



Net.Data 參考手冊

目錄

注意事項	vii
商標	vii
第1章 簡介	1
誰應閱讀本書	1
如何閱讀語法圖	1
關於本書的範例	2
更深入地了解 Net.Data	2
第2章 Net.Data 巨集語法	3
第3章 語言結構	5
共用結構	5
變數名稱	6
變數參照	7
備註區塊	8
DEFINE 區塊或陳述式	9
ENVVAR 陳述式	12
EXEC 區塊或陳述式	13
FUNCTION 區塊	15
函數呼叫 (@)	20
HTML 區塊	22
HTML IF 陳述式	24
INCLUDE 陳述式	27
INCLUDE_URL 陳述式	29
LIST 陳述式	31
Macro IF 區塊	33
MESSAGE 區塊	36
REPORT 區塊	39
ROW 區塊	41
TABLE 陳述式	43
第4章 變數	45
條件式變數	45
隱藏變數	46
List 變數	47
Table 變數	48
預先定義的變數	49
DTW_CURRENT_FILENAME	50
DTW_CURRENT_LAST_MODIFIED	51
DTW_MACRO_FILENAME	52
DTW_MACRO_LAST_MODIFIED	53
DTW_MP_PATH	54
DTW_MP_VERSION	55
隱含的 Table 變數	55
N_columnName	56
Nn	57
NLIST	58
NUM_COLUMNS	59
RETURN_CODE	60

ROW_NUM	61
TOTAL_ROWS	62
V_columnName	63
Vn	64
VLIST	65
報表變數	65
ALIGN	66
DTW_APPLET_ALTTEXT	67
DTW_DEFAULT_REPORT	68
DTW_HTML_TABLE	69
DTW_PRINT_HEADER	70
DTW_SET_TOTAL_ROWS	71
RPT_MAX_ROWS	72
START_ROW_NUM	73
資料庫變數	73
DATABASE	74
DB_CASE	75
DB2PLAN	76
DB2SSID	77
DTW_SAVE_TABLE_IN	78
LOCATION	79
LOGIN	80
PASSWORD	81
SHOWSQL	82
TRANSACTION_SCOPE	83
第5章 Net.Data 內建函數	85
一般函數	86
DTW_ADDQUOTE	87
DTW_DATE	88
DTW_GETENV	89
DTW_GETINIDATA	90
DTW_HTML_ENCODE	91
DTW_QHTML_ENCODE	93
DTW_SETENV	95
DTW_TIME	96
DTW_URLESCSEQ	98
數學函數	99
DTW_ADD	100
DTW_DIVIDE	101
DTW_DIVREM	102
DTW_FORMAT	103
DTW_INTDIV	106
DTW_MULTIPLY	107
DTW_POWER	108
DTW_SUBTRACT	109
字串函數	110
DTW_ASSIGN	111
DTW_CONCAT	112
DTW_DELSTR	113
DTW_INSERT	114
DTW_LASTPOS	116

DTW_LENGTH	117
DTW_LOWERCASE.	118
DTW_POS	119
DTW_REVERSE	120
DTW_STRIP	121
DTW_SUBSTR.	122
DTW_TRANSLATE	123
DTW_UPPERCASE	125
字組函數.	126
DTW_DELWORD.	127
DTW_SUBWORD.	128
DTW_WORD	129
DTW_WORDINDEX.	130
DTW_WORDLENGTH	131
DTW_WORDPOS.	132
DTW_WORDS	134
表格函數.	134
DTW_TB_DLIST	136
DTW_TB_DUMPH	138
DTW_TB_DUMPV	139
DTW_TB_HTMLLENCODE.	140
DTW_TB_INPUT_CHECKBOX	141
DTW_TB_INPUT_RADIO	142
DTW_TB_INPUT_TEXT	143
DTW_TB_LIST	144
DTW_TB_SELECT	146
DTW_TB_TABLE.	147
DTW_TB_TEXTAREA	149
純本文檔案介面	149
純本文檔案介面區隔字元	150
純本文檔案介面函數.	150
DTWF_APPEND	151
DTWF_CLOSE.	153
DTWF_DELETE	154
DTWF_INSERT	156
DTWF_OPEN	158
DTWF_READ	159
DTWF_REMOVE	161
DTWF_SEARCH	162
DTWF_UPDATE	164
DTWF_WRITE.	166
Web 登記函數.	167
DTWR_ADDENTRY.	168
DTWR_CLEARREG	169
DTWR_CREATEREG	170
DTWR_DELENTY	171
DTWR_DELREG	172
DTWR_LISTREG	173
DTWR_LISTSUB	174
DTWR_RTVENTRY	175
DTWR_UPDATEENTRY	176

附錄A. DB2 WWW Connection	177
%EXEC_SQL	178
HTML_INPUT	179
HTML_REPORT	180
SQL	181
SQL_MESSAGE	182
SQL_REPORT	183
SQL_CODE	184
 附錄B. Net.Data 訊息和訊息碼	 185
附錄C. Net.Data 平台參考手冊	189
附錄D. Net.Data for AIX	195
附錄E. Net.Data for OS/2	197
附錄F. Net.Data for OS/390	199
必要的軟體	199
可選用的軟體	199
安裝與架構概觀	200
安全性與身份驗證	200
將 Net.Data 架構為存取 DataJoiner	200
程式材料	201
機器可讀取材料	202
Net.Data FMID.	202
 附錄G. Net.Data for OS/400	 203
Net.Data for OS/400 簡介.	203
一般的 OS/400 概念	204
HTTP 伺服器	204
整合式檔案系統	204
啟動和執行 Net.Data	206
在 AS/400 上使用 Net.Data	210
所支援的語言環境	211
REXX (DTW_REXX) 語言環境	211
SQL (DTW_SQL 或 SQL) 語言環境.	212
系統 (DTW_SYSTEM) 語言環境	215
從 DB2 WWW Connection 移轉到 Net.Data	216
問題分析.	216
 附錄H. Net.Data for Windows NT.	 219
名詞解釋.	221
索引	223

注意事項

在本書內所提及的各種 IBM 產品、程式或服務，並不暗示 IBM 有意於其營業的國家內，發行這些產品、程式、或服務。凡提及 IBM 產品、程式或服務時，並不表示或暗示只可使用 IBM 產品、程式或服務。受限於 IBM 的有效智慧財產權或其他法律保護的權利，任何功能相同的產品、程式或服務，也可以用來代替 IBM 產品、程式或服務。但是，在使用這些並非由 IBM 專門設計的產品時，一切與產品有關的評估及驗證，均由使用者自行負責。本文件中包含著 IBM 所擁有之專利或暫准專利。使用者不享有本文件內容之專利權您可以用書面方式來查詢特許權限，來信請寄到：

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
U.S.A.

對於本程式所授權使用的人，若爲了下列目的而想取得有關的資訊：(1) 獨立建立的程式與其他程式（包括這個程式）之間的資訊交換，(2) 相互使用已交換之資訊方法，則應聯絡：

IBM Corporation
555 Bailey Avenue, W92/H3
P.O. Box 49023
San Jose, CA 95161-9023

上述資訊之取得有其特殊要件，在某些情況下，必須付費方得使用。

商標

下列詞彙是 IBM 公司在美國及其他國家的商標：

AIX	IBM
AS/400	IMS
DB2	OS/2
Distributed Relational	OS/390
Database Architecture	OS/400
DRDA	RACF

下列詞彙是其他公司的商標：

UNIX 是 X/Open Company Limited 在美國及其他國家獨家授權的註冊商標。

Microsoft、Windows、Windows NT 和 Windows 95 等標誌，都是 Microsoft Corporation 的商標或註冊商標。

前面標有雙星號 (**) 的其他公司、產品及服務名稱，可能是其他公司的商標或服務標示。

第1章 簡介

誰應閱讀本書

本書將對 Net.Data 語言架構、變數、和功能用法以及語法做一般性的說明。它是專為有涉獵於規劃和撰寫 Net.Data 應用程式的人所寫的。

若要了解本書所討論的概念，您需要先熟悉如何使用 Web 伺服器、簡單的 SQL 陳述式、以及 HTML (包括如何使用 HTML 表格頁面)。

本書可能會提及一些已發表，但尚未上市的产品或特性。

如何閱讀語法圖

下列為本書所使用的語法圖所引用的規則：

- 從左至右，從上至下，並遵循線條的路徑來讀取語法圖。

▶▶—— 符號表示陳述式的開始。

——▶ 符號表示陳述式將續接下一行。

▶—— 符號表示陳述式接自上一行。

——▶▶ 符號表示陳述式終止。

非完整陳述式之語法單位的圖案將以 ▶—— 符號作為開頭，而以 ——▶ 符號作為結尾。

- 必須的項目將會出現在水平線條（主路徑）上。

▶▶—required_item————▶▶

- 可選用的項目會出現在主路徑下方。

▶▶—required_item——[optional_item]————▶▶

如果有某個可選用的項目出現在主路徑的上方，則該項目對陳述式的執行並無任何影響，且只為了可讀性。

▶▶—required_item——[optional_item]————▶▶

- 如果您可以從兩個或兩個以上的項目中選擇，則它們會以垂直堆疊的方式呈現。

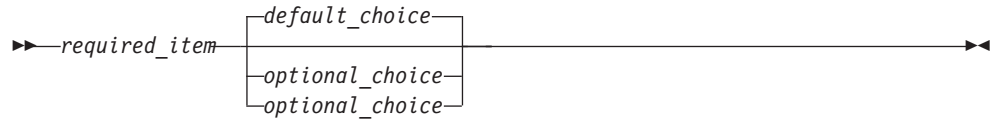
如果您必須 選擇其中的一個項目，則該堆疊的項目會呈現在主路徑上。

▶▶—required_item——[required_choice1
required_choice2]————▶▶

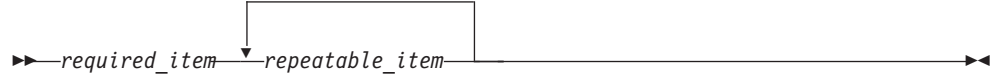
如果您可選擇性地選擇其中一個項目，則整個堆疊將會出現在主路徑的下方。

▶▶—required_item——[optional_choice1
optional_choice2]————▶▶

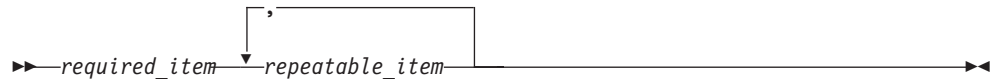
如果其中一個項目是預設值，則它會出現在主路徑的上方，而其他的選項會顯示在下方。



- 若在主線上有一個指回左邊的箭頭，表示該項目可以重複。



如果重複箭頭包含一個標點符號，則您必需使用所指定的標點符號來區隔重複的項目。



在堆疊上方的重複箭頭表示您可以重複堆疊中的項目。

- 關鍵字會以大寫字母來表示（例如 FROM）。在 Net.Data 中，並不限定關鍵字是大寫或小寫。但是非關鍵字的詞彙將會以小寫字母方式呈現（例如，*column-name*）。它們是代表使用者提供的名稱或值。
- 如果有標示標點符號、括弧、數學運算子或其他符號，則您必須將它們當作語法的一部份來輸入。

關於本書的範例

使用於本書中的範例將會儘量簡化，以說明特定的觀念，而不是顯現每個 Net.Data 結構可被使用的方式。某些範例只是片段而已，並不可單獨作業。

更深入地了解 Net.Data

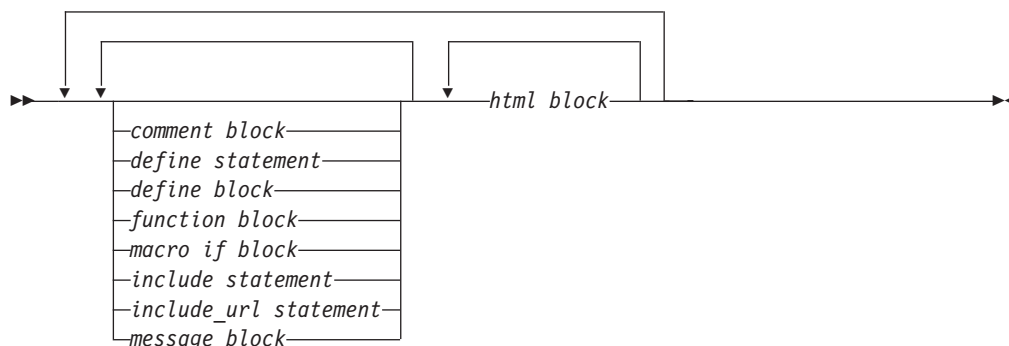
在「全球資訊網」(World Wide Web) 中可獲得更多的資訊，其網址為：

<http://www.software.ibm.com/data/net.data>

您可以取得 Net.Data 的巨集樣本、示範程式、本書籍的最新副本、常問問題列示以及論壇上問到的問題。

第2章 Net.Data 巨集語法

Net.Data 巨集是一個純本文檔案，其是由一連串 Net.Data 巨集語言的陳述式所組成。每個陳述式是由一個或多個語言結構所組成，換句話說，就是由關鍵字、特殊的字元、字串、名稱、和變數所組成。下圖描述了語法上有效之 Net.Data 巨集的整體結構。關於整體結構中每個元素之語法的詳細資訊，請參閱第5頁的『第3章 語言結構』。



Net.Data 巨集包含兩個部分：宣告部分以及 HTML 部分。您可以任何順序來多次使用這些部分。

- 宣告部分 包括使用在 HTML 部分中的定義。在變數參照或函數呼叫使用在宣告部分中的變數和函數之前，必須先對它們加以定義。
- HTML 區塊；在變數參照或函數呼叫使用在 HTML 部分中的變數和函數之前，必須先對它們加以定義。

Net.Data 巨集語言是自由格式的語言，可讓您有彈性地撰寫您的巨集。在您 Net.Data 巨集中一行的長度不可超過 64K 位元組。除非特別指示，否則會忽略額外的空白字元。每個 Net.Data 巨集的語言結構，包括用來定義結構的幾個其他的元素，將於下列的章節一一做說明，。Net.Data 巨集語言支援 DB2 WWW Connection 語言元素以便能夠與先前的版本相容。雖然這些語言元素於 第177頁的『附錄A. DB2 WWW Connection』中有加以說明，但建議您使用 Net.Data 的特性。

將這些範例顯示給您許多有關語言結構、變數、函數、和在您巨集檔案中的其他元素之使用方式。您可以在下列的網址中，從 Net.Data Web 的頁面下載其餘更廣泛的範例和展示程式：

<http://www.software.ibm.com/data/net.data>

第3章 語言結構

本章說明 Net.Data 巨集使用的語言結構。有關 Net.Data 巨集的整體說明，請參閱第3頁的『第2章 Net.Data 巨集語法』。語言結構是由 Net.Data 巨集中的一個關鍵字及一個陳述式或區塊所組成，可指定不同的變數類型，並執行如併入檔案等其它特殊作業。本章說明下列的語言結構關鍵字。

