CHAP 15 自訂函數與順序物件

- 15-1 自訂函數的特色
- 15-2 自訂函數的建立、使用與修改
- 15-3 自訂函數的使用技巧
- 15-4 使用順序物件

15-1 自訂函數的特色

- 預存程序只能傳回一個整數值，自訂函數可傳回各種型別的資料，除 **text**、**ntext**、**image**、**cursor** 與 **rowversion**。
- 預存程序可以經由參數來傳回資料 (將參數設為 **OUTPUT**)，自訂函數只能接收參數，不可由參數傳回資料。
- 在預存程序中可以做任何的資料異動，例如新增或修改資料、更改資料庫的設定...等。自訂函數則不能更改資料庫的狀態或內容。
- 預存程序必須以 **EXECUTE** 來執行，因此不能使用在運算式之中，例如 **myProc** 會傳回 **2**，那麼『**SET @var = myProc**』或『**SELECT * FROM myProc**』都會造成錯誤。自訂函數可(不需) **EXECUTE** 來執行，則可以，例如 **SET @var = myFun(3) + "!"**
- 自訂函數禁止使用 **TRY ... CATCH**

自訂函數的特色

- 兩個呼叫自訂函數的例子：

```
DECLARE @TopProductID int
SET @TopProductID = dbo.GetTopProductID('2016')   ← 傳回 2016 年最賣座的產品

SELECT 姓名, 區域
FROM GetEmployeeFromProdId(@TopProductID)   ← 傳回該產品的銷售人員資料集
WHERE 區域 = '北區'
```

	姓名	區域
1	孟庭亭	北區

15-2 自訂函數的建立、使用與修改

程式類型	建立	修改	刪除
預存程序	CREATE PROC	ALTER PROC	DROP PROC
自訂函數	CREATE FUNCTION	ALTER FUNCTION	DROP FUNCTION

- 純量值函數 **(Scalar-valued function)**

  會傳回單一的資料值。

- 資料表值函數 **(Table-valued function)**
  - 內嵌資料表值函數 **(Inline table-valued function)**：函數內回傳一個 **SELECT** 敘述的查詢結果。
  - 多重陳述式資料表值函數 **(Multistatement table-valued function)**：函數內包含許多的敘述，傳回一個 **table** 型別的資料集。

純量值函數

```
CREATE FUNCTION function_name
( [ {@par am_name scalar_data_type [=defaul t ] [READONLY] } [,...n] ] )
RETURNS scalar_return_data_type
[ WITH { ENCRYPTION | SCHEMABINDING } [ [, ]...n ] ]
[ AS ]
BEGIN
   function_body
   RETURN scalar_expression
END
```

純量值函數

- **CREATE FUNCTION function_name**
- **@param_name scalar_data_type [=default] [READONLY]**

```
CREATE FUNCTION myFunct
(@a char(10) READONLY, @b int = 500)
.....
```

```
CREATE TYPE NameValueTable AS TABLE      ◄── 定義『使用者定義資料表類型』，
(名稱 varchar(50) , 數值 int )                內含名稱、數值 2 個欄位
GO

CREATE FUNCTION 傳回最大者
(@tab NameValueTable READONLY)           ◄── 使用 NameValueTable 型別的參數
...
```

純量值函數

- **RETURNS scalar_return_data_type**
- **WITH <function_option> [[,] …n]**
- **AS**
- **function_body**
- **RETURN scalar_expression**：函數最後的敘述必須是一個 **RETURN**。

純量值函數

	訂單序號	日期	客戶編號	是否付款	備註
1	1	2016-09-22	5	1	NULL
2	2	2016-10-14	6	1	NULL
3	3	2016-10-18	1	1	NULL
4	4	2016-10-19	2	1	NULL
5	5	2016-11-03	8	0	<Long Text>
6	6	2016-11-05	5	1	NULL
7	9	2016-11-06	4	1	NULL
8	10	2016-11-06	6	1	NULL
9	11	2016-11-06	3	1	NULL
10	12	2016-11-06	3	1	NULL

訂單資料表

	細目序號	訂單序號	數量	書籍編號
1	1	1	40	4
2	2	1	9	2
3	3	1	40	10
4	4	2	20	4
5	5	3	15	3
6	6	3	10	8
7	7	3	22	12
8	8	3	20	6

訂單細目資料表

```
/* 傳回在指定年度中最暢銷的書籍 */
CREATE FUNCTION GetTopProductID
(@year varchar(5))
RETURNS int
BEGIN
    DECLARE @id int
    SELECT TOP 1 @id = 書籍編號
    FROM 訂單 JOIN 訂單細目
              ON 訂單.訂單序號 =  訂單細目.訂單序號
    WHERE DATEPART (YYYY, 日期) = @year
    GROUP BY 書籍編號
    ORDER BY SUM(數量) DESC
    IF @@ROWCOUNT = 0
         RETURN 0
     RETURN @id
END
GO
```

```
SELECT dbo.GetTopProduct ID( '2016' ) AS '書籍編號'  ←— 執行自訂函數
GO
```

	書籍編號
1	6

純量值函數

```
CREATE TYPE NameValueTable AS TABLE   --  建立『使用者定義資料表類型』,
(名稱 varchar(50), 數值 int )         --  內含【名稱、數值】 2 個欄位
GO

CREATE FUNCTION 傳回最大者
(@tab NameValueTable READONLY)
RETURNS varchar(50)
BEGIN
    DECLARE @maxName varchar(50)

    SELECT TOP 1 @maxName = 名稱      -- 取加總數量的最大者
    FROM @tab
    GROUP BY 名稱                     -- 依名稱分組
    ORDER BY SUM(數值) DESC           -- 加總數量並遞減排序

    RETURN @maxName
END
GO
```

純量值函數

```
DECLARE @tab NameValueTable

INSERT @tab
SELECT 客戶名稱, 數量
FROM 出貨記錄

SELECT '出貨量最大的客戶為：' + dbo.傳回最大者(@tab)

DELETE @tab

INSERT @tab
SELECT 股票名稱, 購買張數
FROM 股票交易記錄

SELECT  '庫存最多的股票為：' + dbo.傳回最大者(@tab)
```

	(沒有資料行名稱)
1	出貨量最大的客戶為：大雄書局

	(沒有資料行名稱)
1	庫存最多的股票為：統一

內嵌資料表值函數

```
CREATE FUNCTION function_name
( [ {@parameter_name scalar_data_type [=default ] [READONLY]} [ , ...n] ] )
RETURNS TABLE
[WITH < function_option > [ [, ]...n] ]
[AS]
RETURN [( ] select-statement [ )]

< function_option > ::=
    { ENCRYPTION | SCHEMABINDING }
```

內嵌資料表值函數

- **RETURNS TABLE**
- **RETURN [( ] select-statement [ )]**

	書籍編號	書籍名稱	單價	負責人
1	1	AutoCAD 操作入門	350.00	3
2	2	Linux 使用手冊	420.00	3
3	3	Excel 使用手冊	450.00	1
4	4	Internet 上線實務	320.00	3
5	5	Internet 精緻之旅	320.00	2
6	6	Flash 學習手冊	400.00	2
7	7	PhotoShop 細說從頭	980.00	2
8	8	PowerPoint 使用手冊	450.00	1
9	9	Visual Basic 學習手冊	350.00	3

書籍資料表

內嵌資料表值函數

```
CREATE FUNCTION 依售價查詢書籍
(@由 money, @到 money)
RETURNS TABLE
RETURN (SELECT 書籍編號, 書籍名稱, 單價
        FROM 書籍
        WHERE 單價 >=@由 AND 單價 <= @到 )
GO

SELECT *
FROM 依售價查詢書籍(400, 450)
ORDER BY 單價
```

	書籍編號	書籍名稱	單價
1	6	Flash 學習手冊	400.00
2	2	Linux 使用手冊	420.00
3	3	Excel 使用手冊	450.00
4	8	PowerPoint 使用手冊	450.00
5	10	Windows 使用手冊	450.00
6	11	Word 使用手冊	450.00

