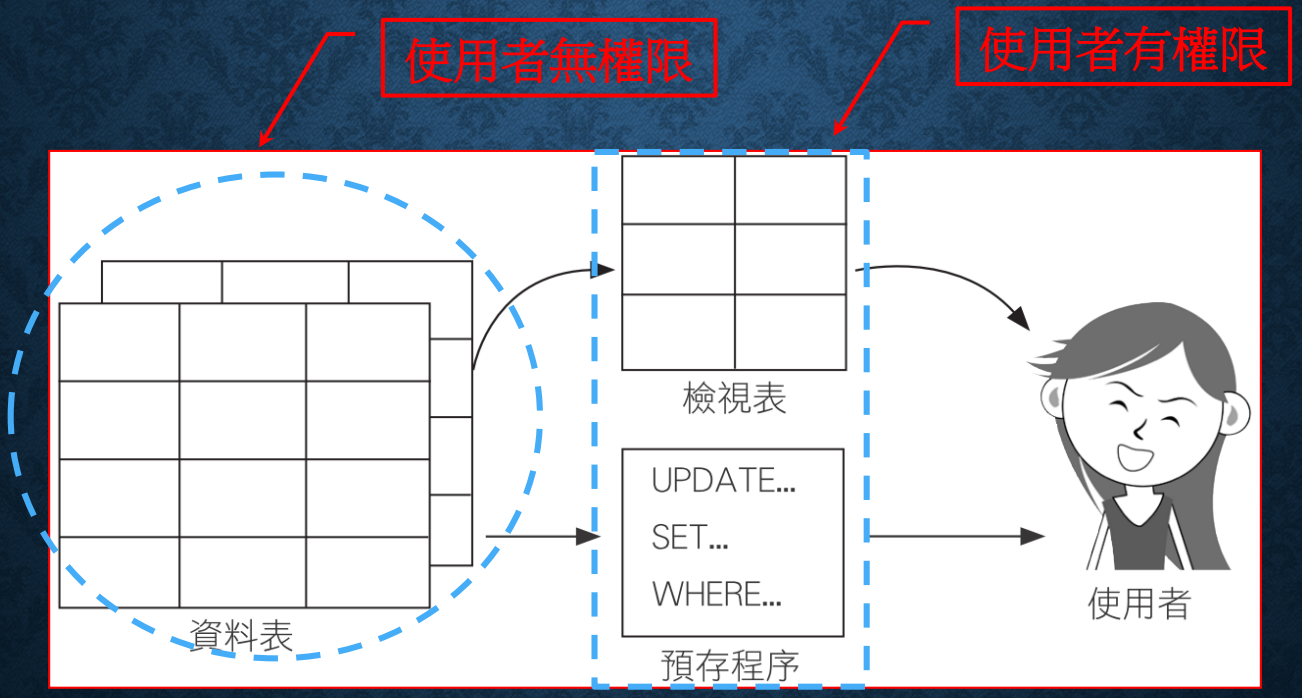
CHAP 14 預存程序

- 14-1 預存程序簡介
- 14-2 預存程序的建立、使用與修改
- 14-3 設計預存程序的技巧
- 14-4 使用 table 型別的參數

14-1 預存程序簡介

- 預存程序的優點
  - 執行效率高、統一的操作流程、重複使用、安全性
- 預存程序的種類
  - 系統預存程序、延伸預存程序、使用者自訂的預存程序

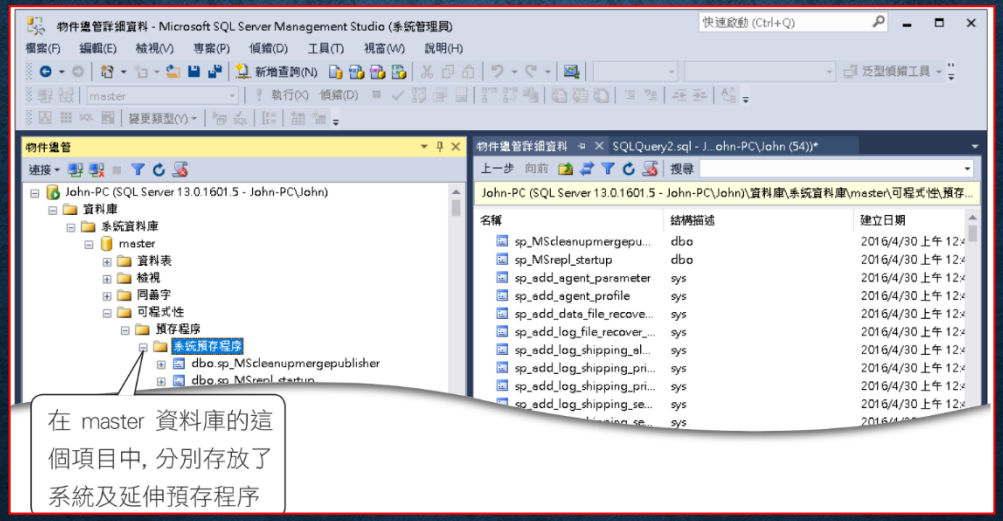
透過檢視表或預存程序，來存取沒有使用權限的資料表



預存程序內最好使用"結構描述.物件名稱"明確指定物件，使用者也要擁有開啟檢視表或執行預存程序的權限。

預存程序的種類

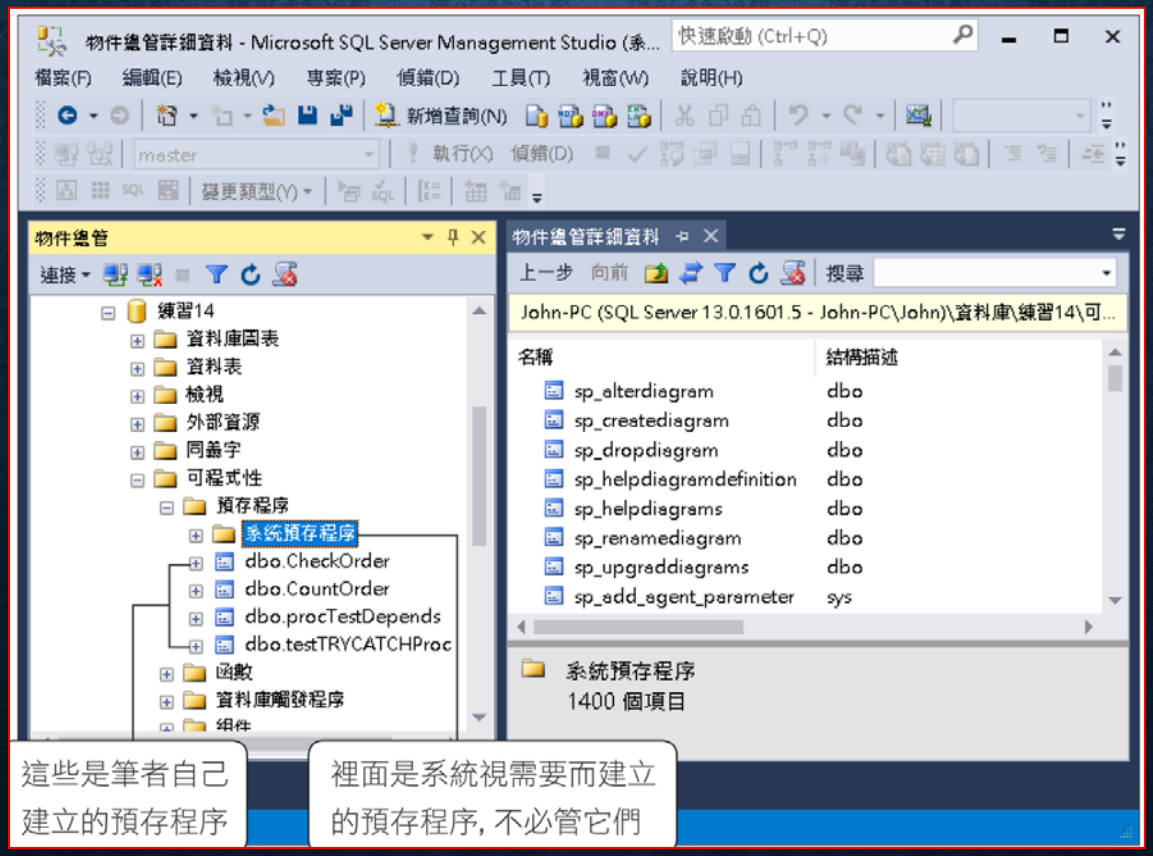
- 系統預存程序 **(System stored procedures)**：進行系統的各項設定、取得資訊或相關管理工作。



- 延伸預存程序 **(Extended stroed procedures)**：通常以 **xp_** 開頭，此類程序大多是以傳統的程式語言 **(例如C++)** 撰寫而成。其內容不是儲存在 **SQL Server**，而是以 **DLL** 的形式單獨存在。

