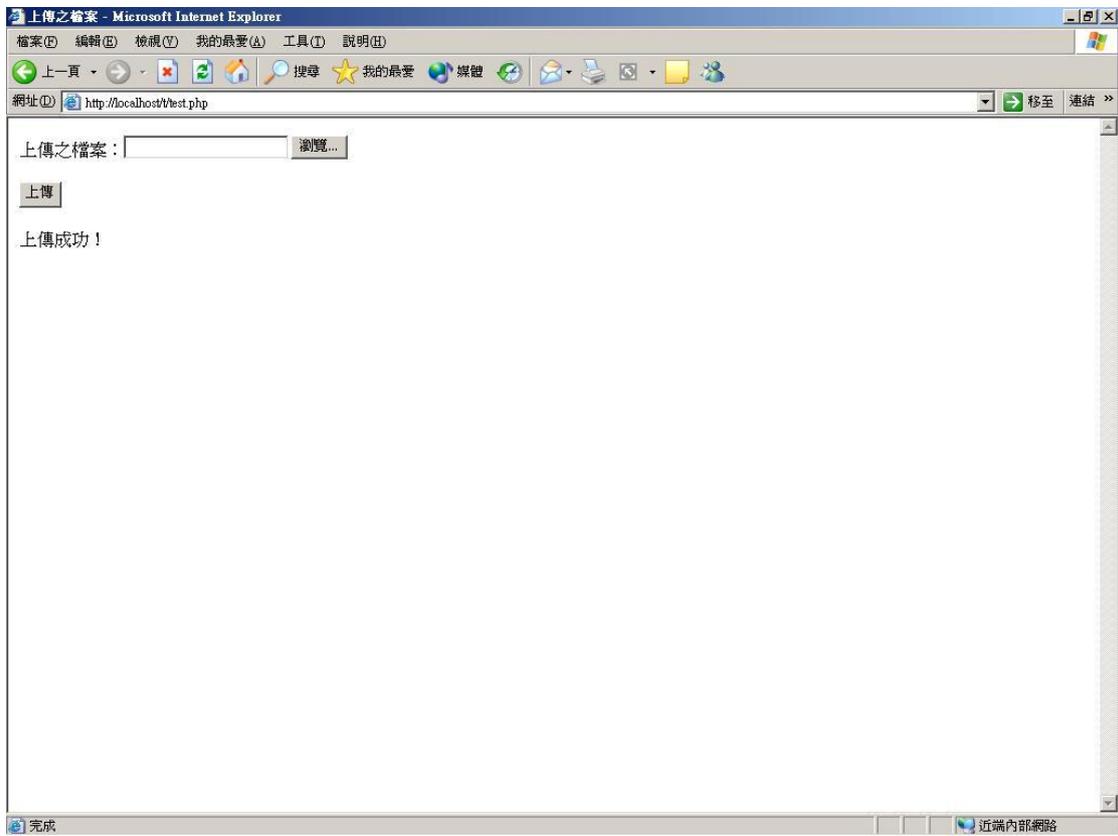
程式碼解析：

上列的程式碼之中，紅色的程式碼部份是表示比較須要注意的地方，而其中又以 `enctype="multipart/form-data"` 最為重要，是「必須」要寫上去的，不然這項功能將不會運作。在此提醒大家，`method` 記得要使用 `post` 的方式，至於 `post` 和 `get` 之間的不同之處的話，相信只要會寫網頁的人士就應該了解這二者其中的差別了。`MAX_FILE_SIZE` 的部份是指定能上傳之檔案的大小，單位是 `byte`。藍色的程式碼部份則是表示可以讓您自己來決定要設成什麼值的，`test.php` 是指您這支程式的檔案名稱，而 `D:\\$userfile_name` 則是指您要將此上傳之檔案置於你這台電腦的何處，可以設成絕對路徑，當然也可以設成相對路徑，我在於這個範例中是以絕對路徑來呈現的。

