MySQL 資料庫實務

Database Management System

DE - LONG CAL

認識資料庫系統

狹義的資料庫定義

- 將這些定義和限制儲存於資料庫
- 將真實現象忠實的 抽象化 & 一般化 後，對資料明確的定義和規範一些完整性的限制。

廣義的資料庫定義

- 一群具有相關性之資料所形成的一個集合體
- 文書處理軟體如微軟公司的Word、Excel、Access...等等的儲存方式，稱為『資料庫』。

認識資料庫系統

資料庫（Database）

- 由作業系統的檔案系統的『檔案』組成。
- 由『資料庫管理系統』來進行管理與存取。
- 底層檔案大小會影響『資料庫』存放的空間大小。
- 儲存空間不足時，必須從底層擴增檔案給該資料庫使用。

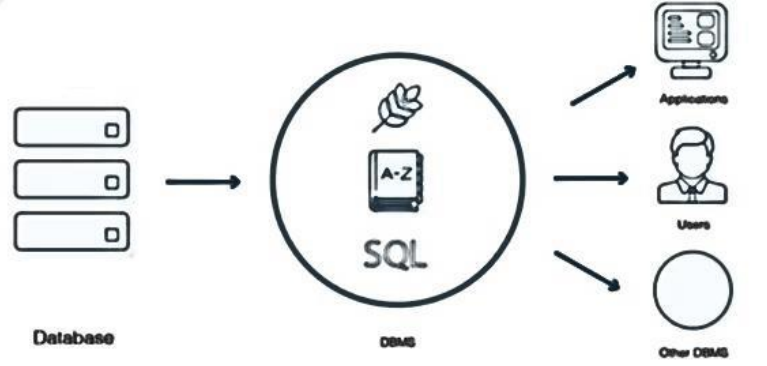
資料庫系統(Database System)

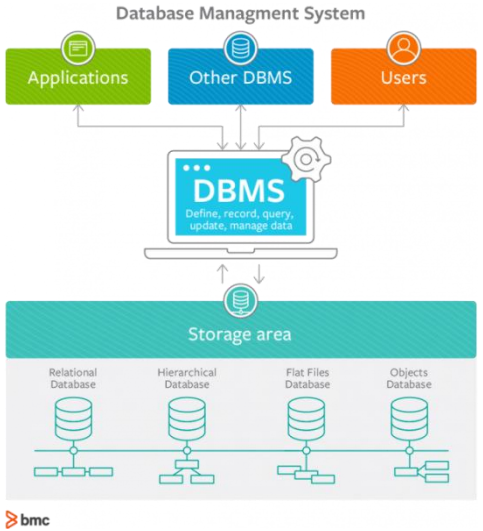
- 藉由資料庫管理系統對資料的儲存和管理，以及具有商業。
- 是由一些彼此相關的資料，
  以及存取這些資料的『應用程式』所組成的一個集合體邏輯的『應用程式』來達成企業需求。

認識資料庫系統

資料庫管理系統(DBMS)

- 讓使用者或程式設計人員可以透過所提供的共用軟體來進行資料的定義、控制和存取的動作。





結構化查詢語言(SQL)

