



PHP+MySQL

互動式網頁程式設計班

[第六課] MySQL 與資料庫入門、管理、
維護資料庫及 SQL 操作語法

國立台灣大學 資訊工程學系暨研究所
資訊系統訓練班
講師：烏明學

章節目錄

- 第一部份 資料庫系統簡介
- 第二部份 MySQL資料庫
- 第三部份 SQL 語言
- 第四部份 PHP + mySQL



◦ 第一部份 資料庫系統簡介

資料庫

- 資料 (data) : 表達事實的語言或符號
- 資訊 (information) : 經過整理和組織的資料稱為資訊
- 資料庫 (database) : 將資訊以**一定的結構**存放在電腦中

資料模型

- 用來表示資料庫如何組成的架構，稱為資料模型
- 關於資料模型的理論不少，其中最著名的是「**關聯式資料模型**」(relational model of data)
 - Dr. E. F. Codd, "A Relational Model of Data for Large Shared Data Banks", 1970

關連式資料模型

- 何謂「關連式資料模型」：
 - 使用者看到的都是**表格**
 - 利用表格來呈現資料，
 - 將表格視為集合來進行處理
 - 當要操作資料時，便是針對表格去執行以
集合理論為基礎的數學運算，而其**執行結果還是表格**

表格

- 下圖是一個「表格」(table)，其中縱向的稱為「行」(column)，或是稱為「欄」(field)，存放著相同性質的資料。橫向的稱為「列」(row)，或是「記錄」(record)，裡頭包含許多不同性質的資料項目。這個表格有 4 欄 3 列

**table
(relation)**

column or field
(attribute)

SN	SName	Phone	Address
75312	Chen	04-22876543	台中市
75524	Chuang	07-7513578	高雄市
75302	Lee	04-6384321	高雄縣

number of columns
(degree)

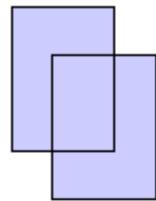
↑ number of rows
(cardinality) ↓

row or record
(tuple)

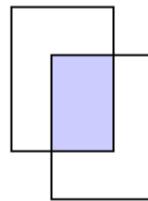
- 圖中黑色名詞與紅色名詞的意義相近，雖然在大多數的場合它們也被視為同義詞，但後者的定義較為嚴謹，才是符合數學理論的正式名稱

八種資料庫的運算

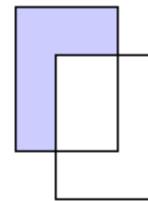
Union



Intersect



Difference



Product

A diagram illustrating the Product operation. It shows two vertical lists of tuples. The left list has three rows labeled 'a', 'b', and 'c'. The right list has four rows labeled 'x', 'y', 'x', and 'y'. Arrows point from each row in the first list to each row in the second list, indicating the Cartesian product.

a	x
b	y
c	

a	x
a	y
b	x
b	y
c	x
c	y

Restrict

A diagram illustrating the Restrict operation. It shows a large table with several rows, and a smaller subset of rows highlighted in light blue, representing a query restriction.

Project

A diagram illustrating the Project operation. It shows a large table with multiple columns, and a subset of columns highlighted in light blue, representing a projection.

(natural) Join

A diagram illustrating the (natural) Join operation. It shows two tables, A and B, being joined. Table A has columns a1, b1, a2, b1, a3, b2. Table B has columns b1, c1, b2, c2, b3, c3. The joined table C has columns a1, b1, c1, a2, b1, c2, a3, b2, c3.

a1	b1
a2	b1
a3	b2

b1	c1
b2	c2
b3	c3

a1	b1	c1
a2	b1	c2
a3	b2	c3

Divide

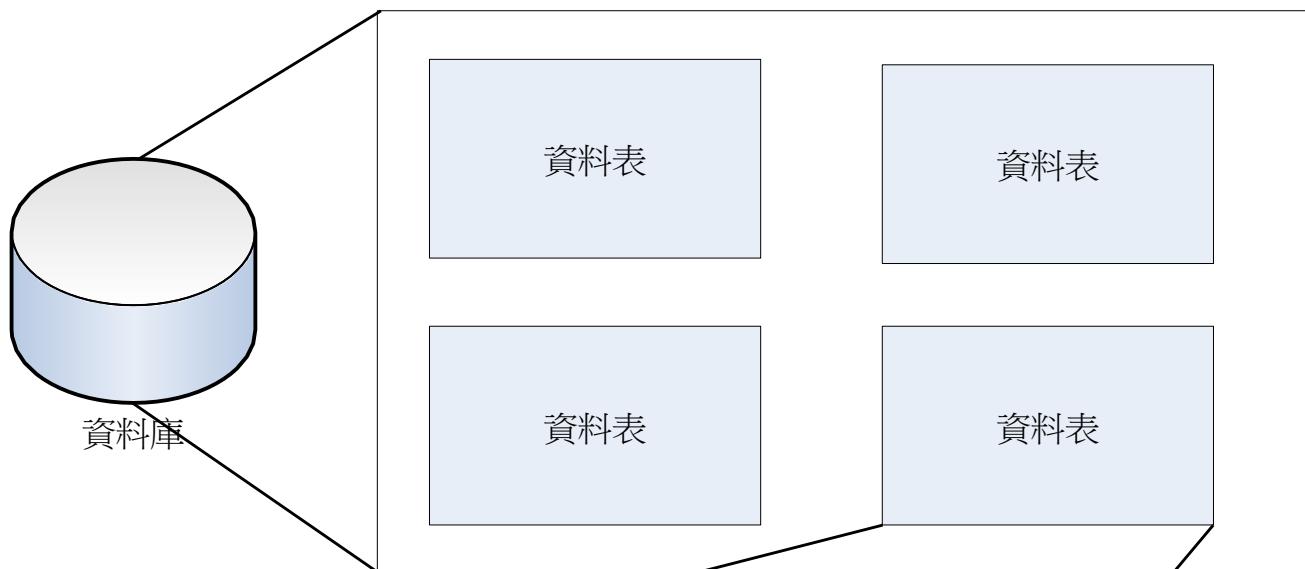
A diagram illustrating the Divide operation. It shows a large table with multiple columns being divided into three separate components: a, x, and z.

a
x
z

資料庫管理系統

- 資料庫管理系統 (Database Management System)：
負責管理資料的軟體
- 常見的資料庫系統：
 - 付費：
 - Microsoft SQL Server (適合中/大型資料庫)
 - Oracle (適合中/大型資料庫)
 - IBM DB2 (適合中/大型資料庫)
 - Access (不能稱為完整的資料庫，只適合用在單機或小型資料庫)
 - 免費：
 - mySQL (適合中/小型資料庫)
 - SQLite (適合中/小型資料庫)

關聯式資料庫的架構 (1)



欄位	欄位	欄位	
資料庫	資料庫	資料庫	紀錄
資料庫	資料庫	資料庫	紀錄
資料庫	資料庫	資料庫	紀錄
資料庫	資料庫	資料庫	紀錄
資料庫	資料庫	資料庫	紀錄
資料庫	資料庫	資料庫	紀錄

關聯式資料庫的架構 (2)

- 一個資料庫伺服器 (如 mySQL Server)
 - 可以擁有 n 個資料庫 (databases)
 - 每個資料庫可以擁有 n 個表格 (tables)
 - 每個表格可以擁有 n 個欄位 (fields)
- 資料庫的優點：
 - 尋找資料快
 - 排序速度快
 - 原因？
 - 索引！

索引 (1)

- 什麼是索引？
 - 在下表中找出學號為「75120」者的成績

75312	Chen	80
75524	Chuang	95
75207	Yeh	92
75302	Lee	90
75101	Chuang	89
75303	Ho	90
75120	Lin	92
75313	Chen	88

- 從表格的開頭逐筆尋找，將在第 7 筆的位置發現目標，也就是總共找了 7 次

索引(2)

- 若針對「學號」這一欄建立「索引」，則會像以下這個樣子：
 - 除了右側的原始表格之外，還會多出一個如左邊這樣的 index table；在 index table 中，學號已經被排序過了，但它仍記錄著每個學號與原始表格之間的對應關係

75101		
75120		
75207		
75302		
75303		
75312		
75313		
75524		

75312	Chen	80
75524	Chuang	95
75207	Yeh	92
75302	Lee	90
75101	Chuang	89
75303	Ho	90
75120	Lin	92
75313	Chen	88

索引 (3)

- 同樣要找出學號為「75120」者的成績，有了 index 之後將有些不同，我們改從 index table 中著手
 - 由於 index table 中的資料已經被排序過了，因此可以採用「二元搜尋法」
 - 8 筆資料取半數，直接看 index table 的第 4 筆記錄（75302）我們的目標（75120）比它小，所以再取 4 的半數，看看第 2 筆記錄（75120），結果就找到了。整個過程中，我們只找了 2 次
 - 但也不見得每次都會比較快，若是要找「75312」的話，在沒有 index 的情況下，反而可以一次命中目標

索引 (4)

- pros and cons
 - 是否將每個資料欄位都設成索引，這樣最快嗎？索引仍然有缺點：
 - 需要更多資料儲存的空間
 - 本來只有一個原始表格，當我們將「學號」設為 index 之後，就多了一個 index table
 - 若再將「姓名」設為 index 的話，又會多出一個 index table 來
 - 增加資料異動所需的時間
 - 在原始表格中加入一位新學生的資料時，index table 就必須掃瞄一次原始表格，更新 index table

那麼誰才該做成索引呢

- 一個資料表至少應該有一個欄位被做為索引，通常是**主鍵 (primary key)**
- 例如下表：

身分證字號	班級	座號	姓名	成績
T120123456	3-1	1	陳小華	89
E120654321	3-1	2	張小文	90
T120989898	3-1	3	林小明	78
...
S220567890	3-2	1	葉小花	95

- 身分證字號是最適合做主鍵的欄位
 - 具辨識性及唯一性



◦ 第二部份 MySQL資料庫

SQL (Structured Query Language)

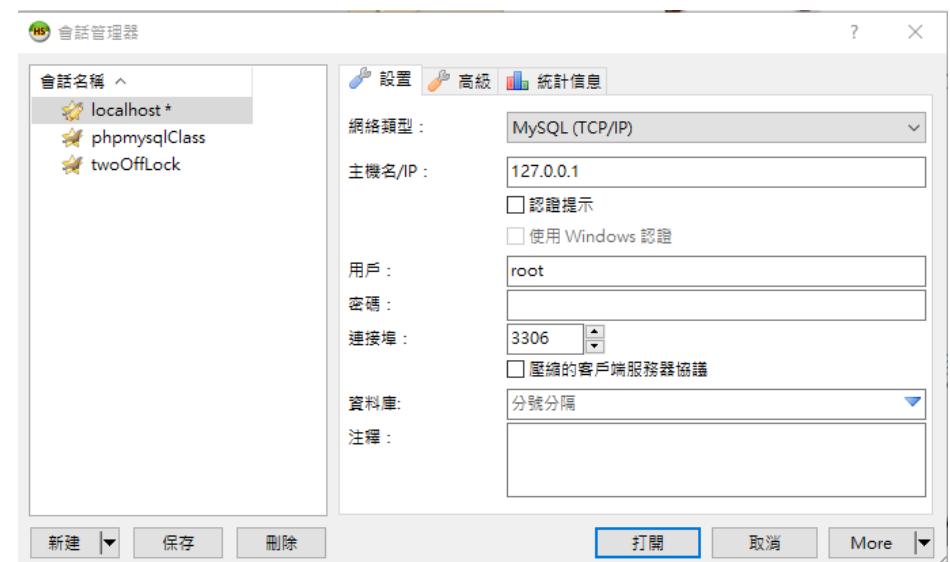
- 結構化查詢語言
 - 與高階程式語言相比，要來得更接近人類在使用的自然語言
 - 專門用於關連式資料庫的一種查詢語言，功能有：
 - 定義資料庫結構、建立資料表
 - 抓出存在資料表中的資料，排序或篩選資料
 - 新增、修改、刪除資料

MySQL = SQL ?

