資料庫管理

實習課:董臻 助教

博碩文化出版發行

博碩文化

第7章 資料操作DML–查詢(SELECT)

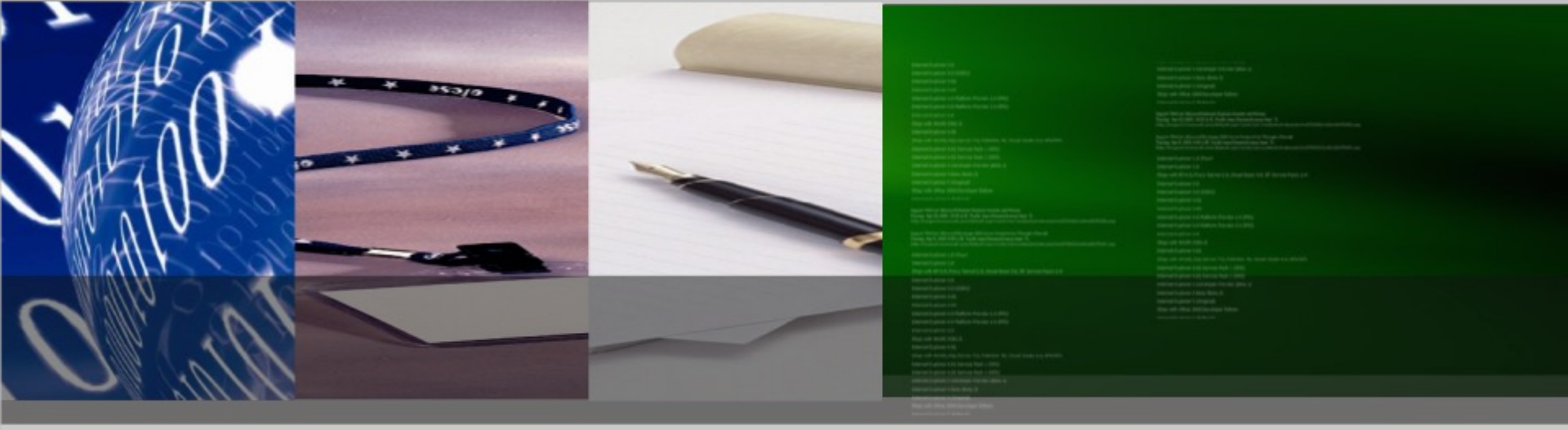
課前指引

結構化查詢語言(Query Structured Language, 簡稱SQL)主要分為三大類，第一類是定義資料的『**資料定義語言**』(Data Definition Language, 簡稱DDL)，包括建立、刪除以及維護資料庫、資料表、檢視表…等等。第二類是資料存取的『**資料操作語言**』(Data Manipulation Language, 簡稱DML)，包括針對資料的新增(INSERT)、刪除(DELETE)、修改(UPDATE)以及本章的重點查詢(SELECT)。第三類主要是針對安全管理的『**資料控制語言**』(Data Control Language, 簡稱DCL)，包括授權、撤銷…等等。

章節大綱

- 7-1 SQL查詢的環境介紹
- 7-2 常用函數的使用
- 7-3 單一資料表的查詢
- 7-4 多個資料表的查詢
- 7-5 不同的條件篩選方式
- 7-6 彙總函數與GROUP BY…HAVING...
- 7-7 子查詢(Sub-Query)
- 7-8 SELECT語法與語意之剖析整理

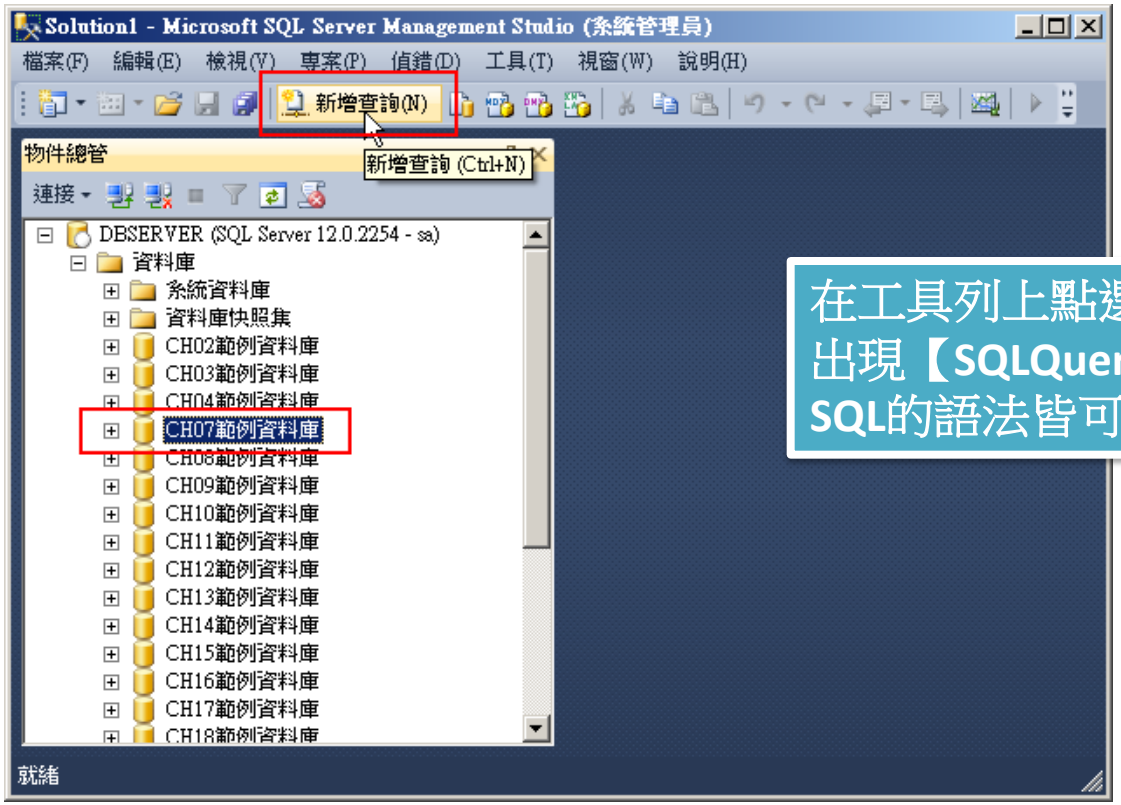
備註：可依進度點選小節



7-1 SQL查詢的環境介紹

7-1 SQL查詢的環境介紹

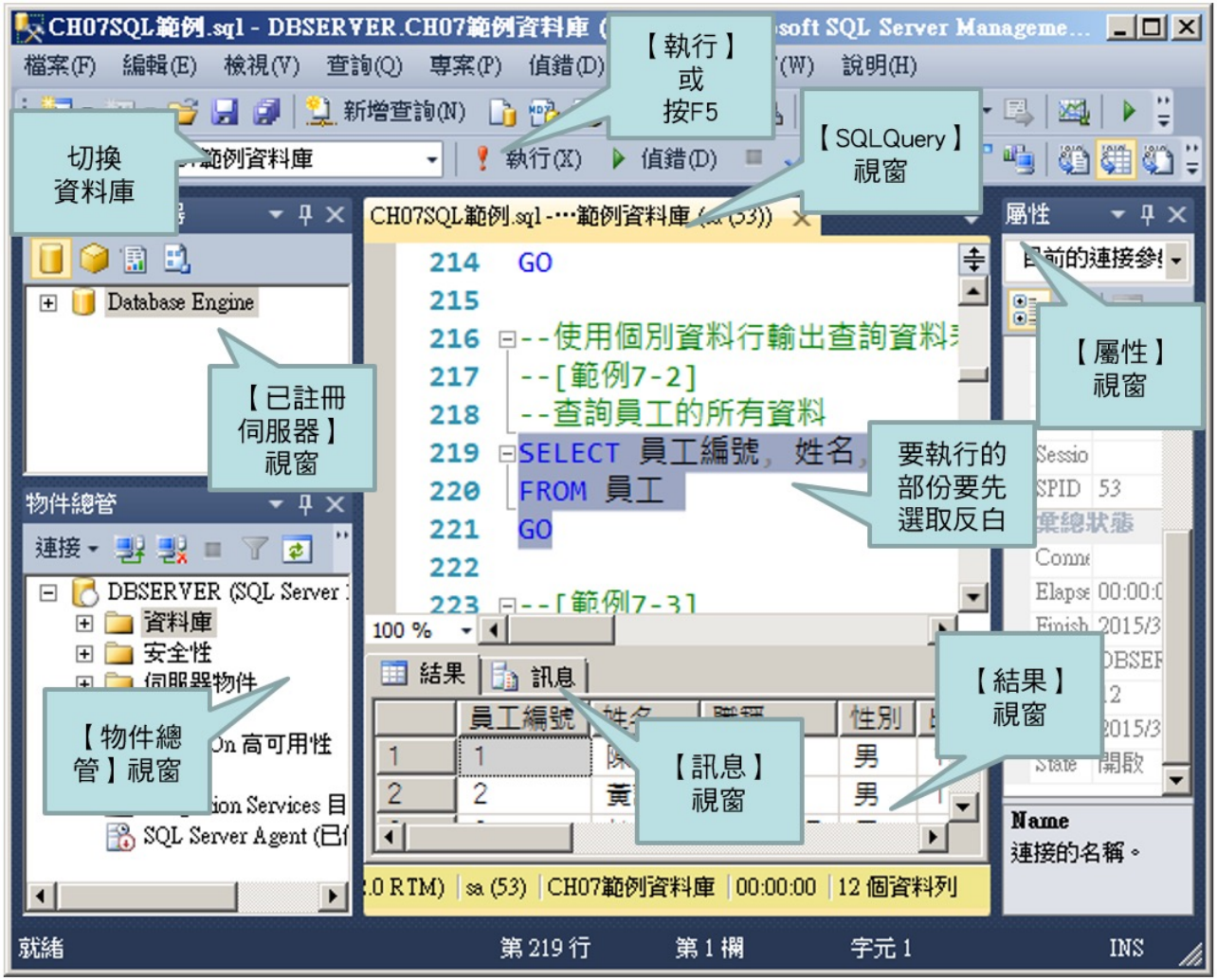
- MS SQL Server連線到伺服器管理的Microsoft SQL Server Management Studio



在工具列上點選【新增查詢(N)】，便會出現【SQLQuery】頁面視窗，所有下達SQL的語法皆可在此執行。

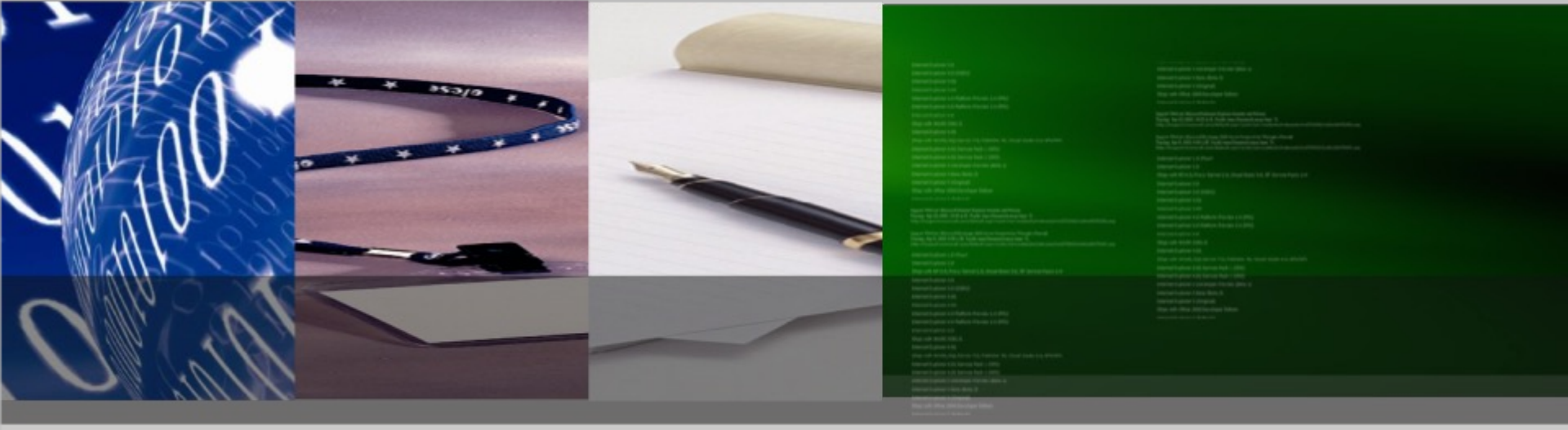
7-1 SQL查詢的環境介紹

- SQL Server Management Studio 的查詢環境



7-1 SQL查詢的環境介紹

利用【檢視(V)】功能表下的功能選項來顯示或關閉各個視窗



7-2 常用函數的使用

7-2 常用函數的使用

- 基本輸出：SELECT expression
  - 輸出字串

```
SELECT '資料庫系統'
結果：資料庫系統
```

  - 輸出運算式

```
SELECT 3+5
結果：8
```

  - 同時輸出多個項目
  - 在SELECT後面要同時輸出多個項目時，必須使用逗號『,』隔開

```
SELECT 3+5, '資料庫系統'
結果：8, 資料庫系統
```

7-2 常用函數的使用

常用的數學函數

四捨五入函數

ROUND (*numeric_expression* , *length*)

引數：

✧*numeric_expression* 可以是精確數值或是近似數值的資料類型之運算式，但不可為bit資料類型。

✧*length* 表示 *numeric_expression* 捨入的有效位數。

- *length* 是正數時，*numeric_expression* 會捨入到 *length* 所指定，在小數點右側的位數。
- *length* 是負數時，*numeric_expression* 會依照 *length* 所指定，在小數點左側捨入。

7-2 常用函數的使用

- 四捨五入函數：ROUND(numeric_expression , length)
  - 四捨五入至小數點以下二位：

```
SELECT ROUND( 84.94, 2 )
結果：84.94
```

  - 四捨五入至小數點以下一位：

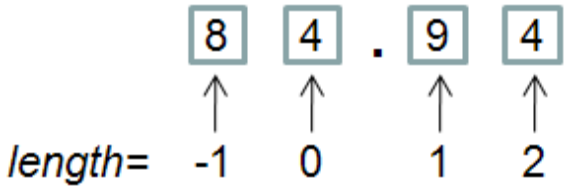
```
SELECT ROUND( 84.94, 1 )
結果：84.90
```

  - 四捨五入至整數第一位：

```
SELECT ROUND( 84.94, 0 )
結果：85.00
```

  - 四捨五入至整數第二位：

```
SELECT ROUND( 84.94, -1 )
結果：80.00
```



7-2 常用函數的使用

常用的數學函數

天花板(CEILING)與地板(FLOOR)函數

CEILING(numeric_expression)，傳回大於或等於指定數值運算式的最小整數。
FLOOR(numeric_expression)，傳回小於或等於指定數值運算式的最大整數。

引數：
✧numeric_expression ，精確數值或近似數值資料型態的數值運算式，但不可為bit資料類型。

7-2 常用函數的使用

- 天花板(CEILING)函數：CEILING(numeric_expression)
  - 天花板函數，小數以下無條件進位至整數
  - 天花板函數，小數以下無條件進位至整數

```
SELECT CEILING( 59.3 )
結果：60
```

```
SELECT CEILING( 59.8 )
結果：60
```

- 地板(FLOOR)函數：FLOOR(numeric_expression)
  - 地板函數，整數以下無條件捨去
  - 地板函數，整數以下無條件捨去

```
SELECT FLOOR( 59.3 )
結果：59
```

```
SELECT FLOOR( 59.8 )
結果：59
```

7-2 常用函數的使用

常用的數學函數

開平方根(SQRT)與平方(SQUARE)函數

SQRT(float_expression)，傳回指定浮點值的平方根。
SQUARE(float_expression)，傳回指定浮點值的平方。

引數：
✧float_expression， float 型態或能夠隱含地轉換成float型態之運算式。

7-2 常用函數的使用

- 開平方根(SQRT)函數：SQRT(float_expression)

```
SELECT SQRT( 16 )
結果：4
SELECT SQRT( 16.556 )
結果：4.06890648700606
```

- 平方(SQUARE)函數：SQUARE(float_expression)

```
SELECT SQUARE( 3 )
結果：9
SELECT SQUARE( 11.25 )
結果：126.5625
```

7-2 常用函數的使用

常用的轉換函數

資料型態轉換函數
(1) CAST(expression AS data_type)
(2) CONVERT(data_type , expression)

引數：
- ✧ expression，任何有效的運算式。
- ✧ data_type，欲轉換後的新資料型態，包括 xml、bigint 和 sql_variant。

7-2 常用函數的使用

- 資料型態轉換函數：
  - CAST(expression AS data_type)
  - CONVERT(data_type , expression)

- 將數值90轉換成變動長度的字串

```
SELECT CAST( 90 AS VARCHAR )
等同於
SELECT CONVERT ( VARCHAR, 90 )
結果：90    /* 此為變動長度的字串  */
```

7-2 常用函數的使用

常用的轉換函數

轉換成具有固定有效位數和小數位數的數值資料類型的函數

DECIMAL(p, [s])
NUMERIC(p, [s])

decimal與numeric的功能相同。引數包含了固定有效位數和小數位數的數字。有效值介於 - 10^38 +1 到 10^38 - 1。

引數：

✧p (有效位數)，最大位數之總數，包括小數點左右兩側的位數在內。有效位數必須是 1 至最大有效位數 38 之間的值。預設有效位數是 18。

✧s (小數位數)，小數點右側的最大位數。小數位數必須是從 0 到 p 的值。只有在指定了有效位數時，才能指定小數位數。預設小數位數是 0。

7-2 常用函數的使用

- 具有固定有效位數和小數位數的數值資料類型:

```
DECIMAL( p, [s] )
NUMERIC( p, [s] )
```

- 將數值先經過四捨五入後，再轉換成固定長度型態

```
SELECT CAST( ROUND( 87.994 , 0 ) AS NUMERIC( 2,0 ) )
等同於
SELECT CONVERT( NUMERIC( 2, 0 ) , ROUND( 87.994 , 0 ) )

結果：88
```

- 取固定長度，總長度為3，小數以下1位，取位前也會自動四捨五入

續

```
SELECT CAST( 87.994 AS NUMERIC( 3, 1 ) )
等同於
SELECT CONVERT( NUMERIC( 3, 1 ) , 87.994 )

結果：88.0
```

7-2 常用函數的使用

- 取固定長度，總長度為4，小數以下2位，取位前也會自動四捨五入：

```
SELECT CAST( 87.994 AS NUMERIC( 4, 2 ) )
等同於
SELECT CONVERT( NUMERIC( 4, 2 ) , 87.994 )

結果：87.99
```

- 解決ROUND(99.95, 1)位數不足所產生的溢位問題

```
SELECT ROUND( CAST( 99.95 AS NUMERIC( 5, 2 ) ), 1 )

結果：100.00
```

