



最新

PHP + MySQL + AJAX 網頁程式設計

第 9 章

SQL 基本語法

本章重點

- 9 - 1 認識 SQL 敘述
- 9 - 2 SELECT 敘述
- 9 - 3 WHERE、ORDER BY、LIMIT 子句
- 9 - 4 多資料表查詢、JOIN 與子查詢
- 9 - 5 INSERT、UPDATE、DELETE 敘述
- 9 - 6 常用函式

9 - 1 認識 SQL 敘述

- 結構化查詢語言 SQL
- 關鍵字、子句與敘述
- 保留字與字串
- 指定資料表與欄位
- 在 phpMyAdmin 中執行 SQL 敘述

結構化查詢語言 SQL

- SQL (Structured Query Language, 一般習慣唸成 "sequel", 但正確的唸法應該是 "S-Q-L") 中文譯為結構化查詢語言。
- 它是目前關聯式資料庫管理系統所使用的查詢語言, 使用者可以利用 SQL 語法直接對關聯式資料庫進行存取與管理的操作。
- SQL 的基本語法是由一些簡單的英文句子所構成, 相當簡單易學。

結構化查詢語言 SQL

- 例如，我們要從 **books** 資料表中找出價格高於 400 元的書籍，並列出所有的欄位資料，用 SQL 語言來寫，只要下面幾行就可以了：

```
SELECT *           ← 查詢資料表中所有欄位
FROM books        ← 指定資料表
WHERE 價格 > 400 ; ← 挑選的條件
```

- 我們只需指出自己所要的資料、條件，根本不必知道資料庫是怎麼找到或整理資料的，可以說是相當省事。

關鍵字、子句與敘述

- SQL 語法的基礎是**子句** (clause), 子句中會包括一些**關鍵字** (keyword)。
- 關鍵字是對 MySQL 有特別意義的字, 例如 "SELECT"、"FROM" 與 "WHERE" ... 等。至於**敘述** (statement) 則是指一組可產生存取資料庫結果的子句集合。例如：

```
SELECT *  
FROM books  
WHERE 價格 > 400 ;
```

← 這是一個 SELECT 子句, 使用了 SELECT 關鍵字
← 這是一個 FROM 子句, 使用了 FROM 關鍵字
← 這是一個 WHERE 子句, 使用了 WHERE 關鍵字

敘述可以任意斷行, 但是最後面要以分號做爲結尾

關鍵字、子句與敘述

- 前面 3 行子句組合起來便成為一組敘述，其作用是：從 (FROM) **books** 資料表中，找出符合條件 (WHERE)。
- 即**價格**欄位超過 400 元的記錄，並將這些記錄的所有欄位都挑選 (SELECT) 出來。
- 但是，敘述不一定要由多個子句組成！如果一個子句就能獨立完成一件事，該子句就是一個敘述。

關鍵字、子句與敘述

■ 例如：

CREATE DATABASE MyDatabase ; ← 這個子句使用了 CREATE 與 DATABASE 這兩個關鍵字

- 上面敘述會建立一個名為 **MyDatabase** 的資料庫，此例中，一個子句就可以完成建立新資料庫的動作，所以這個子句也是一個敘述。

保留字與字串

- 在 MySQL 中，關鍵字、函式、與資料型別的名稱都有特別的意義，所以在 SQL 敘述中這些字屬於**保留字**，不可以直接用來做為資料庫、資料表、欄位等名稱。

- 例如：

```
CREATE DATABASE SELECT ;
```

- 如果想要建立名為 **SELECT** 的資料庫而執行以上敘述時，**SELECT** 對於 MySQL 而言是查詢資料的關鍵字，因此以上敘述會產生語法錯誤，無法正確建立資料庫。

用反引號明確定義名稱

- 為了避免混淆，建議不要使用保留字做為資料庫、資料表或欄位名稱。
- 不過如果真有其必要，或是因為保留字太多擔心不小心誤用，可以使用反引號 (`) 括住名稱，即可讓伺服器知道此為資料庫、資料表或欄位名稱，而非 SQL 關鍵字或函式。
- 例如想要將訂單資料庫取名為 `order`，但是 `ORDER` 卻是 SQL 敘述中用來排序的關鍵字，此時可如下建立資料庫：

```
CREATE DATABASE `order` ;
```

保留字與字串

- 在 SQL 敘述中需要使用字串時，必須使用單引號 (') 括住，例如要從 **employee** 資料表中找出 **性別** 欄位為女的員工，需執行以下敘述：

```
SELECT *  
FROM employee  
WHERE 性別 = '女' ;
```

字串必須用單引號括住

指定資料表與欄位

- 在 SQL 敘述中，可以使用『資料表名稱.欄位名稱』來指定要使用哪一個資料表的欄位，例如：

```
SELECT books.書籍名稱 ← 查詢 books 資料  
FROM books                表的書籍名稱欄位
```

- 當然在上例中，資料來源就只有 **books** 資料表，因此 **books.書籍名稱** 可以省略 **books** 不寫：

```
SELECT 書籍名稱  
FROM books ;
```

