

目次

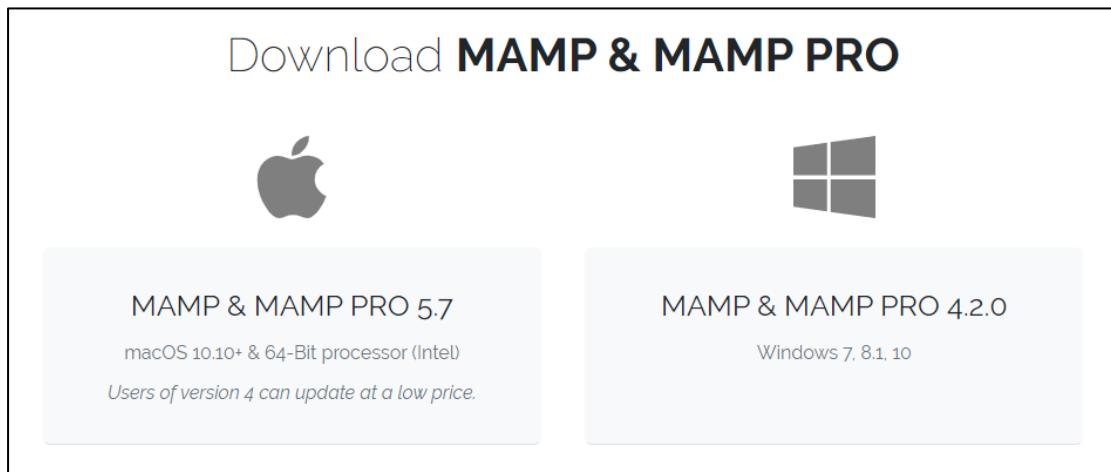
一、MAMP 安裝 (Windows 版本)	1
二、XAMPP 安裝 (上課使用)	9
三、快速導覽.....	30
四、資料庫設計議題.....	39
五、建立資料庫、資料表.....	45
六、新增、修改、刪除資料.....	56
七、資料庫基本查詢.....	59
八、進階查詢.....	66
九、函式.....	73
十、完整性原則與關聯介紹.....	79
十一、觸發器 (Trigger)	80
十二、事件 (Event)	81
十三、子查詢 (Sub Query)	84
十四、檢視表 (View)	85
十五、實務操作 (一)	89
十六、實務操作 (二)	96
十七、實務操作 (三)	100
十八、網頁整合資料庫.....	110
作業.....	122

一、MAMP 安裝 (Windows 版本)

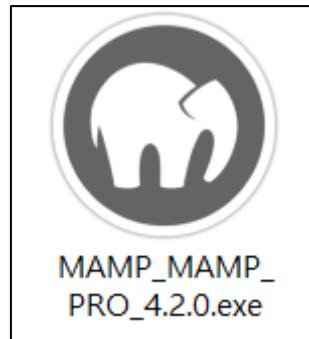
MAMP 是 macOS、Apache、MySQL(MariaDB)、PHP (或 Perl、Python) 的縮寫，通常被視為一個套裝的架站平台，廣泛地整合了許多主流的 open source 工具或技術，是 XAMPP 在 macOS 上的替選方案，我們也可以安裝在 Windows 的環境。



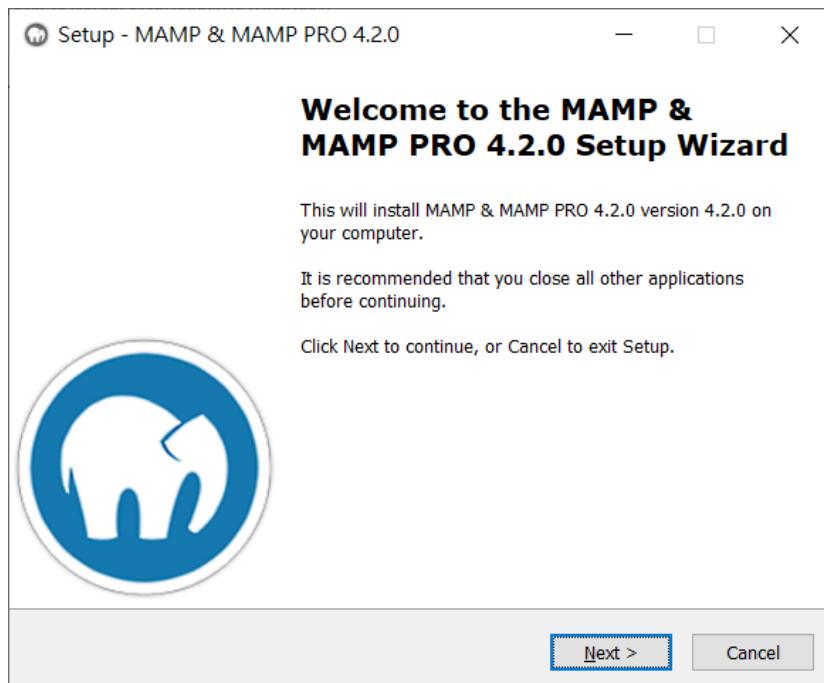
(圖) MAMP 首頁，按下 Free Download



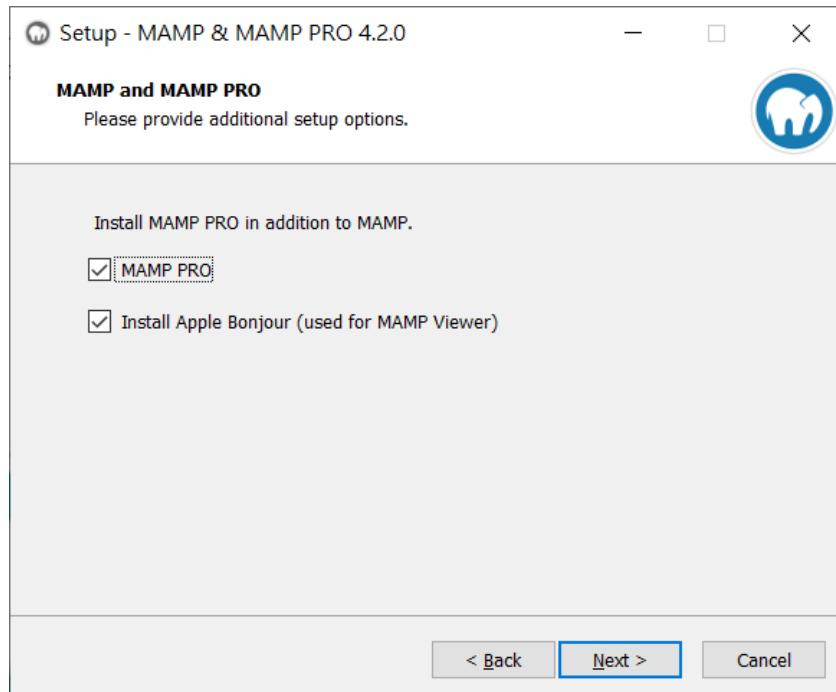
(圖) 選擇合適的作業系統，並按下連結



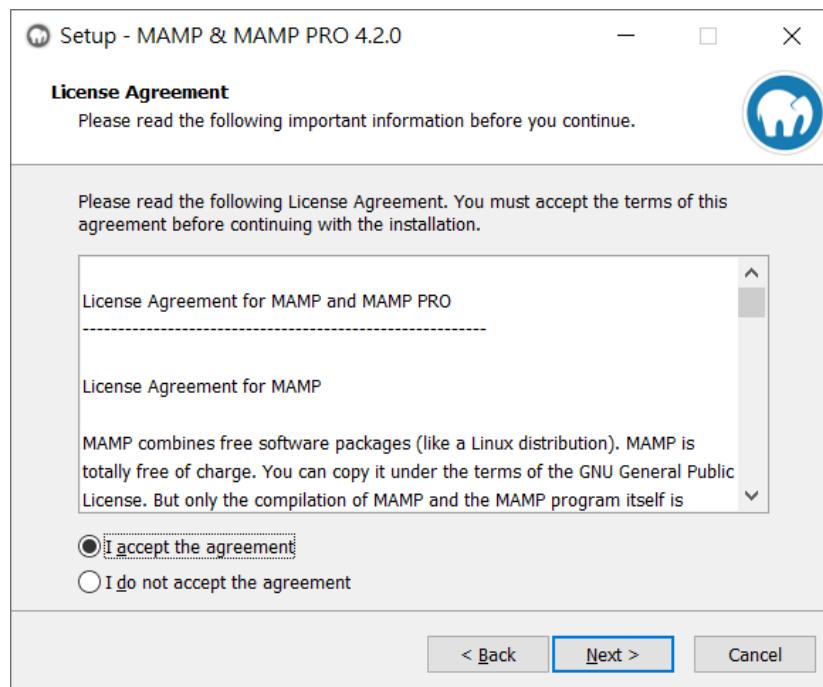
(圖) 下載後的檔案名稱



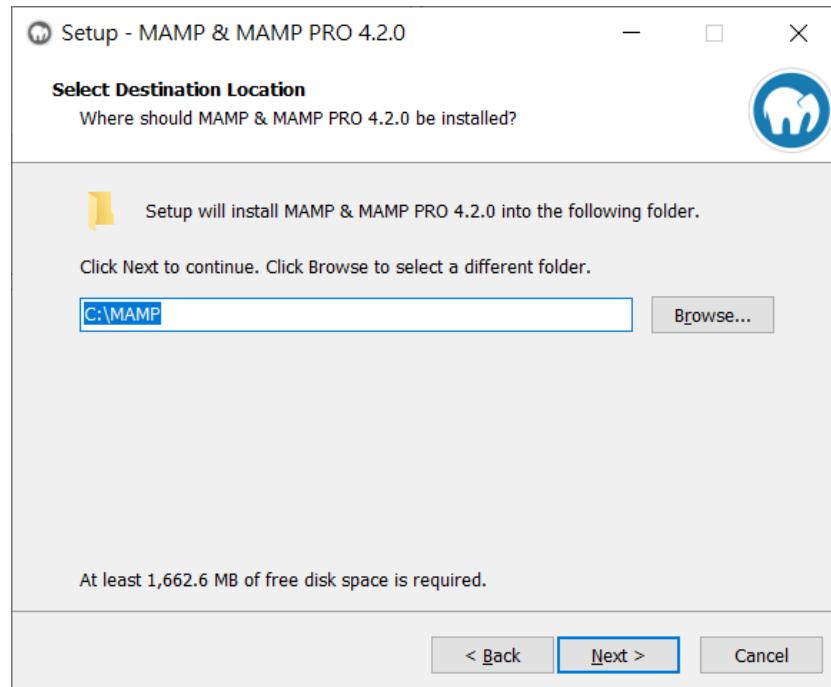
(圖) 按 Next



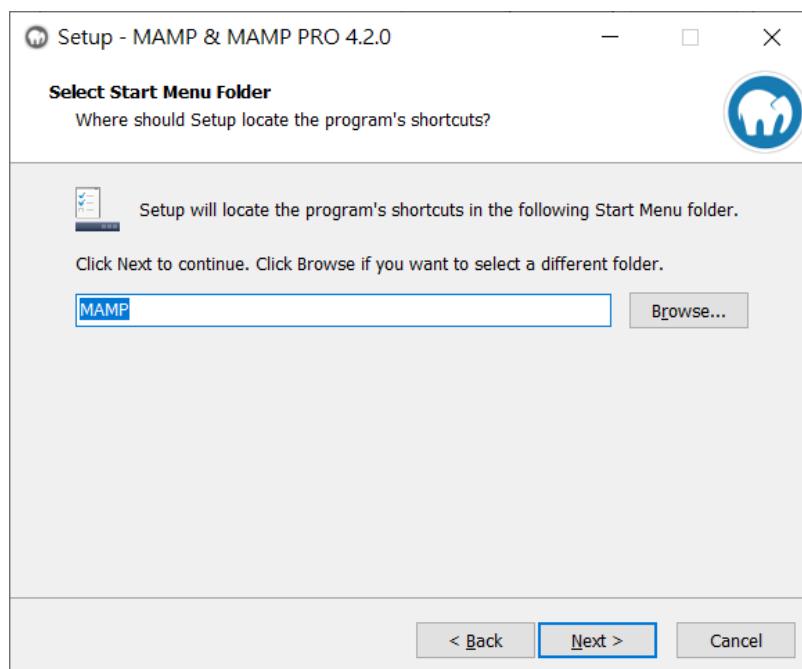
(圖) 按 Next



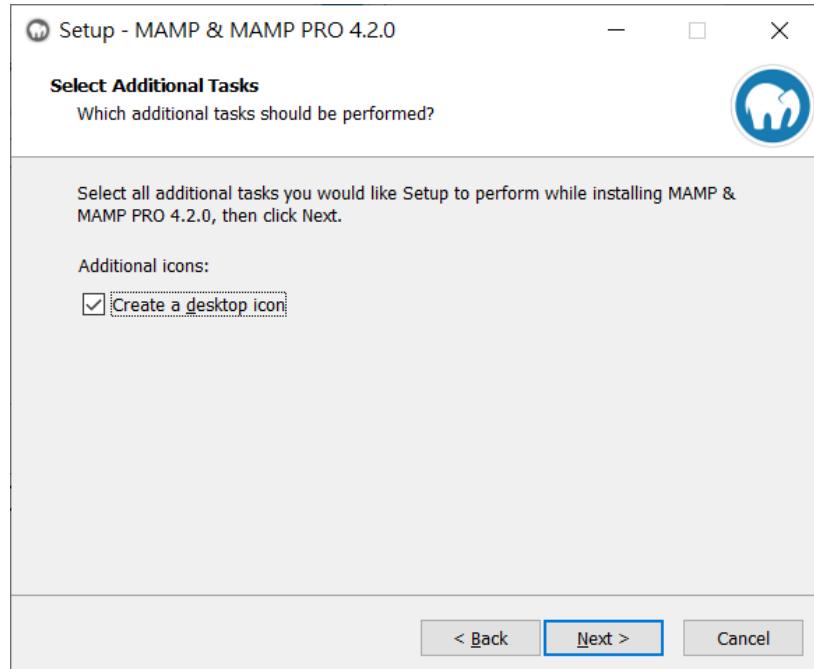
(圖) 選擇 I accept the agreement，按 Next



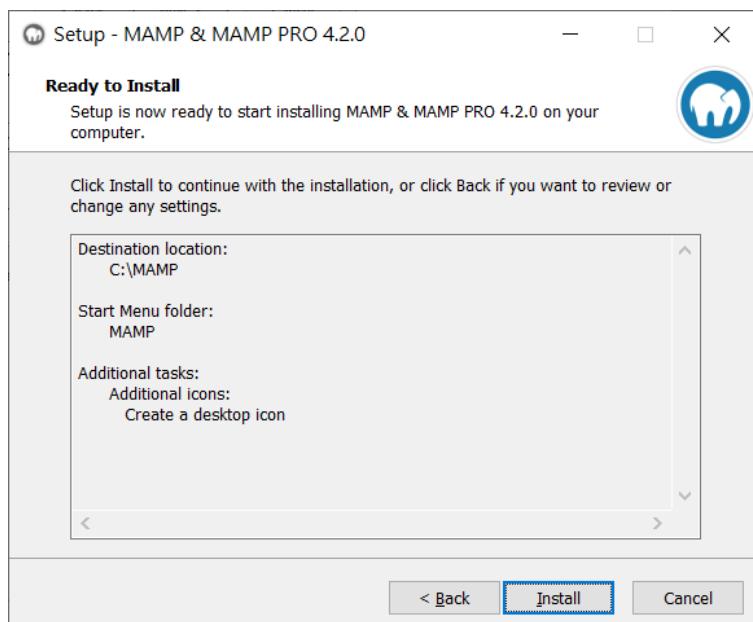
(圖) 依需求變更預設路徑；在這裡我們不變更，直接按 Next



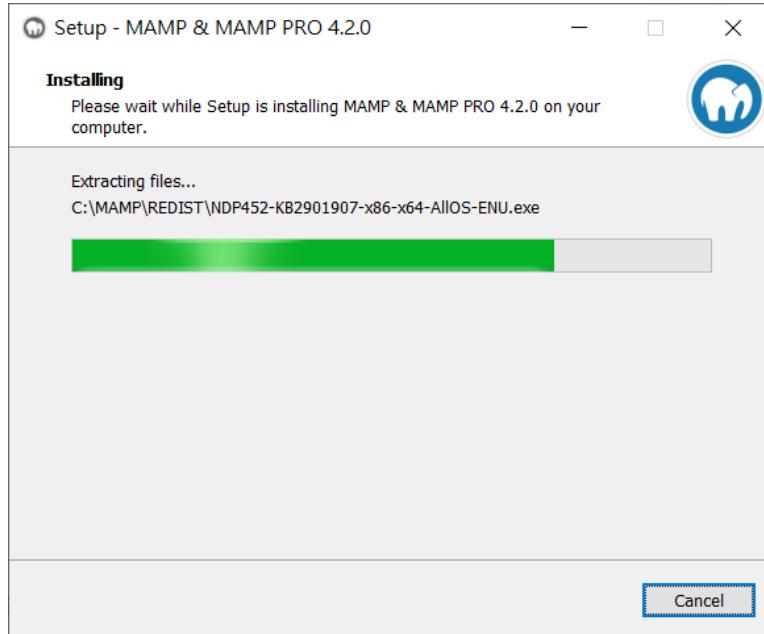
(圖) 按 Next



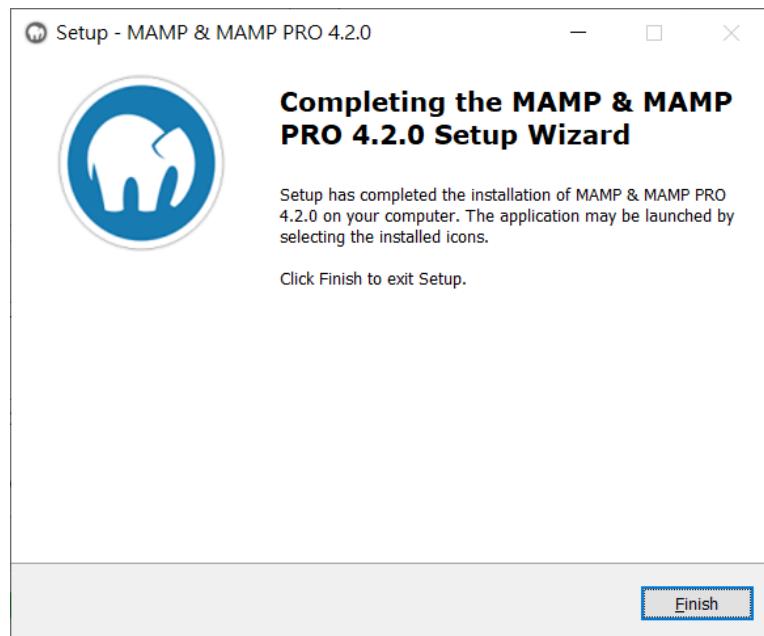
(圖) 依需求決定是否建立桌面圖示，按 Next



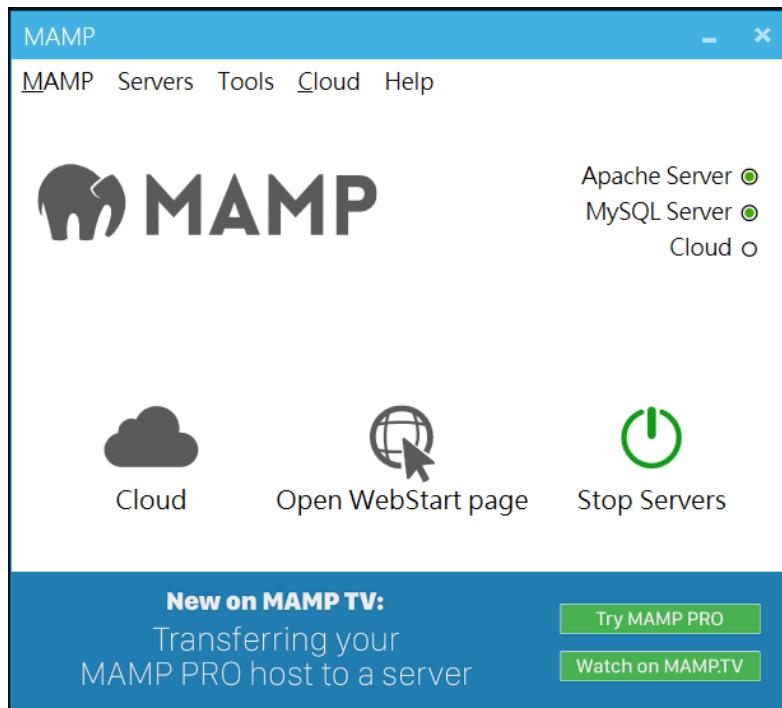
(圖) 確認安裝設定無誤，按下 Install，進行安裝



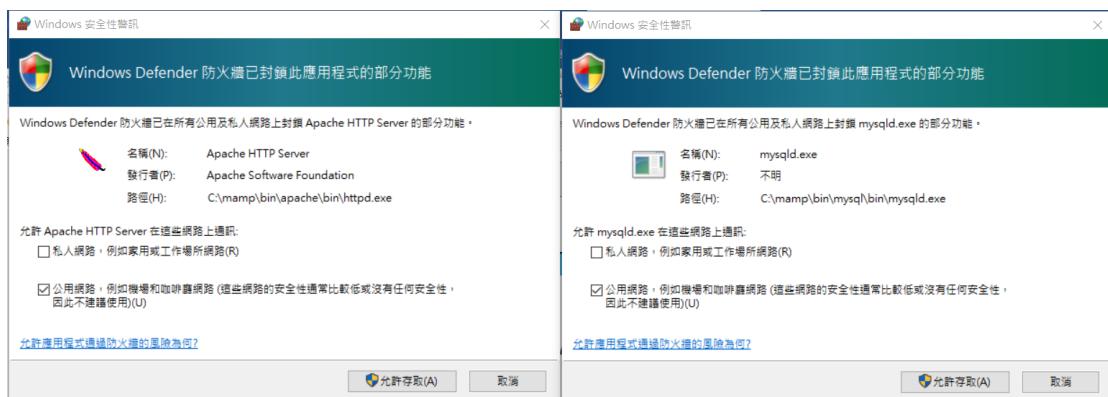
(圖) 安裝過程



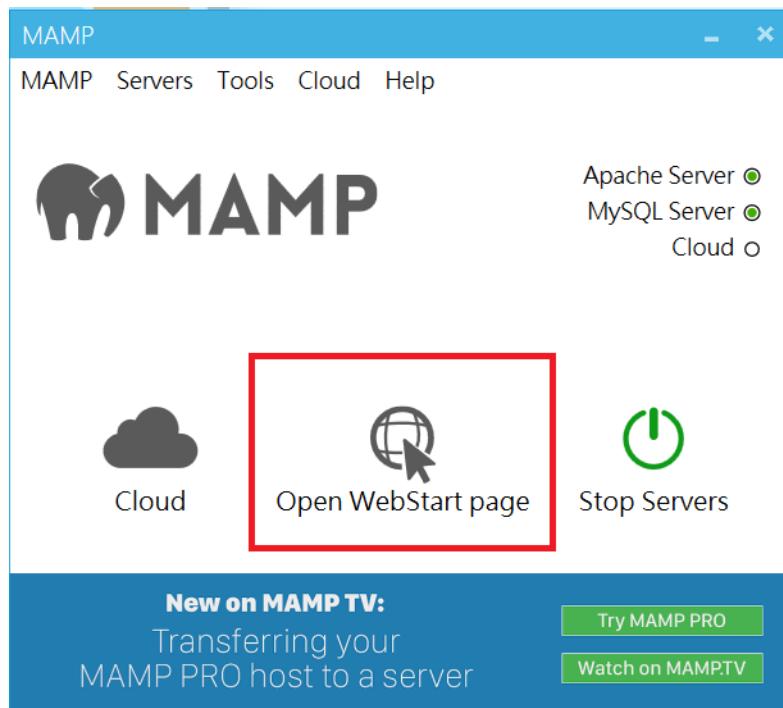
(圖) 安裝完成



(圖) 開啟 mamp 後，看到的畫面



(圖) Apache 跟 mysqld.exe 都按下「允許存取」

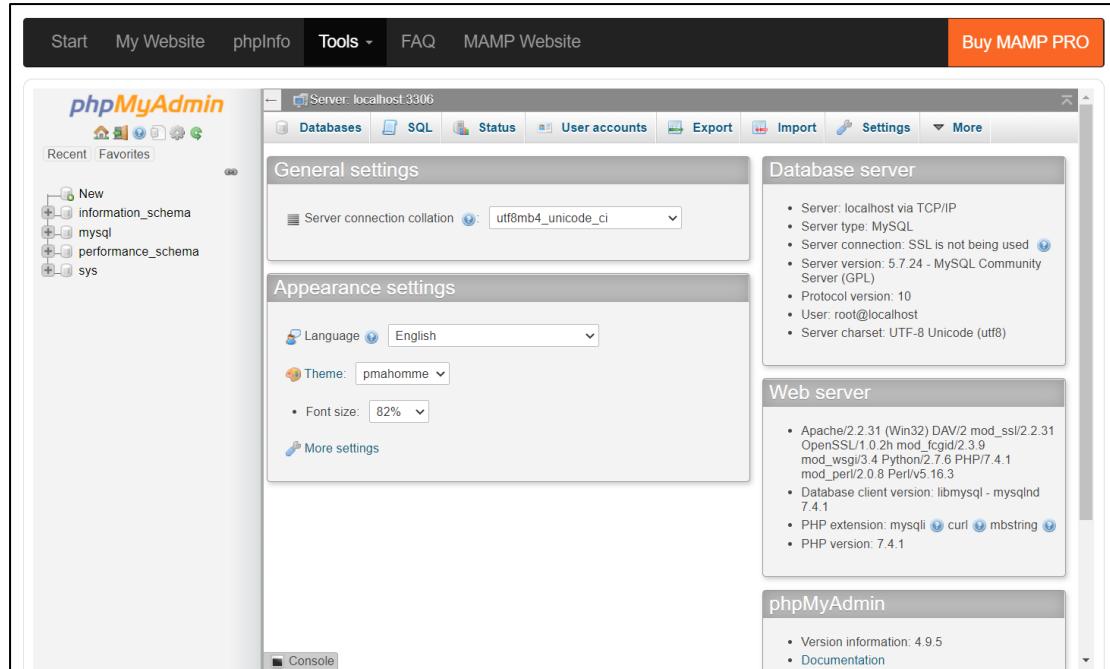


(圖) 按下中間的 Open WebStart page，開啟主要控制台頁面

A screenshot of the PHP configuration page. It starts with a section for PHP, stating that [phpinfo](#) shows the current configuration. Below that is a section for MySQL, stating that MySQL can be administered with [phpMyAdmin](#). It provides connection parameters: Host (localhost), Port (3306), User (root), and Password (root).