續

7-2 常用函數的使用

常用的日期函數

目前系統日期時間函數

GETDATE( )，取自正在執行 SQL Server 執行個體之電腦的系統日期時間。

取得日期的部份資訊

DATEPART(datepart , date)，取得一個日期中的某一部份。

引數：

✧datepart，這是指定傳回date的哪一部分。下表列出常用的datepart。

date的部份	縮寫
年(year)	yy , yyyy
季(quarter)	qq , q
月(month)	mm , m
日(day)	dd , d
一年中的第幾週(week)	wk , ww
一週中的第幾天(weekday)	dw
時(hour)	hh
分(minute)	mi , n
秒(second)	ss , s

✧date，可以是 time、date、smalldatetime、datetime、datetime2 或 datetimeoffset 值的運算式。

7-2 常用函數的使用

- 目前系統日期時間函數:GETDATE()

SELECT GETDATE( )
結果：2015-03-11 13:45:07.663
此結果會依執行當時的日期時間顯示

- 取得日期的部份資訊:DATEPART(datepart,date)

date的部份	縮寫
年(year)	yy , yyyy
季(quarter)	qq , q
月(month)	mm , m
日(day)	dd , d
一年中的第幾週(week)	wk , ww
一週中的第幾天(weekday)	dw
時(hour)	hh
分(minute)	mi , n
秒(second)	ss , s

續

7-2 常用函數的使用

- 僅查詢日期中之『年』的部份

```
SELECT DATEPART( year ,'2015-03-16' )
結果：2015
```

- 僅查詢日期中之『季』的部份

```
SELECT DATEPART( quarter , '2015-03-16' )
結果：1
```

- 僅查詢日期中之『月』的部份

```
SELECT DATEPART( month , '2015-03-16' )
結果：3
```

- 僅查詢日期中之『日』的部份

```
SELECT DATEPART( day , '2015-03-16')
結果：16
```

- 查詢該日期在一週中是第幾天

```
SELECT DATEPART( weekday , '2015-03-16' )
結果：2
```

7-2 常用函數的使用

常用的日期函數

日期累加函數

DATEADD(datepart, number,date)，並將指定的 number (帶正負號的整數) 加入至該 date 的指定datepart。

引數：

✧datepart，這是整數 number 要加入其中的 date 部分。可參考DATEPART()函數中的 datepart 引數之說明。

✧number，要累加的數量。

✧date，可以是 time、date、smalldatetime、datetime、datetime2 或 datetimeoffset 值的運算式。

7-2 常用函數的使用

- 日期累加函數：DATEADD(datepart ,number,date)
  - 將原本日期往前累加一年

```
將原本日期往前累加一年：
SELECT DATEADD( year , 1 , '2015-01-31' )
結果：2016-01-31 00:00:00.000
```

  - 將原本日期往前累加一個月

```
SELECT DATEADD( month , 1 , '2015-01-31' )
結果：注意結果是2月28日，可不是2月31日！
2015-02-28 00:00:00.000
```

  - 將原本日期往前累加一季

```
SELECT DATEADD( quarter , 1 , '2015-01-31' )
結果：2015-04-30 00:00:00.000
```

  - 將原本日期往回推算三個月前

```
SELECT DATEADD( month , -3 , '2015-01-31' )
結果：2014-10-31 00:00:00.000
```

7-2 常用函數的使用

常用的日期函數

日期差異函數

DATEDIFF(datepart , startdate , enddate)

引數：
✧datepart，指定startdate和enddate兩個日期的哪一部分來計算差距。可參考DATEPART()函數中的datepart引數之說明。
✧startdate，起始日期，可以是time、date、smalldatetime、datetime、datetime2或datetimeoffset值的運算式。從enddate中扣除startdate。
✧enddate，終止日期，請參考startdate的說明。

7-2 常用函數的使用

- 日期差異函數：DATEDIFF(datepart ,startdate ,enddate)
  - 計算兩個日期在『年』的部份之差異

```
SELECT DATEDIFF( yy, '2014/12/05', '2015/01/01' )
結果：1
```

  - 計算兩個日期在『年』的部份之差異

```
SELECT DATEDIFF( yy, '2015/12/05', '2013/01/01' )
結果：-2
```

  - 計算兩個日期在『月』的部份之差異

```
SELECT DATEDIFF( mm, '2014/01/01', '2015/05/05' )
結果：16
```

  - 計算兩個日期在『日』的部份之差異

```
SELECT DATEDIFF( dd, '2015/01/01', '2015/01/20' )
結果：19
```

  - 計算兩個日期在『季』的部份之差異

```
SELECT DATEDIFF( qq, '2013/01/01', '2015/05/05' )
結果：9
```

7-2 常用函數的使用

常用的字串函數

大、小寫轉換函數

UPPER(character_expression)，將小寫字元轉換成大寫字元的運算式

LOWER(character_expression) ，將大寫字元轉換成小寫字元的運算式

引數：

- ✧ character_expression，要轉換的字元資料之運算式。
- ✧ character_expression 可以是字元或二進位資料的常數、變數或資料行。
- ✧ character_expression 必須是可以隱含轉換成 varchar 的資料類型。否則，要利用CAST()或CONVERT()函數進行明確轉換character_expression。

```
SELECT UPPER( 'AbCdE' )
結果：ABCDE

SELECT LOWER( 'AbCdE' )
結果：abcde
```

7-2 常用函數的使用

常用的字串函數

計算字串長度函數
LEN(string_expression)，計算指定字串運算式的字元數，但尾端空白不算。

引數：

✧ string_expression，指要計算長度的字串運算式。可以是常數、變數或是字元或二進位資料的資料行。

計算字串長度函數:LEN(string_expression)

```
SELECT LEN('abcde' )
結果：5

SELECT LEN('我有幾個字呢?')
結果：7
```

7-2 常用函數的使用

常用的字串函數

取得一個子字串在另一個字串之起始位置的函數

CHARINDEX (expression1 ,expression2 [, start_location])，從start_location指定的位置開始搜尋，搜尋expression1 在 expression2中的開始位置。

引數：

✧expression1，這是字元運算式，expression1 限制最長為 8000 個字元。

✧expression2，這是要搜尋的字元運算式。

✧start_location，指定起始搜尋的位置。如果未指定此引數，或者它是負數或 0，搜尋就會從 expression2 的開始位置開始搜尋。

7-2 常用函數的使用

- 取得一個子字串在另一個字串起始位置CHARINDEX
  - 沒有指定搜尋的起始位置，便會從頭開始搜尋

```
SELECT CHARINDEX(' DEF', 'ABCDEFGHABCDEFGH')
結果：4
```

  - 有指定搜尋的起始位置，便會從指定的位置開始搜尋

```
SELECT CHARINDEX('DEF', 'ABCDEFGHABCDEFGH' , 5)
結果：12
```

  - 試試中文字

```
SELECT CHARINDEX('2朵', '妳我是天上的2朵雲')
結果：7
```

7-2 常用函數的使用

常用的字串函數

自一個字串之左、右邊開始取出子字串的函數：

LEFT (character_expression , integer_expression)，
從字元字串character_expression的最左邊開始，傳回指定字元個數integer_expression的字元字串。

RIGHT (character_expression , integer_expression)，
從字元字串character_expression的最右邊開始，傳回指定字元個數integer_expression的字元字串。

引數：

- ✧ character_expression，字元或二進位資料的運算式。
- ✧ integer_expression，必須為正整數，指定將傳回的 character_expression 字元數目。

7-2 常用函數的使用

- 取字串的左邊的子字串

  LEFT(character_expression ,integer_expression)

```
SELECT LEFT('ABCDEFG', 3 )
結果：ABC
```

- 取字串的右邊的子字串

  RIGHT(character_expression,integer_expression)

```
SELECT RIGHT('ABCDEFG' , 3 )
結果：EFG
```

7-2 常用函數的使用

常用的字串函數

自一個字串中取出子字串的函數：
SUBSTRING (value_expression , start_expression , length_expression)

引數：

- value_expression ，是 character、binary、text、ntext 或 image 運算式。
- start_expression，指定傳回字元的起始位置。
- length_expression，指定將傳回之 value_expression 的字元數。

從字串中的第2 個位置開始取出長度為3 個字元的子字串

```
SELECT SUBSTRING(' abcdefg' , 2 , 3 )
結果：bcd
```

從字串中的第4個位置開始取出長度為1個字元的子字串

```
SELECT SUBSTRING('日一二三四五六' , 4 , 1 )
結果：三
```

7-2 常用函數的使用

常用的字串函數

重複字串函數

REPLICATE (string_expression , integer_expression)，將字串值重複指定的次數。

引數：

✧string_expression，可以是字元字串或二進位資料型態的資料或運算式。

✧integer_expression，指定要重複輸出string_expression的個數。

重複某字串數次

```
SELECT REPLICATE('*' , 5 )
結果：*****
```

續

7-2 常用函數的使用

- 取當天的系統日期，並轉換成民國XX年XX月XX日 週X

```
SELECT '民國'
    +CAST(DATEPART(YY,GETDATE()) - 1911 AS VARCHAR)+'年'
    +CAST(DATEPART(MM,GETDATE()) AS VARCHAR)+'月'
    +CAST(DATEPART(DD,GETDATE()) AS VARCHAR)+'日  週'
    +SUBSTRING('日一二三四五六',DATEPART(DW,GETDATE()),1)
```

結果：(假設執行當天SQL Server電腦的系統日期是2015/3/11週三)
民國104年3月11日 週三

續

7-2 常用函數的使用

- 將輸出數字部份變該為固定長度2，例如 3月顯示成 03月

```
SELECT '民國'+CAST(DATEPART(YY,GETDATE()) - 1911 as VARCHAR)+'年'
    +REPLICATE('0', 2 - LEN(CAST(DATEPART(MM,GETDATE()) as VARCHAR)))
    +CAST(DATEPART(MM,GETDATE()) as VARCHAR)+'月'
    +REPLICATE('0', 2 - LEN(CAST(DATEPART(DD,GETDATE()) as VARCHAR)))
    +CAST(DATEPART(DD,GETDATE()) as VARCHAR)+'日  週'
    +SUBSTRING('日一二三四五六',DATEPART(DW,GETDATE()),1)
```

結果：(假設執行當天SQL Server電腦的系統日期是2015/3/11週三)
民國104年03月11日 週三

7-2 常用函數的使用

- 隨堂練習 請截圖
  - 輸出字串

    結果：資料庫管理系統

  - 輸出運算式

    結果：31 (9/3*5+16)

  - 同時輸出多個項目

    結果：47 (21/3*5+16-4),資料庫系統

7-2 常用函數的使用

- 隨堂練習 請截圖
  - 四捨五入至小數點以下二位：

    結果：85.47

  - 四捨五入至小數點以下一位：

    結果：85.50 (85.47)

  - 四捨五入至整數第二位：

    結果：90.00 (85.47)

  - 天花板函數，小數以下無條件進位至整數

    結果：70 (69.7)

  - 地板函數，整數以下無條件捨去

    結果：60 (60.8)

7-2 常用函數的使用

- 隨堂練習 請截圖
  - 僅查詢日期中之『年』的部份

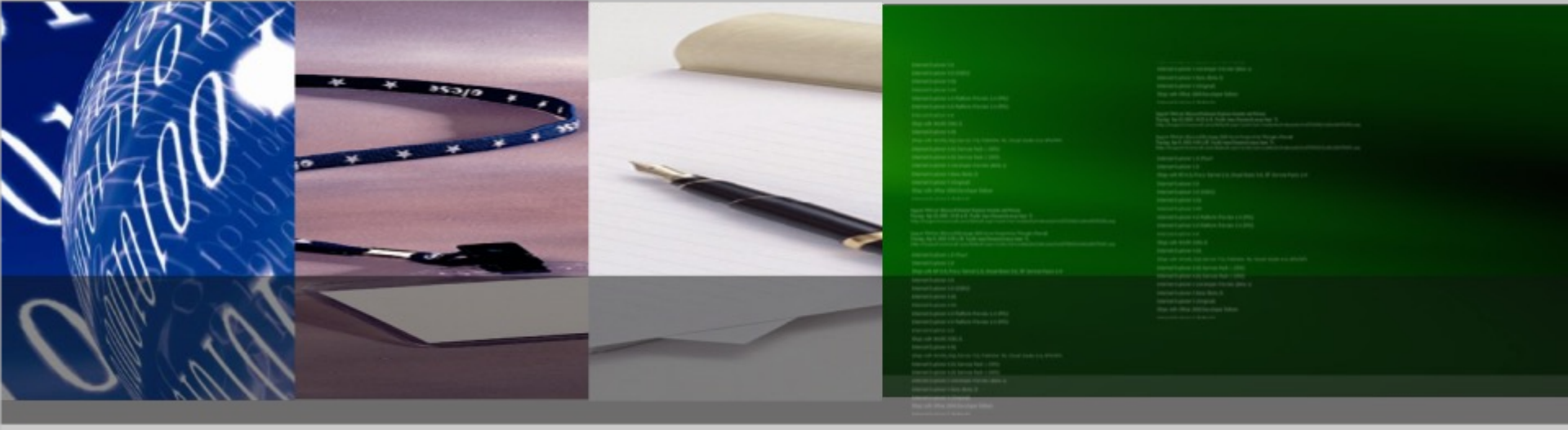
    結果：2022 (2022-11-15)
  - 僅查詢日期中之『月』的部份

    結果：11 (2022-11-15)
  - 僅查詢日期中之『日』的部份

    結果：15 (2022-11-15)
  - 有指定搜尋的起始位置，便會從指定的位置開始搜尋

    結果：13 ('EF', 'ABCDEFGHABCDEFGH' , 5)
  - 重複某字串數次

    結果：******



7-3 單一資料表的查詢

7-3 單一資料表的查詢

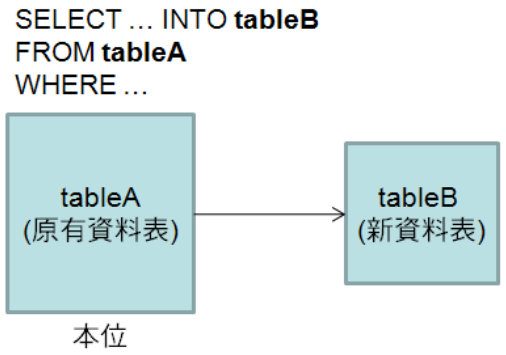
SECLECT基本和常用的語法

```
SELECT select_list [ INTO new_table ]
FROM table_source
[ WHERE search_condition ]
[ GROUP BY group_by_expression ]
[ HAVING search_condition ]
[ ORDER BY order_expression [ ASC | DESC] ]
```

慣例	說　明
大寫	Transact-SQL 關鍵字
斜體字	由使用者提供的 Transact-SQL 語法參數
| (分隔號)	加上括號或大括號來分隔語法項目，其中只可以選擇一個項目
[ ] (中括號)	選擇性的語法項目，但不要輸入中括號

7-3 單一資料表的查詢

- SELECT select_list [INTO new_table]
  - select_list，可以是以下三種情形之一
    - 萬用字元『*』
    - 資料行名稱(column_name)
    - 運算式(expression)
  - new_table
    - 是指一個不存在的資料表
    - 透過SELECT查詢的資料寫入其中



- FROM table_source
  - 一個或多個『資料表』(table)或『檢視』(view)
- [WHERE search_condition]
  - 『資料行』的條件限制

7-3 單一資料表的查詢

- [GROUP BY group_by_expression]
  - 根據一個或多個資料行或是運算式的值，當成一個群組
- [HAVING search_condition]
  - 經過彙總函數計算後的結果
- [ORDER BY order_expression [ASC | DESC]]
  - 針對查詢的資料進行排序的動作
    - ASC，代表遞增 (ASCending) 排序
    - DESC，代表遞減 (DESCending) 排序

7-3 單一資料表的查詢

- 使用萬用字元『*』查詢資料表 – FROM
  - [範例7-1]查詢員工的所有資料

```
SELECT *
FROM 員工
```

員工編號	姓名	職稱	性別	主管	出生日期	任用日期	區域號碼	地址	分機號碼
1	陳祥輝	總經理	男	NULL	1965-07-15	1992-11-13	114	台北市內湖區康寧 路23巷	1888
2	黃謙仁	工程師	男	4	1969-03-22	1992-11-26	407	台中市西屯區工業11路	3087
3	林其達	工程助理	男	2	1971-06-06	1992-12-06	235	台北縣中和市大勇街25巷	2138
4	陳森耀	工程協理	男	1	1968-11-14	1993-01-14	106	台北市大安區忠孝東路4段	3085
5	徐沛汶	業務助理	女	12	1963-09-30	1993-03-16	330	桃園縣桃園市縣府路	2234
6	劉逸萍	業務	女	10	1958-09-15	1993-05-23	111	台北市士林區士東路	2230
7	陳臆如	業務協理	女	1	1987-04-03	1993-09-24	114	台北市內湖區瑞光路513巷	2247
8	胡琪偉	業務	男	10	1963-08-12	1993-10-17	220	台北縣板橋市中山路一段	2238
9	吳志梁	業務	男	10	1960-05-19	1994-07-02	406	台中市北屯區太原路3段	2236
10	林美滿	業務經理	女	7	1958-02-09	1994-08-27	104	台北市中山區 一江街	2344
11	劉嘉雯	業務	女	10	1968-02-07	1994-11-05	111	台北市士林區福志路	2234
12	張懷甫	業務經理	男	7	1952-09-16	1994-12-26	106	台北市大安區仁愛路四段	2342

7-3 單一資料表的查詢

- 使用個別資料行輸出查詢資料表 – FROM
  - [範例7-2]查詢員工的所有資料

```
SELECT 員工編號, 姓名, 職稱, 性別, 出生日期, 任用日期, 區域號碼, 地址, 分機號碼, 主管
FROM 員工
```

員工編號	姓名	職稱	性別	出生日期	任用日期	區域號碼	地址	分機號碼	主管
1	陳祥輝	總經理	男	1965-07-15	1992-11-13	114	台北市內湖區康寧 路23巷	1888	NULL
2	黃謙仁	工程師	男	1969-03-22	1992-11-26	407	台中市西屯區工業11路	3087	4
3	林其達	工程助理	男	1971-06-06	1992-12-06	235	台北縣中和市大勇街25巷	2138	2
4	陳森耀	工程協理	男	1968-11-14	1993-01-14	106	台北市大安區忠孝東路4段	3085	1
5	徐沛汶	業務助理	女	1963-09-30	1993-03-16	330	桃園縣桃園市縣府路	2234	12
6	劉逸萍	業務	女	1958-09-15	1993-05-23	111	台北市士林區士東路	2230	10
7	陳臆如	業務協理	女	1987-04-03	1993-09-24	114	台北市內湖區瑞光路513巷	2247	1
8	胡琪偉	業務	男	1963-08-12	1993-10-17	220	台北縣板橋市中山路一段	2238	10
9	吳志梁	業務	男	1960-05-19	1994-07-02	406	台中市北屯區太原路3段	2236	10
10	林美滿	業務經理	女	1958-02-09	1994-08-27	104	台北市中山區 一江街	2344	7
11	劉嘉雯	業務	女	1968-02-07	1994-11-05	111	台北市士林區福志路	2234	10
12	張懷甫	業務經理	男	1952-09-16	1994-12-26	106	台北市大安區仁愛路四段	2342	7