接下來我們來看看運作的畫面，畫面如下：（包含了選擇檔案（圖二）以及上傳成功之畫面（圖三））



圖二 檔案的選擇之畫面



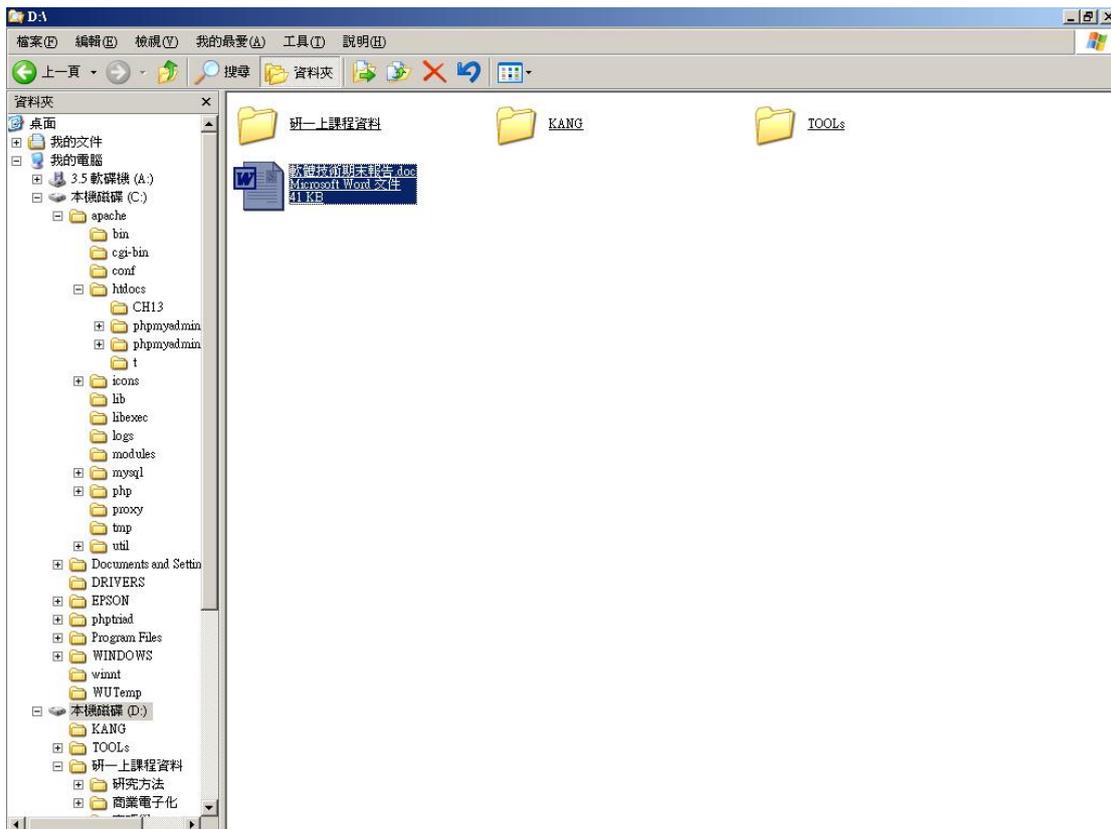
圖三 上傳成功之畫面

於圖二之中，你只要按下「瀏覽」的按鈕之後，就會跑出和 Windows 中選擇檔案的對話方塊是一模一樣的小視窗。「瀏覽」這個按鈕是在設定 input type 的時候，如果設成 file 的話，就會自動生成的。

之後，按下「上傳」按鈕之後，則會執行 PHP 的程式碼部份（如圖三）。「上傳」這個按鈕則是您自己來手動產生的。當檔案被上傳後，PHP 會將檔案放到暫時的目錄(upload_tmp_dir)下，並亂數取檔名，所以 \$userfile 存放的就是上傳「後」的檔案路徑與檔名。而檔案的「真實檔名」，也就是在使用者端原來的檔名（上傳「前」檔名），則會放在 \$userfile_name 變數裡，檔案大小放在 \$userfile_size 裡。這兩個變數名稱是相對於 \$userfile 的，也就是說，如果 form 的 input type = file 的欄位名稱 (name) 是 myfile，則上傳後的檔案路徑與檔名存放於 \$myfile，真實檔名與檔案大小存放於 \$myfile_name、\$myfile_size。

因為先前的動作都是屬於「暫時性」的，所以我們必須要進行一個“複製”的動作，如此才能將上傳之檔案保留於伺服器主機之中。必須要注意的是 \$userfile 是有包含路徑的，而 \$userfile_name 則是只有檔名而已！