- 第8頁的『備註區塊』
- 第9頁的『DEFINE 區塊或陳述式』
- 第12頁的『ENVVAR 陳述式』
- 第13頁的『EXEC 區塊或陳述式』
- 第15頁的『FUNCTION 區塊』
- 第20頁的『函數呼叫 (@)』
- 第22頁的『HTML 區塊』
- 第24頁的『HTML IF 陳述式』
- 第27頁的『INCLUDE 陳述式』
- 第29頁的『INCLUDE_URL 陳述式』
- 第31頁的『LIST 陳述式』
- 第33頁的『Macro IF 區塊』
- 第36頁的『MESSAGE 區塊』
- 第39頁的『REPORT 區塊』
- 第41頁的『ROW 區塊』
- 第43頁的『TABLE 陳述式』

語言結構可能是一個陳述式或一個區塊或二者，各語言結構都會註明。

- 使用陳述式，可以一次定義一個變數
- 使用區塊，可以定義數個變數

您可以在單行上使用雙引號 (") 定義變數，或使用大括弧 ({}) 跨越多行定義變數。

每一個語言結構說明，都有下列資訊：

目的 定義為什麼在 Net.Data 巨集中使用此關鍵字

語法 提供關鍵字陳述式或區塊邏輯結構的圖解。

參數 定義語法圖中的所有元素，並鏈結其它關鍵字語法及範例。

上下文 說明關鍵字在 Net.Data 巨集結構中的使用位置。

限制 定義可以包含哪些元素，並指定用法限制。

範例 提供簡單的範例，並說明如何在 Net.Data 巨集中使用關鍵字陳述式或區塊。

共用結構

語言結構說明中，經常使用下列的語法元素。

變數名稱

目的

一或多個名稱，使用一個句點 (.) 連接每一個額外的名稱。名稱，是一個以英文字母或底線開頭的英文或數字字串，可以是英文字母、數字、或底線字元的任何組合。

語法



變數參照

目的

變數參照，可傳回先前定義的變數值，指定時使用 \$ 及 ()。例如：若 VAR = 'abc'，則 \$(VAR) 傳回值 'abc'。變數參照是在執行時被加以運算。EXEC 陳述式或區塊定義變數後，Net.Data 讀取到變數參照時，即會執行指定的動作。

Net.Data 巨集定義所要參照的變數後，才能執行參照。如果未定義變數，則傳回空字串。

語法

►►\$—(—*variable_name*—)—————►◄

備註區塊

目的

備註區塊，可用來說明 `Net.Data` 巨集功能。

語法

►► `%{text%}` ◄◄

參數

text 一或多行上的任何本文。`Net.Data` 不處理所有備註內容。

上下文

備註，必須在所有其它 `Net.Data` 巨集區塊的外部。

限制

接受任何本文或字元。

範例

範例 1:

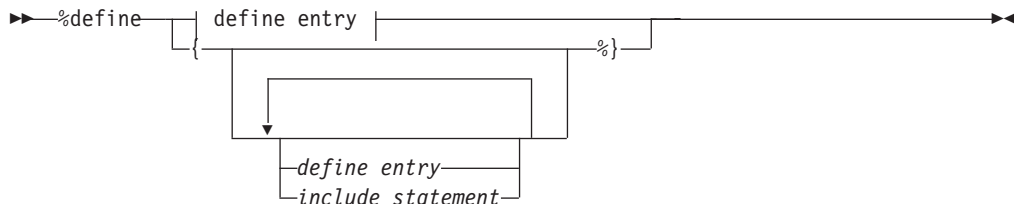
```
%{  
這是備註區塊。不限制行數及字元。Net.Data 不處理它的內容。  
%}
```

DEFINE 區塊或陳述式

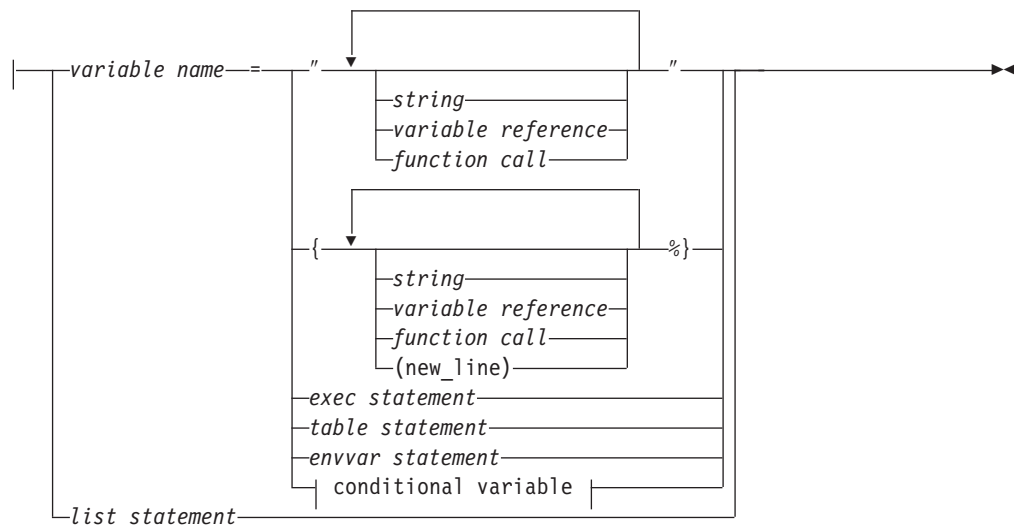
目的

DEFINE 區塊或陳述式，可以定義變數名稱。變數名稱必須以一個字母或底線開頭，可以使用任何英數字元或底線。所有變數名稱都區分大小寫，但 *N_columnName* 及 *V_columnName* 除外 (有關這兩個例外的詳細資訊，請參閱第55頁的『隱含的 Table 變數』)。

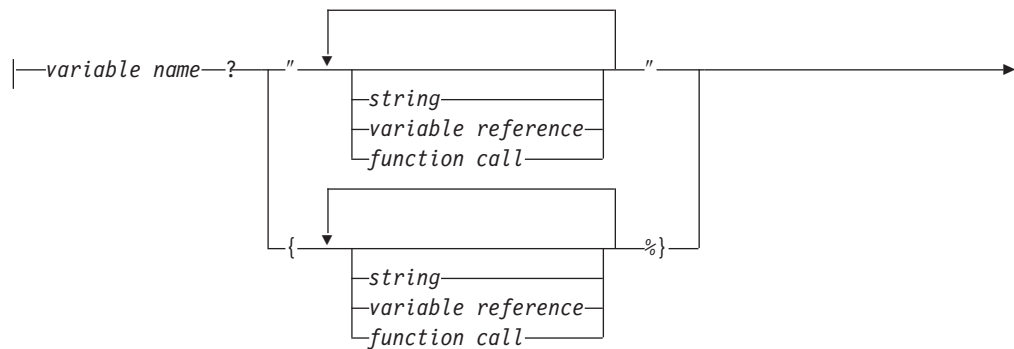
語法

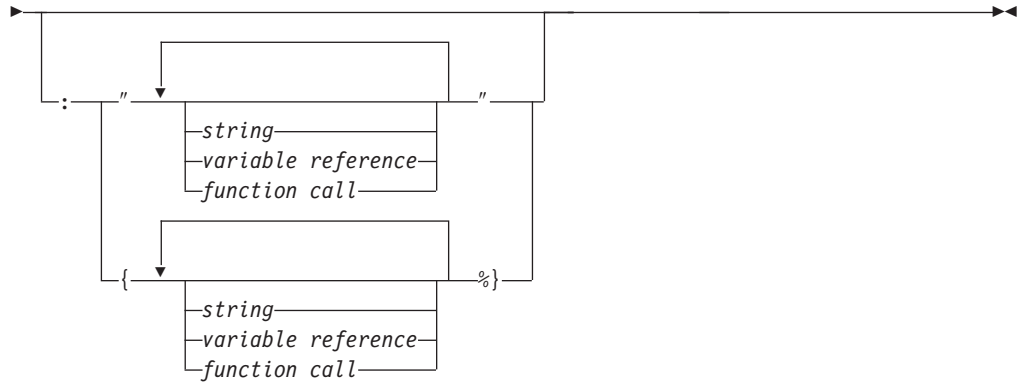


define entry



conditional variable





參數

%define

此關鍵字負責定義變數。

define entry:

variable name

一或多個名稱，使用一個句點 (.) 連接每一個額外的名稱。有關語法的詳細資訊，請參閱第7頁的『變數名稱』。

string

任何順序的英數字元和標點符號，但不含定位字元 (tabulator)、換行字元、或空白。

variable reference

變數參照，可傳回先前定義的變數值，指定時使用 \$ 及 ()。例如，若 VAR = 'abc'，則 \$(VAR) 傳回值 'abc'。有關語法的詳細資訊，請參閱第8頁的『變數參照』。

function call

以指定的引數呼叫先前定義的一或多個 FUNCTION 區塊。有關語法及範例，請參閱第20頁的『函數呼叫 (@)』。

exec statement

EXEC 陳述式。當參照變數或呼叫函數時，所要執行的一個外部程式的名稱。有關語法及範例，請參閱第13頁的『EXEC 區塊或陳述式』。

table statement

TABLE 陳述式。定義相關資料的集合，有一個相同記錄或橫列的陣列、以及一個說明各列欄位的直欄名稱陣列。有關語法及範例，請參閱第43頁的『TABLE 陳述式』。

envvar statement

ENVVAR 陳述式。參照環境變數。有關語法及範例，請參閱第12頁的『ENVVAR 陳述式』。

conditional variable

設定一個根據另一個變數或字串值而產生的變數值。

list statement

LIST 陳述式。定義一些變數，可用來建置一個有定界符號的值列示。有關語法及範例，請參閱第31頁的『LIST 陳述式』。

include statement

INCLUDE 陳述式。讀取並合併檔案至 Net.Data 巨集中。有關語法及範例，請參閱第27頁的『INCLUDE 陳述式』。

上下文

必須在巨集的 IF 區塊之內，或在 Net.Data 巨集宣告部份的其他區塊之外。

限制

可以包含下列元素：

- 條件變數
- LIST 陳述式
- TABLE 陳述式
- 變數參照
- INCLUDE 陳述式
- EXEC 陳述式
- 函數呼叫
- ENVVAR 陳述式

範例

範例 1：簡單的變數定義。

```
%DEFINE var1 = "orders"  
%DEFINE var2 = "${var1}.html"
```

執行時，變數參照 `$(var2)` 被運算為 `orders.html`。

範例 2：使用兩個連續引號，併入字串中的引號。單獨兩個引號則表示空字串。

```
%DEFINE hi = "say ""hello"""  
%DEFINE empty = ""
```

顯示時，變數 `hi` 的值為 `say "hello"`。變數 `empty` 是空值。

範例 3：DEFINE 區塊可讓您用一個 DEFINE 陳述式來定義多個變數。

```
%DEFINE{ DATABASE = "testdb"  
          home = "http://www.software.ibm.com"  
          SHOWSQL = "YES"  
          PI = "3.14150"  
%}
```

範例 4：本例中的定義區塊可讓您定義一個跨越多行的變數。

```
%DEFINE text = {This variable definition  
                spans two lines  
%}
```

ENVVAR 陳述式

目的

ENVVAR 陳述式參照 DEFINE 區塊中的環境變數。使用這個方法參照環境變數，比使用 DTW_GETENV 更有效率。有關詳細資訊，請參閱第89頁的『DTW_GETENV』。

語法

►►—%envvar—◄◄

上下文

%DEFINE 區塊或陳述式。

參數

%envvar

一種關鍵字，用來指定 DEFINE 區塊中的環境變數，可以取得巨集中任何地方的環境變數值。

限制

不可以包含其它元素。

範例

範例 1：本例中的 %ENVVAR，傳回環境變數 SERVER_SOFTWARE 的值，亦即 Web 伺服器名稱。

```
%DEFINE SERVER_SOFTWARE = %ENVVAR
```

```
%HTML (REPORT){  
伺服器是 $(SERVER_SOFTWARE)。  
%}
```

EXEC 區塊或陳述式

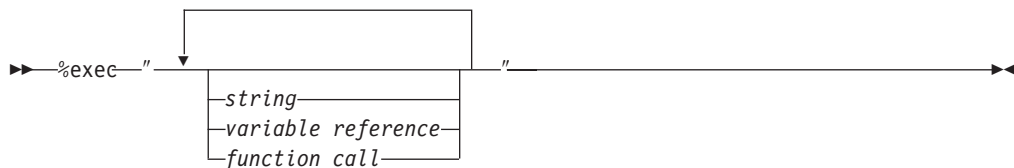
目的

EXEC 陳述式或區塊，指定當您呼叫變數參照或函數時執行的外部程式。

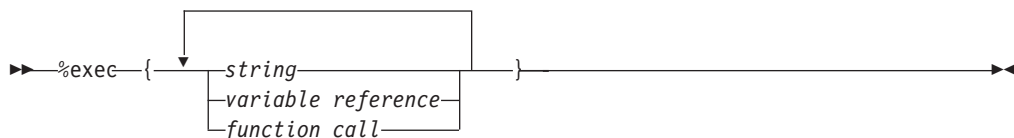
EXEC 陳述式及區塊用於兩個不同的上下文，二者的語法不同，視使用位置而定。
EXEC 陳述式用於 DEFINE 區塊，EXEC 區塊用於 FUNCTION 區塊。

語法

EXEC 陳述式用於 DEFINE 區塊：



EXEC 區塊用於 FUNCTION 區塊：



參數

%exec

一個關鍵字，指定當您參照變數或呼叫函數時執行的外部程式名稱。Net.Data 遇到 EXEC 陳述式定義的變數參照時，它處理 EXEC 陳述式宣告的變數內容。

name

名稱，是一個以英文字母或底線開頭的英文或數字字串，可以是英文字母、數字、或底線字元的任意組合。

string

任何順序的英數字元和標點符號，但不含定位字元 (tabulator)、換行字元、或空白。

variable reference

變數參照，可傳回先前定義的變數值，指定時使用 \$ 及 ()。例如，若 VAR = 'abc'，則 \$(VAR) 傳回值 'abc'。有關語法的詳細資訊，請參閱第8頁的『變數參照』。

function call

以指定的引數呼叫先前定義的一或多個 FUNCTION 區塊。有關語法及範例，請參閱第20頁的『函數呼叫 (@)』。

上下文

在這些上下文中：

- DEFINE 區塊
- FUNCTION 區塊

限制

可以包含這些元素：

- 字串
- 變數參照
- 函數呼叫

範例

範例 1：本例在每次參照變數 mycall 時，執行 MYEXEC.EXE。

```
%DEFINE mycall = %EXEC "MYEXEC.EXE $(empno)"
```

```
%HTML (report){  
<P>這是您要求的報表：  
<HR>$(mycall)  
%}
```

範例 2：本例在呼叫函數 my_rexx_pgm 時執行 mypgm.cmd。

```
%FUNCTION(DTW_REXX) my_rexx_pgm(INOUT a, b, IN c, INOUT d){  
    %EXEC{ mypgm.cmd 這是測試 %}  
%}
```