查出 400～450 元之間的書籍

內嵌資料表值函數

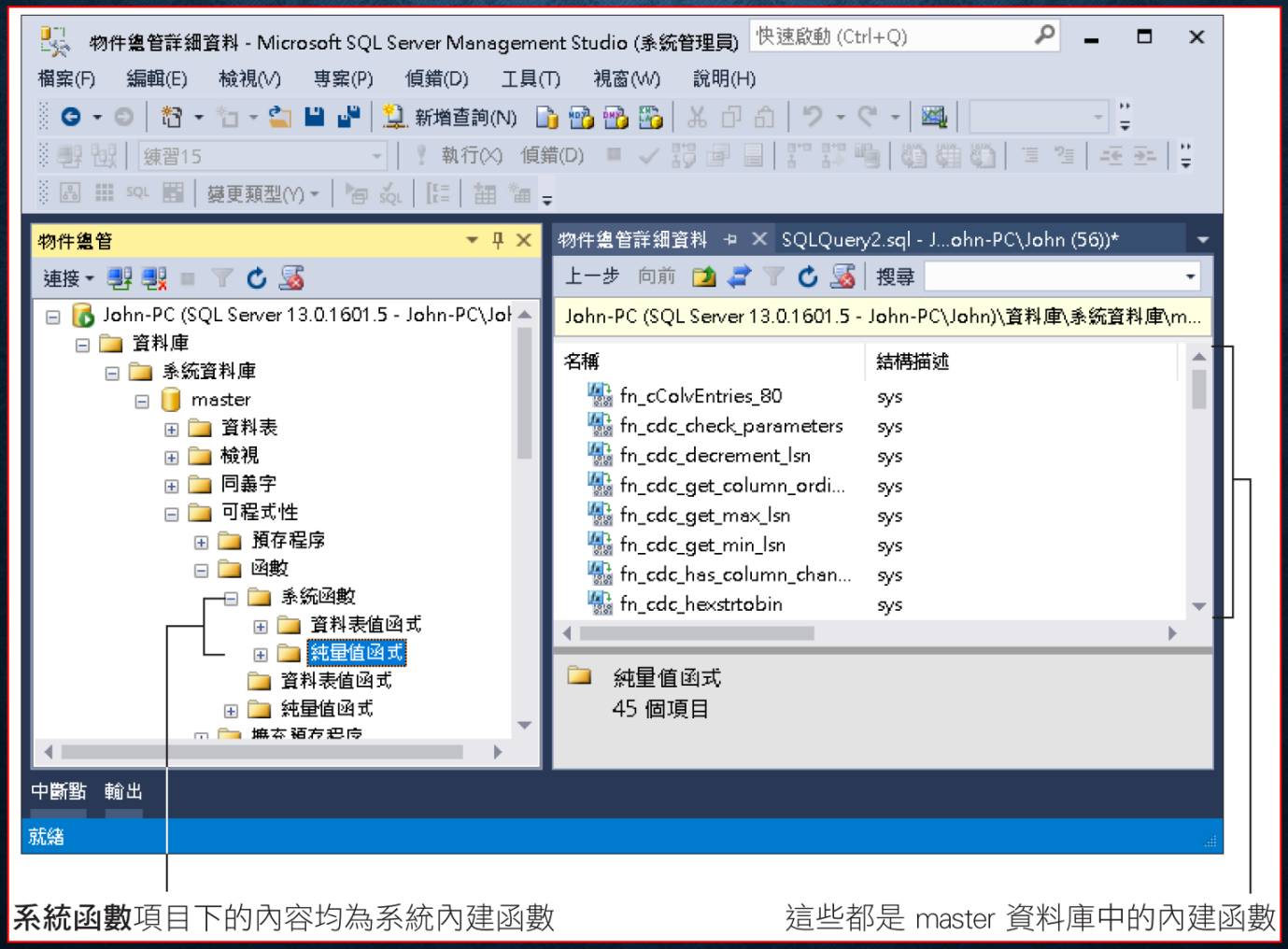
```
SELECT *
FROM ::fn_helpcollations()
```

← 此為系統資料庫中的內建函數

	name	description
1	Albanian_BIN	Albanian, binary sort
2	Albanian_BIN2	Albanian, binary code point comparison sort
3	Albanian_CI_AI	Albanian, case-insensitive, accent-insensitive, ...
4	Albanian_CI_AI_WS	Albanian, case-insensitive, accent-insensitive, ...
5	Albanian_CI_AI_KS	Albanian, case-insensitive, accent-insensitive, ...
6	Albanian_CI_AI_KS_WS	Albanian, case-insensitive, accent-insensitive, ...
7	Albanian_CI_AS	Albanian, case-insensitive, accent-sensitive, ka...
8	Albanian_CI_AS_WS	Albanian, case-insensitive, accent-sensitive, ka...

傳回各國語系的設定值及說明

內嵌資料表值函數



系統函數項目下的內容均為系統內建函數

這些都是 master 資料庫中的內建函數

多重陳述式資料表值函數

```
CREATE FUNCTION function_name
( [ {@parameter_namescalar_data_type [=default ] [READONLY]} [ ,...n] ] )
RETURNS @return_variable TABLE ( <table_type_definition> )
[WITH <function_option> [[, ]...n]]
[AS]
BEGIN
   function_body
   RETURN
END

<function_option> ::=
   { ENCRYPTION | SCHEMABINDING }

< table_type_definition> ::=
   { column_definition | table_constraint } [,...n]
```

多重陳述式資料表值函數

- **RETURNS @return_variable TABLE <table_type_definition>**
- **RETURN**

多重陳述式資料表值函數

	書籍編號	書籍名稱	單價	負責人
1	1	AutoCAD 操作入門	350.00	3
2	2	Linux 使用手冊	420.00	3
3	3	Excel 使用手冊	450.00	1
4	4	Internet 上線實務	320.00	3
5	5	Internet 精緻之旅	320.00	2
6	6	Flash 學習手冊	480.00	2
7	7	PhotoShop 細說從頭	980.00	2
8	8	PowerPoint 使用手冊	450.00	1
9	9	Visual Basic 學習手冊	350.00	3
10	10	Windows 使用手冊	450.00	1

書籍資料表

	員工編號	姓名	性別	主管員工編號	職稱	區域
1	1	張瑾雯	女	0	經理	NULL
2	2	陳季暄	男	0	經理	NULL
3	3	趙飛燕	女	0	經理	NULL
4	4	李美麗	女	1	銷售員	北區
5	5	劉天王	男	3	銷售員	北區
6	6	黎國明	男	3	銷售員	中區
7	7	郭國強	男	2	銷售員	南區
8	8	蘇涵蘊	女	1	銷售員	中區
9	9	孟庭亭	女	2	銷售員	北區
10	10	賴俊良	男	1	銷售員	南區
11	11	何大樓	男	3	銷售員	南區
12	12	王大德	男	2	銷售員	中區
13	13	楊大頭	男	NULL	NULL	NULL

員工資料表

```
/* 找出所有負責銷售指定書籍的相關人員 */
CREATE FUNCTION GetEmployeeFromProdId
(@ProductId int)
RETURNS @Employee TABLE
            ( 員工編號 int NOT NULL,
              姓名 varchar (20) NOT NULL,
              性別 char (2),
              主管員工編號 int,
              職稱 varchar (10),
              區域 varchar (10))
BEGIN
    /* 將產品負責人的資料加入要傳回的 @Employee 中*/
    INSERT @Employee
    SELECT 員工.*
    FROM 員工 JOIN 書籍 ON 員工編號 = 負責人
    WHERE 書籍編號 = @ProductId

    /* 將負責人的員工編號存入 @id 中 */
    DECLARE @id int
    SELECT @id  = 員工編號
    FROM  @Employee

    /* 將負責人的直屬銷售員也加入要傳回的 @Employee 中*/
    INSERT @Employee
    SELECT *
    FROM 員工
    WHERE 主管員工編號 = @id

    RETURN
END
```

