**第二章 基礎查詢**

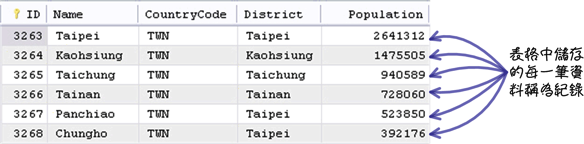
**1 查詢資料前的基本概念**

**1.1 表格、紀錄與欄位**

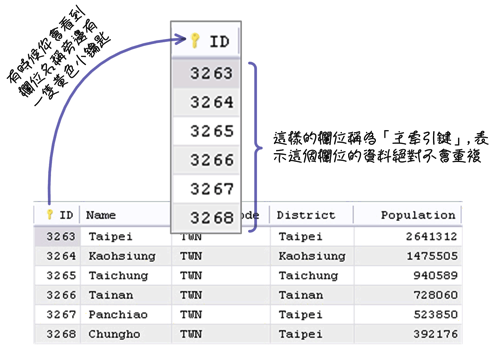
表格是資料庫儲存資料的基本元件，它是由一些欄位組合而成的，儲存在表格中的每一筆紀錄就擁有這些欄位的資料。以儲存城市資料的表格「city」來說，設計這個表格的人希望一個城市資料需要包含編號、名稱、國家代碼、區域和人口數量，所以他為「city」表格設計了這些「欄位(column)」：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_01.png)

儲存在表格中的每一筆資料稱為「列(row)」或「紀錄(record)」：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_02.png)

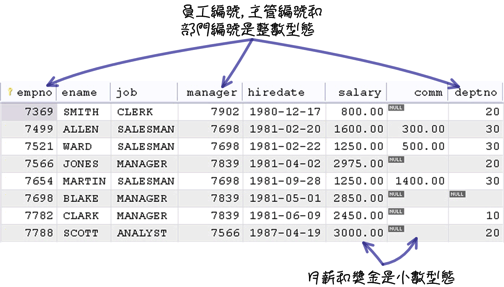
在設計表格的時候，通常會指定一個欄位為「主索引鍵(primary key)」：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_03.png)

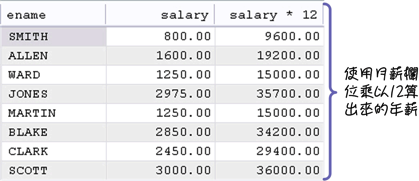
註：主索引鍵會在「第八章、表格與索引」中詳細的討論。

**1.2 認識資料型態**

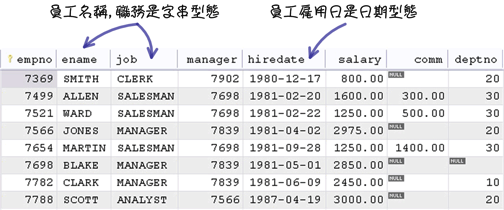
資料庫中可以儲存各種不同的資料，SQL提供許多不同的「資料型態」讓你應付這些不同的需求。在開始查詢資料之前，你要先認識最常見、也是最基本的資料型態。第一種是數值，為了更精準的保存數值資料，SQL提供整數與小數兩種數值型態：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_04.png)

你可以依照自己的需求，使用儲存的數值資料執行數學運算：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_05.png)

常用的資料型態還有「字串」與「日期」：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_06.png)

在SQL敘述中使用字串資料的時候，字串資料的前後要使用單引號或雙引號：

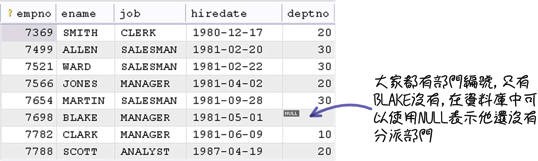
[mysql_03_snap_07](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_07.png)

使用日期資料的時候，MySQL資料庫預設的日期格式是「年-月-日」。與字串資料一樣，前後也要使用單引號或雙引號：

[mysql_03_snap_08](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_08.png)