指定資料表與欄位

- 但是當我們在 SQL 敘述中同時使用多個資料表時，『資料表名稱.欄位名稱』的寫法才能分清各欄位是出自於哪一個資料表，9-4 節我們就會看到這樣的應用。

在 phpMyAdmin 中執行 SQL 敘述

- 本節將先透過 phpMyAdmin 的介面連線 MySQL 資料庫來執行 SQL 敘述，至於如何在 PHP 中執行 SQL 敘述，將留待第 10 章再行說明。
- 請依照 1 - 5 節的說明，以 root 帳號登入 phpMyAdmin，如下操作即可執行 SQL 敘述：

1 選擇要執行 SQL 敘述的資料庫

2 按 SQL 連結



在 phpMyAdmin 中執行 SQL 敘述



3 在此輸入要執行的 SQL 敘述

4 按此鈕執行

在 phpMyAdmin 中執行 SQL 敘述

這是剛剛輸入的 SQL 敘述 (LIMIT 關鍵字是 phpMyAdmin 自動加入, 詳見 9-3 節)

此處顯示查詢、刪除、或更新的記錄筆數

顯示記錄 0 - 11 (12 總計, 查詢需時 0.0002 秒)

SQL 語法

```
SELECT *  
FROM books  
LIMIT 0, 30
```

依鍵名排序: 不適用 [執行]

| 書籍編號 | 書籍名稱 | 價格 | 負責員工編號 |
|------|---------------------|--------|--------|
| 1 | Windows Server 系統實務 | 500.00 | 2 |
| 2 | Outlook 快學快用 | 350.00 | 4 |

按此鈕可以編輯該記錄

按此鈕可以刪除該記錄

下方顯示查詢出來的資料

9 - 2 SELECT 敘述

■ 基本語法

- 當我們要從資料庫中查詢資料時，必須使用 **SELECT** 敘述，SELECT 敘述的基本語法如下：

SELECT 欄位名稱 ← 查詢哪一個欄位

FROM 資料表名稱 ← 指定資料表

- 我們先來看一個簡單的應用範例，下面例子可以查詢 **books** 資料表中 **書籍名稱** 與 **價格** 兩個欄位的資料。

SELECT 敘述

SELECT 書籍名稱, 價格 ← 查詢書籍名稱與價格欄位
兩個名稱之間以逗號分隔
FROM books ; ← 在 books 資料表查詢



| 書籍名稱 | 價格 |
|---------------------|--------|
| Windows Server 系統實務 | 500.00 |
| Outlook 快學快用 | 350.00 |
| AutoCAD 電腦繪圖 | 450.00 |
| Word 使用手冊 | 300.00 |
| 抓住你的 Photoshop | 450.00 |
| Linux 架站實務 | 500.00 |
| EXECL 快速入門 | 350.00 |
| PHP 程式語言 | 460.00 |
| XOOPS 架站王 | 380.00 |
| 防火牆架設實務 | 480.00 |
| Linux 系統管理實務 | 450.00 |
| Windows 使用手冊 | 320.00 |

SELECT 敘述

- 若要查詢資料表中所有欄位，則可用星號 * 來代表：

```
SELECT *      ← 查詢所有欄位  
FROM books ;
```



| 書籍編號 | 書籍名稱 | 價格 | 負責員工編號 |
|------|---------------------|--------|--------|
| 1 | Windows Server 系統實務 | 500.00 | 2 |
| 2 | Outlook 快學快用 | 350.00 | 4 |
| 3 | AutoCAD 電腦繪圖 | 450.00 | 3 |
| 4 | Word 使用手冊 | 300.00 | 8 |
| 5 | 抓住你的 Photoshop | 450.00 | 1 |
| 6 | Linux 架站實務 | 500.00 | 2 |
| 7 | EXECL 快速入門 | 350.00 | 6 |
| 8 | PHP 程式語言 | 460.00 | 2 |

SELECT 敘述

■ 使用 AS 設定別名

- 在 SELECT 敘述中，我們可以替查詢結果的欄位取別名，以變更輸出的欄位名稱。指定別名時必須使用 **AS** 關鍵字，其語法如下：

```
SELECT 原本的欄位名稱 AS 別名  
FROM 資料表名稱
```

- 例如下面例子，便是利用別名讓欄位名稱變成英文。

SELECT 敘述

```
SELECT 書籍名稱 AS Title, 價格 AS Price  
FROM books;
```



| Title | Price |
|---------------------|--------|
| Windows Server 系統實務 | 500.00 |
| Outlook 快學快用 | 350.00 |
| AutoCAD 電腦繪圖 | 450.00 |
| Word 使用手冊 | 300.00 |
| 抓住你的 Photoshop | 450.00 |
| Linux 架站實務 | 500.00 |
| EXECL 快速入門 | 350.00 |
| PHP 程式語言 | 460.00 |
| XOOPS 架站王 | 380.00 |