- 實際上大部份的關聯資料庫雖然都是參照 SQL-92 標準所發展出來，但其中支援的語法不盡然完全相同
 - Oracle 中的 SQL 語言稱做 PL/SQL
 - MS SQL Server 中稱做 Transact-SQL
- 因此以下將介紹部份 MySQL 所支援的 SQL 敘述，其中有些用法可能不適用於其它的資料庫

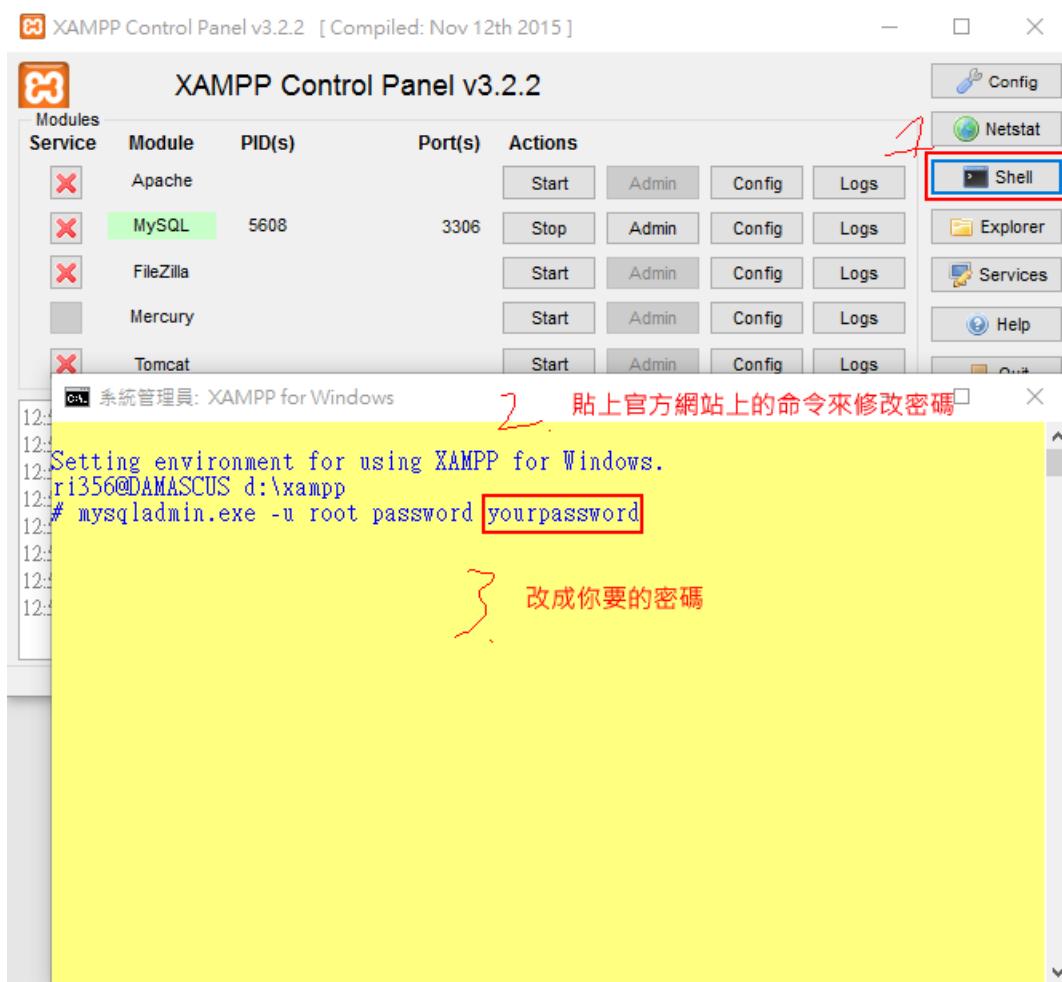
MySQL 權限認證

- xampp 預設所使用的帳號密碼：
 - 帳號: root
 - 密碼:
- 教室的資料庫帳號密碼
 - 帳號: root
 - 密碼: student



Root 帳號擁有最大權限

- 但預設的xampp卻沒有密碼
官方網站上說明之更改方式



HeidiSQL 畫面

The screenshot shows the HeidiSQL 9.4.0.5125 interface connected to the 'bookstore' database on 'localhost'. The left sidebar lists databases: bookstore (selected), information_schema, mysql, performance_schema, phpmyadmin, and test. The main pane displays the 'bookstore' schema with three tables: books, customers, and orders. The 'books' table has 6 rows and 8.2 KiB size. The 'customers' table has 8 rows and 2.2 KiB size. The 'orders' table has 7 rows and 2.1 KiB size. The bottom pane shows a history of SQL queries related to the 'information_schema' database.

名稱	數據條數	大小	創建 ^	修改時間	引擎	注釋	類型
books	6	8.2 KiB	2017-05-03 16:3...	2017-05-03 16:3...	MyISAM		Table
customers	8	2.2 KiB	2017-05-03 16:3...	2017-05-03 16:3...	MyISAM		Table
orders	7	2.1 KiB	2017-05-03 16:3...	2017-05-03 16:3...	MyISAM		Table

```
18 SHOW FUNCTION STATUS WHERE `Db`='information_schema';
19 SHOW PROCEDURE STATUS WHERE `Db`='information_schema';
20 SHOW TRIGGERS FROM `information_schema`;
21 SHOW EVENTS FROM `information_schema`;
22 SELECT *, EVENT_SCHEMA AS `Db`, EVENT_NAME AS `Name` FROM information_schema.EVENTS WHERE `EVENT_SCHEMA`='bookstore';
```

已連接: 00:00 h | MariaDB 10.1.19 | 實行時間: 05:01 h | UTC: 2017-05-03 9:53 | 空閒 .

來個實例吧！

- 假設現在你要開設一家網路書店，你有了 PHP 的技術了，那麼你要將：
 - 書籍資料
 - 客戶資料
 - 訂單資料
- 儲存在哪裡呢？
 - 沒錯！就是存在 MySQL 中！

建立資料庫 (database)

The screenshot shows the HeidiSQL interface for MySQL management. The left sidebar lists databases under the 'localhost' connection: information_schema, mysql, performance_schema, phpmyadmin, and test. The main pane displays the 'Database' tab of the 'Statistics' window, showing metrics for each database. A context menu is open over the 'information_schema' row, with the 'Create New' option expanded to show options like 'Database', 'Table', 'Table Copy', etc. The bottom pane shows the SQL query history, including several SHOW STATUS and SHOW PROCEDURE STATUS commands for the 'information_schema' database, and a final SELECT statement from the 'information_schema.EVENTS' table.

資料庫	大小	資料項	上次...	表	視圖	函數	過程	觸發器	事件	默認的校對規則
information_schema	176.0 KiB	78	2017...	78	0	0	0	0	0	utf8_general_ci
mysql										
performance_sche...										
phpmyadmin										
test										

```
26 SHOW TABLE STATUS
27 SHOW FUNCTION STATUS WHERE `Db` = 'information_schema';
28 SHOW PROCEDURE STATUS WHERE `Db` = 'information_schema';
29 SHOW TRIGGERS FROM `information_schema`;
30 SELECT *, EVENT_SCHEMA AS `Db`, EVENT_NAME AS `Name` FROM information_schema.EVENTS WHERE `EVENT_SCHEMA` = 'information_schema';
```

建立資料庫 (database)

The screenshot shows the HeidiSQL interface for MySQL. On the left, the database tree shows 'localhost' with databases: information_schema, mysql, performance_schema, phpmyadmin, and test. The total size is 176.0 KB. In the center, a table lists databases: information_schema (176.0 KiB, 78 tables), mysql, performance_sche..., phpmyadmin, and test. A 'Create Database...' dialog box is open in the foreground, prompting for a database name ('bookstore') and character set/collation ('utf8_general_ci'). The 'utf8_general_ci' dropdown is highlighted with a red box. Below the dialog, the SQL code for creating the database is shown: `CREATE DATABASE `bookstore` /*!40100 COLLATE`. At the bottom, a command history shows several SHOW STATUS and SHOW VARIABLES commands.

排序規則請選擇 utf8_general_ci 或 utf8_unicode_ci

```
28 SHOW PROCEDURE STATUS WHERE `Db`='information_schema';
29 SHOW TRIGGERS FROM `information_schema`;
30 SELECT *, EVENT_SCHEMA AS `Db`, EVENT_NAME AS `Name` FROM information_schema.EVENTS WHERE `EVENT_SCHEMA`='information_schema';
31 SHOW COLLATION;
32 SHOW VARIABLES LIKE 'collation_server';
```

已連接: 00:18 h MariaDB 10.1.19 運行時間: 21:15 h UTC: 2017-05-04 2:08 空閒 台灣大學 資訊工程系 資訊系統訓練班

建立資料庫 (database)

The screenshot shows the HeidiSQL interface. On the left, the 'Database browser' tree view shows 'localhost' expanded, with 'bookstore' selected. A red box highlights the 'bookstore' entry. The main pane displays a table with columns: 名稱 (Name), 數據條數 (Data rows), 大小 (Size), 創建 (Created), 修改時間 (Modified), 引擎 (Engine), 註釋 (Comments), and 類型 (Type). The 'bookstore' row is listed with 0 B under '大小' (Size). A large red text overlay in the center-right of the main pane reads: '在完成建立資料庫後，之後若是想要使用這個資料庫一定要用滑鼠在資料庫點兩下前方出現綠色的打勾才可以，表示正在使用此資料庫' (After creating the database, if you want to use it later, double-click the database with the mouse. A green checkmark will appear in front of it, indicating it is currently in use). At the bottom, the SQL history shows several SHOW STATUS commands for the 'bookstore' database, followed by a SELECT statement from 'information_schema.EVENTS'. The status bar at the bottom indicates the connection is established, the MariaDB version is 10.1.19, the run time is 21:16 h, the UTC date is 2017-05-04 2:09, and there is 0 free space.

localhost\bookstore - HeidiSQL 9.4.0.5125

文件 編輯 搜索 工具 Go to 幫助

P Donate to the HeidiSQL project

資料庫過濾器 表過濾器 ★

主機: 127.0.0.1 資料庫: bookstore 檢索

名稱 數據條數 大小 創建 ^ 修改時間 引擎 註釋 類型

bookstore 0 B

information_schema mysql performance_schema phpmyadmin test

在完成建立資料庫後，之後若是想要使用這個資料庫
一定要用滑鼠在資料庫點兩下
前方出現綠色的打勾才可以，表示正在使用此資料庫

38 SHOW TABLE STATUS FROM `bookstore`;
39 SHOW FUNCTION STATUS WHERE `Db`='bookstore';
40 SHOW PROCEDURE STATUS WHERE `Db`='bookstore';
41 SHOW TRIGGERS FROM `bookstore`;
42 SELECT *, EVENT_SCHEMA AS `Db`, EVENT_NAME AS `Name` FROM information_schema.EVENTS WHERE `EVENT_SCHEMA`='bookstore';

books 已連接: 00:19 h MariaDB 10.1.19 運行時間: 21:16 h UTC: 2017-05-04 2:09 空閒 :

連線校對

- utf8_general_ci vs. utf8_unicode_ci
 - 速度：
 - utf8_general_ci 快
 - utf8_unicode_ci 慢
 - 精準
 - utf8_general_ci 較差
 - utf8_unicode_ci 較準
 - 例：德文裡的 ß 要轉換成英文的時候如果是用 utf8_unicode_ci 轉換會變成正確的 ss，用 utf8_general_ci 的話則會變成單一的 s

建立資料表 (tables) (1)

- 首先想一想你需要幾個資料表？
 - 書本資料 + 客戶資料 + 訂單資料 = 3 個
 - 書本資料中要有什麼資訊？
 - 書本編號、書名、作者、售價、頁數、出版日期
 - 客戶資料呢？
 - 客戶編號、客戶姓名、職業
 - 訂單資料呢？
 - 訂單編號、購買的書本編號、購買者編號

建立資料表 (tables) (2)

書本資料

書本編號	書名	作者	售價	頁數	出版日期
A112	蛤蠣波特	Allan	690	400	2011-09-15
B334	一粥刊	Bill	199	250	2011-01-08
C556	天龍吼嘆	Cindy	1299	980	2011-02-06
D778	絕代霜蕉	David	880	605	2011-08-20
E990	還豬格格	Evelyn	590	475	2011-03-30
F012	我不是焦尼炸	Frank	990	322	2010-05-30

客戶資料

訂單資料

訂單編號	書本編號	客戶編號
0000000001	A112	8
0000000002	F012	5
0000000003	F012	3
0000000004	D778	3
0000000005	A112	1
0000000006	A112	2
0000000007	B334	3

客戶編號	姓名	職業
1	鷗鷺酒	學生
2	沉水匾	上班族
3	菜櫻紋	家管
4	消萬嘗	工程師
5	無蹲益	學生
6	酥真餽	保母
7	洩常停	學生
8	台灣大學 資訊工程系 資訊系統訓練班 沈橋	上班族

建立資料表 (tables) (4)

- 何謂欄位 (fields) ?
 - 即是要儲存哪些資料到資料庫中 !
 - 以「書本資料」為例 :

書本編號	書名	作者	售價	頁數	出版日期
A112	蛤蠣波特	Allan	690	400	2011-09-15
B334	一粥刊	Bill	199	250	2011-01-08
C556	天龍吼嘆	Cindy	1299	980	2011-02-06
D778	絕代霜蕉	David	880	605	2011-08-20
E990	販豬格格	Evelyn	590	475	2011-03-30
F012	我不是教你炸	Frank	990	322	2010-05-30



建立資料表 (tables) (3)

The screenshot shows the HeidiSQL interface for MySQL. The left sidebar lists databases: localhost, bookstore, information_schema, mysql, performance_schema, phpmyadmin, and test. The main pane shows a table structure with columns: 名稱 (Name), 數據條數 (Data Rows), 大小 (Size), 創建 (Create), 修改時間 (Last Update), 引擎 (Engine), 註釋 (Comments), and 類型 (Type). A context menu is open over the 'bookstore' database entry, with '創建新的(O)' (Create New) selected. A secondary submenu is open under '創建新的(O)', with '表(U)' (Table) selected.

HS localhost\bookstore - HeidiSQL 9.4.0.5125

文件 編輯 搜索 工具 Go to 幫助

localhost 資料庫過濾器 表過濾器

主機: 127.0.0.1 資料庫: bookstore 檢索

名稱 數據條數 大小 創建 ^ 修改時間 引擎 註釋 類型

編輯(K) Alt+Enter
刪除(L)...
清空表(M)... Shift+Del
運行過程(N)...
創建新的(O)
清除資料表過濾(P)
匯出資料腳本(Q)
維護(R)
在服務器上查找文本(S) Shift+Ctrl+F
批量表編輯器(T)
打開全部(U)
折疊全部(V)
對象樹類型選項(W)
打印(X)... Ctrl+P
刷新(Y) F5
斷開連接(Z)

資料庫(T)
表(U)
表複制(M)
視圖(W)
存儲過程(X)
觸發器(Y)
事件(Z)

```
40 SHOW PROCEDURE STATUS WHERE `Db`='bookstore';
41 SHOW TRIGGERS FROM `bookstore`;
42 SELECT *, EVENT_SCHEMA AS `Db`, EVENT_NAME AS `Name` FROM information_schema.EVENTS WHERE `EVENT_SCHEMA`='bookstore';
43 SHOW VARIABLES LIKE 'collation_server';
44 USE `bookstore`;
```

在選擇 已連接: 00:27 h MariaDB 10.1.19 實行時間: 21:24 h UTC: 2017-05-04 2:17 空間:

建立資料表 (tables) (5)

The screenshot shows the HeidiSQL interface for creating a new table named 'books'. The table has one column, 'booksName', defined as a VARCHAR type. The 'CREATE' tab is selected in the top navigation bar.