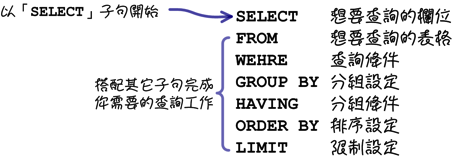
註：字串與日期資料型態會在「第七章、儲存引擎與資料型態、欄位資料型態」中詳細的討論。

另外一種在資料庫中比較特殊的資料型態是「NULL」，它不像數值、字串或日期資料型態是一個明確的資料，「NULL」是用來表示「不確定」、「未知」或「沒有」的資料：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_09.png)

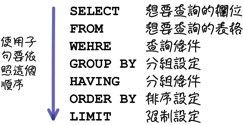
**2 查詢敘述**

在執行資料庫的操作中，查詢算是最常見也是最複雜的工作，所以一個查詢敘述所使用到的子句也最多，下列是查詢敘述的基本語法：

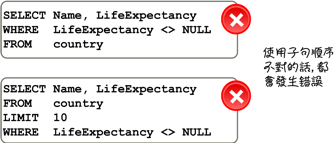
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_10.png)

這一章會討論「SELECT」、「FROM」、「WHERE」、「ORDER BY」和「LIMIT」五個子句組合起來的查詢敘述。其它的子句會在下一章繼續討論。

在你使用「SELECT」搭配各種子句來查詢資料時，要特別注意子句使用的順序：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_64.png)

就算你每一個子句的寫法都沒有出錯，如果順序不對了：

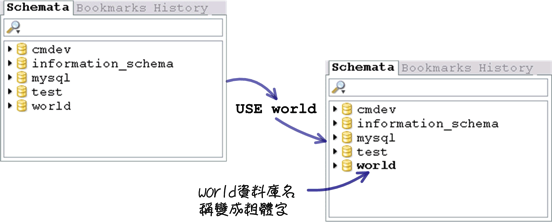
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_65.png)

**2.1 指定使用中的資料庫**

一個資料庫伺服器可以建立許多需要的資料庫，所以在你執行任何資料庫的操作前，通常要先指定使用的資料庫。下列是指定資料庫的指令：

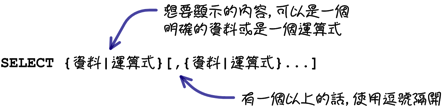
[mysql_03_snap_11](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_11.png)

如果你使用「MySQL Workbench」這類的工具軟體，畫面上看起來會像這樣：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_12.png)

**2.2 只有SELECT**

一個SQL查詢敘述一定要以「SELECT」子句開始，再搭配其它的子句完成查詢資料的工作。你可以單獨使用「SELECT」子句，只不過這樣的用法跟資料庫一點關係都沒有，它只不過把你輸入的內容顯示出來而已：

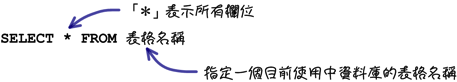
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_13.png)

例如下列的查詢敘述，只是簡單的顯示字串和計算結果，並不會查詢資料庫中的資料：

|  |  |
| --- | --- |
| 1 | SELECT 'My name is Simon Johnson', 35 \* 12 |

**2.3 指定欄位與表格**

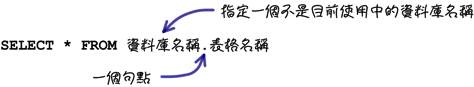
一般所謂的查詢敘述，通常是查詢資料庫中的資料，所以「SELECT」子句會搭配「FROM」子句來使用，而「SELECT」後面可以指定「\*」表示要查詢指定表格的所有欄位：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_14.png)

如果目前使用中的資料庫為「world」，下列的敘述可以查詢「world」資料庫中，「city」表格的所有資料：

|  |  |
| --- | --- |
| 1 | SELECT \* FROM city |

一個資料庫伺服器可以建立許多需要的資料庫，所以在你執行任何資料庫的操作前，都要使用「USE」敘述指定一個使用中的資料庫。不過你也可以在SQL敘述中使用下列的語法來指定資料庫：