欄位名稱變成我們指定的別名

SELECT 敘述

- 除了欄位名稱外，資料表名稱也可以設定別名，當敘述中出現多次資料表名稱時，設定資料表別名可以簡化輸入：

```
SELECT    employee.性別,    employee.姓名,    employee.電話  
FROM    employee;
```

↓ 改用別名

```
SELECT    e.性別,    e.姓名,    e.電話  
FROM    employee AS e; ← 設定 employee 資料表的別名為 e
```

SELECT 敘述

■ 欄位的運算

- SELECT 敘述中的欄位名稱也可以加上運算式，如此該欄位查詢到的資料都會依照運算式進行運算。
- 例如在 **books** 資料表中有一個**價格**欄位，而我們可以將其乘上折數後來產生折扣價。

SELECT 敘述

```
SELECT 書籍名稱, 價格 * 0.8  
FROM books;
```



| 書籍名稱 | 價格 * 0.8 |
|---------------------|----------|
| Windows Server 系統實務 | 400.000 |
| Outlook 快學快用 | 280.000 |
| AutoCAD 電腦繪圖 | 360.000 |
| Word 使用手冊 | 240.000 |
| 抓住你的 Photoshop | 360.000 |
| Linux 架站實務 | 400.000 |
| EXECL 快速入門 | 280.000 |
| PHP 程式語言 | 368.000 |
| XOOPS 架站王 | 304.000 |
| 防火牆架設實務 | 384.000 |
| Linux 系統管理實務 | 360.000 |
| Windows 使用手冊 | 256.000 |

欄位名稱變成
" 價格 * 0.8"

價格欄位內的
數字會乘
上 0.8

SELECT 敘述

- 如果您覺得上例中 **價格 * 0.8** 的欄位名稱不好看，可以利用 **AS** 將其設定為其他名稱：

```
SELECT 書籍名稱, 價格 * 0.8 AS 折扣價  
FROM books;
```

9 - 3 WHERE 、 ORDER BY 、 LIMIT 子句

- WHERE 子句
 - 基本語法
 - 比較運算子
 - AND 、 OR 、 NOT
 - BETWEEN
 - LIKE
- ORDER BY 子句
- LIMIT 子句

WHERE 子句

- 在基本的 SELECT 敘述後面，可以加上各種子句來限制查詢的範圍，其中最常見的是 WHERE 子句，其功能是設定查詢的條件。
 - 基本語法：WHERE 子句的語法如下：
SELECT 欄位名稱
FROM 資料表名稱
WHERE 條件
 - 例如我們想要從 **employee** 資料表中，找出所有女性員工的資料，就可以寫成下頁形式。

WHERE 子句

```
SELECT *  
FROM employee  
WHERE 性別 = '女';
```



| 員工編號 | 性別 | 姓名 | 電話 | 主管編號 |
|------|----|-----|------------|------|
| 1 | 女 | 陳圓圓 | 0223219845 | 2 |
| 2 | 女 | 劉敏敏 | 0246546777 | NULL |
| 4 | 女 | 蘇菲菲 | 0287658764 | 3 |
| 8 | 女 | 簡成琳 | 0358547646 | 1 |

- WHERE 子句可以使用 MySQL 的比較與邏輯運算子來限制查詢的條件，以下簡單介紹較常用的幾種運算子。

WHERE 子句

■ 比較運算子

- 以下是 MySQL 常用的比較運算子，用法相當直覺：

| 運算子 | 說明 | 運算子 | 說明 | 運算子 | 說明 |
|-----|-----|-----|-------|---------|----------|
| = | 等於 | > | 大於 | <= | 小於或等於 |
| <> | 不等於 | < | 小於 | IS NULL | 是否為NULL值 |
| != | 不等於 | >= | 大於或等於 | | |

■ AND、OR、NOT

- 這 3 個運算子為『且』、『或』、『否』的意思，可以組合出複雜的條件。

WHERE 子句

■ 例如：

```
SELECT *  
FROM books  
WHERE 價格 < 360 OR 價格 >= 500;
```



| 書籍編號 | 書籍名稱 | 價格 | 負責員工編號 |
|------|---------------------|--------|--------|
| 1 | Windows Server 系統實務 | 500.00 | 2 |
| 2 | Outlook 快學快用 | 350.00 | 4 |
| 4 | Word 使用手冊 | 300.00 | 8 |
| 6 | Linux 架站實務 | 500.00 | 2 |
| 7 | EXECL 快速入門 | 350.00 | 6 |
| 12 | Windows 使用手冊 | 320.00 | 7 |

查詢價格小於
360 或大於等
於 500 的書籍

WHERE 子句

■ BETWEEN

- 和字面的意思一樣，此運算子可以設定某個範圍的條件，範圍的上下限間要用 AND 連接。

```
SELECT *  
FROM books  
WHERE 價格 BETWEEN 400 AND 500;
```



| 書籍編號 | 書籍名稱 | 價格 | 負責員工編號 |
|------|---------------------|--------|--------|
| 1 | Windows Server 系統實務 | 500.00 | 2 |
| 3 | AutoCAD 電腦繪圖 | 450.00 | 3 |
| 5 | 抓住你的 Photoshop | 450.00 | 1 |
| 6 | Linux 架站實務 | 500.00 | 2 |
| 8 | PHP 程式語言 | 460.00 | 2 |
| 10 | 防火牆架設實務 | 480.00 | 1 |
| 11 | Linux 系統管理實務 | 450.00 | 5 |