Table Creation Steps:

- Table Name: books
- Table Description: 此表格存放書籍資料
- Column Definition:
 - Column Name: booksName
 - Data Type: VARCHAR (selected from dropdown)
- Save Button: 保存

SQL History at the bottom:

```
42 SELECT *, EVENT_SCHEMA AS `Db`, EVENT_NAME AS `Name` FROM information_schema.EVENTS WHERE `EVENT_SCHEMA`='bookstore';
43 SHOW VARIABLES LIKE 'collation_server';
44 USE `bookstore`;
45 SHOW ENGINES;
46 SHOW VARIABLES LIKE 'collation_database';
```

Bottom Status Bar:

已連接: 00:30 h | MariaDB 10.1.19 | 運行時間: 21:28 h | UTC: 2017-05-04 2:21 | 空間:

資料型態 (1)

▪ 整數資料

型態	空間	範圍
TINYINT[(M)]	1 byte	Signed: -128 to 127 (-2 ⁷ to 2 ⁷ -1) Unsigned: 0 to 255 (0 to 2 ⁸ -1)
SMALLINT[(M)]	2 bytes	Signed: -32768 to 32767 (-2 ¹⁵ to 2 ¹⁵ -1) Unsigned: 0 to 65535 (0 to 2 ¹⁶ -1)
MEDIUMINT[(M)]	3 bytes	Signed: -8388608 to 8388607 (-2 ²³ to 2 ²³ -1) Unsigned: 0 to 16777215 (0 to 2 ²⁴ -1)
INT[(M)] INTEGER[(M)]	4 bytes	Signed: -2147483648 to 2147483647 (-2 ³¹ to 2 ³¹ -1) Unsigned: 0 to 4294967295 (0 to 2 ³² -1)
BIGINT[(M)]	8 bytes	Signed: -9223372036854775808 to 9223372036854775807 (-2 ⁶³ to 2 ⁶³ -1) Unsigned: 0 to 18446744073709551615 (0 to 2 ⁶⁴ -1)

M 代表「最大顯示寬度」，其值不得大於 255

若存入的數值超過該欄位的範圍時，MySQL 只會取其所能處理的最大值。例如在 TINYINT 型態的欄位中存入「300」這個值，結果將只剩下「127」

資料型態 (2)

• 浮點數資料

型態	空間	範圍
FLOAT(precision)	4 or 8	若 precision <= 24 的話，視為 FLOAT (單精度) 若 25 <= precision <= 53 的話，則視為 DOUBLE (倍精度)
FLOAT[(M,D)]	4 bytes	$\pm 1.175494351E-38$ $\pm 3.402823466E+38$
DOUBLE[(M,D)] REAL[(M,D)]	8 bytes	$\pm 1.7976931348623157E+308$ $\pm -2.2250738585072014E-308$

M 代表「最大顯示寬度」，其值不得大於 255

D 代表「小數位數」，其值不得大於 30，也不能大於 M-2

資料型態 (3)

日期與時間資料

型態	空間	範圍
DATE	3 bytes	'0000-00-00' to '9999-12-31'
TIME	3 bytes	'00:00:00' to '23:59:59'
DATETIME	8 bytes	'0000-00-00 00:00:00' to '9999-12-31 23:59:59'
TIMESTAMP[(M)]	4 bytes	自 1970 年起 · 至 2037 年的某時
YEAR[(2 4)]	1 bytes	4-digit format: 1901 to 2155 2-digit format: 1970 to 2069

資料型態 (4)

• 字串資料

型態	空間	範圍
CHAR(M)	M bytes	M bytes
VARCHAR(M)	L+1 bytes	M bytes
TINYBLOB, TINYTEXT	L+1 bytes	2^8-1 bytes
BLOB, TEXT	L+2 bytes	$2^{16}-1$ bytes
MEDIUMBLOB, MEDIUMTEXT	L+3 bytes	$2^{24}-1$ bytes
LONGBLOB, LONGTEXT	L+4 bytes	$2^{32}-1$ bytes

上表中的 L 代表「實際儲存的空間大小」，上述多種型態的空間需求都與實際存入的空間大小有關，意即它們實際儲存的空間是變動的

字串資料

- 使用 CHAR 及 VARCHAR 型態時，必須宣告「最大儲存長度」，就是 **M**
- 這兩種型態是相似的，所能儲存的最大長度都是 255 bytes。其相異之處在於 CHAR 是個固定長度的型態，而 VARCHAR 是個長度可變的型態

字串內容	CHAR(4)	空間需求	VARCHAR(4)	空間
"	' '	4 bytes	"	1 byte
'ab'	'ab '	4 bytes	'ab'	3 bytes
'abcd'	'abcd'	4 bytes	'abcd'	5 bytes
'abcdefg'h	'abcd'	4 bytes	'abcd'	5 bytes

CHAR vs. VARCHAR

- CHAR: 以空間換時間
- VARCHAR: 以時間換空間

CHAR:

實際只用了 12

剩下的 8 會存空白進來 (浪費)

假設總長度 20，即 CHAR(20)

VARCHAR:

實際只用了 12

剩下的 8 會縮小成 12

假設總長度 20，即 VARCHAR(20)

字串資料

▪ BLOB與TEXT

- 這兩種型態都是用在很大的字串資料，也就是超過255，在CHAR的範圍外
- 兩者都是可變長度，以實際輸入長度來決定儲存空間
- BLOB就是binary large object，也就是以BINARY儲存，與TEXT的差別在於大小寫的分別，TEXT沒有大小寫區別
- BLOB可以儲存其他二進位資料，如聲音圖檔資料等

為每個欄位決定名稱

- 欄位名稱的取名如同變數名稱一樣，取一個好記憶的、可以看字面意思就得知欄位目地的為佳
- 通常可以與table名稱做組合，如書本名稱可取做 **booksName** (**books** 為該資料表名稱)

書本編號	書名	作者	售價	頁數	出版日期
booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish

為每個欄位決定資料型態

- 如同為變數決定資料型態一樣，我們要每個欄位決定它要存放哪種型態的資料
- 以 books 資料表為例：

書本編號	書名	作者	售價	頁數	出版日期
A112	蛤蠣波特	Allan	690	400	2011-09-15

- 書本編號、書名、作者：
 - 皆是有中文、英文、數字的組合，故屬文字型態 (char / varchar)
- 售價、頁數：
 - 皆只有數字，故屬數字型態 (int / tinyint)
- 出版日期：
 - 日期型態資料 (date)

屬性的設定

- 屬性的選擇：
 - UNSIGNED 不分正負數
 - 以 1Byte 的 tinyint 來說
 - Signed: -128 to 127 (- 2^7 to 2^7-1)
 - Unsigned: 0 to 255 (0 to 2^8-1)
 - UNSIGNED ZEROFILL
 - 表示將把不足的位數填 0
 - 舉例來說，原先 4 位數的 zerofill 會變成：
 - 0001、0002、0003、0004、0005...

屬性的設定

基本 選項 索引 外鍵 Partitions CREATE 代碼

名稱 : books
注釋 : 此表格存放書籍資料

字段 : unsigned ZEROFILL

#	名稱	數據類型	長度/設置	無符號的	允許NU...	填零	默認	注釋	排序	表達式	虛擬
1	booksNo	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="radio"/> 無默認值				
2	booksName	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="radio"/> NULL				
3	booksAuthor	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="radio"/> CURRENT_TIMESTAMP				

ON UPDATE CURRENT_TIMESTAMP
 AUTO_INCREMENT

確定 取消

索引鍵的設定

- 索引鍵的選擇：
 - 如前所述，選擇具唯一性與鑑識性的欄位，如書本編號
 - auto_increment：不會重複的流水號
 - primary key: 主鍵
 - index: 主鍵以外的其他索引
 - unique: 不會重覆的鍵
 - fulltext: 全文檢索用索引

索引鍵的設定

The screenshot shows the HeidiSQL interface for MySQL. A context menu is open over a table structure, specifically the 'PRIMARY KEY' row. The steps are numbered as follows:

1. 索引 (Index) button in the toolbar.
2. 基本 (Basic) tab selected in the top navigation bar.
3. 項目/長度 (Type/Length) column showing 'PRIMARY'.
4. Context menu item '增加字段 (V) Shift+Ctrl+Ins' (Add Field) is highlighted.

The table structure below shows three fields: booksNo, booksName, and booksAuthor, all defined as VARCHAR(50). The 'booksNo' field is currently designated as the primary key.

#	名稱	數據類型	長度/設置	無符號的	允許NU...	填零	默認	注釋
1	booksNo	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	無默認值	書籍編號
2	booksName	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	無默認值	
3	booksAuthor	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	無默認值	

At the bottom, the SQL query history shows:

```
42 SELECT *, EVENT_SCHEMA AS `db`, EVENT_NAME AS `Name` FROM information_schema.EVENTS WHERE `EVENT_SCHEMA`='bookstore';
43 SHOW VARIABLES LIKE 'collation_server';
44 USE `bookstore`;
45 SHOW ENGINES;
46 SHOW VARIABLES LIKE 'collation_database';
```

The status bar at the bottom indicates: 已連接: 00:44 h | MariaDB 10.1.19 | 21:42 h | UTC: 2017-05-04 2:34 | 空間 .

索引鍵的設定

點此在選擇要讓哪個欄位變成PK

#	名稱	數據類型	長度/設置	無符號的	允許NU...	填零	默認	注 書
1	booksNo	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	無默認值	
2	booksName	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	無默認值	
3	booksAuthor	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	無默認值	

索引鍵的設定(另一種設定)

基本 選項 索引 外鍵 Partitions CREATE 代碼 ALTER 代碼

添加
刪除
清除
向上
向下

名稱

字段 : 添加 刪除 向上 向下

#	名稱	數據類型	長度/設置	無符號的	允許NU...	填零	默認	注釋	排序	表達式	虛擬
1	booksNo	INT		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT	書籍編號			
2	booksName	VARCHAR		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	無默認值				
3	booksAuthor	VARCHAR		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	無默認值				

複制(C) Ctrl+C
複制選定字段(S)
粘貼字段(T)
增加字段(U) Ctrl+Ins
刪除字段(V) Ctrl+Del
向上移動(W) Ctrl+U
向下移動(X) Ctrl+D
創建新索引(Y)
加入索引(Z)

PRIMARY
KEY
UNIQUE
FULLTEXT
SPATIAL

直接在欄位這邊點右鍵選擇

auto_increment (自動流水號)

基本 選項 索引 外鍵 Partitions CREATE 代碼 ALTER 代碼

+ 添加 - 刪除 ✎ 清除 ▲ 向上 ▼ 向下

名稱 > PRIMARY KEY

字段 : 添加 - 刪除 ▲ 向上 ▼ 向下

1. #	名稱	數據類型 2.	長度/設置	無符號的	允許NU...	填零	默認	注釋	排序	表達式	虛擬
1	booksNo	INT					<input type="radio"/> 無默認值				
2	booksName	VARCHAR	50				<input type="radio"/> Custom:	<input type="text"/>			
3	booksAuthor	VARCHAR	50				<input type="radio"/> NULL				
							<input type="radio"/> CURRENT_TIMESTAMP				
							<input type="checkbox"/> ON UPDATE CURRENT_TIMESTAMP				
							<input checked="" type="radio"/> AUTO_INCREMENT				

3. 默認
4. 確定 取消

The screenshot shows the MySQL Workbench interface for creating a new table. The table has three columns: booksNo (INT, primary key, auto-increment), booksName (VARCHAR(50)), and booksAuthor (VARCHAR(50)). The 'booksNo' column is highlighted with a red box. A context menu is open over the '默認' (Default) field of the 'booksNo' row, listing options: '無默認值' (No Default Value), 'Custom:' (with a text input field), 'NULL', 'CURRENT_TIMESTAMP', 'ON UPDATE CURRENT_TIMESTAMP', and 'AUTO_INCREMENT'. The 'AUTO_INCREMENT' option is selected and highlighted with a red box. The '確定' (Confirm) button is also highlighted with a red box.

books 資料表

書本編號	書名	作者	售價	頁數	出版日期
booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
varchar(255)	varchar(255)	varchar(255)	int	int	date
primary key			unsigned	unsigned	