預存程序的種類

- 使用者自訂的預存程序 **(User-defined stored procedures)**



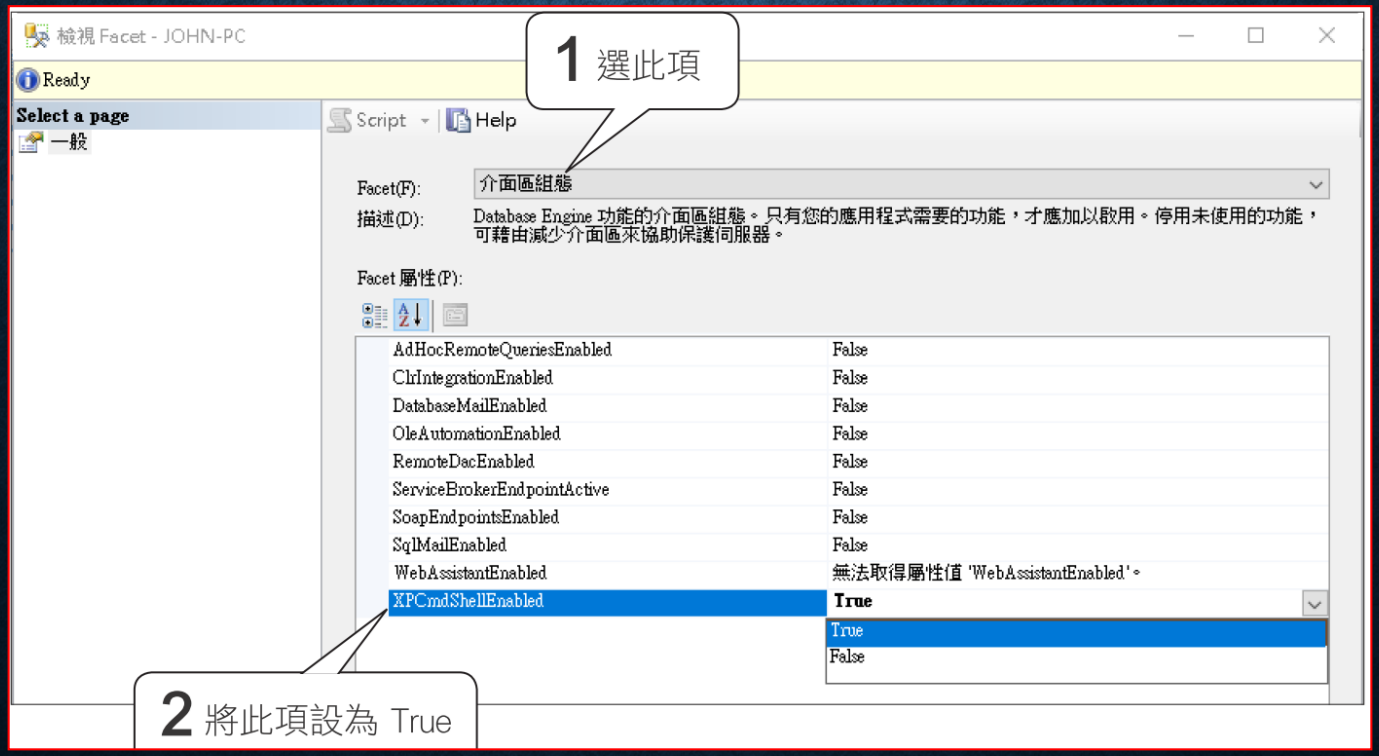
"SP_" 與 "XP_" 的特殊意義

```
                                    設定進階 (具危險性) 組態前, 必須先執行這行敘述
EXEC sp_configure ' show advanced options ', 1 ←┘
RECONFIGURE     ←── 更改組態後, 要執行此敘述 (或重新啟動伺服器) 才會生效
EXEC sp_configure 'xp_cmdshell', 1 ←── 1 表示允許 (True), 0 表示禁用
RECONFIGURE
```

在伺服器任何資料庫都可以執行以 "SP_" 開頭的預存程序，"XP_" 開頭的預存程序則必須明確地指明其所在的位置。

"SP_" 與 "XP_" 的特殊意義

- 也可以在 **SMSS** 中物件總管的伺服器名稱按右鍵，執行『 **Facet** 』進行設定。



"SP_" 與 "XP_" 的特殊意義

```
EXEC master..xp_cmdshell 'DIR C:\TEMP'
```
← 執行 DOS 的 DIR 命令

```
USE master
```
← 先切換到 master 資料庫

```
EXEC xp_cmdshell 'DIR C:\TEMP'
```
← 再執行延伸預存程序

	output
1	磁碟區 C 中的磁碟是 S3A4888D003
2	磁碟區序號: A49A-1A41
3	NULL
4	C:\TEMP 的目錄
5	NULL
6	2016/10/27 上午 02:41 <DIR> .
7	2016/10/27 上午 02:41 <DIR> ..
8	2016/09/14 下午 11:51 2,852,032 SQLServer...
9	1 個檔案 2,852,032 位元組
10	2 個目錄 148,966,780,928 位元組可用
11	NULL

14-2 預存程序的建立、使用與修改

- 用 **SQL** 語言建立預存程序
- 使用 **SQL Server Management Studio** 建立預存程序
- 更改預存程序的名稱
- 修改與刪除自訂預存程序
- 使用 **SQL Server Management Studio** 執行、管理預存程序

用 SQL 語言建立預存程序

```
CREATE PROC[EDURE] procedure_name [;number]
    [ @parameter data_type [VARYING] [= default] [OUTPUT] [READONLY] ]
    [, ...n]
    [WITH { RECOMPILE | ENCRYPTION | RECOMPILE, ENCRYPTION } ]
    [FOR REPLICATION]
AS sql_statement [...n]
```

VARYING：用在 **CURSOR** 的資料型別。
RECOMPLIE：每次執行前重新編譯，會依照現況產生最佳的執行計畫。
FOR REPLICATION： 只有資料複寫的時候才可執行。

用 SQL 語言建立預存程序

- **CREATE PROC[EDURE] procedure_name [;number]**
- **@parameter data_type [VARYING] [= default] [OUTPUT] [READONLY]**
- **WITH { RECOMPILE | ENCRYPTION | RECOMPILE, ENCRYPTION }**

```
DECLARE @地址 VARCHAR(100)
EXEC Lookup '楊小雄', @地址 OUTPUT
```

← 查詢到的地址可由 **@地址**傳回

```
EXEC MyProc WITH RECOMPILE
```

← 請求系統先重新編譯一次再執行

用 SQL 語言建立預存程序

```
CREATE PROCEDURE MyProc1                                    / * 建立預存程序 * /
AS SELECT * FROM 標標公司 WHERE 價格 > 400
GO
EXEC MyProc1                                                / * 執行預存程序 * /
```

	產品名稱	價格
1	Linux 架站實務	490.00
2	SQL 指令寶典	440.00

用 SQL 語言建立預存程序

```
CREATE PROCEDURE MyProc2                    / * 建立預存程序，並指定 2 個參數 * /
@param1 char(10), @param2 money
WITH ENCRYPTION
AS INSERT 標標公司 (產品名稱, 價格)
    VALUES (@param1, @param2)
GO
EXEC MyProc2 '組合語言', 520                / * 執行預存程序，並給定參數值 * /
GO
SELECT *
FROM   標標公司
```

	產品名稱	價格
1	Windows 使用手冊	400.00
2	Linux 架站實務	490.00
3	SQL 指令寶典	440.00
4	組合語言	520.00

新增的紀錄

用 SQL 語言建立預存程序

```
/* MyProc3 預存程序 */
CREATE PROCEDURE MyProc3
@param1 char(10), @param2 money, @param3 money OUTPUT
AS      INSERT 標標公司 (產品名稱, 價格)
        VALUES (@param1, @param2)
        SELECT @param3 = SUM(價格)
        FROM 標標公司
GO

/* MyProc4 預存程序 */
CREATE PROCEDURE MyProc4
@param1 money
AS PRINT '目前的總價為:' + CONVERT(varchar, @param1)
GO
```

此為可傳回值的參數

用 SQL 語言建立預存程序

```
DECLARE @sum money
EXEC MyProc3 'MATHLAB 手冊', 320, @sum OUTPUT
EXEC MyProc4 @sum
```

← 執行 MyProc3 預存程序並且將傳回值指定給 @sum

← 再將 MyProc3 的傳回值 @sum 傳給 MyProc4

```
( 1 個資料列受到影響)
目前的總價為: 2170.00
```

用 SQL 語言建立預存程序

```
CREATE PROCEDURE 取得客戶地址
@客戶編號 int ,
@地址 varchar(100) OUTPUT
AS  SELECT @地址 = 地址
    FROM 客戶
    WHERE 客戶編號 = @ 客戶編號

IF @@rowcount > 0
        RETURN 0        /* 如果查詢到則傳回 0 * /
   ELSE
        RETURN 1        /* 沒有查到就傳回 1 */
GO
```

用 SQL 語言建立預存程序

```
DECLARE @ret int, @地址 varchar(100)
EXEC @ret = 取得客戶地址 4, @地址 OUTPUT     /* 用 @ret 接收傳回值 */
IF @ret = 0
    PRINT @ 地址
ELSE
    PRINT '找不到！'
```

台北市南京東路三段 3 號

```
/ * 建立 MyProc5 預存程序群組的第 1 個程序 * /

CREATE PROCEDURE MyProc5;1
AS
   SELECT *
   FROM 旗旗公司
GO
```

用 SQL 語言建立預存程序

```
/* 建立 MyProc5 預存程序群組的第 2 個程序 * /

CREATE PROCEDURE MyProc5;2
AS
SELECT *
FROM 標標公司
GO

MyProc5;1                    / * 執行群組中的預存程序要指定其編號 * /

EXEC MyProc5;2               / * 執行第 2 個預存程序 * /
```