Host	localhost
Port	3306
User	root
Password	root

(圖) 觀看 PHP 組態，請按 [phpinfo](#)；操作 MySQL，請按 [phpMyAdmin](#)



(圖) phpMyAdmin 的操作畫面

二、XAMPP 安裝（上課使用）

課程中，我們原則上使用 XAMPP 來進行環境安裝與操作。XAMPP 是 Apache、MariaDB、PHP、Perl 等整合在一起的工具，提供 Web、FTP、phpMyAdmin 等服務，Windows、Linux、Mac 環境都能使用。**本次授課，以 Windows 環境為主，若有其它平台的問題，請在下課或課程結束後，進行討論。**

網址：https://www.apachefriends.org/zh_tw/index.html

參考連結：

XAMPP v7.3.6-2 網頁伺服器自動架站機 (Apache + MariaDB+ PHP + Perl + Tomcat + FTP 伺服器 + Webalizer 流量分析工具)

<https://briian.com/18718/>



圖：可下載不同作業系統版本，也可以切換語系

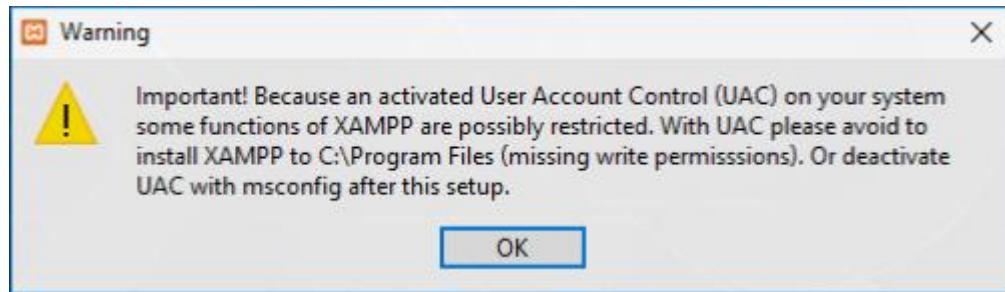


圖：等待下載畫面

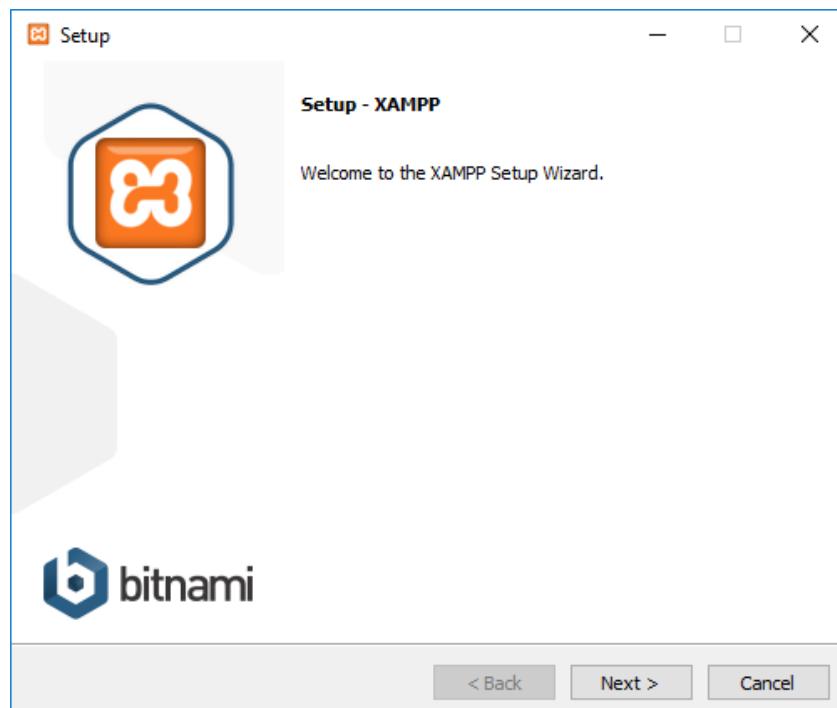


xampp-windows
-x64-8.0.3-0-VS
16-installer.exe

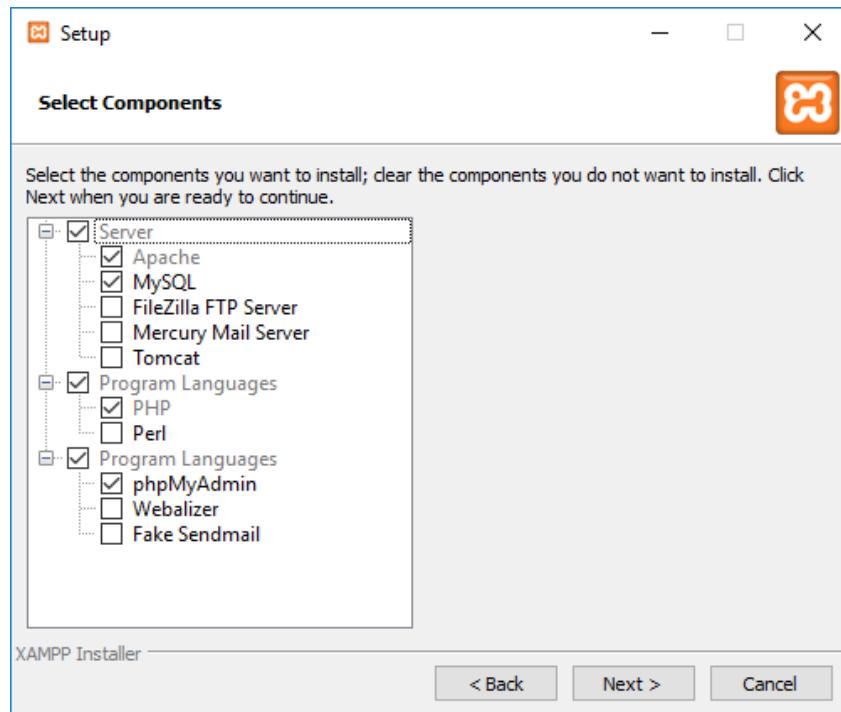
圖：下載後的檔案



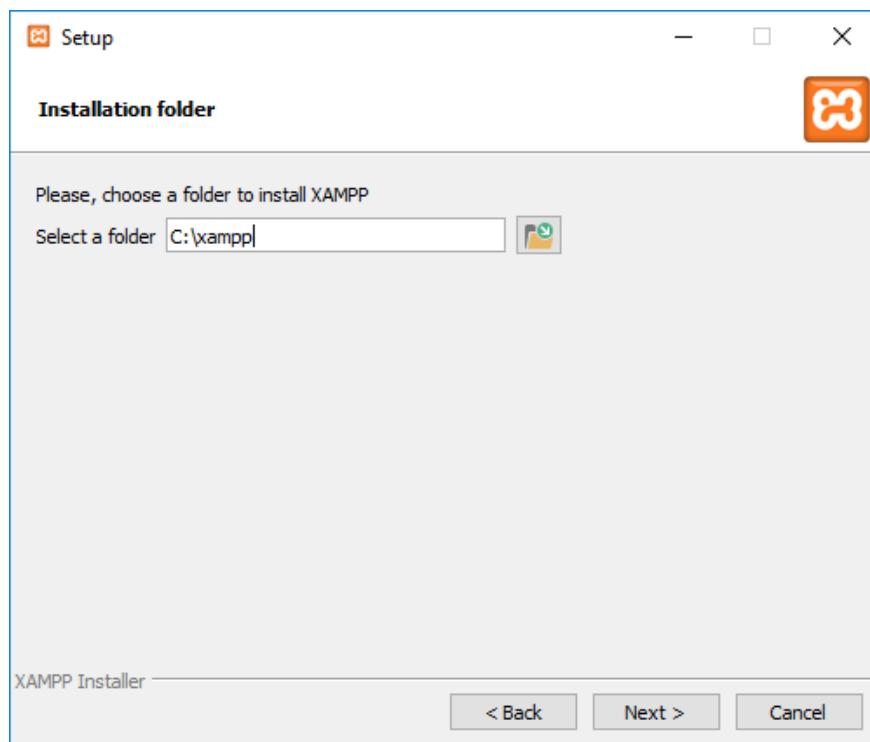
圖：不會安裝到 Program Files 路徑下，可以直接按下 OK



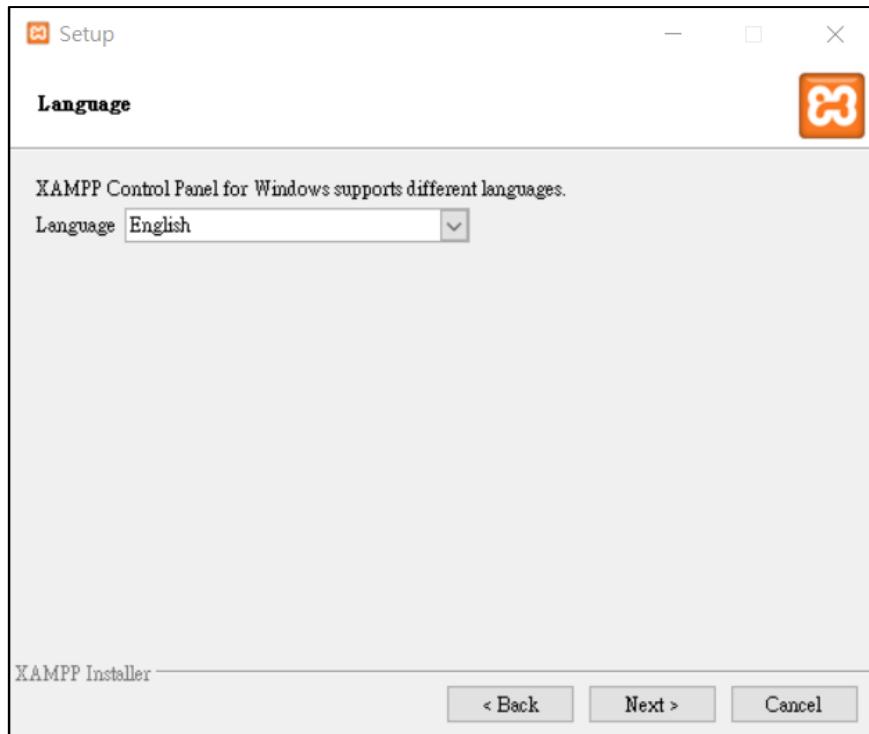
圖：按下一步



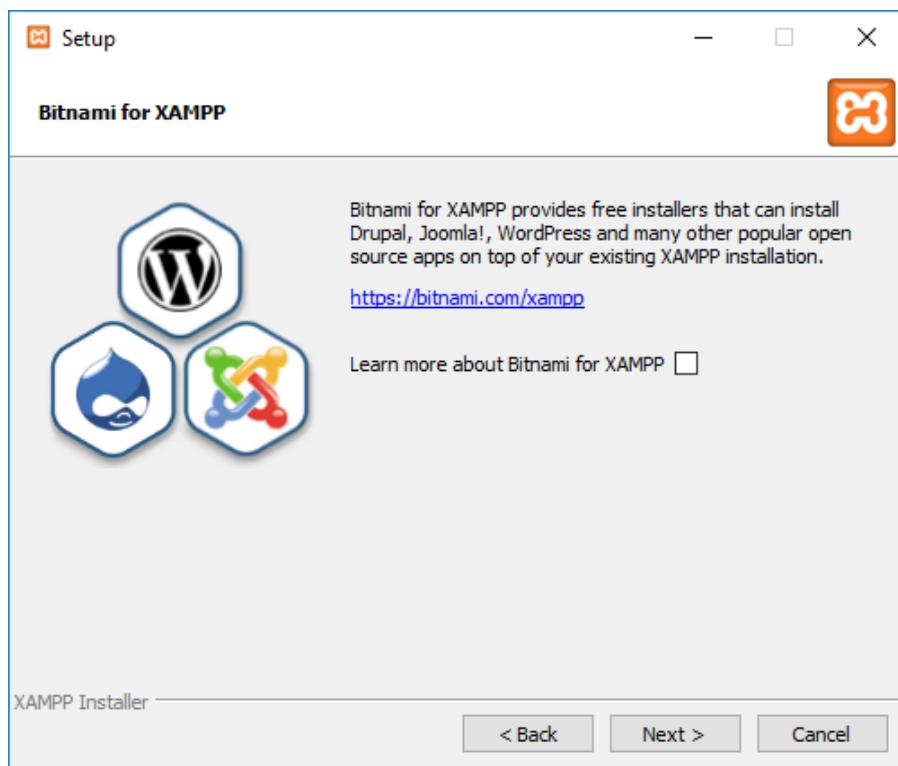
圖：除了預設的 Apache、PHP，僅留下 MySQL、phpMyAdmin 後，按下一步



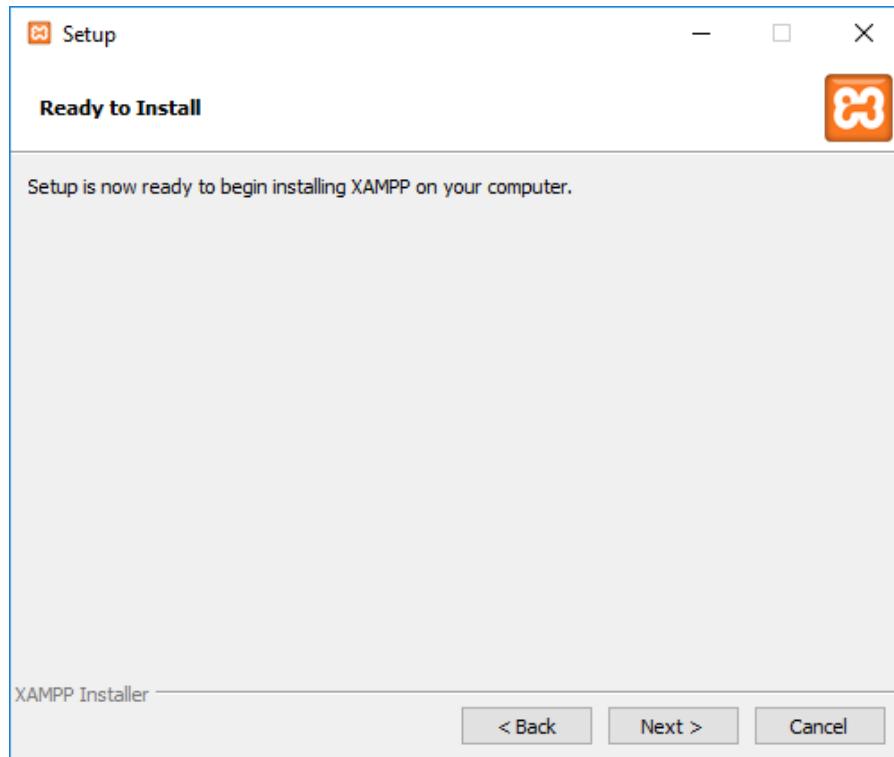
圖：使用預設路徑，按下一步



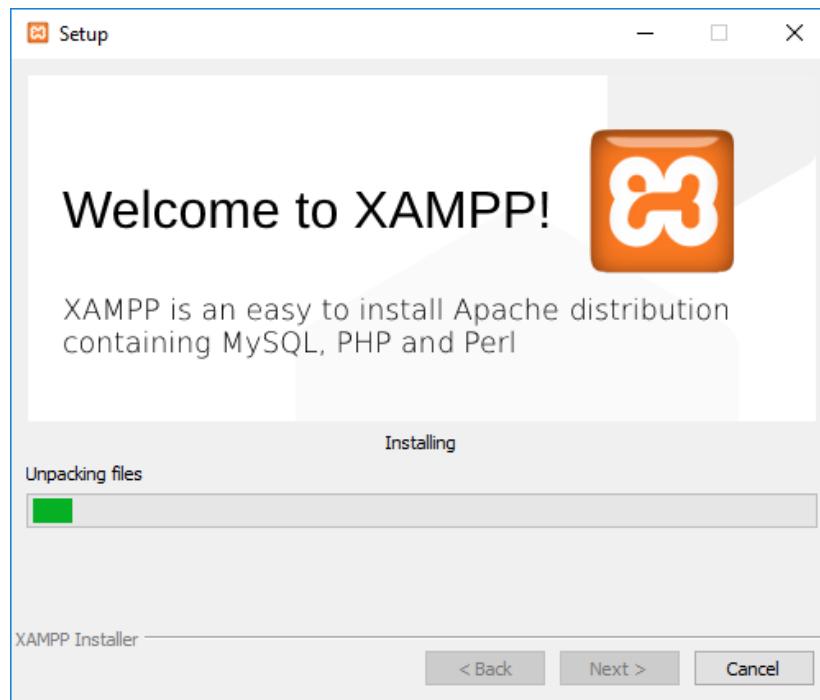
圖：選擇 English，按下一步



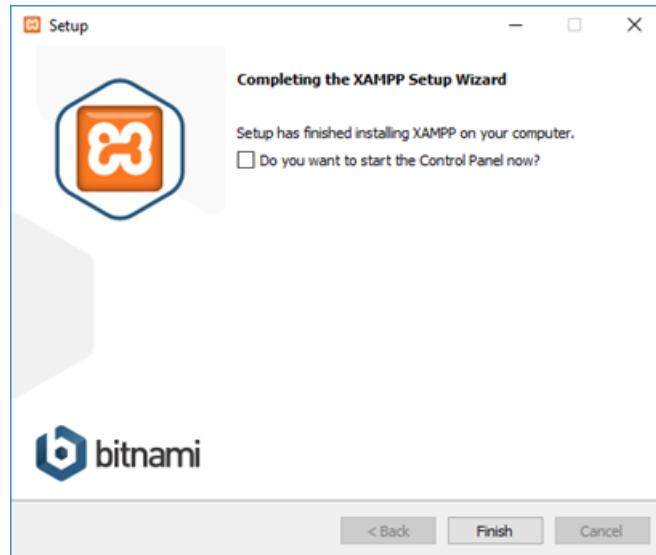
圖：課程中不會用到架站工具，所以取消上方勾選，按下一步



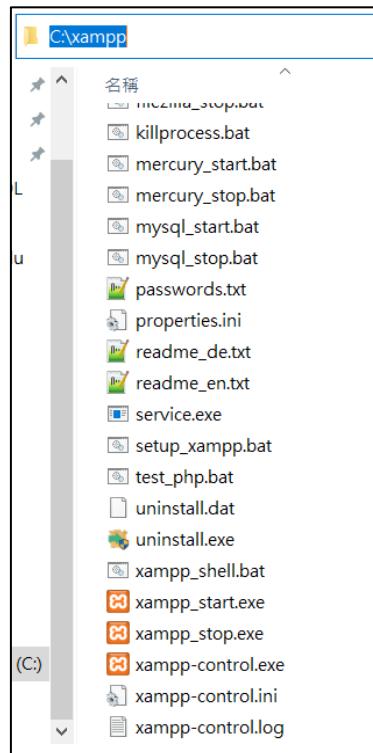
圖：按下一步，開始安裝



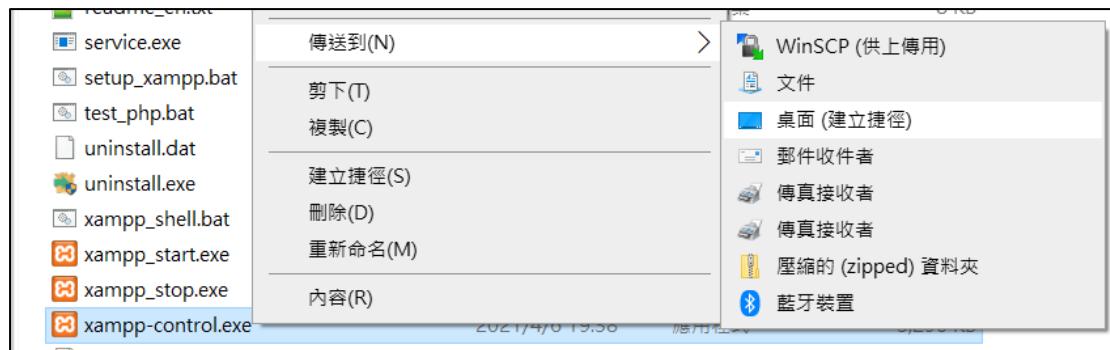
圖：安裝過程



圖：取消勾選上方選項，按下完成



圖：到 C:\xampp 裡面，找到 xampp-control.exe



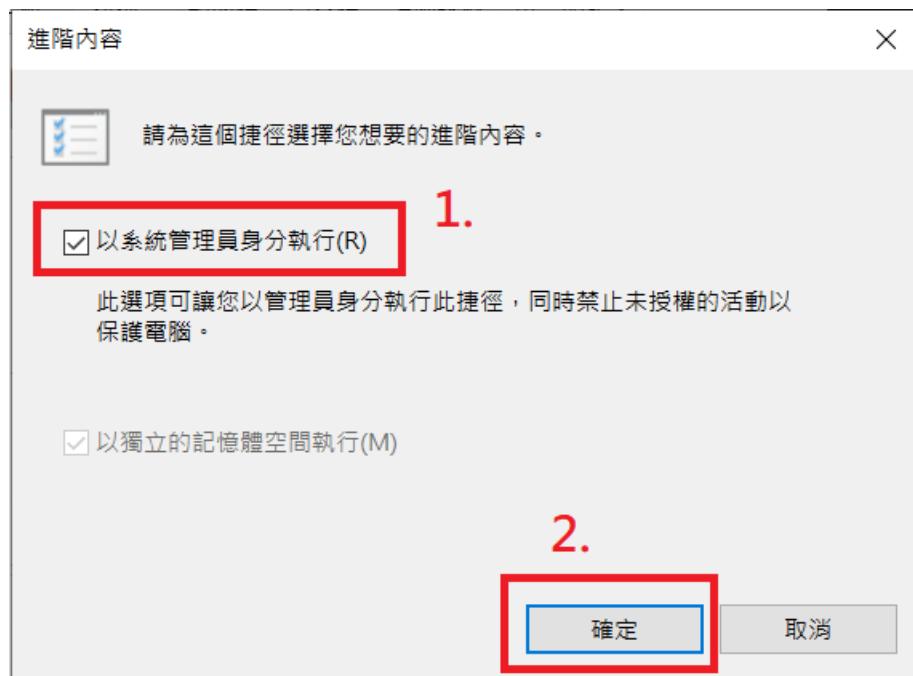
圖：對 xampp-control.exe 按滑鼠右鍵，選擇「傳送到」，並在桌面建立捷徑



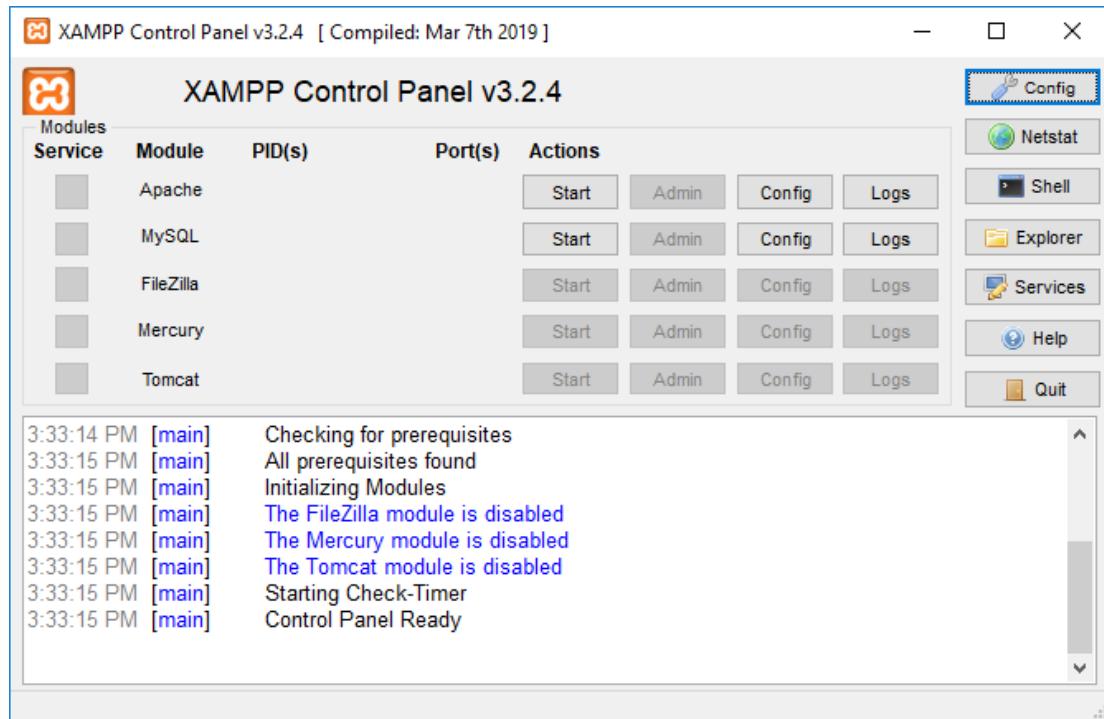
圖：對桌面連結按右鍵，點選「內容」



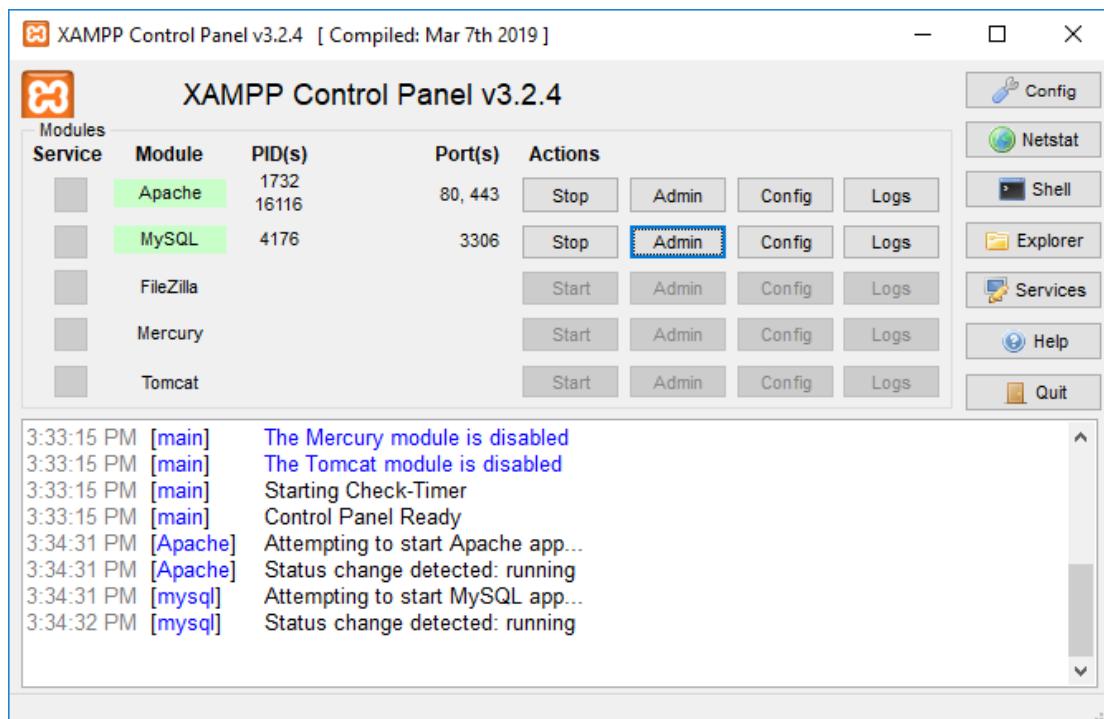
圖：選擇「捷徑」標籤，按下右下角的「進階」



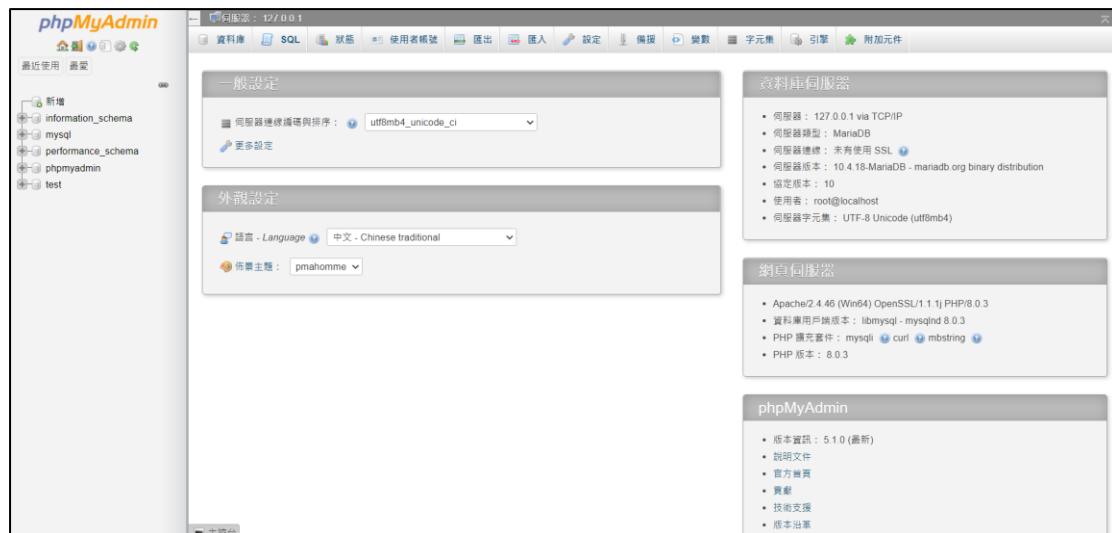
圖：勾選「以系統管理員身分執行」，按下確定，執行 xampp-control.exe



圖：按下 Apache、MySQL 右側的 Start 鍵



圖：兩個服務開啟後，按下 MySQL 右側的 Admin，開啟 phpMyAdmin 網頁



圖：看到 phpMyAdmin 的畫面，代表安裝成功

備註

預設的 MySQL 帳號為 root，密碼為空字串（就是不用輸入任何字，或是直接打上成雙的雙引號或單引號）

使用者名稱	主機名稱	密碼	全域權限	使用者群組	允許授權(Grant)	動作
任何	%	否	USAGE		否	編輯權限 匯出
pma	localhost	否	USAGE		否	編輯權限 匯出
root	127.0.0.1	否	ALL PRIVILEGES		是	編輯權限 匯出
root	::1	否	ALL PRIVILEGES		是	編輯權限 匯出
root	localhost	否	ALL PRIVILEGES		是	編輯權限 匯出
root	win-87hdnpkno3	否	ALL PRIVILEGES		是	編輯權限 匯出

使用者帳號一覽

新增 [新增使用者帳號](#)

（圖）選擇上面的使用者帳號，並按下圖片左下角的連結，新增使用者帳號

範例用資料庫帳號、密碼

課堂中所使用的範例，大部分會需要寫入資料庫，我們需要建立一組資料庫的帳號、密碼。

帳號：test

密碼：T1st@localhost

登入資訊

使用者名稱：
使用文字方塊: test

主機名稱：
本機 localhost

密碼：
使用文字方塊: Strength: 很好

重新輸入：.....