7-3 單一資料表的查詢

- 重複值僅輸出一筆記錄 – DISTINCT
  - [範例7-3]透過『員工』資料表查詢該公司有多少種不同的職務。
    - 相同職稱會重複輸出

```
SELECT 職稱
FROM 員工
```

    - 相同職稱僅會出現一筆

```
SELECT DISTINCT 職稱
FROM 員工
```

    - 輸出兩個資料行時，會以兩個資料行的組合來判斷重複性

```
SELECT DISTINCT 職稱, 姓名
FROM 員工
```

	職稱
1	總經理
2	工程師
3	工程助理
4	工程協理
5	業務助理
6	業務
7	業務協理
8	業務
9	業務
10	業務經理
11	業務
12	業務經理

相同『職稱』會重複輸出

	職稱
1	工程助理
2	工程協理
3	工程師
4	業務
5	業務助理
6	業務協理
7	業務經理
8	總經理

使用DISTINCT後相同『職稱』僅會出現一筆

	職稱	姓名
1	工程助理	林其達
2	工程協理	陳森耀
3	工程師	黃謙仁
4	業務	吳志梁
5	業務	胡琪偉
6	業務	劉逸萍
7	業務	劉嘉雯
8	業務助理	徐沛汶
9	業務協理	陳臆如
10	業務經理	林美滿
11	業務經理	張懷甫
12	總經理	陳祥輝

是區分(職稱+姓名)之值不是僅區分『職稱』

7-3 單一資料表的查詢

- 資料的基本條件篩選 – WHERE
  - [範例7-4]查詢職稱為業務的員工
  - [輸出](員工編號, 姓名, 職稱)

```
SELECT 員工編號, 姓名, 職稱
FROM 員工
WHERE 職稱 = '業務'
```

員工編號	姓名	職稱
6	劉逸萍	業務
8	胡琪偉	業務
9	吳志梁	業務
11	劉嘉雯	業務

7-3 單一資料表的查詢

- [範例7-5]查詢公司有哪些男業務
- [輸出](員工編號, 姓名, 職稱, 性別)
- [提示]男業務代表所要篩選的條件為，『性別』等於'男'，且『職稱』等於'業務'

```
SELECT 員工編號, 姓名, 職稱, 性別
FROM 員工
WHERE 職稱 = '業務' AND 性別 = '男'
```

員工編號	姓名	職稱	性別
8	胡琪偉	業務	男
9	吳志梁	業務	男

7-3 單一資料表的查詢

- [範例7-6]查詢男工程師和女業務的基本資料
- [輸出](員工編號, 姓名, 職稱, 性別)
- [提示]男工程師和女業務代表所要篩選的條件為：『性別』等於'男'且『職稱』等於'工程師')或(『性別』等於'女'且『職稱』等於'業務'

```
SELECT 員工編號, 姓名, 職稱, 性別
FROM 員工
WHERE ( 性別= '男' AND 職稱 = '工程師' ) OR
            ( 性別= '女' AND 職稱 = '業務')
```

員工編號	姓名	職稱	性別
2	黃謙仁	工程師	男
6	劉逸萍	業務	女
11	劉嘉雯	業務	女

7-3 單一資料表的查詢

- 資料行與函數的結合運算
  - [範例7-7]查詢當月生日的員工資料
  - [輸出](員工編號, 姓名, 出生日期)
  - [提示]必須使用getdate()函數取得當天日期，再使用datepart()或month()函數來取得出生日期與當天的月份來比對

```
SELECT 員工編號, 姓名, 出生日期
FROM 員工
WHERE datepart( month, 出生日期 ) = datepart( month,getdate() )
```

員工編號	姓名	出生日期
2	黃謙仁	1969-03-22

(假設執行當時的資料庫伺服器電腦之系統日期為 3 月)

7-3 單一資料表的查詢

- 資料行的別名
  - 利用『別名』的方式將輸出的資料行重新命名。在資料行後面加上『AS 別名 』。
  - [範例7-8]列出每一位員工的主管之員工編號，將員工的『姓名』更名為『員工姓名』、『主管』更名為『上司編號』。
  - [輸出] (員工編號, 員工姓名, 上司編號)

```
SELECT 員工編號, 姓名 AS 員工姓名, 主管 AS 上司編號
FROM 員工

省略『AS』的寫法

SELECT 員工編號, 姓名 員工姓名, 主管 上司編號
FROM 員工
```

員工編號	員工姓名	上司編號
1	陳祥輝	NULL
2	黃謙仁	4
3	林其達	2
4	陳森耀	1
5	徐沛汶	12
6	劉逸萍	10
7	陳臆如	1
8	胡琪偉	10
9	吳志梁	10
10	林美滿	7
11	劉嘉雯	10
12	張懷甫	7

7-3 單一資料表的查詢

- 衍生資料行的計算與輸出
  - 原本並不存在於資料表或檢視表內的資料行，它是透過運算式所產生的資料行。
  - [範例7-9]查詢每位員工的『年齡』
  - [輸出] (員工編號, 姓名, 出生日期, 年齡)
  - [提示] 年齡 = 當日的年份 - 出生年份

```
SELECT 員工編號, 姓名, 出生日期, datediff( year, 出生日期,  getdate() ) AS 年齡
FROM 員工
```

假設執行當時的資料庫伺服器電腦之系統日期為 2015 年

員工編號	姓名	出生日期	年齡
1	陳祥輝	1965-07-15	50
2	黃謙仁	1969-03-22	46
3	林其達	1971-06-06	44
4	陳森耀	1968-11-14	47
5	徐沛汶	1963-09-30	52
6	劉逸萍	1958-09-15	57
7	陳臆如	1987-04-03	28
8	胡琪偉	1963-08-12	52
9	吳志梁	1960-05-19	55
10	林美滿	1958-02-09	57
11	劉嘉雯	1968-02-07	47
12	張懷甫	1952-09-16	63

7-3 單一資料表的查詢

- [範例7-10]查詢從任用日期計算起，超過16年(含)年資的員工，預計在25年後退休的資料。
- [輸出](員工編號, 姓名, 任用日期, 預計退休日)
- [提示1]預計退休日=任用日期+25*12個月(25年,每年12個月)
- [提示2]篩選條件：任用日期至當月超過(含)16*12個月(16年,每年12個月)

```
SELECT 員工編號, 姓名, 任用日期, dateadd (month,25*12,任用日期) AS 預計退休日
FROM 員工
WHERE datediff( month, 任用日期, getdate() ) >= 16*12
```

員工編號	姓名	任用日期	預計退休日
1	陳祥輝	1992-11-13	2017-11-13
2	黃謙仁	1992-11-26	2017-11-26
3	林其達	1992-12-06	2017-12-06
4	陳森耀	1993-01-14	2018-01-14
5	徐沛汶	1993-03-16	2018-03-16
6	劉逸萍	1993-05-23	2018-05-23
8	胡琪偉	1993-10-17	2018-10-17
9	吳志梁	1994-07-02	2019-07-02
10	林美滿	1994-08-27	2019-08-27
12	張懷甫	1994-12-26	2019-12-26

假設執行當時的資料庫伺服器電腦之系統日期為 **2015 年 3 月**

7-3 單一資料表的查詢

- 資料排序 Order By …[ASC | DESC]
  - 輸出的資料可以依據資料表或檢視表中的資料行來排列，可分為『遞增』排序和『遞減』排序。
  - [範例7-11]請列出所有員工資料，並依據單一『任用日期』資料行的遞減排序。
  - [輸出](員工編號, 姓名, 職稱, 任用日期)

```
SELECT 員工編號, 姓名, 職稱, 任用日期
FROM 員工
ORDER BY 任用日期 DESC
```

	員工編號	姓名	職稱	任用日期
1	7	陳臆如	業務協理	2009-08-01
2	11	劉嘉雯	業務	2005-11-05
3	12	張懷甫	業務經理	1994-12-26
4	10	林美滿	業務經理	1994-08-27
5	9	吳志梁	業務	1994-07-02
6	8	胡琪偉	業務	1993-10-17
7	6	劉逸萍	業務	1993-05-23
8	5	徐沛汶	業務助理	1993-03-16
9	4	陳森耀	工程協理	1993-01-14
10	3	林其達	工程助理	1992-12-06
11	2	黃謙仁	工程師	1992-11-26
12	1	陳祥輝	總經理	1992-11-13

7-3 單一資料表的查詢

- [範例7-12]請列出所有員工資料，並依據『職稱』與『員工編號』多個資料行的排序，其中『職稱』資料行遞減排序，『員工編號』資料行遞增排序。
- [輸出] (員工編號, 姓名, 職稱, 任用日期)

```
SELECT 員工編號, 姓名, 職稱, 任用日期
FROM 員工
ORDER BY 職稱 DESC, 員工編號 ASC
```

	員工編號	姓名	職稱	任用日期
1	1	陳祥輝	總經理	1992-11-13
2	10	林美滿	業務經理	1994-08-27
3	12	張懷甫	業務經理	1994-12-26
4	7	陳臆如	業務協理	2009-08-01
5	5	徐沛汶	業務助理	1993-03-16
6	6	劉逸萍	業務	1993-05-23
7	8	胡琪偉	業務	1993-10-17
8	9	吳志梁	業務	1994-07-02
9	11	劉嘉雯	業務	2005-11-05
10	2	黃謙仁	工程師	1992-11-26
11	4	陳森耀	工程協理	1993-01-14
12	3	林其達	工程助理	1992-12-06

7-3 單一資料表的查詢

- 輸出前 <n> 筆(或百分比)資料 – TOP <n>
  - 可以僅查詢前<n>筆資料，亦或是前面的<n>百分比的資料
  - 使用TOP敘述，通常會搭配ORDER BY的排序，再取排序後的前<n>筆(或百分比)的資料較有其意義。
  - 倘若是想將相同值也都同時輸出時，可以在TOP <n> (PERCENT)後面加上 WITH TIES。

續

7-3 單一資料表的查詢

- [範例7-13]試將員工先依據『職稱』遞增排序後，再取其前5筆資料。也試著比較加入WITH TIES後的結果
- [輸出](員工編號, 姓名, 職稱)

```
SELECT TOP 5 員工編號, 姓名, 職稱
FROM 員工
ORDER BY 職稱

SELECT TOP 5 WITH TIES 員工編號, 姓名, 職稱
FROM 員工
ORDER BY 職稱
```

	員工編號	姓名	職稱
1	3	林其達	工程助理
2	4	陳森耀	工程協理
3	2	黃謙仁	工程師
4	6	劉逸萍	業務
5	8	胡琪偉	業務

不使用WITH TIES

	員工編號	姓名	職稱
1	3	林其達	工程助理
2	4	陳森耀	工程協理
3	2	黃謙仁	工程師
4	6	劉逸萍	業務
5	8	胡琪偉	業務
6	9	吳志梁	業務
7	11	劉嘉雯	業務

使用WITH TIES

7-3 單一資料表的查詢

- [範例7-14]試將員工先依據『職稱』遞增排序後，再取其前50%的資料。也試著比較加入WITH TIES後的結果。
- [輸出](員工編號, 姓名, 職稱)

```
SELECT TOP 50 PERCENT 員工編號, 姓名, 職稱
FROM 員工
ORDER BY 職稱

SELECT TOP 50 PERCENT WITH TIES 員工編號, 姓名,職稱
FROM 員工
ORDER BY 職稱
```

	員工編號	姓名	職稱
1	3	林其達	工程助理
2	4	陳森耀	工程協理
3	2	黃謙仁	工程師
4	6	劉逸萍	業務
5	8	胡琪偉	業務
6	9	吳志梁	業務

不使用WITH TIES

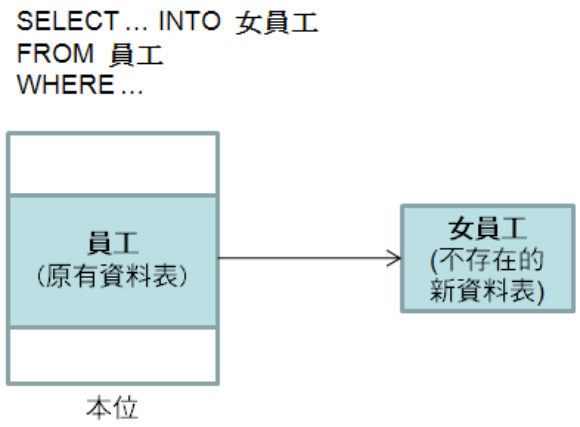
	員工編號	姓名	職稱
1	3	林其達	工程助理
2	4	陳森耀	工程協理
3	2	黃謙仁	工程師
4	6	劉逸萍	業務
5	8	胡琪偉	業務
6	9	吳志梁	業務
7	11	劉嘉雯	業務

使用WITH TIES

7-3 單一資料表的查詢

- 複製資料與結構到另一個新的資料表
- SELECT ... INTO ...
  - SELECT亦可將查詢出來的結果，透過INTO來轉入另一個不存在的新資料表。
  - [範例7-15]從員工資料表中挑選出女性員工，並輸出至另一個新的資料表。
  - [輸出] (員工編號, 姓名, 職稱, 性別)

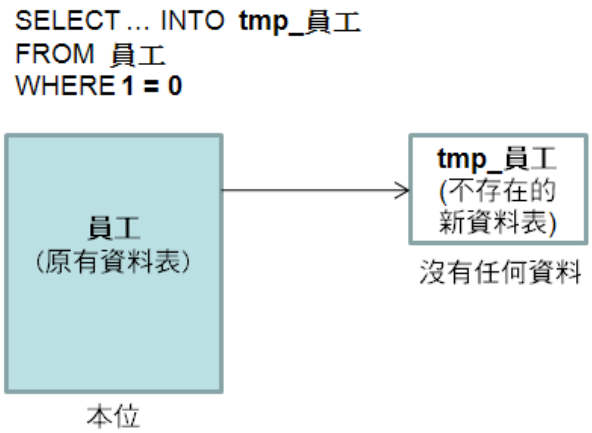
```
SELECT 員工編號, 姓名, 職稱, 性別 INTO 女員工
FROM 員工
WHERE 性別='女'
```



7-3 單一資料表的查詢

- 複製結構到另一個新的資料表
- SELECT .. INTO ..WHERE 1=0
  - 利用WHERE 1=0的矛盾方式，複製結構到另一個新的資料表。
  - [範例7-16]將員工資料表中(員工編號, 姓名, 職稱, 性別)的結構，複製到另一個新的資料表 『tmp_員工』，並將『姓名』資料行更名為『員工姓名』。

```
SELECT 員工編號, 姓名 AS 員工姓名, 職稱, 性別 INTO tmp_員工
FROM 員工
WHERE 1=0
```



7-3 單一資料表的查詢

- 次序函數(Ranking Functions)與資料表的結合使用
  - 次序函數會依據每一個分割(partition)，將每一個分割內的資料大小依序傳回1,2,3…的排名值。

依順序回傳序號，序號是不會重複的
ROW_NUMBER () OVER ([PARTITION BY value_expression , ... [n]]order_by_clause)

傳回類型：bigint

例如：有十筆資料排序後將會依序回傳：
1, 2, 3, 4, 5, 6, 7, 8, 9, 10

續

7-3 單一資料表的查詢

- [範例7-17]請依據員工的員工編號遞增排序，給予每一位員工一個序號。
- 【輸出】(員工編號, 姓名, 性別, 序號)

```
SELECT 員工編號, 姓名, 性別
        , ROW_NUMBER() OVER (ORDER BY 員工編號 ASC) 序號
FROM 員工
```

員工編號	姓名	性別	序號
1	陳祥輝	男	1
2	黃謙仁	男	2
3	林其達	男	3
4	陳森耀	男	4
5	徐沛汶	女	5
6	劉逸萍	女	6
7	陳臆如	女	7
8	胡琪偉	男	8
9	吳志梁	男	9
10	林美滿	女	10
11	劉嘉雯	女	11
12	張懷甫	男	12

7-3 單一資料表的查詢

- [範例7-18]請依據員工性別分別給定每一位員工一個序號，再依據員工編號來作遞增排序。
- 【輸出】(員工編號, 姓名, 性別, 序號)

```
SELECT 員工編號, 姓名, 性別
        , ROW_NUMBER () OVER (PARTITION BY 性別 ORDER BY 員工編號 ASC) 序號
FROM 員工
```

員工編號	姓名	性別	序號
5	徐沛汶	女	1
6	劉逸萍	女	2
7	陳臆如	女	3
10	林美滿	女	4
11	劉嘉雯	女	5
1	陳祥輝	男	1
2	黃謙仁	男	2
3	林其達	男	3
4	陳森耀	男	4
8	胡琪偉	男	5
9	吳志梁	男	6
12	張懷甫	男	7

續

7-3 單一資料表的查詢

依順序回傳名次，相同資料會有相同名次
RANK () OVER ([partition_by_clause] order_by_clause)
例如：十筆資料排序後可能會有名次重複的情形，後續名次將會跳掉
1, 2, 2, 2, 5, 5, 7, 8, 8, 10

DENSE_RANK () OVER ([<partition_by_clause>] < order_by_clause >)
例如：有十筆資料排序後可能會有名次重複的情形，後續名次會緊密排列
1, 2, 2, 2, 3, 3, 4, 5, 5, 6

傳回類型：bigint

7-3 單一資料表的查詢

- [範例7-19]請依據員工的年齡排名，年齡較長者排名較前面，依序遞增。
- 【輸出】(員工編號, 姓名, 性別, 排名1, 排名2)
- 【說明】排名1為RANK(), 排名2為DENSE_RANK()

```
SELECT 員工編號, 姓名, 性別
        , YEAR(GETDATE())-YEAR(出生日期) 年齡
        , RANK() OVER (ORDER BY YEAR(GETDATE())-YEAR(出生日期) DESC) 排名1
        , DENSE_RANK() OVER (ORDER BY YEAR(GETDATE())-YEAR(出生日期) DESC) 排名2
FROM 員工
```

	員工編號	姓名	性別	年齡	排名1	排名2
1	12	張懷甫	男	63	1	1
2	10	林美滿	女	57	2	2
3	6	劉逸萍	女	57	2	2
4	9	吳志梁	男	55	4	3
5	8	胡琪偉	男	52	5	4
6	5	徐沛汶	女	52	5	4
7	1	陳祥輝	男	50	7	5
8	4	陳森耀	男	47	8	6
9	11	劉嘉雯	女	47	8	6
10	2	黃謙仁	男	46	10	7
11	3	林其達	男	44	11	8
12	7	陳臆如	女	28	12	9