執行預存程序在批次的最前面時，可以不使用**EXECUTE**。

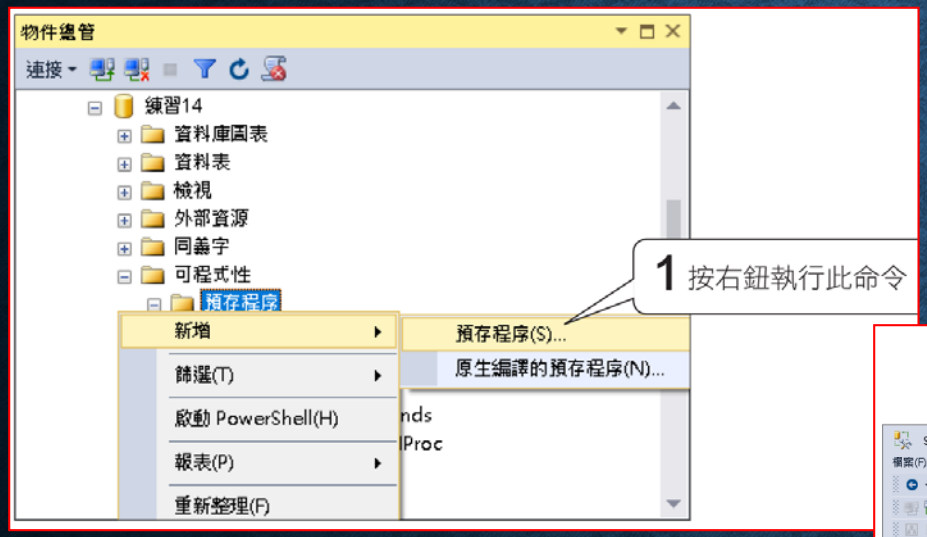
	產品名稱	價格
1	Windows 使用手冊	400.00
2	Linux 架站實務	500.00
3	JAVA 程式語言	420.00
4	PHP程式語言	500.00

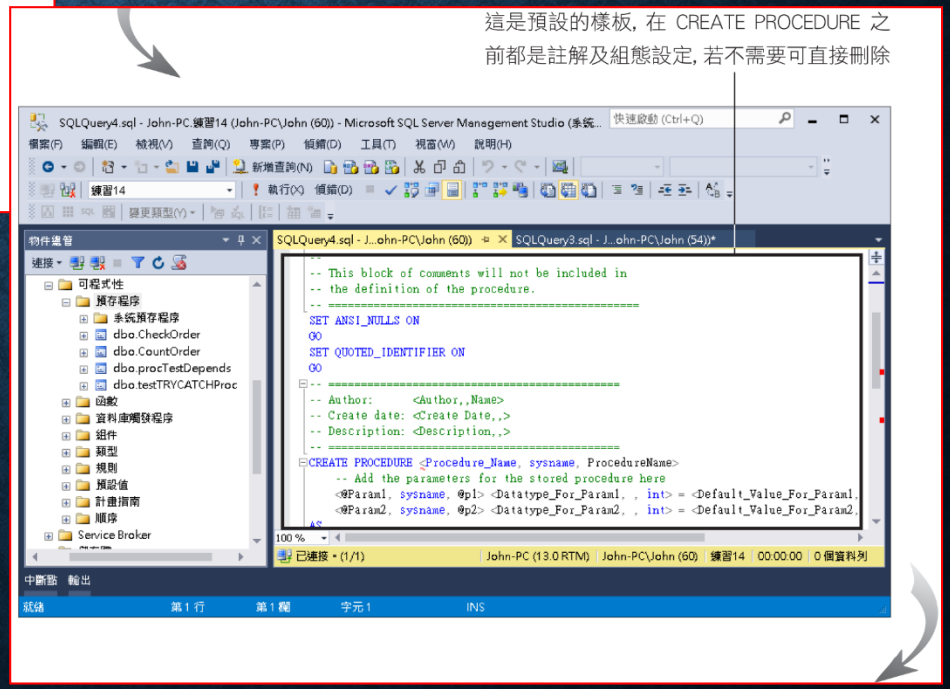
MyProc5;1 的執行結果

	產品名稱	價格
1	Windows 使用手冊	400.00
2	Linux 架站實務	490.00
3	SQL 指令寶典	440.00
4	組合語言	520.00
5	MATHLAB 手	320.00

