# 第十六章 預存程序的使用

***SQL Server 2000***

***資料庫實務應用***

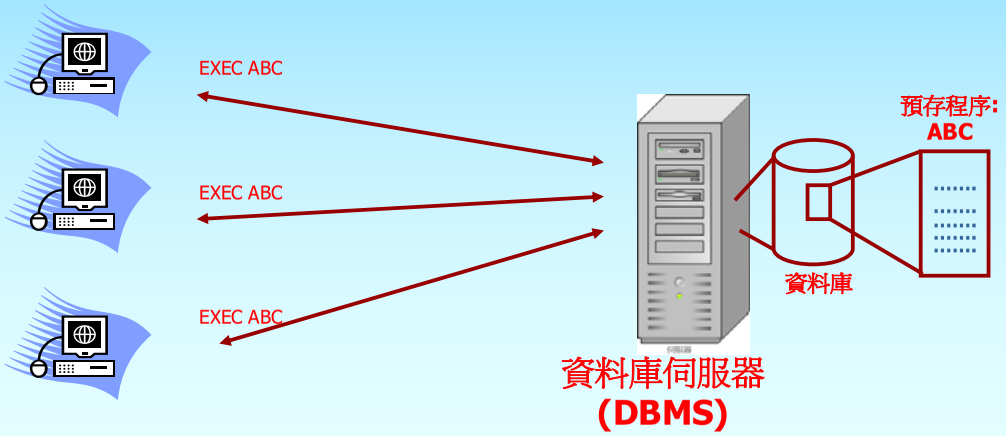
*許正憲，張嘉琪 著*

# 學習重點

- 何謂預存程序(Store Procedure)
- 建立與維護預存程序
- 帶有參數輸出/輸入的預存程序
- 執行預存程序命令
- 巢狀式預存程序
- 重新編譯預存程序
- 預存程序的安全性管理
- 預存程序的限制

# 何謂預存程序(Store Procedure)

- 建立在資料庫上的程序，使用者透過呼叫這些程序來對資料庫作資料的存取作業。



# 預存程序的優點：

1. 提昇執行效率：
2. 降低網路負載
3. 改善資料安全性
4. 模組化的設計

# 預存程序的種類

- 系統預存程序(System Stored procedure)
  - 儲存在master系統資料庫中，透過系統預存程序可執行SQL Server內部的許多管理或系統相關的資訊。(名稱皆以sp_開頭，如：sp_dboption、sp_help…)
- 延伸預存程序(Extended Store Procedure)
  - 讓使用者可用其它的程式語言(如：C/C++)來撰寫一些用T-SQL程式敘述無法達成的作業。必須存放在master系統資料庫中的【延伸預存程序】項目中，它的名稱通常是以xp_開頭來命名(如：xp_sendmail)
  - 要執行xp_開頭的延伸預存程序時，必須以三部份名稱(如：master.. xp_dsninfo)來呼叫。
- 使用者自訂預存程序(User-defined store procedure)
  - 使用者可以自己設計預存程序，如同程式的程序一般，並賦予一個名稱

範例16-1 使用系統預存程序sp_helptext來顯示某個規則、預設值或未加密預存程序、使用者定義函數、觸發程序、或檢視的文字。

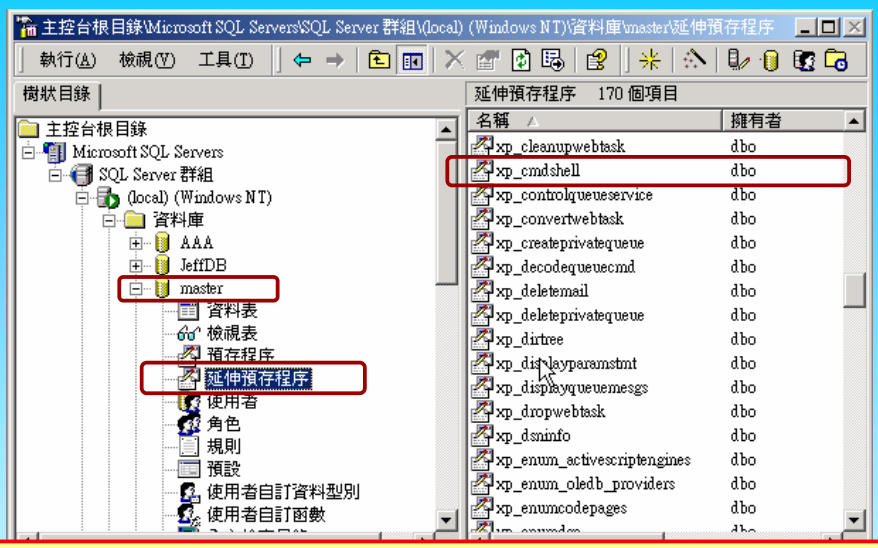
```
CREATE VIEW vw_台北市
AS
SELECT 員工編號,姓名,部門,地址
FROM 學貫工作坊
WHERE SUBSTRING(地址,1,3)='台北市'
GO
SELECT * FROM vw_台北市

EXEC sp_helptext 'vw_台北市'
```

Sp_helptext 可查閱 'vw_台北市' 檢視表的內容

結果：

| | 員工編號 | 姓名 | 部門 | 地址 |
|---|---|---|---|---|
| 1 | 1 | 王凱翔 | 資訊部 | 台北市內湖路一段50號 |
| 2 | 2 | 陳美惠 | 會計部 | 台北市忠孝東路五段56號2樓 |
| 3 | 5 | 李美君 | 財務部 | 台北市內湖區康寧路一段45號3樓 |
| 4 | 7 | 黃賢昌 | 行銷部 | 台北市民生東路二段38巷2號5樓 |
| 5 | 10 | 李思涵 | 財務部 | 台北市北投區中央北路二段285巷10號6樓 |
| 6 | 11 | 陳建業 | 財務部 | 台北市忠孝西路一段48號10樓 |
| 7 | 12 | 李家銘 | 製造部 | 台北市長安西路一段125-1號 |
| 8 | 13 | 徐志明 | 資訊部 | 台北市郵政146號信箱 |
| 9 | 15 | 江淑芬 | 會計部 | 台北市北投區大度路59-1號 |
| 10 | 16 | 曾玉姍 | 製造部 | 台北市松山區八德路四段125巷40-1號 |
| 11 | 24 | 張志強 | 資訊部 | 台北市松山區南京東路五段250巷36弄36號... |
| 12 | 25 | 吳惠玉 | 會計部 | 台北市仁愛路一段128號3樓 |
| 13 | 27 | 梁美珍 | 財務部 | 台北市大安區溫州街68巷6號6樓 |
| 14 | 29 | 簡志瑋 | 資訊部 | 台北市中山北路六段136號3樓 |