多重陳述式資料表值函數

```
SELECT 姓名
FROM GetEmployeeFromProdId(6)
WHERE 區域 = '北區'
GO
```

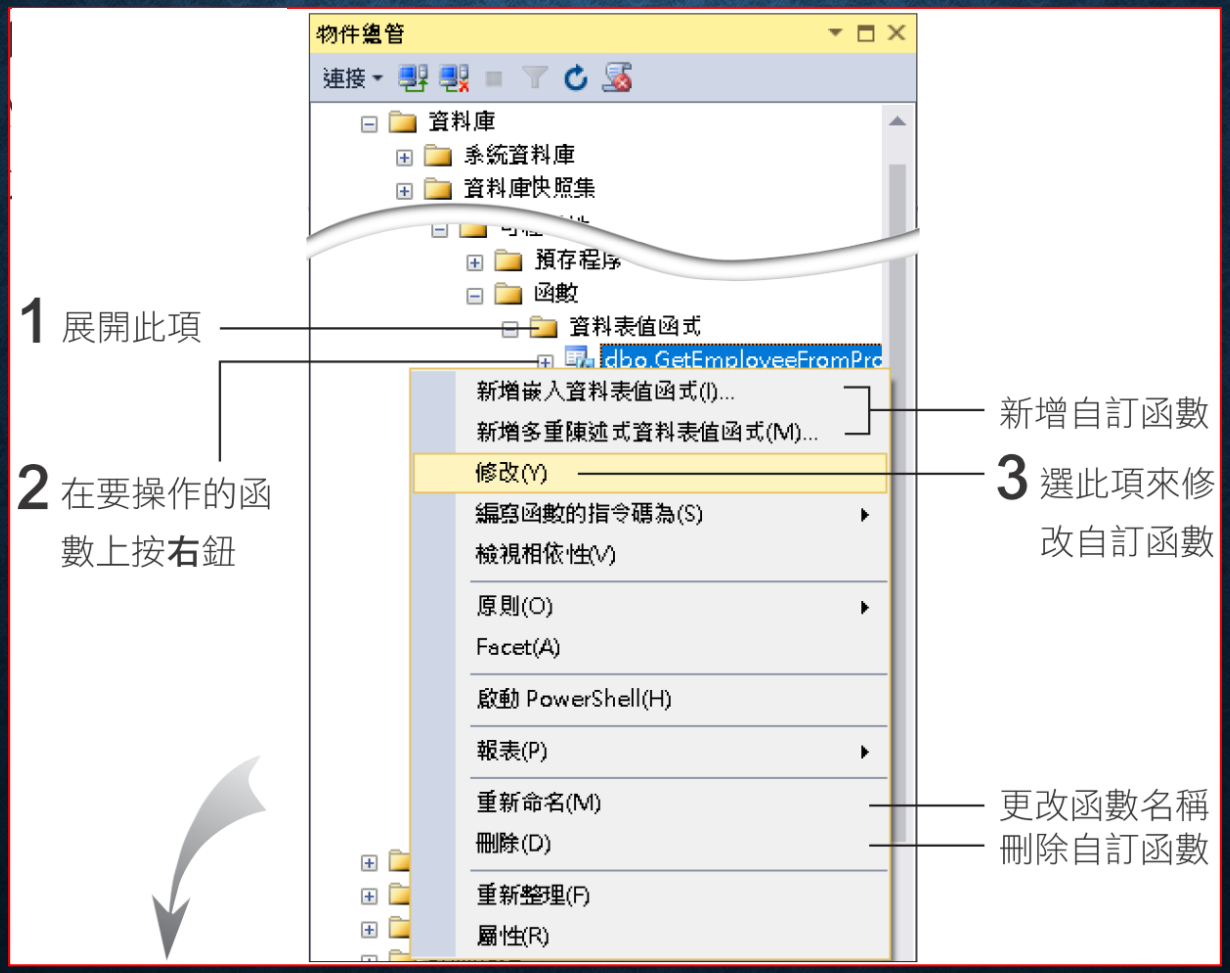
	員工編號	姓名	性別	主管員工編號	職稱	區域
1	2	陳季暄	男	0	經理	NULL
2	7	郭國珮	男	2	銷售員	南區
3	9	孟庭亭	女	2	銷售員	北區
4	12	王大德	男	2	銷售員	中區