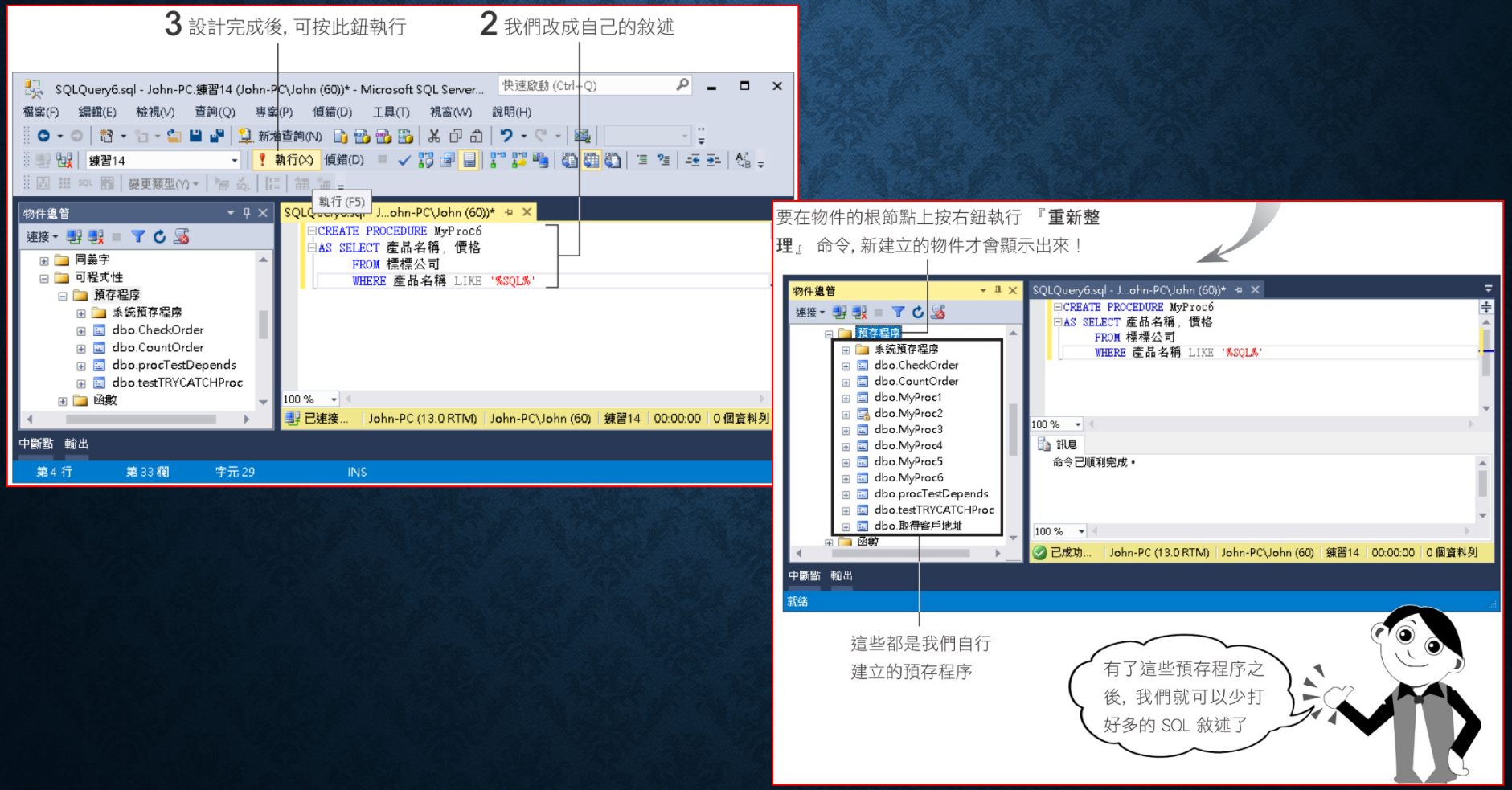
MyProc5;2 的執行結果

使用 SQL SERVER MANAGEMENT STUDIO 建立預存程序

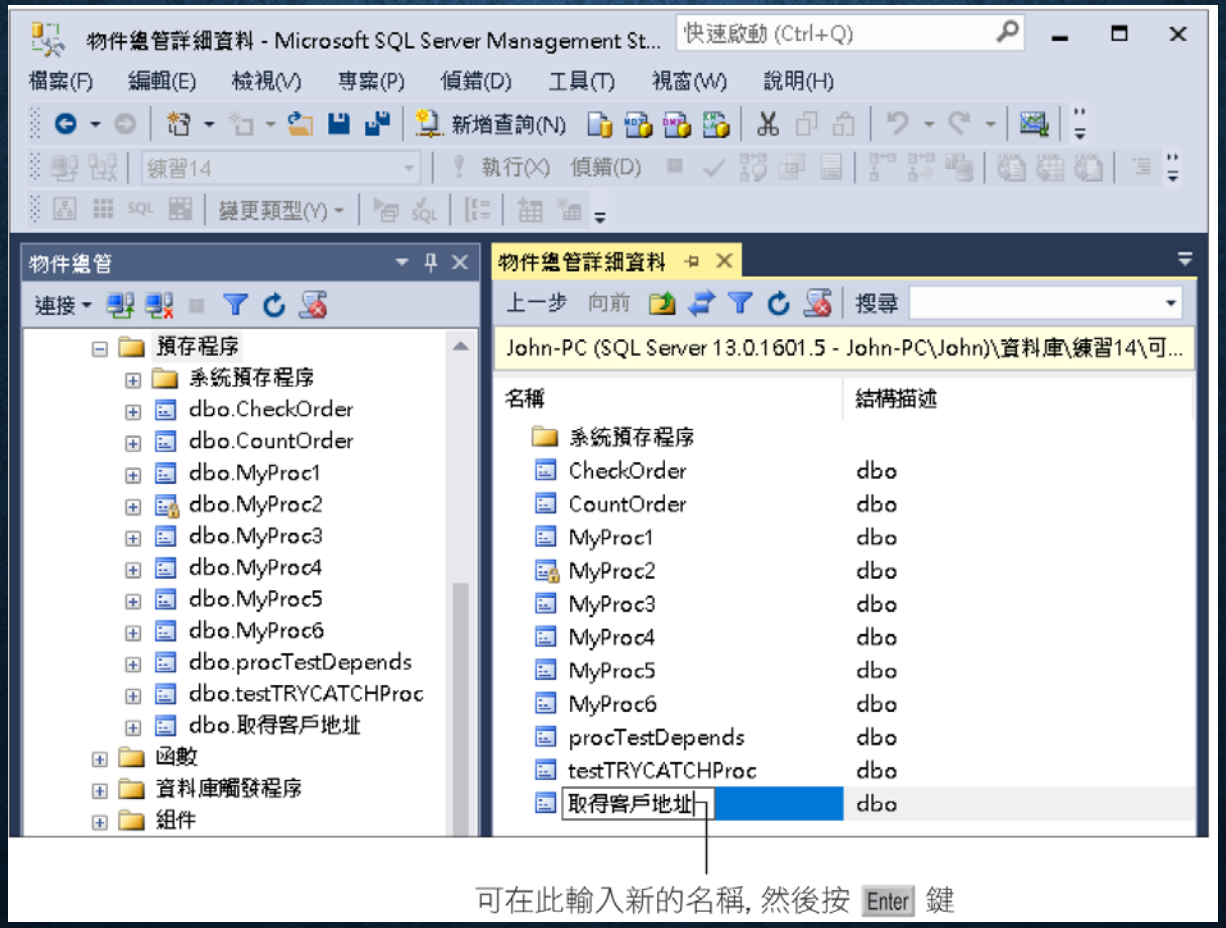




使用 SQL SERVER MANAGEMENT STUDIO 建立預存程序



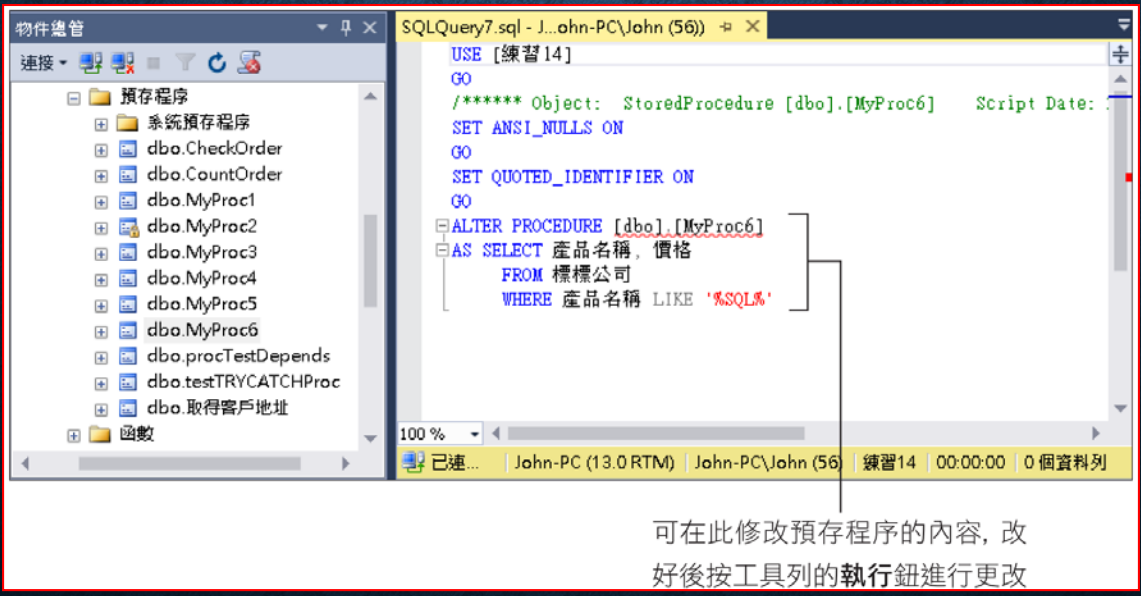
更改預存程序的名稱


物件總管詳細資料 - Microsoft SQL Server Management St...
快速啟動 (Ctrl+Q)
檔案(F) 編輯(E) 檢視(V) 專案(P) 偵錯(D) 工具(T) 視窗(W) 說明(H)
新增查詢(N)
執行(X) 偵錯(D)
變更類型(Y)
物件總管
連接
預存程序
系統預存程序
dbo.CheckOrder
dbo.CountOrder
dbo.MyProc1
dbo.MyProc2
dbo.MyProc3
dbo.MyProc4
dbo.MyProc5
dbo.MyProc6
dbo.procTestDepends
dbo.testTRYCATCHProc
dbo.取得客戶地址
函數
資料庫觸發程序
組件
物件總管詳細資料
上一步 向前
搜尋
John-PC (SQL Server 13.0.1601.5 - John-PC\John)\資料庫\練習14\可...
名稱
結構描述
系統預存程序
CheckOrder dbo
CountOrder dbo
MyProc1 dbo
MyProc2 dbo
MyProc3 dbo
MyProc4 dbo
MyProc5 dbo
MyProc6 dbo
procTestDepends dbo
testTRYCATCHProc dbo
取得客戶地址 dbo
可在此輸入新的名稱, 然後按 Enter 鍵

修改與刪除自訂預存程序

- 修改預存程序

```
ALTER PROCEDURE MyProc2          /* 可直接將原來的預存程序改掉 */
AS SELECT *
   FROM 標標公司
```