FUNCTION 區塊

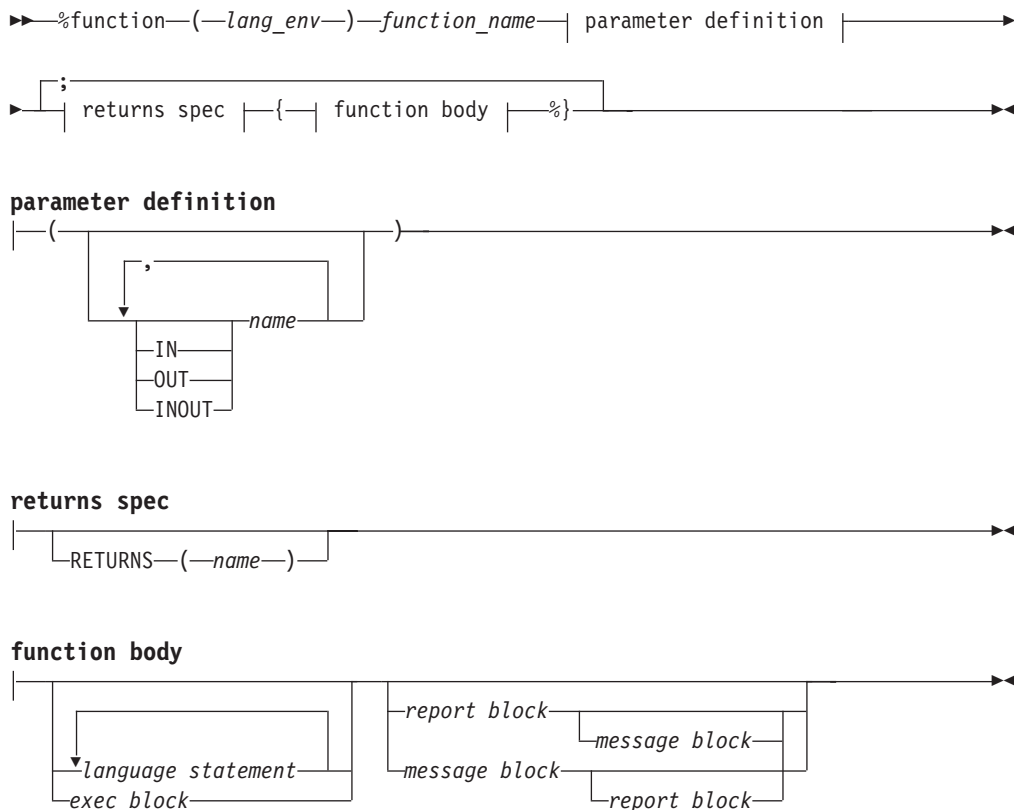
目的

FUNCTION 區塊定義一個可以從 `Net.Data` 巨集呼叫的次常式。FUNCTION 區塊中的可執行檔陳述式，可以是直接由語言環境解譯過來的語言陳述式，也可以指示呼叫外部程式。

函數定義中使用的 EXEC 陳述式或區塊，必須是 FUNCTION 區塊中的唯一可執行檔陳述式。將可執行檔陳述式傳給語言環境之後，必須先把 EXEC 陳述式中的程式檔案名稱，附加到起始設定檔之 `EXEC_PATH` 架構陳述式決定的路徑名稱中。最後將產生的字串，傳給要執行的語言環境。

語言環境處理 EXEC 陳述式時使用的方法，取決於特殊語言環境。僅 REXX、系統、以及 Perl 語言環境，支援 EXEC 陳述式。

語法



參數

%function

一種關鍵字，指定一個可以從 `Net.Data` 巨集呼叫的次常式。FUNCTION 區塊中的可執行檔陳述式，可以包含直接由語言環境解譯過來的語言陳述式，也可以指示呼叫外部程式。

lang_env

支援函數定義的語言環境。有關詳細資訊請參閱 *Net.Data 語言環境手冊*。

function_name

定義的函數名稱。一個以英文字母或底線開頭的英文或數字字串，可以是英文字母、數字、或底線字元的任意組合。

name

一個以英文字母或底線開頭的英文或數字字串，可以是英文字母、數字、或底線字元的任意組合。

parameter definition:

IN 關鍵字。有關詳細資訊請參閱「Net.Data 程式設計手冊」。

OUT

關鍵字。有關詳細資訊請參閱「Net.Data 程式設計手冊」。

INOUT

關鍵字。有關詳細資訊請參閱「Net.Data 程式設計手冊」。

returns spec:**RETURNS**

關鍵字。有關詳細資訊請參閱「Net.Data 程式設計手冊」。

function body:**language statement**

函數定義中指定之語言環境的有效語法陳述式，例如：REXX、SQL、或 Perl。有關現用語言的語法及用法，請參閱「參考手冊」。

exec block

EXEC 陳述式。當參照變數或呼叫函數時，所要執行的一個外部程式名稱。有關語法及範例，請參閱第14頁的『EXEC 區塊或陳述式』。

report block

REPORT 區塊。為函數呼叫的輸出製作格式的指示。您可以在報表中使用表頭及註腳資訊。有關語法及範例，請參閱第39頁的『REPORT 區塊』。

message block

MESSAGE 區塊。一組回覆碼、相關訊息、以及傳回函數呼叫時 Net.Data 執行的動作。有關語法及範例，請參閱第36頁的『MESSAGE 區塊』。

上下文

必須在 Net.Data 巨集宣告部份的陳述式及區塊之外。

限制

可以包含這些元素：

- EXEC 陳述式
- MESSAGE 區塊
- REPORT 區塊
- Language 陳述式

範例

下面是一般範例，並未涵蓋所有的語言環境。有關特定語言環境使用 FUNCTION 區塊的詳細資訊，請參閱 *Net.Data 語言環境手冊*。

範例 1：REXX 子字串函數。

```
%DEFINE lstring = "longstring"
%FUNCTION(DTW_REXX) substring(IN x, y, z) RETURNS(s) {
    s = substr("$x)", $(y), $(z));
}%
%DEFINE a = {@substring(lstring, "1", "4")%} %{ assigns "long" to a %}
```

計算 *a* 後，發現 *@substring* 函數呼叫，並執行子字串 FUNCTION 區塊。替換 FUNCTION 區塊中的可執行檔陳述式中的變數，然後將本文字串

s = substr("longstring", 1, 4) 傳給 REXX 直譯器執行。因為已指定 *RETURNS* 子句，所以計算 *a* 時的 *@substring* 函數呼叫值會被置換成 "long"，也就是 *s* 值。

範例 2：呼叫外部 REXX 程式。

- Net.Data 巨集：

```
%FUNCTION(DTW_REXX) my_rexx_pgm(INOUT a, b, IN c, OUT d) {
    %EXEC{ mypgm.cmd 這是測試 %}
}%
%HTML(INPUT) {
    <P> 原始變數值：$(w) $(x) $(z)
    <P> @my_rexx_pgm(w, x, y, z)
    <P> 修改後的變數值：$(w) $(x) $(z)
}%
```

變數 *w* 及 *x*，對應函數中的 INOUT 參數 *a* 及 *b*。他們的值以及對應 IN 參數 *c* 的 *y* 值，應該先由 HTML 表格頁面輸入或從 DEFINE 陳述式定義。變數 *a* 及 *b*，在參數 *a* 及 *b* 傳回值時會指定新值。當 OUT 參數 *d* 傳回一值時，則會定義變數 *z*。

- REXX 程式 mypgm.cmd：

```
/* REXX 程式範例
範例 2 */
/* 測試引數 */
num_args = arg();
say '有' num_args '個引數';
do i = 1 to num_args;
    say '引數' i '是'"arg(i)'"';
end;
/* 設定 Net.Data 傳來的變數 */
d = a || b || c; /* 連結 a, b, c 形成 d */
a = ''; /* 重設 a 為空字串 */
b = ''; /* 重設 b 為空字串 */
return;
```

- mypgm.cmd 的輸出：

```
有 1 引數
引數 1 是 "這是測試"
```

EXEC 要求 REXX 語言環境讓 REXX 直譯器執行外部 REXX 程式 mypgm.cmd。由於 REXX 語言環境可以直接與 REXX 程式共用 Net.Data 變數，所以它先指定 REXX 變數 *a*、*b*、以及 *c* (Net.Data 變數 *w*、*x*、以及 *y* 的值)，才執行 mypgm.cmd。Mypgm.cmd 可以直接使用 REXX 陳述式中的變數 *a*、*b*、以及 *c*。程式終止時，取回 REXX 程式中的 REXX 變數 *a*、*b*、以及 *d*，並將這些值指定給 Net.Data 變數 *w*、

x、以及 z。由於定義 my_rexx_pgm %FUNCTION 區塊時未使用 RETURNS 子句，所以 @my_rexx_pgm 函數呼叫值是空字串 "" (回覆碼為 0) 或是 REXX 程式的回覆碼值 (回覆碼不為零)。

範例 3：SQL 查詢及報表。

```
%FUNCTION(DTW_SQL) query_1(IN x, IN y) {
    SELECT customer.num, order.num, part.num, status
    FROM customer, order, shippingpart
    WHERE customer.num = '$(x)'
        AND customer.ordernumber = order.num
        AND order.num = '$(y)'
        AND order.partnumber = part.num
    %REPORT{
        <P>訂單狀態：
        <P>$(NLIST)
        <UL>
        %ROW{
            <LI>$(V1) $(V2) $(V3) $(V4)
        %}
        </UL>
    %}
    %}
%DEFINE customer_name="IBM"
%DEFINE customer_order="12345"
%HTML(REPORT) {
    @query_1(customer_name, customer_order)
%}
```

@query_1 函數呼叫以 "IBM" 替換 SELECT 陳述式中的 \$(x)、以 "12345" 替換 \$(y)。由於定義 SQL 函數 query_1 無法識別輸出表格變數，所以使用預設表格 (詳細資訊請參閱 TABLE 變數區塊)。REPORT 區塊中參照的 NLIST 及 Vi 變數，是由預設表格定義所定義。REPORT 區塊產生的報表，放在呼叫 query_1 函數的輸出 HTML 中。

範例 4：系統呼叫執行 Perl script。

- Net.Data 巨集：

```
%FUNCTION(DTW_SYSTEM) today() RETURNS(result) {
    %exec{ perl "today.prl" %}
    %}
%HTML(INPUT) {
    @today()
    %}
```

- Perl 程式 today.prl：

```
$date = 'date';
chop $date;
open(DTW, "> $ENV{DTWPIPE}") || die "無法開啓：$!";
print DTW "result = \"$date\"\n";
```

「系統」語言環境解譯 FUNCTION 區塊中的可執行檔陳述式時，是經由 C 語言 system() 函數呼叫，傳給作業系統。這個方法無法向 REXX 語言環境般，直接向可執行檔陳述式傳收 Net.Data 變數。「系統」語言環境是以下列方法傳收變數：

- 把輸入參數當作系統環境變數，以 putenv() 函數來傳送，以執行程式來取回。不同語言參照不同的變數。Unix cshell script 參照環境變數時，在環境變數名稱之前加一個 '\$'，例如，\$x。Perl 語言參照環境變數時，是參照相關陣列 %ENV，例如，%ENV{'x'}。DOS 批次 (.BAT) 檔參照用 '%' 圍住的變數名稱，例如，%x%。
- 把輸出參數傳回給語言環境時，是將輸出參數寫至在環境變數 DTWPIPE 中傳送的管路名稱。寫至指名之管路的資料，它的格式是 name="value"，同 DEFINE 陳述式。若以這種方式編寫輸出參數對應的變數名稱，新值會置換現行值。如果您編寫的變數名稱未對應到輸出參數，系統不予處理。

遇到 @today 函數呼叫時，Net.Data 在可執行檔陳述式上替代變數。本例的可執行檔陳述式中沒有 Net.Data 變數，所以不替代變數。可執行檔陳述式及參數傳給「系統」語言環境後，此語言環境立即建立一個指名之管路，並將環境變數 DTWPIPE 設定為指名之管路。

然後，再以 C system() 函數呼叫，呼叫外部程式。外部程式以唯寫方式開啓管路，再以標準資料流檔的方式寫入管路中。如果您在 CGI 模式中使用 Net.Data，則 HTML 輸出會寫至 STDOUT 資料流。本例將系統日期程式的輸出，指定至 FUNCTION 區塊 RETURNS 子句識別的變數結果中。結果變數的這個值，則會置換 HTML 區塊中的 @today() 函數呼叫。

範例 5：Perl 語言環境。

```
%FUNCTION(DTW_PERL) today() RETURNS(result) {
    $date = 'date';
    chop $date;
    open(DTW, "> $ENV{DTWPIPE}") || die "無法開啓：$!";
    print DTW "result = \"\$date\"\n";
}%
%HTML(INPUT) {
    @today()
}%
```

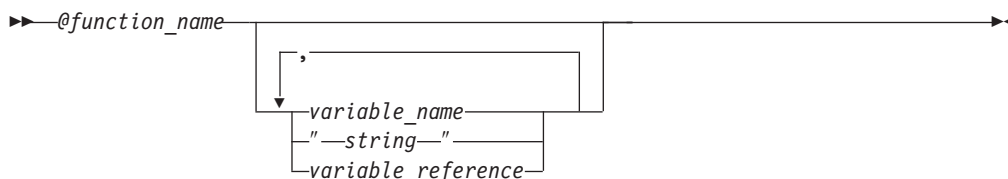
您可以比較本例與「範例 4」中 EXEC 陳述式的使用方式。範例 4 的「系統」語言環境不瞭解如何解譯 Perl 程式，但知道如何呼叫外部程式。EXEC 陳述式要它以外部程式的方式呼叫 perl 程式。實際的 Perl 語言陳述式，是由外部 Perl 程式解譯。範例 5 的 Perl 語言環境可以直接解譯 Perl 語言陳述式，所以沒有 EXEC 陳述式。

函數呼叫 (@)

目的

函數呼叫可以以指定引數來呼叫先前定義的 FUNCTION 區塊。Net.Data 巨集定義函數後，您才能指定函數呼叫。

語法



參數

@function_name

任何現有函數的名稱。一個以英文字母或底線開頭的英文或數字字串，可以是英文字母、數字、或底線字元的任意組合。

variable name

一或多個名稱，使用一個句點 (.) 連接每一個額外的名稱。有關語法的詳細資訊，請參閱第7頁的『變數名稱』。

string

任何順序的英數字元和標點符號，但不含定位字元 (tabulator)、換行字元、或空白。

variable reference

變數參照，可傳回先前定義的變數值，指定時使用 \$ 及 ()。例如，若 VAR = 'abc'，則 \$(VAR) 傳回值 'abc'。有關語法的詳細資訊，請參閱第8頁的『變數參照』。OS/400 的函數呼叫不支援變數參照。

上下文

在這些上下文中可以找到：

- HTML 區塊
- REPORT 區塊
- ROW 區塊
- DEFINE 區塊
- Macro IF 區塊
- HTML IF 區塊
- MESSAGE 區塊