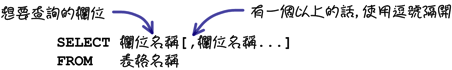
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_15.png)

如果目前使用中的資料庫是「world」，你不用先使用「USE cmdev」敘述切換使用中的資料庫，可以使用下列的語法查詢「cmdev」資料庫中的「emp」表格：

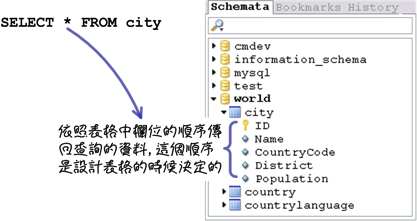
|  |  |
| --- | --- |
| 1 | SELECT \* FROM cmdev.emp |

**2.4 指定需要的欄位**

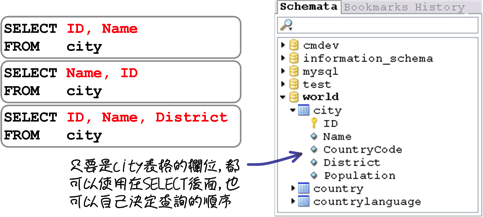
有時候你並不需要查詢一個表格中所有的欄位，所以你可以在「SELECT」子句後面自己指定需要的欄位：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_16.png)

如果你在「SELECT」後面使用「\*」的話：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_17.png)

你可以依照自己的需要決定要查詢哪些欄位和順序：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_18.png)

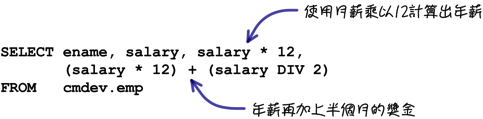
**2.5 數學運算**

除了查詢表格中的欄位外，你可以加入任何需要的運算，這裡先討論一般常見的數學運算。下列是很常用來執行數學運算的運算子：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **優先順序** | **運算子** | **說明** | **範例** | **運算結果** |
| 1 | % | 餘數 | 7 % 3 | 1 |
| 1 | MOD | 餘數 | 7 MOD 3 | 1 |
| 1 | \* | 乘 | 7 \* 3 | 21 |
| 1 | / | 除 | 7 / 3 | 2.333 |
| 1 | DIV | 除（整數） | 7 DIV 3 | 2 |
| 2 | + | 加 | 7 + 3 | 10 |
| 2 | - | 減 | 7 – 3 | 4 |

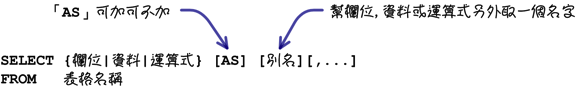
註：優先順序的數字從1開始，1表示優先權比較高，2比較低，以此類推。就跟一般數學運算的先乘除後加減一樣：在一個運算式中，優先權高的先算完，再換低優先權繼續算；同樣優先權的就由左到右計算。你也可以在運算式中使用左右括號，括號中的運算會先執行。

以「cmdev」資料庫中的員工表格(emp)來說，想要計算員工的年薪，就可以使用這些運算子來完成你的查詢工作：

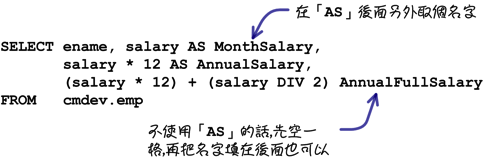
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_19.png)

**2.6 別名**

你可以另外為「SELECT」後面查詢的資料取一個自己想要的名稱，這個作法稱為「別名(alias name)」：

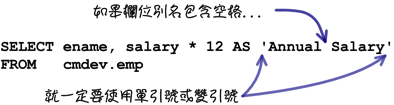
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_20.png)

取欄位別名會讓執行查詢後的結果，使用你自己取的名稱為欄位名稱：

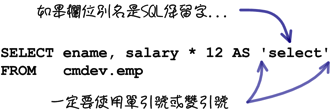
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_21.png)