認證外掛程式
原生 MySQL 認證

產生密碼: **產生**

(圖) 登入資訊



(圖) 全域權限，請勾選「SELECT, INSERT, UPDATE, DELETE」



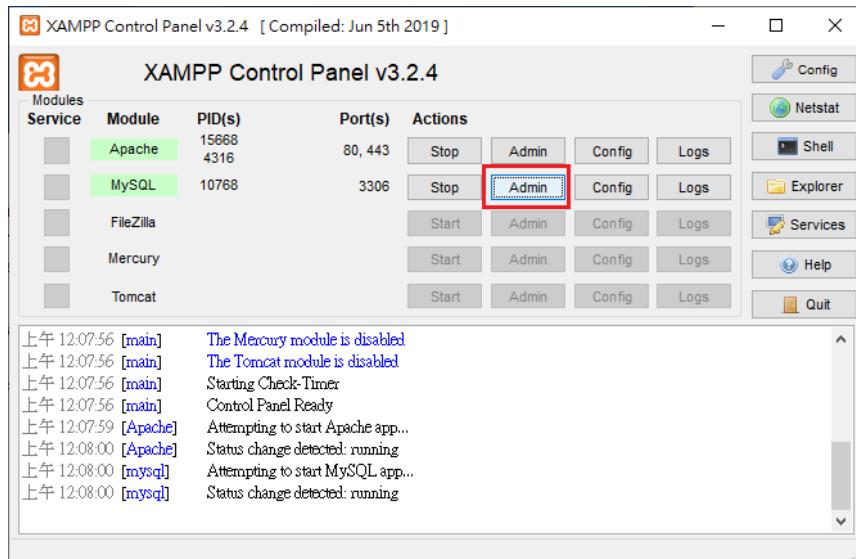
(圖) 最後按下網頁右下角的執行，建立新帳號

補充說明

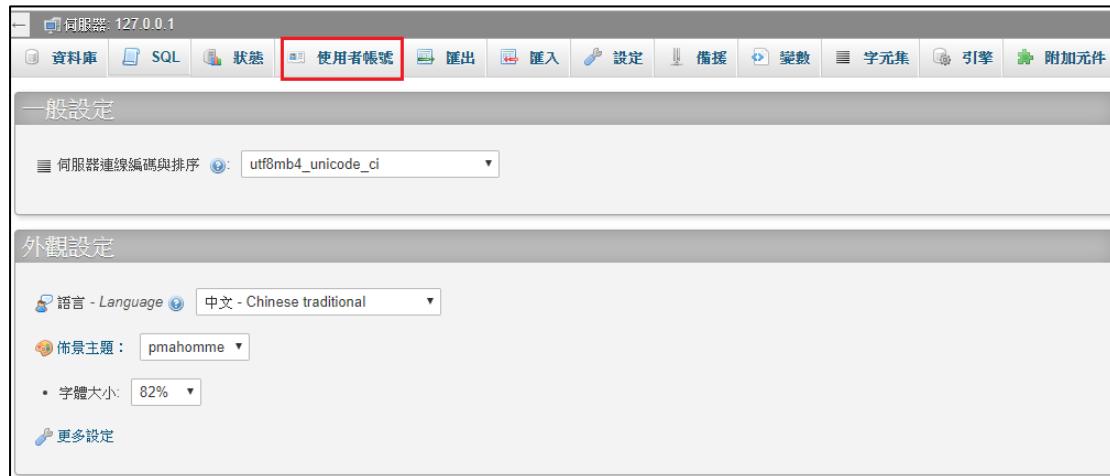
資料庫使用者帳號的增修，有很大的學問。課程中，為了教學方便，可以按照前述的流程建立使用者；若是公司有使用到資料庫，帳號管理不能草率，一定要請教有經驗的資料庫管理人員，協助帳號新增與權限設定。

設定 root 密碼的方式，有兩種，一個是透過 phpMyAdmin 的 SQL 介面、一個是使用 mysql 執行檔，透過命令視窗來執行 SQL 語法。

phpMyAdmin



(圖) 先開啟 Apache、MySQL，再按下 MySQL 的 Admin 來開啟



(圖) 點選「使用者帳號」連結

使用者名稱	主機名稱	密碼	全域權限	使用者群組	允許授權(Grant)	動作
任何	%	否	USAGE		否	
pma	localhost	否	USAGE		否	
root	127.0.0.1	否	ALL PRIVILEGES		是	
root	::1	否	ALL PRIVILEGES		是	
root	localhost	否	ALL PRIVILEGES		是	
test	localhost	是	ALL PRIVILEGES		是	

↑ 全選 已選擇項目：

新增

[新增使用者帳號](#)

(圖) 使用者帳號一覽

這裡有 3 個 root，我們先修改主機名稱為「localhost」的那一個帳號。

(圖) 按下 SQL 連結



(圖) 輸入 / 執行 SQL 的文字欄位

修改特定帳號的密碼：

```
SQL 語法
# 選擇資料庫
use mysql;

# 設定 特定帳號@主機名稱 = PASSWORD(' 你的密碼' );
SET PASSWORD FOR 'root'@'localhost' = PASSWORD(' T1st@localhost' );

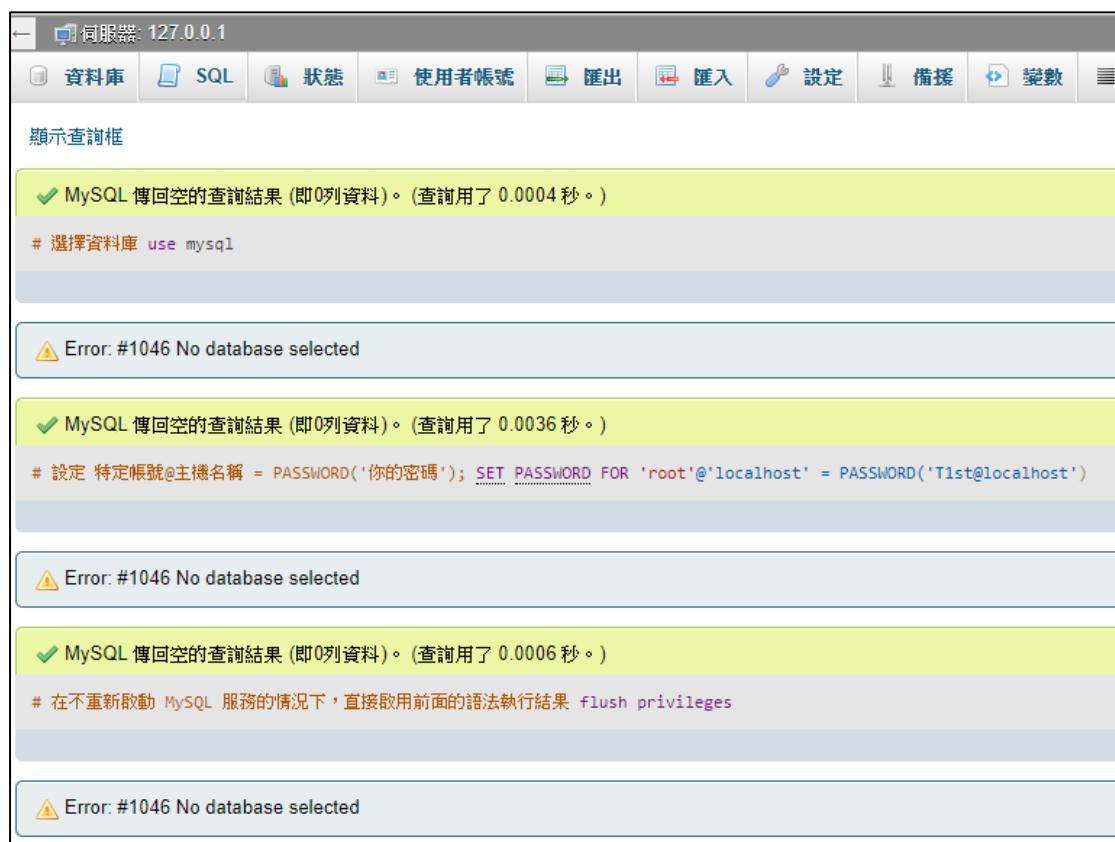
# 在不重新啟動 MySQL 服務的情況下，直接啟用前面的語法執行結果
flush privileges;
```



The screenshot shows the phpMyAdmin interface for server 127.0.0.1. The SQL tab is selected. The query editor contains the following SQL code:

```
1 # 選擇資料庫
2 use mysql;
3
4 # 設定 特定帳號@主機名稱 = PASSWORD('你的密碼');
5 SET PASSWORD FOR 'root'@'localhost' = PASSWORD('T1st@localhost');
6
7 # 在不重新啟動 MySQL 服務的情況下，直接啟用前面的語法執行結果
8 flush privileges;
```

(圖) 修改 root 密碼的 SQL 語法



The screenshot shows the phpMyAdmin interface for server 127.0.0.1. The SQL tab is selected. The query editor contains the same SQL code as the previous screenshot. The results pane shows the following output:

- MySQL 傳回空的查詢結果 (即0列資料)。 (查詢用了 0.0004 秒。)
- # 選擇資料庫 use mysql
- ⚠ Error: #1046 No database selected
- MySQL 傳回空的查詢結果 (即0列資料)。 (查詢用了 0.0036 秒。)
- # 設定 特定帳號@主機名稱 = PASSWORD('你的密碼'); SET PASSWORD FOR 'root'@'localhost' = PASSWORD('T1st@localhost')
- ⚠ Error: #1046 No database selected
- MySQL 傳回空的查詢結果 (即0列資料)。 (查詢用了 0.0006 秒。)
- # 在不重新啟動 MySQL 服務的情況下，直接啟用前面的語法執行結果 flush privileges
- ⚠ Error: #1046 No database selected

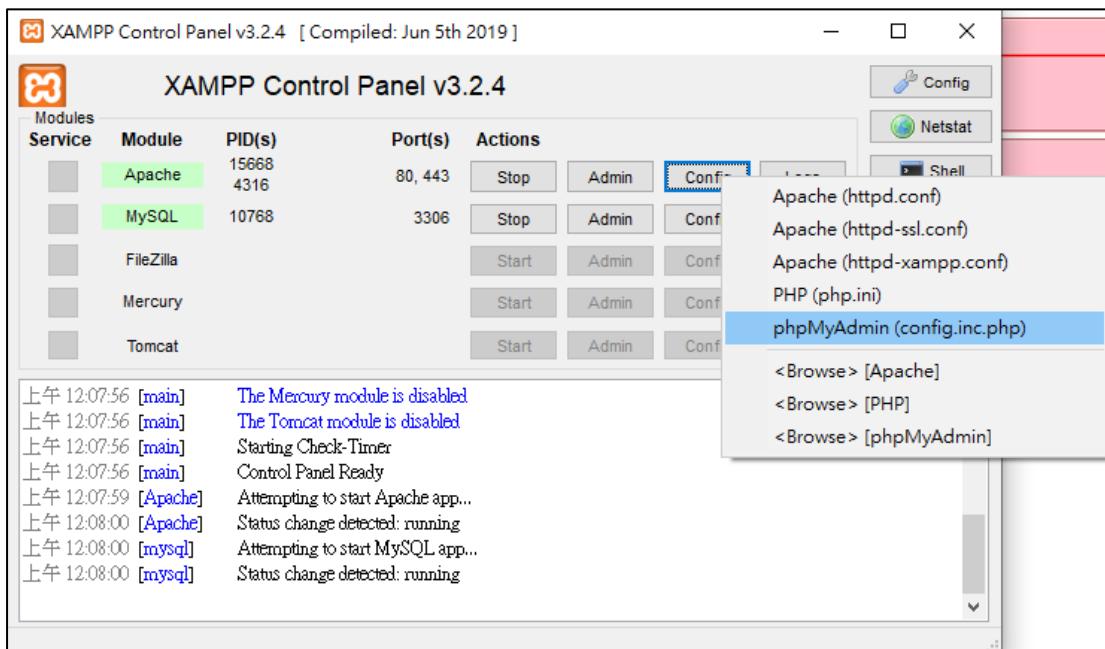
(圖) root 密碼修正完畢

改完密碼後，我們重新整理 phpMyAdmin 的頁面，會發現連線失敗的畫面。



(圖) 先前的 root 密已經無法使用

我們需要到 phpMyAdmin 裡面的 config.inc.php 去修正登入密碼。



(圖) 選擇 phpMyAdmin (config.inc.php)

我們接下來要將新密碼放到 config.inc.php 裡面。

```
/* Authentication type and info */
$cfg['Servers'][$i]['auth_type'] = 'config';
$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = '';
$cfg['Servers'][$i]['extension'] = 'mysqli';
$cfg['Servers'][$i]['AllowNoPassword'] = true;
$cfg['Lang'] = '';
```

(圖) 將新密碼放到 \$cfg['Servers'][\$i]['password'] = '你的新密碼'

```
/* Authentication type and info */
$cfg['Servers'][$i]['auth_type'] = 'config';
$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = 'T1st@localhost';
$cfg['Servers'][$i]['extension'] = 'mysqli';
$cfg['Servers'][$i]['AllowNoPassword'] = true;
$cfg['Lang'] = '';
```

(圖) 放入新密碼



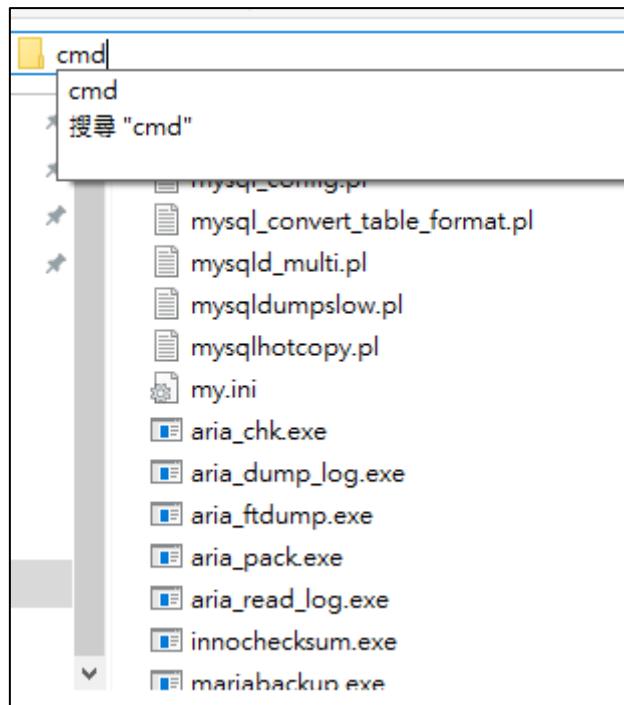
(圖) 再刷新 phpMyAdmin 網頁後，恢復到原先的樣子，一樣是 root 帳號

mysql 執行檔

確認兩件事：

- 有「C:\xampp\mysql\bin」路徑
- 路徑裡面，有幾個執行檔，其中一個叫作「mysql」

首先，我們先在「C:\xampp\mysql\bin」上頭的路徑列，輸入「cmd」。



(圖) 路徑列上，輸入 cmd

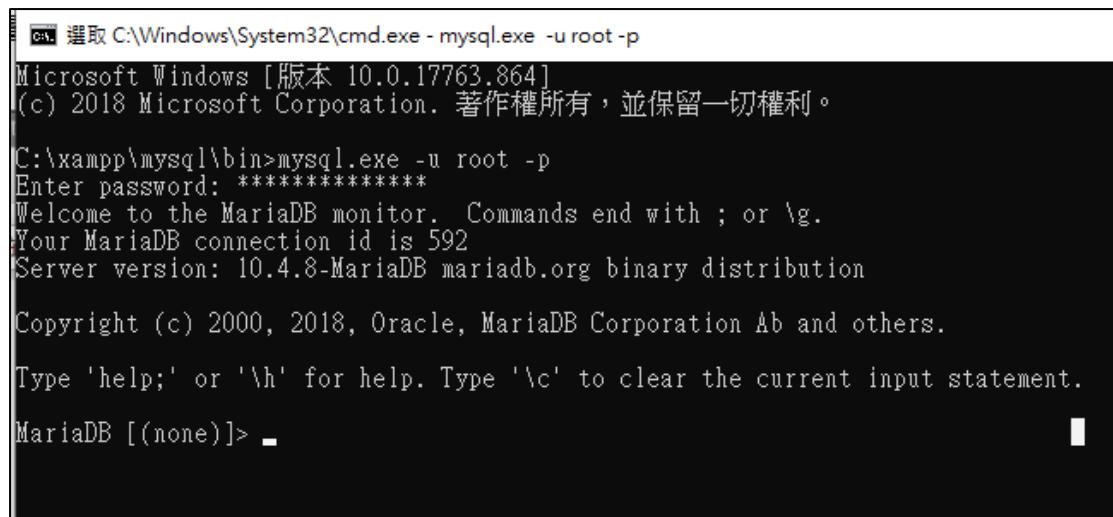


(圖) 出現命令提示字元

接下來，我們要輸入剛才修改的 root 帳號與密碼。

說明

```
# 使用 mysql 執行檔，使用者名稱為 root，密碼之後會詢問
> mysql.exe -u root -p
Enter password: *****
```



cmd 選取 C:\Windows\System32\cmd.exe - mysql.exe -u root -p
Microsoft Windows [版本 10.0.17763.864]
(c) 2018 Microsoft Corporation. 著作權所有，並保留一切權利。
C:\xampp\mysql\bin>mysql.exe -u root -p
Enter password: *****
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 592
Server version: 10.4.8-MariaDB mariadb.org binary distribution
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MariaDB [(none)]> ■

(圖) 登入成功的畫面

指令流程

```
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input  
statement.  
MariaDB [(none)]> use mysql;  
MariaDB [mysql]> SET PASSWORD FOR 'root'@'localhost' =  
PASSWORD('use2@localhost');  
Query OK, 0 rows affected (0.004 sec)  
  
MariaDB [mysql]> flush privileges;  
Query OK, 0 rows affected (0.000 sec)  
  
MariaDB [mysql]>
```

此時我們回到 phpMyAdmin，重新刷新頁面，便會出現錯誤訊息。

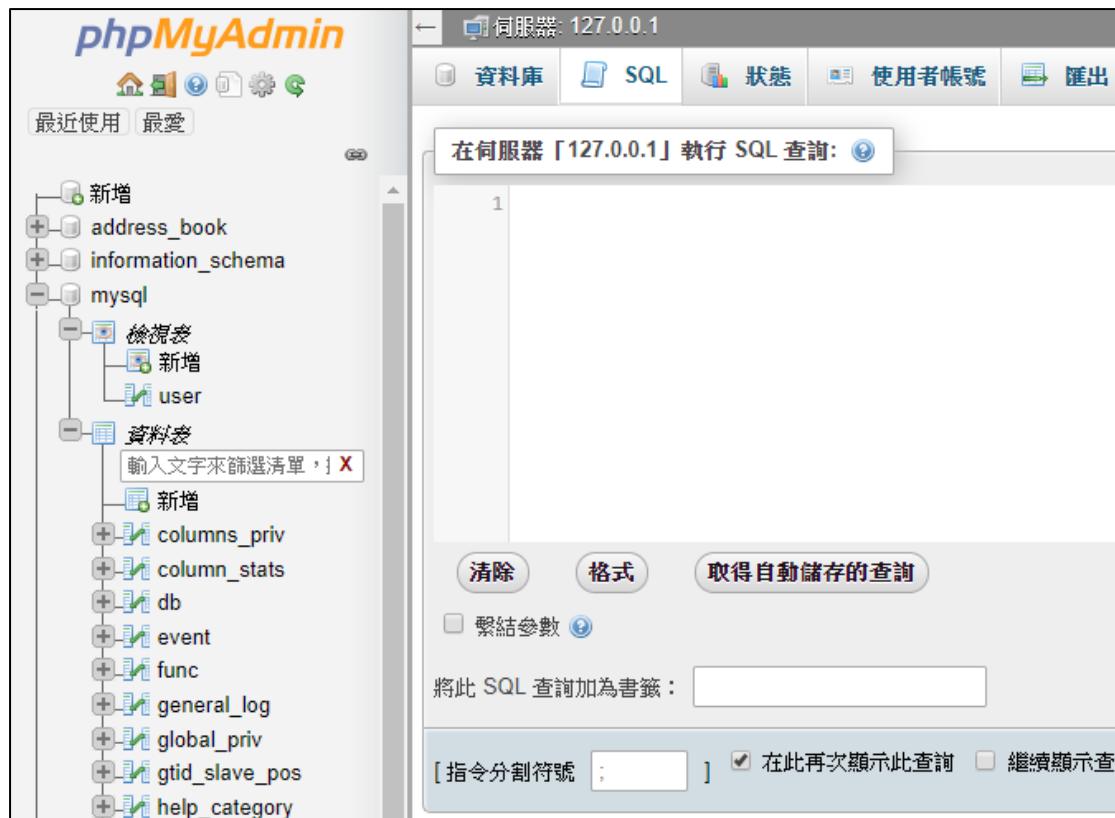


(圖) 連線錯誤的畫面

我們要再一次將新密碼放到 config.inc.php 裡面。

```
/* Authentication type and info */
$cfg['Servers'][$i]['auth_type'] = 'config';
$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = 'use2@localhost';
$cfg['Servers'][$i]['extension'] = 'mysqli';
$cfg['Servers'][$i]['AllowNoPassword'] = true;
$cfg['Lang'] = '';
```

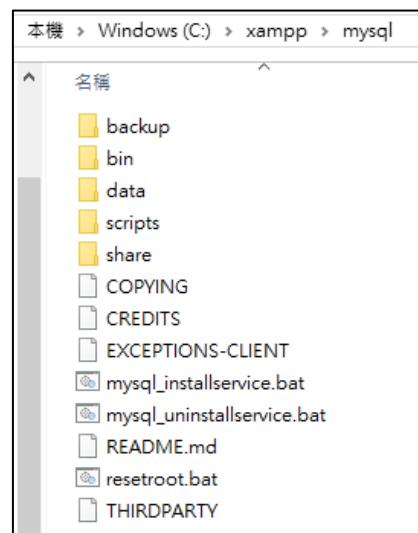
(圖) 輸入新密碼



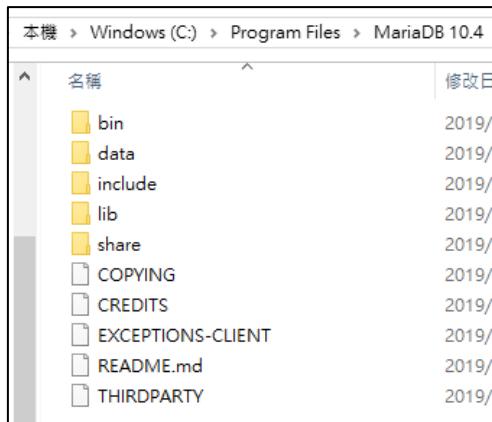
(圖) 此時又回到正常畫面

三、快速導覽

以 XAMPP 的 MySQL 資料夾為例，路徑在「C:\xampp\mysql」。



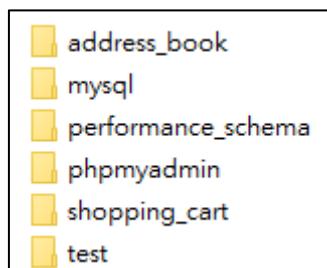
(圖) XAMPP 的 MySQL 資料夾結構



(圖) 直接官方網站下載 MariaDB 的資料夾結構

C:\xampp\mysql\data

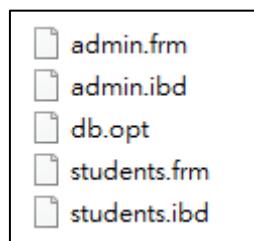
MySQL 放置「資料」的資料夾，也是日誌 (log) 被儲存放置的地方。在 data 中，子資料夾 (mysql、performance_schema、phpmyadmin、test) 也是放置資料庫 (database) 的地方，在放置資料庫的資料夾中，也存放著這些資料庫所擁有的資料表 (tables)。



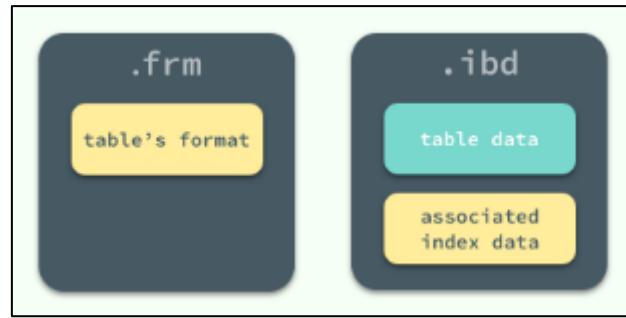
(圖) 以資料庫(database)為名的資料夾，裡面是資料表(tables)的檔案

一個新建立的資料庫，可能會有以下的檔案格式（以 InnoDB 為例）：

- frm：描述資料表的格式，或許說定義 (table definition)
- ibd：放置資料表中的資料，以及關聯的索引資料
- db.opt：設定預設文字編碼和文字排序規則



(圖) address_book



(圖) .frm 和 .ibd 檔案結構

備註：ibdata1 是儲存 MySQL InnoDB 引擎的結構、資料、索引和快取

C:\xampp\mysql\data\mysql

放置主要的 mysql 資料庫資料夾，也是管理 MySQL 用的資料庫。

C:\xampp\mysql\share\charset

放置支援字元集的檔案

```
MariaDB [(none)]> use mysql;
Database changed
MariaDB [mysql]> show character set;
+ Charset  | Description          | Default collation | Maxlen |
+-----+-----+-----+
| big5    | Big5 Traditional Chinese | big5_chinese_ci   | 2       |
| dec8    | DBC West European      | dec8_swedish_ci   | 1       |
| cp850   | DOS West European      | cp850_general_ci | 1       |
| hp8     | HP West European       | hp8_english_ci   | 1       |
| koi8r   | KOI8-R Relcom Russian | koi8r_general_ci | 1       |
| latin1  | cp1252 West European   | latin1_swedish_ci | 1       |
| latin2  | ISO 8859-2 Central European | latin2_general_ci | 1       |
| swe7    | 7bit Swedish           | swe7_swedish_ci   | 1       |
| ascii   | US ASCII               | ascii_general_ci  | 1       |
| ujis    | EUC-JP Japanese        | ujis_japanese_ci | 3       |
| sjis   | Shift-JIS Japanese     | sjis_japanese_ci | 2       |
| hebrew  | ISO 8859-8 Hebrew      | hebrew_general_ci | 1       |
| tis620  | TIS620 Thai            | tis620_thai_ci   | 1       |
| euckr   | EUC-KR Korean          | euckr_korean_ci  | 2       |
| koi8u   | KOI8-U Ukrainian       | koi8u_general_ci | 1       |
| gb2312  | GB2312 Simplified Chinese | gb2312_chinese_ci | 2       |
| greek   | ISO 8859-7 Greek        | greek_general_ci | 1       |
| cp1250  | Windows Central European | cp1250_general_ci | 1       |
| gbk    | GBK Simplified Chinese | gbk_chinese_ci   | 2       |
| latin5  | ISO 8859-9 Turkish      | latin5_turkish_ci | 1       |
| armSCII8 | ARMSSCII-8 Armenian    | armSCII8_general_ci | 1       |
| utf8    | UTF-8 Unicode           | utf8_general_ci  | 3       |
| ucs2    | UCS-2 Unicode           | ucs2_general_ci  | 2       |
| cp866   | DOS Russian              | cp866_general_ci | 1       |
| keybcs2 | DOS Kamenicky Czech-Slovak | keybcs2_general_ci | 1       |
| macce   | Mac Central European     | macce_general_ci  | 1       |
| macroman | Mac West European       | macroman_general_ci | 1       |
| cp852   | DOS Central European    | cp852_general_ci | 1       |
| latin7  | ISO 8859-13 Baltic      | latin7_general_ci | 1       |
| utf8mb4 | UTF-8 Unicode           | utf8mb4_general_ci | 4       |
| cp1251  | Windows Cyrillic        | cp1251_general_ci | 1       |
| utf16   | UTF-16 Unicode           | utf16_general_ci  | 4       |
| utf16le | UTF-16LE Unicode        | utf16le_general_ci | 4       |
| cp1256  | Windows Arabic           | cp1256_general_ci | 1       |
| cp1257  | Windows Baltic           | cp1257_general_ci | 1       |
| utf32   | UTF-32 Unicode           | utf32_general_ci  | 4       |
| binary  | Binary pseudo_charset     | binary           | 1       |
| geostd8 | GEOSTD8 Georgian         | geostd8_general_ci | 1       |
| cp932   | SJIS for Windows Japanese | cp932_japanese_ci | 2       |
| eucjpms | UJIS for Windows Japanese | eucjpms_japanese_ci | 3       |
+-----+-----+-----+
40 rows in set (0.007 sec)

MariaDB [mysql]>
```