限制

- 不可以包含其它巨集元素。
- 由於不接受巢狀 SQL 陳述式，所以 SQL 函數定義中不可以有 SQL 函數的函數呼叫。

- 函數定義中只能有 IN 參數定義的變數參照。
- OS/400 的函數呼叫不支援變數參照。

範例

範例 1：呼叫 SQL 函數 formQuery。

```
%FUNCTION(DTW_SQL) formQuery(){
SELECT $(queryVal) from $(tableName)
%}

%HTML (input){
<P>察看 $(tableName) 的哪些直欄？
<FORM METHOD="POST" ACTION="report">
<INPUT NAME="queryVal" TYPE="CHECKBOX" VALUE="NAME">名稱
<INPUT NAME="queryVal" TYPE="CHECKBOX" VALUE="MAIL">電子郵件
<INPUT NAME="queryVal" TYPE="CHECKBOX" VALUE="FAX">傳真
<INPUT TYPE="SUBMIT" VALUE="Submit request">
%}

%HTML (report){
<P>您選取的直欄：
<HR>@formQuery()
%}
```

範例 2：呼叫 REXX 函數 (含輸入及輸出參數)。

```
%FUNCTION(DTW_REXX) my_rexx_pgm(INOUT a, b, IN c, OUT d) {
%EXEC{ mypgm.cmd 這是測試 %}
%}

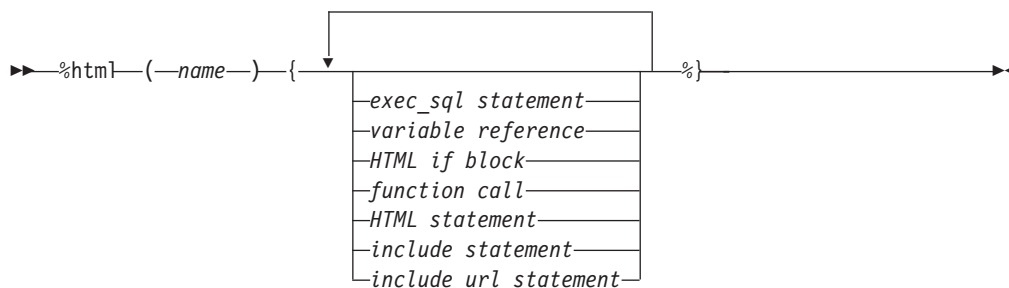
%HTML(INPUT) {
<P> 原始變數值：$(w) $(x) $(z)
<P> @my_rexx_pgm(w, x, y, z)
<P> 修改後的變數值：$(w) $(x) $(z)
%}
```

HTML 區塊

目的

HTML 區塊中也可以含有從屬站 Web 瀏覽器或任何能夠識別 HTML 之工具，所能處理的任何 HTML 標籤或本文。HTML 區塊中也有大部份的 Net.Data 巨集語言陳述式，這些陳述式在執行時被加以運算及執行。Net.Data 會尋找 Net.Data 巨集陳述式並執行它們，而且會假設其它本文都是 HTML，傳給從屬站。

語法



參數

%html

這個關鍵字指定從屬站瀏覽器顯示之 HTML 標籤及本文的所在區塊。

name

一個以英文字母或底線開頭的英文或數字字串，可以是英文字母、數字、或底線字元的任意組合。

exec_sql statement

為了提供相容性才支援的 DB2WWW 版次 1 語言元素。請參閱第177頁的『附錄A. DB2 WWW Connection』或「DB2 全球資訊網版次 1」文件。

variable reference

變數參照，可傳回先前定義的變數值，指定時使用 \$ 及 ()。例如，若 VAR = 'abc'，則 \$(VAR) 傳回值 'abc'。有關語法的詳細資訊，請參閱第8頁的『變數參照』。

HTML if block

Net.Data 巨集 HTML 部份使用的 HTML IF 區塊。執行條件字串處理。比較時，把數值視為字串。有關語法及範例，請參閱第24頁的『HTML IF 陳述式』。

function call

以指定的引數呼叫先前定義的一或多個 FUNCTION 區塊。有關語法及範例，請參閱第21頁的『函數呼叫 (@)』。

HTML statements

包含任何英數字元，以及為從屬站瀏覽器製作格式的 HTML 標籤。

include statement

INCLUDE 陳述式。讀取檔案，併入 Net.Data 巨集中。有關語法及範例，請參閱第27頁的『INCLUDE 陳述式』。

include_url statement

INCLUDE_URL 陳述式。讀取另一個檔案，併入指定此陳述式的 Net.Data Web 巨集中。指定的檔案，可能在區域或遠端伺服器中。有關語法及範例，請參閱第29頁的『INCLUDE_URL 陳述式』。

上下文

必須為一個 Net.Data 區塊或陳述式，而且不能包含在其它 Net.Data 區塊或陳述式中。

可以包含這些元素：

- EXEC_SQL 陳述式
- HTML IF 區塊
- HTML 陳述式
- INCLUDE 陳述式
- INCLUDE_URL 陳述式
- 變數參照
- 函數呼叫

範例

範例 1:

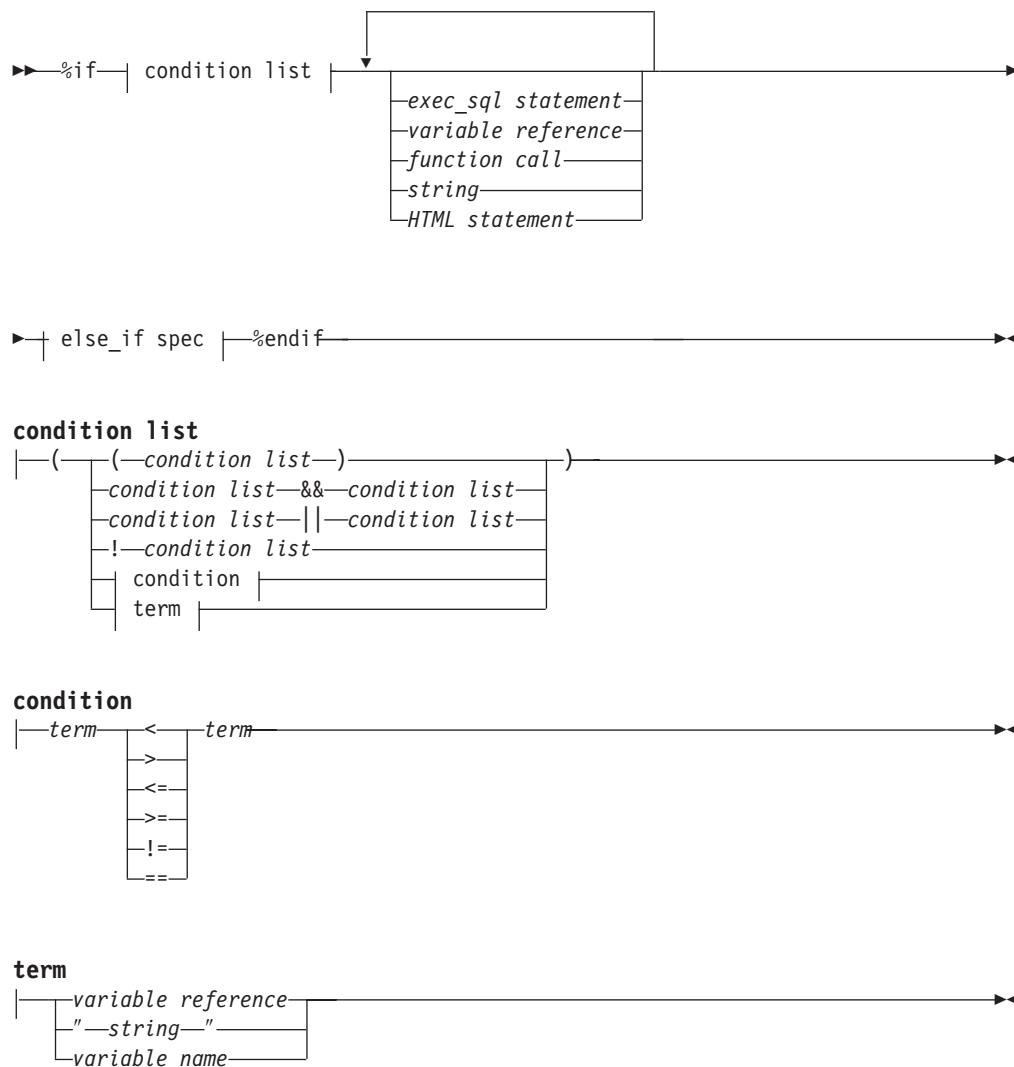
```
%HTML(example1){  
%INCLUDE(header.html)  
<P>您可以將 <EM>任何</EM> HTML 放入 HTML 區塊中。  
SQL 函數呼叫大致如下：  
@xmp1()  
%INCLUDE(footer.html)  
%}
```


HTML IF 陳述式

目的

HTML IF 陳述式執行 Net.Data 巨集 HTML 部份中的條件字串處理。HTML IF 陳述式可用於 HTML 區塊、REPORT 區塊、以及 ROW 區塊。比較時，把數值視為字串。

語法



%elif

此關鍵字可以啟動替代處理路徑，可以包含條件列示及大部份的 `Net.Data` 巨集陳述式。

%endif

此關鍵字可以關閉 `%IF` 區塊。

%else

此關鍵字可以在不符合其它條件列示時，執行相關陳述式。

上下文

在這些上下文中可以找到：

- HTML 區塊
- REPORT 區塊
- ROW 區塊

限制

可以包含這些元素：

- EXEC_SQL 陳述式
- 字串
- HTML 陳述式
- 函數呼叫
- 變數參照

範例

範例 1:

```
%IF ($RETURN CODE) == $(failure_rc))
  <P> 函數呼叫失敗，失敗碼 $(RETURN_CODE)。
%ELIF ($(RETURN_CODE) == $(warning_rc))
  <P> 函數呼叫成功，警告碼 $(RETURN_CODE)。
%ELIF ($(RETURN_CODE) == $(success_rc))
  <P>函數呼叫順利完成。
%ELSE
  <P>函數呼叫傳回不明回覆碼 $(RETURN_CODE)。
%ENDIF
```

範例 2:

```
%IF (name == "world!")
<H2>Hello world!</H2>
%ENDIF
```

INCLUDE 陳述式

目的

INCLUDE 陳述式讀取一個檔案，納入指定此陳述式的 Net.Data 巨集中。

Net.Data 在起始設定檔 INCLUDE_PATH 陳述式指定的目錄中，尋找併入檔。

併入檔的使用方式，與大部份的高階語言相同。您可以插入共用標題及註腳、定義共用變數設定、或將 FUNCTION 區塊定義的共用次常式程式庫納入 Net.Data 巨集中。

語法

►► `%include—"string"` ◄◄

參數

%include

此關鍵字指定讀取檔案後納入 Net.Data 巨集中。

string

任何順序的英數字元和標點符號，但不含換行字元。

上下文

在這些上下文中可以找到：

- DEFINE 區塊
- HTML 區塊
- REPORT 區塊
- ROW 區塊
- HTML IF 區塊
- Macro IF 區塊
- Net.Data 巨集宣告部份的區塊之外

限制

可以包含這些元素：

- 字串

範例

範例 1:

```
%HTML(start){  
%INCLUDE "header.hti"  
...  
%}
```

範例 2:

```
%REPORT {  
  %INCLUDE "report_header.txt"  
  %ROW {  
    %INCLUDE "row_include.txt"  
  }  
  %INCLUDE "report_footer.txt"  
}
```

INCLUDE_URL 陳述式

目的

INCLUDE_URL 陳述式讀取另一個檔案，納入指定此陳述式的 Net.Data 巨集中。指定的檔案，可能在區域或遠端伺服器中。

範例 3 顯示如何呼叫現行巨集中的一個巨集，而不須應用程式使用者選取「提出」按鈕。

語法

►►%include_url—"string"—◄◄

參數

%include_url

此關鍵字指定讀取檔案後納入區域或遠端伺服器中的 Net.Data 巨集。

string

任何順序的英數字元和標點符號，但不含換行字元。

上下文

在這些上下文中可以找到：

- HTML 區塊
- REPORT 區塊
- ROW 區塊
- HTML IF 區塊
- Macro IF 區塊
- Net.Data 巨集宣告部份的區塊之外

限制

可以包含這些元素：

- 字串

OS/390 或 OS/400 平台不支援 INCLUDE_URL。

範例

範例 1：本例將區域伺服器中的一個檔案併入。

```
%include_url "celdemo.htm"
```

範例 2：本例將另一個伺服器中的 gif 檔併入。

```
%include_url "http://www.ibm.com/images/netdata.gif"
```

範例 3：您可以將指向其他 Net.Data 巨集的 URL 併入。本例呼叫巨集檔 custqadd.mac，並將custno當作變數傳送。

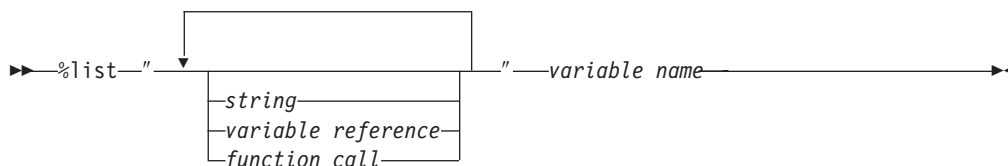
```
%REPORT{  
<P>Current hot pick as of @DTW_rTIME():  
%include_url "http://www.ibm.com/cgi-bin/db2www/hotpic.mac/report?custno=$(custno)"
```

LIST 陳述式

目的

您可以使用 **LIST** 變數，建置以分隔符號區隔的值的列示一些值的區隔列示。當您建構的 SQL 有多個項目時 (就像 **WHERE** 或 **HAVING** 子句中的項目)，**LIST** 變數可以發揮相當大的作用。

語法



上下文

在這些上下文中可以找到：

- **DEFINE** 陳述式

參數

%list

此關鍵字指定使用變數來建置以分隔符號區隔的值的列示。

字串

任何順序的英數字元和標點符號，但不含定位字元 (tabulator)、換行字元、或空白。

變數參照

變數參照，可傳回先前定義的變數值，指定時使用 **\$** 及 **()**。例如，若 **VAR = 'abc'**，則 **\$(VAR)** 傳回值 **'abc'**。有關語法的詳細資訊，請參閱第8頁的『變數參照』。

函數呼叫

以指定的引數呼叫先前定義的一或多個 **FUNCTION** 區塊。有關語法及範例，請參閱第21頁的『函數呼叫 (@)』。

變數名稱

變數名稱。一個以英文字母或底線開頭的英文或數字字串，可以是英文字母、數字、或底線字元的任意組合。有關語法的詳細資訊，請參閱第7頁的『變數名稱』。

限制

可以包含這些元素：

- 變數參照
- 函數呼叫
- 字串

範例

範例 1:

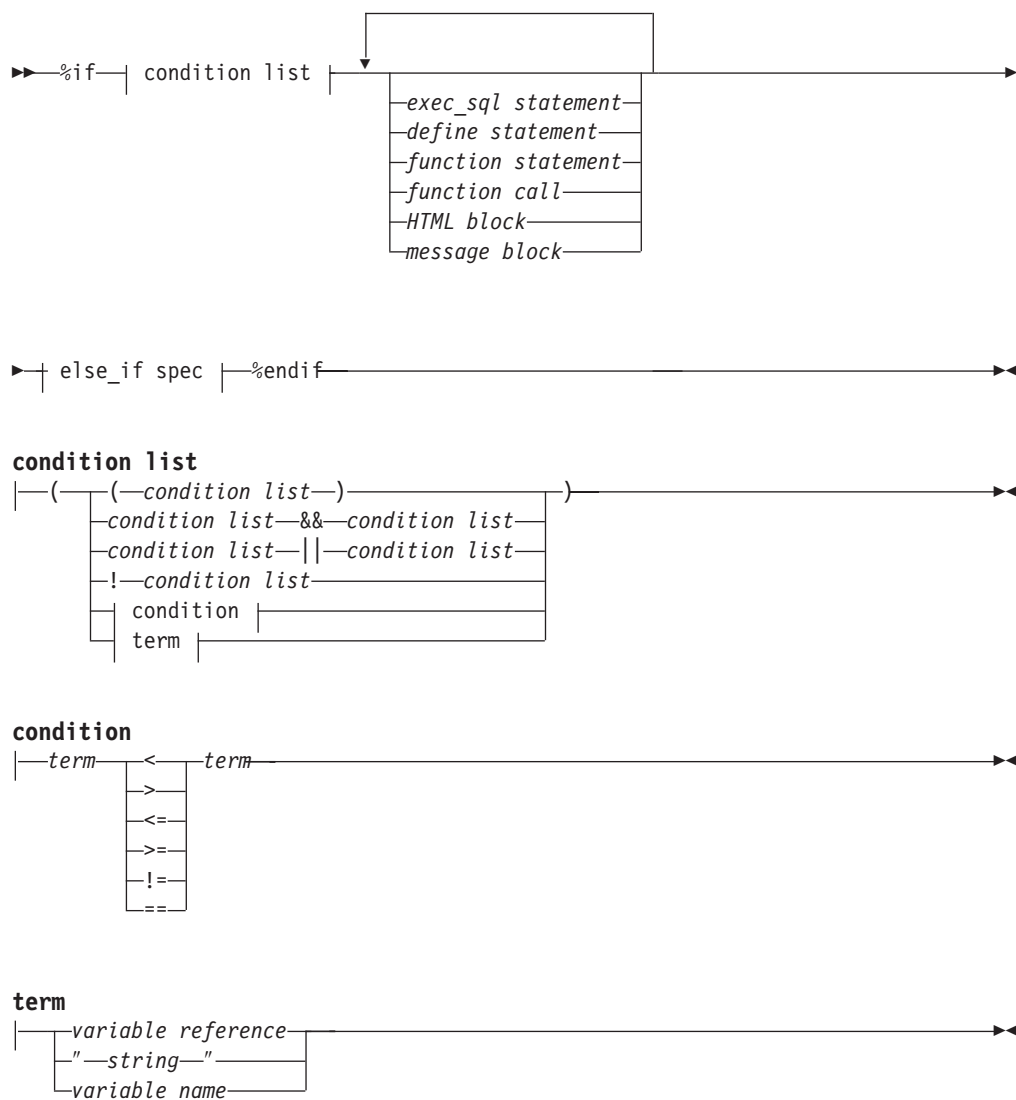
```
%DEFINE{  
DATABASE="custcity"  
%LIST " OR " conditions  
cond1="Sao Paulo"  
cond2="Seattle"  
cond3="Shanghai"  
whereClause=Conditions ? "WHERE ${conditions}" : ""  
%}
```

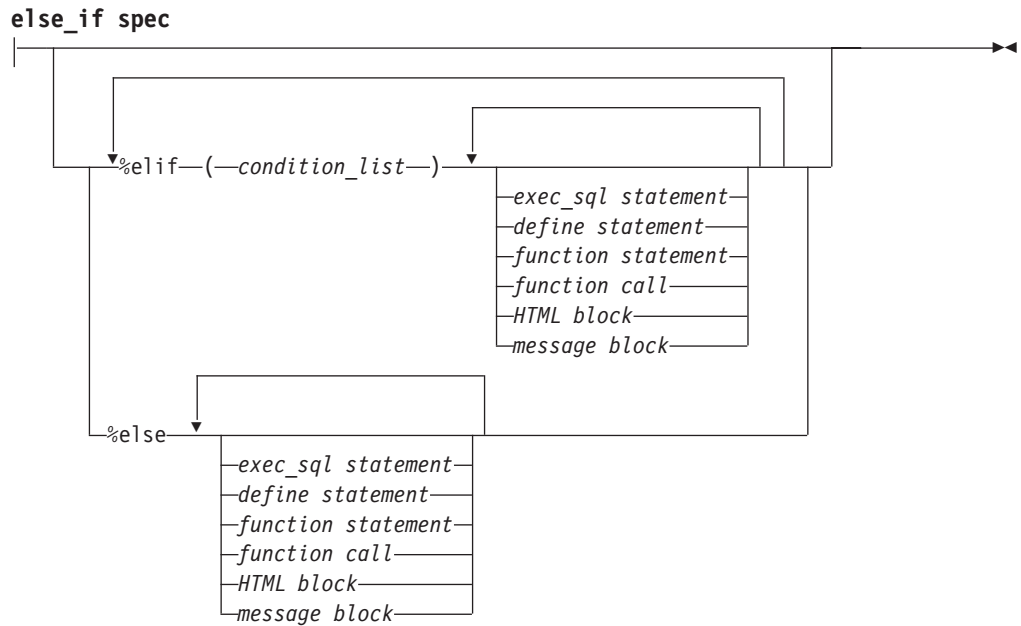
Macro IF 區塊

目的

巨集 IF 區塊執行條件字串處理，可用於 Net.Data 巨集的宣告部份。比較時，把數值視為字串。

語法





參數

%if

此關鍵字指定條件字串處理。比較時，把數值視為字串。

condition list:

condition list

比較值與變數值。列示可以包含條件及詞彙。條件列示可以用布林運算子來連接。您可以在另一個條件列示中建立巢狀條件列示。

condition

使用比較運算子，比較兩個詞彙。Net.Data 只比較字串。代表數值的變數或字串，可以評估為字元字串，但不能評估為數值。

term

一個變數名稱、字串、或變數參照。

exec_sql statement

針對相容性支援的 DB2WWW 版次 1 語言元素。請參閱第177頁的『附錄A. DB2 WWW Connection』或「DB2 全球資訊網版次 1」文件。

define statement

DEFINE 區塊或陳述式。定義變數，設定架構變數。變數名稱必須以一個字母或底線開頭，可以使用任何英數字元或底線。有關語法及範例，請參閱第10頁的『DEFINE 區塊或陳述式』。

function block

一種關鍵字，指定一個可以從 Net.Data 巨集呼叫的次常式。FUNCTION 區塊中的可執行檔陳述式，可以包含直接由語言環境解譯過來的語言陳述式，也可以指示呼叫外部程式。有關語法及範例，請參閱第16頁的『FUNCTION 區塊』。

function call

呼叫先前定義的一或多個 FUNCTION 區塊，含指定引數。有關語法及範例，請參閱第21頁的『函數呼叫 (@)』。

HTML block

包含任何英數字元，以及為從屬站瀏覽器製作格式的 HTML 標籤。

message block

MESSAGE 區塊。一組回覆碼、相關訊息、以及傳回函數呼叫時Net.Data 執行的動作。有關語法及範例，請參閱第36頁的『MESSAGE 區塊』。

%elif

此關鍵字可以啟動選擇方案處理路徑，可以包含條件列示及大部份的 Net.Data 巨集。

%endif

此關鍵字可以關閉 IF 區塊。

%else

此關鍵字可以在不符合其它條件列示時，執行相關陳述式。

上下文

在這些上下文中可以找到：

- Net.Data 巨集宣告部份中

限制

可以包含這些元素：

- EXEC_SQL 陳述式
- DEFINE 陳述式
- FUNCTION 陳述式
- 函數呼叫
- HTML 區塊
- MESSAGE 區塊

範例

範例 1:

```
%IF ($(DTW_HTML_TABLE))
%define OUT_FORMAT = "HTML"
%ELSE
%define OUT_FORMAT = "CHARACTER"
%endif
```

MESSAGE 區塊

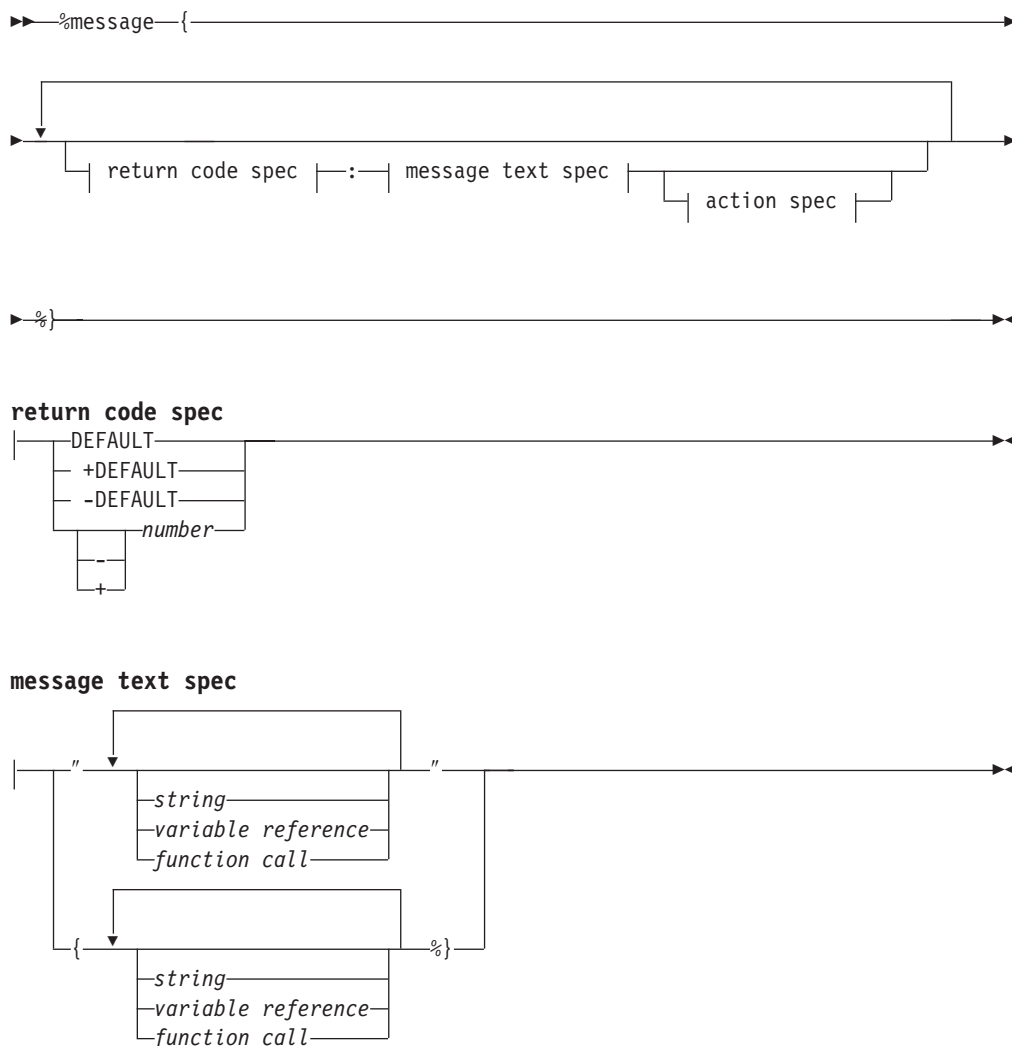
目的

MESSAGE 區塊根據函數回覆碼，指定顯示的訊息及採行的建議動作。

MESSAGE 區塊中定義一組回覆碼、對應訊息、以及建議動作。函數呼叫完成後，`Net.Data` 比較它的回覆碼與 MESSAGE 區塊中定義的回覆碼。如果函數的回覆碼符合 MESSAGE 區塊中定義的回覆碼，`Net.Data` 立即顯示訊息，並評估建議動作，判斷應該繼續處理或跳出 `Net.Data` 巨集。

MESSAGE 區塊可以是整體範圍，也可以是區域單一 FUNCTION 區塊。巨集最外層次定義的 MESSAGE 巨集，視為整體範圍。如果定義多個整體 MESSAGE 區塊，僅最後一個區塊有效。如果 MESSAGE 區塊定義在 FUNCTION 區域中，則該區域在所屬 FUNCTION 區塊的區域範圍中。有關回覆碼的處理規則，請參閱 *Net.Data* 程式設計手冊中的「Message 區塊」。

語法





參數

%message

此區塊關鍵字定義一組回覆碼、相關訊息、以及傳回函數呼叫時 `Net.Data` 執行的動作。

return_code spec:

正整數或負整數。Net.Data RETURN_CODE 變數值符合 *return_code spec* 值時，message 陳述式的剩餘資訊，可用來處理函數呼叫。您也可以為未特別在 \$MESSAGE 區塊中輸入的回覆碼，指定訊息。

DEFAULT

此關鍵字可用來指定預設訊息碼。 RETURN_CODE 不等於零 (0) 而且沒有指定完全相符時，此 message 陳述式中的資訊可用來處理函數呼叫。

+DEFAULT

此關鍵字可用來指定預設負訊息碼。RETURN_CODE 小於零 (0) 而且沒有指定完全相符時，此 message 陳述式中的資訊可用來處理函數呼叫。

-DEFAULT

此關鍵字可用來指定預設正訊息碼。 RETURN_CODE 大於零 (0)、沒有指定完全相符、而且未指定 +DEFAULT (RETURN_CODE 大於 0) 或 -DEFAULT (RETURN_CODE 小於 0) 值時，此 message 陳述式中的資訊可用來處理函數呼叫。

number

此訊息碼指定處理期間可能發生的錯誤及警告。0 到 9 的數值字串。

message text spec

RETURN_CODE 符合此訊息之 message 陳述式中的 *return_code* 值時，立即將此字串傳給 Web 瀏覽器。

string

任何順序的英數字元和標點符號，但不含定位字元、換行字元、或空白。

variable reference

變數參照，可傳回先前定義的變數值，指定時使用 `$` 及 `()`。例如：若 `VAR = 'abc'`，則 `$(VAR)` 傳回值 `'abc'`。有關語法的詳細資訊，請參閱第8頁的『變數參照』。

function call

呼叫先前定義的一或多個 FUNCTION 區塊，含指定引數。有關語法及範例，請參閱第21頁的『函數呼叫 (@)』。

action spec:

決定當 RETURN_CODE 符合此訊息之 message 陳述式中的 *return_code* 值時，Net.Data 採行的動作。

EXIT

此關鍵字指定當發生訊息碼對應的錯誤或警告時，立即跳出巨集。

CONTINUE

此關鍵字指定當發生訊息碼對應的錯誤或警告時，繼續處理。

上下文

在這些上下文中可以找到：

- FUNCTION 區塊
- Net.Data 巨集宣告部份所有區塊或陳述式之外

限制

可以包含這些元素：

- 函數呼叫
- 變數參照
- HTML 陳述式
- 字串

範例

範例 1:

```
%MESSAGE{
-601: {<H3>已建立此表。請返回並輸入您的名稱。</H3>
<P><a href="input">返回</a>
%}
default: "<H3>由於錯誤無法繼續執行，出現錯誤 $(RETURN_CODE)</H3>"
%}
```