註：幫一般欄位取一個欄位別名是比較沒有必要的，如果是運算式的話，通常就要幫它取一個欄位別名來取代原來一大串的運算式。

在取欄位別名的時候要特別注意下列的狀況：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_22.png)

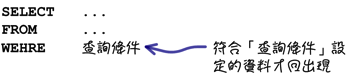
另外如果你「堅持」要使用SQL語法中的保留字來當作欄位別名的話：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_23.png)

如果違反上列兩個規定，執行敘述以後會發生錯誤。

**3 條件查詢**

使用「SELECT」和「FROM」執行的查詢敘述，是把你在「FROM」子句指定表格裡所有的紀錄傳回來。資料庫最大的好處就是可以隨時依照需要查詢部份紀錄資料，你可以搭配「WHERE」子句執行查詢條件的設定：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_24.png)

**3.1 比較運算子**

要使用「WHERE」執行查詢條件的設定，你會使用下列基礎的比較運算子：

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

註：「<=>」運算子在後面「NULL值的判斷」會討論。

使用這些基礎的比較運算子就可以完成一些簡單的條件設定：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_25.png)

設定日期資料型態的條件也是很常見的：

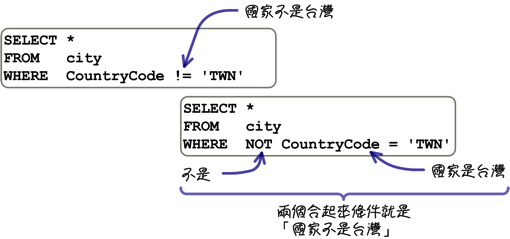
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_26.png)

**3.2 邏輯運算子**

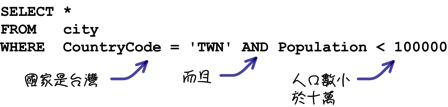
查詢條件的設定，有時候會像前面討論的單一條件一樣，並不會太複雜；不過也很常遇到在一個查詢的需求中，需要設定一個以上的條件，那你就會用到下列的運算子：

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

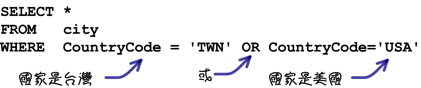
「NOT」運算子比較特殊一些，在一般的需求中，比較不會用到它。以下列的需求來說：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_29.png)

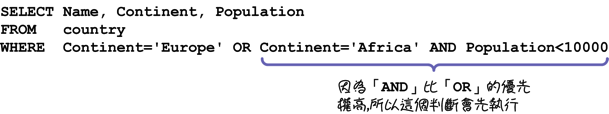
如果想要查詢國家代碼是「TWN」，而且人口數量小於十萬的城市，就必須設定兩個條件，而兩個條件之間，依照「而且」的需求，使用「AND」來結合兩個條件：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_27.png)

如果想要查詢國家代碼是「TWN」或是「USA」的城市，在兩個條件之間依照「或」的需求，使用「OR」來結合兩個條件：

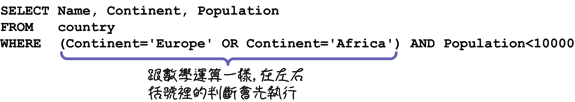
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_28.png)

在邏輯運算子的介紹中，它們也同樣有「優先順序」的。如果你想要查詢在歐洲(Europe)或非洲(Aftica)國家，而且人口數要小於一萬。使用下列的查詢條件所得到的資料，跟你想要的卻不一樣：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_30.png)

**會造成錯誤的結果！**

如果有多個查詢條件的設定，全部都是「AND」或全部都是「OR」的話，就沒有這類問題；如果查詢條件中，有「AND」和「OR」同時出現的話，就要依照你的需要，視情況加上左右刮號來控制條件的設定：

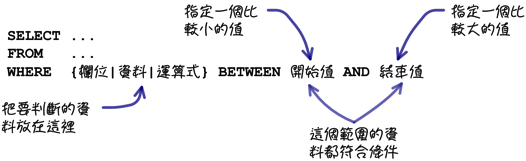
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_31.png)