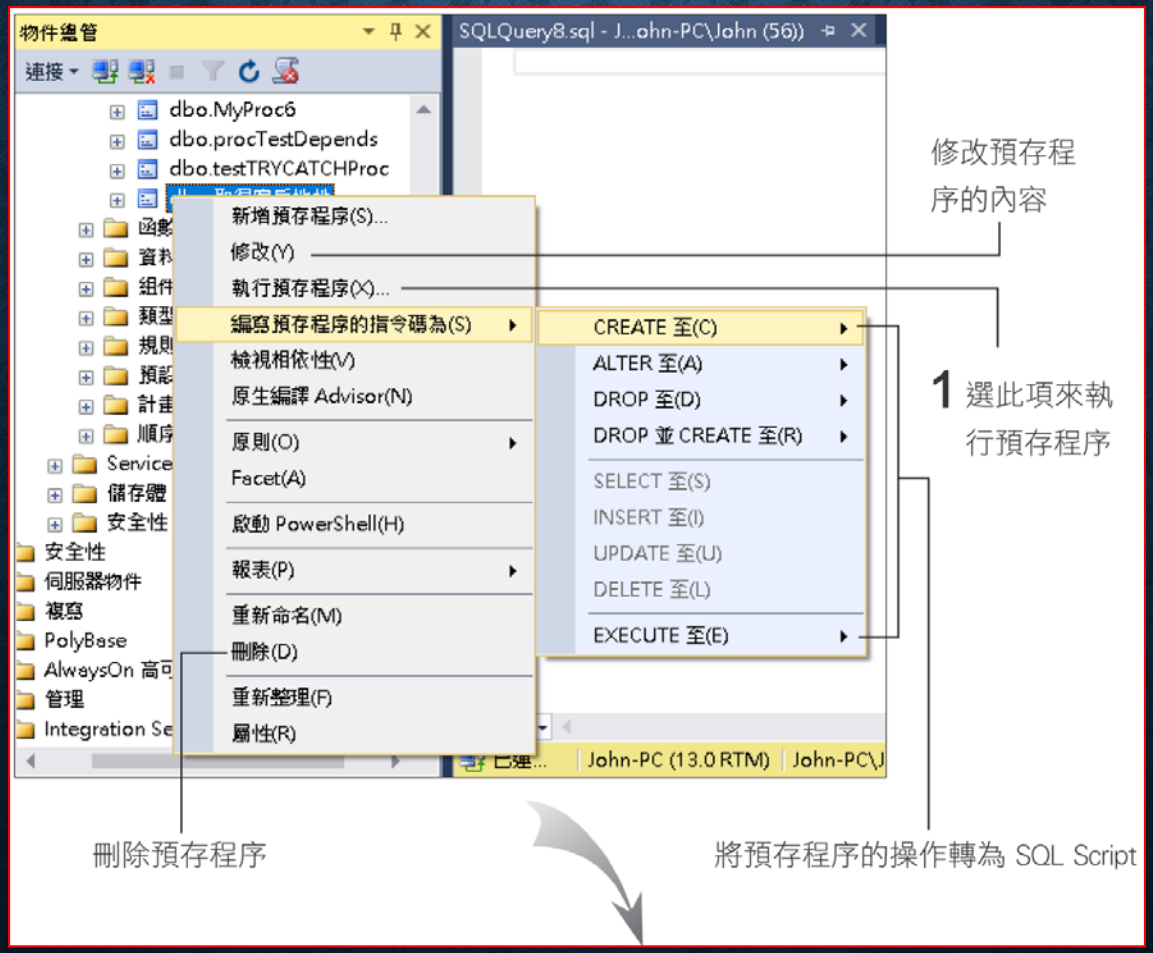


修改與刪除自訂預存程序

- 刪除預存程序

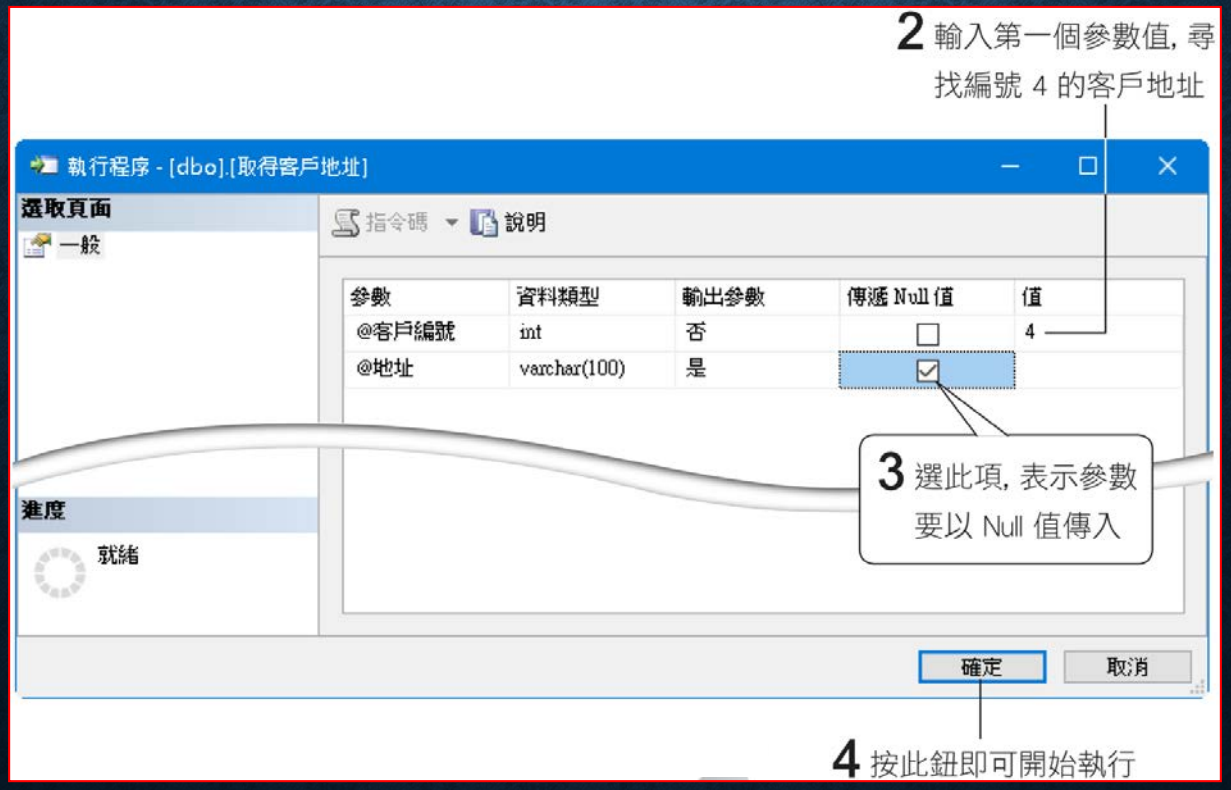
```
DROP PROCEDURE procedure_name [, ...n]
```

使用 SQL SERVER MANAGEMENT STUDIO 執行、管理預存程序

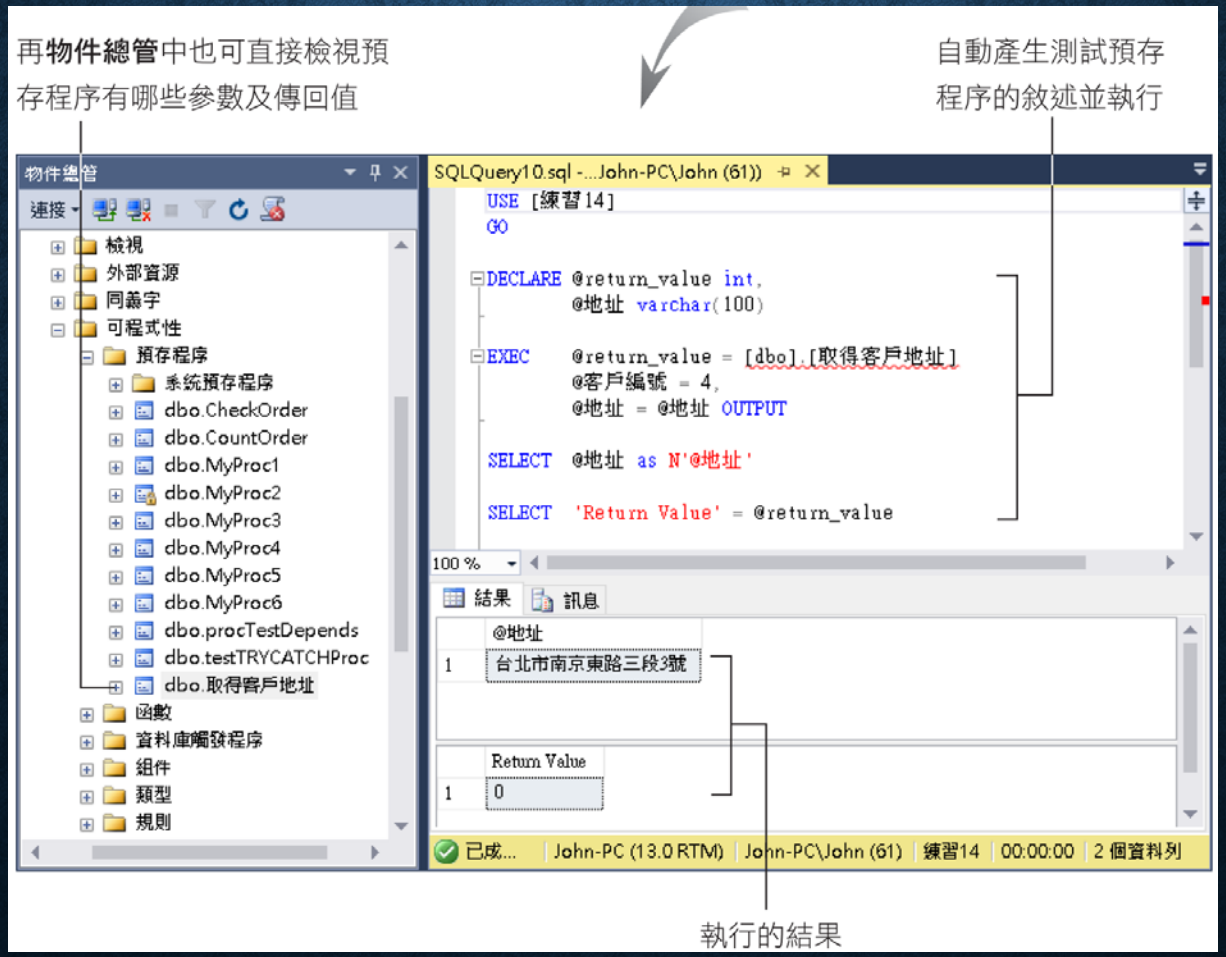


使用 SQL SERVER MANAGEMENT STUDIO 執行、管理預存程序

- 執行『取得客戶地址』的預存程序



使用 SQL SERVER MANAGEMENT STUDIO 執行、管理預存程序



14-3 設計預存程序的技巧

- 在預存程序中使用敘述的限制
- 參數傳遞的技巧
- 預存程序的 3 種傳回值
- 自訂預存程序傳回資料集的格式
- **SET NOCOUNT** 選項

設計預存程序的技巧

- **SET QUOTED_IDENTIFIER** 及 **SET ANSI_NULLS** 選項
- 暫存性的預存程序
- 巢狀呼叫
- 檢視預存程序的使用與被使用關係
- 執行遠端 **SQL Server** 中的預存程序

在預存程序中使用敘述的限制

- 在預存程序中，有些敘述不可使用，包括

CREATE AGGREGATE	CREATE RULE
CREATE DEFAULT	CREATE SCHEMA
CREATE/ALTER FUNCTION	CREATE/ALTER FUNCTION
CREATE/ALTER PROCEDURE	CREATE/ALTER VIEW
SET PARSEONLY	SET SHOWPLAN_ALL
SET SHOWPLAN_TEXT	SET SHOWPLAN_XML
USE 資料庫名稱	

在預存程序中使用敘述的限制

- 在同一個資料庫中，只要使用不同的結構描述，便可以建立相同名稱的物件，例如 **dbo.test**、**abc.test**、**sales.test** 三個資料表可以同時存在。

預存程序	所屬的結構描述
procAll	dbo
procAll	abc
procAll	sales
procsales	sales

資料表	所屬的結構描述
tabAll	dbo
tabAll	abc
tabAll	sales
tabsales	sales

每個 **procAll** 中都會存取到 **tabAll** 及 **tabsales** 資料表，使用者使用時都未指明資料表的結構描述。

在預存程序中使用敘述的限制

使用者	使用者預設的結構描述	執行 procAll	在 procAll 中存取 tabAll	在 procAll 中存取 tabsales	執行 procsales
dbo	dbo	dbo.procAll	dbo.tabAll	名稱錯誤	名稱錯誤
ken	sales	sales.procAll	sales.tabAll	sales.tabsales	sales.procsales
joe	abc	abc.procAll	abc.tabAll	名稱錯誤	名稱錯誤
john	dbo	dbo.procAll	dbo.tabAll	名稱錯誤	名稱錯誤

在預存程序中使用敘述的限制

- 有些指令在執行時若未指定結構描述，會固定以目前使用者的預設結構描述來尋找或建立物件，這些指令包括

CREATE TABLE	DROP TABLE
ALTER TABLE	TRUNCATE TABLE
CREATE INDEX	DROP INDEX
UPDATE STATISTICS	DBCC

```
CREATE PROCEDURE abc.test
AS
ALTER TABLE book                ← book 未指明結構描述
ADD writer varchar (30)
```

Ken 的預設結構描述為 sales，執行此預存程序，即會去找 sales.book，會產生錯誤

參數傳遞的技巧

```
CREATE PROCEDURE test
@a int ,
@b int = NULL,
@c int = 3
AS
SELECT @a, @b, @c
GO
```

```
EXEC test                  /* 錯誤，第一個參數不可省 */
GO
EXEC test 1                /* OK，第 2、3 參數用預設值 */
GO
EXEC test 1, DEFAULT       /* OK，可用 DEFAULT 表示使用預設值 */
GO
EXEC test 1, DEFAULT, 5 /* OK */
GO
EXEC test 1, 2, 5          /* OK */
GO
```