接著，我們就來看看這個範例執行之後的結果吧，看看檔案是否有依照我們的設定出現於我們指定的位置中，如下圖所示：



圖四 檔案位置的畫面

如我們所預期的一樣，檔案出現在指定的路徑中了。

至於多個檔案上傳的話，就只是多了幾行程式碼而已，程式碼如下：（多個檔案的上傳）

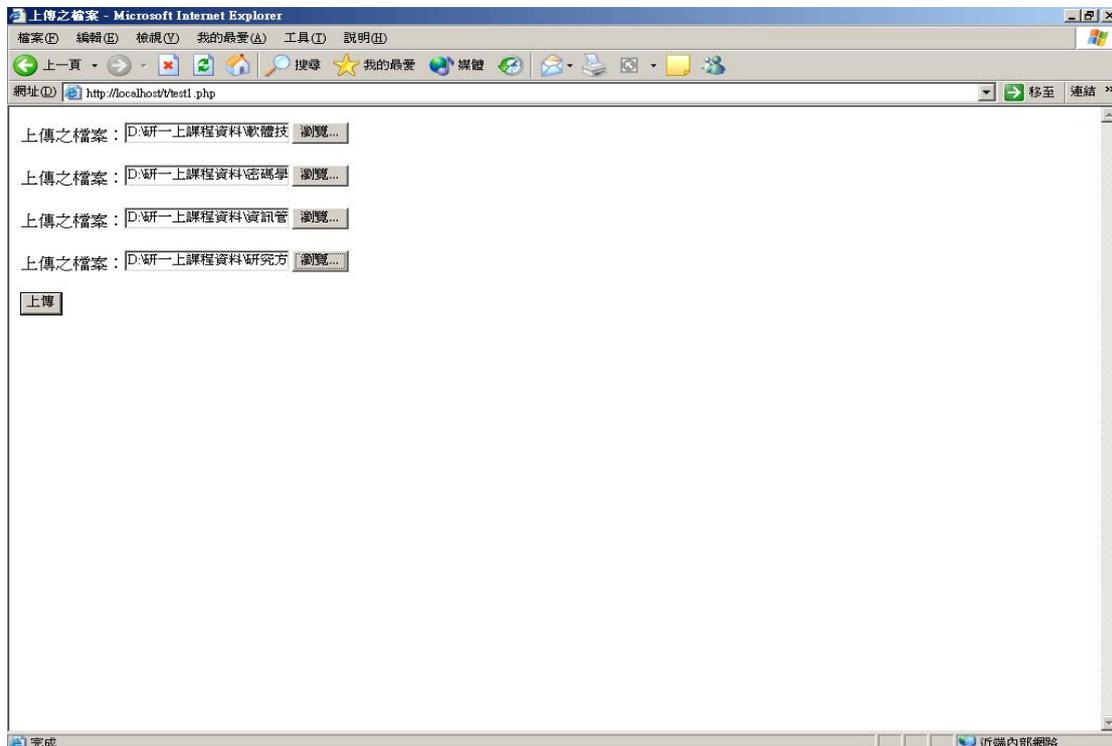
```
<form action="test1.php" method="post" enctype="multipart/form-data">
  上傳之檔案：<input type="file" name=userfile[] size=20><p>
  上傳之檔案：<input type="file" name=userfile[] size=20><p>
  上傳之檔案：<input type="file" name=userfile[] size=20><p>
  上傳之檔案：<input type="file" name=userfile[] size=20><p>
  <input type=submit value=上傳>
  <input type=hidden name=MAX_FILE_SIZE value=40960> </p>
</form>
```

程式碼解析：

上面的程式碼只比「單一檔案上傳」的程式碼多個幾行「上傳之檔案：<input type="file" name=userfile size=20><p>」而已。但要記得以「陣列」的形式來寫，即要在欄位名稱的命名上必須以「陣列」來呈現，也就是 name=userfile[] 。

執行此畫面之後，PHP 會以「陣列」的方式來接收，當然 \$myfile_name、\$myfile_size 也會以「陣列」的形式產生出來。

畫面如下：



圖五 多檔上傳的畫面

9.2 線上投票

9.2.1 簡介

提供網站線上投票功能，加強使用者與網站的互動，並可更加了解使用者的喜好，票選出最受歡迎的產品或活動。

9.2.2 程式架構

線上投票程式整體分為兩大部份，一為管者介面，一為使用者介面。整體架構強調動態處理，讓網站管理者有更大的彈性。

9.2.2.1 管理者介面

管理者介面主要為網站線上投票的主題及選項的維護。

9.2.2.1.1 投票主題維護

讓網站管理者可以自由新增網站目前的投票主題，並可設定投票主題為單選或複選。

9.2.2.1.1 投票選項維護

讓網站管理者可以自由對目前的投票主題新增投票選項。

9.2.2.2 使用者介面

投票選項選擇

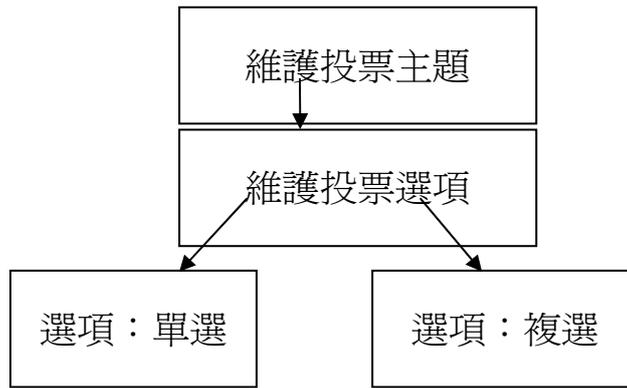
使用者可對目前網站開放的投票主題，選擇投票的選項。

投票結果觀看

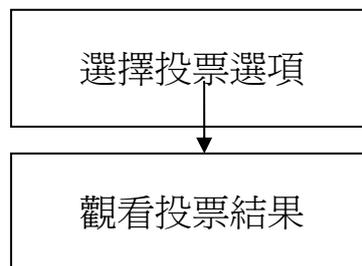
使用者可隨時觀看目前投票結果。

9.2.3 程式流程

9.2.3.1 管理者介面流程



9.2.3.2 使用者介面流程



9.2.4 表單規格

9.2.4.1 投票主題表單

表單名稱	vote_subject		說明
欄位名稱	型態	其他屬性	
<u>id</u>	tinyint(3)	UNSIGNED Not null auto_increment	投票主題主鍵
name	varchar(200)	Not null	主題名稱
type	char(1)	Not null	投票型態 M：多選 S：單選

9.2.4.2 投票選項表單

表單名稱	vote_item		說明
欄位名稱	型態	其他屬性	
vg_id	tinyint(3)	UNSIGNED Not null	主題名稱
<u>id</u>	int(10)	UNSIGNED Not null	選項主鍵

		auto_increment	
name	varchar(200)	Not null	選項名稱
count	int(10)	Not null	選項票數

9.2.5 程式解說

9.2.5.1 投票主題

9.2.5.1.1 投票主題畫面 (VoteSubject_Form.php)

```

<html>
<head>
<title>投票主題</title>
</head>
<body>
<form name="frmData" method="post" action="VoteSubject_Modify.php">

    名稱
    <input type="text" name="name" size="20" maxlength="200" />
    <br />

    類型
    <select name="type" >
        <option value="S">單選</option>
        <option value="M">多選</option>
    </select>
    <br />
    <input type="submit" name="btnSubmit" value="送出" />
    <input type="reset" name="btnReset" value="重填" />
</form>
</body>
</html>