**3.3 其它條件運算子**

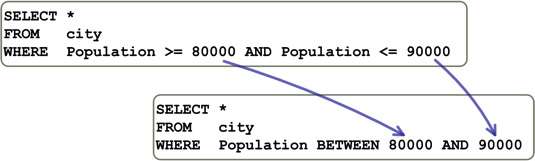
一般的條件和邏輯運算子，已經可以應付大部份的查詢條件需求。下列還有一些可以用在特殊用途或是提供替代寫法的條件設定：

* BETWEEN … AND …：範圍比較
* IN (…)：成員比較
* IS：是…
* IS NOT：不是…
* LIKE：像…

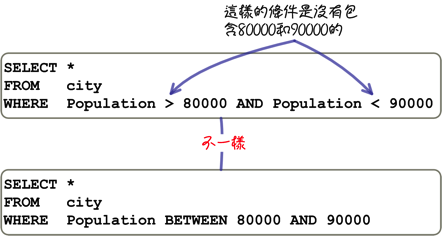
「BETWEEN … AND …」用來執行一個指定範圍條件的設定：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_32.png)

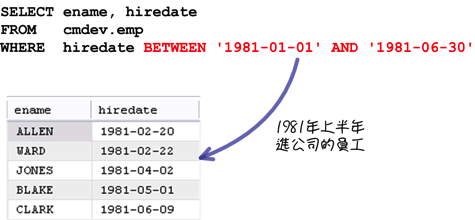
如果要查詢人口數量在八萬到九萬之間的城市資料，可以有下列兩種條件的寫法，它們執行以後的結果是完全一樣的：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_33.png)

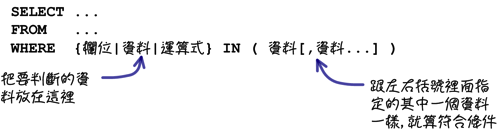
使用「BETWEEN … AND …」的條件設定會包含指定的資料，所以下列兩個查詢條件所得到的結果就不一樣了：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_34.png)

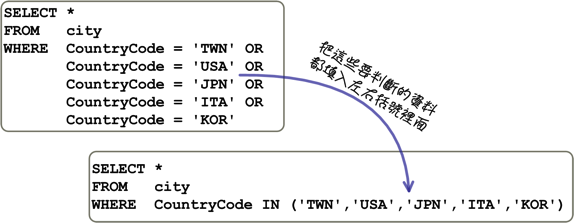
「BETWEEN … AND …」使用在日期資料時，也可以完成某一個日期範圍的判斷：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_63.png)

「IN (…)」使用在一組成員資料的比對條件設定：

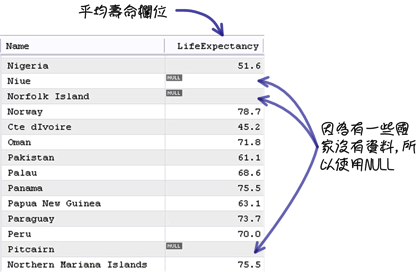
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_35.png)

下列兩個查詢敘述，都可以得到國家代碼是「TWN、USA、JPN、ITA和KOR」的城市資料，可是使用「IN (…)」來設定條件的話，看起來會簡潔很多：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_36.png)

**3.4 NULL值的判斷**

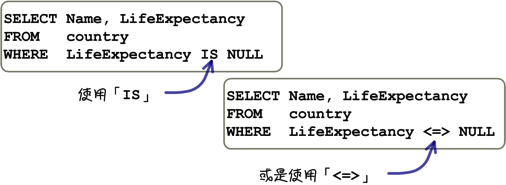
在國家表格中，有一個儲存平均壽命的欄位「LifeExpectancy」，不過資料庫中的資料並沒有很完整，所以有一些國家是沒有這個資料的，所以會使用「NULL」值來表示：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_37.png)