參數傳遞的技巧

```
EXEC test @c = 5, @b = DEFAULT, @a = 1     /* 不用照順序傳了 */
```

	(沒有資料行名稱)	(沒有資料行名稱)	(沒有資料行名稱)
1	1	NULL	5

```
EXEC test 1, @c = 2          ◄── OK！ 1 傳入 @a, 而 @b 使用預設值
GO
EXEC test @c = 2, 1          ◄── 錯誤！因為使用過 @name = value 後
GO                                就必須一直使用此方式來傳參數
EXEC test @c = 5             ◄── 錯誤！因為 @a 參數不可省略
```

	(沒有資料行名稱)	(沒有資料行名稱)	(沒有資料行名稱)
1	1	NULL	2

```
( 1 個資料列受到影響)
訊息 119, 層級 15, 狀態 1, 行 1
必須以 '@name = value ' 傳遞參數編號 2 及後續參數。使用 '@name = value' 格式之後, 所有後續的參數都必須以 '@n ame = value' 的格式來傳遞。
訊息 201 , 層級 16, 狀態 4, 程序 test, 行 0
程序或函數 'test' 必須有參數 '@a', 但是並未提供。
```

預存程序的 3 種傳回值

- 在程序中以 **"RETURN n"** 傳回整數值。
- 在參數中指定 **OUTPUT** 選項的參數。
- 預存程序中執行敘述 **(**例如 **SELECT)** 所傳回的資料集 **(RecordSet)** 及通知訊息。

預存程序的 3 種傳回值

```
CREATE PROCEDURE TestRetVal
@TableName varchar(30) OUTPUT
AS
DECLARE @sqlstr varchar(100)
SET @sqlstr = 'SELECT * FROM ' + @TableName
EXEC (@sqlstr)                          /* 執行字串中的 SQL 敘述 */

IF @@ERROR = 0
    BEGIN
       SET @TableName = 'Hello'
       RETURN 0
    END
ELSE
    RETURN 1
```

預存程序的 3 種傳回值

```
GO

DECLARE @ret int, @name varchar(30)
SET @name = '旗旗公司'
EXEC @ret = TestRetVal @name OUTPUT
PRINT @name + ', RETURN = ' + CAST(@ret AS CHAR)
```

	產品名稱	價格
1	Windows 使用手冊	400.00
2	Linux 架站實務	500.00
3	JAVA 程式語言	420.00
4	PHP程式語言	500.00

預存程序在執行敘述時所傳回到應用程式中的資料集

```
(4 個資料列受到影響)
Hello, RETURN = 0
```

← 執行敘述時所傳回的通知訊息

← 參數傳回值及 Return 傳回值

自訂預存程序傳回資料集的格式

```
CREATE PROCEDURE testWithResultSet
AS SELECT * FROM 旗旗公司
GO

EXEC testWithResultSet

EXEC testWithResultSet
WITH RESULT SETS
( (產品 nvarchar(20),
   價格 int)
)
```

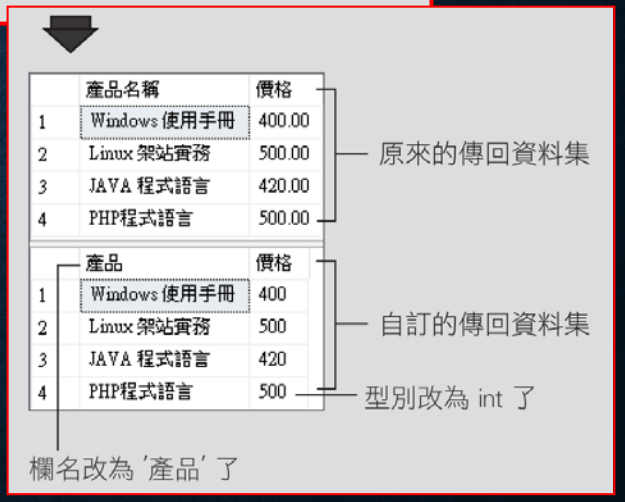
← 建立預存程序

← 執行預存程序

← 執行預存程序並自訂傳回資料集格式

← 將欄位名稱改為 '產品'

← 將價格欄的型別改為 int



自訂預存程序傳回資料集的格式

```
EXEC . . .
WITH RESULT SETS
( ( { column_name data_type [ NULL | NOT NULL ] }
    [, ...n] )
)
```

自訂預存程序傳回資料集的格式

- **column_name**
- **data_type**
- **[NULL | NOT NULL]**
- **EXEC …WITH RESULT SETS UNDEFINED**
- **EXEC …WITH RESULT SETS NONE**
- **EXEC …WITH RESULT SETS ((資料集定義),(資料集定義),…)**

自訂預存程序傳回資料集的格式

```
CREATE PROCEDURE testWithResultSet2
AS SELECT * FROM 旗旗公司
    SELECT * FROM 標標公司
GO
```

← 建立預存程序

```
EXEC testWithResultSet2
WITH RESULT SETS
( (旗旗產品 nvarchar(20), 價格 int),
  (標標產品 nvarchar(20), 價格 int)
)
```

← 指定第 1 個資料集的格式

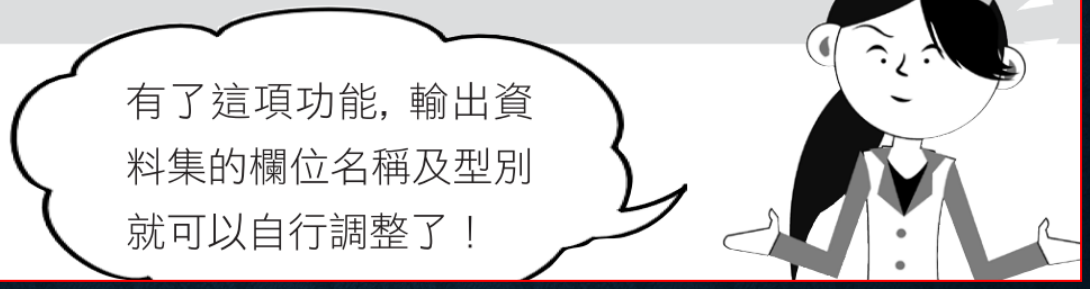
← 指定第 2 個資料集的格式

自訂預存程序傳回資料集的格式

	旗旗產品	價格
1	Windows 使用手冊	400
2	Linux 架站實務	500
3	JAVA 程式語言	420
4	PHP程式語言	500

	標標產品	價格
1	Windows 使用手冊	400
2	Linux 架站實務	490
3	SQL 指令寶典	440
4	組合語言	520
5	MATHLAB 手	320

二個傳回資料集的欄名已更改為
'旗旗產品' 及 '標標產品' 了



SET NOCOUNT 選項

- 執行各類查詢或修改資料的 **SQL** 敘述，都會傳回該敘述影響了多少筆紀錄的通知訊息。為避免干擾應用程式的運作，可以使用下列語法設定

```
SET NOCOUNT {ON | OFF}
```

SET QUOTED_IDENTIFIER 及 SET ANSI_NULLS 選項

執行 SET_QUOTED_IDENTIFIER ON 後，雙引號就可用來標示物件的別名

執行 SET ANSI_NULLS ON 後，任何資料與 NULL 做 = 或 <> 的比較時都會是 False。若設為 OFF，則可以直接使用 = 或 <> 做比。因此預存程序中要注意設定，離開預存程序時，要設回原來的狀態。

```
SELECT *
FROM 客戶
WHERE 聯絡人 = NULL          ← 找出未填寫聯絡人的記錄
```

暫存性的預存程序

- 暫存預存程序會存放在 **tempdb** 資料庫中。
- 當暫存預存程序的使用者都離線之後，暫存預存程序會自動被刪除。
- 區域暫存預存程序的名稱以 **#** 開頭，只有建立它的人可以使用。
- 全域暫存預存程序的名稱以 **##** 開頭，所有的使用者都可使用它。

```
CREATE PROCEDURE #tempproc
AS PRINT 'Test'
GO

EXEC #tempproc
```

```
Test
```