(圖) 使用 SHOW CHARACTER SET；來看字元集列表



(圖) 從 phpMyAdmin 中一覽字元集

C:\xampp\mysql\share/{語系}

資料夾包含了各種語系的錯誤訊息

執行檔放置路徑為「C:\xampp\mysql\bin」我們介紹幾個常用執行檔的簡易使用範例。

mysqldump

資料庫匯出/備份

1. 備份單一資料庫：

```
$ mysqldump -u root -p 你的資料庫名稱 > 你的備份名稱.sql;
```

2. 僅備份特定資料庫下的資料表：

```
$ mysqldump -u root -p 你的資料庫名稱 Table1 Table2... > 你的備份名稱.sql;
```

3. 僅匯出資料表結構：

```
$ mysqldump -u root -p --no-data 你的資料庫名稱 > 你的備份名稱.sql;
```

4. 備份全部的資料庫：

```
$ mysqldump -u root -p -A > 你的備份名稱.sql;
```

mysql/mysqldump/mysqladmin 與「登入」有關的參數說明：

-u Username

用來存取 mysql 資料表的用戶名稱。此參數可省略，若省略，則表示為目前登入 linux 的用戶名稱。

-p

於執行時詢問密碼。

mysql

登入

```
> mysql -u 使用者帳號 -p 要求輸入密碼
```

```
> mysql -u root -p
```

Enter Password:

資料庫匯入/還原

1. 建立資料庫，再從備份檔匯入單一資料庫：

```
$ mysqladmin -u root -p create 你的資料庫名稱;
```

```
$ mysql -u root -p 你的資料庫名稱 < 你的備份名稱..sql
```

2. 從備份檔匯入全部的資料庫（不需先建立資料庫）：

```
$ mysql -u root -p < 所有資料庫備份檔.sql;
```

```
C:\xampp\mysql\bin>mysql -u root -p
Enter password: *****
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 296
Server version: 10.4.8-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

（圖）登入 mysql 後，出現 Welcome to the MariaDB monitor

安裝完 mysql 後，系統預設會建立一個不需要密碼的 root 用戶，和一個無用戶名、無密碼的匿名用戶（Anonymous Account）。進行下面的初始化操作以合理授權，增強安全。我們可以直接在 phpMyAdmin 裡面，使用 SQL 語法，來刪除匿名帳號。

SQL 語法

```
# 選擇資料庫
```

```
use mysql;
```

移除匿名使用者

```
DROP USER ''@'localhost';
```

或是

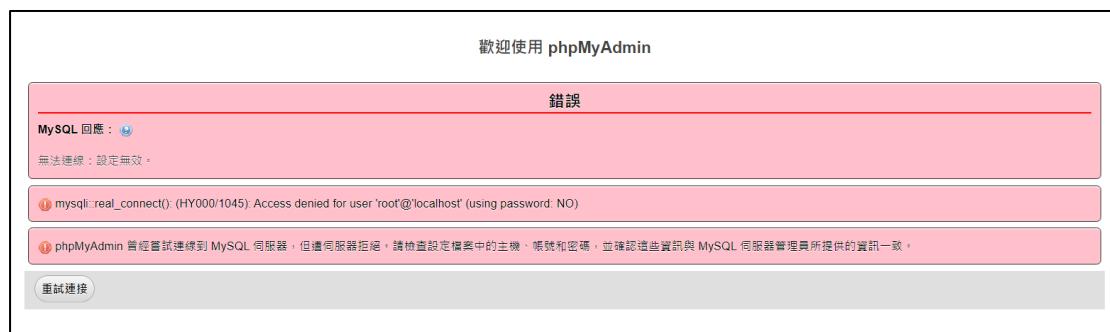
```
DELETE FROM `User` WHERE `User`='';
```

不重新啟動 MySQL 服務，直接啟用前面的語法執行結果（刷新權限）

```
flush privileges;
```

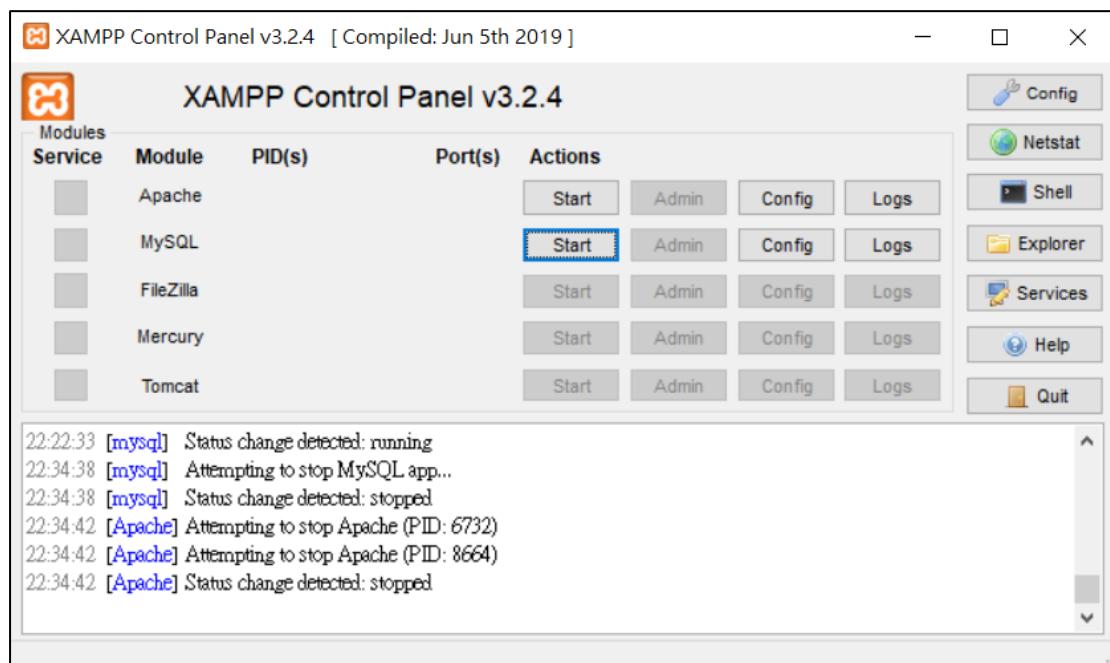
忘記 root 密碼，重新設定的方式

1. 發現無法使用 root 預設的設定登入



圖：發現 root@localhost 登不進去了

2. 先將 mysql 服務關閉



圖：確認 MySQL 服務狀態為 stopped

3. 進入命令提示字元，切換到 mysql 執行檔的目錄下

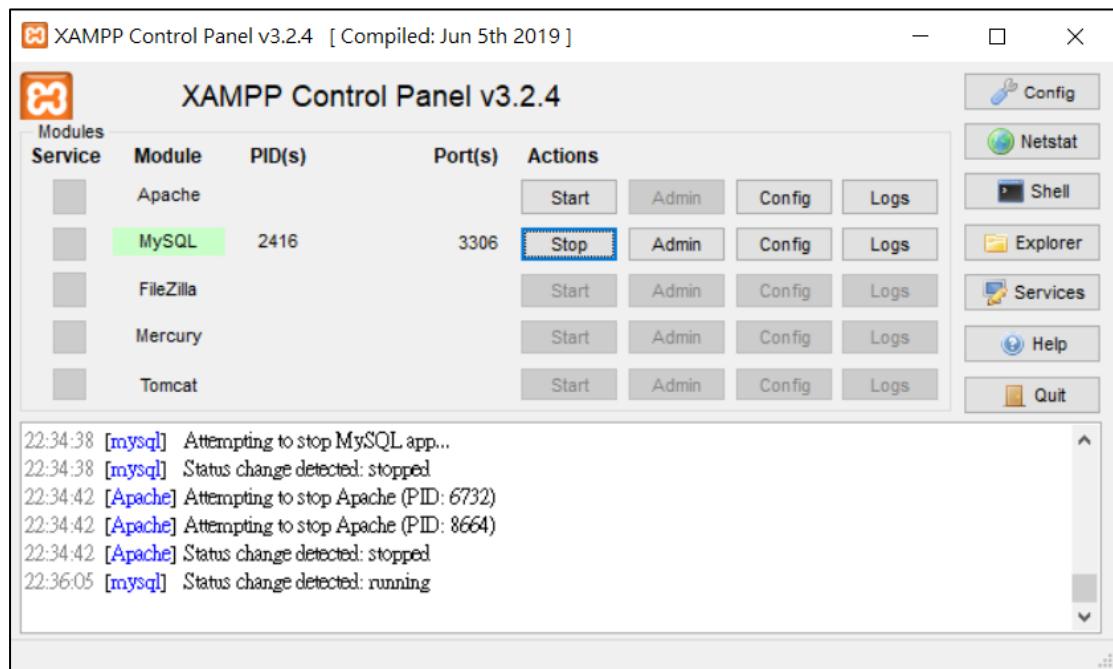
```
命令提示字元
Microsoft Windows [版本 10.0.19042.928]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。
C:\Users\Owner>cd c:\xampp\mysql\bin
c:\xampp\mysql\bin>
```

圖：開啟 cmd，切換目錄到 C:\xampp\mysql\bin\

4. 執行「mysqld --skip-grant-tables」指令，MySQL 服務再度啟動，但此視窗已經被服務佔用，無法操作

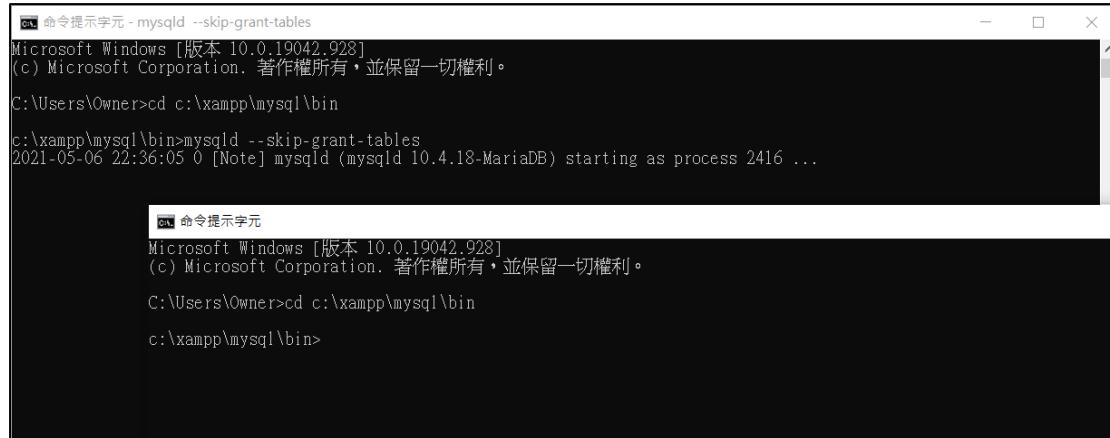
```
命令提示字元 - mysqld --skip-grant-tables
Microsoft Windows [版本 10.0.19042.928]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。
C:\Users\Owner>cd c:\xampp\mysql\bin
c:\xampp\mysql\bin>mysqld --skip-grant-tables
2021-05-06 22:36:05 0 [Note] mysqld (mysqld 10.4.18-MariaDB) starting as process 2416 ...
```

圖：執行指令後，MySQL 服務將會再度啟動



圖：此時會看到 MySQL 服務又啟動了

5. 原先的 cmd 視窗被 MySQL 服務佔用了，我們新開一個 cmd 視窗，再到 mysql 執行檔目錄下

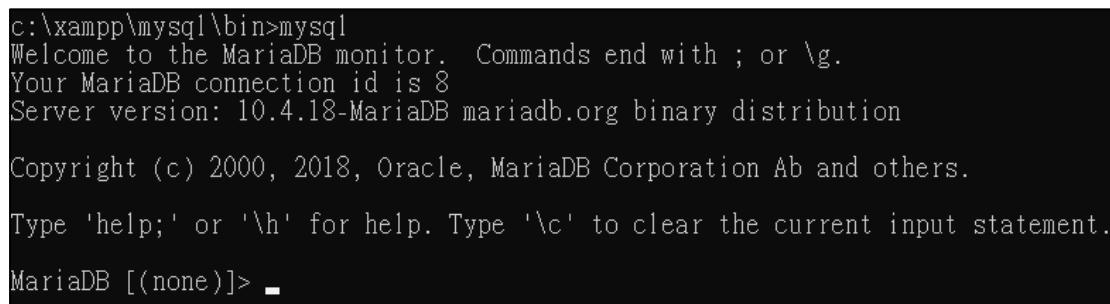


```
c:\命令提示字元 - mysqld --skip-grant-tables
Microsoft Windows [版本 10.0.19042.928]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。
C:\Users\Owner>cd c:\xampp\mysql\bin
c:\xampp\mysql\bin>mysqld --skip-grant-tables
2021-05-06 22:36:05 0 [Note] mysqld (mysqld 10.4.18-MariaDB) starting as process 2416 ...

c:\命令提示字元
Microsoft Windows [版本 10.0.19042.928]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。
C:\Users\Owner>cd c:\xampp\mysql\bin
c:\xampp\mysql\bin>
```

圖：另開 cmd 視窗，再次切換到 mysql 執行檔目錄下

6. 輸入 mysql，直接進入 mysql 的指令操作畫面



```
c:\xampp\mysql\bin>mysql
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 8
Server version: 10.4.18-MariaDB mariadb.org binary distribution

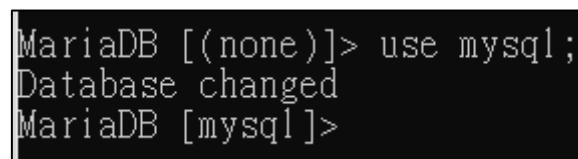
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> ■
```

圖：此時直接輸入 mysql，無須密碼即可進入資料庫指令介面

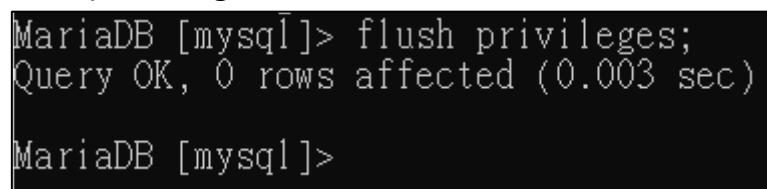
7. 輸入「use mysql;」，使用 user 資料庫



```
MariaDB [(none)]> use mysql;
Database changed
MariaDB [mysql]>
```

圖：進入 user 資料庫

8. 輸入「flush privileges;」，刷新權限表



```
MariaDB [mysql]> flush privileges;
Query OK, 0 rows affected (0.003 sec)

MariaDB [mysql]>
```

圖：刷新權限表

9. 輸入「`set password for 'root'@'localhost' = password('');`」，更新密碼，再使用指令「`flush privileges;`」刷新權限表

```
MariaDB [mysql]> set password for 'root'@'localhost' = password('');
Query OK, 0 rows affected (0.002 sec)

MariaDB [mysql]> flush privileges;
Query OK, 0 rows affected (0.000 sec)

MariaDB [mysql]> -
```

圖：更新密碼後，再刷新權限表

注意：9. 的畫面，所使用密碼為「''」，為空字串

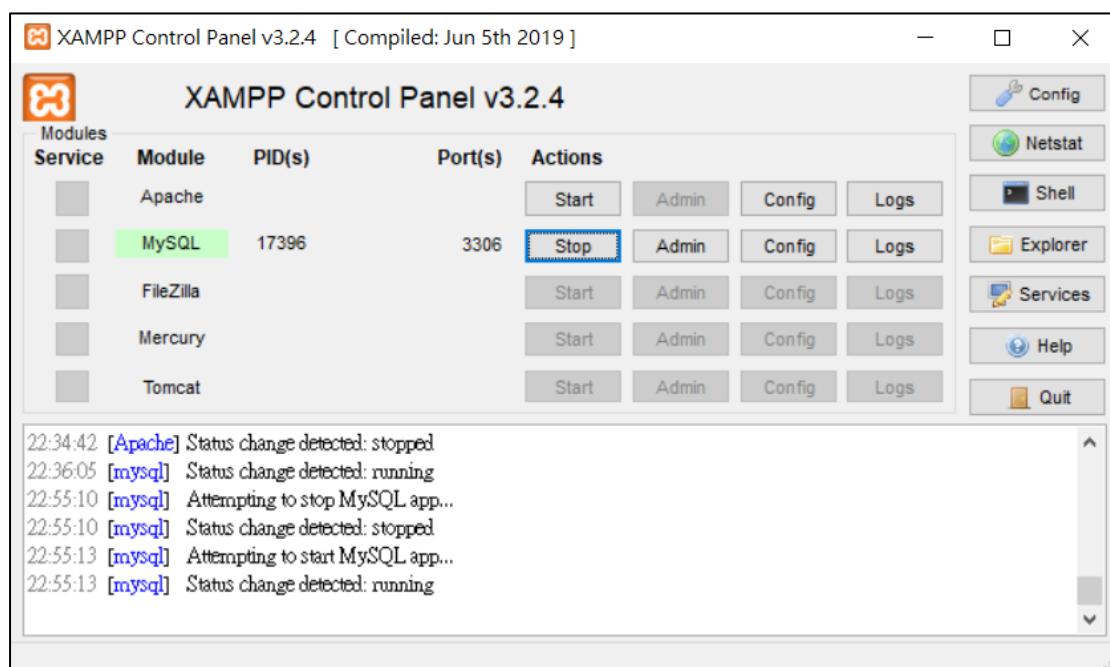
10. 輸入「`exit`」離開 MySQL 指令介面

```
MariaDB [mysql]> exit
Bye

c:\xampp\mysql\bin>
```

圖：回到命令提示字元

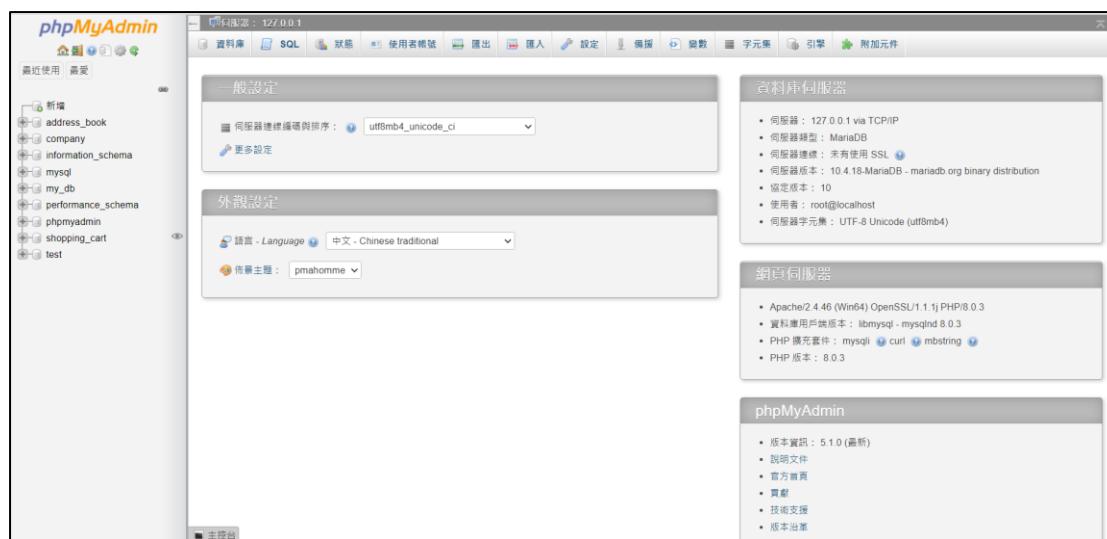
11. 重新啟動 MySQL 服務，即 Stopped 後，再按 Start，變成 running 狀態



圖：此時重新啟動 MySQL 服務

12. 回到 phpMyAdmin 網頁，此時刷新頁面 (Ctrl + F5 或是 Ctrl + R)，便

能正常進入



圖：若是使用 root 帳號，此時可以正常登入

參考資料

[1] MySQL 修改 root 密碼的 4 種方法

https://blog.csdn.net/qq_33285112/article/details/78982766

[2] mysql 错误：The MySQL server is running with the --skip-grant-tables option so it cannot execute this statement 解决方法

<https://www.cnblogs.com/kingxiaozi/p/10619680.html>

[3] xampp 修改 mysql 初始密码出现 column password is not updatable 问题及解决方案

<https://www.codenong.com/cs106631509/>

四、資料庫設計議題

資料庫的設計原則如下：

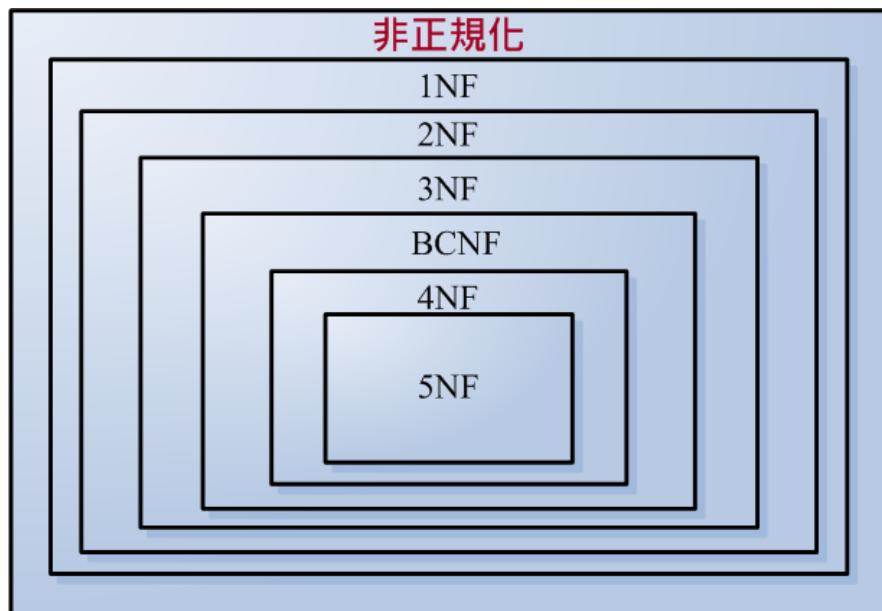
- 確認綱要 (schema) 中屬性 (attributes) 的語意 (semantic) 清楚無誤。
- 減少值組中的重覆資訊。
- 減少值組中的 NULL 值。
- 不允許產生假值組 (spurious tuple)

異常 (anomalies) 原則上可以分成三種：

- 新增異常 (insertion anomaly)
- 刪除異常 (deletion anomaly)

- 修改異常 (modification anomaly)

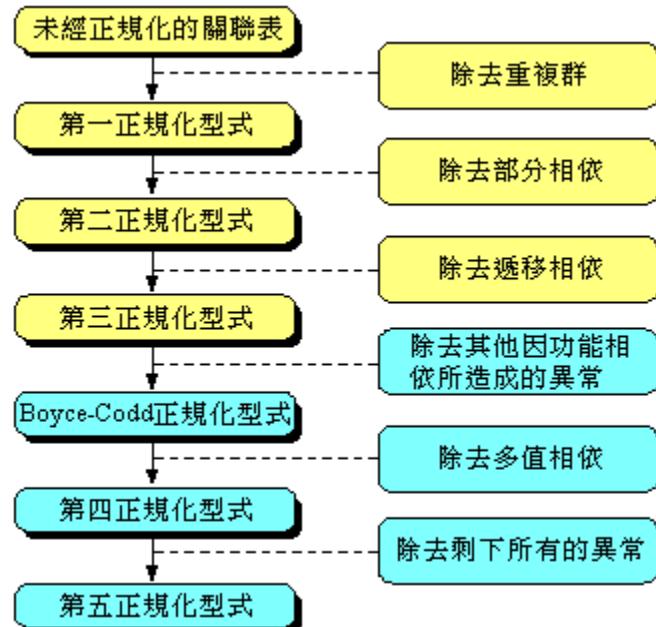
正規化 (normalization) 是在結構化分析與設計中，建立資料模式所用的技術，一種資料表分割的法則，目的是為了降低資料的重覆性，同時避免異常 (anomalies) 情況的發生。



(圖) 正規化的常見形式

正規化步驟	規則
1NF	<ul style="list-style-type: none"> 每一個欄位只能有一個值。 沒有任何資料完全重覆。 資料表中有主鍵，其它欄位相依與主鍵。
2NF	<ul style="list-style-type: none"> 符合 1NF。 每一個非鍵的屬性（例如姓名、性別）必須<u>完全相依</u>於主鍵（例如學號）。 也就是不可以部分功能相依於主鍵。
3NF	<ul style="list-style-type: none"> 符合 2NF。 各欄位與主鍵之間，沒有<u>遞移相依</u>的關係。 若要找出資料表中的遞移相依性，簡單的方法就正從左到右掃瞄各個欄位，有沒有<u>與主鍵無關的相依性</u>存在。
BCNF	<ul style="list-style-type: none"> 符合 3NF。 主鍵中的各個欄位（一張表有多個主鍵），不可以相依於其它非主鍵的欄位。

(表) 實務上最多進行到 3NF



(圖) 正規化步驟

備註：

功能相依 ($X \rightarrow Y$)，代表欄位 X 可以決定或找到欄位 Y，或是稱欄位 Y 相依於欄位 X，例如

- 「學生編號」可以決定「學生姓名」或「學生暱稱」等。
- 「老師編號」可以決定「老師姓名」等。
- 「課程編號」可以決定「學分數」或「必選修」等。

尚未正規化的資料表：學生選課資料表

學號	姓名	性別	暱稱	課程代號	課程名稱	學分數	必選修	成績	老師編號	老師姓名
087	楊○○	男	好人	C001	程式設計	4	必	74	T001	曾○○
				C002	網頁設計	3	選	93	T002	林○○
088	陳○○	女	小白	C002	網頁設計	3	選	63	T002	林○○
				C003	視覺設計	2	必	82	T003	王○○
				C004	網路教學	4	選	94	T005	謝○○

1NF 後的資料表

學號	姓名	性別	暱稱	課程 代號	課程 名稱	學分數	必選修	成績	老師 編號	老師 姓名
087	楊○○	男	好人	C001	程式設計	4	必	74	T001	曾○○
087	楊○○	男	好人	C002	網頁設計	3	選	93	T002	林○○
088	陳○○	女	小白	C002	網頁設計	3	選	63	T002	林○○
088	陳○○	女	小白	C003	視覺設計	2	必	82	T003	王○○

088	陳○○	女	小白	C004	網路教學	4	選	94	T005	謝○○
-----	-----	---	----	------	------	---	---	----	------	-----

新增異常：

學號	姓名	性別	暱稱	課程 代號	課程 名稱	學分數	必選修	成績	老師 編號	老師 姓名
NULL				C005	系統分析				NULL	

若要新增一門課程，要等學生選課、確定授課老師之後，才能新增。

修改異常：

學號	姓名	性別	暱稱	課程 代號	課程 名稱	學分數	必選修	成績	老師 編號	老師 姓名
087	楊○○	男	好人	C001	程式設計	4	必	74	T001	曾○○
087	楊○○	男	好人	C002	網頁設計	3	選	93 98	T002	林○○
088	陳○○	女	小白	C002	網頁設計	3	選	63	T002	林○○
088	陳○○	女	小白	C003	視覺設計	2	必	82	T003	王○○
088	陳○○	女	小白	C004	網路教學	4	選	94	T005	謝○○

若是希望有修「網頁設計」的同學，全班加 5 分，卻可能因為加分行為重覆多次，有些人加到、有些人沒加到，而造成資料不一致的異常現象（意即是沒有辦法透過指令進行一次性更新，甚至在更新過程中，因疏忽而缺漏修改資料，便是一種異常）。

刪除異常：

學號	姓名	性別	暱稱	課程 代號	課程 名稱	學分數	必選修	成績	老師 編號	老師 姓名
087	楊○○	男	好人	C001	程式設計	4	必	74	T001	曾○○
087	楊○○	男	好人	C002	網頁設計	3	選	93	T002	林○○
088	陳○○	女	小白	C002	網頁設計	3	選	63	T002	林○○
088	陳○○	女	小白	C003	視覺設計	2	必	82	T003	王○○
088	陳○○	女	小白	C004	網路教學	4	選	94	T005	謝○○

當我們刪除學生記錄時，同時也會刪除課程、老師的資訊，造成資料遺失的現象。

2NF 後的資料表

將部分功能相依的欄位分割出去，組成新的資料表。

↓				↓						
學號	姓名	性別	暱稱	課程 代號	課程 名稱	學分數	必選修	成績	老師 編號	老師 姓名

087	楊○○	男	好人	C001	程式設計	4	必	74	T001	曾○○
087	楊○○	男	好人	C002	網頁設計	3	選	93	T002	林○○
088	陳○○	女	小白	C002	網頁設計	3	選	63	T002	林○○
088	陳○○	女	小白	C003	視覺設計	2	必	82	T003	王○○
088	陳○○	女	小白	C004	網路教學	4	選	94	T005	謝○○

分割出去的結果：

學生資料表

學號	姓名	性別	暱稱
087	楊○○	男	好人
088	陳○○	女	小白

成績資料表

學號	課程代號	成績
087	C001	74
087	C002	93
088	C002	63
088	C003	82
088	C004	94

課程資料表

課程代號	課程名稱	學分數	必選修	老師編號	老師姓名
C001	程式設計	4	必	T001	曾○○
C002	網頁設計	3	選	T002	林○○
C002	網頁設計	3	選	T002	林○○
C003	視覺設計	2	必	T003	王○○
C004	網路教學	4	選	T005	謝○○

3NF 後的資料表

從左到右掃瞄各個欄位，有沒有與主鍵無關的相依性存在。

老師編號相依於課程代號					
課程代號	課程名稱	學分數	必選修	老師編號	老師姓名
C001	程式設計	4	必	T001	曾○○
C002	網頁設計	3	選	T002	林○○
C002	網頁設計	3	選	T002	林○○
C003	視覺設計	2	必	T003	王○○
C004	網路教學	4	選	T005	謝○○

↑

↑

老師姓名相依於老師編號

老師姓名 遞移相依 於課程代號

將「老師編號、老師姓名」獨立成一個資料，僅保留「老師編號」在課程資料表中，以便在資料查詢的過程中，還原資料的原貌。

老師資料表

老師編號	老師姓名
T001	曾○○
T002	林○○
T003	王○○
T005	謝○○

課程資料表

課程代號	課程名稱	學分數	必選修	老師編號
C001	程式設計	4	必	T001
C002	網頁設計	3	選	T002
C002	網頁設計	3	選	T002
C003	視覺設計	2	必	T003
C004	網路教學	4	選	T005

資料表正規化完成結果

學生資料表

學號	姓名	性別	暱稱
087	楊○○	男	好人
088	陳○○	女	小白

成績資料表

學號	課程代號	成績
087	C001	74
087	C002	93
088	C002	63
088	C003	82
088	C004	94

老師資料表

老師編號	老師姓名
T001	曾○○
T002	林○○
T003	王○○
T005	謝○○

課程資料表

課程代號	課程名稱	學分數	必選修	老師編號
C001	程式設計	4	必	T001
C002	網頁設計	3	選	T002
C003	視覺設計	2	必	T003
C004	網路教學	4	選	T005

五、建立資料庫、資料表

我們將透過 phpMyAdmin 的管理介面，進行資料庫建置。



(圖) 點選資料庫

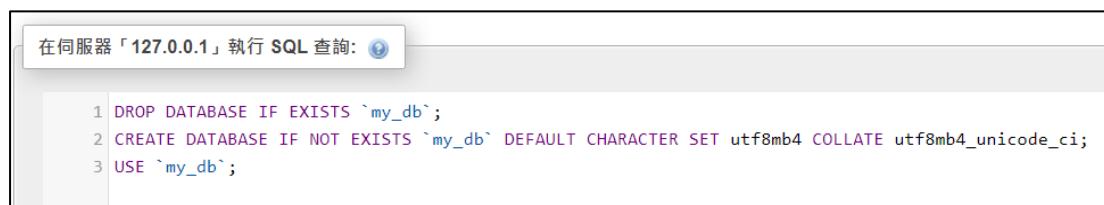


(圖) 輸入自訂的資料庫名稱，選擇排序規則