查詢價格在
400 ~ 500
之間的書籍

WHERE 子句

- 上例的 WHERE 條件與『價格 \geq 400 AND 價格 \leq 500』的意思相同。
- 另外, 也可以加上 NOT, 例如『價格 NOT BETWEEN 400 AND 500』, 查詢『不在』範圍內的資料。

■ LIKE

- LIKE 運算子可以用部分字串來找尋記錄, 一般會搭配以下萬用字元:

| 萬用字元 | 代表意義 |
|--------|--|
| _ (底線) | 表示1個不確定的字元 (一個中文字也算一個字元)。例如『產品名稱 LIKE '_ _筆』, 可以找出 "原子筆"、"螢光筆" 等記錄 |

WHERE 子句

%

代表零或多個字元,用於不確定有幾個字元時。例如『產品名稱 LIKE '鋼%'』,可以找出"鋼"開頭的所有記錄(如:鋼筆、鋼珠筆);『產品名稱 LIKE '%鋼%'』,則可找出包含"鋼"這個字(不論位置)的記錄(如:鋼筆、大鋼筆)

- 下面範例將查詢書名倒數第二個字為"實"的書籍：

```
SELECT *  
FROM books  
WHERE 書籍名稱 LIKE '%實 _';
```



| 書籍編號 | 書籍名稱 | 價格 | 負責員工編號 |
|------|---------------------|--------|--------|
| 1 | Windows Server 系統實務 | 500.00 | 2 |
| 6 | Linux 架站實務 | 500.00 | 2 |
| 10 | 防火牆架設實務 | 480.00 | 1 |
| 11 | Linux 系統管理實務 | 450.00 | 5 |

ORDER BY 子句

- ORDER BY 子句可以將查詢的結果排序，語法如下：

```
SELECT 欄位名稱  
FROM 資料表名稱  
ORDER BY 用以排序的欄位名稱 排序方式
```

- 排序方式分成 ASC (升冪, 由小而大) 與 DESC (降冪, 由大而小) 兩種, 若未指定, 則預設值為 ASC。
- 下面範例會依照價格欄位, 由大到小排序書籍。

ORDER BY 子句

```
SELECT *  
FROM books  
ORDER BY 價格 DESC;
```



所有記錄依價格
由大到小排序



| 書籍編號 | 書籍名稱 | 價格 | 負責員工編號 |
|------|---------------------|--------|--------|
| 1 | Windows Server 系統實務 | 500.00 | 2 |
| 6 | Linux 架站實務 | 500.00 | 2 |
| 10 | 防火牆架設實務 | 480.00 | 1 |
| 8 | PHP 程式語言 | 460.00 | 2 |
| 11 | Linux 系統管理實務 | 450.00 | 5 |
| 3 | AutoCAD 電腦繪圖 | 450.00 | 3 |
| 5 | 抓住你的 Photoshop | 450.00 | 1 |
| 9 | XOOPS 架站王 | 380.00 | 8 |
| 7 | EVEOL 快速入門 | 350.00 | 6 |

ORDER BY 子句

- 在 ORDER BY 子句中可指定多個排序 (用逗號分隔), 例如 :

```
SELECT *  
FROM books  
ORDER BY 價格 DESC, 書籍名稱;
```

← 先以價格做降冪排序, 價格相同的資料再以書籍名稱做升冪排序

LIMIT 子句

- LIMIT 子句可以指定查詢的筆數範圍，例如我們使用 SELECT 查詢時，如果只想要取得查詢結果的第 11~20 筆記錄，便需要使用 LIMIT 子句。
- 其語法如下：
SELECT 欄位名稱
FROM 資料表名稱
LIMIT 啓始筆數, 要取得的筆數
- 請注意，啓始筆數是從 0 開始計算，例如下面範例會取得查詢結果的第 1~5 筆記錄。

LIMIT 子句

```
SELECT *  
FROM books  
ORDER BY 價格 DESC ← 依照價格由高至低排列  
LIMIT 0, 5 ;
```

從查詢結果的第 1 筆開始，取得 5 筆記錄，可以找出價格最高的 5 本書

| 書籍編號 | 書籍名稱 | 價格 | 負責員工編號 |
|------|---------------------|--------|--------|
| 1 | Windows Server 系統實務 | 500.00 | 2 |
| 6 | Linux 架站實務 | 500.00 | 2 |
| 10 | 防火牆架設實務 | 480.00 | 1 |
| 8 | PHP 程式語言 | 460.00 | 2 |
| 11 | Linux 系統管理實務 | 450.00 | 5 |