```

9.2.5.1.2 投票主題處理 (VoteSubject_Modify.php)

```
<?php
include("gConfig.php");

// 取得前頁值
$mstrName = $HTTP_POST_VARS["name"]; // 群組名稱
$mstrType = $HTTP_POST_VARS["type"]; // 群組類型

// 連線到資料庫
$mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],
                        $GLOBALS["GSTR_MYSQL_USER"],
                        $GLOBALS["GSTR_MYSQL_PWD"]);

// 選擇資料庫
mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);

// 新增資料
$mstrSQL = "INSERT INTO vote_ subject (name,type) " .
          " VALUES (" . $mstrName . ", " . $mstrType . ")";

// 執行 SQL Statement
mysql_query($mstrSQL,$mrsConn);

// 關閉連線
mysql_close($mrsConn);

?>
```

```

<?php
include("gConfig.php");
?>

<html>
<head><title>投票項目</title></head>
<body>

<form name="frmData" method="post" action="VoteItem_Modify.php">
    群組名稱
    <select name="vg_id" >
<?php
    /**
     * 動態產生"群組名稱"
     */
    // 連線到資料庫
    $mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],
        $GLOBALS["GSTR_MYSQL_USER"],
        $GLOBALS["GSTR_MYSQL_PWD"]);
    // 選擇資料庫
    mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);

    // 選取 vote_subject 所有資料
    $mstrSQL = "SELECT id,name FROM vote_subject";

    $mrsResult = mysql_query($mstrSQL,$mrsConn);

    while(list($id,$name) = mysql_fetch_row($mrsResult))
    {
        echo "<option value=\"" . $id . "\">" . $name . "</option> \n";
    }

    // 釋放資源
    mysql_free_result($mrsResult);
    // 關閉連線
    mysql_close($mrsConn);
?>
</select><br />

```

9.2.5.2 投票選項

9.2.5.2.1 投票選項畫面 (VoteItem_Form.php)

9.2.5.2.1 投票選項處理 (VoteItem_Modify.php)

```
<?php
include("gConfig.php");

// 取得前頁值
$mstrVg_id = $HTTP_POST_VARS["vg_id"]; // 群組名稱
$mstrName = $HTTP_POST_VARS["name"]; // 項目名稱

// 連線到資料庫
$mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],
                        $GLOBALS["GSTR_MYSQL_USER"],
                        $GLOBALS["GSTR_MYSQL_PWD"]);

// 選擇資料庫
mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);

// 新增資料
$mstrSQL = "INSERT INTO vote_item (vg_id,name) " .
           " VALUES (" . $mstrVg_id . ", " . $mstrName . ")";

// 執行 SQL Statement
mysql_query($mstrSQL,$mrsConn);

// 關閉連線
mysql_close($mrsConn);
```

9.2.5.3 使用者投票

```

<?php
include("gConfig.php");
?>
<html>
<head><title>使用者投票</title></head>
<body>
<form name="frmData" method="post" action="Vote_Modify.php">
<?php
    // 連線到資料庫
    $mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],
        $GLOBALS["GSTR_MYSQL_USER"],
        $GLOBALS["GSTR_MYSQL_PWD"]);
    // 選擇資料庫
    mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);
    // 選取 vote_subject 資料的最後一筆
    $mstrSQL = "SELECT id AS vg_id,name AS vg_name,type AS vg_type " .
        " FROM vote_subject ORDER BY id DESC LIMIT 1";
    // 執行 SQL Statement
    $mrsResult = mysql_query($mstrSQL,$mrsConn);
    // 讀出投票主題的資料
    list($vg_id,$vg_name,$vg_type) = mysql_fetch_row($mrsResult);
    // 印出投票主題名稱
    echo $vg_name . "<br />";
    // 選取 "投票選項" 符合所選擇的 "主題 ID"
    $mstrSQL = "SELECT id,name FROM vote_item WHERE vg_id = " . $vg_id . " ";
    $mrsResult = mysql_query($mstrSQL,$mrsConn); // 執行 SQL Statement
    while(list($id,$name) = mysql_fetch_row($mrsResult)) // 讀出資料
    {
        if($vg_type == "M") // 選項為多選
        {
            $mstrInputType = "checkbox";
        }
        else if($vg_type == "S") // 選項為單選
        {
            $mstrInputType = "radio";
        }
        // 印出選項
        echo $name . "<input type=\"\" . $mstrInputType . \"\" " .
            " name=\"count[]\" value=\"\" . $id . \"\" /> <br /> \n";
    }
    mysql_free_result($mrsResult); // 釋放資源
    mysql_close($mrsConn); // 關閉連線
?>
<br /><input type="hidden" name="vg_id" value="<?=$vg_id?>" />
<input type="submit" name="btnSubmit" value="送出" />
<input type="reset" name="btnReset" value="重填" />
</form>