7-3 單一資料表的查詢

- [範例7-20]請依據員工性別分別排名，排名依據是年齡較長者排名較前面，依序遞增。
- 【輸出】(員工編號, 姓名, 性別, 排名1, 排名2)
- 【說明】排名1是指RANK(), 排名2是指DENSE_RANK()

```
SELECT 員工編號, 姓名, 性別
        , YEAR(GETDATE())-YEAR(出生日期) 年齡
        , RANK() OVER (PARTITION BY 性別 ORDER BY YEAR(GETDATE())-YEAR(出生日期) DESC) 排名1
        , DENSE_RANK() OVER (PARTITION BY 性別 ORDER BY YEAR(GETDATE())-YEAR(出生日期) DESC) 排名2
FROM 員工
```

可將所有資料分成N群，每群回傳編號1, 2, 3, …

NTILE (integer_expression) OVER ([<partition_by_clause>] < order_by_clause >)

傳回類型：bigint

例如：有十筆資料要分成五群, 將會回傳1, 1, 1, 2, 2, 2, 3, 3, 4, 4

	員工編號	姓名	性別	年齡	排名1	排名2
1	6	劉逸萍	女	57	1	1
2	10	林美滿	女	57	1	1
3	5	徐沛汶	女	52	3	2
4	11	劉嘉雯	女	47	4	3
5	7	陳膳如	女	28	5	4
6	12	張懷甫	男	63	1	1
7	9	吳志梁	男	55	2	2
8	8	胡琪偉	男	52	3	3
9	1	陳祥輝	男	50	4	4
10	4	陳森耀	男	47	5	5
11	2	黃謙仁	男	46	6	6
12	3	林其達	男	44	7	7

7-3 單一資料表的查詢

- [範例7-21]請依據員工的員工編號遞增排序後，分成五個群體。
- 【輸出】(員工編號, 姓名, 性別, 群組編號)

```
SELECT 員工編號, 姓名, 性別
        , NTILE(5) OVER (ORDER BY 員工編號 ASC) 群組編號
FROM 員工
```

員工編號	姓名	性別	群組編號
1	陳祥輝	男	1
2	黃謙仁	男	1
3	林其達	男	1
4	陳森耀	男	2
5	徐沛汶	女	2
6	劉逸萍	女	2
7	陳臆如	女	3
8	胡琪偉	男	3
9	吳志梁	男	4
10	林美滿	女	4
11	劉嘉雯	女	5
12	張懷甫	男	5

7-3 單一資料表的查詢

- [範例7-22]請依據員工性別各自分群，每個性別皆依員工編號遞增排序，分成三個群體。
- 【輸出】(員工編號, 姓名, 性別, 排名1, 排名2)

```
SELECT 員工編號, 姓名, 性別
        , NTILE(3) OVER (PARTITION BY 性別 ORDER BY 員工編號 ASC) 群組編號
FROM 員工
```

員工編號	姓名	性別	群組編號
5	徐沛汶	女	1
6	劉逸萍	女	1
7	陳臆如	女	2
10	林美滿	女	2
11	劉嘉雯	女	3
1	陳祥輝	男	1
2	黃謙仁	男	1
3	林其達	男	1
4	陳森耀	男	2
8	胡琪偉	男	2
9	吳志梁	男	3
12	張懷甫	男	3

7-3 單一資料表的查詢

- 邏輯函數(Logical Functions)與資料表的結合使用
  - 邏輯函數是MS SQL SERVER 2012版本新增的函數，可以透過邏輯判斷式來回傳不同的值，包括CHOOSE()以及IIF()兩個函數
  - 依據索引值來回傳值
  - CHOOSE(index, val_1, val_2 [, val_n])
  - [範例7-23]請查詢2005年訂單的相關資料。
  - 【輸出】(訂單編號, 訂貨日期, 季別)

訂單編號	訂貨日期	季別
94010104	2005-01-10	第一季
94010105	2005-01-11	第一季
94010201	2005-03-12	第一季
94010202	2005-05-12	第二季
94010301	2005-07-03	第三季
94010302	2005-08-03	第三季
94010303	2005-09-03	第三季
94010401	2005-11-04	第四季
94010501	2005-12-15	第四季
94010601	2005-12-16	第四季

```
SELECT 訂單編號, 訂貨日期
        , CHOOSE(DATEPART(QQ, 訂貨日期),'第一季','第二季','第三季','第四季') 季別
FROM 訂單
WHERE YEAR(訂貨日期)=2005
```

7-3 單一資料表的查詢

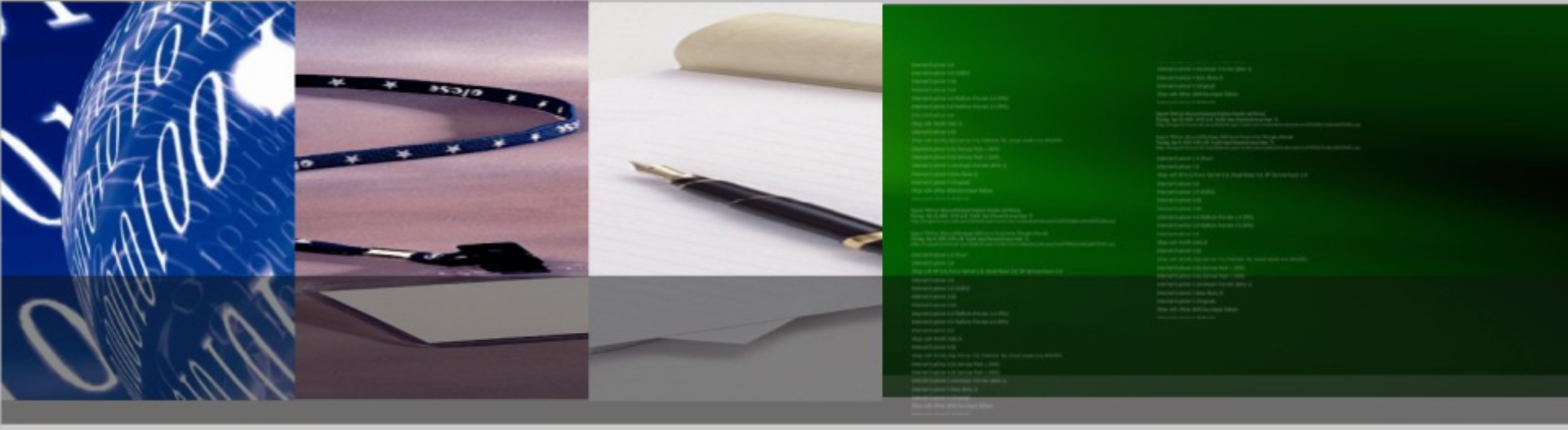
依據布林值來回傳值
依據boolean_expression的條件判斷，倘若bolean_expression判斷為true，則回傳後續的第一個值，否則就回傳後續的第二個值：

IIF (boolean_expression, true_value, false_value)

- [範例7-24]查詢員工的基本資料並增加一個稱謂，男生就稱為某先生，女生就稱為某小姐。
- 【輸出】(員工編號, 姓名, 性別, 稱謂)

```
SELECT 員工編號, 姓名
          , LEFT(姓名, 1) + IIF( 性別='男' , '先生' , '小姐') 稱謂
FROM 員工
```

員工編號	姓名	稱謂
1	陳祥輝	陳先生
2	黃謙仁	黃先生
3	林其達	林先生
4	陳森耀	陳先生
5	徐沛汶	徐小姐
6	劉逸萍	劉小姐
7	陳臆如	陳小姐
8	胡琪偉	胡先生
9	吳志梁	吳先生
10	林美滿	林小姐
11	劉嘉雯	劉小姐
12	張懷甫	張先生



7-4 多個資料表的查詢

7-4 多個資料表的查詢

- SELECT語法的分解動作解析
  - Step1.找出需要的『資料表』(以資料表1, 資料表2, 資料表3為例)，此步驟的結果等同於『交叉合併』(CROSS JOIN)

```
SELECT *
FROM 資料表1, 資料表2, 資料表3
```

  - Step2.進行『合併』(以資料表1與2合併為例)

```
SELECT *
FROM 資料表1, 資料表2, 資料表3
WHERE ( 資料表1 與 資料表2 的關聯性 )
   AND ( 資料表2 與 資料表3 的關聯性 )
```

7-4 多個資料表的查詢

- Step3.加入『篩選』條件

```
SELECT *
FROM資料表1, 資料表2, 資料表3
WHERE ( ( 資料表1 與 資料表2 的關聯性 )
   AND   ( 資料表2 與 資料表3 的關聯性 ) )
   AND ( 條件1 AND 條件 2 )
```

- Step4.填入要輸出的『資料行』

```
SELECT 資料行1, 資料行2, ...
FROM 資料表1, 資料表2, 資料表3
WHERE ( ( 資料表1 與 資料表2 的關聯性 )
  AND   ( 資料表2 與 資料表3 的關聯性 ) )
  AND  ( 條件1 AND 條件 2 )
```

7-4 多個資料表的查詢

- Step5.群組與彙總函數』計算

```
SELECT 資料行1, 資料行2, 彙總函數1, 彙總函數2
FROM 資料表1, 資料表2, 資料表3
WHERE ( ( 資料表1 與 資料表2 的關聯性 )
  AND   ( 資料表2 與 資料表3 的關聯性 ) )
  AND   ( 條件1 AND 條件 2 )
GROUP BY 資料行1, 資料行2
```

- Step6.彙總函數後的結果篩選

```
SELECT 資料行1, 資料行2, 彙總函數1, 彙總函數2
FROM 資料表1, 資料表2, 資料表3
WHERE ( ( 資料表1 與 資料表2 的關聯性 )
  AND   ( 資料表2 與 資料表3 的關聯性 ) )
  AND   ( 條件1 AND 條件 2 )
GROUP BY 資料行1, 資料行2
HAVING 彙總函數的條件篩選
```

7-4 多個資料表的查詢

- Step7.資料行的『排序』(以資料行1 遞增, 彙總函數2遞減為例)

```
SELECT 資料行1, 資料行2, 彙總函數1, 彙總函數2
FROM 資料表1, 資料表2, 資料表3
WHERE ( ( 資料表1 與 資料表2 的關聯性 )
  AND   ( 資料表2 與 資料表3 的關聯性 ) )
  AND   ( 條件1 AND 條件 2 )
GROUP BY 資料行1, 資料行2
HAVING 彙總函數的條件篩選
ORDER BY 資料行1 ASC, 彙總函數2 DESC
```

- Step8.再加入其他不同的需求。

7-4 多個資料表的查詢

- 『內部合併』的基本方法
  - [範例7-25]查詢有承接訂單的男業務資料，依員工編號遞增排序，訂單編號遞減排序
  - [輸出] (員工編號, 姓名, 性別, 職稱, 訂單編號, 訂貨日期, 產品編號, 數量)
  - [提示] 此範例主要是進行『員工』與『訂單』基本的『內部合併』

```
步驟1：找出需要的資料表，此步驟的結果就是『交叉合併』
SELECT *
FROM 員工, 訂單, 訂單明細

步驟2：進行『內部合併』
SELECT *
FROM 員工, 訂單, 訂單明細
WHERE 員工.員工編號 = 訂單.員工編號
  AND   訂單.訂單編號 = 訂單明細.訂單編號
```

續

7-4 多個資料表的查詢

步驟3：加入『篩選』條件，性別是男生且職稱是業務

```
SELECT *
FROM 員工, 訂單, 訂單明細
WHERE (員工.員工編號 = 訂單.員工編號
  AND    訂單.訂單編號 = 訂單明細.訂單編號 )
  AND   (性別 = '男'  AND 職稱 = '業務')
```

步驟4：填入要輸出的『資料行』

```
SELECT 員工.員工編號, 姓名, 性別, 職稱, 訂單.訂單編號
          , 訂貨日期, 產品編號, 數量
FROM 員工, 訂單, 訂單明細
WHERE (員工.員工編號 = 訂單.員工編號
  AND    訂單.訂單編號 = 訂單明細.訂單編號 )
  AND   (性別 = '男'  AND 職稱 = '業務')
```

續

7-4 多個資料表的查詢

步驟5：『群組與彙總函數』計算
此需求沒必要此步驟

步驟6：彙總函數後的結果篩選
此需求沒必要此步驟

步驟7：資料行的『排序』

```
SELECT 員工.員工編號, 姓名, 性別, 職稱, 訂單.訂單編號, 訂貨日期, 產品編號, 數量
FROM 員工, 訂單, 訂單明細
WHERE ( 員工.員工編號 = 訂單.員工編號
  AND     訂單.訂單編號 = 訂單明細.訂單編號 )
  AND   ( 性別 = '男' AND 職稱 = '業務' )
ORDER BY 員工.員工編號 ASC, 訂單.訂單編號 DESC
```

	員工編號	姓名	性別	職稱	訂單編號	訂貨日期	產品編號	數量
1	8	胡琪偉	男	業務	94010601	2005-12-16	1	50
2	8	胡琪偉	男	業務	94010601	2005-12-16	2	10
3	8	胡琪偉	男	業務	94010301	2005-07-03	6	20
4	8	胡琪偉	男	業務	94010301	2005-07-03	12	22
5	9	吳志梁	男	業務	94010701	2006-01-27	10	13

7-4 多個資料表的查詢

- 資料表的『別名』
  - [範例7-26]查詢出 2005 年第 4 季，有承接訂單的員工與其訂單相關資料，並依據員工編號及訂單編號遞增排序
  - [輸出] (員工編號, 姓名, 訂單編號, 訂貨日期, 產品名稱, 數量)

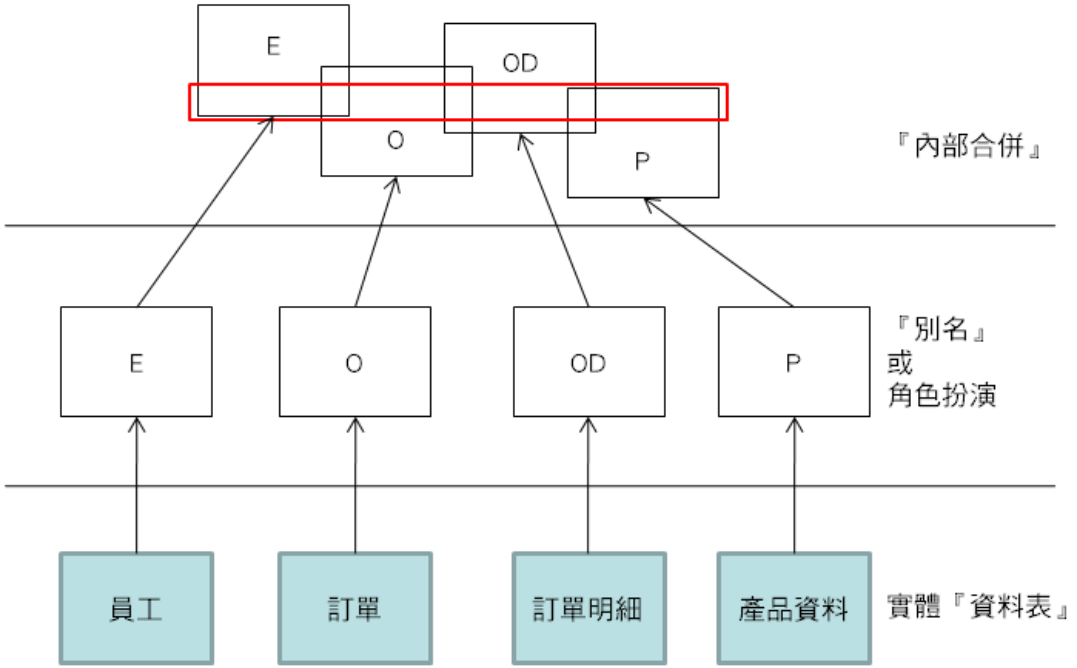
```
SELECT E.員工編號, 姓名, O.訂單編號, 訂貨日期, 產品名稱,數量
FROM 員工 AS E, 訂單 AS O, 訂單明細 OD, 產品資料 P
WHERE ( E.員工編號 = O.員工編號
  AND     O.訂單編號 = OD.訂單編號
  AND     OD.產品編號 = P.產品編號 )
  AND   ( datepart( year, 訂貨日期) = 2005 AND datepart( quarter, 訂貨日期 ) = 4)
ORDER BY E.員工編號 ASC, O.訂單編號 ASC
```

員工編號	姓名	訂單編號	訂貨日期	產品名稱	數量
7	陳臆如	94010401	2005-11-04	汽水	9
7	陳臆如	94010401	2005-11-04	運動飲料	6
7	陳臆如	94010501	2005-12-15	汽水	9
8	胡琪偉	94010601	2005-12-16	蘋果汁	50
8	胡琪偉	94010601	2005-12-16	蔬果汁	10

續

7-4 多個資料表的查詢

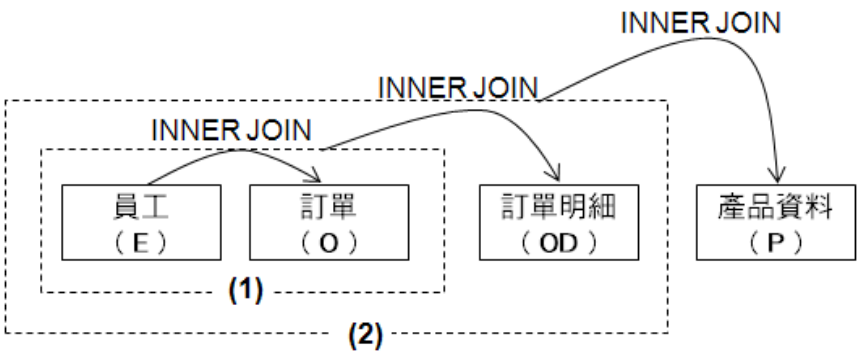
- 此例必須使用到4 個實體資料表來做內部合併處理。
- 包括『員工』、『訂單』、『訂單明細』以及『產品資料』，分別為『E』、『O』、『OD』以及『P』，再將此四個新『別名』進行『內部合併』處理後，篩選出所要的資料。



7-4 多個資料表的查詢

- 合併的另一種語法 – JOIN ... ON ...
  - [範例7-27]題目如同[範例7-26]，將SELECT語法改用JOIN...ON... 的方式寫出。

```
SELECT E.員工編號, 姓名, O.訂單編號, 訂貨日期, 產品名稱, 數量
FROM 員工 AS E
        INNER JOIN 訂單 AS O ON E.員工編號 = O.員工編號
        INNER JOIN 訂單明細 AS OD ON O.訂單編號 = OD.訂單編號
        INNER JOIN 產品資料 AS P ON OD.產品編號 = P.產品編號
WHERE datepart( year, 訂貨日期) = 2005
    AND datepart( quarter, 訂貨日期 ) = 4
ORDER BY E.員工編號 ASC, O.訂單編號 ASC
```

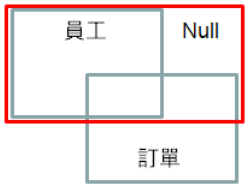


7-4 多個資料表的查詢

- 外部合併
  - [範例7-28]查詢每一位員工所承接的訂單資料，縱使沒有承接訂單的員工也要列出，並依據員工編號遞增排序
  - [輸出](員工編號, 姓名, 訂單編號, 訂貨日期)