```
SQL
DROP DATABASE IF EXISTS `my_db`;

CREATE DATABASE IF NOT EXISTS `my_db` DEFAULT CHARACTER SET utf8mb4 COLLATE
utf8mb4_unicode_ci;

USE `my_db`;
```



圖：新增資料庫 my_db



(圖) 建立資料表，選擇欄位

DATE TYPE	SPEC	DATA TYPE	SPEC
CHAR	String (0 - 255)	INT	Integer (-2147483648 to 214748-3647)
VARCHAR	String (0 - 255)	BIGINT	Integer (-9223372036854775808 to 9223372036854775807)
TINYTEXT	String (0 - 255)	FLOAT	Decimal (precise to 23 digits)
TEXT	String (0 - 65535)	DOUBLE	Decimal (24 to 53 digits)
BLOB	String (0 - 65535)	DECIMAL	"DOUBLE" stored as string
MEDIUMTEXT	String (0 - 16777215)	DATE	YYYY-MM-DD
MEDIUMBLOB	String (0 - 16777215)	DATETIME	YYYY-MM-DD HH:MM:SS
LONGTEXT	String (0 - 4294967295)	TIMESTAMP	YYYYMMDDHHMMSS
LONGBLOB	String (0 - 4294967295)	TIME	HH:MM:SS
TINYINT	Integer (-128 to 127)	ENUM	One of preset options
SMALLINT	Integer (-32768 to 32767)	SET	Selection of preset options
MEDIUMINT	Integer (-8388608 to 8388607)	BOOLEAN	TINYINT(1)

圖：資料型態

依照我們先前的範例，我們可以建立正規化後的那四張表。

- students (學生資料表)
- scores (成績資料表)
- teachers (老師資料表)
- courses (課程資料表)



(圖) 學生資料表，6 個欄位

(圖) 學生資料表設計規格

(圖) 學生資料表結構

SQL 語法

```

CREATE TABLE `my_db`.`students` (
    `sId` VARCHAR(3) NOT NULL COMMENT '學生編號',
    `sName` VARCHAR(20) NOT NULL COMMENT '學生姓名',
    `sGender` VARCHAR(1) NOT NULL COMMENT '學生性別',
    `sNickname` VARCHAR(50) NOT NULL COMMENT '學生暱稱',
    `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '新增時間',
    `updated_at` DATETIME ON UPDATE CURRENT_TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
        COMMENT '更新時間',
    PRIMARY KEY (`sId`)
) ENGINE = InnoDB COMMENT = '學生資料表';

```

在資料庫 my_db 執行 SQL 查詢: [?](#)

```

1 CREATE TABLE `my_db`.`students` (
2     `sId` VARCHAR(3) NOT NULL COMMENT '學生編號',
3     `sName` VARCHAR(20) NOT NULL COMMENT '學生姓名',
4     `sGender` VARCHAR(1) NOT NULL COMMENT '學生性別',
5     `sNickname` VARCHAR(50) NOT NULL COMMENT '學生暱稱',
6     `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '新增時間',
7     `updated_at` DATETIME ON UPDATE CURRENT_TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '更新時間',
8     PRIMARY KEY (`sId`)
9 ) ENGINE = InnoDB COMMENT = '學生資料表';

```

圖：新增 students 資料表的語法



(圖) 成績資料表，5 個欄位

This screenshot displays the MySQL Workbench table structure editor for the 'scores' table. The table has five columns:

- sld**: VARCHAR(3) primary key, comment '學生編號'.
- cld**: VARCHAR(4), comment '課程編號'.
- score**: TINYINT(3), comment '成績'.
- created_at**: DATETIME with default CURRENT_TIME, comment '新增時間'.
- updated_at**: DATETIME with default CURRENT_TIMESTAMP on update, comment '更新時間'.

The table uses the InnoDB storage engine.

(圖) 成績資料表設計規格

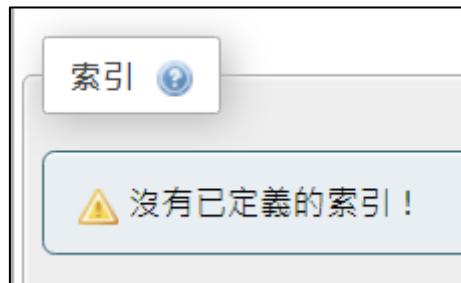
#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊
1	sId	varchar(3)	utf8mb4_unicode_ci		否	無	學生編號	
2	cId	varchar(4)	utf8mb4_unicode_ci		否	無	課程編號	
3	score	tinyint(3)			否	無	成績	
4	created_at	datetime			否	current_timestamp()	新增時間	
5	updated_at	datetime			否	current_timestamp()	更新時間	ON UPDATE CURRENT_TIMESTAMP()

(圖) 成績資料表結構(尚未建立主鍵)

SQL

```
CREATE TABLE `my_db`.`scores` (
  `sId` VARCHAR(3) NOT NULL COMMENT '學生編號' ,
  `cId` VARCHAR(4) NOT NULL COMMENT '課程編號' ,
  `score` TINYINT(3) NOT NULL COMMENT '成績' ,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '新增時間' ,
  `updated_at` TIMESTAMP on update CURRENT_TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
  COMMENT '更新時間'
) ENGINE = InnoDB COMMENT = '成績資料表';
```

在這裡，我們尚未在「索引」的下拉式選單選擇主鍵（Primary Key），待執行完成後，額外設定。



(圖) 在這裡會看見沒有定義索引的提示

1.

#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值
<input checked="" type="checkbox"/>	1 <u>sId</u>	varchar(3)	utf8mb4_unicode_ci		否	無
<input checked="" type="checkbox"/>	2 <u>cId</u>	varchar(4)	utf8mb4_unicode_ci		否	無
<input type="checkbox"/>	3 <u>score</u>	tinyint(3)			否	無
<input type="checkbox"/>	4 <u>created_at</u>	datetime			否	current_timestamp
<input type="checkbox"/>	5 <u>updated_at</u>	datetime			否	current_timestamp

2.

(圖) 選擇 sId、cId 後，按下「主鍵」圖示

#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊
<input type="checkbox"/>	1 <u>sId</u>	varchar(3)	utf8mb4_unicode_ci		否	無	學生編號	
<input type="checkbox"/>	2 <u>cId</u>	varchar(4)	utf8mb4_unicode_ci		否	無	課程編號	
<input type="checkbox"/>	3 <u>score</u>	tinyint(3)			否	無	成績	
<input type="checkbox"/>	4 <u>created_at</u>	datetime			否	current_timestamp()	新增時間	
<input type="checkbox"/>	5 <u>updated_at</u>	datetime			否	current_timestamp()	更新時間	ON UPDATE CURRENT_TIMESTAMP()

(圖) 看到 sId 和 cId 右側，各有一把主鍵圖示

SQL
ALTER TABLE `scores` ADD PRIMARY KEY(`sId`, `cId`);

建立資料表

名稱: teachers 欄位數: 4

執行

(圖) 老師資料表，4 個欄位

資料表名稱: teachers
新增 1 欄位 執行

名稱	類型	長度 / 值	預設值	編碼與排序	屬性	空值 (Null)	索引	A_J	備註
tId	VARCHAR	4	無			PRIMARY			老師編號
tName	VARCHAR	10	無						老師姓名
created_at	DATETIME		CURRENT_TIME						新增時間
updated_at	DATETIME		CURRENT_TIME		on update CURRENT_TIMESTAMP				更新時間

資料表備註: 儲存引擎: InnoDB 連線:

(圖) 老師資料表設計規格

資料表結構

#	名稱	類型	編碼與排序	屬性	空值 (Null)	預設值	備註	額外資訊
1	tId	varchar(4)	utf8mb4_unicode_ci		否	無	老師編號	
2	tName	varchar(10)	utf8mb4_unicode_ci		否	無	老師姓名	
3	created_at	datetime			否	current_timestamp()	新增時間	
4	updated_at	datetime			否	current_timestamp()	更新時間	ON UPDATE CURRENT_TIMESTAMP

(圖) 老師資料表結構

SQL

```
CREATE TABLE `my_db`.`teachers` (
  `tId` VARCHAR(4) NOT NULL COMMENT '老師編號',
  `tName` VARCHAR(10) NOT NULL COMMENT '老師姓名',
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '新增時間',
  `updated_at` DATETIME ON UPDATE CURRENT_TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
  COMMENT '更新時間',
  PRIMARY KEY (`tId`)
) ENGINE = InnoDB COMMENT = '老師資料表';
```

建立資料表

名稱: courses 欄位數: 7

執行

(圖) 課程資料表，7 個欄位

資料表名稱: courses
 新增 1 單位 (執行)

名稱	類型	長度 / 值	預設值	編碼與排序	屬性	空值 (Null)	索引	A	備註
cId	VARCHAR	4	無			□	---	□	課程編號
cName	VARCHAR	10	無			□	---	□	課程名稱
credit	TINYINT	1	無			□	---	□	學分
isCompulsory	TINYINT	1	無			□	---	□	是否必修
tId	VARCHAR	4	無			□	---	□	老師編號
created_at	DATETIME		CURRENT_TIME			□	---	□	新增時間
updated_at	DATETIME		CURRENT_TIME		on update CURRENT_TIMESTAMP	□	---	□	更新時間

結構
 資料表備註：
 儲存引擎： InnoDB
 連線：

(圖) 課程資料表設計規格

資料表結構

#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊
1	cId	varchar(4)	utf8mb4_unicode_ci	否	無		課程編號	
2	cName	varchar(10)	utf8mb4_unicode_ci	否	無		課程名稱	
3	credit	tinyint(1)		否	無		學分	
4	isCompulsory	tinyint(1)		否	無		是否必修	
5	tId	varchar(4)	utf8mb4_unicode_ci	否	無		老師編號	
6	created_at	datetime		否	current_timestamp()		新增時間	
7	updated_at	datetime		否	current_timestamp()	ON UPDATE CURRENT_TIMESTAMP()	更新時間	

圖：課程資料表結構(尚未建立索引)

SQL

```

CREATE TABLE `my_db`.`courses` (
  `cId` VARCHAR(4) NOT NULL COMMENT '課程編號',
  `cName` VARCHAR(10) NOT NULL COMMENT '課程名稱',
  `credit` TINYINT(1) NOT NULL COMMENT '學分',
  `isCompulsory` TINYINT(1) NOT NULL COMMENT '是否必修',
  `tId` VARCHAR(4) NOT NULL COMMENT '老師編號',
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '新增時間',
  `updated_at` DATETIME ON UPDATE CURRENT_TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
)

```

```
COMMENT '更新時間'
) ENGINE = InnoDB COMMENT = '課程資料表';
```



(圖) 沒有索引的提示

1.	#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值
<input checked="" type="checkbox"/>	1	cld	varchar(4)	utf8mb4_unicode_ci	否	無	
<input type="checkbox"/>	2	cName	varchar(10)	utf8mb4_unicode_ci	否	無	
<input type="checkbox"/>	3	credit	tinyint(1)		否	無	
2.	4	isCompulsory	tinyint(1)		否	無	
<input checked="" type="checkbox"/>	5	tld	varchar(4)	utf8mb4_unicode_ci	否	無	
<input type="checkbox"/>	6	created_at	datetime		否	current_timestamp()	
<input type="checkbox"/>	7	updated_at	datetime		否	current_timestamp()	

3. 主鍵

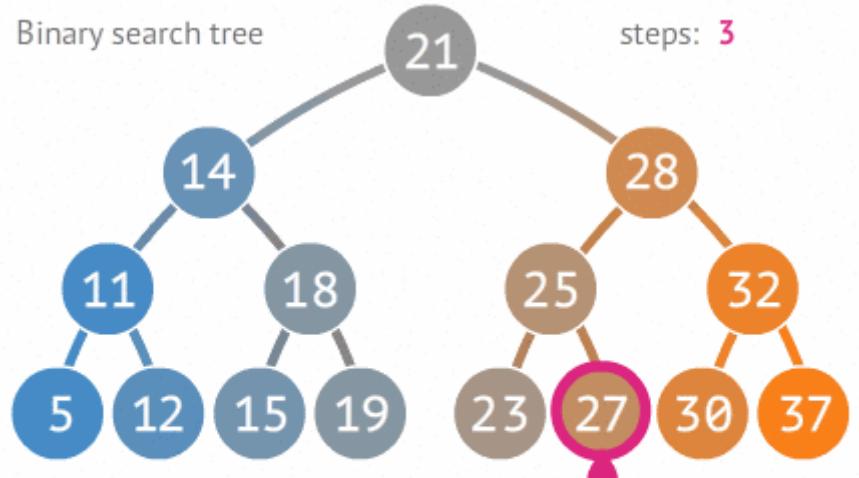
Buttons at the bottom: 全選, 已選擇項目 : , 檢覽, 修改, 刪除, 主鍵, 獨一, 索引.

(圖) 勾選 cId、tId 後，按下「主鍵」圖示

#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊
<input type="checkbox"/>	1 cld	varchar(4)	utf8mb4_unicode_ci	否	無		課程編號	
<input type="checkbox"/>	2 cName	varchar(10)	utf8mb4_unicode_ci	否	無		課程名稱	
<input type="checkbox"/>	3 credit	tinyint(1)		否	無		學分	
<input type="checkbox"/>	4 isCompulsory	tinyint(1)		否	無		是否必修	
<input type="checkbox"/>	5 tld	varchar(4)	utf8mb4_unicode_ci	否	無		老師編號	
<input type="checkbox"/>	6 created_at	datetime		否	current_timestamp()		新增時間	
<input type="checkbox"/>	7 updated_at	datetime		否	current_timestamp()		更新時間	ON UPDATE CURRENT_TIMESTAMP()

(圖) 課程資料表結構

建立索引的目的，在於提升資料查詢的效率，我們可以針對單獨的欄位建立索引，也可以勾選多欄位，成立索引。讓我們檢視學生資料表的的結構，以及它的索引設定（主鍵也是一種索引）。



(圖) 建立索引後的資料儲存方式

我們可以設定單一欄位為索引：

A screenshot of the MySQL Workbench interface showing the 'Indexes' tab for a table. The table has six columns: sId, sName, sGender, sNickname, created_at, and updated_at. The 'sNickname' column is selected and checked for indexing. The 'Index' button at the bottom is highlighted with a red box and the number '2.'.

#	名稱	類型	編碼與排序	屬性	空值(NULL)	預設值	備註	額外資訊
1	sId	varchar(3)	utf8mb4_unicode_ci	否	無		學生編號	
2	sName	varchar(20)	utf8mb4_unicode_ci	否	無		學生姓名	
3	sGender	varchar(1)	utf8mb4_unicode_ci	否	無		學生性別	
4	sNickname	varchar(50)	utf8mb4_unicode_ci	否	無		學生暱稱	
5	created_at	datetime		否	current_timestamp()		新增時間	
6	updated_at	datetime		否	current_timestamp()	ON UPDATE CURRENT_TIMESTAMP()	更新時間	

(圖) 單一欄位建立索引

A screenshot of the MySQL Workbench interface showing the 'Indexes' tab for the same table. It lists two indexes: 'PRIMARY' (the primary key) and 'sNickname'. The 'sNickname' index is shown with a unique constraint and a non-clustered storage type.

動作	鍵名	類型	獨一	緊湊	欄位	基數	編碼與排序	空值(NULL)	備註
編輯	PRIMARY	BTREE	是	否	sId	0	A	否	
編輯	sNickname	BTREE	否	否	sNickname	0	A	否	

(圖) 除了主鍵，多了一個索引

SQL - 新增單一欄位索引

```
ALTER TABLE `students` ADD INDEX(`sNickname`);
```

SQL - 刪除單一欄位索引

```
ALTER TABLE `students` DROP INDEX `sNickname`;
```

六、新增、修改、刪除資料

格式

```
INSERT INTO `資料表名稱` (`欄位 1`, `欄位 2`, `欄位 3`, ..., `欄位 N`)
VALUES (值 1, 值 2, 值 3, ..., 值 N),
       (值 1, 值 2, 值 3, ..., 值 N),
       (值 1, 值 2, 值 3, ..., 值 N);
```

sId 學生編號	sName 學生姓名	sGender 學生性別	sNickname 學生暱稱	created_at 新增時間	updated_at 更新時間
087	楊oo	男	好人	2019-12-02 11:11:37	2019-12-02 11:11:37
088	陳oo	女	小白	2019-12-02 11:11:37	2019-12-02 11:11:37

(圖) 學生資料表 - 新增前

SQL

```
INSERT INTO `students` (`sId`, `sName`, `sGender`, `sNickname`)
VALUES
    ('003', '王oo', '男', '小王'),
    ('004', '江oo', '女', '小江'),
    ('005', '周oo', '女', '小周'),
    ('006', '黃oo', '男', '小黃'),
    ('007', '丁oo', '男', '小丁'),
    ('008', '鄭oo', '男', '小鄭'),
    ('087', '楊oo', '男', '好人'),
    ('088', '陳oo', '女', '小白');
```

sId 學生編號	sName 學生姓名	sGender 學生性別	sNickname 學生暱稱	created_at 新增時間	updated_at 更新時間
003	王oo	男	小王	2019-12-02 18:27:16	2019-12-02 18:27:16
004	江oo	女	小江	2019-12-02 18:27:16	2019-12-02 18:27:16
005	周oo	女	小周	2019-12-02 18:27:16	2019-12-02 18:27:16
006	黃oo	男	小黃	2019-12-02 18:27:16	2019-12-02 18:27:16
007	丁oo	男	小丁	2019-12-02 18:27:16	2019-12-02 18:27:16
008	鄭oo	男	小鄭	2019-12-02 18:27:16	2019-12-02 18:27:16
087	楊oo	男	好人	2019-12-02 11:11:37	2019-12-02 11:11:37
088	陳oo	女	小白	2019-12-02 11:11:37	2019-12-02 11:11:37

(圖) 學生資料表 - 新增後

格式
UPDATE `資料表名稱`
SET
`欄位 1` = 值 1,
`欄位 2` = 值 2,
`欄位 3` = 值 3
WHERE `欄位 A` = 值 A
AND `欄位 B` = 值 B
OR `欄位 C` = 值 C

tId 老師編號	tName 老師姓名	created_at 新增時間	updated_at 更新時間
T001	曾oo	2019-12-02 00:00:00	2019-12-02 00:00:00
T002	林oo	2019-12-02 00:00:00	2019-12-02 00:00:00
T003	王oo	2019-12-02 00:00:00	2019-12-02 00:00:00
T005	謝oo	2019-12-02 00:00:00	2019-12-02 00:00:00

(圖) 老師資料表 - 修改前

SQL
UPDATE `teachers`
SET `tName` = '黃oo'
WHERE `tId` = 'T001'

tId 老師編號	tName 老師姓名
T001	黃oo
T002	林oo
T003	王oo
T005	謝oo

(圖) 老師料表 - 修改後

格式

```
DELETE FROM `資料表名稱`  

WHERE `欄位 1` = 值 1,  

AND `欄位 B` = 值 B  

OR `欄位 C` = 值 C;
```

sId 學生編號	cId 課程編號	score 成績
087	C001	74
087	C002	93
088	C002	63
088	C003	82
088	C004	94

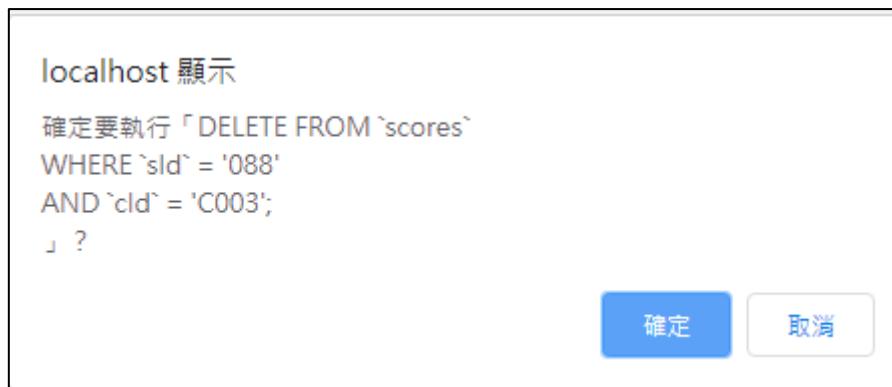
(圖) 成績資料表 - 刪除前

SQL

```
DELETE FROM `scores`  

WHERE `sId` = '088'  

AND `cId` = 'C003';
```



(圖) 跳出警示視窗

sld 學生編號	cld 課程編號	score 成績
087	C001	74
087	C002	93
088	C002	63
088	C004	94

(圖) 成績資料表 - 刪除後

七、資料庫基本查詢

格式
<pre>SELECT * FROM `資料表名稱` WHERE `欄位 A` = 值 1 AND `欄位 B` > 值 B, OR `欄位 C` != 值 C ORDER BY `欄位 D` ASC</pre>

格式
<pre>SELECT `欄位 1`, `欄位 2`, ..., `欄位 N` FROM `資料表名稱` WHERE `欄位 A` = 值 1 AND `欄位 B` > 值 B, OR `欄位 C` != 值 C</pre>

```
ORDER BY `欄位 D` ASC
```

格式

```
SELECT `資料庫`.`資料表名稱`.`欄位 1`,
       `資料庫`.`資料表名稱`.`欄位 2`, ... ,
       `資料庫`.`資料表名稱`.`欄位 N`
FROM `資料庫`.`資料表名稱`
WHERE `資料庫`.`資料表名稱`.`欄位 A` = 值 1
AND `資料庫`.`資料表名稱`.`欄位 B` > 值 B,
OR `資料庫`.`資料表名稱`.`欄位 C` != 值 C
ORDER BY `資料庫`.`資料表名稱`.`欄位 D` ASC
```

以上課的「my_db」資料為例：

M0：列出所有的學生資料

```
SELECT *
FROM `students`
```

sId 學生編號	sName 學生姓名	sGender 學生性別	sNickname 學生暱稱
003	王。。	男	小王
004	江。。	女	小江
005	周。。	女	小周
006	黃。。	男	小黃
007	丁。。	男	小丁
008	鄭。。	男	小鄭
087	楊。。	男	好人
088	陳。。	女	小白

(圖) M0 執行結果

M1：列出學生的學號、姓名、暱稱

```
SELECT `sId`, `sName`, `sNickname`
FROM `students`
```

sId 學生編號	sName 學生姓名	sNickname 學生暱稱
003	王oo	小王
004	江oo	小江
005	周oo	小周
006	黃oo	小黃
007	丁oo	小丁
008	鄭oo	小鄭
087	楊oo	好人
088	陳oo	小白

(圖) M1 執行結果

M2: 找出老師編號為 T001 或 T003 的老師姓名

```
SELECT `tName`
FROM `teachers`
WHERE `tId` = 'T001'
OR `tId` = 'T003'
```

tName 老師姓名
黃oo
王oo

(圖) M2 執行結果

M3: 將學生依學生編號排序，號碼最大的排前面，全部列出來

```
SELECT *
FROM `students`
ORDER BY `students`.`sId` DESC
```

sId	sName	sGender	sNickname
學生編號	學生姓名	學生性別	學生暱稱
088	陳○○	女	小白
087	楊○○	男	好人
008	鄭○○	男	小鄭
007	丁○○	男	小丁
006	黃○○	男	小黃
005	周○○	女	小周
004	江○○	女	小江
003	王○○	男	小王

(圖) M3 執行結果

M4: 找出性別為女、暱稱為小白的學生姓名

```
SELECT `sName`
FROM `students`
WHERE `sGender` = '女'
AND `sNickname` = '小白';
```

M4A: 找出性別為女、暱稱為小白的學生姓名（用 AS 來替代資料表名稱）

```
SELECT `s`.`sName`
FROM `students` AS `s`
WHERE `s`.`sGender` = '女'
AND `s`.`sNickname` = '小白';
```

補充說明：

‘資料表’ AS ‘資料表代稱’，其中 AS 可以省略。

sName 學生姓名
陳○○

(圖) M4、M4A 執行結果

以下將透過 LIKE 關鍵字，搭配 SQL 內建的「%」、「_」模糊查詢功能，來查詢資料：

- %：配對任何文字，同時字數長度不限，可以放置在字串的前後，或是在每個分隔文字之間。例如
 - %人：可配對「好人」、「擺渡人」、「這裡沒有人」等。

- %：可配對「**好人**」、「**好東西**」、「**好像很厲害**」等。
- %**好人**：可配對「**做好人**」、「**當個好人**」、「**你真是個好人**」等。
- **好人%**：可配對「**好人家**」、「**好人好事**」、「**好人村村長**」等
- %**好人%**：可配對「**是好人嗎**」、「**走入好人村**」、「**當個好人村村長**」。
- **好%人**：可配對「**好男人**」、「**好嚇唬人**」等。
- %**好%人%**：可配對「**新好男人！**」、「**好好照顧人家**」等。
- _:配對任何文字，一個「_」代表一個字，可以放置在字串的前後，或是在每個分隔文字之間。例如
 - _**人**：可配對「**好人**」、「**男人**」、「**女人**」等。
 - **好_**：可配對「**好人**」、「**好事**」、「**好高**」等。
 - **_好人**：可配對「**當好人**」、「**做好人**」等。
 - **好人_**：可配對「**好人家**」、「**好人選**」、「**好人村**」等。
 - **_好人_**：可配對「**是好人嗎**」、「**良好人選**」等。
 - **好_人**：可配對「**好男人**」、「**好女人**」等。
 - **_好_人_**：可配對「**當好男人吧**」、「**有好男人嗎**」等。

M5: 找出暱稱是「小」開頭的學生姓名與性別