```

9.2.5.3.2 使用者投票畫面 (Vote_Modify.php)

```
<?php
include("gConfig.php");

// 取得前頁值
$mstrVg_id = $HTTP_POST_VARS["vg_id"]; // 群組 id
$mstrCountArr = $HTTP_POST_VARS["count"]; // 投票數

// 最得使用者勾選的項目
foreach($mstrCountArr as $k => $v)
{
    $mstrCount .= $v . ",";
}

// 去掉最後一個 ,
$mstrCount = substr($mstrCount,0,strlen($mstrCount)-1);

// 連線到資料庫
$mrsConn = mysql_connect($GLOBALS["GSTR_MYSQL_HOST"],
                        $GLOBALS["GSTR_MYSQL_USER"],
                        $GLOBALS["GSTR_MYSQL_PWD"]);

// 選擇資料庫
mysql_select_db($GLOBALS["GSTR_DB_FLORIST"],$mrsConn);

// 更新資料
$mstrSQL = "UPDATE vote_item SET count=count+1 WHERE id IN (" . $mstrCount . ")";

// 執行 SQL Statement
mysql_query($mstrSQL,$mrsConn);

// 關閉連線
mysql_close($mrsConn);

?>

<a href="Vote_Query.php">觀看結果</a>
```

9.2.5.3 投票票數 (VoteImage.php)

```
<?php

/**
 * 製造投票數的圖形
 * 參數: (主題 ID, 選項 ID, 投票數)
 * 圖形檔名: "主題 ID_選項 ID.png"
 */
function createVoteImage($vintVG_ID, $vintID, $vintCount)
{
    // 定義圖形
    $img = ImageCreate(10 * $vintCount, 20);

    // 定義顏色
    $color = ImageColorAllocate($img, 255, 0, 0);

    ImageFilledRectangle($img, 0, 0, 10 * $vintCount, 30, $color);

    // 輸出成 PNG 等格式
    ImagePNG($img, "vote" . $vintVG_ID . "_" . $vintID . ".png");

    // 釋放記憶體
    ImageDestroy($img);
}

?>
```

9.2.5.3 設定檔 (gConfig.php)

```
<?php

// MySQL 的主機名稱
$GSTR_MYSQL_HOST = "localhost";

// MySQL 的使用者帳號
$GSTR_MYSQL_USER = "root";

// MySQL 的使用者密碼
$GSTR_MYSQL_PWD = "123456";

// 花店資料庫
$GSTR_DB_FLORIST = "florist";

?>
```

9.3 猜謎

9.3.1 簡介

一般的商業網站會提供給會員一些折扣，有時候可以用抽獎方式，有時候則可採用一些網路遊戲來與會員之間做互動。猜謎是一個常見的網路遊戲，我們在這裡介紹簡單的猜謎程式，做為製作一個線上遊戲的雛型。

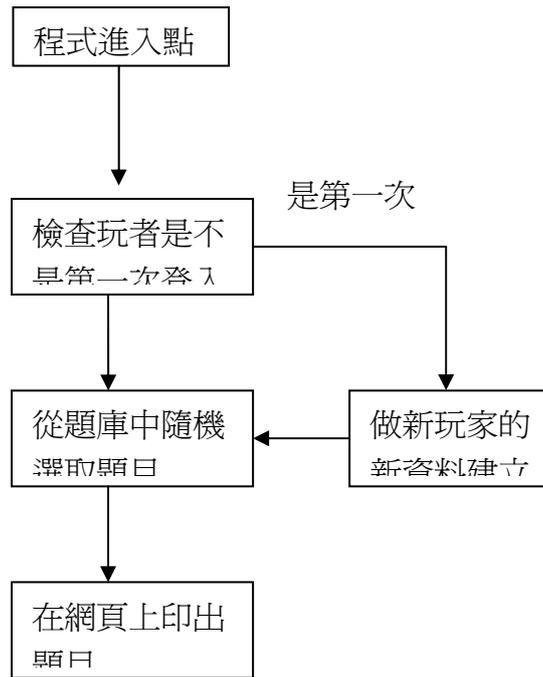
9.3.2 程式架構

我們簡單的製作了三隻程式和用到兩個 MySQL 的資料表：

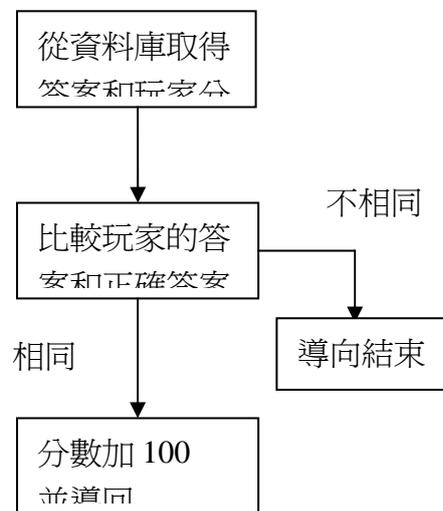
- **Guess_Form.php**：
這是我們的主程式，也是程式的進入點，本程式會隨機的自資料庫中抓取題目。
- **Guess_Modify.php**：
這隻程式是負責處理玩者選擇答案之後的動作。
- **Guess_End.php**：
這是結束的程式，一旦玩者答錯問題或放棄，就會被導到這個程式來，這個程式主要的動作是清除玩家的資料，並把分數秀出來。
- **Question Table**：
這是我們用來存放題庫的資料表，每一個題目固定會有四個選擇答案。
- **Player Table**：
玩家的基本資料表，用來暫存玩家的一些資料，一旦玩家結束遊戲，我們就會把資料清除。