巢狀呼叫

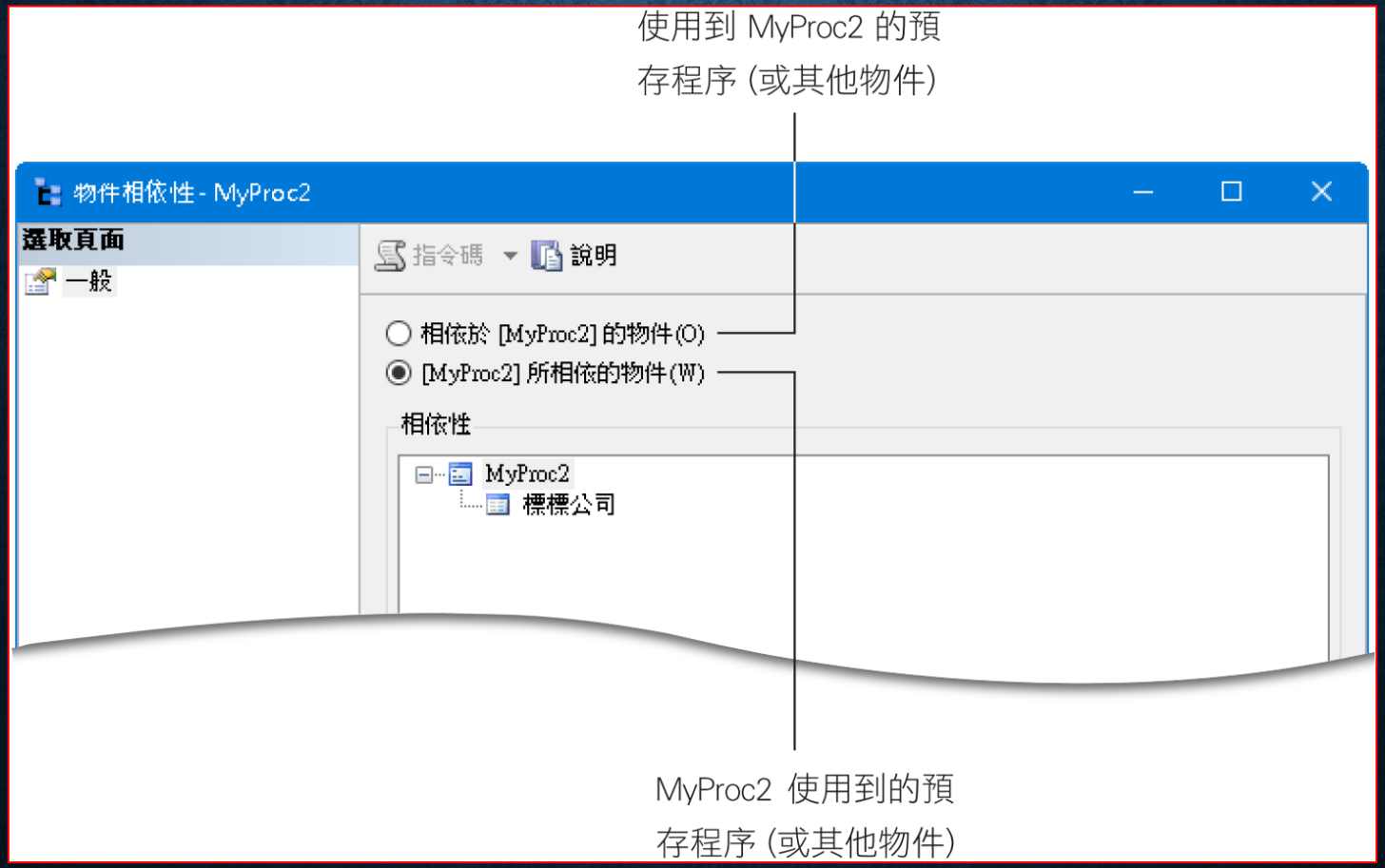
@@NESTLEVEL 是全域變數，查看目前程序所在的層數

```
CREATE PROCEDURE proc3
AS PRINT 'Proc3: at level ' + CAST(@@NESTLEVEL AS CHAR)
GO

CREATE PROCEDURE proc2
AS PRINT 'Proc2 start: at level ' + CAST(@@NESTLEVEL AS CHAR)
  EXEC proc3
  PRINT 'Proc2 end: at level ' + CAST(@@NESTLEVEL AS CHAR)
GO
```

```
CREATE PROCEDURE proc1
AS PRINT 'Proc1 start: at level ' + CAST(@@NESTLEVEL AS CHAR)
  EXEC proc2
  PRINT 'Proc1 end: at level ' + CAST(@@NESTLEVEL AS CHAR)
GO

EXEC proc1
```

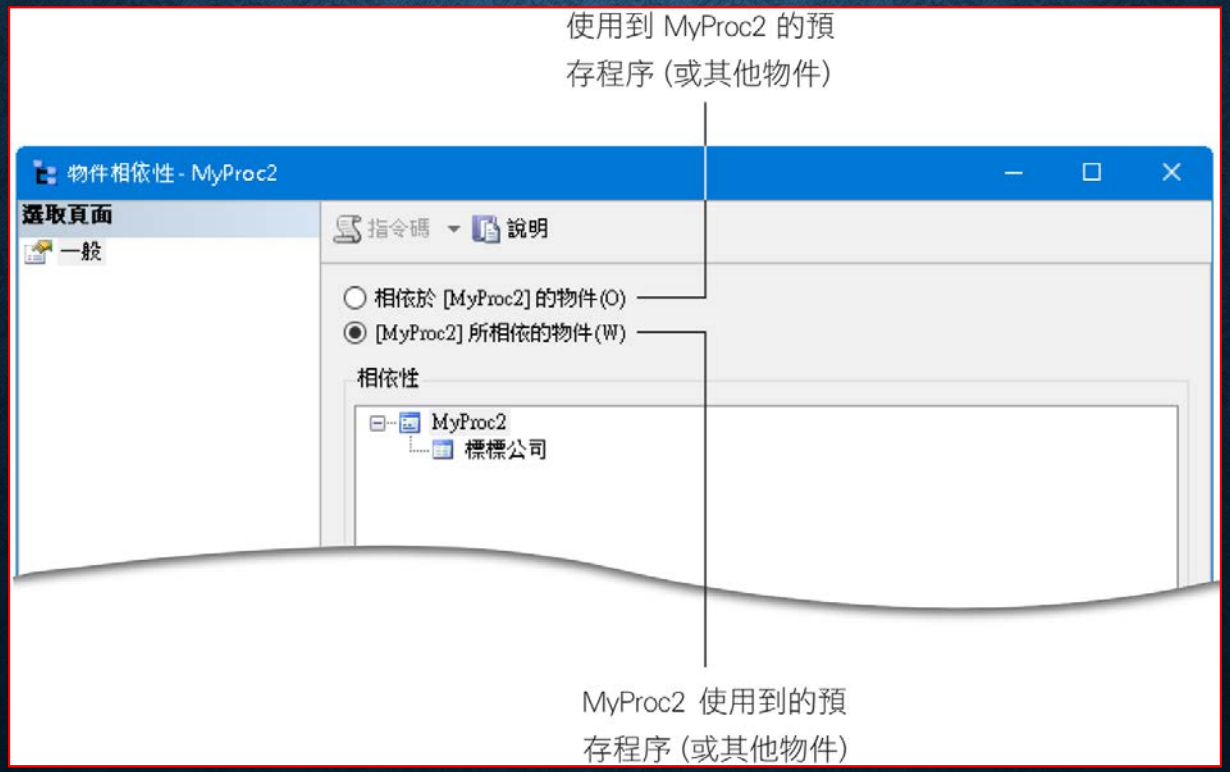
```
Proc1 start:at level 1
Proc2 start:at level 2
Proc3: at level 3
Proc2 end: at level 2
Proc1 end: at level 1
```