所有負責銷售該書的主管與員工

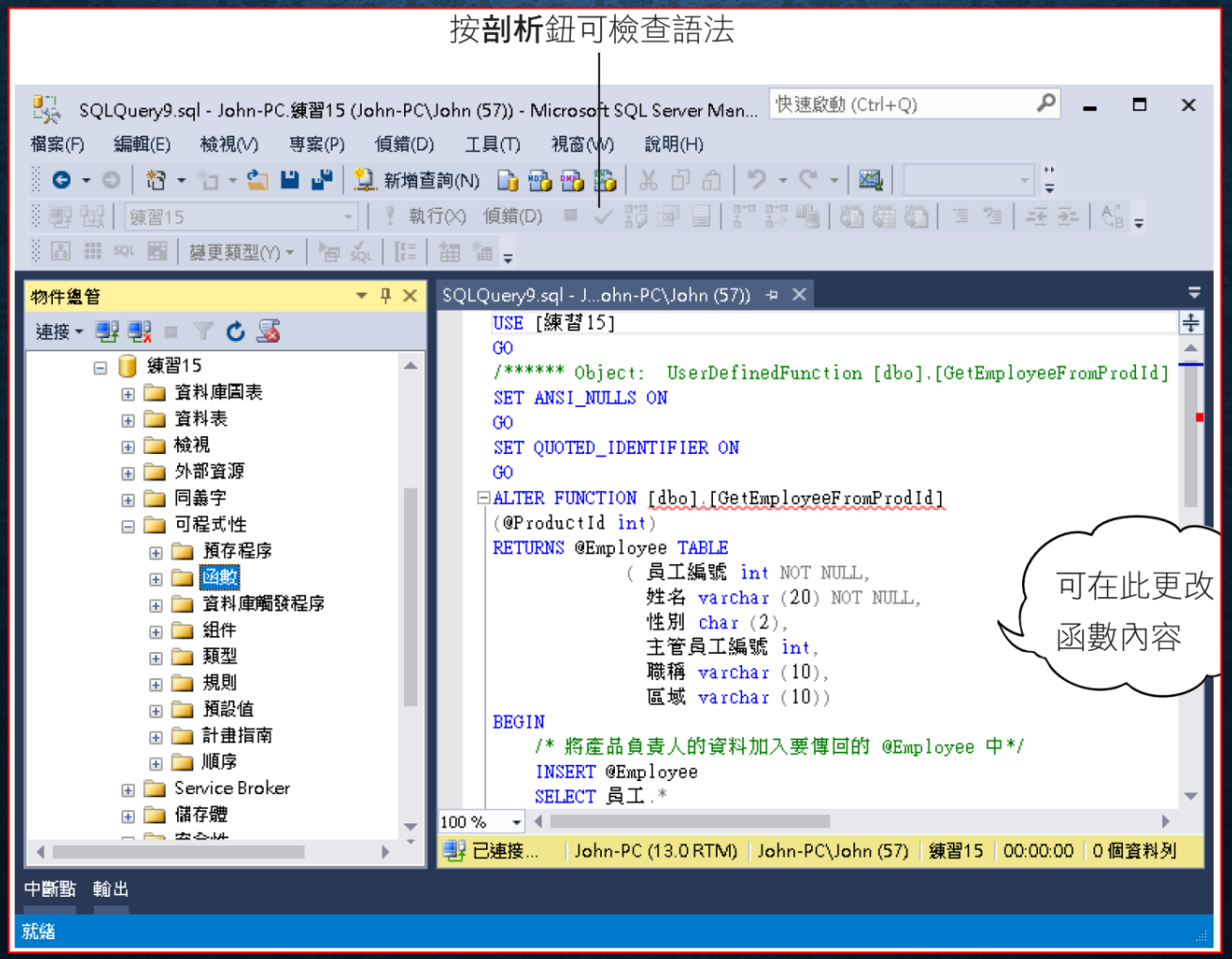
	姓名
1	孟庭亭

該書的北區負責人員

如何建立、修改、或刪除自訂函數

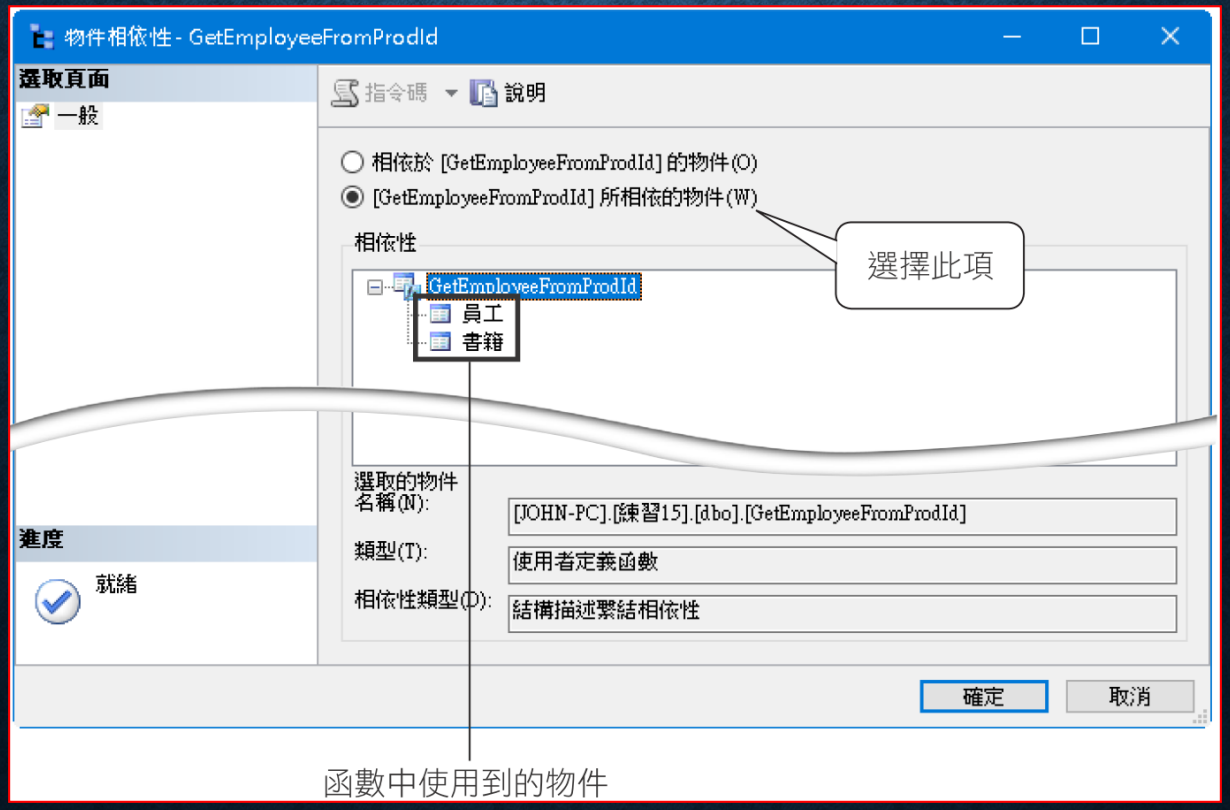


如何建立、修改、或刪除自訂函數



自訂函數物件相依性

- 自訂函數按右鍵執行『檢視相依性』



15-3 自訂函數的使用技巧

- 執行自訂函數時的注意事項
- 在資料表的設定中使用函數
- 在自訂函數中使用敘述的限制
- 決定性與非決定性函數
- 設定 **SCHEMABINDING** 時的限制

執行自訂函數時的注意事項

- 執行純量函數時，不可省略結構描述，傳入的參數量要依規定，有預設值，也必須使用 **Default**。

```
CREATE FUNCTION Calc
(@a int , @b int =3)              ◄── @b 有預設值
RETURNS int
BEGIN
    RETURN @a + @b
END
GO

SELECT dbo.Calc(5, 6)             ◄── OK
SELECT dbo.Calc(5, Default)       ◄── OK
GO
SELECT dbo.Calc(5)                ◄── 錯誤
```

	(沒有資料行名稱)
1	11

	(沒有資料行名稱)
1	8

(1 個資料列受到影響)
(1 個資料列受到影響)
訊息 313，層級 16 ，狀態 2 ，行 1
提供給程序或函數 dbo.Calc 的引數數量不足。

在資料表的設定中使用函數

出貨記錄資料表的編號為文字式

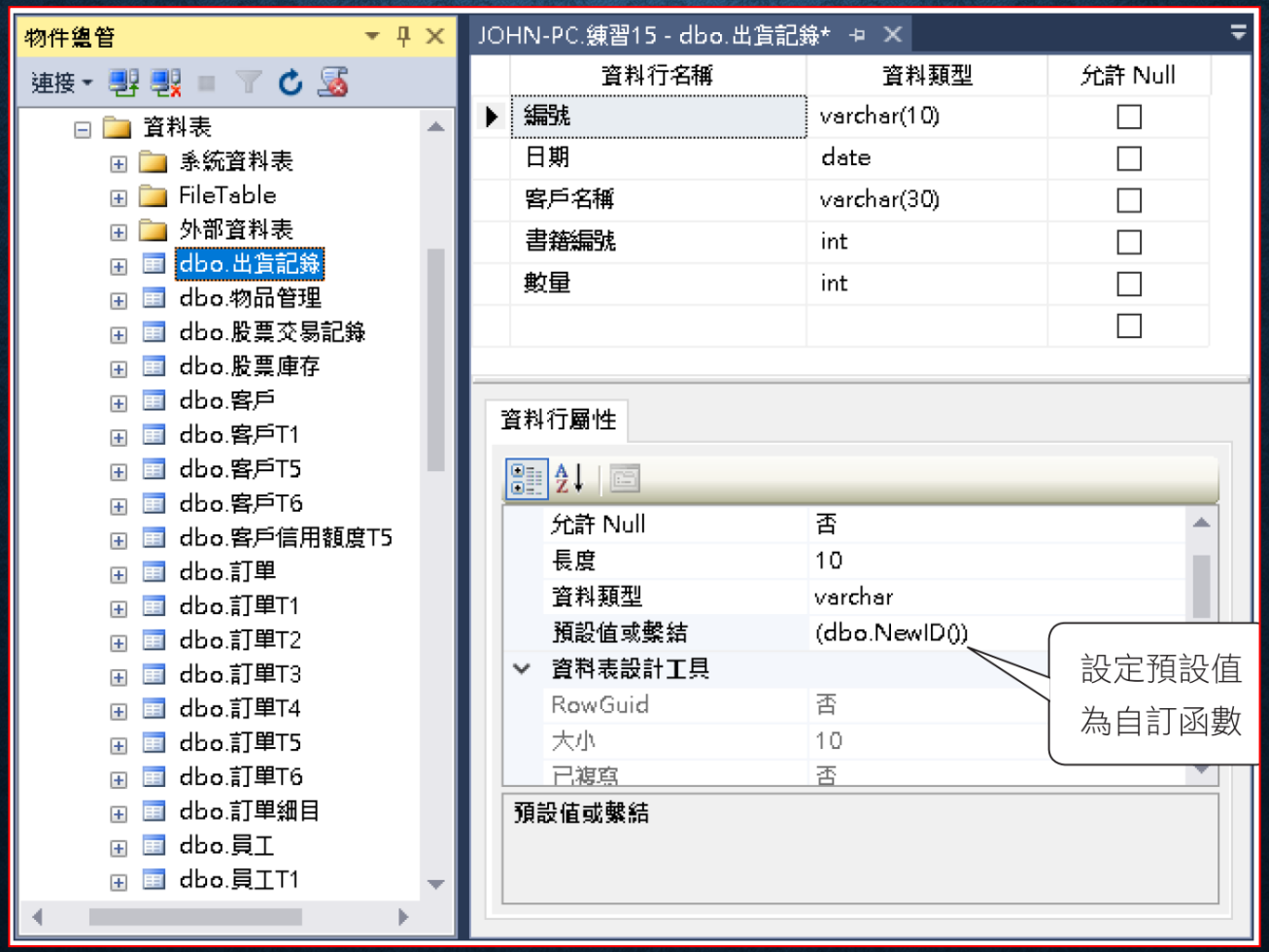
	編號	日期	客戶名稱	書籍編號	數量
1	A0001	2016-12-01	天天書局	2	10
2	A0002	2016-12-02	天天書局	5	5
3	A0003	2016-12-02	大雄書局	12	7
4	A0004	2016-12-02	大雄書局	6	2
5	A0005	2016-01-05	天天書局	3	6
6	A0006	2017-01-10	大雄書局	5	8
7	A0007	2017-02-20	大雄書局	11	2
8	A0008	2017-02-25	大雄書局	1	6

```
CREATE FUNCTION NewID()
RETURNS varchar(5)
BEGIN
    DECLARE @id varchar(5), @i int

    / * 找出目前最大的編號 * /
    SELECT TOP 1 @id = 編號
    FROM 出貨記錄
    ORDER BY 編號 DESC

    IF @@ROWCOUNT = 0      /* 如果沒有記錄 */
         RETURN 'A0001'
    SET @i = CAST(RIGHT(@id, 4) AS int) + 1
    SET @id = CAST(@i AS varchar)
    RETURN 'A' + REPLICATE('0', 4-LEN(@id)) + @id
END
GO

SELECT dbo.NewID()
```