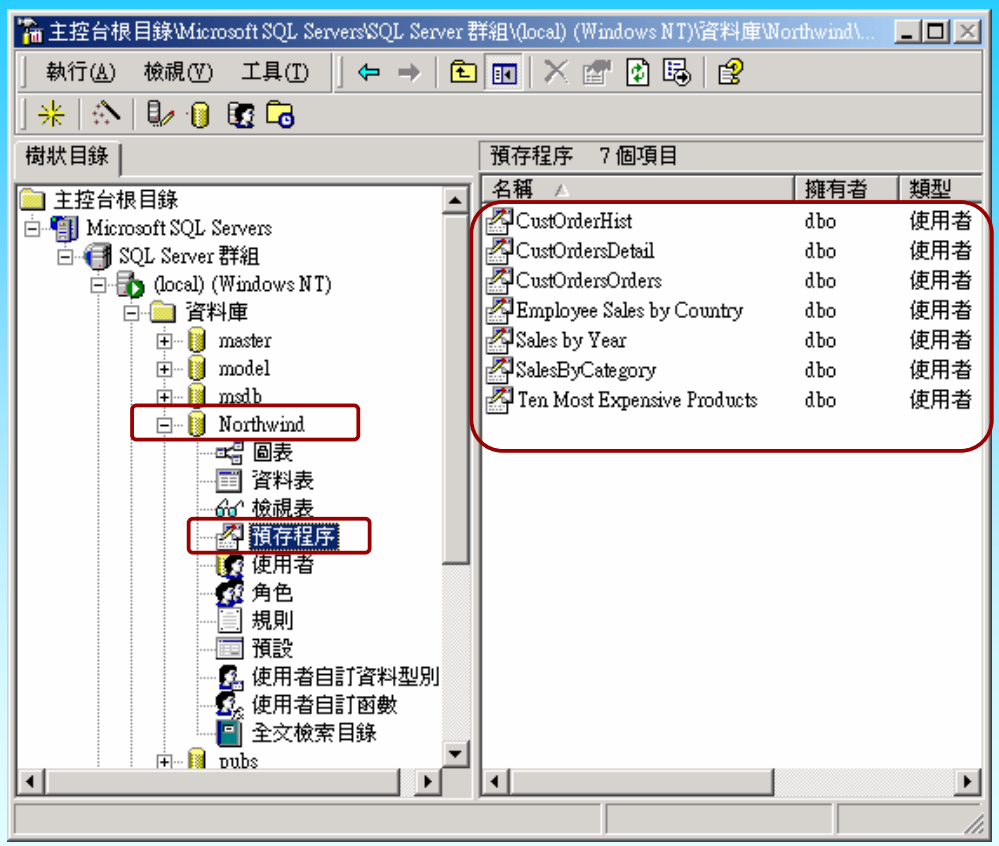
| | Text |
|---|---|
| 1 | CREATE VIEW vw_台北市 |
| 2 | AS |
| 3 | SELECT 員工編號,姓名,部門,地址 FROM 學貫工作坊 WHERE SUBSTRING(地址,1,3)='台北市' |




範例：**16-2** 執行作業系統**DIR**命令。

```
USE master                  -- 先切換到MASTER系統資料庫
GO
EXEC xp_cmdshell 'dir c:\'
--若未指名master資料庫，則指令更正爲EXEC master..xp_cmdshell 'dir c:\'
```

| | output | | |
|---|---|---|---|
| 6 | 2002/10/02 07:21p | <DIR> | CA_LIC |
| 7 | 2004/03/10 11:09p | <DIR> | database |
| 8 | 2003/10/05 09:32p | <DIR> | database 6 |
| 9 | 2002/05/19 09:28p | <DIR> | Documents ... |
| 10 | 2003/10/08 04:15a | <DIR> | downloads |
| 11 | 2002/09/25 10:08p | <DIR> | hoffer6e |


主控台根目錄\Microsoft SQL Servers\SQL Server 群組\(local) (Windows NT)\資料庫\Northwind\...
執行(A)
檢視(V)
工具(T)
樹狀目錄
主控台根目錄
Microsoft SQL Servers
SQL Server 群組
(local) (Windows NT)
資料庫
master
model
msdb
Northwind
圖表
資料表
檢視表
預存程序
使用者
角色
規則
預設
使用者自訂資料型別
使用者自訂函數
全文檢索目錄
pubs
預存程序 7 個項目
名稱
擁有者
類型
CustOrderHist dbo 使用者
CustOrdersDetail dbo 使用者
CustOrdersOrders dbo 使用者
Employee Sales by Country dbo 使用者
Sales by Year dbo 使用者
SalesByCategory dbo 使用者
Ten Most Expensive Products dbo 使用者

# 建立與維護預存程序

- 建立預存程序
  - 透過Enterprise Manager建立
  - 透過 Create Procedure 指令建立
- 修改預存程序
  - 透過Enterprise Manager修改
  - 透過 Alter Procedure 指令修改
- 刪除預存程序
  - 透過Enterprise Manager 刪除
  - 透過 Drop Procedure 指令刪除