LIMIT 子句

- 啟始筆數是可以省略的參數，如果省略，則表示從查詢結果的第 1 筆開始取資料，所以上面 LIMIT 子句也可以改成『LIMIT 5』。
- LIMIT 子句通常會搭配 ORDER BY 子句，用來將查詢排序後，取出某個範圍的記錄。
- 如上例中將書籍依價格由高至低排列，即可使用 LIMIT 子句找出價格最高的 5 本書。

LIMIT 子句

- 此外, LIMIT 子句也常用於分頁顯示, 例如購物網站可能有上百種商品, 所以顯示商品時會採用分頁的方式。
- 假設每頁顯示 10 種商品, 則會以下面方式從資料庫取得記錄：

| 頁數 | 資料筆數 | LIMIT 子句 |
|----|---------|--------------|
| 1 | 1 ~ 10 | LIMIT 0, 10 |
| 2 | 11 ~ 20 | LIMIT 10, 10 |
| 3 | 21 ~ 30 | LIMIT 20, 10 |

9 - 4 多資料表查詢、JOIN 與子查詢

- 多資料表查詢
 - 多資料表查詢的原理
 - 實際範例
- JOIN
- 子查詢 Subquery

多資料表查詢

- 除了在一個資料表中查詢，**SELECT** 敘述還可以從多個相關（不一定要建立關聯）的資料表中取出資料，用法相當有彈性。
 - 多資料表查詢的原理：下面是 A、B 兩個資料表的內容：

A 資料表

| 書籍編號 | 書籍名稱 | 負責員工編號 |
|------|--------------|--------|
| 1 | Windows 使用手冊 | 2 |
| 2 | Linux 架站實務 | 1 |
| 3 | SQL 指令寶典 | 2 |

多資料表查詢

B 資料表

| 員工編號 | 姓名 |
|------|-----|
| 1 | 張大頭 |
| 2 | 陳小小 |

- 我們可以在 **SELECT** 敘述的 **FROM** 子句中指定多個資料表，即可進行多資料表的查詢。

多資料表查詢

SELECT * ← 從 A、B 兩個資料表查詢
FROM A, B ;

↓
用逗號分隔資料表

| 書籍編號 | 書籍名稱 | 負責員工編號 | 員工編號 | 姓名 |
|------|--------------|--------|------|-----|
| 1 | Windows 使用手冊 | 2 | 1 | 蘇大頭 |
| 1 | Windows 使用手冊 | 2 | 2 | 陳小小 |
| 2 | Linux 架站實務 | 1 | 1 | 蘇大頭 |
| 2 | Linux 架站實務 | 1 | 2 | 陳小小 |
| 3 | SQL 指令寶典 | 2 | 1 | 蘇大頭 |
| 3 | SQL 指令寶典 | 2 | 2 | 陳小小 |

多資料表查詢

- 上面可以看到，進行多資料表的查詢時，A 資料表的每一筆記錄都會和 B 資料表的每筆記錄連接為一筆新記錄，而產生了 $3 * 2 = 6$ 筆的記錄。
- 多資料表查詢所產生的記錄並不是每一筆都是我們想要的，如上例中只有 3 筆 (加框線的記錄) 是有意義的資料。
- 此時可以使用 WHERE 子句來限制條件，即可將查詢限制在我們需要的資料上。

多資料表查詢

```
SELECT *  
FROM A, B  
WHERE 負責員工編號 = 員工編號;
```

← 限制查詢條件為兩資料表中員工編號相同的記錄



| 書籍編號 | 書籍名稱 | 負責員工編號 | 員工編號 | 姓名 |
|------|--------------|--------|------|-----|
| 1 | Windows 使用手冊 | 2 | 2 | 陳小小 |
| 2 | Linux 架站實務 | 1 | 1 | 蘇大頭 |
| 3 | SQL 指令寶典 | 2 | 2 | 陳小小 |

■ 實際範例

- 接著就來看看多資料表查詢的實際範例，下面是 **books** 與 **employee** 資料表的內容。

多資料表查詢

| 書籍名稱 | 價格 | 負責員工編號 |
|---------------------|--------|--------|
| Windows Server 系統實務 | 500.00 | 2 |
| Outlook 快學快用 | 350.00 | 4 |
| AutoCAD 電腦繪圖 | 450.00 | 3 |
| Word 使用手冊 | 300.00 | 8 |
| 抓住你的 Photoshop | 450.00 | 1 |
| Linux 架站實務 | 500.00 | 2 |
| EXECL 快速入門 | 350.00 | 6 |
| PHP 程式語言 | 460.00 | 2 |
| XOOPS 架站王 | 380.00 | 8 |
| 防火牆架設實務 | 480.00 | 1 |
| Linux 系統管理實務 | 450.00 | 5 |
| Windows 使用手冊 | 320.00 | 7 |

books 資料表

| 員工編號 | 性別 | 姓名 | 電話 | 主管編號 |
|------|----|-----|------------|------|
| 1 | 女 | 陳圓圓 | 0223219845 | 2 |
| 2 | 女 | 劉敏敏 | 0246546777 | NULL |
| 3 | 男 | 劉國城 | 0246465465 | 2 |
| 4 | 女 | 蘇菲菲 | 0287658764 | 3 |
| 5 | 男 | 郭逸洋 | 0354686546 | 1 |
| 6 | 男 | 邱大熊 | 0266873546 | 1 |
| 7 | 男 | 王國維 | 0688643546 | 3 |
| 8 | 女 | 簡成琳 | 0358547646 | 1 |

employee 資料表

多資料表查詢

- 假如我們想要從 **books** 與 **employee** 資料表中, 找出所有書籍及其負責人的資料, 則可以如下查詢:

```
SELECT 書籍名稱, 價格, 姓名, 電話  
FROM books, employee  
WHERE 負責員工編號 = 員工編號;
```