範例 2:

```
%{ 整體訊息區塊 %}
%MESSAGE {
-100      : "回覆碼 -100 訊息"      : exit
  100      : "回覆碼 100 訊息"       : continue
+default : {
這個長訊息跨越多行。
您可以在此訊息中使用 HTML 標籤，
包含作者錨點和格式。%}      : continue
%}

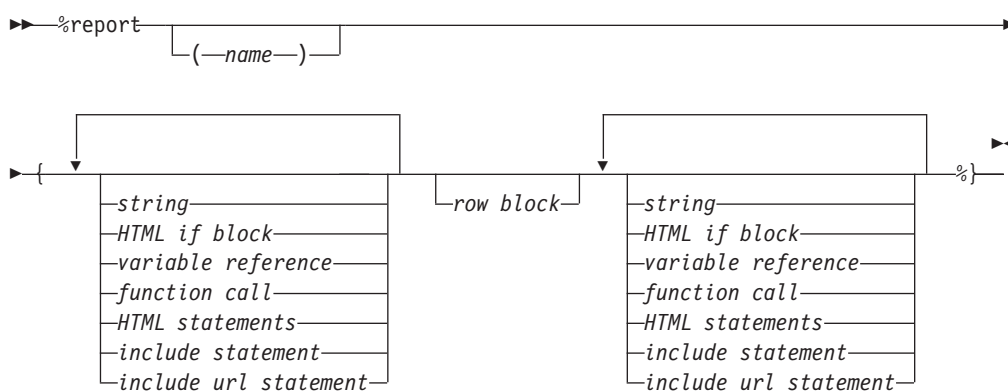
%{ FUNCTION 區塊中的區域訊息區塊 %}
%FUNCTION(DTW_REXX) my_function() {
%EXEC { my_command.cmd %}
%MESSAGE {
-100      : "回覆碼 -100 訊息"      : exit
  100      : "回覆碼 100 訊息"       : continue
-default : {
這個長訊息跨越多行。
您可以在此訊息中使用 HTML 標籤，
包含作者錨點和格式。%}      : exit
%}
%}
```

REPORT 區塊

目的

REPORT 區塊製作函數呼叫輸出的格式。表格名稱可以作為參數。如果指定表格名稱，則以指定表格中的資料來產生報表。否則，則以在函數參數列示中找到的第一個輸出表格來產生報表。如果參數列示中沒有表格，則使用預設表格資料。

語法



參數

%report

此關鍵字指定製作函數呼叫輸出指示的格式。您可以在報表中使用表頭及註腳資訊。

name

一個以英文字母或底線開頭的英文或數字字串，可以結合英文字母、數字、或底線字元。

string

任何順序的英數字元和標點符號，但不含定位字元、換行字元、或空白。

variable reference

變數參照，可傳回先前定義的變數值，指定時使用 \$ 及 ()。例如，若 VAR = 'abc'，則 \$(VAR) 傳回值 'abc'。有關語法的詳細資訊，請參閱第8頁的『變數參照』。

function call

呼叫先前定義的一或多個 FUNCTION 區塊，含指定引數。有關語法及範例，請參閱第21頁的『函數呼叫 (@)』。REPORT 區塊無法將 SQL 函數呼叫併入，除非是在 OS/400 平台上。

HTML if block

Net.Data 巨集 HTML 部份使用的 HTML IF 區塊。執行條件字串處理。比較時，把數值視為字串。有關語法及範例，請參閱第25頁的『HTML IF 陳述式』。

HTML statements

包含任何英數字元，以及為從屬站瀏覽器製作格式的 HTML 標籤。

include statement

INCLUDE 陳述式。讀取檔案，納入 Net.Data 巨集中。有關語法及範例，請參閱第28頁的『INCLUDE 陳述式』。

include_url statement

INCLUDE_URL 陳述式。讀取另一個檔案，納入指定此陳述式的 Net.Data 巨集中。指定的檔案，可能在區域或遠端伺服器中。有關語法及範例，請參閱第30頁的『INCLUDE_URL 陳述式』。

row block

ROW 區塊。函數呼叫傳回的資料各列的 HTML 格式化資料，只顯示一次。有關語法及範例，請參閱第41頁的『ROW 區塊』。

上下文

在這些上下文中可以找到：

- SQL 陳述式或區塊
- FUNCTION 陳述式或區塊

限制

可以包含這些元素：

- HTML IF 區塊
- INCLUDE 陳述式
- INCLUDE_URL 陳述式
- ROW 區塊
- 函數呼叫

例外：內部 SQL 函數無法呼叫 SQL 函數。

- HTML 陳述式
- 字串
- 變數參照

範例

範例 1：二欄式 HTML 表格中列出名稱及位置。選取表格中的一個名稱，呼叫 *name.mac* Net.Data 巨集的明細 HTML 區塊，將它的二個值當作 URL 的一部份來傳送。您可以在本例的 *name.mac* 中使用這些值來尋找名稱的其它明細。

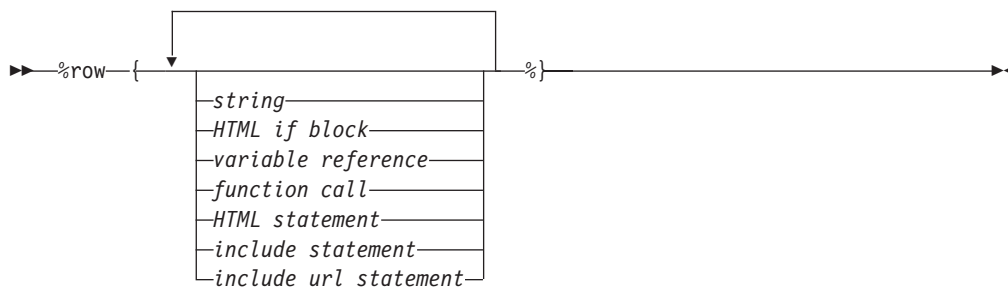
```
%REPORT{
<H2>查詢結果</H2>
<P>選取名稱明細。
<TABLE BORDER=1>
<TR><TD>名稱</TD><TD>位置</TD>
%ROW{
<TR>
<TD>
<a href="/cgi-bin/db2www/name.mac/details?name=$(V1)&location=$(V2)">$(V1)</a></TD>
<TD>$(V2)</TD>
%}
</TABLE>
%}
```

ROW 區塊

目的

ROW 區塊可用來處理函數呼叫傳回的每一個表格列。Net.Data 只為各列處理一個 ROW 區塊中的陳述式。

語法



參數

%row

此關鍵字指定函數呼叫傳回的資料各列的 HTML 格式化資料，只顯示一次。

string

任何順序的英數字元和標點符號，但不含定位字元、換行字元、或空白。

variable reference

變數參照，可傳回先前定義的變數值，指定時使用 \$ 及 ()。例如，若 VAR = 'abc'，則 \$(VAR) 傳回值 'abc'。有關語法的詳細資訊，請參閱第8頁的『變數參照』。

function call

呼叫先前定義的一或多個 FUNCTION 區塊，含指定引數。有關語法及範例，請參閱第21頁的『函數呼叫 (@)』。ROW 無法將 SQL 函數呼叫併入，除非是在 OS/400 平台上。

HTML if block

Net.Data 巨集 HTML 部份使用的 HTML IF 區塊。執行條件字串處理。比較時，把數值視為字串。有關語法及範例，請參閱第25頁的『HTML IF 陳述式』。

HTML statements

包含任何英數字元，以及為從屬站瀏覽器製作格式的 HTML 標籤。

include statement

INCLUDE 陳述式。讀取檔案，納入 Net.Data 巨集中。有關語法及範例，請參閱第28頁的『INCLUDE 陳述式』。

include_url statement

INCLUDE_URL 陳述式。讀取另一個檔案，納入指定此陳述式的 Net.Data 巨集中。指定的檔案，可能在區域或遠端伺服器中。有關語法及範例，請參閱第30頁的『INCLUDE_URL 陳述式』。

上下文

在這些上下文中可以找到：

- REPORT 區塊

限制

可以包含這些元素：

- HTML IF 區塊
- INCLUDE 陳述式
- INCLUDE_URL 陳述式
- 函數呼叫
- 變數參照
- HTML 陳述式
- 字串

範例

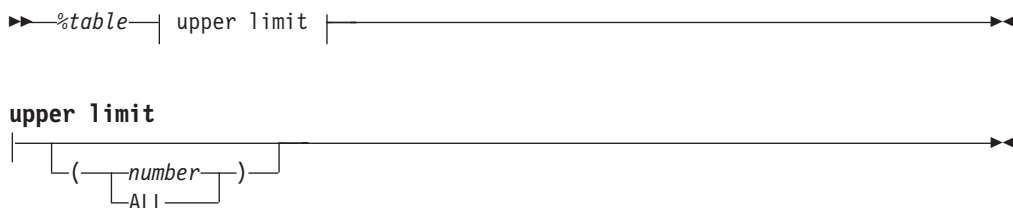
請參閱第40頁的『REPORT 區塊』中的 ROW 區塊範例。

TABLE 陳述式

目的

TABLE 陳述式可用來定義相關資料集合。它裡面有一個相同記錄或橫列的陣列、以及一個說明各列欄位的直欄名稱陣列。 `table` 陳述式只能用於 `define` 陳述式或區塊。

語法



參數

%table

此關鍵字指定相關資料的集合定義，裡面有一個相同記錄或橫列的陣列、以及一個介紹各列欄位的直欄名稱陣列。

upper limit

表格列數。

number

0 到 9 的數值字串。

ALL

此關鍵字容許表格列數不受限制。

上下文

在這些上下文中可以找到：

- `DEFINE` 陳述式

限制

可以包含這些元素：

- 號碼

範例

```
%DEFINE myTable=%TABLE(30)
```

第4章 變數

Net.Data 有四種類型的變數：預先定義、隱含、報表和 SQL。

- 第49頁的『預先定義的變數』是由 Net.Data 設定，無法變更。這些變數可以提供檔案位置和日期的相關資訊。
- 第55頁的『隱含的 Table 變數』是由 Net.Data 定義，可讓您從 SQL 查詢和函數呼叫存取資料。除非另外指定，否則它們只有在 REPORT 區塊中才能被認得。
- 第65頁的『報表變數』可幫助您從函數中自行設定報表。在參照這些變數之前，必須先定義它們。您可以在任何 Net.Data 巨集區塊中，設定或參照報表變數。
- 第73頁的『資料庫變數』可幫助您自行設定 FUNCTION 區塊的處理方式。在參照這些變數之前，必須先定義它們。您可以在任何 Net.Data 巨集區塊中，設定或參照資料庫變數。

有許多 Net.Data 變數的輸出，都是根據它所執行的平台而定。

此外，Net.Data 有下列幾種變數結構：

- 第45頁的『條件式變數』
- 第46頁的『隱藏變數』
- 第47頁的『List 變數』
- 第48頁的『Table 變數』

Net.Data 巨集中的常數長度最多可達 64K。因此，在巨集檔中不可以起始設定一個變數或設定一個預設值，而使其長度超過 64K。

條件式變數

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

條件式變數的值，是根據另一變數或字串的值，做條件式設定。它又稱為三元作業。

`test ? trueValue : falseValue`

test 要測試的條件。

trueValue

測試為真時所採用的值。

falseValue

測試為偽時所採用的值。

範例 1：如果 varB 存在，則 varA=value_1，否則 varA=value_2。

`varA = varB ? "value_1" : "value_2"`

範例 2：在這個案例當中，如果 value_1 為空值，則 varname 為空值，否則 varname 為 value_1。

`varname = ? "${value_1}"`

範例 3: 條件式和 LIST 變數一起使用時，其效果最佳。這個範例將告訴您，如何在 DEFINE 區塊中設置 WHERE 列示。 cust_inp 和 prod_inp 這兩個變數是從 CGI 傳來的 HTML 輸入變數，通常是從 HTML 表格頁面傳來。 where_list 變數是一個 LIST 變數，它是由兩個條件式陳述式所組成，每一個陳述式中都含有一個來自 CGI 的變數。

```
%DEFINE{
%list " AND " where_list
where_list    = ? "custid = $(cust_inp)"
where_list    = ? "product_name LIKE '$(prod_inp)%'"
where_clause  = ? "WHERE $(where_list)"
%}

%FUNCTION(DTW_SQL) mySelect() {
    SELECT * FROM prodtbale $(where_clause)
%}
```

如果 CGI 傳回 cust_inp 和 prod_inp 兩者所用的值，（例如，IBM 和 755C），則 where_clause 為：

```
WHERE custid = IBM AND product_name LIKE '755C'
```

如果 cust_inp 或 prod_inp 其中一個變數為空值，或者沒有定義，則 WHERE 子句會改而省略該空值。比方說，如果 prod_inp 是空值，則 WHERE 子句將如下所示：

```
WHERE custid = IBM
```

如果兩者皆為空值，或者兩者都沒有定義，則 where_clause 變數為空值，而且含有 \$(where_clause) 的 SQL 查詢中，也不會出現任何 WHERE 子句。

隱藏變數

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

當您在 HTML 來源中隱藏實際變數名稱時，隱藏變數可以讓您參照變數。要使用隱藏變數並不難，只要執行下列步驟即可：

1. 為每一個您要隱藏的字串，定義一個變數。
2. 在變數被參照的 HTML 區塊中，使用兩個貨幣符號（而不是一個）來參照變數。例如，採用 \$\$(X) 而不是 \$(X)。

範例 1:

```
%HTML(INPUT) {
<FORM ...>
<P>請選取要檢視的欄位：
<SELECT NAME="Field">
<OPTION VALUE="$(name)"> Name
<OPTION VALUE="$(addr)"> Address
.
.
.
</FORM>
%}

%DEFINE{
name="customer.name"
addr="customer.address"
%}

%FUNCTION(DTW_SQL) mySelect() {
```

```
SELECT $(Field) FROM customer
%}
.
.
.
```

當 HTML 表格頁面顯示在 Web 瀏覽器上時， \$\$\$(name) 和 \$\$\$\$(addr) 便會分別換成 \$(name) 和 \$(addr)，這樣一來，實際的表格和直欄名稱就不會在 HTML 表格頁面顯示出來，而您也無法察覺真正的變數名稱已被隱藏。當客戶提出該表格頁面時，會呼叫 HTML(REPORT) 區塊。當 @mySelect() 呼叫 FUNCTION 區塊時，SQL 陳述式中的 \$(Field)，到了 SQL 查詢中會換成 customer.name 或 customer.addr。

List 變數

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

List 變數可讓您建置一串以分隔符號區隔的值。在您希望以多個項目建構一個 SQL 查詢時，（例如，常見於一些 WHERE 或 HAVING 子句中），這類變數尤其有用。

一定要留空格。若要區分兩邊的值，通常都是用空格加以分隔。大部份的查詢都使用布耳運算子或算術運算子（例如，AND、OR 和 >）。請參閱第32頁的『LIST 陳述式』，以取得語法和其他相關資訊。