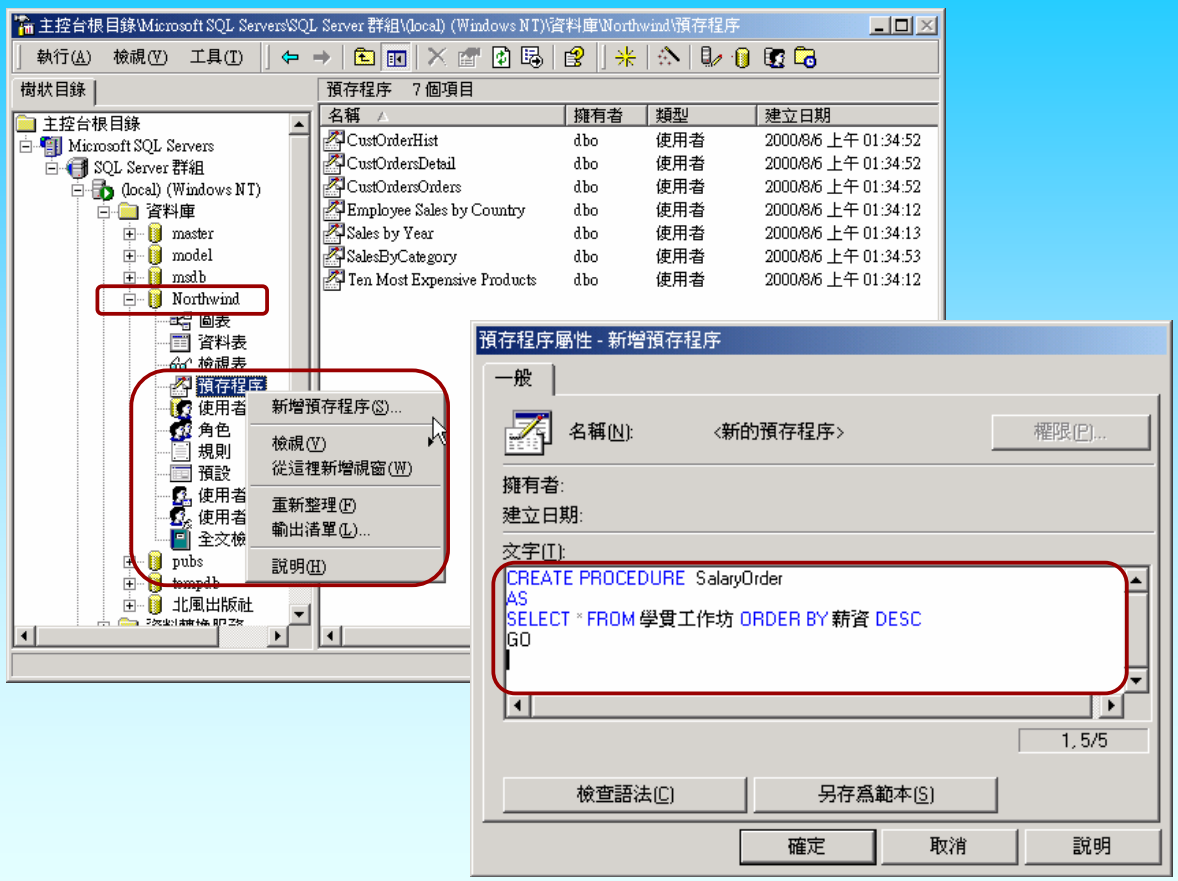
# 建立預存程序

- 建立預存程序語法：

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

範例16-4 建立一SalaryOrder的預存程序，內容為「學貫工作坊」員工薪資的遞減排列。

```
USE 北風出版社
GO
CREATE PROCEDURE  SalaryOrder
AS
SELECT * FROM 學貫工作坊 ORDER BY 薪資 DESC
GO
```