9.3.3 程式流程

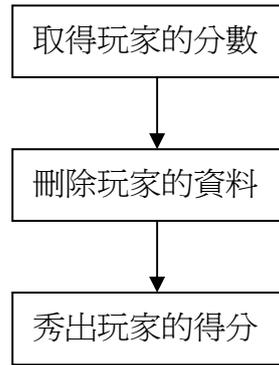
9.3.3.1 Guess_Form 流程



9.3.3.2 Guess_Modify 流程



9.3.3.3 Guess_Modify 流程



9.3.4 表單規格

9.3.4.1 題庫表單

表單名稱	question	型態	其他屬性	說明
<u>questionid</u>	tinyint(3)		UNSIGNED Not null auto_increment	題庫主鍵
question	char(128)			題目內容
ans1	char(32)			此題第一個答案
ans2	char(32)			此題第二個答案
ans3	char(32)			此題第三個答案
ans4	char(32)			此題第四個答案
realans	int(10)		UNSIGNED	此題正確答案

9.3.4.2 玩者表單

表單名稱	player	型態	其他屬性	說明
<u>playID</u>	char(32)			玩者唯一的 ID
<u>playName</u>	char(32)			玩者名字 (預留)
playScore	int(11)			玩者得分

9.3.5 程式解說

9.3.5.1 Guess_Form.php

```
<?

// 建立對 MySQL 的連結
$mrsConn = mysql_pconnect("localhost","","");

// 選擇目標資料庫
mysql_select_db("guess",$mrsConn);

// 判斷是否是第一次登入
if (empty($playerID))
{
    // 產生 unique id
    mt_srand((double)microtime()*1000000);
    $playerID = md5(uniqid(mt_rand()));
    setcookie("playerID",$playerID,time()+3600);
    $playerScore = 0;

    // 在 player 資料表建立新的 player 資料
    $mstrSQL = "INSERT INTO player (playerID, playerName,playerScore) " .
        " VALUES (' . $playerID . ',' . $playerName . ',' . $playerScore . ')";

    mysql_query ($mstrSQL, $mrsConn);

}
?>
<html>
<head>
    <title>猜謎範例</title>
    <meta http-equiv="Content-Type" content="text/html;CHARSET=big5">
</head>

<body>

<?

```

```

// 取亂數,for 隨機選題目
mt_srand((double)microtime()*1000000);

// 依亂數選擇題目的 SQL 語法
$mstrSQL = "SELECT questionid, question, ans1, ans2, ans3, ans4 " .
    " FROM question WHERE questionid=" . mt_rand(1,2);

$result = mysql_query($mstrSQL,$mrsConn);

list($questionid,$question,$ans1,$ans2,$ans3,$ans4) = mysql_fetch_row($result);

mysql_free_result($result);

// 取得目前這位玩家的得分
$mstrSQL = "SELECT playerScore FROM player WHERE playerID = " .
    $playerID."";

$result = mysql_query($mstrSQL, $mrsConn);

list($playerScore) = mysql_fetch_row($result);

mysql_free_result($result);
?>

<form method="POST" action="Guess_Modify.php">
    <p>
        問題<?echo($questionid." : ".$question);?></p>
    <p><input type="radio" value="V1" name="R1"><?echo($ans1);?></p>
    <p><input type="radio" value="V2" name="R1"><?echo($ans2);?></p>
    <p><input type="radio" value="V3" name="R1"><?echo($ans3);?></p>
    <p><input type="radio" value="V4" name="R1"><?echo($ans4);?></p>
    <input type="hidden" value="<?=$questionid;?>" name="questionid">
    <p>
        <input type="submit" value="確定" name="B1">
        <input type="button" value="放棄" name="B2"
onClick="location.href=('Guess_End.php');">
    </p>
    <p>目前累積分數：<? echo($playerScore);?></p>
</form>

```

</body>

</html>

9.3.5.2Guess_Modify.php

```
<script language="javascript" type="text/javascript">

<?

// 建立對 MySQL 的連結
$mrsConn = mysql_pconnect("localhost","","");

// 選擇目標資料庫
mysql_select_db("guess",$mrsConn);

// 找料出題庫中符合現在題目的答案
$mstrSQL = "SELECT realans FROM question " .
    "WHERE questionid=" . $_POST[questionid];

$result = mysql_query($mstrSQL,$mrsConn);

list($realans) = mysql_fetch_row($result);

mysql_free_result($result);

$realans = "V".$realans;

$playans = $_POST[R1];

// 取得目前這位玩家的得分
$mstrSQL = "SELECT playerScore FROM player WHERE playerID = " . $playerID . """;

$result = mysql_query($mstrSQL, $mrsConn);

list($playerScore) = mysql_fetch_row($result);

mysql_free_result($result);

// 判斷答案是否正確
if ($playans == $realans)
{
```

```

echo ("alert('恭禧你！！你答對了^^');");

// 如果答對就將成績加 100 並回存回資料庫
$playerScore += 100;

$mstrSQL = "REPLACE INTO player (playerID,playerScore) VALUES
('".$playerID."','".$playerScore.)";

mysql_query($mstrSQL, $mrsConn);

// 回到原程式
echo ("location.replace('Guess_Form.php');");
}
else
{
// 答錯就導向結束網頁
echo ("alert('Oh Oh 答錯了唷');");

echo("location.replace('Guess_End.php');");
}

?>

//location.href = "Guess_Form.php";

</script>