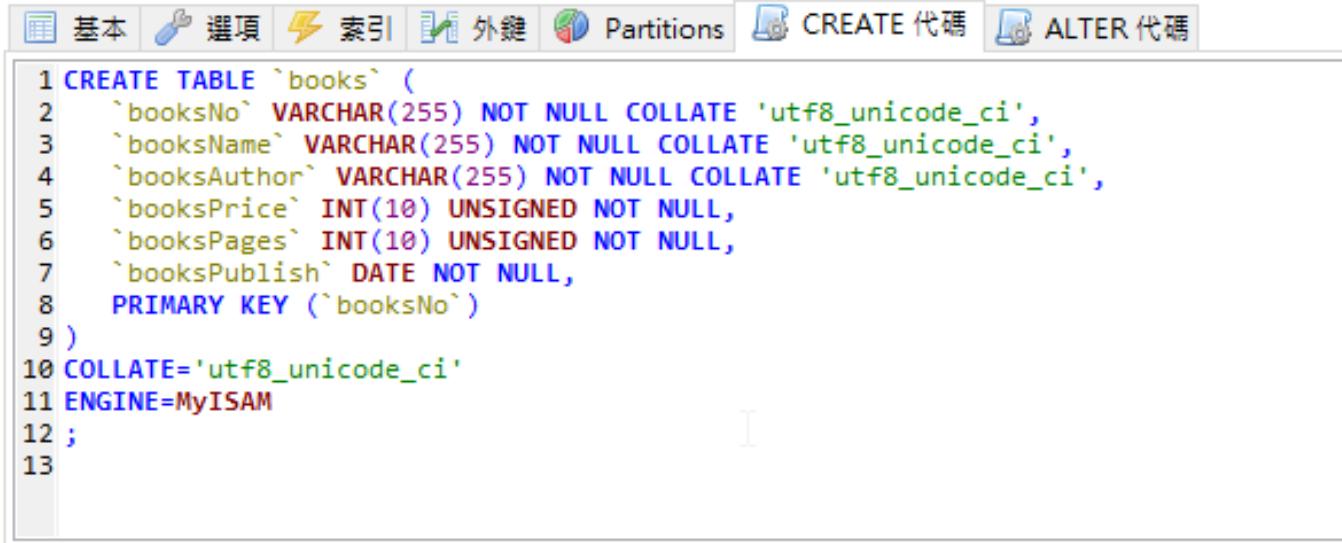
基本 選項 索引 外鍵 Partitions CREATE 代碼 ALTER 代碼

名稱 : books
注釋 : 書籍資料表，此表格存放書籍的基本資料

字段 : + 添加 - 刪除 ▲ 向上 ▼ 向下

#	名稱	數據類型	長度/設置	無符號的	允許NU...	填零	默認	注釋	排序	表達式	虛擬
1	booksNo	VARCHAR	255	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	無默認值		utf8_unicode_ci		
2	booksName	VARCHAR	255	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	無默認值		utf8_unicode_ci		
3	booksAuthor	VARCHAR	255	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	無默認值		utf8_unicode_ci		
4	booksPrice	INT	10	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	無默認值				
5	booksPages	INT	10	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	無默認值				
6	booksPublish	DATE		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	無默認值				

新增完 books 資料表後



The screenshot shows the 'CREATE 代碼' tab selected in MySQL Workbench. The code displayed is:

```
1 CREATE TABLE `books` (
2     `booksNo` VARCHAR(255) NOT NULL COLLATE 'utf8_unicode_ci',
3     `booksName` VARCHAR(255) NOT NULL COLLATE 'utf8_unicode_ci',
4     `booksAuthor` VARCHAR(255) NOT NULL COLLATE 'utf8_unicode_ci',
5     `booksPrice` INT(10) UNSIGNED NOT NULL,
6     `booksPages` INT(10) UNSIGNED NOT NULL,
7     `booksPublish` DATE NOT NULL,
8     PRIMARY KEY (`booksNo`)
9 )
10 COLLATE='utf8_unicode_ci'
11 ENGINE=MyISAM
12 ;
13
```

這段便是 PMA 自動生成的 SQL 指令碼，其中可以看到關鍵字：

- CREATE TABLE 表示建立一個表格
- 表格名稱 books
- 其它資訊為資料型態與屬性，每個欄位最後用 , 做分隔

客戶資料表

- 客戶資料表：
 - 客戶編號：數字 1~n (int, unsigned, A_I)
 - 姓名：文字 (char / varchar)
 - 職業：文字 (char / varchar)

客戶編號	姓名	職業
1	鴟鴞酒	學生
2	沉水匾	上班族
3	菜櫻紋	家管
4	消萬嘗	工程師
5	無蹲益	學生
6	酥真鯧	保母
7	洩常停	學生
8	沉橘	上班族

新增客戶資料表

▪ 取名 customers

客戶編號	姓名	職業
customersNo	customersName	customersJob
int	varchar(255)	varchar(255)
unsigned		
primary key		
auto_increment		

基本 選項 索引 外鍵 Partitions CREATE 代碼 ALTER 代碼

名稱 : customers
注釋 : 客戶資料表，此表格存放客戶資料

字段 : 添加 □ 檢查 向上 ▼ 向下

#	名稱	數據類型	長度/設置	無符號的	允許NU...	填零	默認	注釋	排序	表達式	虛擬
1	customersNo	INT	10	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREMENT				
2	customersName	VARCHAR	255	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	無默認值		utf8_unicode_ci		
3	customersJob	VARCHAR	255	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	無默認值		utf8_unicode_ci		

新增客戶資料表的指令碼

基本 選項 索引 外鍵 Partitions CREATE 代碼 ALTER 代碼

```
1 CREATE TABLE `customers` (
2     `customersNo` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
3     `customersName` VARCHAR(255) NOT NULL COLLATE 'utf8_unicode_ci',
4     `customersJob` VARCHAR(255) NOT NULL COLLATE 'utf8_unicode_ci',
5     PRIMARY KEY (`customersNo`)
6 )
7 COMMENT='客戶資料表，此表格存放客戶資料'
8 COLLATE='utf8_unicode_ci'
9 ENGINE=MyISAM
10 AUTO_INCREMENT=9
11 ;
12
```

訂單資料表

- 訂單資料表：
 - 訂單編號：數字 (int, unsigned zerofill, A_I)
 - 書本編號：文字 (char / varchar)
 - 客戶編號：數字 (int, unsigned)

訂單編號	書本編號	客戶編號
0001	A112	8
0002	F012	5
0003	F012	3
0004	D778	3
0005	A112	1
0006	A112	2
0007	B334	3

建立訂單資料表

▪ 取名 orders

訂單編號	書本編號	客戶編號
ordersNo	ordersBook	ordersCustomer
int	varchar(255)	int
Unsigned zerofill		unsigned
primary key		
auto_increment		

基本 選項 索引 外鍵 Partitions CREATE 代碼 ALTER 代碼

名稱 : orders

注釋 :

字段 : 添加 向上

#	名稱	數據類型	長度/設置	無符號的	允許NU...	填零	默認	注釋	排序	表達式	虛擬
1	ordersNo	INT	10	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	AUTO_INCREMENT				
2	ordersBook	VARCHAR	255	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	無默認值			utf8_unicode_ci	
3	ordersCustomer	INT	10	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	無默認值				

新增訂單資料表的指令碼

The screenshot shows the MySQL Workbench interface with the 'CREATE 代碼' tab selected. The code pane displays the SQL command to create the 'orders' table:

```
1 CREATE TABLE `orders` (
2     `ordersNo` INT(10) UNSIGNED ZEROFILL NOT NULL AUTO_INCREMENT,
3     `ordersBook` VARCHAR(255) NOT NULL COLLATE 'utf8_unicode_ci',
4     `ordersCustomer` INT(10) UNSIGNED NOT NULL,
5     PRIMARY KEY (`ordersNo`)
6 )
7 COLLATE='utf8_unicode_ci'
8 ENGINE=MyISAM
9 AUTO_INCREMENT=8
10 ;
11
```

Database 的管理

The screenshot shows the HeidiSQL interface connected to the 'bookstore' database on 'localhost'. The 'orders' table is selected, displaying its structure. A red box highlights the refresh icon in the toolbar.

名稱	數據條數	大小	創建 ^	修改時間	引擎	注釋	類型
books	6	8.2 KiB	2017-05-04 10:5...	2017-05-04 10:5...	MyISAM	書籍資料表，...	Table
customers	8	2.2 KiB	2017-05-04 10:5...	2017-05-04 10:5...	MyISAM	客戶資料表，...	Table
orders	7	2.1 KiB	2017-05-04 10:5...	2017-05-04 10:5...	MyISAM		Table

若是表格或是資料看起來不如預期，先試試看重新整理
案右鍵可以找到，或是看上邊的註記

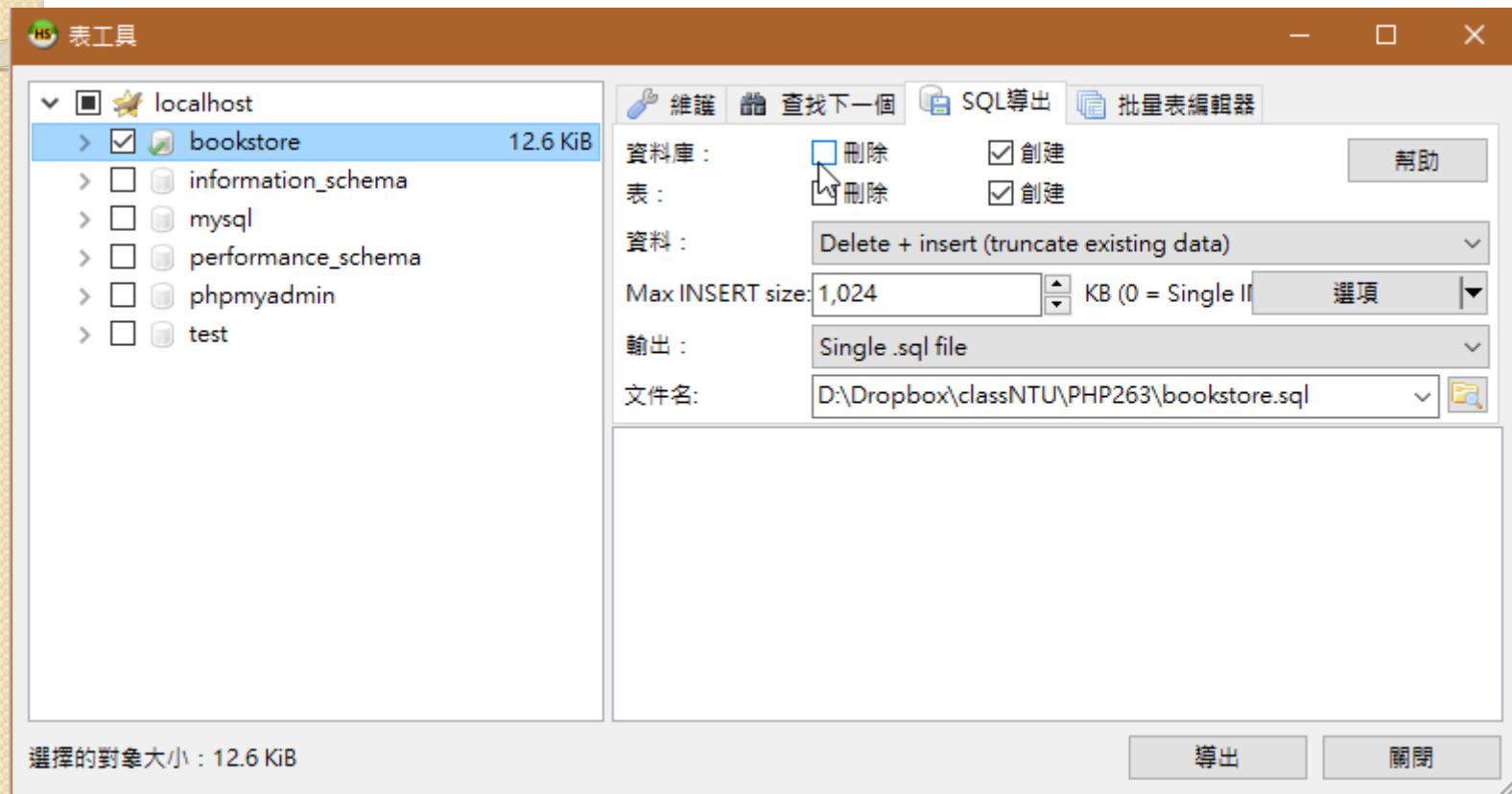
輸出

The screenshot shows the HeidiSQL interface for a MySQL connection to localhost. The current database is 'bookstore'. A context menu is open over the 'bookstore' entry in the left sidebar, specifically over the '匯出資料腳本(Q)' (Export Script) option. The menu includes:

- 編輯(E) (Alt+Enter)
- 刪除(L)...
- 清空表(M)...
- 運行過程(N)...
- 創建新的(O)
- 清除資料表過濾(P)
- 匯出資料腳本(Q)** (highlighted)
- 維護(R)
- 在服務器上查找文本(S) Shift+Ctrl+F
- 批量表編輯器(T)
- 打開全部(U)
- 折疊全部(V)
- 對象樹類型選項(W)
- 打印(X)... Ctrl+P
- 刷新(Y) F5
- 斷開連接(Z)

The main window displays the 'bookstore' database structure with tables like 'books' and 'authors'.

輸出



載入

localhost\bookstore - HeidiSQL 9.4.0.5125

文件 編輯 搜索 工具 Go to 幫助

主機: 127.0.0.1 資料庫: bookstore 挑查

資料庫過濾器 表過濾器 開啟

localhost

bookstore

information_schema

mysql

performance_schema

phpmyadmin

test

搜尋位置(I): PHP263

名稱 | 修改日期 | 類型 | 大小

ch03simple	2016-03-08 16:55	檔案資料夾	
ch6	2016-05-17 10:24	檔案資料夾	
ch7	2016-05-17 16:02	檔案資料夾	
ch8	2016-05-17 16:27	檔案資料夾	
pic	2016-03-03 15:15	檔案資料夾	
pic mysql	2016-03-22 17:46	檔案資料夾	
pt4-1	2016-03-15 17:40	檔案資料夾	
vscode install setting sync	2017-05-04 10:21	檔案資料夾	
vscode 教學	2017-04-10 12:25	檔案資料夾	
上課錄影	2017-05-02 10:06	檔案資料夾	
上課錄影280	2017-04-06 22:16	檔案資料夾	
bookstore	2017-05-03 16:30	SQL-Script	4 KB
ch7	2016-04-08 16:55	SQL-Script	6 KB

快速存取 桌面 媒體櫃 本機 網路

檔案名稱(N): bookstore 開啟(O) 3.