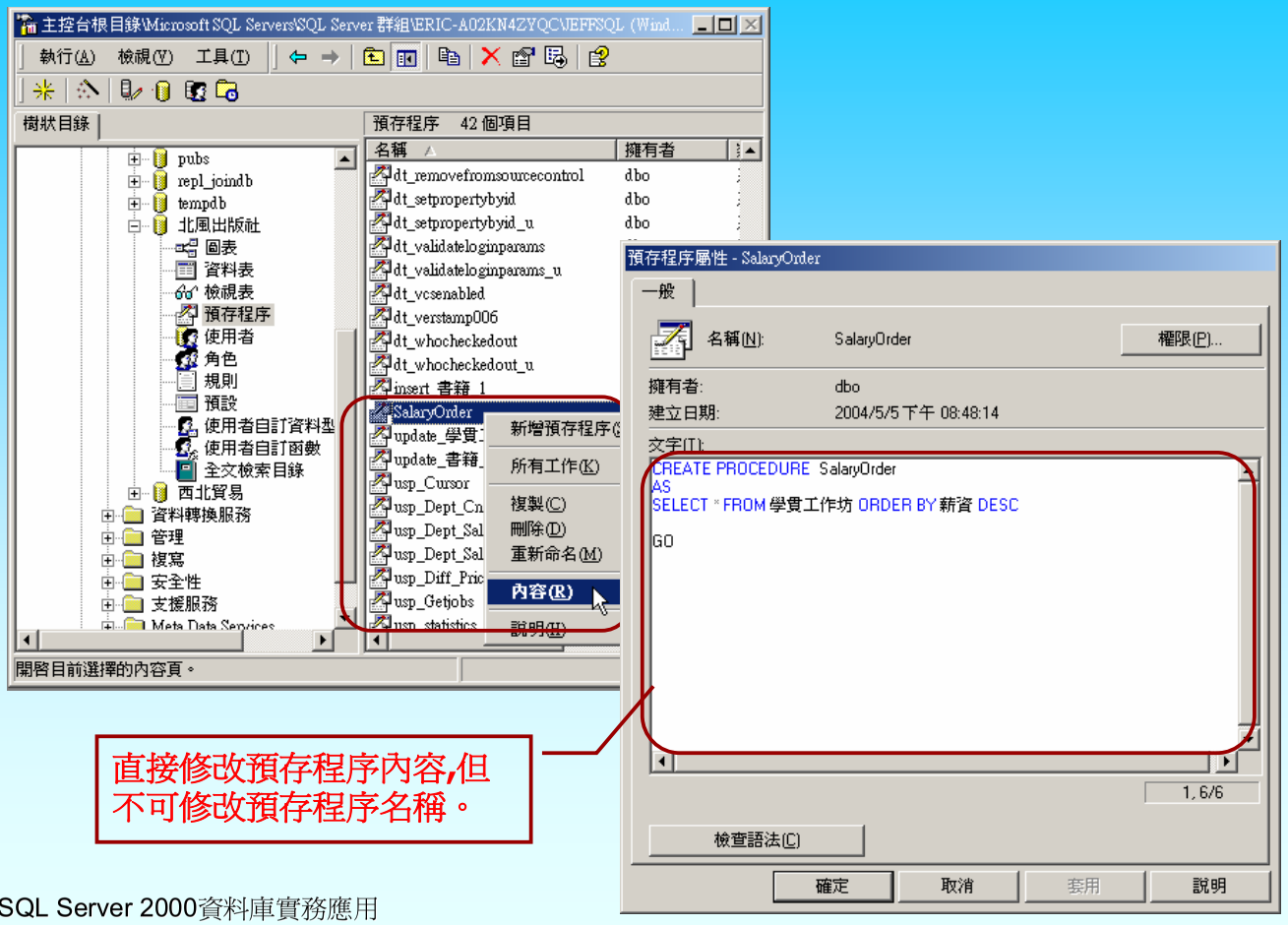

主控台根目錄\Microsoft SQL Servers\SQL Server 群組\(local) (Windows NT)\資料庫\Northwind\預存程序
執行(A) 檢視(V) 工具(T)
樹狀目錄
主控台根目錄
Microsoft SQL Servers
SQL Server 群組
(local) (Windows NT)
資料庫
master
model
msdb
Northwind
圖表
資料表
檢視表
預存程序
使用者
角色
規則
預設
使用者
使用者
全文檢
pubs
tempdb
北風出版社
預存程序 7 個項目
名稱
擁有者
類型
建立日期
CustOrderHist dbo 使用者 2000/8/6 上午 01:34:52
CustOrdersDetail dbo 使用者 2000/8/6 上午 01:34:52
CustOrdersOrders dbo 使用者 2000/8/6 上午 01:34:52
Employee Sales by Country dbo 使用者 2000/8/6 上午 01:34:12
Sales by Year dbo 使用者 2000/8/6 上午 01:34:13
SalesByCategory dbo 使用者 2000/8/6 上午 01:34:53
Ten Most Expensive Products dbo 使用者 2000/8/6 上午 01:34:12
新增預存程序(S)...
檢視(V)
從這裡新增視窗(W)
重新整理(F)
輸出清單(L)...
說明(H)
預存程序屬性 - 新增預存程序
一般
名稱(N): <新的預存程序>
權限(P)...
擁有者:
建立日期:
文字(T):
CREATE PROCEDURE SalaryOrder
AS
SELECT * FROM 學貫工作坊 ORDER BY 薪資 DESC
GO
1, 5/5
檢查語法(C)
另存爲範本(S)
確定
取消
說明

# 修改預存程序

- 可使用ALTER PROCEDURE命令，它的指令語法與建立預存程序的命令相同，僅僅差在 CREATE 改成 ALTER 而已。

範例16-5 將預存成程序SalaryOrder內容改爲只列薪資前十名。

```
ALTER PROCEDURE SalaryOrder
AS
SELECT TOP 10 * FROM 學貫工作坊 ORDER BY 薪資 DESC
GO
```




直接修改預存程序內容,但不可修改預存程序名稱。

# 刪除預存程序

- 刪除預存程序的語法如下：

  **DROP PROCEDURE { *procedure* } [ ,...*n* ]**

  - DROP PROCEDURE 權限預設是預存程序的擁有人才有的，而且權限是不能轉讓。
  - 但是 db_owner 與 db_ddladmin 資料庫角色及 sysadmin 伺服器角色的成員可以利用在 DROP PROCEDURE 中指定擁有人的方式，卸除任何物件。
  - 預存程序的刪除一次允許刪除多個。

範例16-6：刪除目前資料庫中的SalaryOrder預存程序。

```
USE 北風出版社
GO
DROP PROCEDURE SalaryOrder
EXEC sp_help
```

查看SalaryOrder預存程序是否被刪除了。

# 帶有參數輸出/輸入的預存程序

- 帶參數輸入的預存程序的語法片段：

  [ { @parameter  data_type } [ = default ]] [ ,...n ]

  - @parameter：輸入參數名稱，輸入參數須以@開頭命名，可以宣告一個以上個輸入參數，宣告的參數僅在該程序範圍內使用。
  - data_type：參數的資料型別，所有資料型別都可以用。
  - default：程序參數的預設值，預設值必須為常數或NULL，也可配合 LIKE 關鍵字使用。
  - n：一個預存程序最多可以有2100個參數。

範例16-7 在「北風出版社」中，建立一預存程序，依員工編號、加班時數來更改「學貫工作表」中員工的員工編號、加班時數。

```
USE 北風出版社
GO
CREATE PROCEDURE usp_update_overtime
        (@M員工編號      int,
         @M加班時數      int)
AS UPDATE 學貫工作坊
  SET   加班時數        = @M加班時數
  WHERE ( 員工編號  = @M員工編號 )
GO
--執行預存程序，以調整員工編號2的加班時數爲18。
EXEC usp_update_overtime 2,18
SELECT 員工編號 , 姓名  FROM  學貫工作坊  WHERE 員工編號 = 2
```

結果：

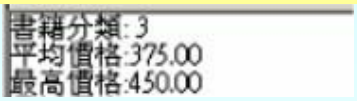
| | 員工編號 | 姓名 | 加班時數 |
|---|---|---|---|
| 1 | 2 | 陳美惠 | 18 |

- 帶參數輸出的預存程序的語法片段：
- [ { @parameter data_type } [ VARYING ] [ OUTPUT ] ] [ ,...n ]
  - 輸出參數的名稱的後面必須加上OUTPUT關鍵字。
  - 輸出參數的資料型別可以是任意型別的資料，包含：text、ntext、image、cursor與使用者自訂的資料型別等。
  - 如果是Cursor資料指標型別，其傳回值是會因為執行不同而有所改變的，故必須加上VARYING的關鍵字。
  - 在用 EXEC 執行帶有輸出參數的預存程序時，程序後面輸出參數所對應的位置，應該以變數做為參數，同時後面也要加上OUTPUT關鍵字。

範例16-8(a) 建立一預存程序列出查詢各類書籍的平均價格、最高價格。

```
USE 北風出版社
GO
CREATE PROCEDURE usp_statistics_salary
(@M書籍分類 int,
@M平均價格 money OUTPUT,
@M最高價格 money OUTPUT)
AS
SELECT @M平均價格=AVG(價格), @M最高價格= MAX(價格)
FROM 書籍
WHERE 分類代號 = @M書籍分類
GO
-- 執行預存程序
DECLARE @MKIND int
DECLARE @MAVG money
DECLARE @MMAX money
SET @MKIND = 3
EXEC  usp_statistics_salary  @MKIND , @MAVG  OUTPUT , @MMAX  OUTPUT
PRINT '書籍分類: ' + CONVERT( varchar , @MKIND)
PRINT '平均價格:'  + CONVERT( varchar , @MAVG)
PRINT '最高價格:' + CONVERT( varchar , @MMAX)
```