	(沒有資料行名稱)
1	A0009

找出下一個號碼了

在資料表的設定中使用函數



在資料表的設定中使用函數

	書籍編號	書籍名稱	單價	負責人	封面照片	類別編號
1	1	AutoCAD 操作入門	350.00	3	0x151C2D000200...	1
2	2	Linux 使用手冊	420.00	3	0x151C2D000200...	1
3	3	Excel 使用手冊	450.00	1	0x151C2D000200...	1
4	4	Internet 上線實務	320.00	3	0x151C2D000200...	2
5	5	Internet 精緻之旅	320.00	2	0x151C2D000200...	3
6	6	Flash 學習手冊	400.00	2	0x151C2D000200...	1
7	7	PhotoShop 細說從頭	980.00	2	NULL	4
8	8	PowerPoint 使用手冊	450.00	1	NULL	1

書籍資料表

	類別編號	類別名稱	折扣	備註
1	1	入門	0.8	NULL
2	2	實例	0.75	NULL
3	3	技巧	0.9	NULL
4	4	技術	0.88	NULL

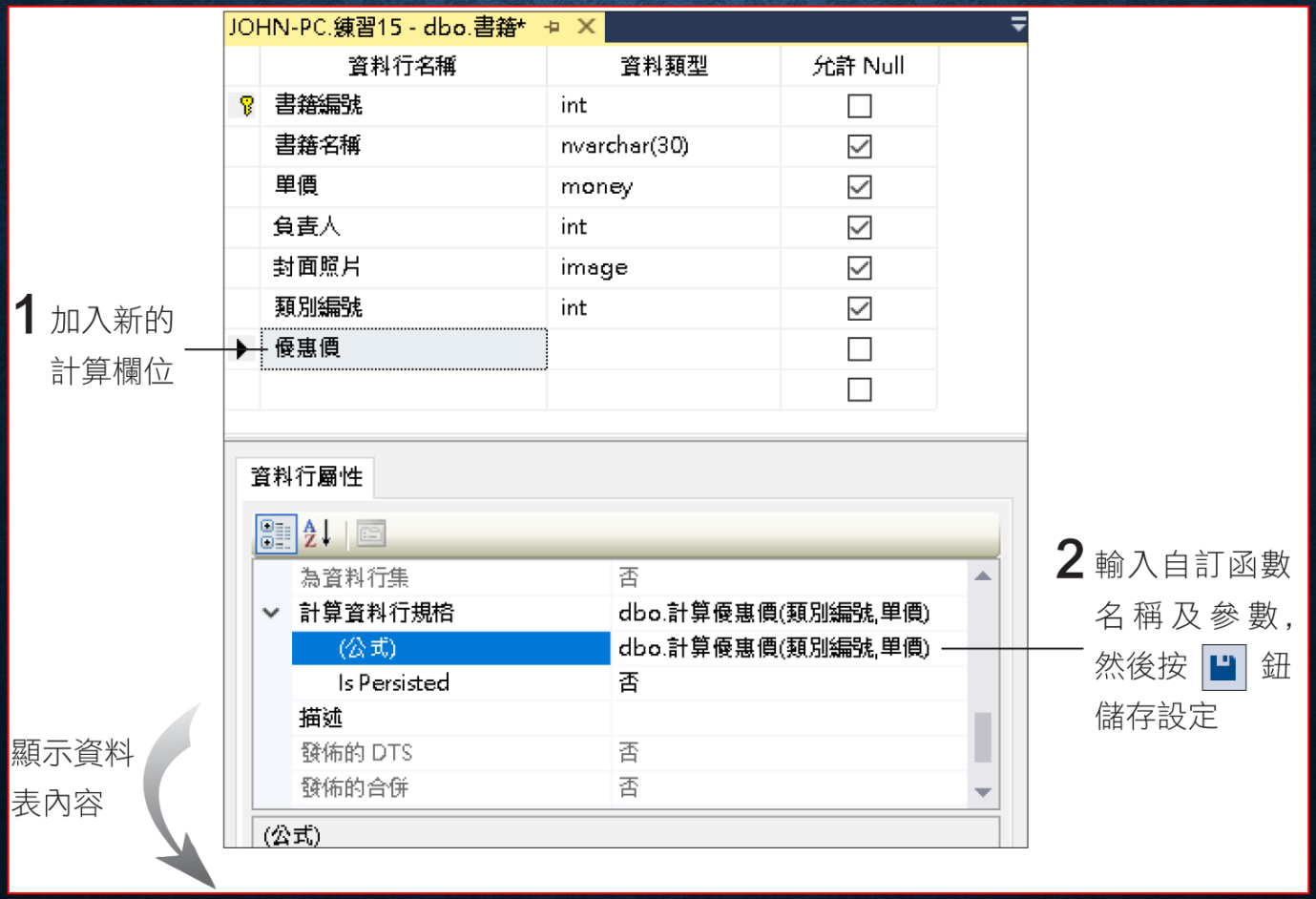
書籍類別資料表

```
CREATE FUNCTION 計算優惠價
(@ 類別編號 int , @ 單價 money)
RETURNS money
BEGIN
    DECLARE @優惠價 money
    SELECT @ 優惠價 = 折扣 * @ 單價
    FROM 書籍類別
    WHERE 類別編號 = @ 類別編號
    RETURN @優惠價
END
GO

SELECT dbo.計算優惠價(1, 20)
```

	(沒有資料行名稱)
1	16.00

類別 1 是打 8 折

在資料表的設定中使用函數



在資料表的設定中使用函數

	書籍編號	書籍名稱	單價	負責人	封面照片	類別編號	優惠價
1	1	AutoCAD 操作入門	350.00	3	0x151C2D00020000000B0...	1	280.00
2	2	Linux 使用手冊	420.00	3	0x151C2D00020000000B0...	1	336.00
3	3	Excel 使用手冊	450.00	1	0x151C2D00020000000B0...	1	360.00
4	4	Internet 上線實務	320.00	3	0x151C2D00020000000B0...	2	240.00
5	5	Internet 精緻之旅	320.00	2	0x151C2D00020000000B0...	3	288.00
6	6	Flash 學習手冊	400.00	2	0x151C2D00020000000B0...	1	320.00
7	7	PhotoShop 細說從頭	980.00	2	NULL	4	862.40
8	8	PowerPoint 使用手冊	450.00	1	NULL	1	360.00
9	9	Visual Basic 學習手冊	350.00	3	NULL	1	280.00
10	10	Windows 使用手冊	450.00	1	NULL	1	360.00
11	11	Word 使用手冊	450.00	2	NULL	1	360.00
12	12	WWW 實用寶典	350.00	2	NULL	3	315.00
13	13	遊戲程式設計	350.00	3	NULL	4	308.00
14	14	電腦低階應用實務	320.00	1	NULL	2	240.00
15	15	計算機概論	480.00	2	NULL	1	384.00