```

9.3.5.3 Guess_End.php

```
<?

// 建立對 MySQL 的連結
$mrsConn = mysql_pconnect("localhost","","");

// 選擇目標資料庫
mysql_select_db("guess",$mrsConn);

// 取得目前玩家的分數
$mstrSQL = "SELECT playerScore FROM player WHERE playerID = " . $playerID.
"";

$result = mysql_query($mstrSQL, $mrsConn);

list($playerScore) = mysql_fetch_row($result);

mysql_free_result($result);

// 刪除玩家的在資料庫中的資料
$mstrSQL = "DELETE FROM player WHERE playerID = " . $playerID . """;

mysql_query($mstrSQL, $mrsConn);

// 刪除玩家的 cookie
setcookie("playerID");
?>
<html>
<head>
    <title>猜謎結束</title>
    <meta http-equiv="Content-Type" content="text/html;CHARSET=big5">
</head>

<body>
你所得到的總分是：
<?echo ($playerScore);?>
<a href="Guess_Form.php">再玩一次</a>
</body>
```

</html>

第十章 購物車

10.1 基本商品資料

我們將在這裡建立整個應用網站的，主體包括基本商品資料庫以供訪客選購。

規劃商品資料庫 tables

在先前 florist 資料庫儲存花束商品的資料。首先考慮分類：情人、祝賀、生日、結婚四大類，用 style 來儲存。

```
create table style(  
    styleid tinyINT unsigned not null auto_increment,  
    stylename CHAR(20),  
    primary key (styleid)  
);
```

有三個欄位，分別為 styleid、stylename 和 primary

在建立花束資料的 tables

```
create table flowers(  
    flowered INT unsigned not null auto_increment,  
    styleid tinyINT unsigned not null,  
    flowername CHAR(20),  
    material CHAR(60),  
    price smallINT unsigned,  
    picture CHAR(20),  
    primary key (flowerid),  
    index (styleid),  
);
```

有六個欄位，分別為 flowered ,styleid ,flowername ,material , price ,picture ,主鍵是 flowerid。

簡要 MYSQL 資料庫列表

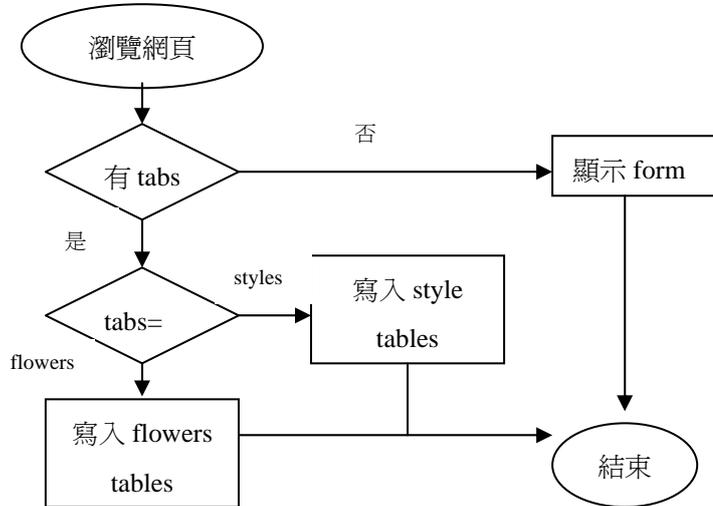
```
C:\mysql\bin>mysqldump -u root florist
# MySQL dump 8.14
#
# Host: localhost    Database: florist
#-----# Server
version          3.23.47-max-debug
#
# Table structure for table 'flowers'
#
CREATE TABLE flowers
(
                                                    flowerid
int(10) unsigned NOT NULL auto_increment,
styleid tinyint(3) unsigned NOT NULL default '0',
flowername varchar(20) default NULL,
material varchar(60) default NULL,
price smallint(5) unsigned default NULL,
picture varchar(20) default NULL,
PRIMARY KEY (flowerid),
KEY styleid
(styleid)
)
TYPE=MyISAM;
#
# Dumping data for table 'flowers'
#
INSERT INTO flowers VALUES (1,1,'天長地久','天堂鳥、玫瑰',500,'pic01.jpg');
INSERT INTO flowers VALUES (2,1,'真愛一世','薔薇、文竹',800,'pic02.jpg');
INSERT INTO flowers VALUES (3,1,'嫁給我吧','玫瑰、滿天星',1000,'pic03.jpg');
INSERT INTO flowers VALUES (4,2,'金榜題名','火鶴、鬱金香',1200,'pic04.jpg');
INSERT INTO flowers VALUES (5,2,'門庭若市','水仙',1500,'pic05.jpg');
INSERT INTO flowers VALUES (6,2,'平步青雲','桔梗',1700,'pic06.jpg');
INSERT INTO flowers VALUES (7,3,'白頭偕老','玫瑰、向日葵',700,'pic07.jpg');
INSERT INTO flowers VALUES (8,3,'百年好合','百合、玫瑰',1100,'pic08.jpg');
INSERT INTO flowers VALUES (9,4,'壽比南山','蘭花',800,'pic09.jpg');
INSERT INTO flowers VALUES (10,4,'生日快樂','小雛菊',500,'pic10.jpg');
#
# Table structure for table 'shopping'
#
```

```
CREATE TABLE shopping
(
char(32) NOT NULL default "",
flowerid int(10) unsigned default NULL,
amount tinyint(3) unsigned default '1',
shoppingtime datetime default NULL,
KEY shpid
(shoppingid)
TYPE=MyISAM;
#
```