結果：



# 執行預存程序命令

- 執行預存程序的敘述語法如下：

```
[ [ EXEC [ UTE ] ]
  [ @return_status = ]
    { procedure_name [ ;number ] | @procedure_name_var }
    [ [ @parameter = ] { value | @variable [ OUTPUT ] | [ DEFAULT ] } ]
       [ ,...n ]
  [ WITH RECOMPILE ]
```

*@return_status*：是用來儲存預存程序傳回碼(return code)的變數。預存程序的傳回碼可以用來判斷預存程序的執行狀態。

***procedure_name*** [ ;*number* ]：指相同程序名稱之群組預存程序，若要執行群組中的某一編號(*number*)的預存程序，就需在預存程序後加上編號數目。例如： EXEC MyPoP;2。

*@procedure_name_var*：可將預存程序的名稱存於某一定義的變數名稱中。

範例16-10 執行儲存在變數中的預存程序名稱

```
DECLARE @ProcName  char(30)
SET @ProcName = 'Northwind.dbo.Employees'
EXEC @ProcName
```

範例16-12 建立一帶參數的預存程序，可計算某部門的員工人數與平均薪資。

```
SE 北風出版社
GO
CREATE PROC usp_Dept_Cnt_Avg
(@Dept char(10),
@DeptCount tinyint OUTPUT,
@DeptAvg money OUTPUT)
AS
SELECT @DeptCount=COUNT(*),@DeptAvg=AVG(薪資) FROM 學貫工作坊 WHERE 部門=@Dept
GO
------執行usp_Dept_Cnt_Avg 預存程序
DECLARE @Mcnt tinyint
DECLARE @Mavg money

----- 執行帶輸出參數的預存程序後面也要加上OUTPUT關鍵字
EXEC usp_Dept_Cnt_Avg  '業務部' ,  @Mcnt   OUTPUT ,  @Mavg   OUTPUT

PRINT '業務部門人數:' + CONVERT( varchar , @Mcnt )
PRINT '業務部門平均薪資:'+CONVERT( varchar , @Mavg )
```

結果：

```
業務部門人數:5
業務部門平均薪資:45000.00
```

# 傳遞輸入參數的方法

- ◎方法一：未指名輸入參數名稱
  指直接在執行的預存程序後填入傳入值，傳入值必須依預存程序參數的位置給定，參數值間須以逗點(，)隔開。
- ◎方法二：指名輸入參數名稱
  執行預存程序時，傳入的參數值要指定給哪個輸入參數，因此，輸入的次序可不按照輸入參數的次序排列。

範例16-11請寫出指定員工編號，並調整加薪的預存程序。

```
CREATE PROC EmpAddSalary
@員工編號 varchar(5),
@薪資調整 money
AS
UPDATE 員工表
  SET AdjSalary = AdjSalary + @薪資調整
  WHERE EmployeeId = @員工編號
GO

-- 方法一：未指名輸入參數名稱的執行
EXECUTE EmpAddSalary  'A105' , 4000                    ---必須按順序傳入值

--方法二：未指名輸入參數名稱的執行
EXECUTE EmpAddSalary @薪資調整=4000 , @員工編號='A105'   ---可不按順序給定值
```

# 使用傳回碼與RETURN敘述

- 在呼叫預存程序時都應該有回傳碼，以用來判斷預存程序的呼叫是否成功，若成功，程式下一步的流程可依回傳值來決定應如何執行。
- SQL Server系統將 -1到14作為預存程序的系統回傳值，其值所代表意義如下：

| 傳回碼(return code) | 註 解 說 明 |
|---|---|
| 0 | 執行成功 |
| -1 | 找不到物件 |
| -2 | 資料型別錯誤 |
| -3 | 執行程序爲 deadlock victim |
| -4 | 存取權限錯誤 |
| -5 | 語法錯誤 |
| -6 | 其他使用者錯誤 |
| -7 | 資源上的錯誤 |
| -8 | non-fatal 內部問題 |
| -9 | 達到系統上限 |
| -10 | Fatal 內部一致性錯誤 |
| -11 | Fatal 內部一致性錯誤 |
| -12 | 資料表或索引表損毀 |
| -13 | 資料庫損毀 |
| -14 | 硬體錯誤 |

# 使用傳回碼與RETURN敘述(Cont.)

- 除了上面系統回傳值，也可以在預存程序中利用下面語法:

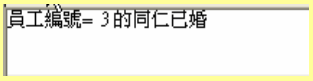
  **RETURN <integer expression>**

  回傳自己設定的回傳值(整數值) 。
- 至於預存程序中如果有其他資料型別要回傳，則得運用預存程序指令語法中的輸出參數**OUTPUT**關鍵字來完成。

範例16-13 建立一預存程序，查詢某員工編號的婚姻狀況。

```
USE 北風出版社
GO
CREATE  PROC  usp_Employee_Married
@M編號  int,
@M婚姻  char(4)  OUTPUT
AS
SELECT @M婚姻=婚姻狀況  FROM 學貫工作坊 WHERE 員工編號 = @M編號
IF @@ROWCOUNT = 0
   RETURN 0
ELSE
   RETURN 1
GO
--- 執行預存程序傳回值做判斷的
DECLARE @RetValue int
DECLARE @M婚姻 char(4)
DECLARE  @M編號 int
SET @M編號 = 3
EXEC  @RetValue = usp_Employee_Married  @M編號 , @M婚姻 OUTPUT

IF @RetValue =0
   PRINT '沒有員工編號=' + STR(@M編號,3) + ' 的資料'
ELSE
   PRINT '員工編號='  + STR(@M編號,3) + ' 的同仁' + @M婚姻
```

結果：



# 動態命令EXECUTE執行預存程序

- 預存程序在執行期間才會動態建立的命令敘述或指派參數值，這稱為動態命令執行預存程序。 其執行技巧有下面兩種：
  - 使用EXECUTE命令執行字元字串。
  - 使用sp_executesql執行Unicode字元字串。

- 方法一：用EXECUTE執行字元字串：

  **EXEC [ UTE ] ( { *@string_variable* | [ N ] '*tsql_string*' }**

  **[ + *...n* ] )**

- 將要執行的命令敘述以字元字串方式儲存一個變數*(@string_variable)*中，再使用EXECUTE來直接執行該字元字串或變數。
- 也可以將多個命令敘述串在一起，再以 EXECUTE 執行，執行命令後的字串必需包含在一對小括號中，且命令字串必須全都是由字元資料所組成的。
- 變數可以是 char、varchar、nchar 或 nvarchar 資料型別。