檔案類型(T): SQL文件 (*.sql) 取消

編碼: 自動檢測(可能失敗)

159 /* 進入書籍 "localhost" 160 SHOW CREATE TABLE `book` 161 SELECT * FROM `book` 162 SHOW CREATE TABLE `book` 163 SHOW CHARSET;

已連接: 01:11 h MariaDB 10.1.19 實行時間: 22:09 h UTC: 2017-05-04 3:02 空間:

在課程網站上下載 bookstore.sql，試著載入看看

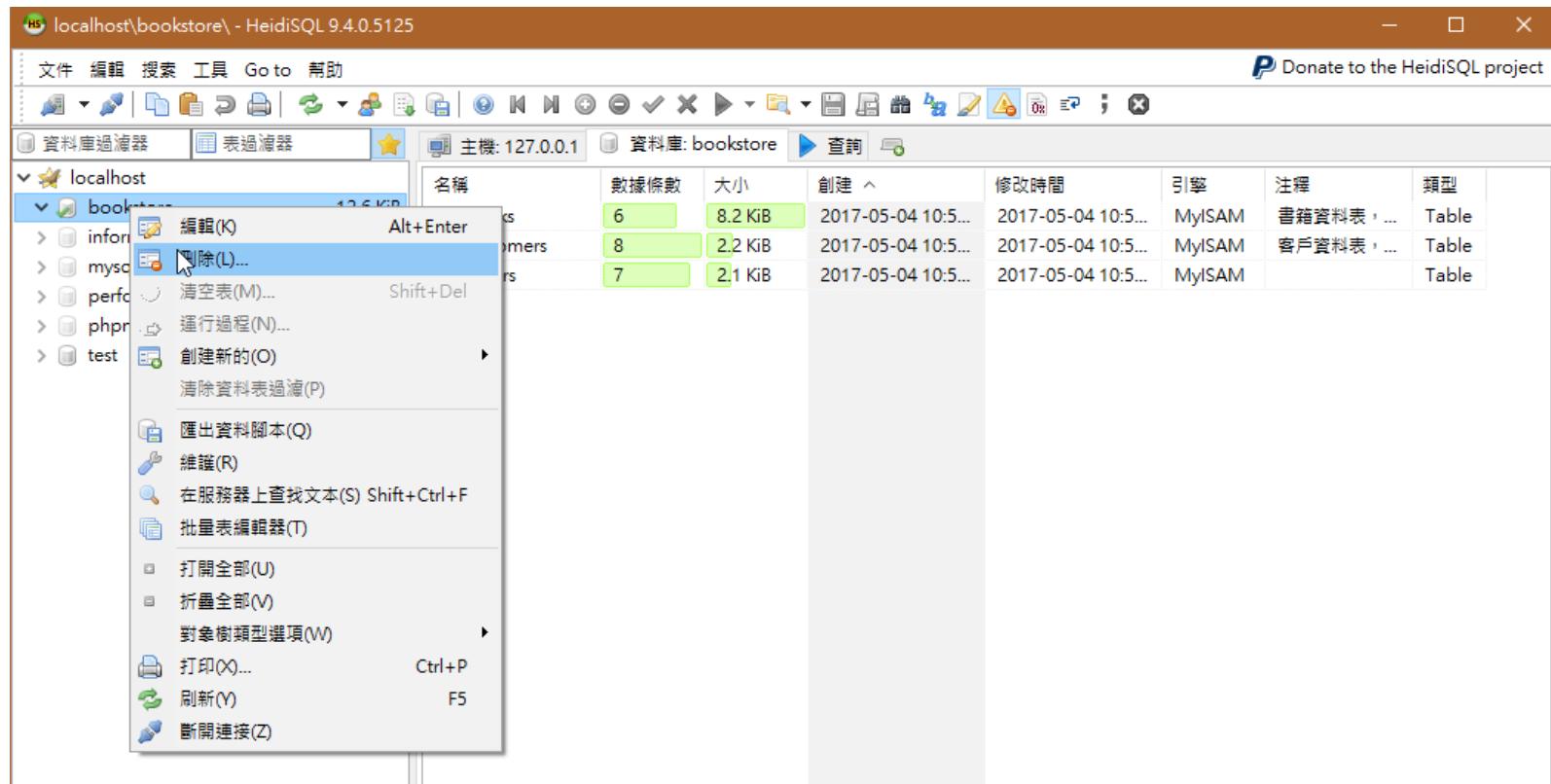
載入 bookstore

The screenshot shows the HeidiSQL interface with the following details:

- Title Bar:** localhost\bookstore - HeidiSQL 9.4.0.5125
- Toolbar:** Includes standard database management icons.
- Left Panel (Database Browser):** Shows the database structure under "localhost". The "bookstore" database is selected, highlighted in blue, and its size is listed as 12.6 KiB.
- Central Panel (Script Editor):** Displays the SQL script "bookstore.sql". A red box highlights the play button icon in the toolbar above the editor area, indicating the script is ready to be executed.
- Right Panel (Tool Buttons):** A vertical list of buttons for various database tasks: 字段 (Fields), SQL函數 (SQL Functions), SQL關鍵字 (SQL Keywords), SQL片段 (SQL Snippets), 查詢歷史 (Query History), 查詢分析 (Query Analysis), and Bind parameters.
- Bottom Panel (Filter Bar):** A search bar labeled "過濾:" (Filter:).

```
1 --  
2 -- 主機: 127.0.0.1  
3 -- 服務器版本: 10.1.19-MariaDB - mariadb.org binary  
4 -- 服務器操作系統: Win32  
5 -- HeidiSQL 版本: 9.4.0.5125  
6 --  
7  
8 /*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;  
9 /*!40101 SET NAMES utf8 */;  
10 /*!50503 SET NAMES utf8mb4 */;  
11 /*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=NONZERO */;  
12 /*!40101 SET @OLD_SQL_MODE=@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */.  
13  
14  
15 -- 導出 bookstore 的資料庫結構  
16 DROP DATABASE IF EXISTS `bookstore`;  
17 CREATE DATABASE IF NOT EXISTS `bookstore` /*!40100 DEFAULT CHARACTER SET u1  
18 USE `bookstore`;  
19  
20 -- 導出 表 bookstore.books 結構  
21 DROP TABLE IF EXISTS `books`;  
22 CREATE TABLE IF NOT EXISTS `books` (  
23   `booksNo` varchar(255) COLLATE utf8_unicode_ci NOT NULL,  
24   `booksName` varchar(255) COLLATE utf8_unicode_ci NOT NULL,  
25   `booksAuthor` varchar(255) COLLATE utf8_unicode_ci NOT NULL,  
26   `booksPrice` int(10) unsigned NOT NULL,  
<
```

刪除



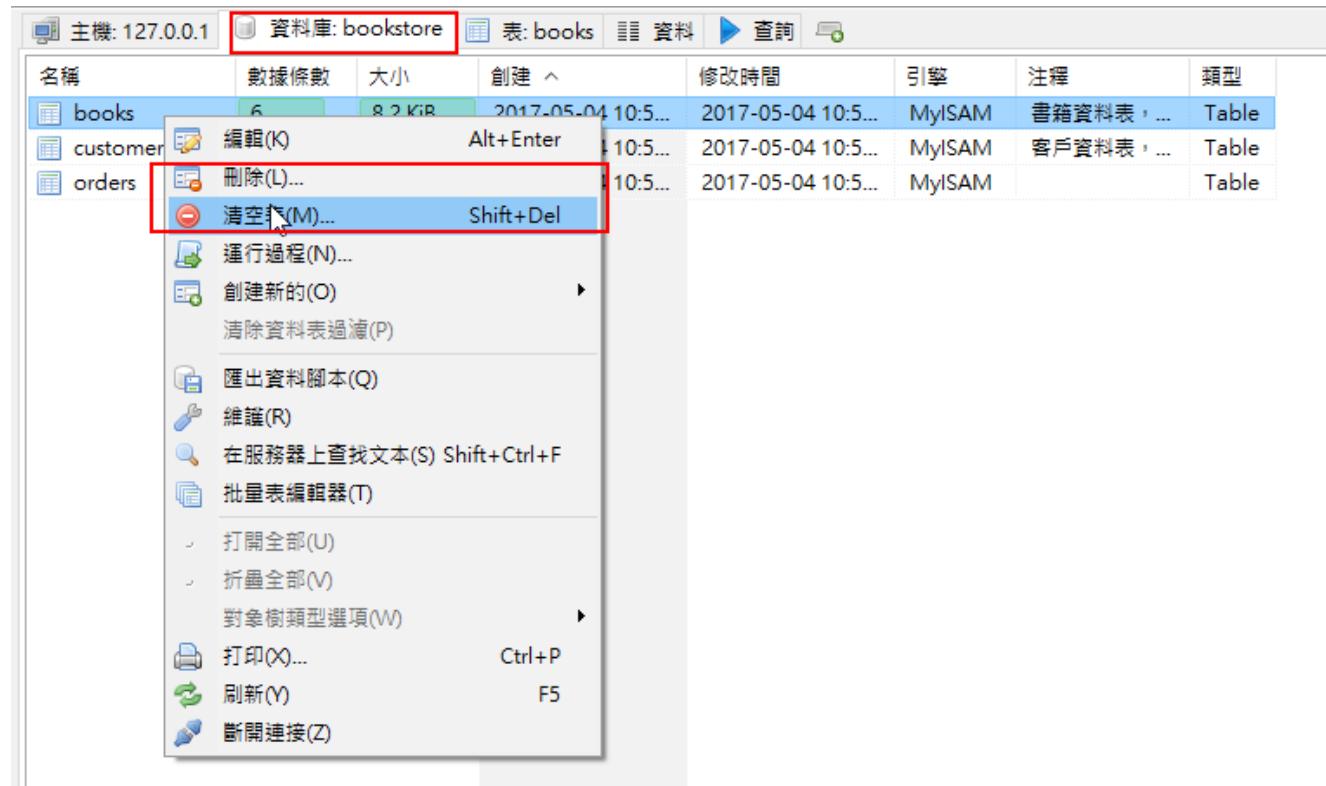
按下確定就
救不回來資
料囉！

新增資料 (1)

The screenshot shows the HeidiSQL interface for MySQL. The left sidebar lists databases: localhost, bookstore, information_schema, mysql, performance_schema, phpmyadmin, and test. The bookstore database is selected, showing a size of 12.6 KiB. The main window displays the 'books' table with 7 records. A new row is being added, indicated by a red box around the '資料' tab in the toolbar and a red box around the last row in the data grid. The row being added has booksNo 'G8563', booksName 'phpMySql實戰', booksAuthor 'Tom', booksPrice '200', booksPages '200', and booksPublish '2017-05-04'. The toolbar icons include: back, forward, search, insert (highlighted with a red box), update, delete, refresh, and other database management tools.

booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
A112	蛤蠣波特	Allan	690	400	2011-09-15
B334	一粥刊	Bill	199	250	2011-01-08
C556	天龍吼嘆	Cindy	1,299	980	2011-02-06
D778	絕代霜蕉	David	880	605	2011-08-20
E990	還豬格格	Evelyn	590	475	2011-03-30
F012	我不見尼作	Frank	990	332	2010-05-30
G8563	phpMySql實戰	Tom	200	200	2017-05-04

清空及刪除



按下確定就
救不回來資
料囉！

編輯

主機: 127.0.0.1 | 資料庫: bookstore | 表: books | 資料 | 檢視 | 查詢 | 印列

bookstore.books: 7 總記錄 | 下一個 | 顯示所有 | 排序

booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish	
A112	蛤蠣波特	Allan	690	400	2011-09-15	
B334	一粥刊	Bill	199	250	2011-01-08	
C556	天龍吼嘆	Cindy	1,299	980	2011-02-06	
D778	絕代霜蕉	David	880	605	2011-08-20	
E990	還豬格格	Evelyn	590	475	2011-03-30	
F012	我不是焦尼炸	Frank	990	332	2010-05-30	
G8563	phpMysql實戰	Tom	200	200	2017-05-04	