員工編號	姓名	訂單編號	訂貨日期
1	陳祥輝	NULL	NULL
2	黃謙仁	NULL	NULL
3	林其達	NULL	NULL
4	陳森耀	NULL	NULL
5	徐沛汶	NULL	NULL
6	劉逸萍	94010202	2005-05-12
6	劉逸萍	94010705	2006-02-27
6	劉逸萍	94010801	2006-04-18
6	劉逸萍	94010806	2006-11-08
7	陳臆如	94010104	2005-01-10
7	陳臆如	94010401	2005-11-04
7	陳臆如	94010501	2005-12-15
7	陳臆如	94010804	2006-06-20
7	陳臆如	94010805	2006-09-20
8	胡琪偉	94010301	2005-07-03
8	胡琪偉	94010601	2005-12-16
9	吳志梁	94010701	2006-01-27
10	林美滿	94010105	2005-01-11
10	林美滿	94010201	2005-03-12
10	林美滿	94010302	2005-08-03
10	林美滿	94010303	2005-09-03
10	林美滿	94010702	2006-02-27
10	林美滿	94010803	2006-05-20
11	劉嘉雯	NULL	NULL
12	張懷甫	NULL	NULL

```
SELECT 員工.員工編號, 姓名, 訂單編號,訂貨日期
FROM 員工 LEFT OUTER JOIN 訂單 ON 員工.員工編號 = 訂單.員工編號
ORDER BY 員工.員工編號
```



7-4 多個資料表的查詢

- [範例7-29]查詢出2005年第4季，所有員工與其訂單相關資料，並依據員工編號及訂單編號遞增排序，無承接訂單之員工也要全部列出。
- [輸出](工編號,姓名,訂單編號,訂貨日期,產品名稱,數量)

[錯誤寫法]

```
SELECT E.員工編號, 姓名, O.訂單編號, 訂貨日期, 產品名稱, 數量
FROM 員工 AS E
        LEFT OUTER JOIN 訂單 AS O ON E.員工編號 = O.員工編號
        LEFT OUTER JOIN 訂單明細 AS OD ON O.訂單編號 = OD.訂單編號
        LEFT OUTER JOIN 產品資料 AS P ON OD.產品編號 = P.產品編號
WHERE datepart( year, 訂貨日期 ) = 2005
    AND datepart( quarter, 訂貨日期 ) = 4
ORDER BY E.員工編號 ASC, O.訂單編號 ASC
```

[範例7-29]錯誤的查詢結果

	員工編號	姓名	訂單編號	訂貨日期	產品名稱	數量
1	7	陳臆如	94010401	2005-11-04	汽水	9
2	7	陳臆如	94010401	2005-11-04	運動飲料	6
3	7	陳臆如	94010501	2005-12-15	汽水	9
4	8	胡琪偉	94010601	2005-12-16	蘋果汁	50
5	8	胡琪偉	94010601	2005-12-16	蔬果汁	10

續

7-4 多個資料表的查詢

[正確寫法]

```
SELECT E.員工編號, 姓名, O.訂單編號, 訂貨日期, 產品名稱, 數量
FROM 員工 AS E
        LEFT OUTER JOIN ( SELECT *
                          FROM 訂單
                          WHERE datepart( year, 訂貨日期 ) = 2005
                              AND  datepart( quarter, 訂貨日期 ) = 4 )
                          AS O ON  E.員工編號 = O.員工編號
         LEFT OUTER JOIN 訂單明細 OD ON O.訂單編號 = OD.訂單編號
         LEFT OUTER JOIN 產品資料 P ON OD.產品編號 = P.產品編號
ORDER BY E.員工編號 ASC, O.訂單編號 ASC
```

[範例7-29]正確的查詢結果

員工編號	姓名	訂單編號	訂貨日期	產品名稱	數量
1	陳祥輝	NULL	NULL	NULL	NULL
2	黃謙仁	NULL	NULL	NULL	NULL
3	林其達	NULL	NULL	NULL	NULL
4	陳森耀	NULL	NULL	NULL	NULL
5	徐沛汶	NULL	NULL	NULL	NULL
6	劉逸萍	NULL	NULL	NULL	NULL
7	陳臆如	94010401	2005-11-04	汽水	9
7	陳臆如	94010401	2005-11-04	運動飲料	6
7	陳臆如	94010501	2005-12-15	汽水	9
8	胡琪偉	94010601	2005-12-16	蘋果汁	50
8	胡琪偉	94010601	2005-12-16	蔬果汁	10
9	吳志梁	NULL	NULL	NULL	NULL
10	林美滿	NULL	NULL	NULL	NULL
11	劉嘉雯	NULL	NULL	NULL	NULL
12	張懷甫	NULL	NULL	NULL	NULL

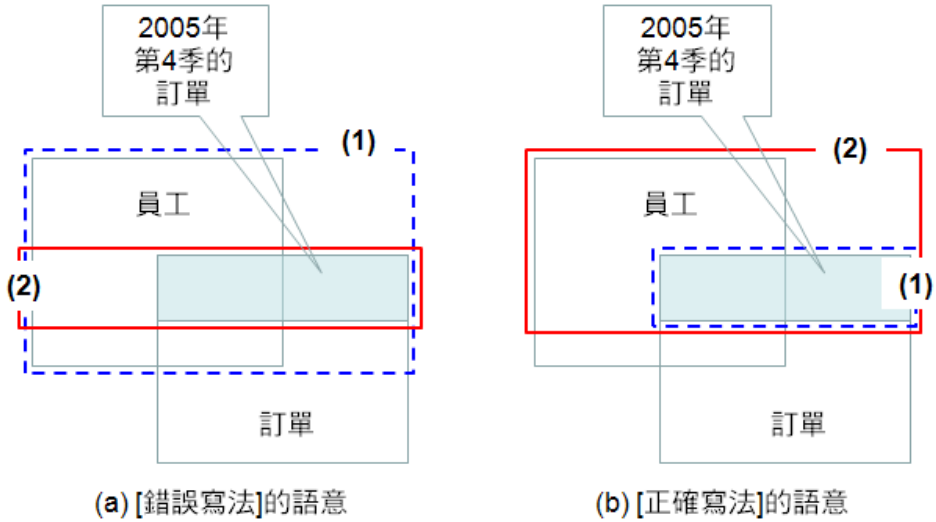
7-4 多個資料表的查詢

範例7-29 之錯誤SQL 敘述的解說

```
[錯誤寫法]
SELECT E.員工編號, 姓名, O.訂單編號, 訂貨日期, 產品名稱, 數量
FROM 員工 AS E
        LEFT OUTER JOIN 訂單 AS O ON E.員工編號 = O.員工編號
        LEFT OUTER JOIN 訂單明細 OD ON O.訂單編號 = OD.訂單編號
        LEFT OUTER JOIN 產品資料 P ON OD.產品編號 = P.產品編號
WHERE datepart( year, 訂貨日期 ) = 2005 AND datepart( quarter, 訂貨日期 ) = 4
ORDER BY E.員工編號 ASC, O.訂單編號 ASC
```

```
[正確寫法]
SELECT E.員工編號, 姓名, O.訂單編號, 訂貨日期, 產品名稱, 數量
FROM 員工 AS E
        LEFT OUTER JOIN ( SELECT *
                          FROM 訂單
                          WHERE datepart( year, 訂貨日期 ) = 2005 AND
                                datepart( quarter, 訂貨日期 ) = 4 )
                        AS O ON E.員工編號 = O.員工編號
        LEFT OUTER JOIN 訂單明細 OD ON O.訂單編號 = OD.訂單編號
        LEFT OUTER JOIN 產品資料 P ON OD.產品編號 = P.產品編號
ORDER BY E.員工編號 ASC, O.訂單編號 ASC
```

範例7-29正確及錯誤SQL 敘述的語意的比較



7-4 多個資料表的查詢

- 自我合併+內部合併
  - [範例7-30]試將所有業務和業務助理的上司資料列出。
  - [輸出](部屬編號, 部屬姓名, 上司編號, 上司姓名)

```
SELECT 部屬.員工編號 AS 部屬編號, 部屬.姓名 AS 部屬姓名
          , 上司.員工編號 AS 上司編號, 上司.姓名 AS 上司姓名
FROM 員工 AS 部屬, 員工 AS 上司
WHERE 部屬.主管 = 上司.員工編號
    AND  (部屬.職稱 = '業務' OR 部屬.職稱 = '業務助理')
```

	部屬編號	部屬姓名	上司編號	上司姓名
1	5	徐沛汶	12	張懷甫
2	6	劉逸萍	10	林美滿
3	8	胡琪偉	10	林美滿
4	9	吳志梁	10	林美滿
5	11	劉嘉雯	10	林美滿

7-4 多個資料表的查詢

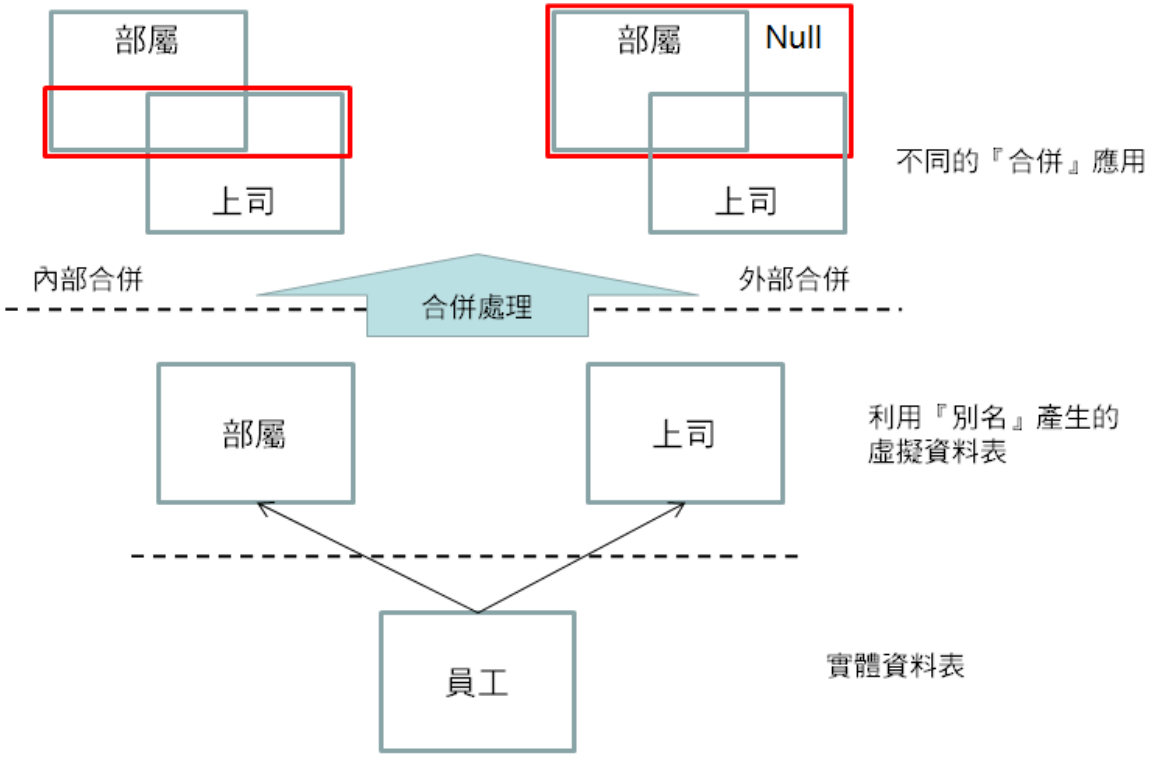
- 自我合併+外部合併
  - [範例7-31]查詢全公司員工與其上司的基本資料，縱使沒有上司也必須列出該筆資料。
  - [輸出](部屬編號,部屬姓名,上司編號,上司姓名)

```
SELECT 部屬.員工編號 AS 部屬編號, 部屬.姓名 AS 部屬姓名
        , 上司.員工編號 AS 上司編號, 上司.姓名 AS 上司姓名
FROM 員工 AS 部屬
        LEFT OUTER JOIN 員工 AS 上司 ON 部屬.主管 = 上司.員工編號
```

	部屬編號	部屬姓名	上司編號	上司姓名
1	1	陳祥輝	NULL	NULL
2	2	黃謙仁	4	陳森耀
3	3	林其達	2	黃謙仁
4	4	陳森耀	1	陳祥輝
5	5	徐沛汶	12	張懷甫
6	6	劉逸萍	10	林美滿
7	7	陳臆如	1	陳祥輝
8	8	胡琪偉	10	林美滿
9	9	吳志梁	10	林美滿
10	10	林美滿	7	陳臆如
11	11	劉嘉雯	10	林美滿
12	12	張懷甫	7	陳臆如

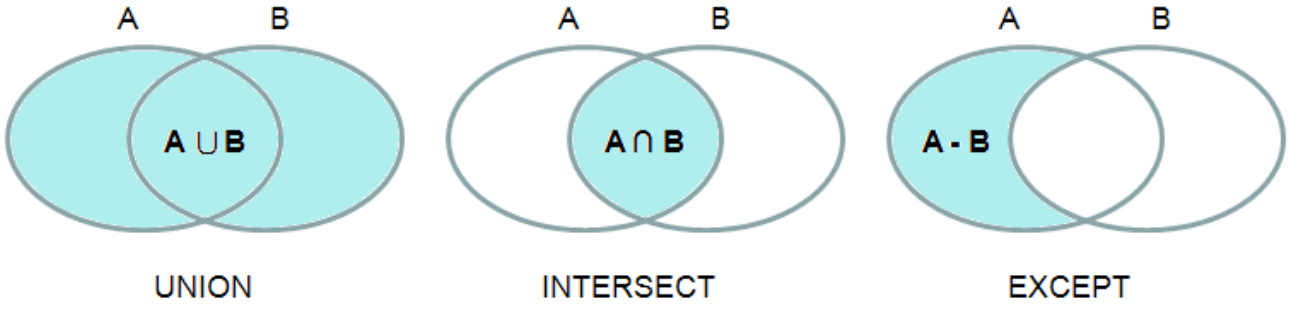
7-4 多個資料表的查詢

- 範例7-30、7-31皆是透過『自我合併』的基本方式，利用『別名』將一個實體的『員工』資料表，扮演成兩個不同的虛擬資料表『部屬』與『上司』。



7-4 多個資料表的查詢

- UNION(聯集)
  - A聯集B的結果，包括A與B兩者所有的元素，若有重複的元素只會出現一次。在SQL SERVER中倘若希望重複的元素也能重複出現，必須在UNION後面再加上ALL。
- INTERSECT(交集)
  - A交集B的結果，僅包括A與B兩者共同有的元素。
- EXCEPT(排除或差集)
  - A差集B的結果，會從A中扣除B也有的元素。反之，B差集A的結果，會從B中扣除A也有的元素。



7-4 多個資料表的查詢

- [範例7-32]將『客戶』與『供應商』的資料聯集，重複資料僅會出現一筆－使用UNION。
- [輸出] (公司名稱)

```
SELECT 公司名稱
FROM 客戶
UNION
SELECT 供應商名稱
FROM 供應商
```

	公司名稱
1	丁泉
2	五金行
3	心宥川公司
4	日盛金樓
5	日新日公司
6	正心
7	妙恩
8	宏詮工業
9	東信銀行
10	林木材料
11	玫瑰花卉
12	信義建設
13	科瑞棧藝品
14	悅式海鮮店
15	富同公司
16	新統
17	業永房屋
18	優勢企業
19	權勝

7-4 多個資料表的查詢

- [範例7-33]將『客戶』與『供應商』的資料聯集，所有重複資料皆會出現－使用UNION ALL。
- [輸出](公司名稱)

```
SELECT 公司名稱
FROM 客戶
UNION ALL
SELECT 供應商名稱
FROM 供應商
```

	公司名稱
1	心宥川公司
2	玫瑰花卉
3	日盛金樓
4	東信銀行
5	五金行
6	優勢企業
7	業永房屋
8	信義建設
9	林木材料
10	悅式海鮮店
11	丁泉
12	富同公司
13	權勝
14	科瑞棧藝品
15	宏詮工業
16	日新日公司
17	新統
18	權勝
19	妙恩
20	丁泉
21	正心

7-4 多個資料表的查詢

- [範例7-34]查詢既是『客戶』也是『供應商』的資料 - 使用INTERSECT
- [輸出] (公司名稱)

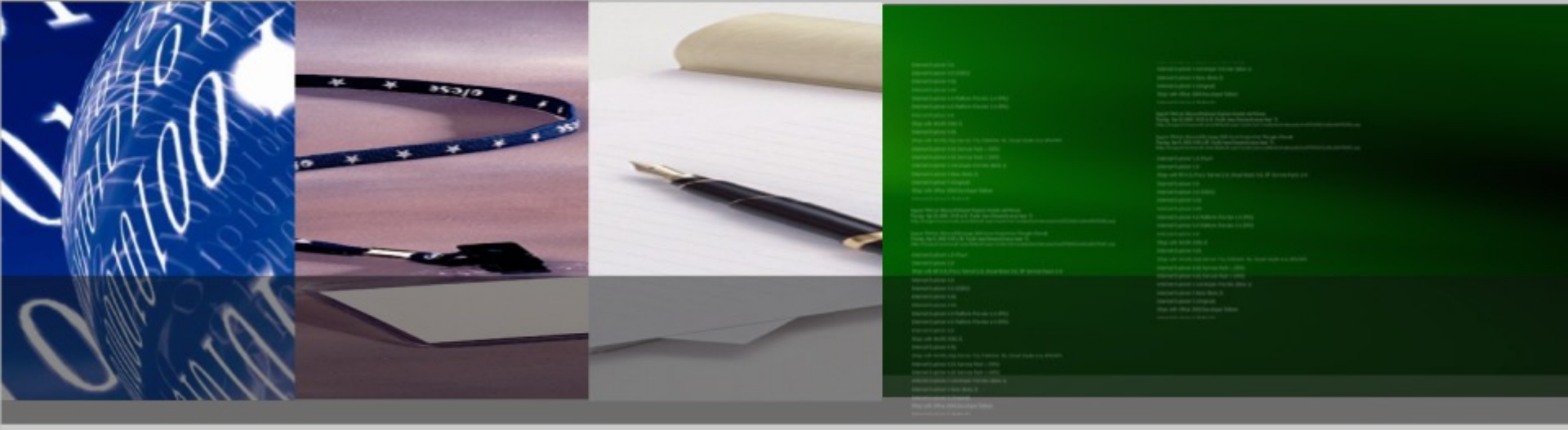
```
SELECT 公司名稱
FROM 客戶
INTERSECT
SELECT 供應商名稱
FROM 供應商
```