範例 1：這個範例是使用條件式、隱藏、和 list 變數：

```
%HTML(INPUT){
<FORM METHOD="POST" ACTION="/cgi-bin/db2www/example2.max/report">
請選出一或多個城市：<BR>
<INPUT TYPE="checkbox" NAME="conditions" VALUE="$$$(cond1)">Sao Paolo<BR>
<INPUT TYPE="checkbox" NAME="conditions" VALUE="$$$(cond2)">Seattle<BR>
<INPUT TYPE="checkbox" NAME="conditions" VALUE="$$$(cond3)">Shanghai<BR>
<INPUT TYPE="submit" VALUE="Submit Query">
</FORM>
%}

%DEFINE{
DATABASE="custcity"
%LIST " OR " conditions
cond1="Sao Paolo"
cond2="Seattle"
cond3="Shanghai"
whereClause=Conditions ? "WHERE $(conditions)" : ""
%}

%FUNCTION(DTW_SQL) mySelect(){
SELECT name, city FROM citylist
$(whereClause)
%}

%HTML(REPORT){
@mySelect()
%}
```

在 HTML 表格頁面中，如果沒有勾選任何方框，條件將為空值，因而查詢中的 whereClause 也是空值。否則，whereClause 將以 OR 分隔所選的值。比方說，如果三個城市都被選取，則 SQL 查詢將如下所示：

```
SELECT name, city FROM citylist
WHERE cond1='Sao Paolo' OR cond2='Seattle' OR cond3='Shanghai'
```

範例 2：在此例當中，隱含變數 `VLIST` 是使用 `" | "` 作為值的區隔符號。字值串是由用引號括住的值加以分隔。

```
%DEFINE %LIST " | " VLIST
%REPORT{
%ROW{
<EM>$(ROW_NUM):</EM> $(VLIST)
%}
%}
```

Table 變數

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

Table 變數中含有一個陣列的值以及相關的直欄名稱。您可以使用 `table` 變數，將幾組值傳給函數。表格的個別元素，可以在函數的 `REPORT` 區塊中被加以參照。Table 變數通常都作為 `SQL` 函數的輸出以及報表的輸入使用，但您也可以將它們當作 `IN`、`OUT` 或 `INOUT` 參數，傳給任何非 `SQL` 函數。但表格只能當作 `OUT` 參數，傳給 `SQL` 函數。請參閱 第44頁的『TABLE 陳述式』，以取得語法和其他相關資訊。

範例 1：HTML 報表區塊呼叫 `SQL` 查詢、將結果儲存在 `TABLE` 變數中、然後將該變數傳給 `REXX` 函數。

```
%DEFINE{
DATABASE = "iddata"
MyTable = %TABLE(ALL)
DTW_DEFAULT_REPORT = "no"
%}

%FUNCTION(DTW_SQL) Query(OUT table) {
select * from survey
%}

%FUNCTION(DTW_REXX) showTable(IN table) {
Say '列數：' table_ROWS
Say '欄數：' table_COLS
do j=1 to table_COLS
Say "下面是所有的直欄值" table_N.j "： "
do i = 1 to table_ROWS
Say "<B>"i"</B>： " table_V.i.j
end
end
end
%}

%HTML (report){
<HTML>
<PRE>
@Query(MyTable)
<p>
@showTable(MyTable)
</PRE>
</HTML>
%}
```

預先定義的變數

這些變數可以提供檔案位置和日期的相關資訊。您可能會覺得這些變數用在您所寫的函數當中，或者用在測試 `Net.Data` 巨集時，相當得心應手。但是，不可以用 `DTW_ASSIGN` 來修改這些變數。

- 第50頁的『`DTW_CURRENT_FILENAME`』
- 第51頁的『`DTW_CURRENT_LAST_MODIFIED`』
- 第52頁的『`DTW_MACRO_FILENAME`』
- 第53頁的『`DTW_MACRO_LAST_MODIFIED`』
- 第54頁的『`DTW_MP_PATH`』
- 第55頁的『`DTW_MP_VERSION`』

DTW_CURRENT_FILENAME

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

現行輸入檔的名稱和副檔名。此檔案可能是 Net.Data 巨集，也可能是來自 INCLUDE 陳述式的一個輸入檔。

範例

<P>這個檔案是 <I>\$(DTW_CURRENT_FILENAME)</I>，
它是在 \$(DTW_CURRENT_LAST_MODIFIED) 上更新。

DTW_CURRENT_LAST_MODIFIED

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

前次修改現行檔案的日期和時間。現行檔案可能是一個 Net.Data 巨集檔，也可能是 INCLUDE 陳述式中所指定的檔案。輸出格式是根據 Net.Data 執行的系統而定。

範例

<P>這個檔案是 <I>\$(DTW_CURRENT_FILENAME)</I>，
它是在 \$(DTW_CURRENT_LAST_MODIFIED) 上更新。

DTW_MACRO_FILENAME

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

現行 Net.Data 巨集的名稱 (包括副檔名在內)。

範例

<P>這個 Net.Data 巨集是 <I>\$(DTW_MACRO_FILENAME)</I>，
它是在 \$(DTW_MACRO_LAST_MODIFIED) 上更新。

DTW_MACRO_LAST_MODIFIED

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

前次修改 Net.Data 巨集的日期和時間。輸出格式是依 Net.Data 執行的系統而定。

範例

<P>這個 Net.Data 巨集是 <I>\$(DTW_MACRO_FILENAME)</I>，
它是在 \$(DTW_MACRO_LAST_MODIFIED) 上更新。

DTW_MP_PATH

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

Net.Data 可執行檔的路徑和名稱。輸出的內容將如下所示 (根據您的系統而定)：

usr/lpp/internet/server_root/cgi-bin/db2www

範例

Net.Data 可執行檔是 \$(DTW_MP_PATH)。

DTW_MP_VERSION

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

在該伺服器上執行之 Net.Data 的版本和版次號碼。輸出的格式如下：

Net.Data Version 1.0.3.

範例

這個 Web 應用程式是使用 \$(DTW_MP_VERSION)。

隱含的 Table 變數

這些變數是由 Net.Data 所定義，而且除非另外指定，否則只有在 REPORT 和 ROW 區塊中才能被認得。您可以使用這些變數，來參照查詢所傳回的值。

- 第56頁的『N_columnName』
- 第57頁的『Nn』
- 第58頁的『NLIST』
- 第59頁的『NUM_COLUMNS』
- 第60頁的『RETURN_CODE』
- 第61頁的『ROW_NUM』
- 第62頁的『TOTAL_ROWS』
- 第63頁的『V_columnName』
- 第64頁的『Vn』
- 第65頁的『VLIST』

N_columnName

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

指定直欄的名稱。它們可以用於 **REPORT** 和 **ROW** 區塊中。

範例

範例 1:

```
%REPORT{  
<P>將電子郵件傳到 <a href="mailto:$(N_email)">$(N_name)</a> 。  
.  
.  
.  
%}
```


Nn

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

含有對直欄 **n** 的函數呼叫或查詢所傳回的直欄名稱。它們可以用於 **REPORT** 和 **ROW** 區塊中。

範例

範例 1:

直欄 2 的名稱是 \$(N2)。

範例 2：這個範例可以告訴您，如何利用 **DTW_ASSIGN**，在 **REPORT** 區塊之外使用這個變數。其他相關資訊，請參閱 第111頁的『**DTW_ASSIGN**』。

```
...
%ROW{
  @DTW_ASSIGN(col1, N1)
%}
```

```
%HTML(report){
@
直欄 1 是 $(col1)。
%}
```

範例 3:

```
%REPORT{
<H2>產品目錄</H2>
<TABLE BORDER=1 CELLPADDING=3>
<TR><TD>$(N1)</TD><TD>$(N2)</TD><TD>$(N5)</TD>
%ROW{
<TR><TD>$(V1)</TD><TD>$(V2)</TD><TD>$(V3)</TD>
%}
</TABLE>
找到 $(ROW_NUM) 模式符合您的說明。
%}
```

NLIST

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

含所有來自函數呼叫或查詢結果的直欄名稱。預設的區隔符號是一個空值，雖然它將所有的直欄名稱一起執行，但您還是可以在 **DEFINE** 陳述式或區塊中，用 **list** 變數指定另一個區隔符號或者設定 **ALIGN="YES"**，將空格字元作為區隔符號使用。請參閱第 66 頁的『**ALIGN**』以取得其他相關資訊。

範例

範例 1：在直欄名稱的列示中，直欄名稱之間是以空格加以分隔（將 **ALIGN** 設為 **YES**）。

```
%DEFINE ALIGN="YES"
%REPORT{
您的查詢是在這些直欄上：$(NLIST) 。
.
.
.
%}
```

範例 2：這個範例是使用 **%LIST** 變數，將區隔符號改成 " | "。

```
%DEFINE %LIST " | " NLIST
%REPORT{
您的查詢是在這些直欄上：$(NLIST) 。
.
.
.
%}
```

NUM_COLUMNS

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

由函數呼叫或查詢所傳回的直欄數。

範例

範例 1:

```
%REPORT{  
您的查詢結果有 $(NUM_COLUMNS) 個直欄：$(NLIST)。  
...  
%}
```

RETURN_CODE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

由函數呼叫或查詢所傳回的回覆碼。 `Net.Data` 是使用此值來處理 `MESSAGE` 區塊。您可以使用這個變數，來判斷函數呼叫的結果是成功還是失敗。如果其值為零，表示函數呼叫已經順利完成。

您可以在這些相關上下文當中，找到 `RETURN_CODE` 變數：

- `%REPORT`
- `%ROW`
- `%MESSAGE`
- `%HTML`
- `%IF`

範例

範例 1：應用程式使用者看到了一則訊息，其大意是說該函數已經順利完成。

```
@function1()
%IF ("$(RETURN_CODE)" == "0")
    函數已經順利完成。
%ELSE
    函數失敗，其回覆碼為 $(RETURN_CODE)。
%ENDIF
```

範例 2：如果函數所傳回的回覆碼不是 0，則畫面會出現預設的訊息。

```
%MESSAGE{
default: "<h2>Net.Data 收到回覆碼：$(RETURN_CODE)</h2>" : continue
%}
```

ROW_NUM

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

ROW_NUM 只在 ROW 區塊中才有效，它是一個以定值增量的計數器，其定值增量作業將進行到最後一行處理完畢為止。比方說，如果表格中有 100 列，而您將 RPT_MAX_ROWS 設為 20，則 ROW_NUM 的終值為 20，因為它是所處理的最後一行。

範例

範例 1:

```
%REPORT{
<TABLE BORDER=1>
<TR><TD> Row Number </TD> <TD> Customer </TD>
%ROW{
<TR><TD> $(ROW_NUM) </TD> <TD> $(V_custname) </TD>
%}
</TABLE>
%}
```

TOTAL_ROWS

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
				X			

查詢所傳回的總列數 (無論 RPT_MAX_ROWS 的值為何)。比方說，如果您設定第72頁的『RPT_MAX_ROWS』的最大值為 20 列，但是查詢卻傳回 100 列，則在 ROW 處理之後，此變數將設為 100。您必須將 DTW_SET_TOTAL_ROWS 設為 YES，才能使用這個變數。請參閱 第71頁的『DTW_SET_TOTAL_ROWS』以取得其他相關資訊。

範例

```
%DEFINE DTW_SET_TOTAL_ROWS="YES"

%REPORT{
<H2>電子郵件目錄</H2>
<UL>
%ROW{
<LI>名稱：<a href="mailto:$(V1)">$(V2)</a><BR>
位置：$(V3)
%}
</UL>
所顯示的名稱：$(ROW_NUM)<BR>
所找到的名稱：$(TOTAL_ROWS)
%}
```

V_columnName

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

現行列之指定欄名的值，只在 **ROW** 區塊中才有效。未定義的欄名不會有變數存在。含有兩個同名直欄的查詢，會產生不可預期的結果。請考慮在您的 **SQL** 中使用一個 **AS** 子句，將重複的欄名加以更改。

範例

您已選了 \$(V_destcity)。

Vn

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

含有由函數呼叫或 SQL 查詢，為每一列所傳回的欄位值（由欄位 1 到 n ）。這個變數只在 ROW 區塊中才能被認得。如果要在區塊之外使用它，請使用第111頁的『DTW_ASSIGN』。

範例

範例 1：這個 REPORT 區塊可以顯示 HTML 表格。第二欄則顯示電子郵件位址。您可以按一下該鏈結，傳送一則訊息給某人。

```
%REPORT{
<H2>電子郵件目錄</H2>
<TABLE BORDER=1 CELLPADDING=3>
<TR><TD>名稱</TD><TD>電子郵件位址</TD><TD>位置</TD>
%ROW{
<TR><TD>$(V1)</TD>
<TD><a href="mailto:$(V2)">$(V2)</a></TD>
<TD>$(V3)</TD>
%}
</TABLE>
找到 $(ROW_NUM) 模式符合您的說明。
%}
```


VLIST

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

含目前在 ROW 區塊中所處理的現行列的所有欄位值，這些值只在 ROW 區塊中才有效。預設的區隔符號是一個空值，雖然它將所有的值一起執行，但您還是可以在 DEFINE 陳述式或區塊中，用 list 變數指定另一個區隔符號或者設定 ALIGN="YES"，將空格字元作為區隔符號使用。請參閱 第66頁的『ALIGN』以取得其他相關資訊。

由函數呼叫或查詢所傳回的表格中每一列的欄位值。雖然預設區隔符號是一個空格，但您可以在 DEFINE 區塊或是含有 list 變數的陳述式中，指定另一個區隔符號。

範例

範例 1:

```
%DEFINE ALIGN="YES"
```

```
%REPORT{  
下面是您的查詢結果：  
<OL>  
%ROW{  
<LI>$(VLIST)  
%}  
</OL>  
%}
```

範例 2：這個範例是使用 LIST 變數，將區隔符號改成<P>.

```
%DEFINE %LIST "<P>" VLIST
```

```
%REPORT{  
下面是您的查詢結果：  
%ROW{  
<HR>$(VLIST)  
%}  
%}
```

報表變數

這些變數可以幫助您自行設定報表。您必須先定義這些變數，才能使用它們。

- 第66頁的『ALIGN』
- 第67頁的『DTW_APPLET_ALTTEXT』
- 第68頁的『DTW_DEFAULT_REPORT』
- 第69頁的『DTW_HTML_TABLE』
- 第70頁的『DTW_PRINT_HEADER』
- 第71頁的『DTW_SET_TOTAL_ROWS』
- 第72頁的『RPT_MAX_ROWS』

ALIGN

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

這個變數負責控制報表變數 `NLIST` 和 `VLIST` 中，前頭和後頭的空格。其預設值是前頭和後頭都沒有空格，以方便將查詢結果內含在 `HTML` 錨或格式動作中。如果這個變數設為 `yes`，則報表變數的前後都會加上空格，這樣一來，在輸出報表時，才能做適當的調整。