範例16-14 以EXEC來查詢動態的前3名資料

```
DECLARE @ExecStr varchar(100)
DECLARE @DBNAME varchar(20)
DECLARE @TOPN int
SET @DBNAME = '北風出版社'
SET @TOPN = 3
SET @ExecStr ='USE ' + @DBNAME +CHAR(13)
SET @ExecStr = @ExecStr  +  'SELECT TOP  ' + STR(@TOPN,3) + ' 姓名,部門,薪資 '
                         + 'FROM 學貫工作坊 ORDER BY 薪資 DESC'
EXEC (@ExecStr)
```

結果：

| | 姓名 | 部門 | 薪資 |
|---|---|---|---|
| 1 | 楊維婷 | 業務部 | 68000.0000 |
| 2 | 張志強 | 資訊部 | 65000.0000 |
| 3 | 陳天生 | 製造部 | 55000.0000 |

範例16-15 以EXEC()執行帶有參數的命令字串。下面是查詢加班時數超過某數值的員工資料。

```
DECLARE @Movertime  int
DECLARE @ExecSQLStr varchar(200)
SET @Movertime = 10
SET @ExecSQLStr = 'SELECT 姓名,部門,加班時數  FROM 北風出版社.dbo
                  WHERE 加班時數 > ' + CONVERT(varchar,@Movert
EXEC (@ExecSQLStr)

SET @Movertime = 20
--由於傳遞參數值改變,執行字串必須重新建立
SET @ExecSQLStr = 'SELECT 姓名,部門,加班時數 FROM 北風出版社.dbo.
                  WHERE 加班時數 >' + CONVERT(varchar,@Movertir
EXEC (@ExecSQLStr)
```

結果：

| | 姓名 | 部門 | 加班時數 |
|---|---|---|---|
| 1 | 王凱翔 | 資訊部 | 20 |
| 2 | 陳美惠 | 會計部 | 18 |
| 3 | 黃賢昌 | 行銷部 | 12 |
| 4 | 楊維婷 | 業務部 | 20 |
| 5 | 李思涵 | 財務部 | 35 |
| 6 | 陳建業 | 財務部 | 12 |
| 7 | 李家銘 | 製造部 | 30 |

| | 姓名 | 部門 | 加班時數 |
|---|---|---|---|
| 1 | 李思涵 | 財務部 | 35 |
| 2 | 李家銘 | 製造部 | 30 |
| 3 | 白珍珍 | 行銷部 | 21 |
| 4 | 江淑芬 | 會計部 | 23 |
| 5 | 曾玉娴 | 製造部 | 23 |
| 6 | 葉國華 | 業務部 | 40 |

- 方法二：用 sp_executesql 執行 Unicode字元字串：

```
sp_executesql [@stmt =] stmt
  [    {, [@params =] N'@parameter_name  data_type
  [,...n]' }
       {, [@param1 =] 'value1' [,...n] } ]
```

- 說明：
- 將要執行的命令敘述以Unicode字元字串方式儲存一個變數中，再使用系統預存程序sp_executesql來直接執行該字元字串或變數。
- 但此種方式須注意，若有多列命令敘述要執行時，必須在sp_executesql執行前先串接在一起才可給sp_executesql來執行。
- 這種方法可傳回一結果集，而回傳碼值0 (成功)、1 (失敗)。

範例16-17 使用sp_executesql執行動態命令字串查詢。下面是執行查詢「北風出版社」
資料庫中「作者」資料表的所有資料。

```
DECLARE @MDBNA nvarchar(30)
DECLARE @MTBNA nvarchar(30)
DECLARE @MSQLStr nvarchar(200)

SET @MDBNA = N'北風出版社'
SET @MTBNA = N'作者'

-- 將執行指令串成字串,在使用sp_executesql來執行

SET @MSQLStr = N'USE ' + @MDBNA + ' SELECT * FROM ' + @MTBNA

EXEC sp_executesql @MSQLStr
```

| | 作者代號 | 作者姓名 | 電話 | 地址 | 是否簽訂合約 |
|---|---|---|---|---|---|
| 1 | 101 | 李民立 | (02)8245-3434 | 台北市內湖一段111巷1號1樓 | 1 |
| 2 | 102 | 施銘東 | (02)2222-1234 | 台北市南京東路五段4號3樓 | 1 |
| 3 | 103 | 林玉芬 | (02)2755-1245 | 台北市中山北路六段14號11樓 | 0 |
| 4 | 104 | 方大榮 | (02)2288-1111 | 台北市延平北路五段13巷12號3樓 | 0 |
| 5 | 105 | 林軍之 | (02)2921-1142 | 台北縣新店市中山路一段154號 | 1 |
| 6 | 106 | 郭利永 | (02)2142-9999 | 台北縣板橋市文化路一段14巷1號4樓 | 0 |
| 7 | 107 | 陳佳玲 | (02)2212-4205 | 台北縣中和市中山路一段12巷12號2樓 | 0 |
| 8 | 108 | 白珍珍 | (02)2363-1234 | 台北市羅斯福路二段24號11樓 | 0 |
| 9 | 109 | 張一飛 | (03)245-1236 | 桃園縣經國路14號12樓 | 1 |

範例16-18 使用sp_executesql執行帶有參數的動態命令字串查詢。下面是查詢「學貫工作坊」資料表內，員工年資在N 年內，但薪資超過 M 元的員工資料。

```
DECLARE @MSQLStr nvarchar(200)          --設定動態命令執行字串
DECLARE @MArgStr   nvarchar(100)        --設定參數資料型態字串
DECLARE @MArg1    int                   --設定年資變數
DECLARE @MArg2    money                 --設定薪資變數
--設定帶有@MYearc 和 @MSalary 參數的動態執行字串
        SET @MSQLStr = N'USE 北風出版社  SELECT 姓名,DateDiff(yy ,僱用日期,getdate()) AS 年資,薪資 FROM 學貫工作坊
                        WHERE DateDiff(yy ,僱用日期,getdate()) < @MYear  AND 薪資 >  @MSalary'
-- 設定相關使用到參數資料型態的字串
SET @MArgStr = N'@MYear int , @MSalary money'
-- 設定傳入參數的值
SET @MArg1 = 3
SET @MArg2 = 35000
-- 執行動態命令並傳遞參數
EXECUTE sp_executesql @MSQLStr , @MArgStr , @MArg1 ,@MArg2
-- 動態改變傳入參數值,重新執行SQL命令字串
SET @MArg1 = 4
SET @MArg2 = 50000
EXECUTE sp_executesql @MSQLStr , @MArgStr , @MArg1 ,@MArg2
```