← 限制查詢條件為兩資料表中員工編號相同的記錄



| 書籍名稱 | 價格 | 姓名 | 電話 |
|---------------------|--------|-----|------------|
| Windows Server 系統實務 | 500.00 | 劉敏敏 | 0246546777 |
| Outlook 快學快用 | 350.00 | 蘇菲菲 | 0287658764 |
| AutoCAD 電腦繪圖 | 450.00 | 劉國城 | 0246465465 |
| Word 使用手冊 | 300.00 | 簡成琳 | 0358547646 |
| 抓住你的 Photoshop | 450.00 | 陳圓圓 | 0223219845 |
| Linux 架站實務 | 500.00 | 劉敏敏 | 0246546777 |

多資料表查詢

- 我們也可以在 WHERE 子句中使用多個限制條件來查詢，例如下面將查詢所有 Linux 書籍的負責人：

```
SELECT 書籍名稱, 姓名, 電話  
FROM books, employee  
WHERE 書籍名稱 LIKE '%Linux%'  
      AND 負責員工編號 = 員工編號;
```



| ←T→ | | |
|--------------|-----|------------|
| 書籍名稱 | 姓名 | 電話 |
| Linux 架站實務 | 劉敏敏 | 0246546777 |
| Linux 系統管理實務 | 郭逸洋 | 0354686546 |

多資料表查詢

- 上述範例中，因為兩個資料表使用不同的欄位名稱，所以 WHERE 子句可以僅指定欄位名稱，而不會產生分不清楚是哪一個資料表的問題。
- 反之，如果不同資料表內使用相同名稱的欄位，那麼便必須使用『資料表名稱.欄位名稱』的方式來指定資料表欄位。
- 下面例子為您示範指定資料表欄位的寫法：

```
SELECT  books.書籍名稱,    employee.姓名,    employee.電話  
FROM    books,    employee  
WHERE   books.負責員工編號 = employee.員工編號;
```

多資料表查詢

- 上面寫法顯得相當繁雜，此時可以使用別名來簡化：

```
SELECT  b.書籍名稱, e.姓名, e.電話  
FROM    books AS b, employee AS e  
WHERE   b.負責員工編號 = e.員工編號;
```

JOIN

- JOIN 的意義是將多個資料表的記錄橫向連接起來，然後利用 ON 來設定條件以過濾不需要的記錄，其實前述多資料表查詢就是 JOIN，只是省略了 JOIN 的關鍵子。
- 例如前面的例子可以如下改用 JOIN：

```
SELECT *  
FROM books, employee  
WHERE 負責員工編號 = 員工編號;
```



改用 JOIN

```
SELECT *  
FROM books JOIN employee  
ON 負責員工編號 = 員工編號;
```

JOIN

- 雖然兩個可以得到相同的結果，可是比較起來，用 JOIN...ON... 的方式較具有可讀性，因為 JOIN 的條件不用和其他查詢條件混在一起：

```
SELECT 書籍名稱, 姓名, 電話  
FROM books, employee  
WHERE 書籍名稱 LIKE '%Linux%'  
      AND 負責員工編號 = 員工編號;
```

多資料表查詢的
條件與其他限制
條件混在一起



改用 JOIN

```
SELECT 書籍名稱, 姓名, 電話  
FROM books JOIN employee  
      ON 負責員工編號 = 員工編號  
WHERE 書籍名稱 LIKE '%Linux%';
```

WHERE 子句僅需
設定限制條件

JOIN

- 除了可讀性較高以外，JOIN 還有多種方式可運用，例如 LEFT JOIN、CROSS JOIN... 等，使用上較具彈性。
- 由於篇幅所限，此處不加以說明 JOIN 的各式種類，若您想要深入瞭解，請參考 <http://dev.mysql.com/doc/refman/5.0/en/join.html> 網頁。

子查詢 Subquery

- 所謂子查詢 (Subquery), 是指包含在主要查詢中的另一個 SELECT 查詢。
- 通常我們會利用子查詢先挑選出部份資料, 以做為主要查詢的資料來源或選取條件。
- 子查詢的語法和 SELECT 敘述一樣, 但整個子查詢敘述需用小括弧 () 括住, 我們來看個子查詢的應用範例, 下面例子將找出劉敏敏所負責的所有書籍。

子查詢 Subquery

```
SELECT *  
FROM books  
WHERE 負責員工編號 = ( SELECT 員工編號  
                        FROM employee  
                        WHERE 姓名 = '劉敏敏' );
```