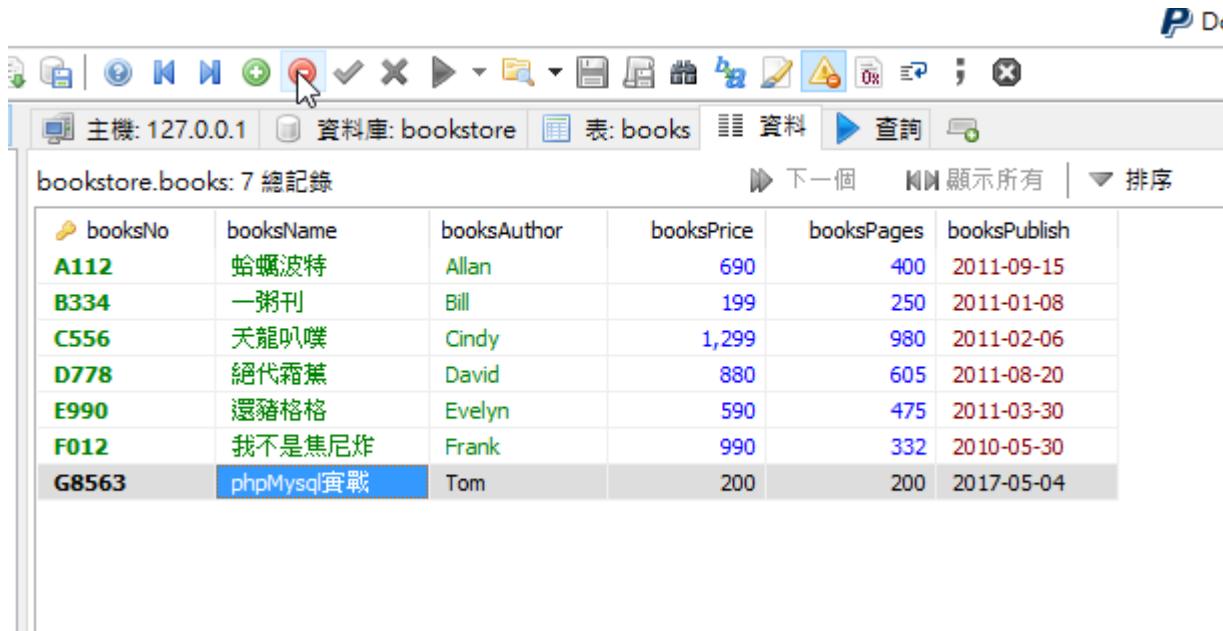
編輯

The screenshot shows the HeidiSQL interface for editing data in the 'books' table of the 'bookstore' database. The table has 7 records. A red box highlights the save icon (a green checkmark) in the toolbar.

booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
A112	蛤蠣波特	Allan	690	400	2011-09-15
B334	一粥刊	Bill	199	250	2011-01-08
C556	天龍吼嘆	Cindy	1,299	980	2011-02-06
D778	絕代霜蕉	David	880	605	2011-08-20
E990	還豬格格	Evelyn	590	475	2011-03-30
F012	我不是焦尼炸	Frank	990	332	2010-05-30
G8563	phpMysql實戰	Tom	200	200	2017-05-04

左上角的紅色標記表示資料尚未儲存要按打勾

刪除 (資料)



The screenshot shows the MySQL Workbench interface. The toolbar at the top has various icons for database management. The main window displays the 'books' table from the 'bookstore' database, which contains 7 records. The table has columns: booksNo, booksName, booksAuthor, booksPrice, booksPages, and booksPublish. The last row, with booksNo G8563 and booksName 'phpMysql實戰', is selected and highlighted with a blue border. The 'Delete' icon in the toolbar is circled in red, indicating it is the target of the current operation.

booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
A112	蛤蠣波特	Allan	690	400	2011-09-15
B334	一粥刊	Bill	199	250	2011-01-08
C556	天龍吼嘍	Cindy	1,299	980	2011-02-06
D778	絕代霜蕉	David	880	605	2011-08-20
E990	還豬格格	Evelyn	590	475	2011-03-30
F012	我不是焦尼炸	Frank	990	332	2010-05-30
G8563	phpMysql實戰	Tom	200	200	2017-05-04



◦ 第三部份：SQL 語言

SQL語言簡介

- SQL 語言的分類：
 - 管理資料庫及資料表相關
 - 建立 CREATE、卸除 DROP
 - 管理資料表欄位相關
 - 修改資料表 ALTER
 - 管理資料內容相關
 - 新增 INSERT、編輯 UPDATE、刪除 DELETE
 - 查詢相關
 - 查詢 SELECT
 - 條件過濾 WHERE
 - 群組 GROUP
 - 排序 ORDER

Database – 建立/卸除

- 建立：
 - **CREATE DATABASE {database_name};**
 - 例：
 - CREATE DATABASE test;
- 卸除：
 - **DROP DATABASE {database_name};**
 - 例：
 - DROP DATABASE test;

Tables – 建立

- 建立：
 - **CREATE TABLE {table_name}**
(
 欄位名稱 資料型態 [(資料大小)]
 [NOT NULL | NULL]
 [AUTO_INCREMENT],
 ...
 [PRIMARY KEY (欄位名稱)]
);
 - 例：
 - **CREATE TABLE test (testNo int(10) NOT NULL AUTO_INCREMENT, testName varchar(255), PRIMARY KEY(testNo));**

Table – 卸除

- 卸除：
 - **DROP TABLE {table_name};**
 - 例：
 - **DROP TABLE test;**

Field – 修改 (新增/修改/刪除)

- ALTER TABLE {**table_name**}
 - 新增 - **ADD** 欄位名稱 資料型態 [NOT NULL | NULL];
 - 例：
 - ALTER TABLE test ADD testData char(10);
 - 修改 - **CHANGE** 欄位名稱 欄位名稱 資料型態
 - 例：
 - ALTER TABLE test CHANGE testData testText varchar(20);
 - 刪除 - **DROP** 欄位名稱
 - 例：
 - ALTER TABLE test DROP testText;

資料內容相關 – 新增

- 新增資料內容 INSERT
 - **INSERT INTO {table_name}**
(欄位1, 欄位2, …, 欄位n)
VALUES
('內容1', '內容2', …, '內容n');
 - 例：
 - **INSERT INTO test (testName, testData)**
VALUES ('Hello', 'World');

資料內容相關 - 修改

- 修改資料內容 UPDATE

- UPDATE {table_name}

SET 欄位1 = '內容1',
 欄位2 = '內容2',

...

 欄位n = '內容n';

- 例：

- UPDATE test SET testName = 'Hi';

資料內容相關 – 刪除

- 刪除資料內容 DELETE
 - **DELETE FROM {table_name}**
 - 例：
 - `DELETE FROM test;`

查詢相關 – SELECT (1)

- 選取查詢使用SELECT指令，用於將一個或多個資料表的資料選取出來
 - SELECT {*|欄位} [AS 別名]
FROM {table_name}
[WHERE 過濾條件]
[GROUP BY 群組項目, ...]
[ORDER BY 排序條件 [ASC|DESC], ...]
[LIMIT [offset,] rows]

查詢相關 – SELECT (2)

The screenshot shows the HeidiSQL interface. The top bar displays the connection information "主機: 127.0.0.1" and "資料庫: bookstore". The toolbar includes standard database management icons. The query editor window contains the following SQL code:

```
1 select *
2 from books
```

To the right of the query editor is a sidebar with various navigation links:

- > books 表的字段
- > SQL函數
- > SQL關鍵字
- > SQL片段
- > 查詢歷史
- > 查詢分析
- > Bind parameters

Below the sidebar is a preview window titled "books (6×7)" showing the data from the "books" table:

booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
A112	蛤蠣波特	Allan	690	400	2011-09-15
B334	一粥刊	Bill	199	250	2011-01-08
C556	天龍喇叭	Cindy	1,299	980	2011-02-06
D778	絕代霜葉	David	880	605	2011-08-20
E990	還豬格格	Evelyn	590	475	2011-03-30
F012	我不是焦尼炸	Frank	990	332	2010-05-30
G8563	phpMysql實戰	Tom	200	200	2017-05-04

過濾查詢 – WHERE (1)

- WHERE 子句可用來建立篩選資料的條件，篩選條件分四類：
 - 文字查詢
 - 數字查詢
 - 日期查詢
 - 多重查詢

過濾查詢 – WHERE (2)

- 文字查詢 (1/2)
 - 類似 if 的概念，用 `=` 與 `!=` 判斷條件是否成立，文字部份需用單引號 ' 框起
 - WHERE** 欄位 = '文字';
 - 例：
 - `SELECT * FROM books WHERE booksAuthor = 'Allan';`
 - `SELECT * FROM books WHERE booksName != '天龍吼嘆';`

←↑→	booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
□	✎	✖				
	A112	蛤蠣波特	Allan	690	400	2011-09-15

←↑→	booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
□	✎	✖				
	A112	蛤蠣波特	Allan	690	400	2011-09-15
	B334	一粥刊	Bill	199	250	2011-01-08
	D778	綿岱霜蕉	David	880	605	2011-08-20
	E990	還豬格格	Evelyn	590	475	2011-03-30
	F012	我不是焦尼炸	Frank	990	332	2010-05-30

過濾查詢 – WHERE (3)

- 文字查詢 (2/2)
 - 文字查詢時可搭配萬用字元來做模糊比對，此時需將等號改成 **LIKE**
 - % (百分比)代表任何 n 個字元
 - WHERE 欄位 **LIKE '[%]文字[%]'**
 - _ (底線) 代表任何 1 個字元
 - WHERE 欄位 **LIKE '[_]文字[_]'**
 - 例：
 - SELECT * FROM books WHERE booksNo LIKE '%2';**

← ↑ →	booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
<input type="checkbox"/>  	A112	蛤蠣波特	Allan	690	400	2011-09-15
<input type="checkbox"/>  	F012	我不是焦尼炸	Frank	990	332	2010-05-30

- SELECT * FROM books WHERE booksNo LIKE '_112';**

← ↑ →	booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
<input type="checkbox"/>  	A112	蛤蠣波特	Allan	690	400	2011-09-15

過濾查詢 – WHERE (4)

- 數字查詢 (1/2)
 - 數字查詢時一樣可以使用 = 、 != 來做比對，同時也可以使用 > 、 < 或 BETWEEN 子句做比對，數字部份不加單引號！
 - WHERE 欄位 > 某數值；
 - 例：
 - **SELECT * FROM books WHERE booksPrice > 750;**

← ↑ →	booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
<input type="checkbox"/>  	C556	天龍吼嘆	Cindy	1299	980	2011-02-06
<input type="checkbox"/>  	D778	絕代霜蕉	David	880	605	2011-08-20
<input type="checkbox"/>  	F012	我不是焦尼炸	Frank	990	332	2010-05-30

過濾查詢 – WHERE (5)

- 數字查詢 (2/2)
 - WHERE 欄位 BETWEEN 數值1 AND 數值2;
 - 例：
 - **SELECT * FROM books WHERE booksPages BETWEEN 400 AND 800;**

←↑→	booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
<input type="checkbox"/>  	A112	蛤蠣波特	Allan	690	400	2011-09-15
<input type="checkbox"/>  	D778	綴代霜蕉	David	880	605	2011-08-20
<input type="checkbox"/>  	E990	還豬格格	Evelyn	590	475	2011-03-30

過濾查詢 – WHERE (6)

- 日期查詢
 - 同樣我們也可以將剛才的查詢用在日期欄位中，比對日期與文字相同，需加單引號：
 - 尋找2011上半年所出版的書
 - SELECT * FROM books WHERE booksPublish BETWEEN '2011-01-01' AND '2011-06-30';**

← ↑ →	booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
<input type="checkbox"/>  	B334	一粥刊	Bill	199	250	2011-01-08
<input type="checkbox"/>  	C556	天龍吼嘆	Cindy	1299	980	2011-02-06
<input type="checkbox"/>  	E990	還豬格格	Evelyn	590	475	2011-03-30

- 當然也可以使用 > 或 < 找某日期以前或以後出版的書

過濾查詢 – WHERE (7)

- 多重查詢 (1/2)
 - 在 SQL 指令中我們一樣可以使用邏輯 AND 與 OR 做多個查詢
 - 例：
 - 找出售價 \$1,000 內，頁數大於 400 的書
 - SELECT * FROM books WHERE booksPages > 400 AND booksPrice < 1000;**

←↑→	booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
<input type="checkbox"/>  	D778	絕代霜蕉	David	880	605	2011-08-20
<input type="checkbox"/>  	E990	還豬格格	Evelyn	590	475	2011-03-30

過濾查詢 – WHERE (8)

- 多重查詢 (2/2)

- 例：

- 找出書名為「天」開頭，或作者為 Frank 的書
 - SELECT * FROM books WHERE booksName LIKE '天%' OR booksAuthor = 'Frank';**

←↑→	booksNo	booksName	booksAuthor	booksPrice	booksPages	booksPublish
<input type="checkbox"/>  	C556	天龍吼嘆	Cindy	1299	980	2011-02-06
<input type="checkbox"/>  	F012	我不是焦尼炸	Frank	990	332	2010-05-30

AS 子句的使用

- 有時我們會在 SQL 指令中做一些計算，這些計算結果可以暫存在某個不存在的欄位，以方便我們顯示
 - 例：計算每一本書的印製成本 (頁/售價)
 - SELECT booksPages, booksPrice, booksPages/booksPrice AS cost FROM books;**

booksPages	booksPrice	cost
400	690	0.5797
250	199	1.2563
980	1299	0.7544
605	880	0.6875
475	590	0.8051
332	990	0.3354

mySQL 內建函數

• 常用的 mySQL 內建函數

函數	說明
AVG(欄位)	計算欄位的算數平均值
COUNT(欄位)	計算SELECT敘述查詢所得的紀錄筆數
MAX(欄位)	取得該欄位的最大值
MIN(欄位)	取得該欄位的最小值
SUM(欄位)	計算該欄位值總合

mySQL 內建函數 – AVG()

- 使用範例：
 - 計算書城中所有書本的平均售價
 - $(690 + 199 + 1299 + 880 + 590 + 990) / 6 = 774.66$
 - **SELECT AVG(booksPrice) AS average FROM books;**

average
774.6667

mySQL 內建函數 – COUNT()

- 使用範例：
 - 計算目前書城中的會員人數
 - 目測 customers 資料表中有 8 人
 - **SELECT COUNT(*) AS counting FROM customers;**

counting
8

mySQL 內建函數 – MAX()

- 使用範例：
 - 想知道目前書城中哪一本書的售價最高
 - **SELECT MAX(booksPrice) FROM books;**

MAX(booksPrice)

1299

mySQL 內建函數 – MIN()