	公司名稱
1	丁泉
2	權勝

- [範例7-35]查詢單純是『客戶』，不是『供應商』的資料 - 使用EXCEPT。
- [輸出] (公司名稱)

```
SELECT 公司名稱
FROM 客戶
EXCEPT
SELECT 供應商名稱
FROM 供應商
```

	公司名稱
1	五金行
2	心宥川公司
3	日盛金樓
4	日新日公司
5	宏詮工業
6	東信銀行
7	林木材料
8	玫瑰花卉
9	信義建設
10	科瑞機藝品
11	悅式海鮮店
12	富同公司
13	業永房屋
14	優勢企業



7-5 不同的條件篩選方式

7-5 不同的條件篩選方式

- 利用 IN 篩選資料
  - 若是所要比對的對象不是單一個值，而是多個值(或稱為集合)時，可以使用IN的篩選方式。
    - [NOT] IN (單值或多值的運算式)
  - [範例7-35]從『客戶』資料表中挑選出住在台北市或高雄市的客戶資料。
  - [輸出] (客戶編號, 公司名稱, 聯絡人, 地址)

```
SELECT 客戶編號, 公司名稱, 聯絡人, 地址
FROM 客戶
WHERE LEFT( 地址, 3 ) = '台北市'
      OR LEFT( 地址, 3 ) = '高雄市'
```

續

7-5 不同的條件篩選方式

等同於以下寫法

```
SELECT 客戶編號, 公司名稱, 聯絡人, 地址
FROM 客戶
WHERE LEFT( 地址, 3 ) IN ('台北市', '高雄市' )
```

客戶編號	公司名稱	聯絡人	地址
C0001	心宥川公司	謝方怡	台北市南港區忠孝東路五段
C0002	玫瑰花卉	徐禹維	高雄市三民區克武路4巷
C0004	東信銀行	謝世彬	高雄市楠梓區興楠路
C0007	業永房屋	蔡爵如	台北市中山區八德路
C0008	信義建設	林美孜	台北市松山區健康路
C0009	林木材料	吳嘉修	高雄市三民區金山路
C0012	富同公司	邵雲龍	台北市中山區農安街

7-5 不同的條件篩選方式

- [範例7-37]查詢員工編號為1、3、5、7和9的員工資料。
- [輸出] (員工編號, 姓名, 職稱, 地址)

```
SELECT 員工編號, 姓名, 職稱, 地址
FROM 員工
WHERE 員工編號 = 1
      OR 員工編號 = 3
      OR 員工編號 = 5
      OR 員工編號 = 7
      OR 員工編號 = 9
```

等同於以下寫法

```
SELECT 員工編號, 姓名, 職稱, 地址
FROM 員工
WHERE 員工編號 IN ( 1, 3, 5, 7, 9 )
```

員工編號	姓名	職稱	地址
1	陳祥輝	總經理	台北市內湖區康寧路23巷
3	林其達	工程助理	台北縣中和市大勇街25巷
5	徐沛汶	業務助理	桃園縣桃園市縣府路
7	陳臆如	業務協理	台北市內湖區瑞光路513巷
9	吳志梁	業務	台中市北屯區太原路3段

7-5 不同的條件篩選方式

- 利用 BETWEEN...AND... 篩選資料
  - 當要篩選一個範圍內(或外)的資料時，可以使用 >、>=、<、<= 來限制資料的有效範圍
    - [NOT] BETWEEN 起始運算式 AND 終止運算式
  - [範例7-38]查詢員工編號為7 至 11的員工資料
  - [輸出] (員工編號,姓名,職稱,地址)

```
SELECT 員工編號, 姓名, 職稱, 地址
FROM 員工
WHERE 員工編號 >= 7 AND 員工編號 <= 11

等同於以下寫法

SELECT 員工編號, 姓名, 職稱, 地址
FROM 員工
WHERE 員工編號 BETWEEN 7 AND 11
```

員工編號	姓名	職稱	地址
7	陳臆如	業務協理	台北市內湖區瑞光路513巷
8	胡琪偉	業務	台北縣板橋市中山路一段
9	吳志梁	業務	台中市北屯區太原路3段
10	林美滿	業務經理	台北市中山區 一江街
11	劉嘉雯	業務	台北市士林區福志路

7-5 不同的條件篩選方式

- 利用 LIKE 篩選資料
  - 針對部份字串比對
  - [NOT] LIKE pattern [ESCAPE escape_character]
  - 引數pattern的使用說明

萬用字元	描述	範例
%	含有任何零個或多個字元的字串	WHERE 書名 LIKE '%資料庫%' 可查詢出所有書名中含有 '資料庫' 這三個字的任何書名資料
_ (底線)	含有任何單一字元	WHERE 書名 LIKE '__庫' 可查詢出所有以 '庫' 結尾，且只有三個字的書名(例如 『資料庫』 或『知識庫』等書名)
[ ]	包含指定範圍例如中的任何單一字元，例如也就是包括A到F的字元，等同於[ABCDEF]	WHERE 城市 LIKE '[新台]北市' 查詢第一個字是 '新' 或 '台'，第2&3字是'北市'的城市。
[^]	排除指定範圍例如中的任何單一字元，例如也就是排除A到F的字元，等同於[^ABCDEF]	WHERE 地址 LIKE '[^新]_[縣市]%' 查詢第一個字非 '新' 的所有縣市。

續

7-5 不同的條件篩選方式

- [範例7-39]查詢住在『中山區』的客戶資料。
- [輸出](客戶編號, 公司名稱, 地址)

```
SELECT 客戶編號, 公司名稱, 地址
FROM 客戶
WHERE 地址 LIKE '___中山區%'
```

客戶編號	公司名稱	地址
C0007	業永房屋	台北市中山區八德路
C0012	富同公司	台北市中山區農安街

- [範例7-40]查詢客戶地址的第一個字為『宜』或『花』或『高』的客戶資料，並依地址遞增排序。
- [輸出](客戶編號, 公司名稱, 地址)

```
SELECT 客戶編號, 公司名稱, 地址
FROM 客戶
WHERE 地址 LIKE '[宜花高]%'
ORDER BY 地址
```

客戶編號	公司名稱	地址
C0006	優勢企業	宜蘭縣頭城鎮協天路706巷
C0005	五金行	花蓮縣壽豐鄉大學路二段
C0002	玫瑰花卉	高雄市三民區克武路4巷
C0009	林木材料	高雄市三民區金山路
C0004	東信銀行	高雄市楠梓區興楠路

7-5 不同的條件篩選方式

- [範例7-41]查詢客戶地址的第一個字排除『宜』、『花』、『高』的客戶資料。也就是地址的第一個字不是『宜』、『花』、『高』的客戶資料，並依地址遞增排序。
- [輸出](客戶編號, 公司名稱, 地址)

```
SELECT 客戶編號, 公司名稱, 地址
FROM 客戶
WHERE 地址 LIKE '[^宜花高]% '
ORDER BY 地址
```

客戶編號	公司名稱	地址
C0013	權勝	台中市仁愛路四段55號
C0016	日新日公司	台中市西屯區協和里工業區37路
C0003	日盛金樓	台中市南屯區向學路
C0007	業永房屋	台北市中山區八德路
C0012	富同公司	台北市中山區農安街
C0008	信義建設	台北市松山區健康路
C0001	心宥川公司	台北市南港區忠孝東路五段
C0014	科瑞棧藝品	台北縣汐止市大同路三段
C0010	悅式海鮮店	台北縣汐止市莊敬街
C0011	丁泉	屏東縣石光村中巷1號
C0015	宏詮工業	新竹縣竹北市光明六路

7-5 不同的條件篩選方式

- 利用 NOT 的『互補性』篩選資料
  - [範例7-42]從『客戶』資料表中挑選出 不住在 台北市和高雄市的客戶資料。
  - [輸出] (客戶編號,公司名稱,聯絡人,地址)

```
SELECT 客戶編號, 公司名稱, 聯絡人, 地址
FROM 客戶
WHERE LEFT(地址, 3 ) NOT IN ('台北市', '高雄市' )
```

客戶編號	公司名稱	聯絡人	地址
C0003	日盛金樓	吳中平	台中市南屯區向學路
C0005	五金行	莊海川	花蓮縣壽豐鄉大學路二段
C0006	優勢企業	劉顯忠	宜蘭縣頭城鎮協天路706巷
C0010	悅式海鮮店	王中志	台北縣汐止市莊敬街
C0011	丁泉	周俊安	屏東縣石光村中巷1號
C0013	權勝	李姿玲	台中市仁愛路四段59號
C0014	科瑞棧藝品	黃婧貿	台北縣汐止市大同路三段
C0015	宏詮工業	朱晉陞	新竹縣竹北市光明六路
C0016	日新日公司	李豫恩	台中市西屯區協和里工業區37路

7-5 不同的條件篩選方式

- [範例7-43]查詢員工編號在7至11以外的員工資料。
- [輸出](員工編號, 姓名, 職稱,地址)

```
SELECT 員工編號, 姓名, 職稱, 地址
FROM 員工
WHERE 員工編號 NOT BETWEEN  7 AND  11
```

員工編號	姓名	職稱	地址
1	陳祥輝	總經理	台北市內湖區康寧 路23巷
2	黃謙仁	工程師	台中市西屯區工業11路
3	林其達	工程助理	台北縣中和市大勇街25巷
4	陳森耀	工程協理	台北市大安區忠孝東路4段
5	徐沛汶	業務助理	桃園縣桃園市縣府路
6	劉逸萍	業務	台北市士林區士東路
12	張懷甫	業務經理	台北市大安區仁愛路四段

7-5 不同的條件篩選方式

- [範例7-44]查詢客戶地址的第一個字排除『宜』、『花』、『高』的客戶資料，並依地址遞增排序。
- [輸出] (客戶編號, 公司名稱, 地址)

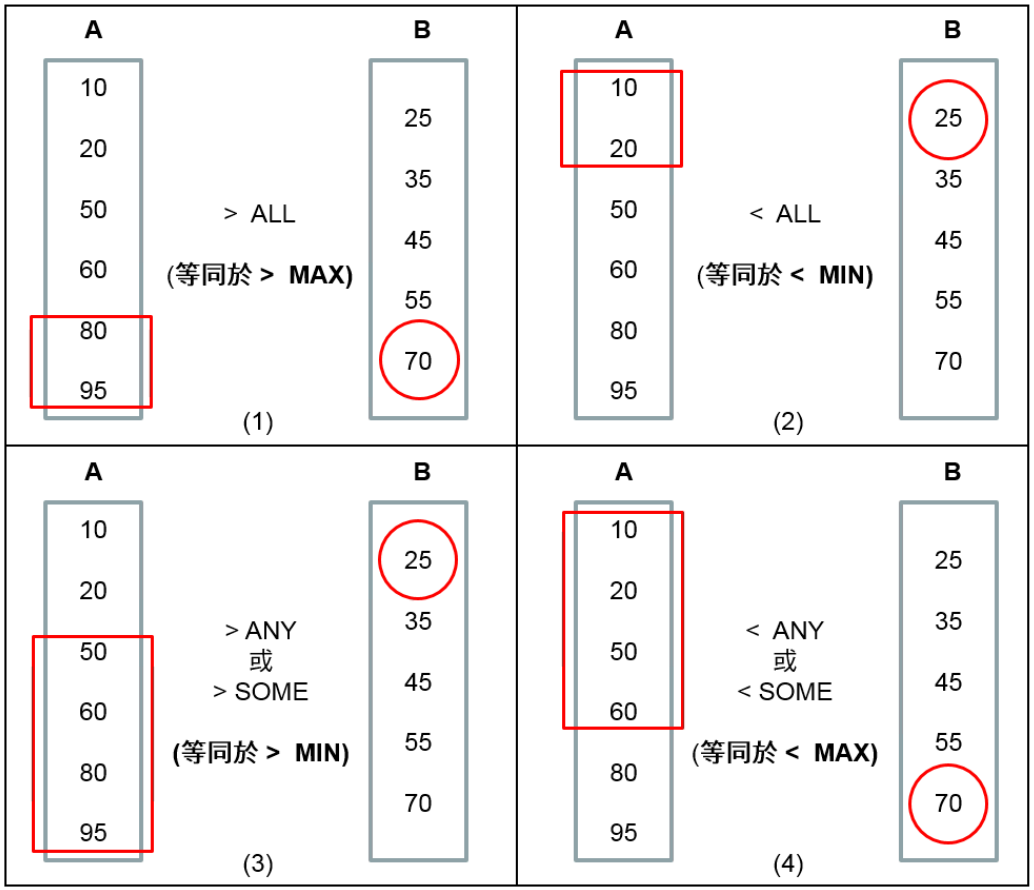
```
SELECT 客戶編號, 公司名稱, 地址
FROM 客戶
WHERE 地址 NOT LIKE ' [宜花高]% '
ORDER BY 地址
```

結果和寫法等同於範例7-41，就是利用『LIKE '[^宜花高]%'』

客戶編號	公司名稱	地址
C0013	權勝	台中市仁愛路四段59號
C0016	日新日公司	台中市西屯區協和里工業區37路
C0003	日盛金樓	台中市南屯區向學路
C0007	業永房屋	台北市中山區八德路
C0012	富同公司	台北市中山區農安街
C0008	信義建設	台北市松山區健康路
C0001	心宥川公司	台北市南港區忠孝東路五段
C0014	科瑞棧藝品	台北縣汐止市大同路三段
C0010	悅式海鮮店	台北縣汐止市莊敬街
C0011	丁泉	屏東縣石光村中巷1號
C0015	宏詮工業	新竹縣竹北市光明六路

7-5 不同的條件篩選方式

- 利用ALL與ANY/SOME篩選資料



使用ALL 與ANY /SOME來篩選符合的資料－以A、B 兩集合為例

續

7-5 不同的條件篩選方式

- 範例練習
  - 基於下列的兩個檢視表，分別為『果汁類產品』與『茶類產品』，透過ALL與ANY來比較兩個類別產品的建議單價。

果汁類產品資料

	類別編號	類別名稱	產品編號	產品名稱	建議單價
1	1	果汁	1	蘋果汁	18
2	1	果汁	2	蔬果汁	20
3	1	果汁	4	蘆筍汁	15

茶類產品資料

	類別編號	類別名稱	產品編號	產品名稱	建議單價
1	2	茶類	6	烏龍茶	25
2	2	茶類	7	紅茶	15

續

7-5 不同的條件篩選方式

- [範例7-45]查詢『茶類產品資料』中有哪些產品的『建議單價』比『果汁類產品資料』中所有產品的『建議單價』高。
- [輸出](類別名稱, 產品名稱, 建議單價)

	類別名稱	產品名稱	建議單價
1	茶類	烏龍茶	25

```
SELECT 類別名稱, 產品名稱, 建議單價
FROM 茶類產品資料
WHERE 建議單價 > ALL ( SELECT 建議單價 FROM 果汁類產品資料 )
```

- [範例7-46]查詢『果汁類產品資料』中有哪些產品的『建議單價』比『茶類產品資料』中任何一項產品的『建議單價』高。
- [輸出](類別名稱, 產品名稱, 建議單價)

```
SELECT 類別名稱, 產品名稱, 建議單價
FROM 果汁類產品資料
WHERE 建議單價 > ANY (SELECT 建議單價 FROM 茶類產品資料 )
```

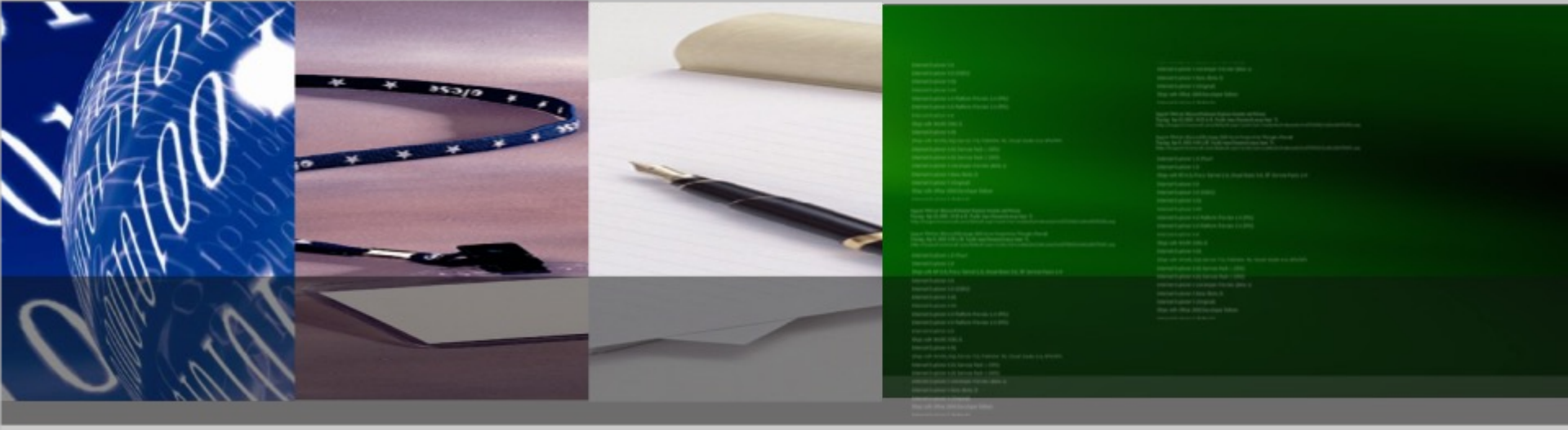
	類別名稱	產品名稱	建議單價
1	果汁	蘋果汁	18
2	果汁	蔬果汁	20

7-5 不同的條件篩選方式

- 篩選空值(Null)的資料
  - [範例7-47]查詢訂單資料表中，貨品尚未到達的資料，也就是判斷實際到貨日期是否為空值。
  - [輸出](訂單編號,姓名,公司名稱,訂貨日期,預計到貨日期,實際到貨日期)

```
SELECT 訂單編號, 姓名, 公司名稱, 訂貨日期, 預計到貨日期, 實際到貨日期
FROM 員工 E, 訂單 O, 客戶 C
WHERE E.員工編號 = O.員工編號
   AND   O.客戶編號 = C.客戶編號
   AND   實際到貨日期 is null
```

訂單編號	姓名	公司名稱	訂貨日期	預計到貨日期	實際到貨日期
94010702	林美滿	林木材料	2006-02-27	2006-03-03	NULL
94010803	林美滿	權勝	2006-05-20	2005-06-01	NULL
94010804	陳膣如	日新日公司	2006-06-20	2006-07-01	NULL
94010806	劉逸萍	丁泉	2006-11-08	2006-11-12	NULL



7-6 彙總函數與 GROUP BY…HAVING…

7-6 彙總函數與GROUP BY…HAVING...