這個子查詢先
找出劉敏敏的
員工編號



| 書籍編號 | 書籍名稱 | 價格 | 負責員工編號 |
|------|---------------------|--------|--------|
| 1 | Windows Server 系統實務 | 500.00 | 2 |
| 6 | Linux 架站實務 | 500.00 | 2 |
| 8 | PHP 程式語言 | 460.00 | 2 |

子查詢 Subquery

- 上例中，我們的目的是要從 **book** 資料表查出劉敏敏所負責的所有書籍，可是每本書只有負責員工的編號，員工的詳細資料是儲存在 **employee** 資料表中。
- 所以先利用子查詢在 **employee** 資料表中找出劉敏敏的員工編號，然後再根據此員工編號找出其負責的所有書籍。

9 - 5 INSERT、UPDATE、DELETE 敘述

- 前面介紹了許多查詢資料的語法，瞭解如何讀取資料之後，再來自然是輸入資料。
- 本節會為您說明資料編輯 (插入、更新、刪除) 的 SQL 語法。
 - 新增記錄 — INSERT 敘述
 - 更新記錄 — UPDATE 敘述
 - 刪除記錄 — DELETE 敘述

新增記錄 — INSERT 敘述

- 首先介紹為資料表新增記錄的 INSERT 敘述。
基本語法如下：

```
INSERT [INTO] 資料表名稱 [ (欄位名稱 1, 欄位名稱 2, 欄位名稱 3...) ]  
VALUES ( 欄位值 1, 欄位值 2, 欄位值 3... )
```

- 我們就利用上述的語法，替 **employee** 資料表新增記錄。
- 為方便對照欄位的屬性，先將 **employee** 資料表的結構列示如下。

新增記錄 — INSERT 敘述

設定了 `auto_increment` 屬性 (參見 8-25 頁), 表示此欄位會自動編號, 所以新增記錄時不需要輸入此欄位的值

| 欄位 | 型態 | 校對 | 屬性 | Null | 預設值 | 附加 |
|------|-------------|-----------------|----|------|------|----------------|
| 員工編號 | int(11) | | | 否 | | auto_increment |
| 性別 | char(1) | utf8_unicode_ci | | 否 | | |
| 姓名 | varchar(10) | utf8_unicode_ci | | 否 | | |
| 電話 | varchar(10) | utf8_unicode_ci | | 否 | | |
| 主管編號 | int(11) | | | 是 | NULL | |

此欄位允許 NULL, 所以新增記錄時可輸入也可不輸入此欄位的值

- 接著利用 INSERT 敘述為 **employee** 資料表新增兩筆記錄。

新增記錄 — INSERT 敘述

```
INSERT employee ( 性別, 姓名, 電話, 主管編號 )  
VALUES ( '男', '吳風', '0223963257', 2 );
```

```
INSERT employee ( 性別, 姓名, 電話 )  
VALUES ( '女', '許淳梅', '0235408731' );
```

```
SELECT * FROM employee;
```



| 員工編號 | 性別 | 姓名 | 電話 | 主管編號 |
|------|----|-----|------------|------|
| 1 | 女 | 陳圓圓 | 0223219845 | 2 |
| 2 | 女 | 劉敏敏 | 0246546777 | NULL |
| 3 | 男 | 劉國城 | 0246465465 | 2 |
| 4 | 女 | 蘇菲菲 | 0287658764 | 3 |
| 5 | 男 | 郭逸洋 | 0354686546 | 1 |
| 6 | 男 | 邱大熊 | 0266873546 | 1 |
| 7 | 男 | 王國維 | 0688643546 | 3 |
| 8 | 女 | 簡成琳 | 0358547646 | 1 |
| 9 | 男 | 吳風 | 0223963257 | 2 |
| 10 | 女 | 許淳梅 | 0235408731 | NULL |

剛剛新增的
兩筆記錄

新增記錄 — INSERT 敘述

- 上述兩個 INSERT 敘述中有些欄位被省略了，對於沒有被指定資料的欄位，MySQL 會如下處理：
 - 如果欄位設定了 `auto_increment` 屬性，那麼該欄位將自動累加編號。
 - 如果欄位有設定預設值，則填入預設值。
 - 如果欄位允許 `NULL`，則填入 `NULL`。
 - 若前幾項都不符合時，則會顯示錯誤訊息而取消操作，不輸入任何資料。

更新記錄 — UPDATE 敘述

- 我們可以使用 UPDATE 敘述來更新、編輯資料表中的記錄，基本語法如下：

UPDATE 資料表名稱

SET 欄位名稱 1 = 欄位值, ← 如果只更新一個欄位，則不需加上逗號
 欄位名稱 2 = 欄位值,
 欄位名稱 3 = 欄位值,

...