```
SELECT `sName`, `sGender`
FROM `students`
WHERE `sNickname` LIKE '小%'
```

M5A: 找出暱稱是「人」結尾的學生姓名與性別

```
SELECT `sName`, `sGender`
FROM `students`
WHERE `sNickname` LIKE '%人'
```

sName 學生姓名	sGender 學生性別
王。。	男
江。。	女
周。。	女
黃。。	男
丁。。	男
鄭。。	男
陳。。	女

sName 學生姓名	sGender 學生性別
楊。。	男

(圖) M5A 執行結果

(圖) M5 執行結果

M6: 找出學號（三位數）第二碼為 8 的學生編號、姓名與暱稱

```
SELECT `sId`, `sName`, `sNickname`
FROM `students`
WHERE `sId` LIKE '_8_'
```

sId 學生編號	sName 學生姓名	sNickname 學生暱稱
087	楊○○	好人
088	陳○○	小白

(圖) M6 執行結果

M7: 找出學生資料表當中，學生編號為 003、005、007、087 等四位學生的所有資料。

```
SELECT *
FROM `students`
WHERE `sId` IN ('003', '005', '007', '087');
```

sId 學生編號	sName 學生姓名	sGender 學生性別	sNickname 學生暱稱	created_at 新增時間	updated_at 更新時間
003	王○○	男	小王	2019-12-02 18:27:16	2019-12-02 18:27:16
005	周○○	女	小周	2019-12-02 18:27:16	2019-12-02 18:27:16
007	丁○○	男	小丁	2019-12-02 18:27:16	2019-12-02 18:27:16
087	楊○○	男	好人	2019-12-02 11:11:37	2019-12-02 11:11:37

圖：M7 行結果

參考資料：

- [1] MySQL 模糊匹配查詢 like、regexp、in
<https://www.itread01.com/content/1539363869.html>

接下來，我們將使用 LIMIT 語法，模擬資料「分頁」(pagination)的情形。首先，我們看一下學生資料表的全貌：

sId 學生編號	sName 學生姓名	sGender 學生性別	sNickname 學生暱稱	created_at 新增時間	updated_at 更新時間
003	王○○	男	小王	2019-12-02 18:27:16	2019-12-02 18:27:16
004	江○○	女	小江	2019-12-02 18:27:16	2019-12-02 18:27:16
005	周○○	女	小周	2019-12-02 18:27:16	2019-12-02 18:27:16
006	黃○○	男	小黃	2019-12-02 18:27:16	2019-12-02 18:27:16
007	丁○○	男	小丁	2019-12-02 18:27:16	2019-12-02 18:27:16
008	鄭○○	男	小鄭	2019-12-02 18:27:16	2019-12-02 18:27:16
087	楊○○	男	好人	2019-12-02 11:11:37	2019-12-02 11:11:37
088	陳○○	女	小白	2019-12-02 11:11:37	2019-12-02 11:11:37

圖：學生資料表所有資料一覽

M8：依學生編號由小到大排序，取前三筆所有欄位資料

```
SELECT *
FROM `students`
ORDER BY `sId` ASC
LIMIT 3
```

sid ▲ 1 學生編號	sName 學生姓名	sGender 學生性別	sNickname 學生暱稱	created_at 新增時間	updated_at 更新時間
003	王○○	男	小王	2019-12-02 18:27:16	2019-12-02 18:27:16
004	江○○	女	小江	2019-12-02 18:27:16	2019-12-02 18:27:16
005	周○○	女	小周	2019-12-02 18:27:16	2019-12-02 18:27:16

圖：M8 執行結果

M8-1：依學生編號由小到大排序，從第 1 筆開始，取得 3 筆所有欄位資料

```
SELECT *
FROM `students`
ORDER BY `sId` ASC
LIMIT 0, 3
```

sid ▲ 1 學生編號	sName 學生姓名	sGender 學生性別	sNickname 學生暱稱	created_at 新增時間	updated_at 更新時間
003	王○○	男	小王	2019-12-02 18:27:16	2019-12-02 18:27:16
004	江○○	女	小江	2019-12-02 18:27:16	2019-12-02 18:27:16
005	周○○	女	小周	2019-12-02 18:27:16	2019-12-02 18:27:16

圖：M8-1 執行結果

M8-2：依學生編號由小到大排序，從第 3 筆開始，取得 3 筆所有欄位資料

```
SELECT *
FROM `students`
ORDER BY `sId` ASC
LIMIT 3, 3
```

sid ▲ 1 學生編號	sName 學生姓名	sGender 學生性別	sNickname 學生暱稱	created_at 新增時間	updated_at 更新時間
006	黃○○	男	小黃	2019-12-02 18:27:16	2019-12-02 18:27:16
007	丁○○	男	小丁	2019-12-02 18:27:16	2019-12-02 18:27:16
008	鄭○○	男	小鄭	2019-12-02 18:27:16	2019-12-02 18:27:16

圖：M8-2 執行結果

M8-3: 依學生編號由小到大排序，從第 6 筆開始，取得 3 筆所有欄位資料

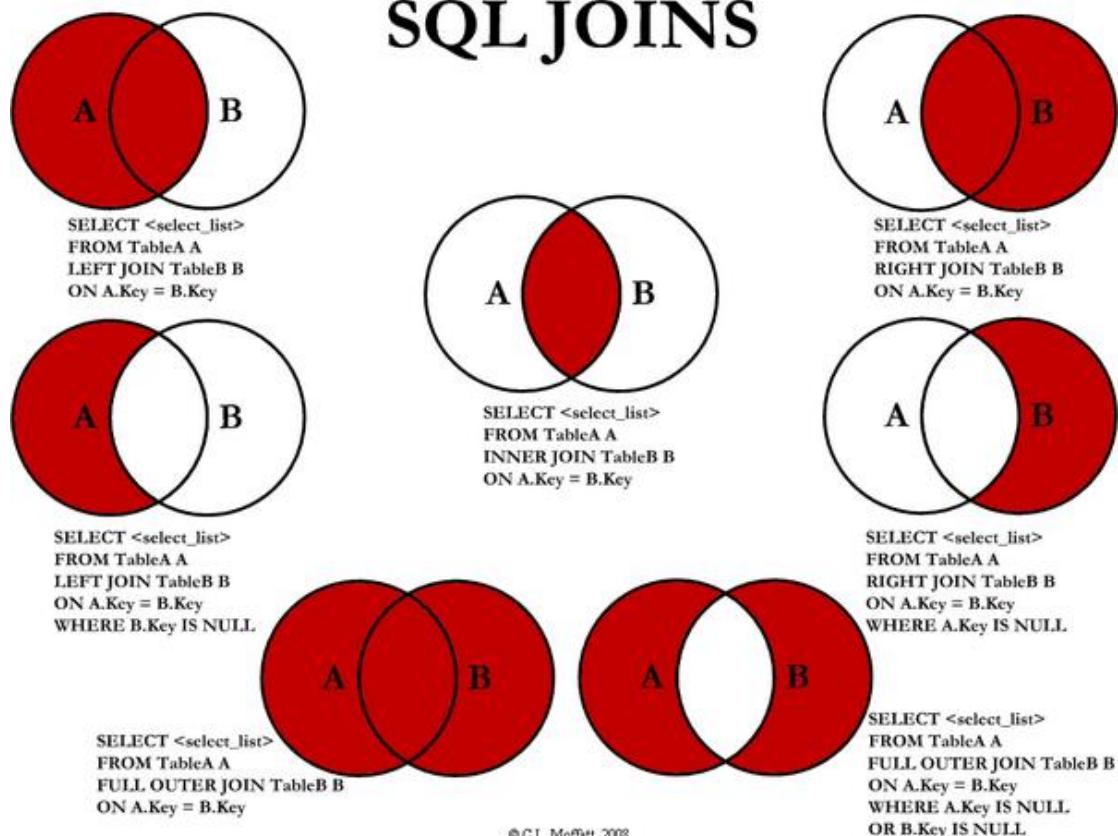
```
SELECT *
FROM `students`
ORDER BY `sId` ASC
LIMIT 6, 3
```

std ▲ 1 學生編號	sName 學生姓名	sGender 學生性別	sNickname 學生暱稱	created_at 新增時間	updated_at 更新時間
087	楊○○	男	好人	2019-12-02 11:11:37	2019-12-02 11:11:37
088	陳○○	女	小白	2019-12-02 11:11:37	2019-12-02 11:11:37

圖：M8-3 執行結果

八、進階查詢

SQL JOINS



(圖) 資料表結合的常見態樣

格式

```
SELECT `欄位 1`, `欄位 2`
```

```
FROM `資料表 1` INNER JOIN `資料表 2`
ON `資料表 1`.`欄位 A` = `資料表 2`.`欄位 A`
```

說明：

INNER JOIN 兩個資料表進行內部結合

LEFT JOIN 以左邊資料表為主，進行結合

RIGHT JOIN 以右邊資料表為主，進行結合

OUTTER JOIN

M0：查詢各個課程中的授課老師姓名

```
SELECT `cName`, `tName`
FROM `courses` INNER JOIN `teachers`
ON `courses`.`tId` = `teachers`.`tId`
```

cName 課程名稱	tName
程式設計	黃○○
網頁設計	林○○
視覺設計	王○○
網路教學	謝○○

(圖) M0 執行結果

M1：列出所有擁有成績的學生姓名、課程名稱與分數

```
SELECT `sName`, `cName`, `score`
FROM `scores`
INNER JOIN `students`
ON `scores`.`sId` = `students`.`sId`
INNER JOIN `courses`
ON `scores`.`cId` = `courses`.`cId`
```

sName	cName	score 成績
楊○○	程式設計	74
楊○○	網頁設計	93
陳○○	網頁設計	63
陳○○	視覺設計	82
陳○○	網路教學	94

(圖) M1 執行結果

接下來，請新增 company 資料庫，並匯入 company.sql。



圖：新增 company 資料庫



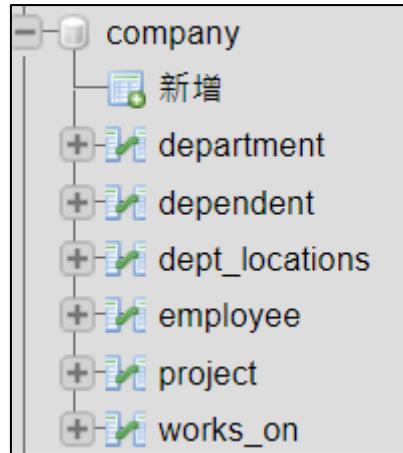
圖：點選左側的 company 資料庫



圖：按下匯入



圖：選擇 company.sql 檔案，按下執行



圖：原先空的 company 資料庫，會多出幾個資料表

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Q1: 找出在 Research 部門工作的所有員工姓名與住址

```
SELECT `employee`.`Fname`, `employee`.`Lname`,
`employee`.`Address`
FROM `employee` INNER JOIN `department`
ON `employee`.`Dno` = `department`.`Dnumber`
WHERE `department`.`Dname` = 'Research'
```

Fname 名	Lname 姓	Address 地址
John	Smith	731 Fondren, Houston, TX
Franklin	Wong	638 Voss, houston, TX
Joyce	English	5631 Rice, Houston, TX
Ramesh	Narayan	975 Fire Oak, Humble, TX

(圖) Q1 執行結果

Q2: 列出所有位在 Stafford 地點的計畫，其計畫編號、部門編號，以及部門經理的姓氏、住址和生日

```
SELECT `project`.`Pnumber`, `project`.`Dnum`,
`employee`.`Lname`, `employee`.`Address`, `employee`.`Bdate`
FROM `project`
INNER JOIN `department`
ON `project`.`Dnum` = `department`.`Dnumber`
INNER JOIN `employee`
ON `department`.`Mgr_ssn` = `employee`.`Ssn`
WHERE `project`.`Plocation` = 'Stafford'
```

Pnumber	Dnum	Lname 姓	Address 地址	Bdate 生日
10	4	Wallace	291 Berry, Bellaire, TX	1941-06-20
30	4	Wallace	291 Berry, Bellaire, TX	1941-06-20

(圖) Q2 執行結果

Q3: 查詢每一位員工的姓名與其直屬上司的姓名

```
SELECT `E`.`Fname`, `E`.`Lname`, `S`.`Fname`, `S`.`Lname`
FROM `employee` AS `E`
INNER JOIN `employee` AS `S`
ON `E`.`Super_ssn` = `S`.`Ssn`
```

Fname	Lname	Fname	Lname
John	Smith	Franklin	Wong
Franklin	Wong	James	Borg
Joyce	English	Franklin	Wong
Ramesh	Narayan	Franklin	Wong
Jennifer	Wallace	James	Borg
Ahmad	Jabbar	Jennifer	Wallace
Alicia	Zelaya	Jennifer	Wallace

(圖) Q8 執行結果

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

(圖) EMPLOYEE 資料表的內容

由於該資料表同時有員工的 Ssn 和上級 Superior 的 Ssn (Super_ssn)，我們可以試著查詢每一個員工的頂頭上司是誰。

Q4: 查詢每一個員工的頂頭上司是誰，包括員工本身的姓名、Ssn，以及上級的姓名、SSn

```

SELECT `InferE`.`Fname` AS `InferE_Fname`,
`InferE`.`Lname` AS `InferE_Lname`,
`InferE`.`Ssn` AS `InferE_Ssn`,
`SuperE`.`Fname` AS `SuperE_Fname`,
`SuperE`.`Lname` AS `SuperE_Lname`,
`SuperE`.`Ssn` AS `SuperE_Ssn`
FROM `employee` AS `InferE`
LEFT JOIN `employee` AS `SuperE`
ON `InferE`.`Super_ssn` = `SuperE`.`Ssn`
```

InterE_Fname	InterE_Lname	InterE_Ssn	SuperE_Fname	SuperE_Lname	SuperE_Ssn
John	Smith	123456789	Franklin	Wong	333445555
Franklin	Wong	333445555	James	Borg	888665555
Joyce	English	453453453	Franklin	Wong	333445555
Ramesh	Narayan	666884444	Franklin	Wong	333445555
James	Borg	888665555	NULL	NULL	NULL
Jennifer	Wallace	987654321	James	Borg	888665555
Ahmad	Jabbar	987987987	Jennifer	Wallace	987654321
Alicia	Zelaya	999887777	Jennifer	Wallace	987654321

(圖) Q1 執行結果

補充說明

from 和 join 均是用於指定需要從哪些表查詢數據。

- from 可以是一個資料表或多個資料表，如果是多個資料表則是生成一個笛卡爾集，會涉及到大量數據。所以通常在涉及到多個資料表的查詢時，通常通過 join 來拼接多個資料表。故如果資料表的數據行多時，則數據量很多，造成巨大的磁碟、記憶體開銷，當然查詢速度也會很慢。
- join 主要是通過多個資料表之間的外鍵關聯來進行拼接，注意用於拼接的鍵需要加上索引，如果沒有則 MySQL 也會默認加上，不過前提是外鍵和引用的主鍵需要是相同的資料型態如數字類型需要是相同的長度和均是有符號或無符號數，字符串類型長度可以不一樣。

九、函式

MySQL 的內建函式繁不及備載，以下僅列出課程會用到函式。

參考連結：

MySQL Functions

https://www.w3schools.com/mysql/mysql_ref_functions.asp

格式 1

SELECT 函式(`欄位`)

FROM `資料表`

備註：也可以加上 WHERE 條件來過濾資料

M1：查詢所有課程中，最大的學分數是多少

SELECT MAX(`credit`)

FROM `courses`

MAX(`credit`)
4

(圖) M1 執行結果

格式 2

```
SELECT `欄位 1`, `欄位 2`, 函式(`欄位 3`)
FROM `資料表`
WHERE `欄位 A` = `欄位 B`
GROUP BY `欄位 1`, `欄位 2`
HAVING 函式(`欄位 3`) > 值
```

以下為聚合 (aggregation) 的範例，類似把離散的資料加以群組起來的概念：

M2：查詢各個學分數，相對應的課程各有幾個

```
SELECT `credit`, COUNT(`cId`)
FROM `courses`
GROUP BY `credit`
```

credit 學分數	COUNT(`cId`)
2	1
3	1
4	2

(圖) M2 執行結果

M3：計算平均成績

```
SELECT AVG(`score`)
FROM `scores`
```

AVG(`score`)
81.2000

(圖) M3 執行結果。結果可能隨著課堂練習而有不同。

M4：查詢各個課程的平均成績

```
SELECT `cId`, AVG(`score`)
FROM `scores`
GROUP BY `cId`
```

cid 課程編號	AVG(`score`)
C001	74.0000
C002	78.0000
C004	94.0000

(圖) M4 執行結果

函式	說明
CASE()	<pre>SELECT CASE column WHEN a THEN b WHEN c THEN d ELSE e END</pre> <p>若 column 等於 a，則返回 b 若 column 等於 c，則返回 d 否則返回 e 另外，a 與 c 可為運算式，例如 $a < 10$，只要是 a 小於 10 都返回 b</p>
IF()	<pre>SELECT IF(input, x, y)</pre> <p>IF() 返回一個數字或字符串值。 如果 input 是 TRUE (input 不為 0 且 x 不為 NULL)，那麼 IF() 返回 x，否則它返回 y。 類似 PHP 中的 三元運算子</p>
DATE_FORMAT()	參數分別為 (你指定的日期時間，格式化字串)。

CASE()**範例**

```
SELECT `sId`, `cId`, `score`,
CASE
    WHEN `score` >= 90 THEN '成績在 90 分以上'
    WHEN `score` < 90 AND `score` >= 70 THEN '成績在 70 到 89 分之間'
    ELSE '成績低於 70'
END AS `msg`
FROM `scores`
```

sId 學生編號	cId 課程編號	score 成績	msg
087	C001	74	成績在70到89分之間
087	C002	93	成績在90分以上
088	C002	63	成績低於70
088	C004	94	成績在90分以上

(圖) CASE() 執行結果

IF()**範例**

```
SELECT `sId`, `cId`, `score`, IF(`score` >= 90, "大於 90 分", "低於 90 分") AS `msg`
FROM `scores`;
```

sId 學生編號	cId 課程編號	score 成績	msg
087	C001	74	低於 90 分
087	C002	93	大於 90 分
088	C002	63	低於 90 分
088	C004	94	大於 90 分

(圖) IF() 執行結果

F1: 找出所有課程的成績，判斷是否高於或低於平均成績並印出提示文字

```
SELECT `sId`, `cId`, `score`,
CASE
    WHEN `score` >= 90 THEN '成績在 90 分以上'
    WHEN `score` < 90 AND `score` >= 70 THEN '成績在 70 到 89 分之間'
    ELSE '成績低於 70'
END AS `msg`, (SELECT AVG(`score`) FROM `scores`) AS `avg_scores`,
CASE
    WHEN `score` >= (SELECT AVG(`score`) FROM `scores`) THEN '大於等於平均'
    ELSE '小於平均'
END AS `avg_msg`
FROM `scores`
```

slid 學生編號	cld 課程編號	score 成績	msg	avg_scores	avg_msg
087	C001	74	成績在70到89分之間	81.2000	小於平均
087	C002	93	成績在90分以上	81.2000	大於等於平均
088	C002	63	成績低於70	81.2000	小於平均
088	C003	82	成績在70到89分之間	81.2000	大於等於平均
088	C004	94	成績在90分以上	81.2000	大於等於平均

(圖) C2 執行結果

F2: F1 的 IF 版本，其中判斷是否大於等於 90、小於 90 即可

```
SELECT `sId`, `cId`, `score` ,
IF(`score` >= 90, '大於等於 90', '小於 90') AS `msg` ,
(SELECT AVG(`score`) FROM `scores`) AS `avg_scores` ,
IF(`score` >= (SELECT AVG(`score`) FROM `scores`), '大於等於平均成績', '小於平均成績'
') AS `avg_msg`
FROM `scores`
```

slid 學生編號	cld 課程編號	score 成績	msg	avg_scores	avg_msg
087	C001	74	小於 90	81.0000	小於平均成績
087	C002	93	大於等於 90	81.0000	大於等於平均成績
088	C002	63	小於 90	81.0000	小於平均成績
088	C004	94	大於等於 90	81.0000	大於等於平均成績

(圖) C3 執行結果。結果可能隨著課堂練習而有不同。

DATE_FORMAT()

以下為 DATE_FORMAT(datetime, format) 的基礎格式列表：

format	description
%Y	西元年，4 位數
%m	月，數值(00-12)
%d	日，數值(00-31)
%H	小時 (00-23)
%i	分鐘，數值(00-59)
%s	秒(00-59)

更多相關格式，請參考：

https://www.w3school.com.cn/sql/func_date_format.asp

顯示查詢框

顯示第 0 - 0 列 (總計 1 筆, 查詢用了 0.0005 秒。)

```
SELECT DATE_FORMAT(NOW(),'%Y-%m-%d')
```

全部顯示 | 資料列數 : 25 ▾ 篩選資料

+ 選項
DATE_FORMAT(NOW(),'%Y-%m-%d')
2020-04-15

(圖) DATE_FORMAT() 的簡單用法

顯示第 0 - 0 列 (總計 1 筆, 查詢用了 0.0013 秒。)

```
1 SELECT `Fname`, `Lname`, `Bdate`, DATE_FORMAT(`Bdate`, '%Y-%m') AS `Byear`, `Sex`, `Salary`
2 FROM `employee`
3 WHERE DATE_FORMAT(`Bdate`, '%Y') = 1937
```

啟用外鍵檢查

執行 取消

全部顯示 | 資料列數 : 25 ▾ 篩選資料列: 搜尋此資料表

+ 選項

	Fname 名	Lname 姓	Bdate 生日	Byear 年	Sex 性別	Salary 薪資
<input type="checkbox"/> 編輯 <input type="checkbox"/> 複製 <input type="checkbox"/> 刪除	James	Borg	1937-11-10	1937-11	M	55000.00

(圖) 可用在 SELECT，也可以用在 WHERE

The screenshot shows a MySQL Workbench interface. At the top, a green bar displays the message: "顯示第 0 - 4 列 (總計 5 筆, 查詢用了 0.0020 秒。) [Bdate: 1972-07-31... - 1962-09-15...]" followed by the SQL query:

```

1 SELECT `employee`.`Fname`, `employee`.`Lname`, `employee`.`Bdate`, DATE_FORMAT(`Bdate`, '%Y-%m') AS `Bdate2`
2 FROM `employee`
3 WHERE DATE_FORMAT(`employee`.`Bdate`, '%Y') > 1960
4 ORDER BY `employee`.`Bdate` DESC
    
```

Below the query, there is a checkbox labeled "啟用外鍵檢查" (Enable foreign key check). Below that are two buttons: "執行" (Execute) and "取消" (Cancel).

The main area shows a results grid with the following columns: Fname (名), Lname (姓), Bdate (生日), and Bdate2. The data rows are:

	Fname	Lname	Bdate	Bdate2
<input type="checkbox"/>	Joyce	English	1972-07-31	1972-07
<input type="checkbox"/>	Ahmad	Jabbar	1969-03-29	1969-03
<input type="checkbox"/>	Alicia	Zelaya	1968-01-19	1968-01
<input type="checkbox"/>	John	Smith	1965-01-09	1965-01
<input type="checkbox"/>	Ramesh	Narayan	1962-09-15	1962-09

(圖) 可以在 WHERE 中限定範例

十、完整性原則與關聯介紹

在規劃資料庫的過程中，需要滿足完整性與資料表間對應的關聯性，才能設計出合適的資料庫結構。

完整性原則

- **參考完整性***

資料表 A 的 Foreign Key (FK) 集合，其值域應該落在資料表 B 的 Primary Key (PK) 當中，否則資料表 A 的 FK 無法參考到資料表 B 的 PK。

- **實體完整性***

資料表 B 的 Primary Key，其值不得為空值 (NULL)，否則無法讓資料表 A 的 Foreign Key 參考到。

- **值域完整性**

若是欄位的結構設定中，指定型態為日期 (例如 datetime)，就不能寫入數值 (例如 int 或 float)。

資料表關聯

- 一對一
- 一對多

- 多對多

十一、觸發器 (Trigger)

資料表之間的同步新增 (insert)、更新 (update)、刪除 (delete)，或是修改一個資料表，同時還要新增、更新、刪除其它表的資料，可以透過觸發器來實現。

參考連結：

[1] MySQL/Trigger

<https://zh.m.wikibooks.org/wiki/MySQL/Trigger>

接下來，我們到 test 資料庫當中，進行測試。

#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊
1	<code>id</code> 🌐	tinyint(3)		UNSIGNED	否	無	流水號	AUTO_INCREMENT
2	<code>username</code> 💬	varchar(10)	utf8mb4_unicode_ci		否	無	使用者名稱	
3	<code>cht_name</code>	varchar(10)	utf8mb4_unicode_ci		否	無	正體中文名稱	

圖：到 test 資料庫新增 users 資料表，結構如上。

#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊
1	<code>id</code> 🌐	tinyint(3)		UNSIGNED	否	無	流水號	AUTO_INCREMENT
2	<code>username</code>	varchar(10)	utf8mb4_unicode_ci		否	無	使用者名稱	
3	<code>updated_at</code>	datetime			否	無	更新時間	

圖：到 test 資料庫中新增 logs 資料表，結構如上。

格式
<pre>DELIMITER / CREATE TRIGGER `trigger 名稱` BEFORE AFTER INSERT UPDATE DELETE ON `事件觸發的來源 table` FOR EACH ROW BEGIN 你的 INSERT/UPDATE/DELETE 的 SQL 語法; END / DELIMITER ;</pre>

說明：

「OLD. 欄位名稱」：修改或刪除前的欄位值

「NEW. 欄位名稱」：修改後或新增的欄位值

TR1：新增名為 insert_logs_01 的觸發器，事件觸發在新增 users 資料表之後，同時新增 logs 資料，其中 logs.username 為 users 資料修改前的 username 欄位值，logs.update_at 為新增當下的格式化後時間欄位(西元年-月-日 時:分:秒)。

```
DELIMITER /
CREATE TRIGGER `insert_logs_01`
AFTER INSERT ON `users`
FOR EACH ROW
BEGIN
    INSERT INTO `logs` (`username`, `updated_at`) VALUES
    (NEW.username, DATE_FORMAT(NOW(), '%Y-%m-%d %H:%i:%s'));
END /
DELIMITER ;
```

TR2：新增名為 insert_logs_02 的觸發器，事件觸發在更新 users 資料表之後，同時新增 logs 資料，其中 logs.username 為 users 資料修改前的 username 欄位值，logs.update_at 為新增當下的格式化後時間欄位(西元年-月-日 時:分:秒)。

```
DELIMITER /
CREATE TRIGGER `insert_logs_02`
AFTER UPDATE ON `users`
FOR EACH ROW
BEGIN
    INSERT INTO `logs` (`username`, `updated_at`) VALUES
    (OLD.username, DATE_FORMAT(NOW(), '%Y-%m-%d %H:%i:%s'));
END /
DELIMITER ;
```

十二、事件 (Event)

若有週期性的觸發事件需要執行，可以透過設定事件排程，在特定的時間或間隔結束後，執行指定的 SQL 指令，例如 24 小時後、每兩小時、幾個月後或每天的特定時間。

參考連結：

[1] MySQL/Event

<https://zh.m.wikibooks.org/zh-tw/MySQL/Event>

開啟 事件排程狀態
SET GLOBAL event_scheduler="ON"



圖：可以在「事件→事件排程狀態」中開啟

格式
<pre>delimiter / CREATE EVENT `event 名稱` ON SCHEDULE 設定時間 DO BEGIN 待執行的 SQL 語法; END / delimiter ;</pre>

可用的時間單位
YEAR, QUARTER, MONTH, WEEK, DAY, HOUR, MINUTE, SECOND, YEAR_MONTH, DAY_HOUR, DAY_MINUTE, DAY_SECOND, HOUR_MINUTE, HOUR_SECOND, MINUTE_SECOND

我們繼續在 test 資料庫當中的 users、logs 資料表，進行測試。

EVT1：每 2 秒在 logs 資料表中，寫入 username 和 updated_at 欄位資料

```

delimiter /
CREATE EVENT `event_every_2_seconds_insert_logs`
ON SCHEDULE EVERY 2 SECOND
DO
BEGIN
    INSERT INTO `logs` (`username`, `updated_at`) VALUES
    ('darren', DATE_FORMAT(NOW(), '%Y-%m-%d %H:%i:%s'));
END /
delimiter ;

```

id 流水號	username 使用者名稱	updated_at 更新時間
1	darren	2021-05-12 14:40:24
2	darren	2021-05-12 14:40:26
3	darren	2021-05-12 14:40:28
4	darren	2021-05-12 14:40:30
5	darren	2021-05-12 14:40:32
6	darren	2021-05-12 14:40:34
7	darren	2021-05-12 14:40:36
8	darren	2021-05-12 14:40:38
9	darren	2021-05-12 14:40:40
10	darren	2021-05-12 14:40:42
11	darren	2021-05-12 14:40:44

圖：每 2 秒自動寫入 username 和 updated_at 欄位

EVT2：1 分鐘後，在 users 資料表中，寫入 username 和 cht_name 欄位資料