以筆者撰稿日2004/4/24爲基準

| | 姓名 | 年資 | 薪資 |
|---|---|---|---|
| 1 | 盧志偉 | 2 | 38000.0000 |
| 2 | 連心蘭 | 2 | 52000.0000 |

| | 姓名 | 年資 | 薪資 |
|---|---|---|---|
| 1 | 連心蘭 | 2 | 52000.0000 |

結果:

# EXECUTE()與sp_executesql的比較

- EXECUTE須將所有型別資料皆轉成字串，會造成程式設計的麻煩，較沒有彈性，
- EXECUTE不具備真正傳遞參數的能力，因為命令字串需要重新建立。
- Sp_executesql執行命令，它具有參數傳遞功能。
- Sp_executesql執行命令， T-SQL的命令敘述只需要建立一次，參數可以不必轉換成字元型別而直接使用。
- 使用sp_executesql執行會較有彈性。

# 利用INSERT…EXECUTE命令

- 當預存程序執行後回傳的是一個結果集，如果希望將結果集加入到某一個已存在的資料表後，可以透過INSERT…EXECUTE指令來完成，其語法如下：

  **INSERT <table_name> EXEC <procedure_name>**

範例16-19 在「北風資料庫」中，建立一預存程序，統計「學貫工作坊」中各部門已婚及未婚員工的平均薪資、最高薪資、最低薪資資料，並將資料存入一資料表中。

```
USE 北風出版社
GO
--建立一個存在的資料表,以儲存預存程序統計後的結果
CREATE TABLE DeptMarriedSalary
(
部門  varchar(20),
婚姻狀況 char(4),
平均薪資 money,
最高薪資 money,
最低薪資 money
)
GO
CREATE PROC  usp_Dept_Salary_Statistics
AS
SELECT 部門 , 婚姻狀況 , AVG(薪資) , MAX(薪資) , MIN(薪資)   FROM 學貫工作坊
      GROUP BY 部門 , 婚姻狀況  ORDER BY  部門
GO
-- 將預存程序usp_Dept_Salary_Statistics執行後回傳結果集存入已存在資料表DeptMarriedSalary中

INSERT  DeptMarriedSalary  EXEC  usp_Dept_Salary_Statistics

SELECT  *  FROM  DeptMarriedSalary
```

結果:

| | 部門 | 婚姻狀況 | 平均薪資 | 最高薪資 | 最低薪資 |
|---|---|---|---|---|---|
| 1 | 行銷部 | 已婚 | 36166.6666 | 38000.0000 | 32500.0000 |
| 2 | 行銷部 | 未婚 | 52000.0000 | 52000.0000 | 52000.0000 |
| 3 | 財務部 | 已婚 | 34750.0000 | 35000.0000 | 34500.0000 |
| 4 | 財務部 | 未婚 | 37666.6666 | 48000.0000 | 25000.0000 |
| 5 | 會計部 | 已婚 | 41750.0000 | 45000.0000 | 38500.0000 |
| 6 | 會計部 | 未婚 | 33833.3333 | 39500.0000 | 28000.0000 |
| 7 | 業務部 | 已婚 | 39500.0000 | 50000.0000 | 28500.0000 |
| 8 | 業務部 | 未婚 | 55500.0000 | 68000.0000 | 43000.0000 |

# 自動執行預存程序

- 預存程序的執行可以由SQL Server來自動執行。
- SQL Server自動來執行預存程序，一般都是一些屬於系統管理性作業、定期執行某些作業等工作。
- 自動執行預存程序，大大減少人工作業與系統管理人員的負擔。自動執行設定應注意下面事項：
  - 預存程序必須在master系統資料庫中，且必須被資料庫擁有者dbo擁有才可被設定成系統自動執行。
  - 自動執行的預存程序，所用的使用權限與伺服器的固定角色 sysadmin 的成員相同。
  - 自動執行檔的錯誤資訊會寫入SQL Server的錯誤記錄檔中。
  - 所撰寫的自動執行的預存程序最好不要傳回任何結果集，因為系統無法處理這些結果集資料。
  - 雖然設定了自動執行的預存程序，但如果設定scan for startup procs 組態，啟動 SQL Server 時就不會自動執行任何預存程序。

設定預存程序是否爲系統自動執行預存程序，可利用sp_procoption系統預存程序來設定，其語法如下：

```
sp_procoption [ @ProcName = ] 'procedure'
              , [ @OptionName = ] 'option'
              , [ @OptionValue = ] 'value'
```

範例16-20 假設在master資料庫下有一個BackupTask預存程序，請先將該預存程序設爲自動執行預存程序後，再下指令取消它自動執行預存程序的功能。

```
USE MASTER
GO
/*設爲自動執行預存程序*/
EXEC sp_procoption 'BackupTack' , 'startup' ,'true'
GO
/*取消自動執行預存程序*/
EXEC sp_procoption 'BackupTack' , 'startup' , 'false'
```

# 巢狀式預存程序

- 在設計預存程序時，程序內也可以再呼叫另一個預存程序，這種情況就稱為「巢狀式預存程序」。
- 巢狀式預存程序可以一層一層的呼叫下去。
- SQL Server的巢狀預存程序呼叫的層數最多至32層，巢狀式層數可由系統全域變數@@NESTLEVEL來查看。
- 在巢狀式預存程序中，被呼叫的預存程序將可存取呼叫它的預存程序中的的所有物件。

範例16-21 巢狀式預存程序呼叫，查看「Northwind」資料庫的「Employees」資料表中某員工的主管有哪些人。

```
USE  Northwind
GO
CREATE PROC usp_SearchBoss
(
@MEmpID  int
)
AS
DECLARE  @MBoss   int

SELECT @MBoss = ReportsTo  FROM Employees  WHERE EmployeeId = @MEmpID

IF @MBoss IS NOT NULL AND @@NESTLEVEL <= 32
BEGIN
  SELECT @MEmpID AS 員工 , @MBoss AS 主管, @@NESTLEVEL AS 巢狀深度
  EXEC  usp_SearchBoss  @MBoss
END
GO

-- 執行預存程序usp_SearchBoss,查看員工編號=6 的主管有哪些人

EXEC usp_SearchBoss  6
```