- GROUP BY常用的彙總函數
  - SUM()：加總函數
  - COUNT()：計數筆數函數
  - AVG()：平均函數
  - MAX()：最大值函數
  - MIN() ：最小值函數

7-6 彙總函數與GROUP BY…HAVING…

- GROUP BY … HAVING…
  - [範例7-48]計算出每位員工所承接的每張訂單之總金額，並依據員工姓名與訂單編號遞增排序。
  - [輸出](姓名,訂單編號,訂貨日期,總金額)

```
SELECT 姓名, 訂單.訂單編號, 訂貨日期, sum( 實際單價 * 數量 ) AS 總金額
FROM 員工, 訂單, 訂單明細
WHERE 員工.員工編號 = 訂單.員工編號
   AND   訂單.訂單編號 = 訂單明細.訂單編號
GROUP BY 姓名, 訂單.訂單編號, 訂貨日期
ORDER BY 姓名 ASC, 訂單.訂單編號 ASC
```

	姓名	訂單編號	訂貨日期	總金額
1	吳志梁	94010701	2006-01-27	455
2	林美滿	94010105	2005-01-11	650
3	林美滿	94010201	2005-03-12	1205
4	林美滿	94010302	2005-08-03	200
5	林美滿	94010303	2005-09-03	595
6	林美滿	94010702	2006-02-27	1468
7	林美滿	94010803	2006-05-20	1225
8	胡琪偉	94010301	2005-07-03	654
9	胡琪偉	94010601	2005-12-16	1000
10	陳臆如	94010104	2005-01-10	616
11	陳臆如	94010401	2005-11-04	270
12	陳臆如	94010501	2005-12-15	180
13	劉逸萍	94010202	2005-05-12	450
14	劉逸萍	94010705	2006-02-27	400
15	劉逸萍	94010801	2006-04-18	880
16	劉逸萍	94010806	2006-11-08	1350

7-6 彙總函數與GROUP BY…HAVING…

- [範例7-49]挑選出2006年(含)以後的訂單，並計算出每位員工所承接的每張訂單之總金額，並依據員工姓名與訂單編號遞增排序。
- [輸出](姓名,訂單編號,訂貨日期,總金額)

```
SELECT 姓名, 訂單.訂單編號, 訂貨日期, sum( 實際單價 * 數量 )AS 總金額
FROM  員工, 訂單, 訂單明細
WHERE 員工.員工編號 = 訂單.員工編號
   AND  訂單.訂單編號 = 訂單明細.訂單編號
   AND  訂貨日期 >= '2006/01/01'
GROUP BY 姓名, 訂單.訂單編號, 訂貨日期
ORDER BY 姓名 ASC, 訂單.訂單編號 ASC
```

	姓名	訂單編號	訂貨日期	總金額
1	吳志梁	94010701	2006-01-27	455
2	林美滿	94010702	2006-02-27	1468
3	林美滿	94010803	2006-05-20	1225
4	劉逸萍	94010705	2006-02-27	400
5	劉逸萍	94010801	2006-04-18	880
6	劉逸萍	94010806	2006-11-08	1350

7-6 彙總函數與GROUP BY…HAVING…

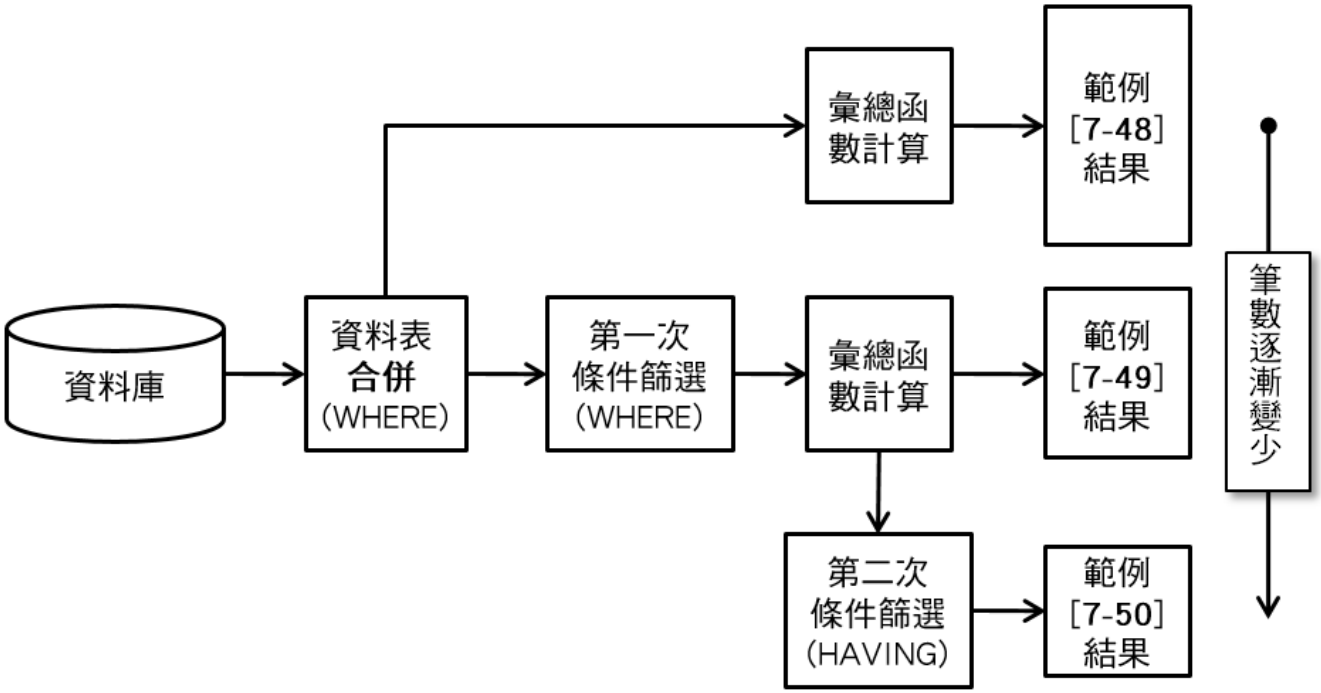
- [範例7-50]挑選出2006年(含)以後的訂單，而且單筆訂單總金額超過1000元(不含)的資料，並依據員工姓名與訂單編號遞增排序。
- [輸出](姓名,訂單編號,訂貨日期,總金額)

```
SELECT 姓名, 訂單.訂單編號, 訂貨日期, sum( 實際單價 * 數量 ) AS 總金額
FROM 員工, 訂單, 訂單明細
WHERE 員工.員工編號 = 訂單.員工編號
   AND   訂單.訂單編號 = 訂單明細.訂單編號
   AND   訂貨日期 >='2006/01/01'
GROUP BY 姓名, 訂單.訂單編號, 訂貨日期
HAVING sum( 實際單價 * 數量 ) > 1000
ORDER BY 姓名 ASC, 訂單.訂單編號 ASC
```

	姓名	訂單編號	訂貨日期	總金額
1	林美滿	94010702	2006-02-27	1468
2	林美滿	94010803	2006-05-20	1225
3	劉逸萍	94010806	2006-11-08	1350

7-6 彙總函數與GROUP BY…HAVING…

- 範例7-48～7-50 的查詢流程



7-6 彙總函數與GROUP BY…HAVING…

- [範例7-51]請依據本公司員工性別，分別列出男、女的人數、平均年齡最高與最低年齡。
- 【輸出】(性別,人數,平均年齡,最高年齡,最低年齡)

```
SELECT 性別
          , COUNT( 員工編號 ) 人數
          , AVG( YEAR(GETDATE())-YEAR(出生日期) ) 平均年齡
          , MAX( YEAR(GETDATE())-YEAR(出生日期) ) 最高年齡
          , MIN( YEAR(GETDATE())-YEAR(出生日期) ) 最低年齡
FROM 員工
GROUP BY 性別
```

NOTE

使用COUNT( )函數時，在小括弧內可以填入員工編號，或是資料表內的任何一個屬性，甚至是填入萬用字元 *。可是，建議不要使用*，否則將會降低執行效率。

性別	人數	平均年齡	最高年齡	最低年齡
女	5	46	55	26
男	7	49	61	42

7-6 彙總函數與GROUP BY…HAVING…

- GROUP BY WITH ROLLUP / CUBE
  - 透過GROUP BY 後面加上ROLLUP或CUBE，可以針對GROUP BY後面的屬性進行不同組合的總計。
  - [範例7-52]請針對個別年度與季別計算出總金額，並依據年度作加總，以及所有的總金額加總。
  - 【輸出】(年度,季別,總金額)

```
SELECT DATEPART(YEAR,訂貨日期) 年度
          , DATEPART(QUARTER,訂貨日期) 季別
          , SUM(實際單價*數量) 總金額
FROM 訂單, 訂單明細
WHERE 訂單.訂單編號=訂單明細.訂單編號
GROUP BY DATEPART(YEAR,訂貨日期)
          , DATEPART(QUARTER,訂貨日期) WITH ROLLUP
```

	年度	季別	總金額
1	2005	1	2471
2	2005	2	450
3	2005	3	1449
4	2005	4	1450
5	2005	NULL	5820
6	2006	1	2323
7	2006	2	2105
8	2006	4	1350
9	2006	NULL	5778
10	NULL	NULL	11598

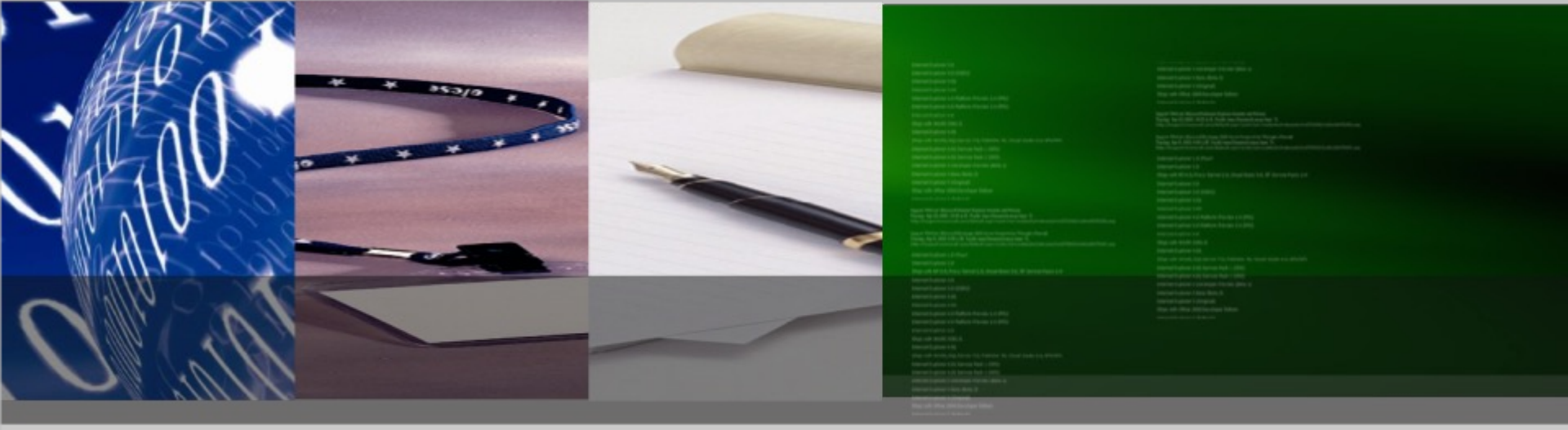
7-6 彙總函數與GROUP BY…HAVING…

- [範例7-53]請針對個別年度與季別計算出總金額，並依據季別作加總，再依年度作加總，最後再加總所有的總金額
- 【輸出】(年度,季別,總金額)

```
SELECT DATEPART(YEAR,訂貨日期) 年度
          , DATEPART(QUARTER,訂貨日期) 季別
          , SUM(實際單價*數量) 總金額
FROM 訂單, 訂單明細
WHERE 訂單.訂單編號=訂單明細.訂單編號
GROUP BY DATEPART(YEAR,訂貨日期)
          , DATEPART(QUARTER,訂貨日期) WITH CUBE
```

	年度	季別	總金額
1	2005	1	2471
2	2006	1	2323
3	NULL	1	4794
4	2005	2	450
5	2006	2	2105
6	NULL	2	2555
7	2005	3	1449
8	NULL	3	1449
9	2005	4	1450
10	2006	4	1350
11	NULL	4	2800
12	NULL	NULL	11598
13	2005	NULL	5820
14	2006	NULL	5778

原本只是以年度＋季別為群組來計算總金額，加上**WITH CUBE**後，就會以年度和季別的不同組合來進行總計



7-7 子查詢(Sub-Query)

7-7 子查詢(Sub-Query)

- 『子查詢』介紹
  - 在 SELECT敘述內還有SELECT敘述
  - 位於內部的SELECT敘述
  - 必須使用小括弧()括起來
  - 子查詢與原本主要查詢之間，可以依據相依關係分為『獨立子查詢』與『相互關聯子查詢』兩種
- 子查詢的傳回資料可分為三種不同結果
  - 單一欄位單筆紀錄
  - 單一欄位多筆紀錄
  - 多個欄位多筆紀錄

7-7 子查詢(Sub-Query)

- 獨立子查詢
  - [範例7-54] 單一欄位多筆紀錄
  - 從所有供應商中，查詢出有哪些供應商也有提供編號為'S0001'供應商所提供的類別產品。
  - [輸出] (供應商編號, 供應商名稱, 類別編號)

```
SELECT DISTINCT 供應商.供應商編號, 供應商名稱, 類別編號
FROM 供應商, 產品資料
WHERE 供應商.供應商編號 = 產品資料.供應商編號
   AND   類別編號 IN ( SELECT DISTINCT 類別編號
                       FROM 產品資料
                       WHERE 供應商編號 ='S0001')
 ORDER BY 供應商編號
```

供應商編號	供應商名稱	類別編號
S0001	新統	1
S0001	新統	3
S0002	權勝	1

續

7-7 子查詢(Sub-Query)

- 此範例中的 IN 子句即是SELECT的子查詢，執行後的結果為一個集合{ 1, 3 }，也就是單一欄位且多筆資料。

```
SELECT DISTINCT 供應商.供應商編號, 供應商名稱, 類別編號
FROM 供應商, 產品資料
WHERE 供應商.供應商編號 = 產品資料.供應商編號 AND
        類別編號 IN ( SELECT DISTINCT 類別編號
                      FROM 產品資料
                      WHERE 供應商編號 = 'S0001' )
ORDER BY 供應商編號
```

獨立子查詢
可以獨立執行
結果：{ 1, 3 }

7-7 子查詢(Sub-Query)

- [範例7-55] 多個欄位多筆紀錄
- 計算在2006/01/01(含)之後的訂單資料，列出每位員工單筆訂單總金額有超過1000元的訂單筆數。
- [輸出](員工姓名,筆數)

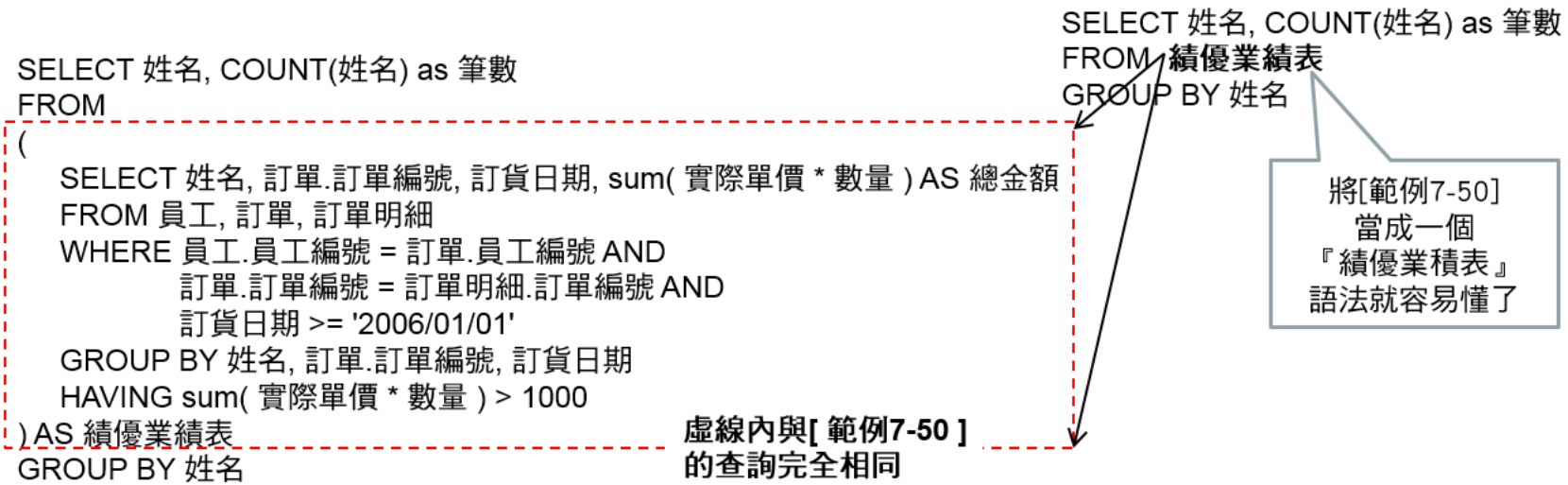
```
SELECT 姓名 AS 員工姓名, COUNT(姓名) AS 筆數
FROM
(
    SELECT 姓名, 訂單.訂單編號, 訂貨日期, sum( 實際單價 * 數量 ) AS 總金額
    FROM 員工, 訂單, 訂單明細
    WHERE 員工.員工編號 = 訂單.員工編號
        AND  訂單.訂單編號 = 訂單明細.訂單編號
        AND  訂貨日期 >= '2006/01/01'
     GROUP BY 姓名, 訂單.訂單編號, 訂貨日期
     HAVING sum( 實際單價 * 數量 ) > 1000
) AS 績優業績表
GROUP BY 姓名
```

員工姓名	筆數
林美滿	2
劉逸萍	1

續

7-7 子查詢(Sub-Query)

- 此查詢可以視為是先將範例7-50的查詢結果，當成一個名為『績優業績表』的資料表(或檢視表)，針對『績優業績表』再進行一次的彙總函數的計數。
- 範例7-50的SELECT子句成為此範例的一個『獨立子查詢』。



7-7 子查詢(Sub-Query)

- 相互關聯子查詢
  - [範例7-56]題目與結果同於範例7-34
  - 查詢既是『客戶』也是『供應商』的資料－使用子查詢。
  - [輸出](公司名稱)

```
SELECT公司名稱
FROM 客戶
WHERE 公司名稱 IN ( SELECT 供應商名稱
                    FROM 供應商
                    WHERE 供應商名稱 = 公司名稱 )
```

  - 等同於以下直接使用內部合併方式和結果
  - [輸出](公司名稱)

```
SELECT 客戶.公司名稱
FROM 客戶, 供應商
WHERE 公司名稱 = 供應商名稱
```

	公司名稱
1	丁泉
2	權勝

7-7 子查詢(Sub-Query)

- [範例7-57]題目與結果同於範例7-35
- 查詢單純是『客戶』，不是『供應商』的資料－使用子查詢。
- [輸出](公司名稱)