```

delimiter /
CREATE EVENT `event_after_1_minute_insert_users`
ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL 1 MINUTE
DO
BEGIN
    INSERT INTO `users` (`username`, `cht_name`) VALUES
    ('darren', '楊oo');
END /

```

```
delimiter ;
```

id 流水號	username 使用者名稱	cht_name 正體中文名稱
1	darren	楊○○

圖：1分鐘後，在 users 資料中進行資料寫入

事件		名稱	狀態	動作	類型
<input type="checkbox"/>	event_after_1_minute_insert_users	ENABLED			ONE TIME
<input type="checkbox"/>	event_every_2_seconds_insert_logs	ENABLED			RECURRING
	<input type="checkbox"/> 全選	已選擇項目：			

圖：共有兩個事件，一個只執行一次，另一個是週期性地執行

事件		名稱	狀態	動作	類型
<input type="checkbox"/>	event_every_2_seconds_insert_logs	ENABLED			RECURRING
	<input type="checkbox"/> 全選	已選擇項目：			

圖：當「一次性事件」執行完畢後，該事件會自動被刪除

十三、子查詢 (Sub Query)

進行多資料表查詢，除了使用 join 外，也可以使用 SQL 子查詢(sub query)。子查詢就是在一個 select 指令內再放入一個 select 查詢指令進行查詢，通常是位在 select 的 where 子句，可以透過子查詢取得查詢條件；子查詢的效能較差，除非資料不大，否則不建議頻繁使用。

格式

```
SELECT 欄位名稱 1, 欄位名稱 2, ..., 欄位名稱 n
FROM 資料表名稱 1
WHERE 欄位名稱 IN (或是 =) (
    SELECT 欄位名稱
```

```
FROM 資料表名稱 2
WHERE 條件
)
```

S1：查詢員工資料，其所屬部門代號在部門資料表當中的編號，不是 4 和 5。

```
SELECT *
FROM `employee`
WHERE `Dno` IN (
    SELECT `Dnumber`
    FROM `department`
    WHERE `Dnumber` NOT IN (4, 5)
)
```

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

(圖) S1 執行結果

十四、檢視表 (View)

倘若我們經常使用複雜的查詢語句，可能會增加程式維護的成本。因此，我們可以將查詢語句建立成一個虛擬的資料表，僅需使用基本的查詢語法，便可達到如同執行複雜語句的效果。檢視表有以下的特性：

- 加強資料庫的安全性，View 可以將實體資料表結構隱藏起來，同時限制使用者只可以檢視及使用哪些資料表欄位。
- 檢視表是唯讀的，亦即外部使用者無法直接透過 View 去修改內部資料。
- 將複雜的 SQL 查詢包裝在 View 中，可以簡化查詢的複雜度。
- 當資料表結構有變更時，只需要更改 View 的設定，不需更改程式。

在資料表 my_db.scores 執行 SQL 查詢：

```
1 SELECT `sId`, `cId`, `score`,
2 IF(`score` >= 90, '大於等於 90', '小於 90') AS `msg`,
3 (SELECT AVG(`score`) FROM `scores`) AS `avg_scores`,
4 IF(`score` >= (SELECT AVG(`score`) FROM `scores`), '大於等於平均成績', '小於平均成績') AS `avg_msg`
5 FROM `scores`
```

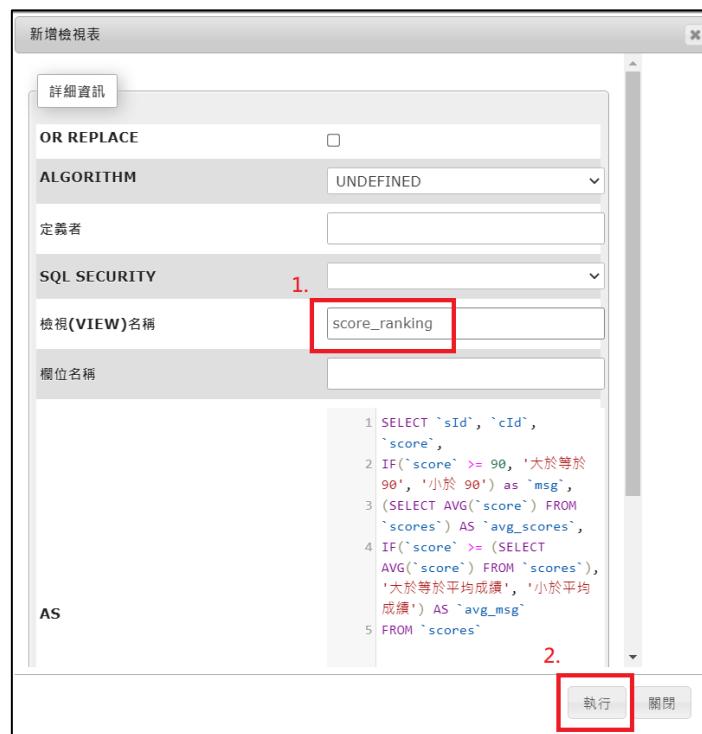
圖：以 C3 為例，執行 SQL 查詢

sId 學生編號	cId 課程編號	score 成績	msg	avg_scores	avg_msg
087	C001	74	小於 90	81.2000	小於平均成績
087	C002	93	大於等於 90	81.2000	大於等於平均成績
088	C002	63	小於 90	81.2000	小於平均成績
088	C003	82	小於 90	81.2000	大於等於平均成績
088	C004	94	大於等於 90	81.2000	大於等於平均成績

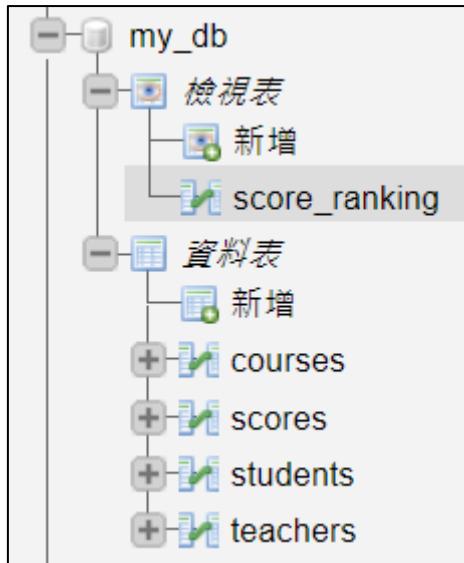
全部顯示 | 資料列數： 25 篩選資料列： 搜尋此資料表

查詢結果選項

圖：顯示查詢結果後，按下「新增檢視表」



圖：輸入 view 的名稱，按下執行



圖：多了「檢視表」→「score_ranking」

SQL 語法

```

CREATE VIEW `score_ranking` AS SELECT `sId`, `cId`, `score`,
IF(`score` >= 90, '大於等於 90', '小於 90') as `msg`,
(SELECT AVG(`score`) FROM `scores`) AS `avg_scores`,
IF(`score` >= (SELECT AVG(`score`) FROM `scores`), '大於等於平均成績', '小於平均成績'
) AS `avg_msg`
FROM `scores`
  
```

	sId 學生編號	cId 課程編號	score 成績	msg 訊息	avg_scores 平均成績	avg_msg 訊息
087	C001	74	小於 90	81.2000	小於平均成績	
087	C002	93	大於等於 90	81.2000	大於等於平均成績	
088	C002	63	小於 90	81.2000	小於平均成績	
088	C003	82	小於 90	81.2000	大於等於平均成績	
088	C004	94	大於等於 90	81.2000	大於等於平均成績	

圖：查詢檢視表，等同於執行先前儲存的複雜查詢語句

V1: 將檢視表所有資料，依成績由大到小進行排序

```

SELECT *
FROM `score_ranking`
ORDER BY `score` DESC
  
```

sid 學生編號	cld 課程編號	score 成績	msg	avg_scores	avg_msg
088	C004	94	大於等於 90	81.2000	大於等於平均成績
087	C002	93	大於等於 90	81.2000	大於等於平均成績
088	C003	82	小於 90	81.2000	大於等於平均成績
087	C001	74	小於 90	81.2000	小於平均成績
088	C002	63	小於 90	81.2000	小於平均成績

圖: V1 執行結果

圖: 編輯檢視表

1.

```

1 select `my_db`.`scores`.* AS
`sId`, `my_db`.`scores`.* AS
`cId`, `my_db`.`scores`.* AS
`score`, if(`my_db`.`scores`.* >= 90, '大於
等於 90', '小於 90') AS `msg`, (select
avg(`my_db`.`scores`.* score) from
`my_db`.`scores`) AS
`avg_scores`, if(`my_db`.`scores`.* >=
(select avg(`my_db`.`scores`.* score) from
`my_db`.`scores`), '大於等於平均成績', '小於平均成績') AS `avg_msg` from `my_db`.`scores`

```

2.

圖: 按下執行，儲存檢視表預先定義的 SQL 語句

十五、實務操作（一）

資料來源：

[1] OpenData-全國宗教資訊系統資料-寺廟

<https://data.moi.gov.tw/MoiOD/Data/DataDetail.aspx?oid=1B56C087-43D9-4B4B-B08C-D7B6A750E033>

請匯入 osm_temple.sql 到 my_db 資料庫當中。

#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊
1	id	int(11)		否	無		流水號	AUTO_INCREMENT
2	temple_name	varchar(100)	utf8mb4_unicode_ci	是	NULL		寺廟名稱	
3	god_name	varchar(100)	utf8mb4_unicode_ci	是	NULL		主祀神祇	
4	district	varchar(10)	utf8mb4_unicode_ci	是	NULL		行政區	
5	address	varchar(100)	utf8mb4_unicode_ci	是	NULL		地址	
6	religion	varchar(20)	utf8mb4_unicode_ci	是	NULL		教別	
7	license	varchar(20)	utf8mb4_unicode_ci	是	NULL		建別	
8	manage_type	varchar(20)	utf8mb4_unicode_ci	是	NULL		組織型態	
9	phone_number	varchar(20)	utf8mb4_unicode_ci	是	NULL		電話	
10	owner	varchar(15)	utf8mb4_unicode_ci	是	NULL		負責人	
11	other	varchar(100)	utf8mb4_unicode_ci	是	NULL		其它	
12	wgs84x	double		是	NULL		經度	
13	wgs84y	double		是	NULL		緯度	
14	created_at	datetime		否	current_timestamp()		新增時間	
15	update_at	datetime		否	current_timestamp()	ON UPDATE CURRENT_TIMESTAMP()	更新時間	

圖：寺廟資料表基本結構

id	寺廟名稱	主祀神祇	行政區	地址	宗教	建別	組織型態	電話	負責人	其它	wgs84x	wgs84y	created_at	更新時間
1	竹園福德祠	福德正神	臺南市白河區大竹里14鄰大排1206號		道教	適用監督 寺廟條例	管理人(住持)制	06-6851562	鍾玉珠	NULL	120.396797180176	23.3648853302002	2020-09-02 01:07:57	2020-09-02 01:07:57
2	福德祠	福德正神	臺南市河底里33之9號		道教	適用監督 寺廟條例	管理人(住持)制	06-6852378	吳朝正	NULL	120.438499450684	23.3971004486084	2020-09-02 01:07:57	2020-09-02 01:07:57
3	紫雲觀	觀世音菩薩	臺中市里中里中正路140號		道教	適用監督 寺廟條例	管理委員會制	04-25566801	侯錦朝	NULL	120.70298278809	24.3052196502686	2020-09-02 01:07:57	2020-09-02 01:07:57
4	永安宮	石府千歲	臺南市豆園里3斯加里29號		道教	適用監督 寺廟條例	其他	06-5719598	郭芳財	NULL	120.238899230957	23.2116107940674	2020-09-02 01:07:57	2020-09-02 01:07:57
5	福德廟	福德正神	宜蘭縣五結鄉平林安平村安平路340號		道教	適用監督 寺廟條例	管理人(住持)制	03-9593889	李汪文	NULL	121.788185119629	24.6245384216309	2020-09-02 01:07:57	2020-09-02 01:07:57
6	蘭陽南海宮	九天娘娘	宜蘭縣頭城鎮中山二路223號		道教	適用監督 寺廟條例	管理委員會制	03-9587103	何豈和	NULL	121.74438171387	24.6360340118408	2020-09-02 01:07:57	2020-09-02 01:07:57
7	蘭陽太和宮	福德正神	宜蘭縣頭城鎮八寶村八寶路172號		道教	適用監督 寺廟條例	管理委員會制	03-9581599	蔣朝生	NULL	121.763397216797	24.6380100250244	2020-09-02 01:07:57	2020-09-02 01:07:57
			臺中市新											

圖：瀏覽內容

T1 取得主祀神祇為「福德正神」的所有資料

```
SELECT *
FROM `temples`
WHERE `god_name` = '福德正神'
```

<u>id</u> 流水號	<u>temple_name</u> 寺廟名稱	<u>god_name</u> 主祀神祇	<u>district</u> 行政區	<u>address</u> 地址	<u>religion</u> 教別	<u>license</u> 證別	<u>manage_type</u> 經營型態	<u>phone_number</u> 電話	<u>owner</u> 負責人	<u>other</u> 其它	<u>wgs84x</u> 經度	<u>wgs84y</u> 緯度	<u>created_at</u> 新增時間	<u>update_at</u> 更新時間
1	竹園仔福德祠	福德正神	臺南市	臺南市白河區大竹里14鄰大排竹206號	道教	適用監督 寺廟條例	管理人(住持)制	06-6851562	鍾玉珠	NULL	120.396797180176	23.3648853302002	2020-09-02 01:07:57	2020-09-02 01:07:57
2	福德祠	福德正神	臺南市	臺南市白河區寶興里33之2號	道教	適用監督 寺廟條例	管理人(住持)制	06-6852378	吳朝正	NULL	120.438499450684	23.3971004486084	2020-09-02 01:07:57	2020-09-02 01:07:57
5	大埤福德廟	福德正神	宜蘭縣	宜蘭縣冬山鄉安平村安平路346號對面	道教	適用監督 寺廟條例	管理人(住持)制	03-9593889	李汪文	NULL	121.788185119629	24.6245384216309	2020-09-02 01:07:57	2020-09-02 01:07:57
7	華陽太和宮	福德正神	宜蘭縣	宜蘭縣冬山鄉八寶村八寶路172號	道教	適用監督 寺廟條例	管理委員會制	03-9581599	羅朝生	NULL	121.763397216797	24.6380100250244	2020-09-02 01:07:57	2020-09-02 01:07:57
8	崙山福德祠	福德正神	臺中市	臺中市新社區崙山里崙南街82-2號	道教	適用監督 寺廟條例	管理人(住持)制	04-25811482	陳光傳	NULL	120.785400390625	24.2429695129395	2020-09-02 01:07:57	2020-09-02 01:07:57
22	珍珠福德廟	福德正神	宜蘭縣	宜蘭縣冬山鄉頭安村農安路347號	道教	適用監督 寺廟條例	管理委員會制	03-9680382	林燈添	NULL	121.764846801758	24.6608009338379	2020-09-02 01:07:57	2020-09-02 01:07:57
23	福德宮	福德正神	雲林縣	雲林縣虎尾鎮立仁里復興路128號 雲林縣虎	道教	適用監督 寺廟條例	管理委員會制	05-6324977	黃昭良	NULL	120.427696228027	23.7073879241943	2020-09-02 01:07:57	2020-09-02 01:07:57

圖:T1 執行結果

T2 取得臺北市裡，主祀神祇為福德正神的寺廟名稱、教別、負責人及電話

```
SELECT `temple_name`, `religion`, `owner`, `phone_number`
FROM `temples`
WHERE `district` = '臺北市'
AND `god_name` = '福德正神'
```

temple_name 寺廟名稱	religion 教別	owner 負責人	phone_number 電話
大橋頭福德廟	道教	劉丁柱	02-25529255
台北神德宮	道教	李隆德	02-25032358
大龍峒福壽宮	道教	陳源陽	02-25910340
北門口福聚宮	道教	陳正賢	02-25550051
福正宮	道教	陳健智	02-25925144
植福宮	道教	陳志堂	02-25323316
台北市松山五分埔福德宮	道教	陳永昌	02-27692247
台北市文山區頂公館福德宮	道教	林鴻昭	02-29325511
福德祠	道教	陳東源	02-27900624
士林街福德宮	道教	陳中和	02-28819326
福興宮	道教	許新榮	02-29311496
新福宮	道教	洪煌龍	02-25631982
長慶廟	道教	何二郎	02-23653426
福慈宮	道教	陳國晏	02-28916058
和德祠	道教	李林世津	02-25550172
新坡尾福德廟	道教	李明和	09-32012098
棍頭福德祠	道教	陳尤雪	02-27928299
承德福德宮	道教	陳福隆	02-25322521
台北市文山區溝子口福德宮	道教	翁傳煌	02-22360769
台北文山興安宮	道教	賴榮富	02-86632817
台北市大安區福安宮	道教	陳黃綉貝	02-27320830
財團法人陽明山五福宮	道教	何正華	02-28725833
大安車層景福宮	道教	詹正宗	02-27528693、27311988

圖: T2 執行結果

T3 取得寺廟名稱當中，只要有「龍山寺」三個字，就列出寺廟名稱與行政區

```
SELECT `temple_name`, `district`
FROM `temples`
WHERE `temple_name` LIKE '%龍山寺%'
```

temple_name 寺廟名稱	district 行政區
龍山寺	高雄市
龍山寺	臺南市
龍山寺	嘉義縣
二重溪龍山寺	臺南市
龍山寺	彰化縣
清龍山寺	高雄市
龍山寺	新北市
龍山寺	臺南市
龍山寺	臺南市
龍山寺	澎湖縣
巒山富龍山寺	高雄市
龍山寺	屏東縣
財團法人桃園市大溪區龍山寺	桃園市
龍山寺	高雄市
內門龍山寺	高雄市
財團法人台北市艋舺龍山寺	臺北市

圖:T3 執行結果

T4 取得以行政區、宗教兩個欄位進行分組的寺廟數量 (別名自訂)

```
SELECT `district`, `religion`, COUNT(`id`) AS `temple_count`
FROM `temples`
GROUP BY `district`, `religion`
```

district 行政區	religion 教別	temple_count
南投縣	NULL	1
南投縣	一貫道	6
南投縣	佛教	142
南投縣	其他	1
南投縣	其他(儒教)	2
南投縣	彌勒大道	1
南投縣	道教	335
嘉義市	一貫道	4
嘉義市	佛教	20
嘉義市	道教	127
嘉義縣	一貫道	9
嘉義縣	佛教	104
嘉義縣	天道	1
嘉義縣	道教	603
基隆市	一貫道	3
基隆市	佛教	52
基隆市	軒轅教	1
基隆市	道教	174
宜蘭縣	一貫道	5
宜蘭縣	佛教	90
宜蘭縣	其他(儒教)	1
宜蘭縣	軒轅教	1
宜蘭縣	道教	559
屏東縣	NULL	1
屏東縣	一貫道	6

圖:T4 執行結果

從 T4 執行結果可以得知，資料裡面有些缺失的資訊(例如 NULL 空值)，若是我們需要省略缺失的資訊，可以加上「`WHERE `religion` IS NOT NULL`」來過濾掉缺失的資訊。

T4-1 取得以行政區、宗教兩個欄位進行分組的寺廟數量 (別名自訂)，省略缺失資訊

```
SELECT `district`, `religion`, COUNT(`id`) AS `temple_count`
FROM `temples`
WHERE `religion` IS NOT NULL
GROUP BY `district`, `religion`
```

district	religion	temple_count
行政區	教別	
南投縣	一貫道	6
南投縣	佛教	142
南投縣	其他	1
南投縣	其他(儒教)	2
南投縣	彌勒大道	1
南投縣	道教	335
嘉義市	一貫道	4
嘉義市	佛教	20
嘉義市	道教	127
嘉義縣	一貫道	9
嘉義縣	佛教	104
嘉義縣	天道	1
嘉義縣	道教	603
基隆市	一貫道	3
基隆市	佛教	52
基隆市	軒轅教	1
基隆市	道教	174
宜蘭縣	一貫道	5
宜蘭縣	佛教	90
宜蘭縣	其他(儒教)	1
宜蘭縣	軒轅教	1
宜蘭縣	道教	559
屏東縣	一貫道	6
屏東縣	佛教	148
屏東縣	其他(儒教)	1

圖: T4-1 執行結果

T5 取得以行政區、宗教兩個欄位進行分組的寺廟數量 (別名自訂)，省略缺失資訊，並依寺廟數量進行由大到小的排序結果。

```
SELECT `district`, `religion`, COUNT(`id`) AS `temple_count`
FROM `temples`
WHERE `religion` IS NOT NULL
GROUP BY `district`, `religion`
ORDER BY `temple_count` DESC
```

district	religion	temple_count
行政區	教別	
臺南市	道教	1382
高雄市	道教	1114
屏東縣	道教	961
臺中市	道教	752
彰化縣	道教	718
新北市	道教	700
雲林縣	道教	658
嘉義縣	道教	603
宜蘭縣	道教	559
南投縣	道教	335
高雄市	佛教	323
苗栗縣	道教	268
臺南市	佛教	238
新北市	佛教	230
桃園市	道教	191
金門縣	道教	181
臺中市	佛教	180
基隆市	道教	174
臺北市	道教	166
臺東縣	道教	164
澎湖縣	道教	149
屏東縣	佛教	148
南投縣	佛教	142
新竹縣	道教	135
嘉義市	道教	127

圖: T5 執行結果

T6 所有宗教各自的數量，同時省略缺失資訊

```
SELECT `religion` , COUNT(`id`) AS `temple_count`
FROM `temples`
WHERE `religion` IS NOT NULL
GROUP BY `religion`
```

religion	temple_count
教別	
一貫道	229
三一(夏)教	1
佛教	2264
其他	3
其他(佛教)	1
其他(儒教)	23
其他(唯心聖教)	1
其他(天主教)	1
其他(藏傳佛教寧瑪巴)	1
其他(道)	1
其他(道教)	2
其他(顯教)	1
其他(黃中)	1
天帝教	2
天德聖教	5
天道	2
彌勒大道	6
理教	6
軒轅教	6
道教	9637

圖: T6 執行結果

T6-1 所有宗教各自的數量，同時省略缺失資訊，並依數量由小到大排序

```
SELECT `religion`, COUNT(`id`) AS `temple_count`
FROM `temples`
WHERE `religion` IS NOT NULL
GROUP BY `religion`
ORDER BY `temple_count` ASC
```

religion	temple_count
教別	
其他(藏傳佛教寧瑪巴)	1
其他(佛教)	1
其他(道)	1
三一(夏)教	1
其他(天主教)	1
其他(黃中)	1
其他(唯心聖教)	1
其他(顯教)	1
其他(道教)	2
天帝教	2
天道	2
其他	3
天德聖教	5
彌勒大道	6
理教	6
軒轅教	6
其他(儒教)	23
一貫道	229
佛教	2264
道教	9637

圖: T6-1 執行結果

十六、實務操作（二）

LINE 貼圖資訊擷取與儲存

實務操作所需要安裝的套件指令

```
pip install selenium beautifulsoup4 requests PyMySQL Flask
```

參考頁面：

[1] PyMySQL Examples

<https://pymysql.readthedocs.io/en/latest/user/examples.html>

[2] Python+MySQL 資料庫操作（PyMySQL）

<https://www.tw511.com/3/39/1388.html>

[3] Python 資料庫學習筆記(四)：使用 PyMySQL 模組

<https://reurl.cc/Q78eD2>

[4] [Day 08] Beautiful Soup 解析 HTML 元素

<https://ithelp.ithome.com.tw/articles/10204390>

新增 LINE 貼圖使用的資料表（SQL 語法，請新增在 my_db 資料庫）

```
CREATE TABLE `line-stickers` (
  `id` varchar(20) COLLATE utf8mb4_unicode_ci NOT NULL COMMENT '貼圖編號',
  `ext` varchar(5) COLLATE utf8mb4_unicode_ci NOT NULL COMMENT '副檔名',
  `category` varchar(15) COLLATE utf8mb4_unicode_ci NOT NULL COMMENT '貼圖類別',
  `link` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL COMMENT '貼圖連結',
  `created_at` datetime NOT NULL DEFAULT current_timestamp() COMMENT '新增時間',
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE current_timestamp()
  COMMENT '更新時間'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci COMMENT='LINE 貼圖資料表';

ALTER TABLE `line-stickers` ADD PRIMARY KEY (`id`);
```

PyMySQL 的操作範例

```
import pymysql

# 資料庫連線
connection = pymysql.connect(
    host='localhost',
    user='root',
    password=''
```

```
database='my_db',
charset='utf8mb4',
cursorclass=pymysql.cursors.DictCursor
)

# 取得 cursor 物件，進行 CRUD
cursor = connection.cursor()

try:
    # 寫入資料
    # sql = "INSERT INTO `users` (`email`, `password`) VALUES (%s, %s)"
    # cursor.execute(sql, ('webmaster@python.org', 'very-secret'))

    # 查詢資料
    sql = "SELECT * FROM `categories`"
    cursor.execute(sql)

    # 查詢結果列數大於 0 ，代表有資料
    if cursor.rowcount > 0:
        # 將查詢結果轉成 list 型態（裡頭元素都是 dict）
        results = cursor.fetchall()
        # 迭代取得資料（dict 型態）
        for result in results:
            print(result)
    else:
        print("rowcount: 0")

    # 提交 SQL 執行結果
    connection.commit()
except Exception as e:
    # 回滾
    connection.rollback()
    print("SQL 執行失敗")
    print(e)

# 釋放 cursor
cursor.close()
```

```
# 關閉資料庫連線  
connection.close()
```

BeautifulSoup 基本用法

```
soup.select() :  
回傳的結果是元素集合 (list 型態, BeautifulSoup ResultSet)  
  
soup.select_one() :  
回傳的結果是單一元素 (BeautifulSoup Result)
```

line-stickers-bs4.py

```
import requests, pprint, os, json, pprint  
from bs4 import BeautifulSoup  
import pymysql  
  
# 放貼圖資訊用  
listLineStickers = []  
  
# 自訂標頭  
my_headers = {  
    'user-  
agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36  
(KHTML, like Gecko) Chrome/86.0.4240.75 Safari/537.36'  
}  
  
# 官方 LINE 貼圖的網址  
url = 'https://store.line.me/stickershop/product/17555/zh-Hant'  
  
# 將自訂標頭加入 GET 請求中  
response = requests.get(url, headers = my_headers)  
  
# 建立 soup 物件  
soup = BeautifulSoup(response.text, 'lxml')  
  
# 取得放置貼圖的 li 元素 (list 型態)  
li_elements = soup.select("ul.mdCMN09Ul.FnStickerList > li.mdCMN09Li.  
FnStickerPreviewItem")
```

```
# 逐一取得 li 元素中的 data-preview 資訊
for li in li_elements:
    # 取得 data-preview 屬性的值(字串)
    strJson = li['data-preview'] # 另一種寫法：li.get("data-preview")

    #把屬性的值(字串)轉成物件
    obj = json.loads(strJson)

    # 將重要資訊放置在 list 當中，幫助我們稍候進行資料下載與儲存
    listLineStickers.append({
        "id": obj['id'],
        "link": obj['staticUrl']
    })

# 資料庫連線
connection = pymysql.connect(
    host='localhost',
    user='root',
    password='',
    database='my_db',
    charset='utf8mb4',
    cursorclass=pymysql.cursors.DictCursor
)

# 取得 cursor 物件，進行 CRUD
cursor = connection.cursor()

try:
    # 建立儲存圖片的資料夾，不存在就新增
    folderPath = 'line_stickers'
    if not os.path.exists(folderPath):
        os.makedirs(folderPath)

    # 定義 SQL 語法
    sql = "INSERT INTO `line-
stickers` (`id`, `ext`, `category`, `link`) VALUES (%s, %s, %s, %s)"

    # 迭代走訪 list 當中的 dict
```

```
for obj in listLineStickers:  
    # 執行 SQL 語法  
    cursor.execute( sql, (obj['id'], 'png', 'stickershop', obj['link']) )  
  
    # 下載  
    os.system(f"curl {obj['link']} -o {FolderPath}/{obj['id']}.png")  
    print(f"貼圖 ID: {obj['id']}, 下載連結: {obj['link']}")  
  
    # 提交 SQL 執行結果  
    connection.commit()  
  
except Exception as e:  
    # 回滾  
    connection.rollback()  
    print("SQL 執行失敗")  
    print(e)  
  