計算欄位完成了！

在自訂函數中使用敘述的限制

- 用 **DECLARE** 敘述宣告要使用於函數中的變數，這些變數的有效範圍只限於函數中；當函數結束時，宣告的變數也會跟著消失。
- 可用 **SET** 敘述將資料指定給變數。
- 若使用 **SELECT** 敘述，則必須將查詢結果指定給變數。
- 可使用 **INSERT**、**UPDATE**、**DELETE** 敘述針對函數中的 **table** 變數做處理。

在自訂函數中使用敘述的限制

- 可使用流程控制敘述，如 **IF**、**WHILE** 等，但禁用 **TRY...CATCH** 敘述。
- 可用 **EXECUTE** 敘述來執行延伸預存程序。其他預存程序或 **SQL** 字串則不可執行。
- 可在函數中建立 **CURSOR** 、前後移動 **CURSOR** 中的記錄位置、或使用 **FETCH...INTO** 將資料存入變數中。

在自訂函數中使用敘述的限制

不需死背前述各項可用的敘述，只需掌握以下個原則：

- 在函數中不可更改到資料庫的任何設計、設定、或資料表內容。我們只能擷取、計算資料，或是修改在函數中宣告的變數內容。
- 所執行的敘述不可直接傳回資料集給應用程式，例如不可用**“SELECT** 姓名 **FROM …”**，而應改為**“SELECT @var=** 姓名 **FROM …"** 將查詢結果存入變數中。
- 禁止呼叫以下非決定性系統內建函數：

NEWID	RAND
NEWSEQUENTIALID	TEXTPTR

決定性與非決定性函數

- 決定性函數：例如 **UPPER()**
- 非決定性函數：例如 **GETDATE()**
- 系統會決定自定函數的『決定性』或『非決定性』屬性。

決定性函數

- 自訂函數已設定了 **SCHEMABINDING** 選項。
- 在自訂函數中沒有呼叫任何『非決定性』的自訂函數或內建函數。
- 在自訂函數中沒有使用到任何的資料庫物件 **(**但在函式中宣告的 **table** 變數除外**)** ，例如資料表、檢視表等。
- 自訂函數中沒有呼叫任何的延伸預存程序。

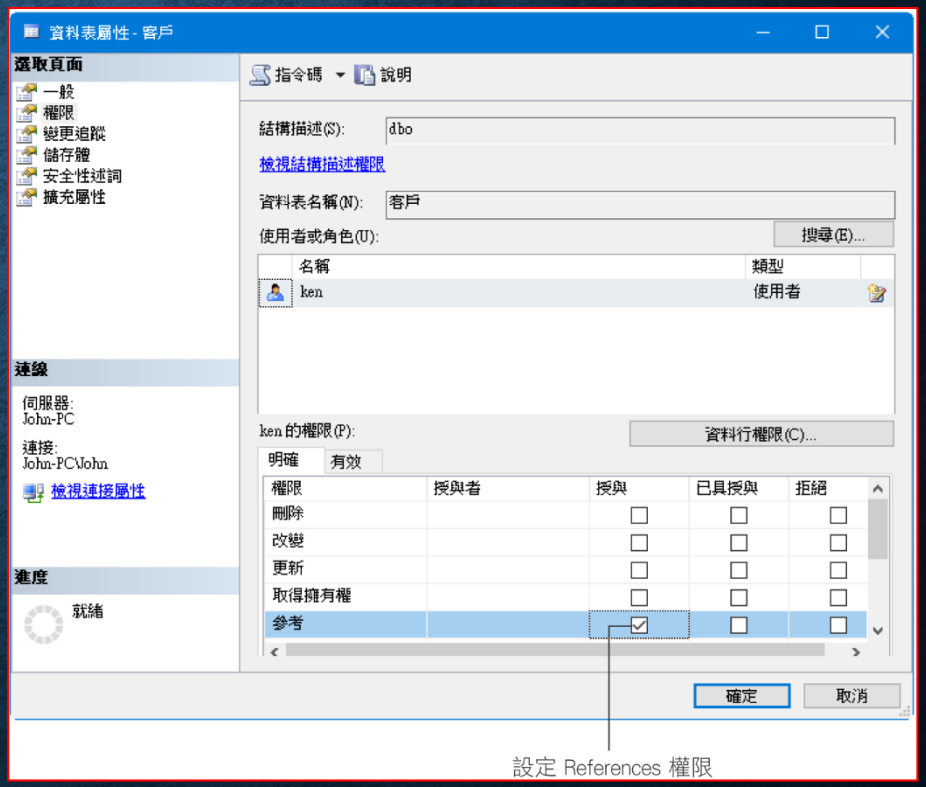
決定性與非決定性函數的差異

- 當資料表或檢視表的計算欄位中有使用到『非決定性』函數時，該欄位將不允許用來建立索引。
- 當檢視表中有使用到『非決定性』函數時，則該檢視表將不允許建立叢集索引**(Clustered index)**。

設定 SCHEMABINDING 時的限制

- 在自訂函數中所使用到的檢視表及其他自訂函數，都必須是已經設定為 **SCHEMABINDING** 的。
- 自訂函數與其中使用到的所有物件，都必須位於同一個資料庫內。
- 建立函數的人，必須對函數中使用到的所有物件 (資料表、檢視表、及自訂函數) 擁有 **References** 權限：

註：未符合以上條件，**SCHEMABINDING** 的自訂函數將無法建立。



15-4 使用順序物件

- 『順序』(Sequence) 是自動編號機制，更方便取得各種序號。

```
CREATE SEQUENCE Seq123          ←— 建立 Seq123 順序物件
START WITH 1                    ←— 由 1 開始給號
INCREMENT BY 1                  ←— 每次 +1

CREATE SEQUENCE Seq246          ←— 建立 Seq246 順序物件
START WITH 2                    ←— 由 2 開始給號
INCREMENT BY 2                  ←— 每次 +2


SELECT NEXT VALUE FOR Seq123 AS 順序 1,   ←— 由 Seq123 取號
       NEXT VALUE FOR Seq246 AS 順序 2,   ←— 由 Seq246 取號
       產品名稱
FROM   旗旗公司
```

	順序1	順序2	產品名稱
1	1	2	Windows 使用手冊
2	2	4	Linux 架站實務
3	3	6	JAVA 程式語言
4	4	8	PHP程式語言

246...給號

123...給號

使用順序物件

```
SELECT NEXT VALUE FOR Seq123 AS 順序 1,
       NEXT VALUE FOR Seq246 AS 順序 2,
       產品名稱
FROM   標標公司                    ← 查詢另一個資料表 (標標公司)
```

	順序1	順序2	產品名稱
1	5	18	Windows 使用手冊
2	6	12	Linux 架站實務
3	7	14	SQL 指令寶典
4	8	16	組合語言
5	9	18	MATHLAB 手

會繼續給號

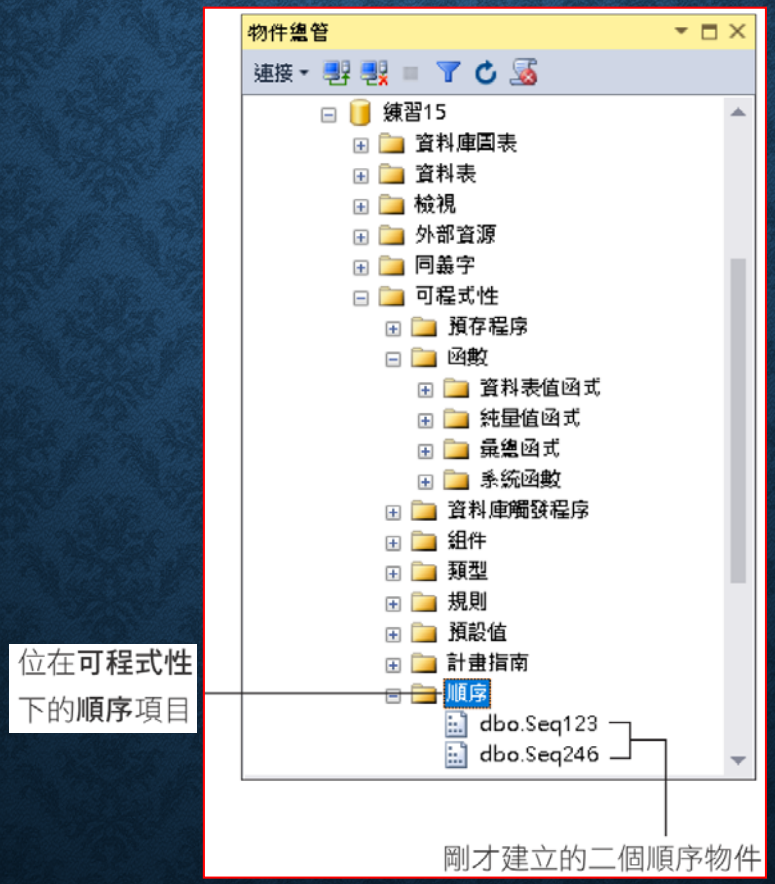
```
ALTER SEQUENCE Seq123              ← 修改 Seq123 的設定
RESTART WITH -2                    ← 重設開始值為 -2
```

```
SELECT NEXT VALUE FOR Seq123 AS 順序 1,
       NEXT VALUE FOR Seq246 AS 順序 2,
       產品名稱
FROM   標標公司
```