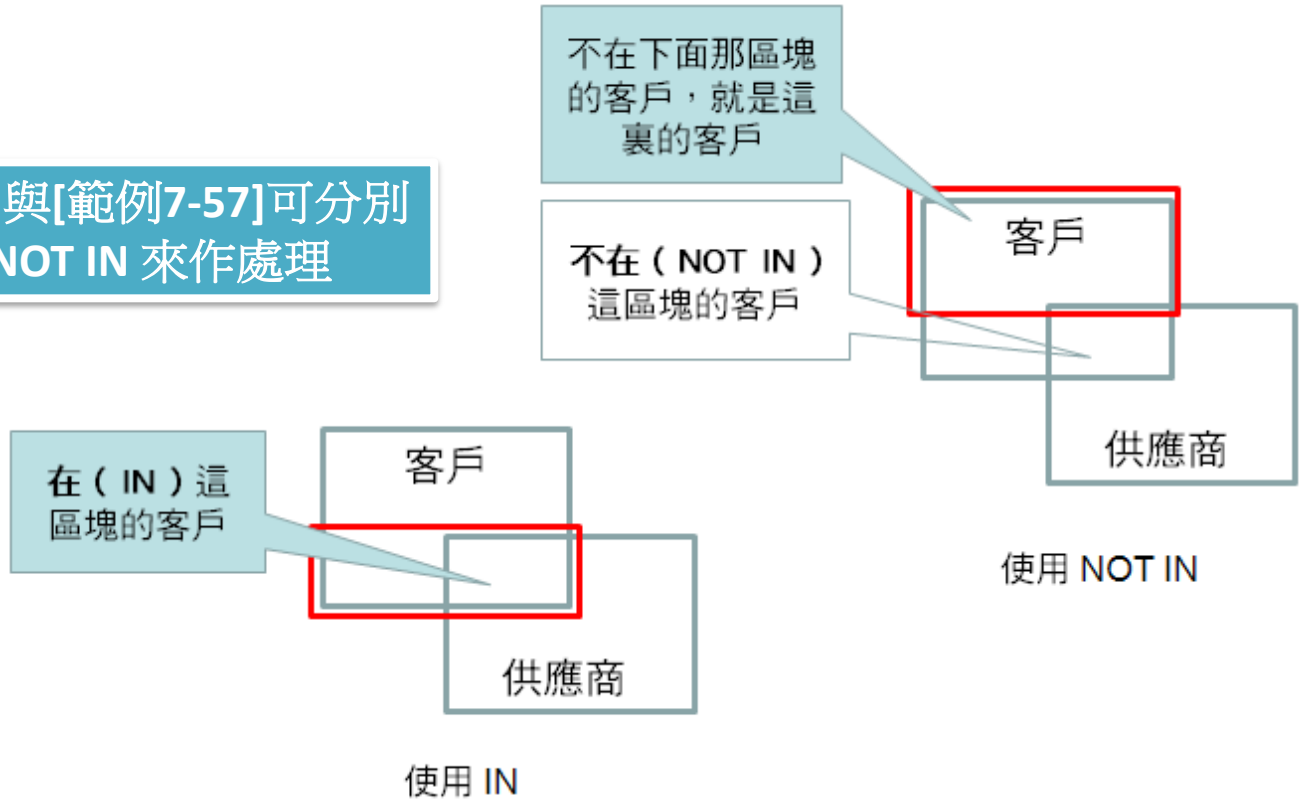
```
SELECT 公司名稱
FROM 客戶
WHERE 公司名稱 NOT IN ( SELECT 供應商名稱
                         FROM 供應商
                         WHERE 供應商名稱 = 公司名稱)
```

	公司名稱
1	五金行
2	心宥川公司
3	日盛金樓
4	日新日公司
5	宏詮工業
6	東信銀行
7	林木材料
8	玫瑰花卉
9	信義建設
10	科瑞棧藝品
11	悅式海鮮店
12	富同公司
13	業永房屋
14	優勢企業

7-7 子查詢(Sub-Query)

- 範例7-56 與7-57之圖解比較

[範例7-56]與[範例7-57]可分別使用IN 與NOT IN 來作處理



7-7 子查詢(Sub-Query)

- 存在性測試的EXISTS
  - [範例7-58]題目與結果同於範例7-56
  - 查詢既是『客戶』也是『供應商』的資料－使用 EXISTS

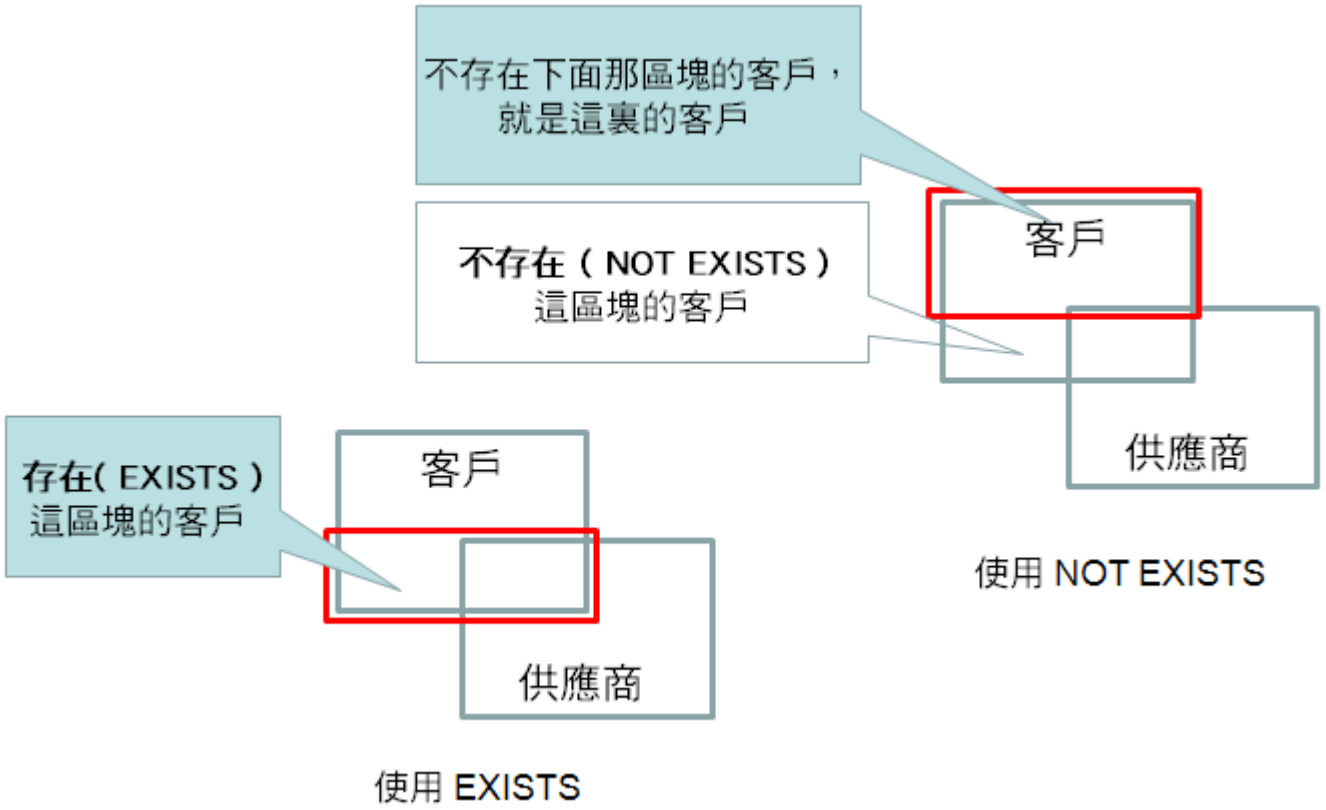
```
SELECT 公司名稱
FROM 客戶
WHERE EXISTS ( SELECT *
                    FROM 供應商
                    WHERE 供應商名稱 = 公司名稱)
```

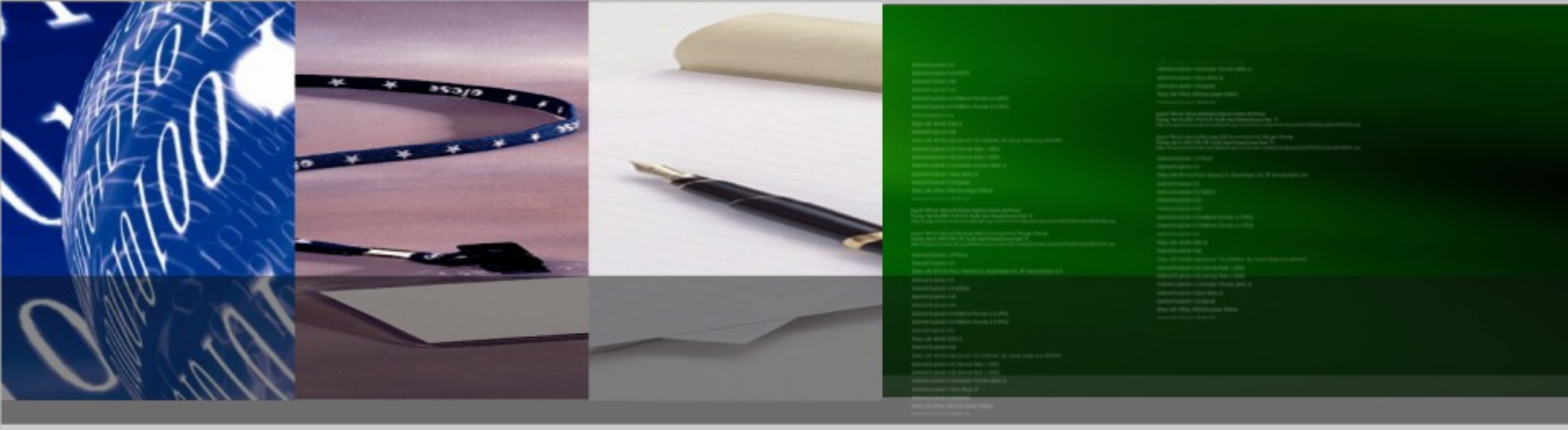
  - [範例7-59]題目與結果同於範例7-57
  - 查詢單純是『客戶』，不是『供應商』的資料－使用NOT EXISTS。

```
SELECT 公司名稱
FROM 客戶
WHERE NOT EXISTS ( SELECT *
                        FROM 供應商
                        WHERE 供應商名稱 = 公司名稱)
```

7-7 子查詢(Sub-Query)

- [範例7-58] 與[範例7-59] 之圖解比較

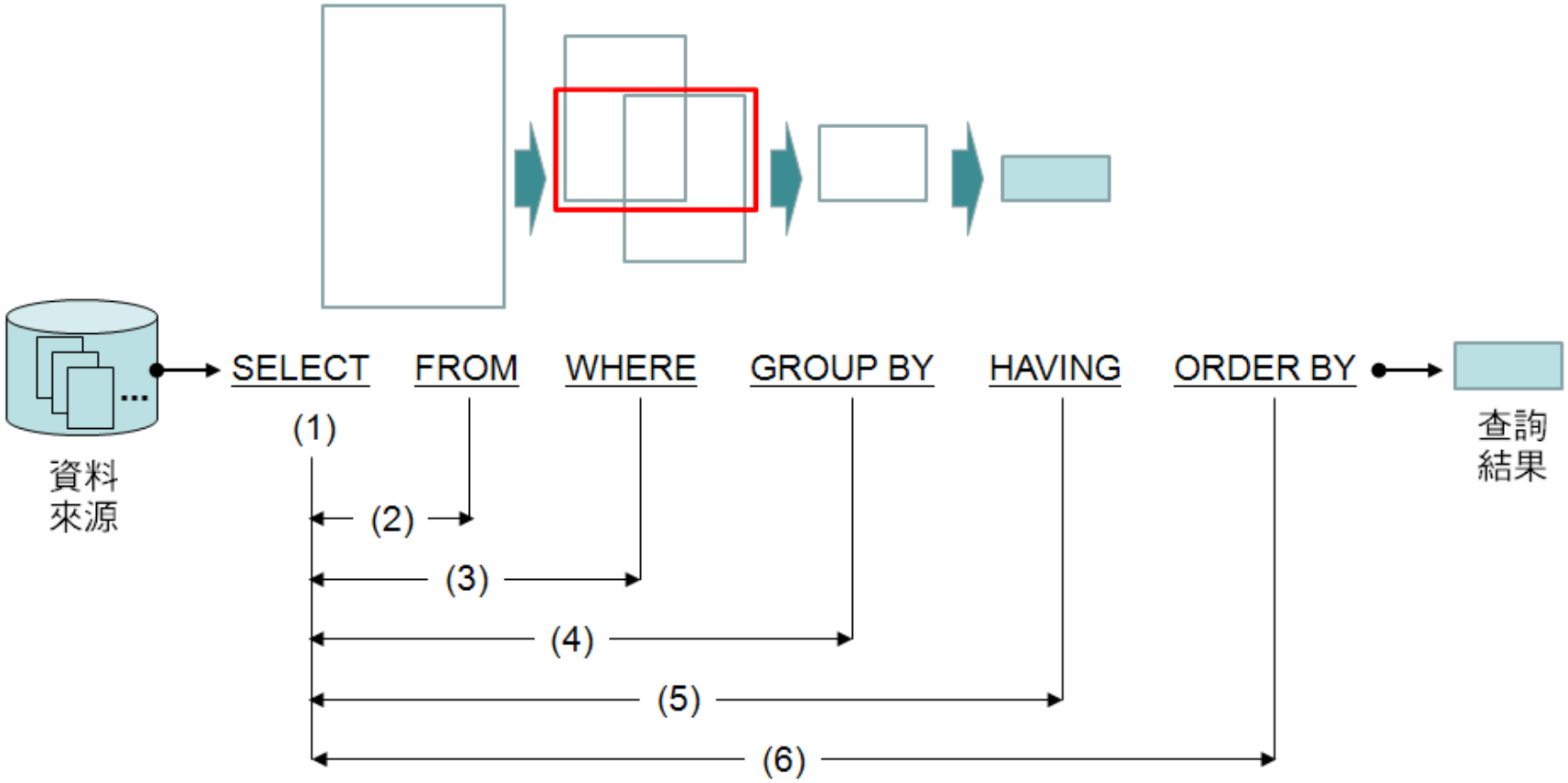




7-8 SELECT語法與語意之剖析整理

7-8 SELECT語法與語意之剖析整理

- SELECT的基本語法分為六個階段



續

7-8 SELECT語法與語意之剖析整理

- (1)SELECT子句主要是針對『運算式』的輸出。
- (2)FROM後面倘若接的是數個資料表，所輸出的結果即為『交叉合併』(CROSS JOIN)。
- (3)WHERE後面主要可為以下兩種情形：
  - 『關聯性』PK與FK之間的對應關係。此關聯性建立後，等同於從(2)的『交叉合併』轉變為『內部合併』。
  - 『條件篩選』對於『內部合併』後的結果再進行條件篩選。
- (4)GROUP BY是針對一些資料行當成分群的依據，再進行資料的彙總(aggregation)處理。
- (5)條件篩選是針對於彙總後的資料進行篩選。
- (6)當資料處理完成後，最後就是將資料進行排序。

7-8 SELECT語法與語意之剖析整理

- [範例7-60]計算2006/01/01(含)之後每張訂單的獲利大於300的相關資料。
  - [輸出](負責人姓名, 訂單編號, 客戶公司名稱, 毛利)
  - [提示]毛利計算 = SUM((實際單價 - 平均成本)*數量)
  - 以分解步驟來說明，並以上圖(1)-(6)的標號來對照
    - (1) - (2) 交叉合併

```
SELECT *
FROM 員工, 訂單, 訂單明細, 產品資料, 客戶
```

續

7-8 SELECT語法與語意之剖析整理

- (3) 內部合併與條件篩選

```
1.SELECT *
  FROM員工, 訂單, 訂單明細, 產品資料, 客戶
  WHERE 員工.員工編號 = 訂單.員工編號
     AND  訂單.訂單編號 = 訂單明細.訂單編號
     AND  訂單明細.產品編號 = 產品資料.產品編號
     AND  訂單.客戶編號 = 客戶.客戶編號

2.SELECT *
  FROM 員工, 訂單, 訂單明細, 產品資料, 客戶
  WHERE 員工.員工編號 = 訂單.員工編號
     AND  訂單.訂單編號 = 訂單明細.訂單編號
     AND  訂單明細.產品編號 = 產品資料.產品編號
     AND  訂單.客戶編號 = 客戶.客戶編號
    AND   訂貨日期 >= '2006/01/01'
```

續

7-8 SELECT語法與語意之剖析整理

- (4) 分群組與彙總函數計算

```
SELECT 姓名 AS 負責人姓名,訂單.訂單編號,公司名稱 AS 客戶公司名稱
       , sum(( 實際單價 - 平均成本 )*數量) AS 毛利
FROM 員工, 訂單, 訂單明細, 產品資料, 客戶
WHERE 員工.員工編號 = 訂單.員工編號
   AND 訂單.訂單編號 = 訂單明細.訂單編號
   AND 訂單明細.產品編號 = 產品資料.產品編號
   AND 訂單.客戶編號 = 客戶.客戶編號
   AND 訂貨日期 >='2006/01/01'
GROUP BY 姓名, 訂單.訂單編號, 公司名稱
```

續

7-8 SELECT語法與語意之剖析整理

- (5) 彙整函數後的條件篩選

```
SELECT 姓名 AS 負責人姓名, 訂單.訂單編號, 公司名稱 AS 客戶公司名稱
          , sum(( 實際單價 - 平均成本 )*數量) AS 毛利
FROM 員工, 訂單, 訂單明細, 產品資料, 客戶
WHERE 員工.員工編號 = 訂單.員工編號
   AND   訂單.訂單編號 = 訂單明細.訂單編號
   AND   訂單明細.產品編號 = 產品資料.產品編號
   AND   訂單.客戶編號 = 客戶.客戶編號
   AND   訂貨日期 >='2006/01/01'
GROUP BY 姓名, 訂單.訂單編號, 公司名稱
HAVING sum( ( 實際單價 - 平均成本 )*數量) > 300
```

續

7-8 SELECT語法與語意之剖析整理

- (6) 排序

```
SELECT 姓名 AS 負責人姓名, 訂單.訂單編號, 公司名稱 AS 客戶公司名稱
          , sum(( 實際單價 - 平均成本 )*數量 ) AS 毛利
FROM 員工, 訂單, 訂單明細, 產品資料, 客戶
WHERE 員工.員工編號 = 訂單.員工編號
   AND   訂單.訂單編號 = 訂單明細.訂單編號
   AND   訂單明細.產品編號 = 產品資料.產品編號
   AND   訂單.客戶編號 = 客戶.客戶編號
   AND   訂貨日期 >='2006/01/01'
GROUP BY 姓名, 訂單.訂單編號, 公司名稱
HAVING sum(( 實際單價 - 平均成本 )*數量 ) > 300
ORDER BY 姓名 ASC, 訂單.訂單編號 ASC
```

本章結束

Q&A討論時間