檢視預存程序的使用與被使用關係

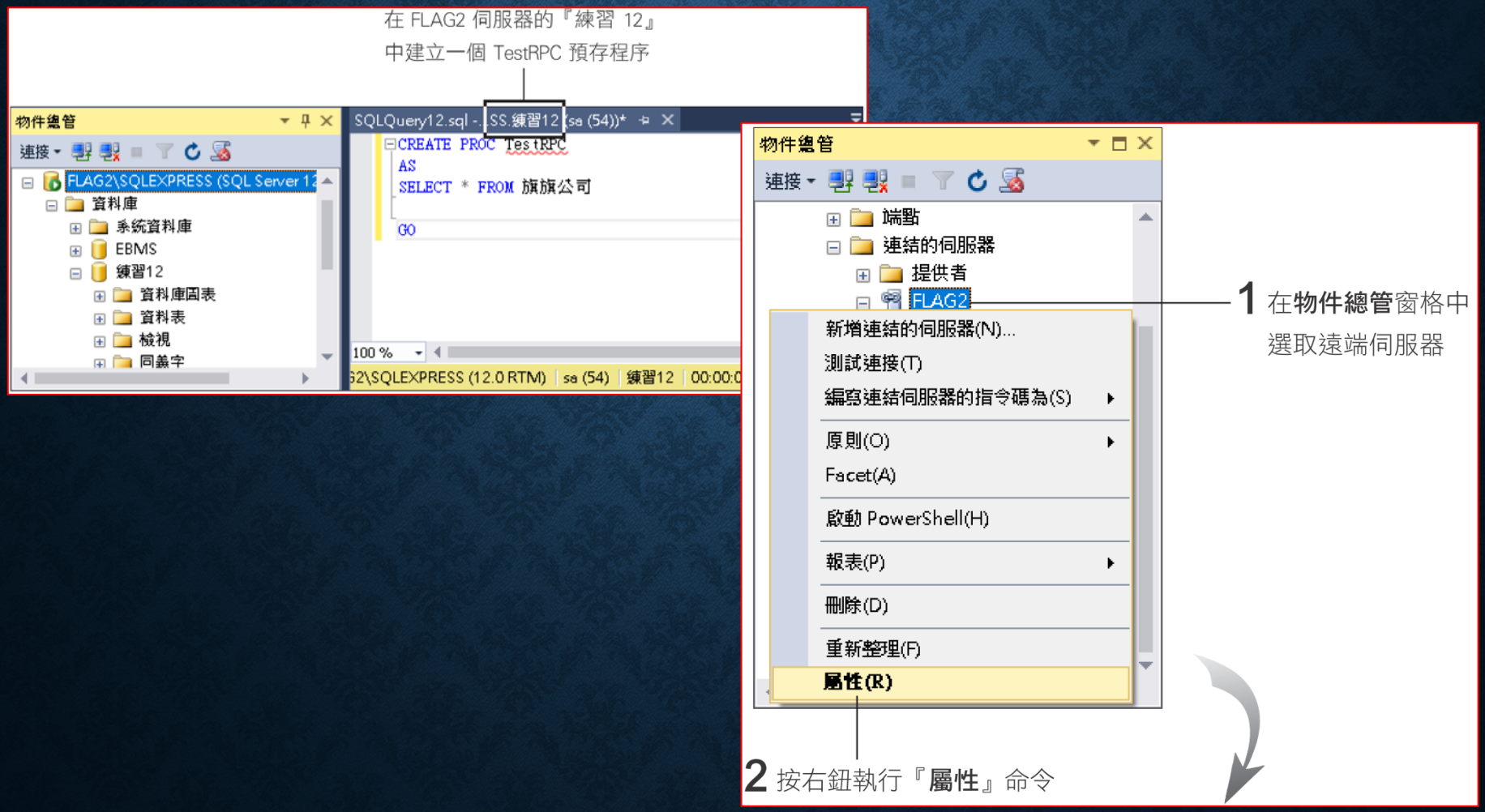


檢視預存程序的使用與被使用關係

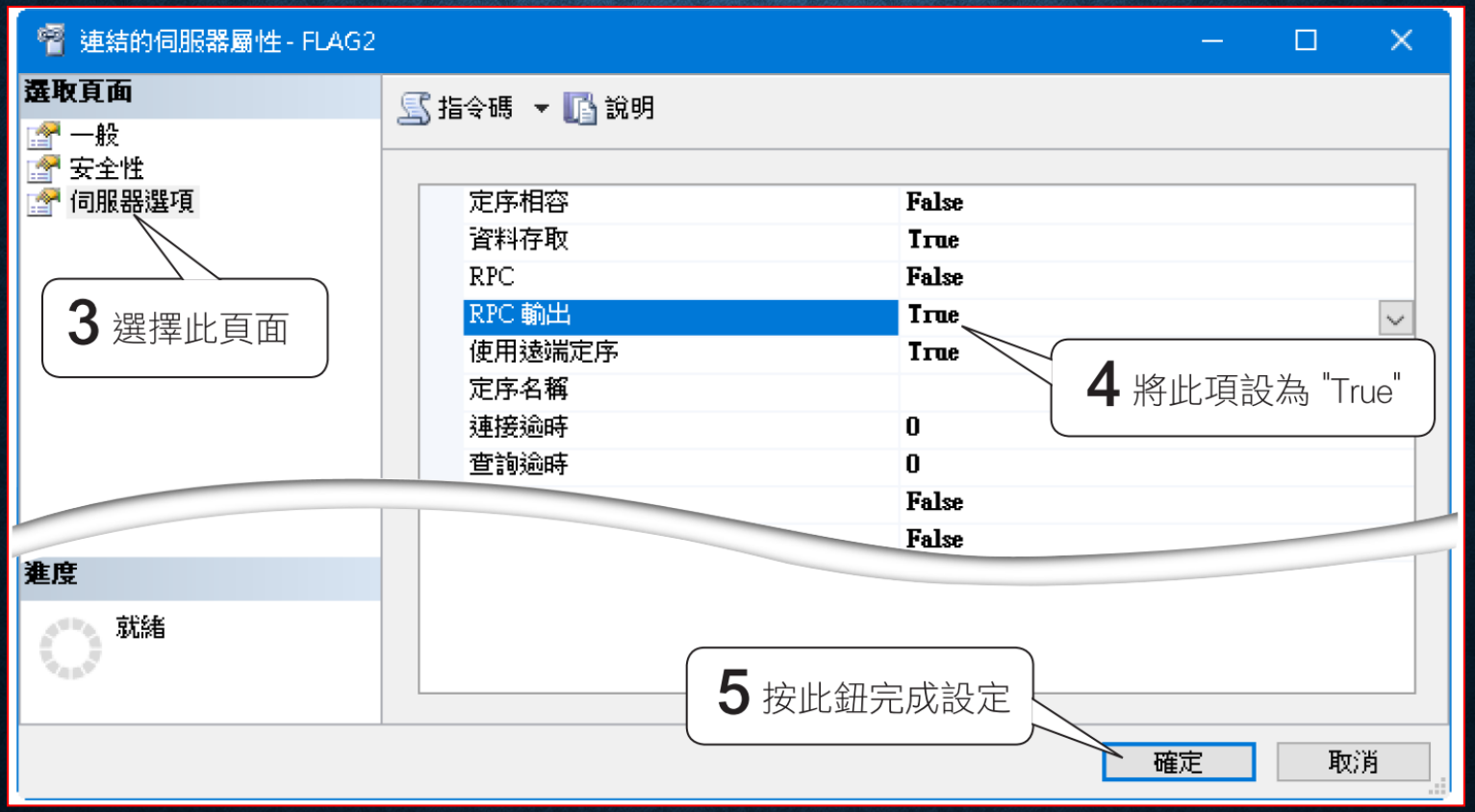
- 物件總管中的預存程序按右鍵，執行『檢視相依姓』



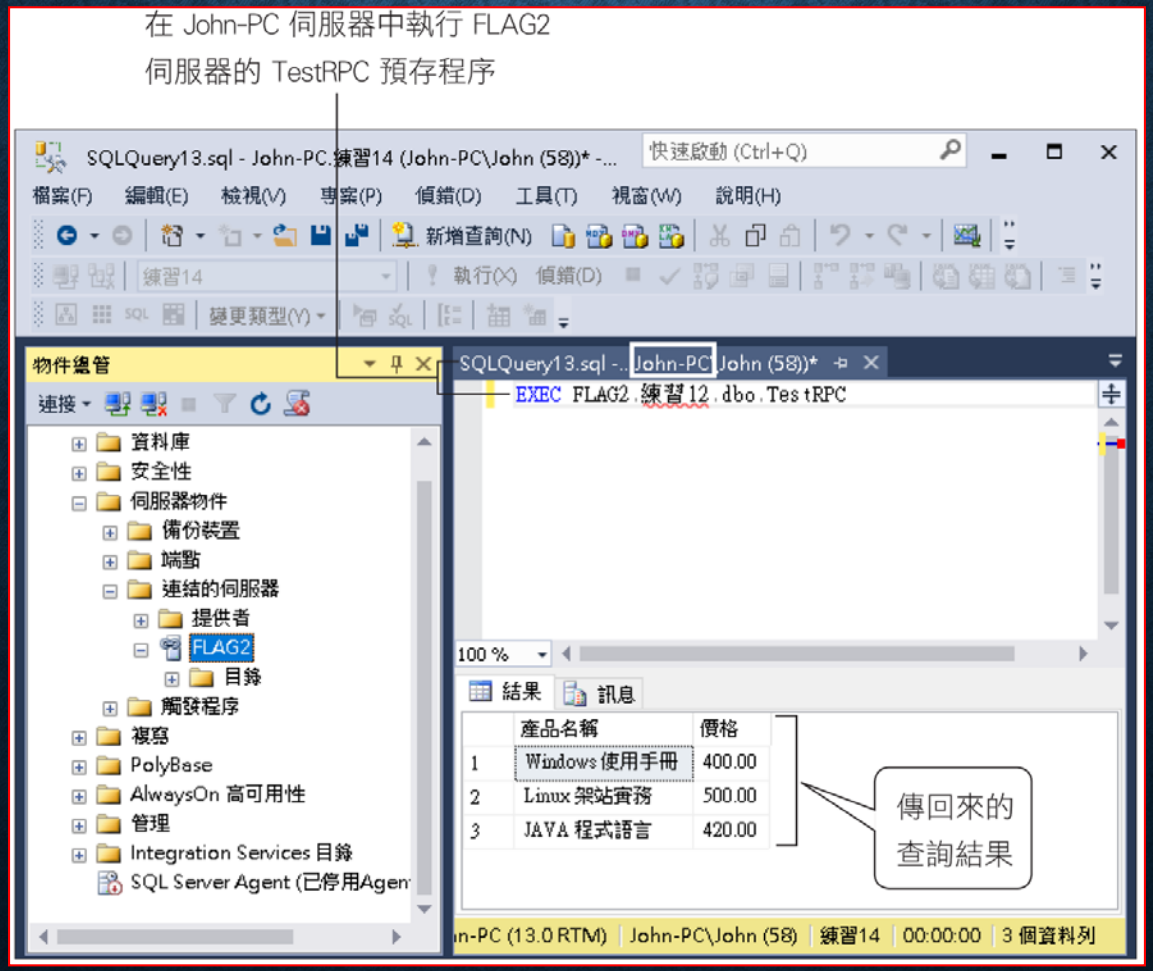
執行遠端 SQL SERVER 中的預存程序



執行遠端 SQL SERVER 中的預存程序



執行遠端 SQL SERVER 中的預存程序



14-4 使用 TABLE 型別的參數

```
CREATE TYPE IntTableType AS TABLE          ◄── 建立『使用者定義資料表類型』,
(名稱 VARCHAR(20), 數值 INT )                  內含名稱、數值 2 個欄位
GO

CREATE PROC 找出最大者
@title varchar(30), @tab IntTableType READONLY
AS
   DECLARE @maxv INT

   SELECT @maxv = MAX(數值) FROM @tab    -- 找出最大值

   SELECT @title 說明, 名稱 最大者, @maxv 數量
   FROM @tab
   WHERE 數值 = @maxv
GO
```

使用 TABLE 型別的參數

```
DECLARE @tab IntTableType          ←— 使用『使用者定義資料表類型』宣告變數

INSERT @tab
SELECT 客戶名稱, sum(數量)          ←— 依客戶名稱分類加總
FROM 出貨記錄
GROUP BY 客戶名稱

EXEC 找出最大者 '出貨量最大的客戶' , @tab

DELETE @tab      -- 清除變數內容

INSERT @tab
SELECT 股票名稱, sum(購買張數)      ←— 依股票名稱分類加總
FROM 股票交易記錄
GROUP BY 股票名稱

EXEC 找出最大者 '庫存最多的股票' , @tab
```

	說明	最大者	數量
1	出貨量最大的客戶	大雄書局	25

	說明	最大者	數量
1	庫存最多的股票	統一	23