	順序1	順序2	產品名稱
1	-2	28	Windows 使用手冊
2	-1	22	Linux 架站實務
3	0	24	SQL 指令寶典
4	1	26	組合語言
5	2	28	MATHLAB 手

由 -2 開始給號

使用順序物件


物件總管
連接
練習15
資料庫圖表
資料表
檢視
外部資源
同義字
可程式性
預存程序
函數
資料表值函式
純量值函式
彙總函式
系統函數
資料庫觸發程序
組件
類型
規則
預設值
計畫指南
順序
dbo.Seq123
dbo.Seq246
位在可程式性
下的順序項目
剛才建立的二個順序物件

使用順序物件

- 用 **SQL** 語法建立順序物件
- 用 **SQL** 語法修改順序物件或重新開始編號
- 用 **SQL** 語法刪除順序物件
- 在 **SSMS** 中新增、修改、更名、刪除順序物件
- **NEXT VALUE FOR** 的使用技巧

用 SQL 語法建立順序物件

```
CREATE SEQUENCE sequence_name
[ AS integer_type ]
[ START WITH <constant> ]
[ INCREMENT BY <constant> ]
[ { MINVALUE [ <constant> ] } | { NO MINVALUE } ]
[ { MAXVALUE [ <constant> ] } | { NO MAXVALUE } ]
[ CYCLE | { NO CYCLE } ]
[ { CACHE [ <constant> ] } | { NO CACHE } ]
```

用 SQL 語法建立順序物件

- **CREATE SEQUENCE sequence_name**

  建立順序物件，在 **sequence_name** 中也可包含結構描述 **(Schema)**。

- **AS integer_type**

  指定順序物件的數值型別，必須為下表中的整數型別，或是小數位數為 **0** 的 **numeric(decimal)**。

整數型別	數值範圍
tinyint	0 到 255
smallint	-32, 768 到 32, 767
int	-2, 147, 483, 648 到 2, 147, 483, 647
bigint	-9, 223, 372, 036, 854, 775, 808 到 9, 223, 372, 036, 854, 775, 807

用 SQL 語法建立順序物件

- START WITH <constant>
- INCREMENT BY <constant>
- MINVALUE <constant>、MAXVALUE <constant>
- CYCLE | { NO CYCLE }
- CACHE [<constant>] | NO CACHE

順序物件的範例

```
CREATE SEQUENCE Seq0123
AS tinyint                ←— 使用 tinyint 型別
START WITH 1              ←— 開始值為 1
MAXVALUE 3                ←— 最大值為 3
CYCLE                     ←— 要循環編號

SELECT NEXT VALUE FOR Seq0123 AS 循環編號, 產品名稱
FROM 標標公司
```

由 1 開始編號

	循環編號	產品名稱
1	1	Windows 使用手冊
2	2	Linux 架站實務
3	3	SQL 指令寶典
4	0	組合語言
5	1	MATHLAB 手

編到最大值了

由最小值開始循環編號

用 SQL 語法修改順序物件或重新開始編號

1. 沒有 **AS integer_type**：
2. 須將 **START** 改為 **RESTART**：

```
ALTER SEQUENCE sequence_name
[ RESTART [ WITH <新的開始值> ] ]
[ INCREMENT BY <constant> ]
[ { MINVALUE [ <constant> ] } | { NO MINVALUE } ]
[ { MAXVALUE [ <constant> ] } | { NO MAXVALUE } ]
[ CYCLE | { NO CYCLE } ]
[ { CACHE [ <constant> ] } | { NO CACHE } ]
```

用 SQL 語法修改順序物件或重新開始編號

```
ALTER SEQUENCE Seq0123
RESTART                          ← 由原來的開始值重新編號

SELECT NEXT VALUE FOR Seq0123 AS 循環編號

ALTER SEQUENCE Seq0123
RESTART WITH 3                   ← 由新的開始值重新編號

SELECT NEXT VALUE FOR Seq0123 AS 循環編號
```

	循環編號
1	1

由原來的開始值重新編號

	循環編號
1	3

由新的開始值重新編號

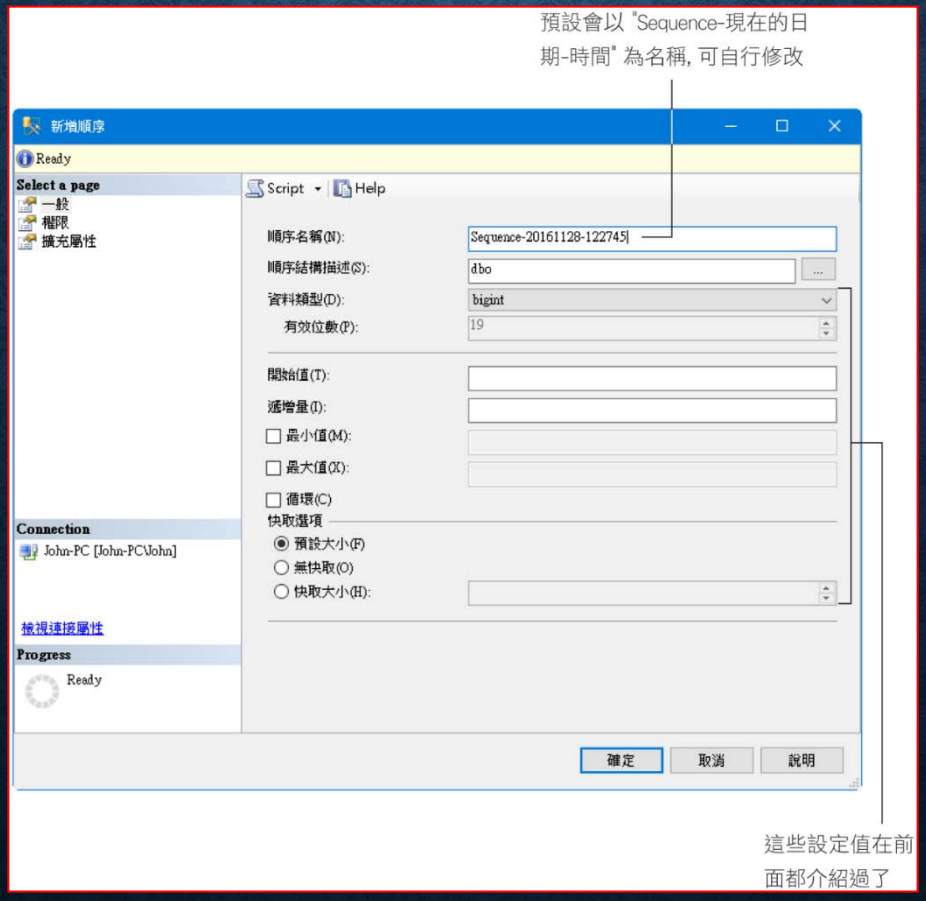
用 SQL 語法刪除順序物件

```
DROP SEQUENCE sequence_name [ , ...n ]
```

```
DROP SEQUENCE Seq246, Seq0123
```