資料操作語言（DML）

- 主要是針對於資料表或檢視表內的資料進行存取的操作。
  常用的指令包括： SELECT 、 INSERT 、 UPDATE 、 DELETE。

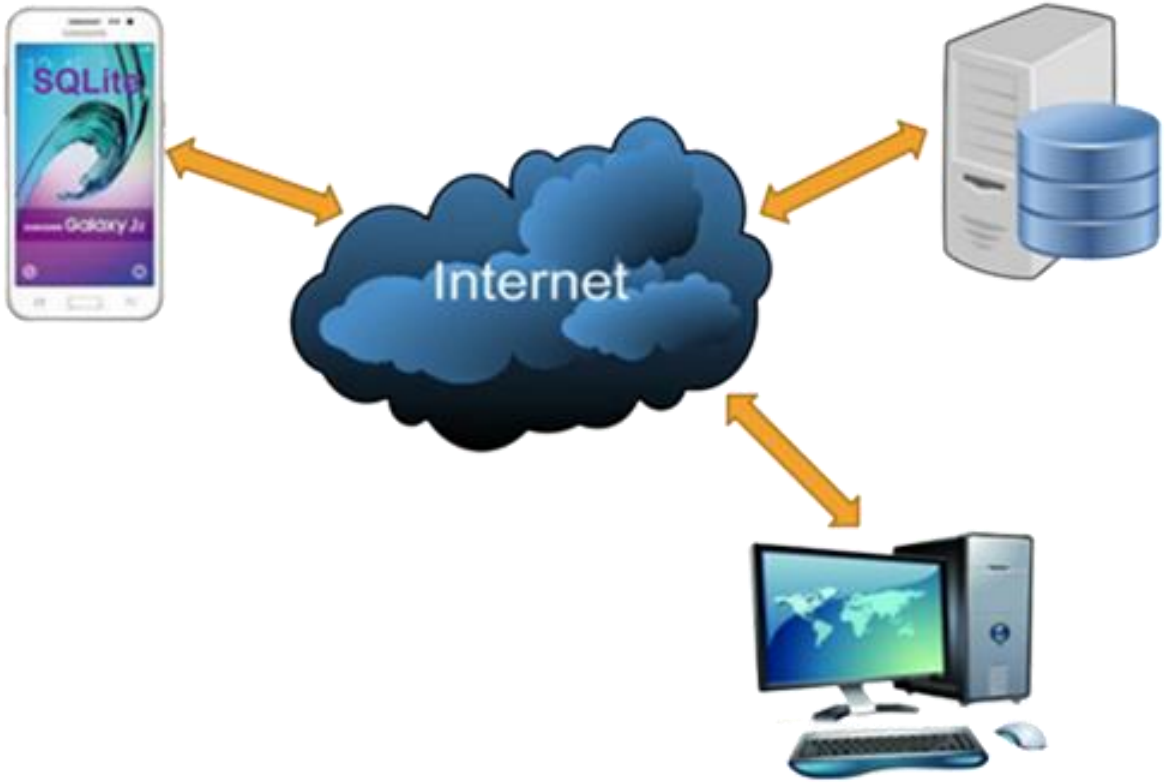
資料定義語言（DDL）

- 主要是針對不同物件的結構定義，進行建立、刪除、修改等指令操作，
  Ex： CREATE、 DROP、 ALTER。

資料控制語言（DCL）

- 主要是針對資料庫管理系統的安全進行控制，常用的指令包括： GRANT、 DENY、 REVOKE。

Server 伺服器



Download

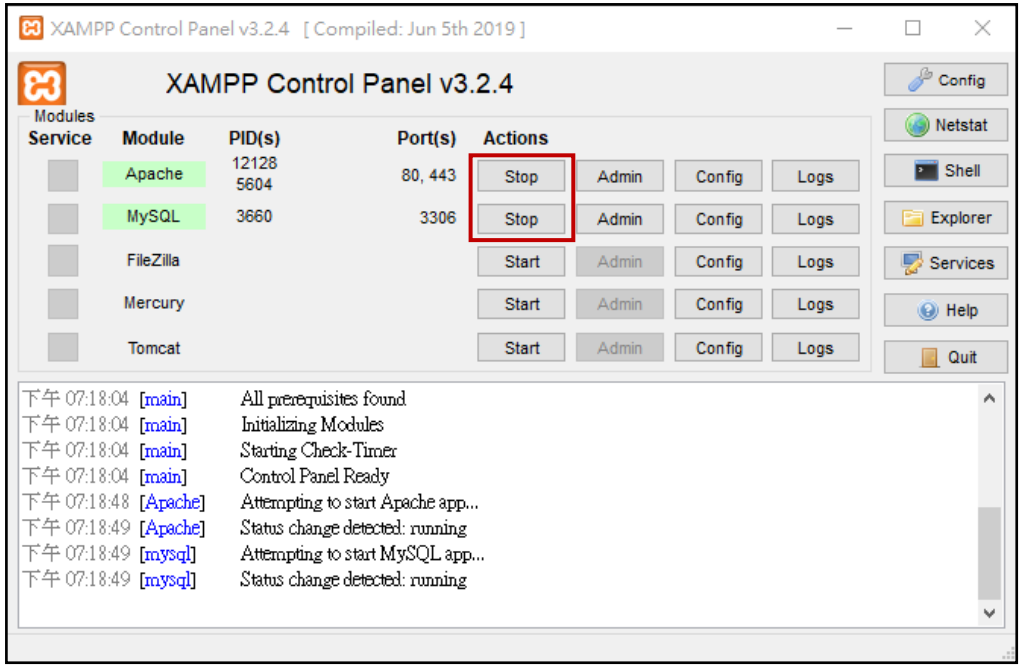
- Go to the below web for your download
- https://www.apachefriends.org/zh_tw/download.html