- 使用範例：
 - 想知道書城中哪一本書的頁數最少
 - **SELECT MIN(booksPages) FROM books;**

MIN(booksPages)
250

mySQL 內建函數 – SUM()

- 使用範例：
 - 將書城中所有書的頁數全部加起來
 - $(400 + 250 + 980 + 605 + 475 + 332 = 3,042)$
 - **SELECT SUM(booksPages) FROM books;**

SUM(booksPages)
3042

群組 - GROUP

- 群組概念是將相同的欄位資料群集起來
 - 例：計算書城會員中，有多少種職業
 - 8個會員，共5種職業
 - (學生、上班族、家管、工程師、保母)
 - SELECT customersJob,
COUNT(customersJob) AS counting FROM
customers GROUP BY customersJob;**

customersJob	counting	customersName	customersJob
上班族	2	鷺鷥酒	學生
保母	1	沉水匾	上班族
學生	3	某櫻紋	家管
家管	1	消萬嘗	工程師
工程師	1	無蹲益	學生
		酥真鰨	保母
		洩常停	學生
		沉櫛	上班族

排序 – ORDER (1)

- 排序即是將欄位做升冪(ASC, 由小到大, 預設)或降冪(DESC, 由大到小)排序
 - 將書本資料以「售價」排序，便宜→貴
 - **SELECT * FROM books ORDER BY booksPrice;**

booksNo	booksName	booksAuthor	booksPrice ▲	booksPages	booksPublish
B334	一粥刊	Bill	199	250	2011-01-08
E990	還豬格格	Evelyn	590	475	2011-03-30
A112	蛤蠣波特	Allan	690	400	2011-09-15
D778	絕代霜蕉	David	880	605	2011-08-20
F012	我不是焦尼炸	Frank	990	332	2010-05-30
C556	天龍吼嘆	Cindy	1299	980	2011-02-06

排序 – ORDER (2)

- 將書本資料以「頁數」排序，由多→少
 - **SELECT * FROM books ORDER BY booksPages DESC;**

booksNo	booksName	booksAuthor	booksPrice	booksPages ▾	booksPublish
C556	天龍吼嘆	Cindy	1299	980	2011-02-06
D778	継代霜蕉	David	880	605	2011-08-20
E990	還豬格格	Evelyn	590	475	2011-03-30
A112	蛤蠣波特	Allan	690	400	2011-09-15
F012	我不是焦尼炸	Frank	990	332	2010-05-30
B334	一粥刊	Bill	199	250	2011-01-08

- 結合計算：用每一本書的印製成本排序
 - **SELECT booksPages, booksPrice, booksPages/booksPrice AS cost FROM books ORDER BY cost ASC;**

booksPages	booksPrice	cost
332	990	0.3354
400	690	0.5797
605	880	0.6875
980	1299	0.7544
475	590	0.8051
250	199	1.2563

跨表格查詢 (1)

- 查詢結合一個表格以上的欄位時，需以「**表格名.欄位名**」表示
 - books.booksName / orders.ordersNo
 - 例：
 - orders (訂單) 資料表中，光看：

ordersNo	ordersBook	ordersCustomer
0000000001	A112	8
0000000002	F012	5
0000000003	F012	3
0000000004	D778	3
0000000005	A112	1
0000000006	A112	2
0000000007	B334	3

- 無法得知誰 (ordersCustomer) 買了什麼書 (ordersBook)

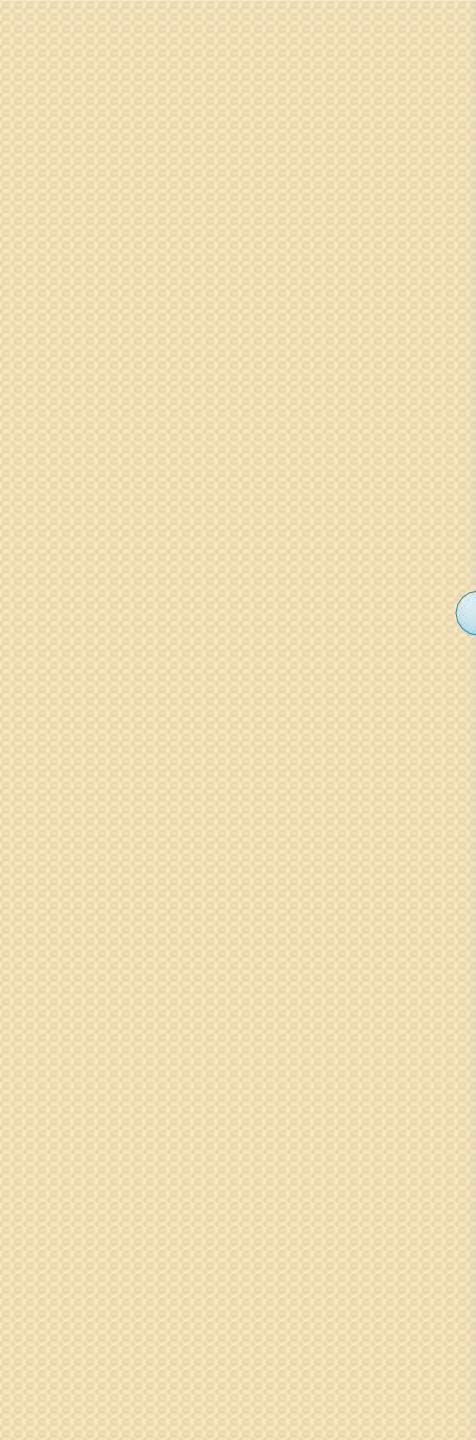
跨表格查詢 (2)

- `SELECT orders.ordersNo,
books.booksName,
customers.customersName FROM orders,
books, customers WHERE
orders.ordersBook = books.booksNo AND
orders.ordersCustomer =
customers.customersNo ORDER BY
orders.ordersNo;`

ordersNo	ordersBook	ordersCustomer
0000000001	A112	8
0000000002	F012	5
0000000003	F012	3
0000000004	D778	3
0000000005	A112	1
0000000006	A112	2
0000000007	B334	3



ordersNo	booksName	customersName
0000000001	蛤蠣波特	沉橘
0000000002	我不是焦尼炸	無蹲益
0000000003	我不是焦尼炸	茱櫻紋
0000000004	綰代霜蕉	茱櫻紋
0000000005	蛤蠣波特	鷓鴣酒
0000000006	蛤蠣波特	沉水區
0000000007	一粥刊	茱櫻紋



- 第四部份：PHP + MySQL

操作資料庫

- 基本步驟如下
 - 1. 建立連結
 - 連結MySQL伺服器
 - 2. 選擇指定資料庫
 - 選擇開啟使用的資料庫
 - 3. 送出查詢
 - 4. 取得結果
 - 5. 關閉連結

建立連結

- 建立連結使用mysql_connect()函數

- 語法

```
mysql_connect([主機名稱][:port][,使用者名稱][,密碼]);
```

- 範例

```
mysql_connect('localhost', 'root', 'student');
```

- 連結成功後會回傳連結指標，可以寫成

```
$link_ID= mysql_connect('localhost', 'root', 'student');
```

關閉連結

- 關閉連結使用mysql_close()函數
 - 語法

```
mysql_close(連結指標);
```

- 範例

```
mysql_close($link_ID);
```

指定資料庫

- 建立連結後，讓PHP知道存取的資料是從哪一個資料庫，使用mysql_select_db()
- 語法
 - `mysql_select_db(資料庫名稱, [連結指標]);`
- 這個函數就像mysql操作的USE指令
 - `mysql_select_db('bookstore');`
- 當連結指標沒指定時，會使用最後開啟的連結指標，此函數回傳真假值

送出查詢

- 送出查詢使用mysql_query()函數，是最常用的函數
- 語法
 - mysql_query(查詢字串, 連結指標);
- 查詢字串就是SQL敘述，本身是一個字串，也可以用變數代替

送出查詢

- 如果查詢字串是一個SELECT敘述，這個函數就會回傳一個查詢的結果指標
- 範例

```
$result=mysql_query("SELECT * FROM list;", $link_ID);
```

- 範例2

```
$str="SELECT * FROM list;";  
$result=mysql_query($str, $link_ID);
```

PHP程式碼撰寫(重要)

- 連結資料庫後，進行第一次資料庫操作前，先對資料庫query底下的命令
SET CHARACTER SET utf8;
- 這是通知MySQL的SQL Parser用utf-8編碼來處理SQL字串。
- SQL語法為
 - SET CHARACTER SET UTF8;

```
//使用 UTF8 編碼
```

```
mysql_query('SET CHARACTER SET UTF8;');
```

取得結果 – 陣列 (1)

- 取得結果主要可以用下列三種函數
 - mysql_fetch_row(結果指標)
 - 傳回型態是陣列，索引值從0開始
 - 範例 ex6-1

input

客戶資料查詢：

請在下列欄位輸入資料後按下 **查詢** 按鈕。

客戶姓名： **查詢**

output

客戶資料查詢結果 (使用 mysql_fetch_row)

主鍵編號：6
客戶姓名：酥真鯧
客戶職業：保母

取得結果 – 陣列 (2)

input – ex6-1.html

```
1 <html>
2 <head>
3 <title>取得查詢資料的使用範例 - ex6-1</title>
4 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5 </head>
6 <body>
7 <h2 align="center">客戶資料查詢 : </h2>
8 請在下列欄位輸入資料後按下
9 <font color="red">查詢</font>按鈕。
10 <br><br>
11 <form action="ex6-1.php" method="POST">
12   客戶姓名 :
13   <input type="text" name="customersName">
14   <input type="submit" value="查詢">
15 </form>
16 </body>
17 </html>
```

取得結果 – 陣列(3)

output – ex6-1.php

```
8 <?php
9 //連接MySQL伺服器
10 $link_ID = mysql_connect('localhost', 'root', 'student');
11
12 //指定使用bookstore資料庫
13 mysql_select_db('bookstore');
14
15 //使用 UTF8 編碼
16 mysql_query('SET CHARACTER SET UTF8;');
17
18 //送出查詢，將結果放入$result
19 $result = mysql_query("SELECT * FROM customers WHERE customersName = '". $_POST['customersName']."'");$_POST['customersName'], $link_ID);
20
21 #echo "SELECT * FROM customers WHERE customersName = '". $_POST['customersName']."'";<br>;
22
23 //關閉資料庫連接
24 mysql_close($link_ID);
25
26 //取得結果，以陣列傳回，放在$record中
27 $record = mysql_fetch_row($result);mysql_fetch_row
28
29 // $record[0] = customersNo 中的資料
30 // $record[1] = customersName 中的資料
31 // $record[2] = customersJob 中的資料
32
33 // 若取回的陣列為空陣列，即表示找不到這個人
34 // 即 $record == false
35 if ($record == false)
36 {
37     echo '查無此人！';
38 }
39 else
40 {
41     echo '主鍵編號：' . $record[0] . '<br>';
42     echo '客戶姓名：' . $record[1] . '<br>';
43     echo '客戶職業：' . $record[2] . '<br>';
44 }
?>
```

取得結果 – 字串索引陣列 (1)

- 用字串索引當陣列較好記憶
 - mysql_fetch_array(結果指標)
 - 傳回型態是陣列，索引值與欄位名稱相同
 - 範例 ex6-2

input

output

客戶資料查詢：

請在下列欄位輸入資料後按下 **查詢**按鈕。

客戶姓名： **查詢**

客戶資料查詢結果 (使用 **mysql_fetch_array**)

主鍵編號：2
客戶姓名：沉水區
客戶職業：上班族

取得結果 – 字串索引陣列 (2)

input – ex6-2.html

```
1  □<html>
2  □<head>
3  <title>取得查詢資料的使用範例 - ex6-2</title>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  </head>
6  □<body>
7  <h2 align="center">客戶資料查詢 : </h2>
8  請在下列欄位輸入資料後按下
9  <font color="red">查詢</font>按鈕。
10 <br><br>
11 □<form action="ex6-2.php" method="POST">
12  客戶姓名 :
13  <input type="text" name="customersName">
14  <input type="submit" value="查詢">
15  </form>
16  </body>
17  </html>
```

取得結果 – 字串索引陣列 (3)

output – ex6-2.php

```
8 <?php
9 //連接MySQL伺服器
10 $link_ID = mysql_connect('localhost', 'root', 'student');
11
12 //指定使用bookstore資料庫
13 mysql_select_db('bookstore');
14
15 //使用 UTF8 編碼
16 mysql_query('SET CHARACTER SET UTF8;');
17
18 //送出查詢，將結果放入$result
19 $result = mysql_query("SELECT * FROM customers WHERE customersName='$_POST['customersName']';", $link_ID);
20
21 //關閉資料庫連接
22 mysql_close($link_ID);
23
24 //取得查詢結果，以陣列傳回，放在$record
25 $record = mysql_fetch_array($result);
26
27 // $record['customersNo'] = customersNo 中的資料
28 // $record['customersName'] = customersName 中的資料
29 // $record['customersJob'] = customersJob 中的資料
30
31 // 若取回的陣列為空陣列，即表示找不到這個人
32 // 即 $record == false
33 // 或是 !$record ($record 若為 false 被 not 後則為 true)
34 if (!$record)
35 {
36     echo '查無此人!';
37 }
38 else
39 {
40     echo '主鍵編號：' . $record['customersNo'] . '<br>';
41     echo '客戶姓名：' . $record['customersName'] . '<br>';
42     echo '客戶職業：' . $record['customersJob'] . '<br>';
43 }
44 ?>
```

取得結果 – 物件 (1)