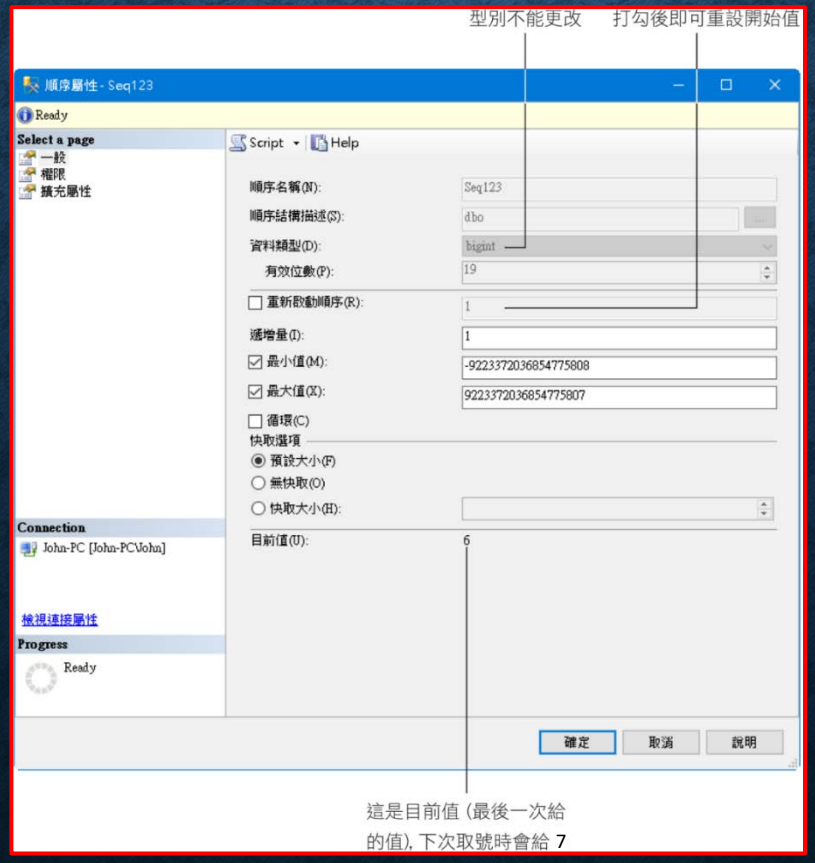
在 SSMS 中新增順序物件

- 在順序項目上按右鍵執行『新增順序』：



在 SSMS 中修改順序物件

- 在順序項目上按右鍵執行『屬性』：



NEXT VALUE FOR 的使用技巧

- 在 **DECLARE**、**SET**、或 **SELECT** 子句中設定變數的值：

```
ALTER SEQUENCE Seq123
RESTART WITH 1                    ← 由 1 開始重新編號

DECLARE @var bigint = NEXT VALUE FOR Seq123
PRINT @var
SET @var = NEXT VALUE FOR Seq123
PRINT @var
SELECT @var = NEXT VALUE FOR Seq123
PRINT @var
```

```
1
2
3
```

NEXT VALUE FOR 的使用技巧

- 使用 **SELECT...INTO** 或 **INSERT...VALUES** 來新增記錄：

```
SELECT NEXT VALUE FOR seq123 AS 序號, 產品名稱, 價格
INTO 大大公司
FROM 旗旗公司 WHERE 價格 = 500

INSERT 大大公司
VALUES (NEXT VALUE FOR seq123, '資料庫實務', 550)

SELECT * FROM 大大公司
```

	序號	產品名稱	價格
1	4	Linux 架站實務	500.00
2	5	PHP程式語言	500.00
3	6	資料庫實務	550.00

第 1 個敘述新增的 2 筆記錄

第 2 個敘述新增的記錄

seq123 的給號

NEXT VALUE FOR 的使用技巧

- 使用於 **UPDATE** 子句更改記錄：

```
UPDATE 大大公司
SET 序號 = NEXT VALUE FOR seq123
WHERE 價格 = 500

SELECT * FROM 大大公司
```

	序號	產品名稱	價格
1	7	Linux 架站實務	500.00
2	8	PHP程式語言	500.00
3	6	資料庫實務	550.00

這 2 筆改為新序號了

NEXT VALUE FOR 的使用技巧

- 用來做為欄位的預設值。底下將順序物件設為序號欄的預設值：

```
ALTER TABLE 大大公司
ADD DEFAULT NEXT VALUE FOR seq123 FOR 序號        ← 設定欄位預設值

INSERT 大大公司 (產品名稱, 價格)                    ← 新增 2 筆記錄
VALUES ('SQL 聖經', 780) , ('精通 SQL', 720)

SELECT * FROM 大大公司
```

	序號	產品名稱	價格
1	7	Linux 架站實務	500.00
2	8	PHP程式語言	500.00
3	6	資料庫實務	550.00
4	9	SQL聖經	780.00
5	10	精通SQL	720.00

新增記錄時, 由 seq123 給號的預設值

NEXT VALUE FOR 的使用技巧

- 順序物件用於 **SELECT** 敘述時若需要排序，不可使用 **ORDER BY** 子句來排序 **(會視為錯誤)**，而必須改用 **OVER (ORDER BY...)** 子句才行：

```
ALTER SEQUENCE Seq123
RESTART WITH 1

SELECT NEXT VALUE FOR seq123 OVER(ORDER BY 價格 DESC) AS 價格排名, *
FROM 大大公司
```

	價格排名	序號	產品名稱	價格
1	1	9	SQL聖經	780.00
2	2	10	精通SQL	720.00
3	3	6	資料庫實務	550.00
4	4	7	Linux 架站實務	500.00
5	5	8	PHP程式語言	500.00

依價格排名 (遞減排序並依序給號)

NEXT VALUE FOR 的使用技巧

- 如果想要向順序物件一次取得“連續的”多個編號，可改用 **sp_sequence_get_range** 預存程序來取號，例如：

```
DECLARE @第一值 sql_variant,     -- 宣告 2 個變數來儲存取得的序號
        @最後值 sql_variant

EXEC sp_sequence_get_range N'Seq123', 5,    -- 由 Seq123 取得 5 個連續編號
        @第一值 OUTPUT, @最後值 OUTPUT      -- 由參數傳回編號的第一值及最後值

SELECT @第一值 AS 第一值, @最後值 AS 最後值
```

	第一值	最後值
1	6	10

取得 5 個連續編號 (6~10) 了

NEXT VALUE FOR 不適用的場合

- 當資料庫在唯讀模式中。
- 不能做為資料表值函式、彙總函式的參數。
- 不能用於子查詢、檢視表、使用者自訂函數、或計算欄位中。
- 不能用在包含 **DISTINCT**、**UNION**、**UNION ALL** 的敘述中。
- 不能用在包含 **ORDER BY** 子句的敘述中 (但可以搭配 **OVER (ORDER BY...)** 子句來排序)。

NEXT VALUE FOR 不適用的場合

- 不能用在下列子句中： **FETCH**、**OVER**、**OUTPUT**、**ON**、**PIVOT**、**UNPIVOT**、**GROUP BY**、**HAVING**、或 **FOR XML**。
- 在使用 **CASE**、**CHOOSE**、**COALESCE**、**IIF**、**ISNULL** 或 **NULLIF** 的條件運算式中。
- 不能用在 **WHERE** 子句、或不屬於 **INSERT** 陳述式的 **VALUES** 子句中。
- 不能用在檢查條件約束的定義中 **(**但可以用在預設值條件約束的定義中**)**。

NEXT VALUE FOR 不適用的場合

- 不能用在規則物件或預設值物件的定義中，也不能做為使用者定義資料表類型的預設值 (有關規則物件、預設值物件、及使用者定義資料表類型，可參見附錄 B)。
- 不能用在包含 TOP、OFFSET 的敘述中，或是設定 ROWCOUNT 選項時。
- 不能用在 MERGE 敘述中 (除非 NEXT VALUE FOR 已用於目標資料表的預設值條件約束中，而且預設值用於 MERGE 陳述式的 CREATE 敘述中)。