PHP 8.0.5
Apache 2.4.47
MariaDB 10.4.18
phpMyAdmin 5.1.0
XAMPP Control Panel 3.2.4

Tomcat 8.5.65
OpenSSL 1.1.1
Webalizer 2.23-04
FileZilla FTP Server 0.9.41
Strawberry Perl 5.32.0.1 Portable
Mercury Mail Transport System 4.63

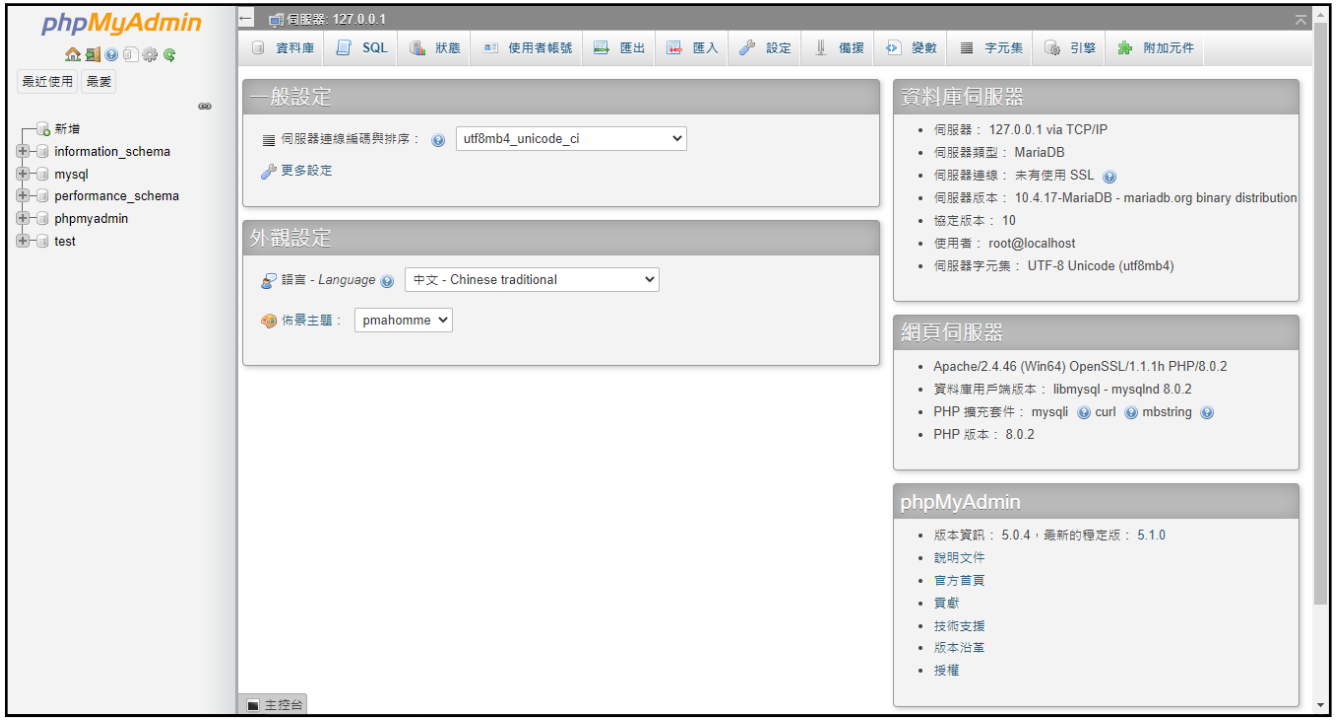
XAMPP

- Run control panel
- Click on the “start” buttons for Apache and MySQL



phpmyadmin

- Run http://localhost/phpmyadmin/ on your browser



數學運算

優先順序	運算子	說明	範例	運算結果
1	%	餘數	7 % 3	1
	MOD	餘數	7 MOD 3	1
	*	乘法	7 * 3	21
	/	除法	7 / 3	2.333
	DIV	除法(整數)	7 DIV 3	2
2	+	加法	7 + 3	10
	-	減法	7 - 3	4