結果：

| | 員工 | 主管 | 巢狀深度 |
|---|---|---|---|
| 1 | 6 | 5 | 1 |

| | 員工 | 主管 | 巢狀深度 |
|---|---|---|---|
| 1 | 5 | 2 | 2 |

# 重新編譯預存程序

- 在SQL Server中如果資料庫有改變時(如：新增或修改索引、變更資料表)，而會影響到某些預存程序的執行效能時，就應該重新編譯這些預存程序，將原始的查詢計劃最佳化。
- 最佳化會在重新啟動 SQL Server後，首次執行預存程序時自動發生。
- SQL Server 提供三種重新編譯預存程序的方法：
  - 利用sp_recompile 系統預存程序，強迫其在下一次執行預存程序時重新編譯。
  - 在建立預存程序的指令敘述中指定 WITH RECOMPILE 選項，指示 SQL Server每次執行這個預存程序時都會重新編譯。
  - 在執行預存程序時，後面指定 WITH RECOMPILE 選項，以強迫在執行時重新編譯預存程序。
- 重新編譯預存程序會耗費掉許多系統資源，所以，除非必要或者是在重新編譯後會獲得好處，否則，應儘量避免無謂的重新編譯作業。

# 預存程序的安全性管理

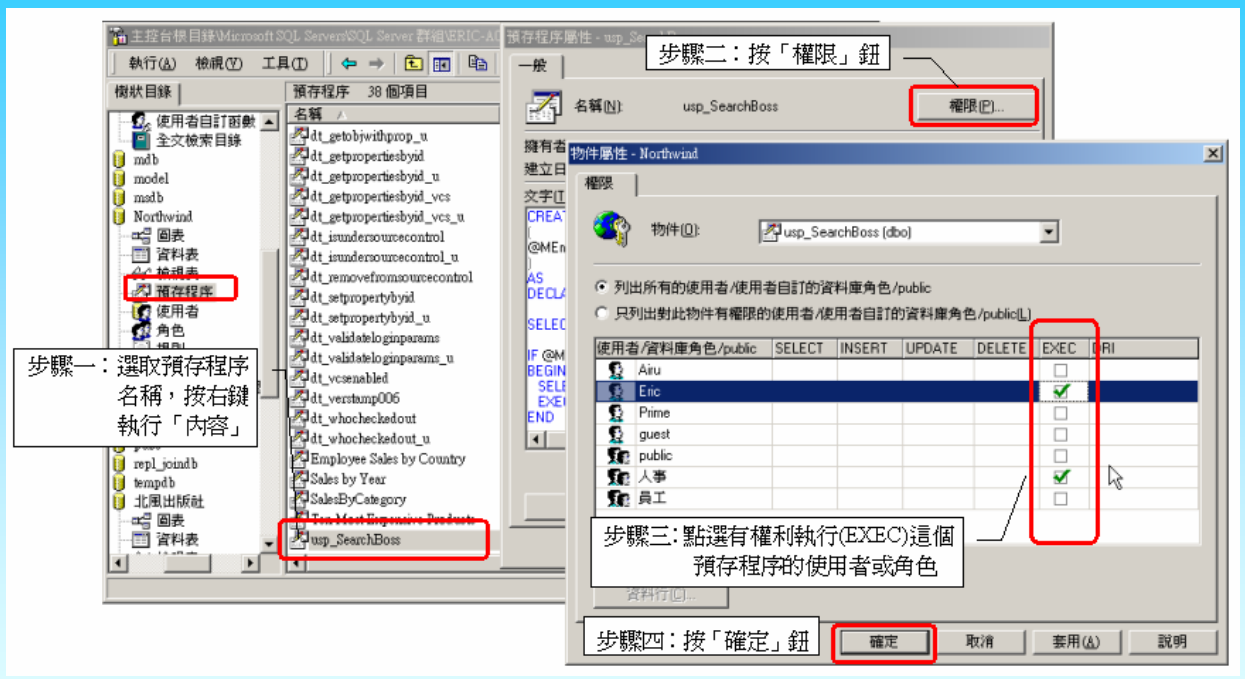
- 預存程序的建立可以讓資料庫的管理人員安全且有效的管理資料
- 由資料庫管理人員來建立預存程序，讓使用者再透過預存程序來存取資料，可以保障資料存取的安全問題。
- 在SQL Server上有權利建立預存程序的使用者：
  - 對伺服器具sysadmin伺服器角色的使用者
  - 對資料庫具db_owner資料庫角色的使用者
  - 對資料庫具db_ddladmin資料庫角色的使用者
- 只要具有上面三種角色之一的使用者就可以來建立預存程序，但是這些使用者除了可以建立預存程序外，還可以建立其他如：資料表、檢視表、規則、預設值等資料庫物件，
- 如果不想讓使用者有那麼大的權利可以建立其他資料庫物件，那就直接賦予使用者CREATE PROCEDURE的權限就可以，無需具備上面的三種角色。

# 預存程序的安全性管理(Cont.)

- 使用者具有建立預存程序的權限外，還需要讓使用者具有EXEC執行預存程序的權限，否則，使用者所建立的預存程序就只有自己、具sysadmin伺服器角色及具db_owner資料庫角色的成員有權可以執行。
- 建立預存程序的使用者可以透過GRANT EXEC敘述將該預存程序的執行權利指定給有權執行的其他使用者。

範例16-22 賦予Eric使用者有EXEC執行usp_SearchBoss的權限。

```
GRANT EXEC on usp_SearchBoss to Eric
```


步驟一：選取預存程序名稱，按右鍵執行「內容」
步驟二：按「權限」鈕
步驟三：點選有權利執行(EXEC)這個預存程序的使用者或角色
步驟四：按「確定」鈕
usp_SearchBoss
物件屬性 - Northwind
權限
確定
取消
套用(A)
說明

# 預存程序的限制

- 在預存程序中的T-SQL敘述，有些是被限制禁止使用的，包括：
  - CREATE DEFAULT
  - CREATE PROCEDURE
  - CREATE RULE
  - CREATE TRIGGER
  - CREATE VIEW
- 在使用預存程序名稱時要注意，不同使用者所建立的預存程序名稱可以是相同的。
- 如果沒有特別使用owner.procedure的形式來指定預存程序的話，則系統會以建立該預存程序的使用者作為該預存程序的擁有者。
- 為了避免如上所說使用到不同使用者的預存程序問題，造成安全管制上困擾，建議預存程序的建立都由dbo來建立處理，讓整個預存程序的安全控管制較單純。