如果想要查詢沒有平均壽命資料的國家，也就是平均壽命的欄位值是「NULL」，你可能會使用下列的敘述：

|  |  |
| --- | --- |
| 1  2  3 | SELECT Name, LifeExpectancy  FROM country  WHERE LifeExpectancy = NULL |

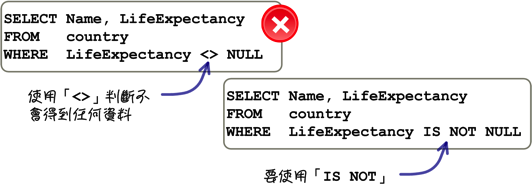
上列的敘述執行以後，並沒有傳回任何紀錄，這表示並沒有資料符合你設定的查詢條件。所以「NULL」值的判斷，不可以使用判斷一般資料的條件設定：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_38.png)

**欄名 IS NULL 同 欄名 <=> NULL 不能使用 欄名 = NULL**

註：「<=>」在判斷一般資料的時候，跟「=」完全一樣；不過它用在判斷「NULL」資料的時候，效果跟「IS」一樣。

如果換成要查詢「有」平均壽命資料的國家，也就是平均壽命的欄位值不是「NULL」：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_39.png)

**欄名 IS NOT NULL 不能使用 欄名 <> NULL**

**3.5 字串樣式**

在使用字串資料的條件判斷時，會有一種很常見、也比較特殊的需求，像是「想要查詢名稱以w字元開始的城市」，如果你使用下列的查詢敘述：

|  |  |
| --- | --- |
| 1 | SELECT Name FROM city WHERE Name = 'w' |

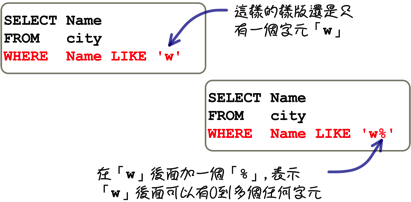
這樣的查詢條件，當然不是「名稱以w字元開始的城市」，而是名稱只有一個「w」字元的城市。所以這類的查詢就會使用下列這個特殊的條件設定：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_40.png)

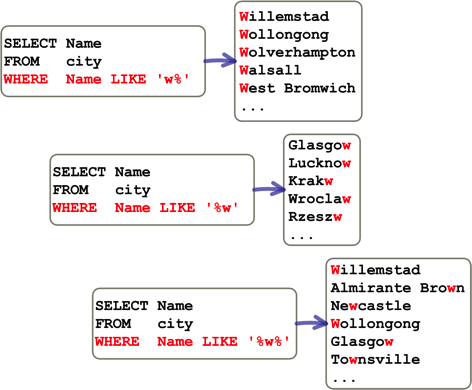
上列語法中，在「LIKE」後面的「樣版」字串中，會使用到下列兩種「樣版字元」：

* %：0到多個任何字元
* \_ ：一個任何字元

所以要查詢「名稱以w字元開始的城市」的話：

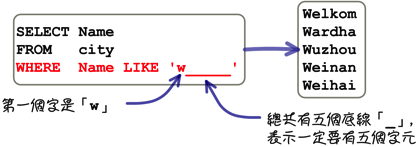
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_41.png)

參考上列的作法，就可以延伸出其它的查詢條件設定了：

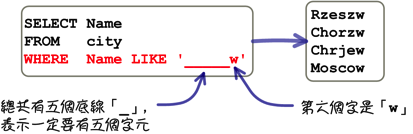
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_42.png)

上列的查詢條件中，「w%」表示第一個字元是「w」就符合條件；「%w」表示最後一個字元是「w」就符合條件；最後一個「%w%」表示不論在什麼位置有「w」字元，都符合條件。

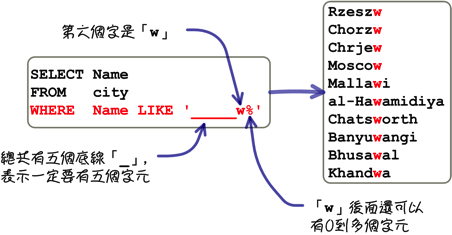
另外一種樣版字元「\_」表示一個任何字元：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_43.png)