比較運算子

優先順序	運算子	說明
1	=	等於
	<=>	等於
	!=	不等於
	<	小於
	<=	小於等於
	>	大於
	>=	大於等於

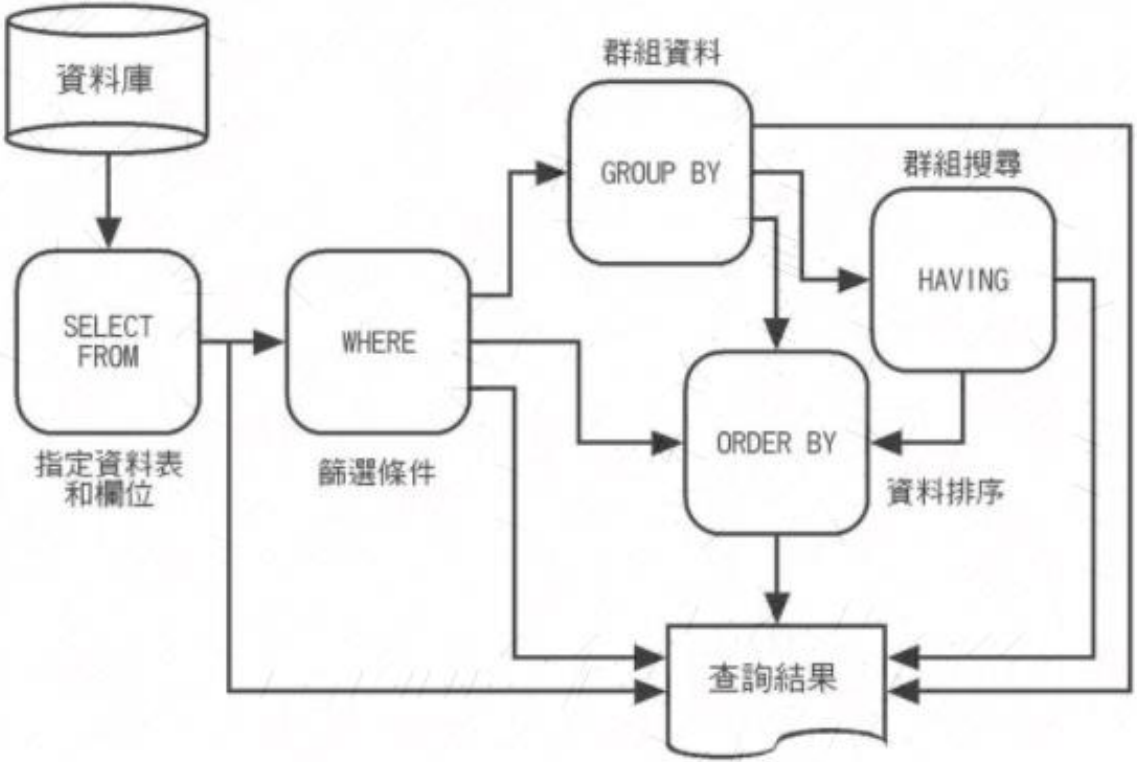
邏輯運算子

優先順序	運算子	說明
1	NOT	非
2	&&	且
	AND	且
3	||	或
	OR	或
	XOR	互斥

其他條件運算子

運算子	說明
BETWEEN ... AND ...	範圍比較
IN (...)	成員比較
IS	是 ...
IS NOT	不是 ...
LIKE	像 ...

SELECT查詢指令-圖例



查詢敘述

以「SELECT」子句開始

所需要的查詢工作搭配其他子句完成

SELECT	想要查詢的欄位
FROM	想要查詢的表格
WHERE	查詢條件
GROUP BY	分組設定
HAVING	分組條件
ORDER BY	排序設定
LIMIT	限制設定

子句的順序

SELECT	想要查詢的欄位
FROM	想要查詢的表格
WHERE	查詢條件
GROUP BY	分組設定
HAVING	分組條件
ORDER BY	排序設定
LIMIT	限制設定