10.2 花束資料的建檔、新增

利用 form 製作輸入類別與花束

流程圖：



mysql_connect: 開啓 MySQL 伺服器連線。

mysql_create_db: 建立一個 MySQL 新資料庫。

mysql_data_seek: 移動內部傳回指標。

mysql_db_query: 送查詢字串 (query) 到 MySQL 資料庫。

mysql_error: 傳回錯誤訊息。

mysql_fetch_array: 傳回陣列資料。

mysql_fetch_field: 取得欄位資訊。

mysql_fetch_lengths: 傳回單列各欄資料最大長度。

mysql_free_result: 釋放傳回占用記憶體。

mysql_insert_id: 傳回最後一次使用 INSERT 指令的 ID。

mysql_pconnect: 開啓 MySQL 伺服器長期連線。

mysql_query: 送出一個 query 字串。

mysql_select_db: 選擇一個資料庫。

mysql_tablename: 取得資料表名稱

PHP 程式碼

```
<html>

<head>
    <title>花束資料輸入</title>
</head>
<body>
<?
// 連線到資料庫
$linkptr = mysql_pconnect("localhost","florist","123456");
mysql_select_db("florist",$linkptr);
//選擇資料庫

// 判斷是否已有資料輸入當$tabs 是 empty< --表示沒有
if (empty($tabs)) :?>

<!-- 這裡是輸入畫面的程式片段 -->
<!-- 第一個 form，輸入類別 -->
<form action="demo7-1.phtml" method="POST">
    <!-- 告訴後面這是要到 style 的 -->
    <input type="Hidden" name="tabs" value="styles">
    請輸入類別：<input type="Text" name="fstyle">
    <input type="Submit" name="submit" value="新增類別">
</form>
<hr>
<!-- 第二個 form，輸入花束資料 -->
<form action="demo7-1.phtml" method="POST">
    <!-- 告訴後面這是要到 flowers 的 -->
    <input type="Hidden" name="tabs" value="flowers">
    請輸入花名：<input type="Text" name="fname"><br>
    請選擇種類：<select name="fstyle"><option value="愛情">愛情<option
value="祝賀">祝賀<option value="結婚">結婚<option value="生日">生日
</select><br>
    請輸入花材：<input type="Text" name="material"><br>
    請輸入價位：<input type="Text" name="price"><br>
    請輸入圖片名稱：<input type="Text" name="pict"><br>
    <input type="Submit" name="submit" value="新增花束">
</form>
```

```

<?
// 判斷是否已有資料輸入當$tabs 不是 empty< --表示有
else : switch ($tabs) :
    case "styles" :
        // 這是新增類別的輸入部分
        $querystring = "insert into style (stylename) values ('$fstyle')";
        echo $querystring ;
        mysql_query($querystring,$linkptr) ;
        break ;
    case "flowers" :
        // 這是新增花束的輸入部分
        $result = mysql_query("select styleid from style where
stylename='$fstyle'", $linkptr) ;
// 一個 MySQL 的查詢語言
        list($styleid) = mysql_fetch_row($result) ; // 先找 style id
// 輸入一筆 MySQL 的資料
        $querystring = "insert into flowers
(styleid,flowername,material,price,picture)

                values ($styleid,'$fname','$material','$price','$pict')";
        mysql_query($querystring,$linkptr) ;
    endswitch ;
// 輸入之後做一個 link 回輸入畫面
echo "<a href=\"demo7-1.phtml\">下一筆? </a>\n";
endif ;
?>
</body>
</html>

```

附錄

單元一.前言

chap1.

陳世偉

單元二.PHP 簡介

chap3. 顧承憲, 簡嘉德

單元三.MYSQL 簡介

chap4.王峻鋒,黃志偉

單元四.電子商務應用實例介紹

chap5 黃彥勳, 黃炳翔

chap6 原孝任, 陳怡良

chap7 王銓成, 陳妍伶, 林東毅

chap8 梁亦銘, 廖明禮

chap9 謝明釗, 林宏澤, 林世唐

chap10 陳彥文, 黃雅鳳

資料搜集與編排

王維君, 周雋偉

總整理 王維君