[WHERE 子句] ← 指定要更新的記錄

- 例如下面敘述可以修改 **employee** 資料表中的記錄。

更新記錄 — UPDATE 敘述

```
UPDATE employee
SET 性別 = '男',
    姓名 = '張西西',
    電話 = '0628439647'
WHERE 員工編號 = 10;
```

修改性別、姓名、電話

← 更新員工編號為 10 的員工資料

```
UPDATE employee
SET 主管編號 = 8
WHERE 主管編號 = 2;
```

修改所有主管編號為 2 的記錄，
將主管編號都改成 8

```
SELECT * FROM employee;
```



更新記錄 — UPDATE 敘述

| 員工編號 | 性別 | 姓名 | 電話 | 主管編號 |
|------|----|-----|------------|------|
| 1 | 女 | 陳圓圓 | 0223219845 | 8 |
| 2 | 女 | 劉敏敏 | 0246546777 | NULL |
| 3 | 男 | 劉國城 | 0246465465 | 8 |
| 4 | 女 | 蘇菲菲 | 0287658764 | 3 |
| 5 | 男 | 郭逸洋 | 0354686546 | 1 |
| 6 | 男 | 邱大熊 | 0266873546 | 1 |
| 7 | 男 | 王國維 | 0688643546 | 3 |
| 8 | 女 | 簡成琳 | 0358547646 | 1 |
| 9 | 男 | 吳風 | 0223963257 | 8 |
| 10 | 男 | 張西西 | 0628439647 | NULL |

這些欄位的值
已經被修改了

更新記錄 — UPDATE 敘述

- 設定新的欄位值時，可以引用同一欄位或是其他欄位的值來做變化。
- 例如下面敘述可以將 **books** 資料表中所有 Linux 書籍的**價格**提高 10%：

```
UPDATE books
SET 價格 = 價格 * 1.1
WHERE 書籍名稱 LIKE '%Linux%';
```

刪除記錄 — DELETE 敘述

- 如果要刪除資料表中的部分記錄, 可以使用 DELETE 敘述, 其語法如下:

DELETE FROM 資料表名稱

[WHERE 子句] ← 指定要刪除的記錄

- 例如下面敘述將刪除 **employee** 資料表中的記錄:

DELETE FROM employee

WHERE 員工編號 = 9; ← 刪除員工編號 9 的記錄

DELETE FROM employee

WHERE 姓名 LIKE '劉%'; ← 刪除所有劉姓員工

9 - 6 常用函式

- 除了前面各節介紹的 SQL 敘述外, MySQL 還提供了許多好用的函數, 可以方便我們統計、處理資料。
- 只要善用這些函數, 便可以幫助我們直接取得或計算想要的資料, 而不需要自行撰寫程式來處理。
- 本節將為您介紹常用的 MySQL 函式。

隨機數字函數

- MySQL 的 RAND() 函式可以產生 0~1.0 之間的隨機浮點數：

```
SELECT RAND( ) ;
```



| RAND() |
|------------------|
| 0.38568092732288 |

- 在 MySQL 中，如果使用 "ORDER BY RAND()" 的語法，則可將查詢結果隨機排序。

隨機數字函數

```
SELECT *  
FROM employee  
ORDER BY RAND( );
```

← 將查詢結果隨機排序



| 員工編號 | 性別 | 姓名 | 電話 | 主管編號 |
|------|----|-----|------------|------|
| 1 | 女 | 陳圓圓 | 0223219845 | 2 |
| 6 | 男 | 邱大熊 | 0266873546 | 1 |
| 7 | 男 | 王國維 | 0688643546 | 3 |
| 4 | 女 | 蘇菲菲 | 0287658764 | 3 |
| 8 | 女 | 簡成琳 | 0358547646 | 1 |
| 5 | 男 | 郭逸洋 | 0354686546 | 1 |
| 2 | 女 | 劉敏敏 | 0246546777 | NULL |
| 3 | 男 | 劉國城 | 0246465465 | 2 |

各記錄的
順序隨機
排列

隨機數字函數

- 如果想在資料表中隨機取出 n 筆記錄, 例如抽獎時需要隨機取出 n 個使用者, 此時可以加上『LIMIT n』來限制範圍。
- 下面例子會隨機從 **employee** 資料表取出 3 筆記錄：

```
SELECT *  
FROM employee  
ORDER BY RAND()  
LIMIT 3;
```

彙總函數

- 以下函式可以用來統計欄位的最大、最小、平均...等值：

| 函數 | 功能 |
|-----------|-----------|
| AVG(欄位) | 計算該欄位的平均值 |
| COUNT(欄位) | 計算該欄位的筆數 |
| MAX(欄位) | 計算該欄位的最大值 |
| MIN(欄位) | 計算該欄位的最小值 |
| SUM(欄位) | 計算該欄位值的總和 |

彙總函數

■ 範例如下：

```
SELECT COUNT(書籍編號) AS 書籍數量,  
       AVG(價格) AS 價格平均,  
       MAX(價格) AS 最高價格,  
       MIN(價格) AS 最低價格,  
       SUM(價格) AS 價格總和  
FROM books;
```



| 書籍數量 | 價格平均 | 最高價格 | 最低價格 | 價格總和 |
|------|------------|--------|--------|---------|
| 12 | 415.833333 | 500.00 | 300.00 | 4990.00 |