    # 關閉 cursor  
cursor.close()  
  
# 關閉 資料庫連線  
connection.close()
```

十七、實務操作（三）

YouTube 搜尋結果擷取與儲存

資料庫操作方式，請參考「十六、實務操作（二）」

在往下閱讀前：

1. 請先下載 ChromeDriver

<https://chromedriver.chromium.org/>

2. 請確認目前你電腦裡面的 chrome 瀏覽器版本

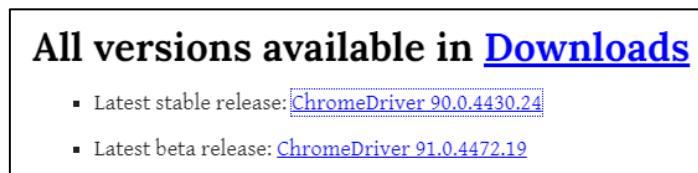


圖：按下瀏覽器右上方的「⋮」→說明→關於 Google Chrome

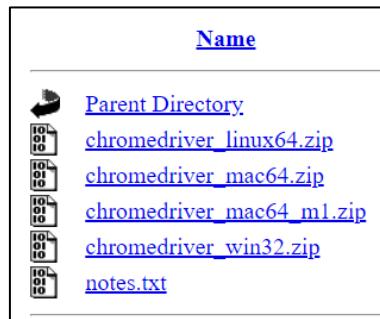


圖：請與下載的 ChromeDriver 版本一致

3. 下載 ChromeDriver 檔案，並放到專案資料夾當中

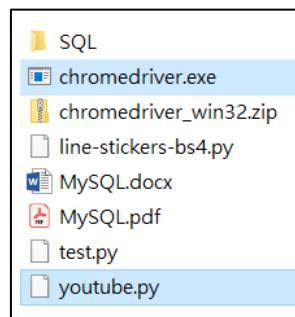


圖：下載合適的 Chrome 版本



圖：Windows 選擇 win32 版本；MacOS 則需要選擇對應的版本

4. 放置 ChromeDriver，到 youtube.py 檔案相同的路徑下



圖：下載到專案資料夾裡面，並解壓縮

新增 YouTube 資訊所使用的資料表（SQL 語法，請新增在 my_db 資料庫）

```

CREATE TABLE `youtube` (
  `id` int(11) NOT NULL COMMENT '影音編號',
  `youtube_id` char(11) COLLATE utf8mb4_unicode_ci NOT NULL COMMENT 'YouTube ID',
  `ext` varchar(5) COLLATE utf8mb4_unicode_ci DEFAULT NULL COMMENT '副檔名',
  `keyword` varchar(15) COLLATE utf8mb4_unicode_ci NOT NULL COMMENT '搜尋關鍵字',
  `title` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL COMMENT '影音標題',
  `link` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL COMMENT '影音連結',
  `img` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL COMMENT '影音封面照片路徑',
  `created_at` datetime NOT NULL DEFAULT current_timestamp() COMMENT '新增時間',
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE current_timestamp()
  COMMENT '更新時間'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci COMMENT='YouTube 資料表';

ALTER TABLE `youtube` ADD PRIMARY KEY (`id`);

ALTER TABLE `youtube` MODIFY `id` int(11) NOT NULL AUTO_INCREMENT COMMENT '影音編號';

```

方法	說明
find_element(by, value)	使用 by 指定之方法取得第一個符合 value 的元素
find_element_by_class_name(name)	傳回符合指定 class 名稱之元素
find_elements_by_class_name(name)	傳回符合指定 class 名稱之元素串列
find_element_by_css_selector(selector)	傳回符合指定 CSS 選擇器名稱之元素
find_elements_by_css_selector(selector)	傳回符合指定 CSS 選擇器名稱之元素串列
find_element_by_id(id)	傳回符合指定 id 之元素
find_elements_by_id(id)	傳回符合指定 id 之元素串列
find_element_by_link_text(text)	傳回符合指定超連結文字之元素
find_elements_by_link_text(text)	傳回符合指定超連結文字之元素串列
find_element_by_partial_link_text(text)	傳回符合部分指定超連結文字之元素
find_elements_by_partial_link_text(text)	傳回符合部分指定超連結文字之元素串列
find_element_by_name(name)	傳回符合指定元素名稱之元素
find_elements_by_name(name)	傳回符合指定元素名稱之元素串列
find_element_by_tag_name(tag)	傳回符合指定標籤名稱之元素
find_elements_by_tag_name(tag)	傳回符合指定標籤名稱之元素串列

圖：列出 driver 取得元素的方法

youtube.py

```
'''
```

匯入套件

```
'''  
# 操作 browser 的 API  
from selenium import webdriver  
  
# 處理逾時例外的工具  
from selenium.common.exceptions import TimeoutException  
  
# 面對動態網頁，等待某個元素出現的工具，通常與 expected_conditions 搭配  
from selenium.webdriver.support.ui import WebDriverWait  
  
# 搭配 WebDriverWait 使用，對元素狀態的一種期待條件，若條件發生，則等待結束，往下一行執行  
from selenium.webdriver.support import expected_conditions as EC  
  
# 期待元素出現要透過什麼方式指定，通常與 EC、WebDriverWait 一起使用  
from selenium.webdriver.common.by import By  
  
# 強制等待（執行期間休息一下）  
from time import sleep  
  
# 整理 json 使用的工具  
import json  
  
# 執行 command 的時候用的  
import os  
  
# 資料庫 CRUD 工具  
import pymysql  
  
'''  
Selenium with Python 中文翻譯文檔  
參考網頁：https://selenium-python-zh.readthedocs.io/en/latest/index.html  
selenium 啓動 Chrome 的進階配置參數  
參考網址：https://stackoverflow.max-everyday.com/2019/12/selenium-chrome-options/  
Mouse Hover Action in Selenium
```

```
參考網址：https://www.toolsqa.com/selenium-webdriver/mouse-hover-action/
...
# 啟動瀏覽器工具的選項
options = webdriver.ChromeOptions()
# options.add_argument("--headless") #不開啟實體瀏覽器背景執行
options.add_argument("--start-maximized") #最大化視窗
options.add_argument("--incognito") #開啟無痕模式
options.add_argument("--disable-popup-blocking") #禁用彈出攔截

# 使用 Chrome 的 WebDriver (含 options)
driver = webdriver.Chrome( options = options )

# driver.set_window_size(1200, 960) #視窗大小設定（寬，高）
# driver.maximize_window() #視窗最大化
# driver.minimize_window() #視窗最小化

# 資料庫連線
connection = pymysql.connect(
    host = 'localhost',
    user = 'root',
    password = '',
    database = 'my_db',
    charset = 'utf8mb4',
    cursorclass = pymysql.cursors.DictCursor
)

# 取得 cursor 物件，進行 CRUD
cursor = connection.cursor()

# 放置爬取的資料
listData = []

# 準備在 YouTube 搜尋的關鍵字
keyword = "蕭敬騰"

...
```

```
以 function 名稱，作為爬蟲流程
...
# 走訪頁面
def visit():
    driver.get('https://www.youtube.com/');

# 輸入關鍵字
def search():
    # 輸入名稱
    txtInput = driver.find_element(By.CSS_SELECTOR, "input#search")
    txtInput.send_keys(keyword)

    #按下送出
    btnInput = driver.find_element(By.CSS_SELECTOR, "button#search-icon-legacy")
    btnInput.click()

    # 等待一下
    sleep(2)

# 篩選 (選項)
def filterFunc():
    try:
        # 等待篩選元素出現
        WebDriverWait(driver, 10).until(
            EC.presence_of_element_located(
                (
                    By.CSS_SELECTOR,
                    "yt-formatted-string#text.style-scope.ytd-toggle-
button-renderer.style-text"
                )
            )
        )

        #按下篩選元素，使項目浮現
        driver.find_element(
            By.CSS_SELECTOR,
```

```
"yt-formatted-string#text.style-scope.ytd-toggle-button-
renderer.style-text"
).click()

# 等待一下
sleep(2)

# 按下選擇的項目
driver.find_elements(
    By.CSS_SELECTOR,
    "yt-formatted-string.style-scope.ytd-search-filter-
renderer"
)[10].click()

# 等待一下
sleep(2)

except TimeoutException:
    print("等待逾時，即將關閉瀏覽器...")
    sleep(3)
    driver.quit()

# 滾動頁面
def scroll():
    # 瀏覽器內部的高度
    innerHeightOfWindow = 0

    # 當前捲動的量(高度)
    totalOffset = 0

    # 在捲動到沒有元素動態產生前，持續捲動
    while totalOffset <= innerHeightOfWindow:
        # 每次移動高度
        totalOffset += 500

        # 撥動的 js code
        js_scroll = '''(
            function (){{
                window.scrollBy(0, 500);
            }}()
        )'''

        # 執行 js code
        driver.execute_script(js_scroll)
```

```
window.scrollTo({  
    top:{}  
    behavior: 'smooth'  
});  
}());''.format(totalOffset)  
  
# 執行 js code  
driver.execute_script(js_scroll)  
  
# 透過執行 js 語法來取得捲動後的當前總高度  
innerHeightOfWindow = driver.execute_script(  
    'return window.document.documentElement.scrollHeight;'  
)  
  
# 印出捲動距離  
print("innerHeightOfWindow: {}, totalOffset: {}".format(inner  
HeightOfWindow, totalOffset))  
  
# 強制等待  
sleep(2)  
  
# 為了實驗功能，捲動超過一定的距離，就結束程式  
# if totalOffset >= 1800:  
#     break  
  
# 分析頁面元素資訊  
def parse():  
    # 取得主要元素的集合  
    ytd_video_renderers = driver.find_elements(  
        By.CSS_SELECTOR,  
        'ytd-video-renderer.style-scope.ytd-item-section-renderer')  
  
    # 逐一檢視元素  
    for element in ytd_video_renderers:  
        # 印出分隔文字  
        print("=" * 30)  
  
        # 取得圖片連結
```

```
img = element.find_element(
    By.CSS_SELECTOR,
    "ytd-thumbnail.style-scope.ytd-video-renderer img#img")
imgSrc = img.get_attribute('src')
print(imgSrc)

# 取得資料名稱
a = element.find_element(By.CSS_SELECTOR, "a#video-title")
aTitle = a.text
print(aTitle)

# 取得 YouTube 連結
aLink = a.get_attribute('href')
print(aLink)

# 取得 影音 ID
youtube_id = aLink.split("v=")[1]

# 放資料到 list 中
listData.append({
    "youtube_id": youtube_id,
    "title": aTitle,
    "link": aLink,
    "img": imgSrc
})

# 將 list 存成 json
def saveJson():
    fp = open("youtube.json", "w", encoding='utf-8')
    fp.write( json.dumps(listData, ensure_ascii=False) )
    fp.close()

# 將取得資料寫入資料庫
def saveDB():
    try:
        # 開啟 json 檔案
        fp = open("youtube.json", "r", encoding='utf-8')
```

```
#取得 json 字串
strJson = fp.read()

# 關閉檔案
fp.close()

# 將 json 轉成 list (裡面是 dict 集合)
listResult = json.loads(strJson)

# 決定資料寫入的語法
sql = "INSERT INTO `youtube` (`youtube_id`, `keyword`, `title` , `link` , `img`) VALUES (%s, %s, %s, %s, %s)"

# ??????
for index, obj in enumerate(listResult):
    cursor.execute(
        sql,
        (
            obj['youtube_id'], keyword, obj['title'], obj['link'],
            obj['img']
        )
    )

# 提交 SQL 執行結果
connection.commit()

except Exception as e:
    # 回滾
    connection.rollback()
    print("SQL 執行失敗")
    print(e)

# 關閉瀏覽器、資料庫等資源
def close():
    driver.quit()
    cursor.close()
    connection.close()

# 主程式
```

```
if __name__ == '__main__':
    visit()
    search()
    filterFunc()
    scroll()
    parse()
    saveJson()
    saveDB()
    close()
```

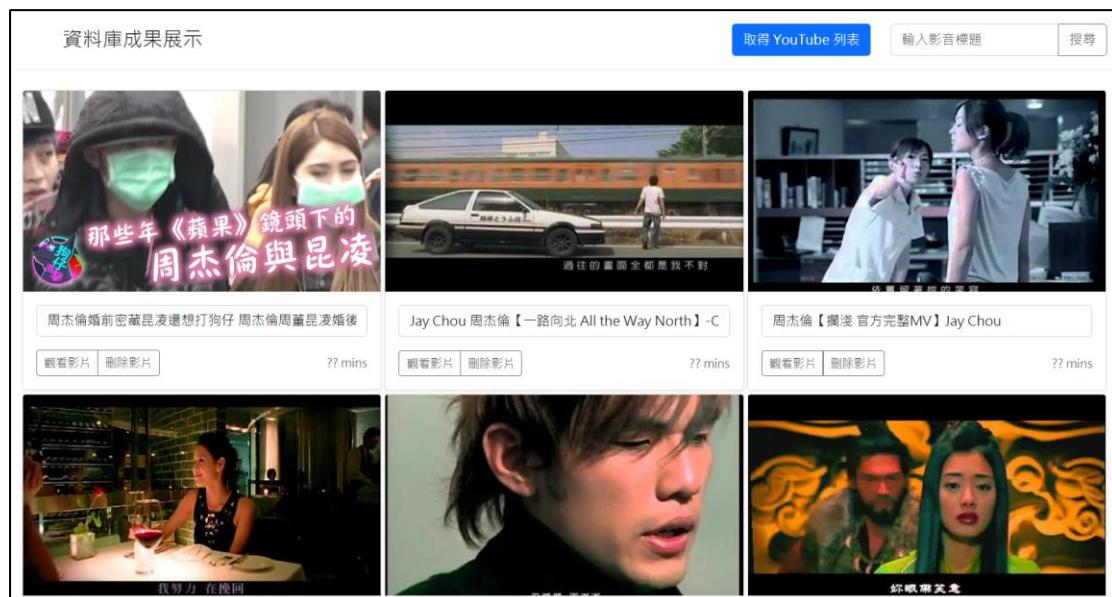
可能用到下載檔案的程式碼

```
os.system("youtube-dl.exe -f mp4 -i {} -o {}".format(obj["aLink"], "%(id)s.(ext)s"))
```

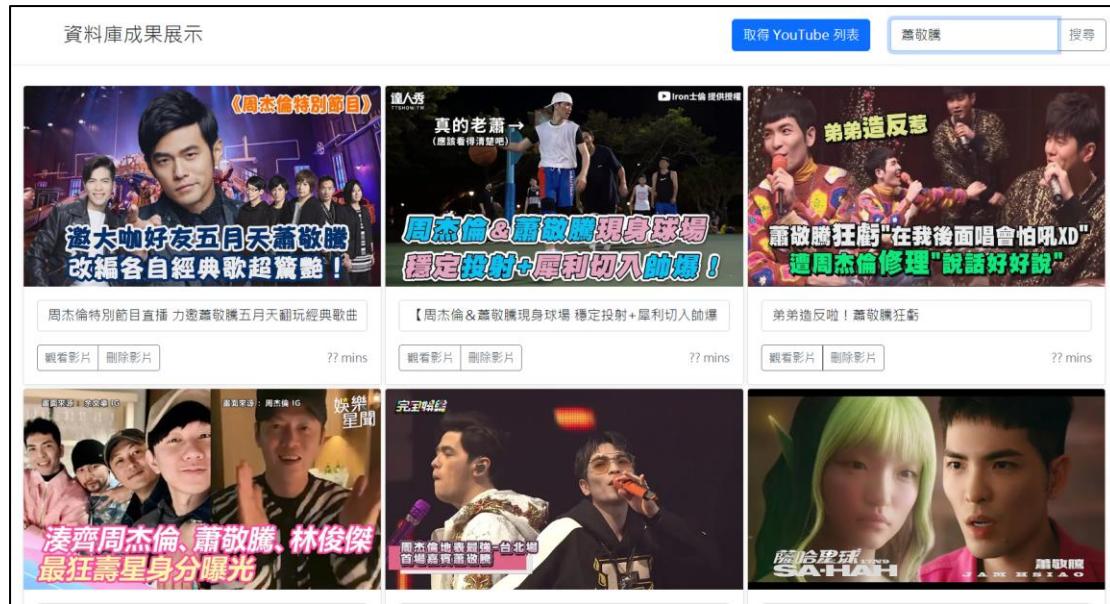
十八、網頁整合資料庫



圖：網頁初始樣式



圖：按下「取得 YouTube 列表」後，會將查詢結果放在網頁上



圖：檢索影音標題的結果

參考資料：

[1] JavaScript Fetch API 使用教學

<https://www.oxxostudio.tw/articles/201908/js-fetch.html>

[2] stack overflow : Get the data received in a Flask request

<https://stackoverflow.com/questions/10434599/get-the-data-received-in-a-flask-request>

[3] Bootstrap · The most popular HTML, CSS, and JS library in the world.

<https://getbootstrap.com/>

```
app.py
import json
from flask import Flask, render_template, jsonify, request
import pymysql

# 資料庫連線
connection = pymysql.connect(
    host='localhost',
    user='root',
    password='',
    database='my_db',
    charset='utf8mb4',
    cursorclass=pymysql.cursors.DictCursor
)
```

```
# 取得 cursor 物件，進行 CRUD
cursor = connection.cursor()

# 建立 Flask 物件
app = Flask(__name__)

''' 樣版 '''
# 套用網頁樣版
@app.route('/', methods=['GET'])
def index():
    return render_template('index.html')

''' Web API '''
# 取得 youtube 所有列表
@app.route('/youtube', methods=['GET'])
def getYouTubeList():
    # 預設回傳訊息
    dictResponse = {"success": False, "info": "查詢失敗"}

    try:
        # 查詢資料
        sql = "SELECT * FROM `youtube`"
        cursor.execute(sql)

        # 查詢結果列數大於 0 ，代表有資料
        if cursor.rowcount > 0:
            # 將查詢結果轉成 list 型態 (list 裡頭元素都是 dict)
            results = cursor.fetchall()

            # 新增屬性 results，將查詢結果送回前端頁面
            dictResponse["success"] = True
            dictResponse["info"] = "查詢成功"
            dictResponse["results"] = results
        else:
            dictResponse["info"] = "查詢結果為空"

    # 提交 SQL 執行結果
    cursor.close()
```

```
connection.commit()

except Exception as e:
    # 回滾
    connection.rollback()
    dictResponse["info"] = f"SQL 執行失敗: {e}"

# 回傳結果
return jsonify(dictResponse)

# 搜尋特定文字
@app.route("/youtube/title", methods = ["POST"])
def getYouTubeData():
    # 回傳訊息
    dictResponse = {"success": False, "info": "搜尋失敗"}

    try:
        # 取得搜尋字串
        title = request.json['title']

        # 處理成 SQL 看得懂的格式
        title = "%" + title.replace(" ", "%") + "%"

        # 查詢資料
        sql = "SELECT * FROM `youtube` WHERE `title` LIKE %s"
        cursor.execute(sql, {title})

        # 查詢結果列數大於 0 ，代表有資料
        if cursor.rowcount > 0:
            # 將查詢結果轉成 list 型態 (list 裡頭元素都是 dict)
            results = cursor.fetchall()

            # 新增屬性 results，將查詢結果送回前端頁面
            dictResponse["success"] = True
            dictResponse["info"] = "查詢成功"
            dictResponse["results"] = results
        else:
            dictResponse["info"] = "查詢結果為空"

    except Exception as e:
        # 回滾
        connection.rollback()
        dictResponse["info"] = f"SQL 執行失敗: {e}"
```

```
# 提交 SQL 執行結果
connection.commit()

except Exception as e:
    # 回滾
    connection.rollback()
    dictResponse["info"] = f"SQL 執行失敗: {e}"

# 回傳結果
return jsonify(dictResponse)

# 刪除或更新指定 youtube id 的資料
@app.route("/youtube/<id>", methods=["DELETE", "POST"])
def setYouTubeData(id):
    # 回傳訊息
    dictResponse = {"success": False, "info": "系統異常"}

    try:
        if request.method == 'DELETE':
            # 預設刪除用的回傳訊息
            dictResponse["info"] = "刪除失敗"

            # 查詢資料
            sql = "DELETE FROM `youtube` WHERE `id` = %s"
            cursor.execute(sql, (id))

            # 查詢結果列數大於 0 ，代表有資料
            if cursor.rowcount > 0:
                dictResponse["success"] = True
                dictResponse["info"] = "刪除成功"

        elif request.method == 'POST':
            # 預設更新用的回傳訊息
            dictResponse["info"] = "更新失敗"

        # 查詢資料
    except Exception as e:
        # 回滾
        cursor.rollback()
        dictResponse["info"] = f"SQL 執行失敗: {e}"
```

```

        sql = "UPDATE `youtube` SET `title` = %s WHERE `id` = %s"
        cursor.execute(sql, (request.json['title'], id))

        # 查詢結果列數大於 0，代表有資料
        if cursor.rowcount > 0:
            dictResponse["success"] = True
            dictResponse["info"] = "更新成功"

        # 提交 SQL 執行結果
        connection.commit()
    except Exception as e:
        # 回滾
        connection.rollback()
        dictResponse["info"] = f"SQL 執行失敗: {e}"

    # 回傳結果
    return jsonify(dictResponse)

# 主程式區域
if __name__ == '__main__':
    app.debug = True
    app.run(host='0.0.0.0', port=5000)

```

templates/index.html

```

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>資料庫成果展示</title>
    <!-- CSS only -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/
bootstrap.min.css" rel="stylesheet"

```

```
integrity="sha384-
+0n0xVW2eSR50omGNYDnhzAbDs0XxcvSN1TPprVMTNDbiYZCxYb0017+AMvyTG2x" cro
ssorigin="anonymous">
</head>

<body>
    <div class="container">
        <div class="row">
            <header class="d-flex flex-wrap justify-content-
center py-3 mb-4 border-bottom">
                <a href="/" class="d-flex align-items-center mb-3 mb-
md-0 me-md-auto text-dark text-decoration-none">
                    <svg class="bi me-2" width="40" height="32">
                        <use xlink:href="#bootstrap"></use>
                    </svg>
                    <span class="fs-4">資料庫成果展示</span>
                </a>
                <form class="form-group row">
                    <div class="col-auto">
                        <button type="button" class="btn btn-
primary" id="query">取得 YouTube 列表</button>
                    </div>
                    <div class="col-auto">
                        <div class="input-group">
                            <input type="text" class="form-
control" placeholder="輸入影音標題" aria-
describedby="search" id="title">
                            <button class="btn btn-outline-
secondary" type="button" id="search">搜尋</button>
                        </div>
                    </div>
                </form>
            </header>
        </div>
        <div id="results" class="row row-cols-1 row-cols-sm-2 row-
cols-md-3 g-1">請按下「取得 YouTube 列表」按鈕</div>
    </div>
```

```
<!-- JavaScript Bundle with Popper -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/bootstrap.bundle.min.js"
integrity="sha384-E7eJyOLLl6JZt1IaFz0t1kjNcDc5lVb30dU4o8lWVQFqP" crossorigin="anonymous"></script>

<!-- 自訂 js 語法 -->
<script>
    //選擇放置查詢結果的 div 元素
    const div_result = document.querySelector("div#results");

    //選擇取得所有 youtube 列表的 button 按鈕元素
    const btn_query = document.querySelector("button#query");

    //註冊「取得 youtube 列表」的按鈕 click 事件
    btn_query.addEventListener("click", function(event){
        //取得所有 youtube 列表
        fetch("/youtube",{
            method: "GET"
        }).then(function(response){
            return response.text();
        }).then(function(json){
            //將 json 轉成物件
            const dictData = JSON.parse(json);

            //若是回傳結果為 True，則將 results 屬性的內容呈現在網路上
            if(dictData["success"]){
                let html = ``;
                for(let result of dictData["results"]){
                    html += `<div class="col" data-
id="${result['id']}>
    <div class="card shadow-sm">
        
            <p class="card-text">

```

```
        <input type="text" class="form-
control update" value="${result['title']}" data-id="${result['id']}">
        </p>
        <div class="d-flex justify-content-
between align-items-center">
            <div class="btn-group">
                <a class="btn btn-sm btn-outline-
secondary" href="${result['link']}" target="_blank">觀看影片</a>
                <a class="btn btn-sm btn-outline-
secondary delete" href="#" data-id="${result['id']}>刪除影片</a>
            </div>
            <small class="text-muted">?? mins</small>
        </div>
    </div>
    </div>
</div>`;
```

}

```
//呈現查詢結果
div_result.innerHTML = html;
} else {
    alert(dictData["info"]);
}
});
```

});

```
//選擇輸入影音標題的 input 元素
const input_title = document.querySelector("input#title");

//選擇搜尋按鈕 button 的元素
const btn_search = document.querySelector("button#search");

//註冊搜尋按鈕的 click 事件
btn_search.addEventListener("click", function(event){
    //取得搜尋文字
    let title_value = input_title.value;

    //進行檢索
```

```
fetch("/youtube/title", {
    method: "POST",
    headers: {'content-type': 'application/json'},
    body: JSON.stringify({ title: title_value })
}).then(function(response){
    return response.text();
}).then(function(json){
    //將 json 轉成物件
    const dictData = JSON.parse(json);

    //若是回傳結果為 True，則將 results 屬性的內容呈現在網路上
    if(dictData["success"]){
        let html = ``;
        for(let result of dictData["results"]){
            html += `<div class="col" data-
id="${result['id']}>
    <div class="card shadow-sm">
        
            <p class="card-text">
                <input type="text" class="form-
control update" value="${result['title']}" data-id="${result['id']}"/>
            </p>
            <div class="d-flex justify-content-
between align-items-center">
                <div class="btn-group">
                    <a class="btn btn-sm btn-outline-
secondary" href="${result['link']}'" target="_blank">觀看影片</a>
                    <a class="btn btn-sm btn-outline-
secondary delete" href="#" data-id="${result['id']}>刪除影片</a>
                </div>
                <small class="text-muted">?? mins</small>
            </div>
        </div>
    </div>
</div>`;
        }
    }
})
```

```
//呈現查詢結果
    div_result.innerHTML = html;
} else {
    alert(dictData["info"]);
}
});

//註冊所有使用到「點擊」事件的元素
document.addEventListener("click", function(event){
    if(event.target){
        if( event.target.tagName.toLowerCase() == 'a' && even
t.target.classList.contains('delete') ){
            //預防預設事件觸發，方便之後撰寫自訂事件觸發的程式碼
            event.preventDefault();

            //取得點擊時的 a 元素
            const a = event.target;

            //取得 a 元素的屬性值 (id)
            let id = a.getAttribute('data-id');

            //刪除指定 id 的資料
            fetch(`youtube/${id}`,{
                method: "DELETE"
            }).then(function(response){
                return response.text();
            }).then(function(json){
                //將 json 轉成物件
                const dictData = JSON.parse(json);

                //若是回傳結果為 True，則將 results 屬性的內容呈現在網路上
                if(dictData["success"]){
                    alert(dictData["info"]);

                    //將點選的元素刪除
                }
            })
        }
    }
})
```

```
        document.querySelector(`div.col[data-
id="${id}"]`).remove();
    } else {
        alert(dictData["info"]);
    }
});

//註冊所有使用到「離開焦點」事件的元素
document.addEventListener("focusout", function(event){
    if(event.target){
        if( event.target.tagName.toLowerCase() == 'input' &&
event.target.classList.contains('update')){
            //預防預設事件觸發，方便之後撰寫自訂事件觸發的程式碼
            event.preventDefault();

            //取得離開焦點時的 input 元素
            const input = event.target;

            //取得 input 元素的屬性值
            let id = input.getAttribute('data-id');
            let title_value = input.value;

            //更新指定 id 的資料
            fetch(`/youtube/${id}`,{
                method: "POST",
                headers: { 'content-type': 'application/json' },
                body: JSON.stringify({ title: title_value })
            }).then(function(response){
                return response.text();
            }).then(function(json){
                //將 json 轉成物件
                const dictData = JSON.parse(json);

                //若是回傳結果為 True，則將 results 屬性的內容呈現在網路上
                if(dictData["success"]){

```

```
        alert(dictData["info"]);
    } else {
        alert(dictData["info"]);
    }
});

}

});

</script>
</body>

</html>
```

作業

- 尋找喜歡的議題，選擇一個網站或網頁，透過上課講過的技術，取得結構性的資料。
- 自動建立資料庫 `homework`，字元編碼為 `utf8mb4`，排序規則為 `utf8mb4_unicode_ci`。
- 自訂資料表，數量不限，依需求規劃，名稱自取。每個資料表至少需要 `id`、`created_at`、`updated_at` 欄位。`id` 可以是自動編號 (`autoincrement`)，或是自行程式寫入。
- 整合 Flask 以及過去學習過的 HTML、CSS 技術，或是其它網頁排版框架（例如 bootstrap），透過網頁元素的事件監聽功能，於事件觸發後，將資料庫的內容呈現在網頁上。