SQL SELECT

語法:

```
SELECT "欄位名"
FROM   "表格名";
```

答案:

```
SELECT Store_Name
FROM   Store_Information;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name
Los Angeles
San Diego
Los Angeles
Boston

SQL AS

語法:

```
SELECT "表格別名"."欄位1" A欄位別名
FROM   "表格名" AS"表格別名";
```

答案:

```
SELECT A1.Store_Name AS Store, SUM(A1.Sales) AS 'Total Sales'
FROM Store_Information AS A1
GROUP BY A1.Store_Name;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name	Sales
Boston	700
Los Angeles	1500
San Diego	250

SQL DISTINCT

語法:

```
SELECT DISTINCT "欄位名"
FROM   "表格名";
```

答案:

```
SELECT DISTINCT Store_Name
FROM   Store_Information;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name
Los Angeles
San Diego
Boston

SQL WHERE

語法:

```
SELECT "欄位名"
FROM   "表格名"
WHERE  "條件";
```

答案:

```
SELECT Store_Name
FROM   Store_Information
WHERE  Sales > 1000;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name
Los Angeles

SQL AND OR

語法:

```
SELECT "欄位名"
FROM   "表格名"
WHERE  "簡單條件"
{[AND|OR] "簡單條件"}+;
```

答案:

```
SELECT Store_Name
FROM   Store_Information
WHERE  Sales > 1000
OR (Sales < 500 AND Sales > 275);
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
San Francisco	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name
Los Angeles
San Francisco

SQL IN

語法:

```
SELECT "欄位名"
FROM "欄位名"= '值一'
WHERE  "欄位名" IN ('值一', '值二', ...);
```

答案:

```
SELECT *
FROM   Store_Information
WHERE  Store_Name IN ('Los Angeles', 'San Diego');
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
San Francisco	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07

SQL BETWEEN

語法:

```
SELECT "欄位名"
FROM   "表格名"
WHERE  "欄位名" BETWEEN '值一' AND '值二';
```

答案:

```
SELECT *
FROM   store_information
WHERE  Txn_Date BETWEEN '2019-10-06' AND '2019-10-08';
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
San Francisco	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name	Sales	Txn_Date
San Diego	250	2019-10-07
Boston	700	2019-10-08

SQL 萬用字元

```
SELECT Name
FROM   City
WHERE  Name LIKE 'w%';
```

```
Walsall
Willemstad
Wolverhampton
...
```

```
SELECT Name
FROM   City
WHERE  Name LIKE '%w';
```

```
Krakw
Glasgow
LuckNow
...
```

```
SELECT Name
FROM   City
WHERE  Name LIKE '%w%';
```

```
Glasgow
Newcastle
Willemstad
...
```

SQL LIKE

語法:

```
SELECT "欄位名"
FROM   "表格名"
WHERE  "欄位名" LIKE {模式};
```

答案:

```
SELECT *
FROM   Store_Information
WHERE  store_name LIKE '%AN%';
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
San Francisco	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
San Francisco	300	2019-10-08

SQL ORDER BY

語法:

```
SELECT "欄位名"
FROM "表格名"
[WHERE "條件"]
ORDER BY "欄位名" [ASC, DESC];
```

```
ORDER BY "欄位一" [ASC, DESC], "欄位二" [ASC, DESC]
```

答案:

```
SELECT   Store_Name, Sales, Txn_Date
FROM     Store_Information
ORDER BY Sales DESC;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
San Francisco	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name	Sales ▼ 1	Txn_Date
Los Angeles	1500	2019-10-05
Boston	700	2019-10-08
San Francisco	300	2019-10-08
San Diego	250	2019-10-07

SQL 平均值

語法:

```
SELECT AVG("欄位名")
FROM   "表格名";
```

答案:

```
SELECT AVG(Sales)
FROM   Store_Information;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

AVG(Sales)
687.5000

SQL COUNT

語法:

```
SELECT COUNT("欄位名")
FROM   "表格名";
```

答案:

```
SELECT COUNT(Store_Name)
FROM   Store_Information
WHERE  Store_Name IS NOT NULL;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

COUNT(Store_Name)
4

SQL COUNT

語法:

```
SELECT COUNT("欄位名")
FROM   "表格名";
```

答案:

```
SELECT COUNT(DISTINCT Store_Name)
FROM   Store_Information;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