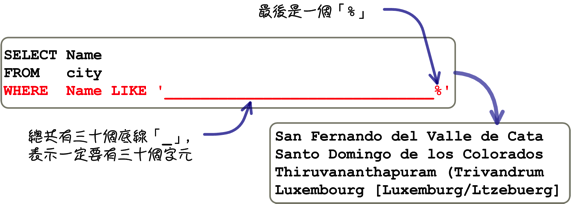
把這些樣版中的底線換到後面的話：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_44.png)

你也可以搭配兩種樣版字元完成條件的設定：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_45.png)

甚至像查詢「名稱是三十(包含)個字元以上的城市」：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_46.png)

註：其實完成上列的查詢條件的需求是不用這麼麻煩的，在後面的章節會討論比較簡單的方式；參考如下：

SELECT name FROM city WHERE LENGTH(name) >= 20;

SELECT name FROM city WHERE name LIKE CONCAT(REPEAT('\_', 20), '%');

SET @pattern := REPEAT('\_', 20);

SELECT name FROM city WHERE name LIKE CONCAT(@pattern, '%');

SELECT name FROM city WHERE name REGEXP '^.{20}';

^ ：表示開頭。這個正則必須從字串的第一個字元開始比對。

. ：表示任意一個字元（除了換行）。

{20}：表示前面的項目（.）重複 20 次。

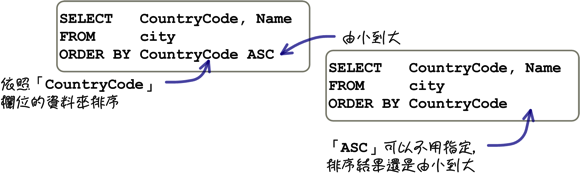
**4 排序**

在你執行任何一個查詢以後，MySQL傳回的資料是依照「自然」的順序排列的。所謂的自然順序，通常是資料新增到表格中的順序，可是在資料庫運作一段時間後，陸續會有各種不同的操作，所以這個「自然」順序對你來說，通常是沒什麼意義的。

一般的查詢通常會有資料排序上的需求，所以你會使用「ORDER BY」子句：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_49.png)

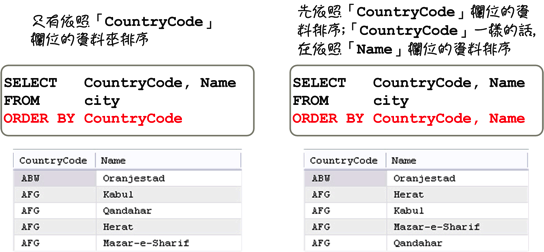
如果你希望在查詢城市資料的時候，資料庫會依照國家代碼幫你排序的話：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_50.png)

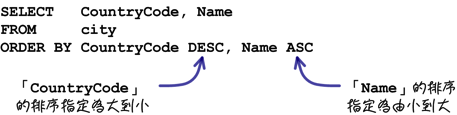
你也可以指定資料排列的順序為由大到小：

[mysql_03_snap_51](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_51.png)

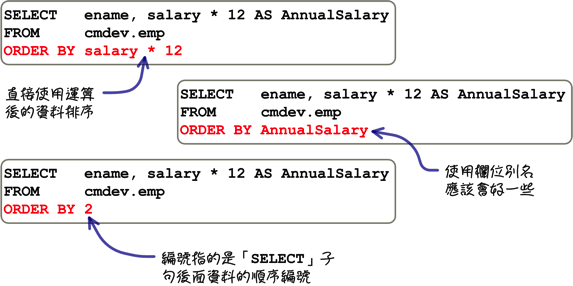
「ORDER BY」子句後面可以依照需求指定多個排序的資料：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_52.png)