- 回傳物件型態的資料
 - mysql_fetch_object(結果指標)
 - 傳回型態是物件，屬性值與欄位名稱相同
 - 範例 ex6-3

input

客戶資料查詢：

請在下列欄位輸入資料後按下 **查詢**按鈕。

客戶姓名： **查詢**

output

客戶資料查詢結果 (使用 mysql_fetch_object)

主鍵編號：4
客戶姓名：消萬嘗
客戶職業：工程師

取得結果 – 物件 (2)

input – ex6-3.html

```
1  □<html>
2  □<head>
3  <title>取得查詢資料的使用範例 - ex6-3</title>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  </head>
6  □<body>
7  <h2 align="center">客戶資料查詢 : </h2>
8  請在下列欄位輸入資料後按下
9  <font color="red">查詢</font>按鈕。
10 <br><br>
11 □<form action="ex6-3.php" method="POST">
12  客戶姓名 :
13  <input type="text" name="customersName">
14  <input type="submit" value="查詢">
15  </form>
16  </body>
17  </html>
```

取得結果 – 物件 (3)

output – ex6-3.php

```
8 <?php
9 //連接MySQL伺服器
10 $link_ID = mysql_connect('localhost', 'root', 'student');
11
12 //指定使用bookstore資料庫
13 mysql_select_db('bookstore');
14
15 //使用 UTF8 編碼
16 mysql_query('SET CHARACTER SET UTF8;');
17
18 //送出查詢，將結果放入$result
19 $result = mysql_query("SELECT * FROM customers WHERE customersName='" . $_POST['customersName'] . "';", $link_ID);
20
21 //關閉資料庫連接
22 mysql_close($link_ID);
23
24 //取得查詢結果，以物件傳回，放在$record
25 $record = mysql_fetch_object($result);
26
27 // $record->customersNo = customersNo 中的資料
28 // $record->customersName = customersName 中的資料
29 // $record->customersJob = customersJob 中的資料
30
31 // 若取回的物件為空，即表示找不到這個人
32 // $record = false 或 !$record 皆可
33 if ($record == false)
34 {
35     echo '查無此人！';
36 }
37 else
38 {
39     echo '主鍵編號：' . $record->customersNo . '<br>';
40     echo '客戶姓名：' . $record->customersName . '<br>';
41     echo '客戶職業：' . $record->customersJob . '<br>';
42 }
43 ?>
```

取得資料筆數 (1)

- 取回傳回的結果有幾筆，使用 mysql_num_rows()函數

- 語法

```
mysql_num_rows(結果指標);
```

- 範例

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

- 範例 ex6-4

input

客戶資料查詢：

請在下列欄位輸入資料後按下 **查詢** 按鈕。

客戶職業：

查詢

output

客戶資料查詢結果

職業為「學生」的資料共 3 筆

取得資料筆數 (2)

input – ex6-4.html

```
1  <html>
2  <head>
3  <title>取得查詢資料筆數的使用範例 - ex6-4</title>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  </head>
6  <body>
7  <h2 align="center">客戶資料查詢 : </h2>
8  請在下列欄位輸入資料後按下
9  <font color="red">查詢</font>按鈕。
10 <br><br>
11 <form action="ex6-4.php" method="POST">
12  客戶職業 :
13  <input type="text" name="customersJob">
14  <input type="submit" value="查詢">
15  </form>
16  </body>
17  </html>
```

取得資料筆數 (3)

output – ex6-4.php

```
1  □ <html>
2  □ <head>
3  <title>取得查詢資料筆數的使用範例 - ex6-4</title>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  </head>
6  □ <body>
7  <h2 align="center">客戶資料查詢結果</h2>
8  □ <?php
9  //連接MySQL伺服器
10 $link_ID = mysql_connect('localhost', 'root', 'student');
11
12 //指定使用bookstore資料庫
13 mysql_select_db('bookstore');
14
15 //使用 UTF8 編碼
16 mysql_query('SET CHARACTER SET UTF8;');
17
18 //送出查詢，將結果放入$result
19 $result = mysql_query("SELECT * FROM customers WHERE customersJob='$_POST['customersJob']';", $link_ID);
20
21 //計算符合查詢結果的資料筆數
22 $count = mysql_num_rows($result);
23
24 //關閉資料庫連接
25 mysql_close($link_ID);
26
27 if ($count == 0)
28 {
29     echo '目前沒有職業為「'. $_POST['customersJob'] .」的客戶！';
30 }
31 else
32 {
33     echo '職業為「'. $_POST['customersJob'] .」的資料共 <font color="red">' . $count . '</font> 筆<br>';
34 }
35 ?>
36 </body>
37 </html>
```

列出所有資料 (1)

- 將查詢完的結果結合迴圈將所有資料輸出

客戶資料列表		
編號	姓名	職業
1	鷗鷺酒	學生
2	沉水匾	上班族
3	菜櫻紋	家管
4	消萬嘗	工程師
5	無蹲益	學生
6	酥真鮕	保母
7	洩常停	學生
8	沉橘	上班族

```
1 <html>
2 <head>
3 <title>多筆查詢結果的使用範例 - ex6-5</title>
4 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5 </head>
6 <body>
7 <?php
8 //連接MySQL伺服器
9 $link_ID = mysql_connect('localhost','root','student');
10
11 //指定使用bookstore資料庫
12 mysql_select_db('bookstore');
13
14 //使用 UTF8 編碼
15 mysql_query('SET CHARACTER SET UTF8;');
16
17 //送出查詢，將結果放入$result
18 $result = mysql_query("SELECT * FROM customers ORDER BY customersNo; ", $link_ID);
19
20 //關閉資料庫連接
21 mysql_close($link_ID);
22 ?>
23 <div align="center">
24 <h2>客戶資料列表</h2>
25 <hr>
26 <table align="center" border="1">
27 <tr align="center">
28 <td>編號</td>
29 <td width="100">姓名</td>
30 <td width="120">職業</td>
31 </tr>
32 <?php
33 //設定迴圈，執行一次印出一row
34 while($record = mysql_fetch_array($result))
35 {
36 ?>
37 <tr align="center">
38 <td><?php echo $record['customersNo'] ?></td>
39 <td width="100"><?php echo $record['customersName'] ?></td>
40 <td width="120"><?php echo $record['customersJob'] ?></td>
41 </tr>
42 <?php
43 }
44 ?>
45 </table>
46 </div>
47 </body>
48 </html>
```

新增資料 (1)

- 基本步驟如下

1. 建立連結

- 連結MySQL伺服器

2. 選擇指定資料庫

- 選擇開啟使用的資料庫

3. 送出查詢

- 送出INSERT

4. 關閉連結

新增資料 (2)

- 新增資料到資料庫的方式，就是送出新增資料的查詢
- 新增資料查詢使用INSERT 敘述
 - 範例 ex6-6

input

新增客戶資料

請在下列欄位輸入資料後按下 **新增**按鈕。

姓名： 職業： **新增**

output

新增完成！[觀看客戶列表](#)

7	洩常停	學生
8	沉橋	上班族
9	艾倫宋	乞丐

新增資料 (3)

input – ex6-6.html

```
1  □<html>
2  □<head>
3      <title>insert的使用範例 - ex6-6</title>
4      <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  □</head>
6  □<body>
7      <h2 align="center">新增客戶資料</h2>
8      <hr>
9      請在下列欄位輸入資料後按下
10     <font color="red">新增</font>按鈕。
11     <br><br>
12    □<form action="ex6-6.php" method="POST">
13        姓名 : <input type="text" name="customersName">
14        職業 : <input type="text" name="customersJob">
15        <input type="submit" value="新增">
16    □</form>
17    □</body>
18  □</html>
```

新增資料 (4)

output – ex6-6.php

```
1  <html>
2  <head>
3  <title>insert的使用範例 - ex6-6</title>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  </head>
6  <body>
7  <?php
8  //判斷姓名及職業欄位是否有輸入
9  if ($_POST['customersName'] && $_POST['customersJob'])
10 {
11     //連接MySQL伺服器
12     $link_ID = mysql_connect('localhost', 'root', 'student');
13
14     //使用bookstore資料庫
15     mysql_select_db('bookstore');
16
17     //使用 UTF8 編碼
18     mysql_query('SET CHARACTER SET UTF8;');
19
20     //送出查詢字串
21     mysql_query("INSERT INTO customers (customersName, customersJob) VALUES ('".$_POST['customersName']."' , '".$_POST['customersJob']."' ); ", $link_ID);
22
23     //關閉資料庫連接
24     mysql_close($link_ID);
25
26     echo '新增完成！<a href="../ex6-5/ex6-5.php">觀看客戶列表</a>';
27 }
28 else
29 {
30     echo '姓名與職業欄都必須填寫！<a href="javascript:go(-1);">回上一頁填寫</a>';
31 }
32 ?>
33 </body>
34 </html>
```

更新資料 (1)

- 基本步驟如下

1. 建立連結

- 連結MySQL伺服器

2. 選擇指定資料庫

- 選擇開啟使用的資料庫

3. 送出查詢

- UPDATE

4. 關閉連結

更新資料 (2)

- 更新資料到資料庫的方式，就是送出更新資料的查詢
- 更新資料查詢使用UPDATE 敘述
 - 範例 ex6-7

input

更新客戶職業資料

請在下列欄位輸入資料後按下 **更新**按鈕。

編號： 職業： **更新**

output

修改完成！[觀看客戶列表](#)

7	洩常停	學生
8	沉橋	上班族
9	艾倫宋	遊民

更新資料 (3)

input – ex6-7.html

```
1  <html>
2  <head>
3      <title>update的使用範例 - ex6-7</title>
4      <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  </head>
6  <body>
7      <h2 align="center">更新客戶職業資料</h2>
8      <hr>
9      請在下列欄位輸入資料後按下
10     <font color="red">更新</font>按鈕。
11     <br><br>
12     <form action="ex6-7.php" method="POST">
13         編號 : <input type="text" name="customersNo">
14         職業 : <input type="text" name="customersJob">
15         <input type="submit" value="更新">
16     </form>
17     </body>
18     </html>
```

更新資料 (4)

output – ex6-7.php

```
1  <html>
2  <head>
3  <title>update的使用範例 - ex6-7</title>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  </head>
6  <body>
7  <?php
8  //判斷是否有輸入編號與職業的 input 欄位
9  if ($_POST['customersNo'] && $_POST['customersJob'])
10 {
11     //連接MySQL伺服器
12     $link_ID = mysql_connect('localhost', 'root', 'student');
13
14     //使用bookstore資料庫
15     mysql_select_db('bookstore');
16
17     //使用 UTF8 編碼
18     mysql_query('SET CHARACTER SET UTF8;');
19
20     //設定查詢字串，將輸入的編號在資料表比對後、更新職業的值
21     mysql_query("UPDATE customers SET customersJob='" . $_POST['customersJob'] . "' WHERE customersNo='" . $_POST['customersNo'] . "' ", $link_ID);
22
23     //關閉資料庫連接
24     mysql_close($link_ID);
25
26     echo '修改完成！<a href="../ex6-5/ex6-5.php">觀看客戶列表</a>';
27 }
28 else
29 {
30     echo '編號與職業欄都必須填寫！<a href="javascript:go(-1);">回上一頁填寫</a>';
31 }
32
33 ?>
34 </body>
35 </html>
```

刪除資料 (1)

- 基本步驟如下

1. 建立連結

- 連結MySQL伺服器

2. 選擇指定資料庫

- 選擇開啟使用的資料庫

3. 送出查詢

- DELETE

4. 關閉連結

刪除資料 (2)

- 刪除資料到資料庫的方式，就是送出刪除資料的查詢
- 刪除資料查詢使用DELETE 敘述
 - 範例 ex6-8

input

刪除資料區

請在下列欄位內輸入資料後按下 **刪除**按鈕。

姓名： **刪除**

output

艾倫宋已被刪除！[觀看客戶列表](#)

6	酥真鰨	保母
7	洩常停	學生
8	沉橘	上班族

刪除資料 (3)

input – ex6-8.html

```
1  
2  
3      <title>delete的使用範例</title>
4      <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  
6  
7      <h2 align="center">刪除資料區</h2>
8      <hr>
9      請在下列欄位內輸入資料後按下
10     <font color="red">刪除</font>按鈕。
11     <br><br>
12     <form action="ex6-8.php" method="POST">
13         姓名 : <input type="text" name="customersName">
14         <input type="submit" value="刪除">
15     </form>
16     </body>
17     </html>
```

刪除資料 (4)

output – ex6-8.php

```
1  □ <html>
2  □ <head>
3  <title>delete的使用範例</title>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5  </head>
6  □ <body>
7  □ <?php
8  //下面if敘述將判斷輸入的姓名是否存在於資料庫中
9  //若存在於資料庫則執行刪除資料的動作
10 if ($_POST['customersName'])
11 {
12     //連接MySQL伺服器
13     $link_ID = mysql_connect('localhost', 'root', 'student');
14
15     //使用bookstore資料庫
16     mysql_select_db('bookstore');
17
18     //使用 UTF8 編碼
19     mysql_query('SET CHARACTER SET UTF8;');
20
21     //設定查詢字串，將輸入的姓名在資料表比對後刪除該筆記錄
22     $a = mysql_query("DELETE FROM customers WHERE customersName = '" . $_POST['customersName'] . "' ", $link_ID);
23
24     //關閉與資料庫的連接
25     mysql_close($link_ID);
26
27     echo $_POST['customersName'] . '已被刪除！<a href="../ex6-5/ex6-5.php">觀看客戶列表</a>';
28 }
29 else
30 {
31     echo '姓名欄必須填寫！<a href="javascript:go(-1);">回上一頁填寫</a>';
32 }
33 ?>
34 </body>
35 </html>
```