COUNT(DISTINCT Store_Name)
3

SQL 最大值

語法:

```
SELECT MAX ("欄位名")
FROM   "表格名";
```

答案:

```
SELECT MAX(Sales)
FROM   Store_Information;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

MAX(Sales)
1500

SQL 總和

語法:

```
SELECT SUM("欄位名")
FROM   "表格名";
```

答案:

```
SELECT SUM(Sales)
FROM Store_Information;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

SUM(Sales)
2750

SQL GROUP BY

語法:

```
SELECT   "欄位1", SUM("欄位2")
FROM     "表格名"
GROUP BY "欄位1";
```

答案:

```
SELECT   Store_Name, SUM(Sales)
FROM     Store_Information
GROUP BY Store_Name;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name	SUM(Sales)
Los Angeles	1800
San Diego	250
Boston	700

SQL HAVING

語法:

```
SELECT   "欄位1", SUM("欄位2")
FROM     "表格名"
GROUP BY "欄位1"
HAVING (函數條件);
```

答案:

```
SELECT   Store_Name, SUM(Sales)
FROM     Store_Information
GROUP BY Store_Name
HAVING SUM(Sales) > 1500;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name	SUM(Sales)
Los Angeles	1800

SQL 別名

語法:

```
SELECT "表格別名"."欄位1" "欄位別名"
FROM   "表格名" "表格別名";
```

答案:

```
SELECT   A1.Store_Name Store, SUM(A1.Sales) "Total Sales"
FROM     Store_Information A1
GROUP BY A1.Store_Name;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

結果:

Store_Name	Total Sales
Los Angeles	1800
San Diego	250
Boston	700

SQL 表格連接

答案:

```
SELECT   A1.Region_Name REGION, SUM(A2.Sales) SALES
FROM     Geography A1, Store_Information A2
WHERE    A1.Store_Name = A2.Store_Name
GROUP BY A1.Region_Name;
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

Geography 表格

Region_Name	Store_Name
East	Boston
East	New York
West	Los Angeles
West	San Diego

結果:

REGION	SALES
East	700
West	2050

SQL CREATE TABLE

語法:

```
CREATE TABLE "表格名"
(
 "欄位 1" "欄位 1 資料種類",
 "欄位 2" "欄位 2 資料種類",
 ... );
```

答案:

```
CREATE TABLE Customer
(First_Name  char(50),
 Last_Name   char(50),
 Address     char(50),
 City        char(50),
 Country     char(25),
 Birth_Date datetime);
```

Customer 表格

First_Name	Last_Name	Address	City	Country	Birth_Date

SQL CREATE TABLE

語法:

```
INSERT INTO "表格名" ("欄位1", "欄位2", ...)
VALUES ("值1", "值2", ...);
```

答案:

```
INSERT INTO Store_Information (Store_Name, Sales, Txn_Date)
VALUES ('Los Angeles', 900, '1999-10-07');
```

Store_Information 表格

欄位名稱	資料種類
Store_Name	Char(50)
Sales	Float
Txn_Date	datetime

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	900	1999-10-07

SQL UPDATE

語法:

```
UPDATE "表格"
SET    "欄位1" = [值1],
WHERE  "條件";
```

語法:

```
UPDATE "表格名"
SET    "欄位1" = [新值] "欄位2" = [值2]
WHERE  "條件";
```

語法:

```
UPDATE Store_Information
SET    Sales = 500
WHERE  Store_Name = 'Los Angeles'
AND    Txn_Date = '2019-10-08';
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	300	2019-10-08
Boston	700	2019-10-08

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	500	2019-10-08
Boston	700	2019-10-08

SQL Delete

語法:

```
DELETE FROM "表格名"
WHERE "條件";
```

答案:

```
DELETE FROM Store_Information
WHERE Store_Name = 'Los Angeles';
```

Store_Information 表格

Store_Name	Sales	Txn_Date
Los Angeles	1500	2019-10-05
San Diego	250	2019-10-07
Los Angeles	500	2019-10-08
Boston	700	2019-10-08

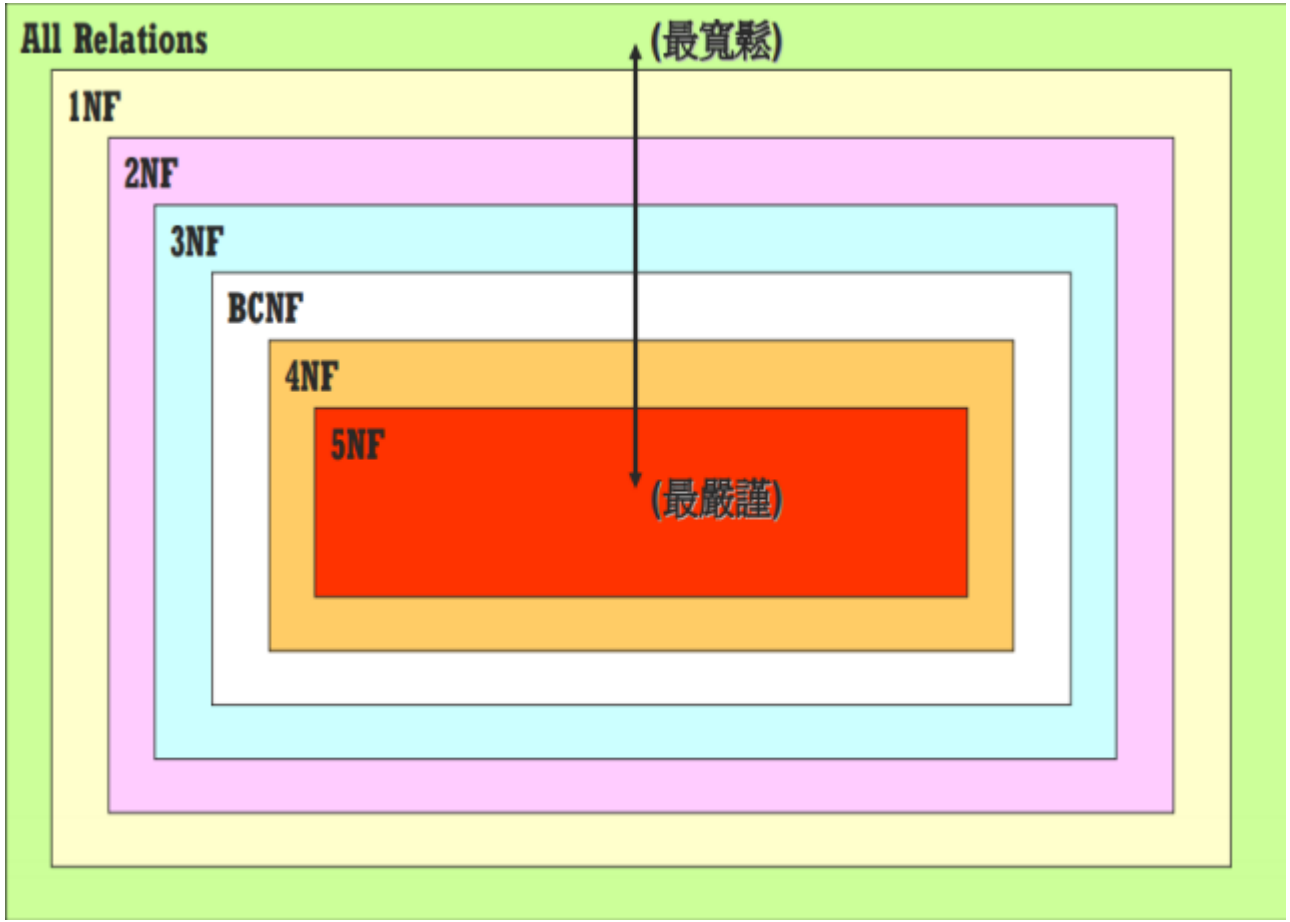
Store_Information 表格

Store_Name	Sales	Txn_Date
San Diego	250	2019-10-07
Boston	700	2019-10-08

Database Design

為何正規化

- 正規化的目的：
  - 降低資料重覆性(Data Redundancy)
  - 避免產生插入、刪除 & 更新 可能的 異常(Anomailes)
- 正規畫主要是對表格做分割的動作。
- 沒有正規化會造成：
  - 容易有 資料重複儲存 的浪費情形
  - 資料在做 插入、刪除 & 更新 動作時產生 異常(Anomailes) 情形


All Relations
(最寬鬆)
1NF
2NF
3NF
BCNF
4NF
5NF
(最嚴謹)

The End