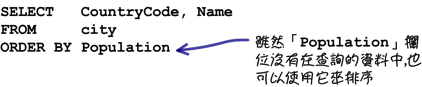
「ORDER BY」子句後面指定多個排序資料的時候，都可以依照需求，各自指定資料排列的方式：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_53.png)

「ORDER BY」子句指定的資料可以是欄位名稱、編號、運算式或是欄位別名：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_54.png)

雖然比較不會有下列這樣的需求，不過你還是可以這樣作：

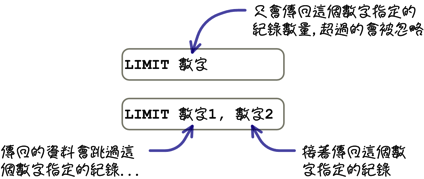
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_55.png)

註：資料排列的順序在「第六章、字元集與資料庫」與「第七章、儲存引擎與資料型態」中進一步詳細的討論。

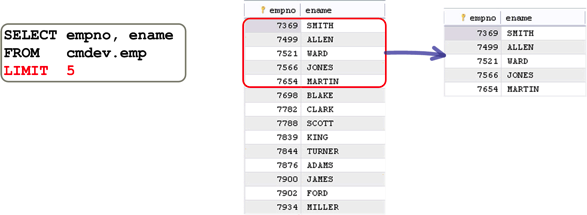
**5 限制查詢**

**5.1 指定回傳紀錄數量**

在你執行一個查詢敘述後，資料庫會將你查詢的資料傳回來給你；如果你使用「WHERE」子句設定查詢條件的話，資料庫就只會傳回符合條件的資料；除了上列的狀況外，你也可以另外使用「LIMIT」子句指定回傳紀錄的數量：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_56.png)

如果你在「LIMIT」子句後面指定一個數字：

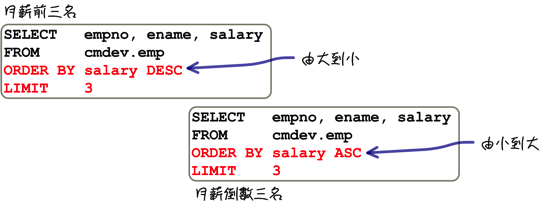
[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_57.png)

「LIMIT」子句後面也可以指定兩個數字：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_58.png)

**※跳過 5筆，開始顯示 5筆**

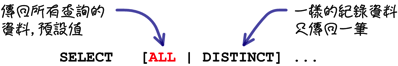
在查詢敘述中，使用「ORDER BY」子句搭配「LIMIT」子句，就可以完成下列查詢「排名」的工作：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_59.png)

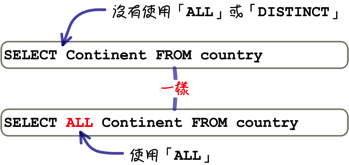
註：如果出現類似「… LIMIT 1000000, 10」這樣的查詢敘述，雖然你只會得到十筆資料，資料庫總共會查詢一百萬零一十筆資料，只不過資料庫會幫你跳過前一百萬筆；類似這樣的需求，還是要使用「WHERE」子句先挑出想要的資料會比較好一些。

**5.2 排除重複紀錄**

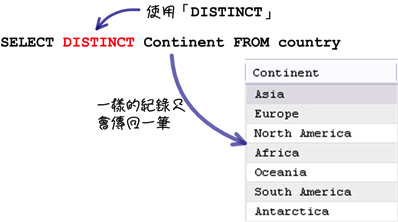
在一個查詢敘述執行以後，資料庫不會幫你檢查回傳的資料是否重複(回傳的兩筆紀錄資料完全一樣)，在「SELECT」子句後面可以讓你設定「回傳的資料是否重複」：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_60.png)

沒有使用「ALL」或「DISTINCT」的效果，跟你自己加上「ALL」的查詢效果是一樣的，資料庫會依照你的查詢傳回所有的資料：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_61.png)

使用「DISTINCT」的話，資料庫會特別執行回傳紀錄是否重複的檢查：

[](https://www.codedata.com.tw/wp-content/uploads/2013/12/mysql_03_snap_62.png)