或者，也可以在 `LIST` 陳述式中，為這些變數指定空格作為區隔符號，如範例 2 所示。

範例

範例 1：由於 `ALIGN` 設為 `YES`，因此列示中的每一欄，都由一個空格加以區隔。

```
%DEFINE ALIGN="YES"
<P>您的查詢是在這些直欄上：$(NLIST)
```

範例 2：此例的輸出與範例 1 相同，但是區隔符號卻在 `LIST` 陳述式中指出。請參閱第47頁的『List 變數』以取得其他相關資訊。

```
%DEFINE %LIST " " NLIST
%REPORT{
<P>您的查詢是在這些直欄上：$(NLIST)
%}
```

DTW_APPLET_ALTTEXT

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

將指定的 HTML，顯示給不認得 APPLET 標籤的瀏覽器。

範例

範例 1:

```
%DEFINE DTW_APPLET_ALTTEXT = "<P>對不起，您的瀏覽器不能用於 java。"
```

DTW_DEFAULT_REPORT

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

這個特殊變數，是用來置換 Net.Data 為沒有 REPORT 區塊之函數所產生的預設報表。您可以將 DTW_DEFAULT_REPORT 設為 "NO"，來置換此行為，也就是說，沒有 REPORT 區塊的函數，不會在瀏覽器出現任何結果。比方說，如果您在表格變數中收到函數呼叫的結果，而希望將這些結果傳給另一個函數加以處理。

範例

範例 1:

```
%DEFINE DTW_DEFAULT_REPORT="NO"
```

DTW_HTML_TABLE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

您可以使用這個變數，將結果顯示在 HTML 表格中，而不是將表格以文字類型的格式顯示 (亦即，使用 TABLE 標籤而不是 PRE 標籤)。YES 可以利用 HTML 表格標籤來顯示表格資料。預設動作是以文字格式顯示表格資料。

所產生的 TABLE 標籤中，包括邊框和資料格填補規格：

```
<TABLE BORDER CELLPADDING=2>
```

範例

範例 1：這個範例所顯示的是，沒有 REPORT 區塊的 SQL 函數。如果您將 DTW_HTML_TABLE 設為 YES，會產生 HTML 表格，而不是文字格式的表格。

```
%DEFINE DTW_HTML_TABLE="YES"

%FUNCTION(DTW_SQL){
SELECT NAME, ADDRESS FROM $(qTable)
%}
```

DTW_PRINT_HEADER

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

您可以使用這個變數，來指定檔案表頭的文字。如果這個變數設為 YES，或者沒有指定，則根據預設值，Net.Data 會印出 "Content-type: text/html" 作為檔案表頭文字。如果 DTW_PRINT_HEADER 變數設為 NO，則可印出 HTTP 表頭資訊。您必須先設定這個變數，才能讓 Net.Data 處理任何傳到瀏覽器的文字，因為 Net.Data 會在輸出文字之前，先讀取這個變數，之後便不會再察看它。一旦 Net.Data 將文字傳給瀏覽器之後，任何對 DTW_PRINT_HEADER 變數所做的變更，一律不予處理。

這個變數最常見的用法，是啟用 Net.Data 巨集來傳送 cookies。要設定 cookies，必須將 DTW_PRINT_HEADER 變數設為 NO，而前三行必須是 Content-type 表頭、Set-Cookie 陳述式、然後是一個空行。

範例

範例 1:

```
%DEFINE DTW_PRINT_HEADER="NO"

%HTML(cookie1) {
Content-type: text/html
Set-Cookie: UsrId=56, expires=Friday, 12-Dec-99, 12:00:00 GMT; path=/

<P>
任何文字
%}
```

DTW_SET_TOTAL_ROWS

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
				X			

您必須將 DTW_SET_TOTAL_ROWS 設為 YES，來使用第62頁的『TOTAL_ROWS』。如果沒有定義這個變數，則預設值不會設定 TOTAL_ROWS 變數。由於在決定總列數時，資料庫語言環境要求必須取回所有的列，因此將 DTW_SET_TOTAL_ROWS 設為 YES 之後，就會影響到執行效能。

範例

範例 1:

```
%DEFINE DTW_SET_TOTAL_ROWS="YES"
```

<P>傳回了 \$(ROW_NUM)。您的查詢限於 \$(TOTAL_ROWS) 列之內。

RPT_MAX_ROWS

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

指定報表所顯示的列數上限。其預設值是傳回表格內所有的列。如果您將此變數設為 0 或 ALL，也會顯示所有的列。

範例

第一個範例將告訴您，如何在 **DEFINE** 陳述式中定義此變數。第二個範例則告訴您，如何使用 **HTML** 輸入，以 **HTML** 表格頁面定義此變數。

範例 1：此例可將任何函數所傳回的列數限制為 20。

```
%DEFINE RPT_MAX_ROWS="20"
```

範例 2：這幾行可以置於 **FORM** 標籤中，讓應用程式使用者設定他們希望讓查詢所傳回的列數。

要傳回的列數上限 (0 表示沒有限制)：

```
<INPUT TYPE="text" NAME="RPT_MAX_ROWS" SIZE=3>
```


START_ROW_NUM

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X		X		X	X

這個變數可以設定列數，開始顯示查詢的結果。您可以將這個變數與第72頁的『RPT_MAX_ROWS』一起使用，將具有大型結果表格的查詢分成幾個小組，並使用「下一個」按鈕，來導覽結果表格。

透過「現場連線」存取時，START_ROW_NUM 只能用於資料庫，但 OS/400 平台則除外。

資料庫變數

您可以將這些變數與 SQL 函數一起使用，幫助您自行設定處理 FUNCTION 區塊的方法。在參照這些變數之前，必須先定義它們。您可以在任何 Net.Data 巨集區塊中，設定或參照報表變數。

- 第74頁的『DATABASE』
- 第75頁的『DB_CASE』
- 第76頁的『DB2PLAN』
- 第77頁的『DB2SSID』
- 第78頁的『DTW_SAVE_TABLE_IN』
- 第79頁的『LOCATION』
- 第80頁的『LOGIN』
- 第81頁的『PASSWORD』
- 第82頁的『SHOWSQL』
- 第83頁的『TRANSACTION_SCOPE』

DATABASE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X		X	X	X	X

Net.Data 是使用這個變數，建立與指定資料庫之間的連線。除非您使用「現場連線」來設定持續的連線，否則，在所有的平台上 (除了 OS/400 之外)，每個巨集只能有一個資料庫連線。如果沒有定義這個變數，則存取資料庫時，就會發生錯誤。

您可以在 OS/400 平台上，建立多個資料庫連線。此外，DATABASE 變數是屬於選用性變數。根據預設值，Net.Data for OS/400 會指定 DATABASE="*LOCAL"；DTW_SQL 語言環境是使用區域關聯式資料庫目錄登錄。

您可以在同一個 Net.Data 巨集中，針對不同的 HTML 區塊變更 DATABASE，如範例 2 和 3 中所示。

範例

範例 1：此 Net.Data 巨集中的任何 SQL，都是在 CELDIAL 資料庫上執行。

```
%DEFINE DATABASE="CELDIAL"
```

範例 2：不管 DATABASE 之前的值是什麼，HTML 區塊都會查詢資料庫 DB2D1。

```
%HTML(monthRpt){  
@DTW_ASSIGN(DATABASE, "DB2D1")  
%INCLUDE "rpthead.htm"  
@getRpt()  
%INCLUDE "rptfoot.htm"  
%}
```

範例 3：這個範例與範例 2 一樣，但它是使用 DEFINE 來設定 DATABASE 變數，而不是 DTW_ASSIGN。由於每次呼叫 HTML 區塊時，Net.Data 都是從頭處理完整的 Net.Data 巨集，因此這個作法行得通。

```
%DEFINE DATABASE="DB2D1"  
%HTML(monthRpt){  
%INCLUDE "rpthead.htm"  
@getRpt()  
%INCLUDE "rptfoot.htm"  
%}
```

DB_CASE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

指定 SQL 命令的大小寫。如果沒有定義這個變數，則預設動作就是不轉換 SQL 命令。您可以指定 "UPPER" 或 "LOWER"，強迫將 SQL 命令中所有的字元都設為大寫或小寫。

範例

範例 1:

```
%DEFINE DB_CASE="UPPER"
```

DB2PLAN

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
			X				

Net.Data for OS/390 是使用這個變數，來規劃與區域 DB2 子系統的連線。此變數會指定在 Net.Data 將存取的區域 DB2 子系統中，Net.Data SQL 語言環境的規劃名稱。如果這個變數沒有在 Net.Data for OS/390 架構檔中指定，也沒有在巨集中定義，這時候，巨集想要執行 SQL 函數時，就會發生錯誤。

範例

範例 1:

```
%DEFINE DB2PLAN="DTWGA105"
```

DB2SSID

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
			X				

Net.Data for OS/390 是使用這個變數，來建立與區域 DB2 子系統的連線。此變數會指定 Net.Data 所要存取的區域 DB2 子系統的子系統 ID。每一個巨集只能有一個區域資料庫連線。如果這個變數沒有在 Net.Data for OS/390 架構檔中指定，也沒有在巨集中定義，這時候，巨集想要執行 SQL 函數時，就會發生錯誤。

範例

範例 1:

```
%DEFINE DB2SSID="DB2G"
```

DTW_SAVE_TABLE_IN

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

這個變數可以識別 SQL 語言環境用來存放來自查詢之表格資料的 TABLE 變數。此表可以稍後再用，例如，用於分析表格資料的 REXX 程式。

範例

在這個範例當中，REXX FUNCTION 區塊會呼叫 REXX 程式 anzTbl.cmd，該程式是使用表格變數 theTable 來分析表格中的資料。變數 theTable 是從前一個 SQL 函數呼叫所傳回。

```
%DEFINE theTable = %TABLE(2)
%DEFINE DTW_SAVE_TABLE_IN = "theTable"

%FUNCTION(DTW_SQL) doQuery() {
SELECT MODNO, COST, DESCRIP FROM EQPTABLE
WHERE TYPE='MONITOR'
%}

%FUNCTION(DTW_REXX) analyze_table(myTable) {
  %EXEC{ anzTbl.cmd %}
%}

%HTML(doTable) {
@doQuery()
@analyze_table(theTable)
%}
```

LOCATION

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
			X				

Net.Data for OS/390 是使用這個變數，來建立與遠端資料庫伺服器的連線。此變數會指定區域 DB2 子系統據以識別該遠端伺服器的名稱。LOCATION 的值，必須在「通信資料庫」(CDB) 的 SYSIBM.SYSLOCATIONS 表格中加以定義。如果巨集中沒有定義這個變數，則該巨集所發出的任何 SQL 要求，都會在區域 DB2 子系統上執行。

範例

範例 1:

```
%DEFINE LOCATION="QMFDJ00"
```

LOGIN

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X		X	X	X	X

輸入使用者 ID 之後，即有權存取受保護的資料。您可以將這個變數與通行碼一起使用，以納入 DB2 的安全演繹法。其預設值是使用啓動 Web 伺服器的使用者 ID。您可以將此值編入 Net.Data 巨集中，或者讓應用程式使用者將它輸入 HTML 表格頁面中。

範例

範例 1：此例所示範的，是將存取權限定於使用者 ID DB2USER 一人。

```
%DEFINE LOGIN="DB2USER"
```

範例 2：此例所示範的這一行，可以併入 HTML 表格頁面中，讓應用程式使用者輸入其使用者 ID。

```
USERID&#58; <INPUT TYPE="text" NAME="LOGIN" SIZE=6>
```


PASSWORD

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X		X	X	X	X

輸入通行碼之後，即有權存取受保護的資料。您可以將這個變數與 LOGIN 一起使用，以納入 DB2 的安全演繹法。您可以將此值編入 Net.Data 巨集中，或者請人將它輸入 HTML 表格頁面中。

範例

範例 1：此例所示範的，是將存取權限於具有通行碼 NETDATA 的人。

```
%DEFINE PASSWORD="NETDATA"
```

範例 2：此例所示範的這一行，可以併入 HTML 表格頁面中，讓應用程式使用者輸入其通行碼。

```
PASSWORD&#58; <INPUT TYPE="password" NAME="PASSWORD" SIZE=8>
```

SHOWSQL

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

隱藏或顯示 Web 瀏覽器所用之查詢的 SQL。預設值是隱藏 SQL。如果將此變數設為 YES，可以顯示傳到資料庫的 SQL。設為 NO，則可隱藏 SQL。在測試時顯示 SQL，特別有助於 Net.Data 巨集的除錯作業。

範例

範例 1：此例將告訴您，每次顯示 SQL 的方法。

```
%DEFINE SHOWSQL="YES"
```

範例 2：此例將告訴您，如何指定是否要使用 HTML 表格頁面輸入來顯示 SQL。

```
SHOWSQL: <INPUT TYPE="radio" NAME="SHOWSQL" VALUE="YES"> Yes  
         <INPUT TYPE="radio" NAME="SHOWSQL" VALUE="" CHECKED> No
```

TRANSACTION_SCOPE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

可指定 SQL 命令的異動範圍。其預設值是 "MULTIPLE"，也就是說，Net.Data 只會在 HTML 區塊中所有的 SQL 命令都順利完成之後，才會發出 COMMIT。只要有一個 SQL 命令沒有順利完成，都會使所有先前對該區塊同一資料庫所執行的 SQL 全數回復。如果是指定 "SINGLE"，就表示只要 HTML 區塊中的每一個 SQL 命令順利完成，Net.Data 就會發出 COMMIT。

在 OS/400 和 OS/390 以外的平台上，如果下列條件全部成立的話，那麼對收到失敗回應之資料庫所做的更新，可能會被回復，而對同一 HTML 區塊中其他資料庫所做的更新，卻可能被確定：

- 指定了 TRANSACTION_SCOPE = "MULTIPLE"
- 在一個 HTML 區塊中存取多個資料庫 (如果您是使用「現場連線」時就有可能)
- SQL 要求傳回失敗回應

如果您是使用 DataJoiner 軟體，從 Net.Data 存取多個資料庫的話，那麼當您從 Net.Data 更新時，即可取得多個資料庫更新協調和一致性。

在 OS/400 和 OS/390 上，如果 TRANSACTION_SCOPE = "MULTIPLE"，則所有從單個 HTML 區塊發出的 IBM 資料庫更新，將一起確定或回復。

在 OS/400、REXX、Perl 和 Java 語言環境以外的平台上，則在它們自己獨立的作業系統處理作業當中執行。因此，不管 Net.Data TRANSACTION_SCOPE 值是什麼，只要是發自這些語言環境的資料庫更新，都將分別從發自某個 Net.Data 巨集檔的資料庫更新加以確定或回復。

範例

範例 1:

```
%DEFINE TRANSACTION_SCOPE="SINGLE"
```


第5章 Net.Data 內建函數

Net.Data 提供您在沒有使用 FUNCTION 區塊的狀況下，即可立即使用廣泛的函數。函數名稱並不區分大小寫。Net.Data 內建函數分為這些種類：

- **一般目的函數** 可協助您使用 Net.Data 來開發 Web 頁面，且不適用於其他的種類。請參閱 第86頁的『一般函數』。
- **數學函數** 執行算術上的運算。請參閱 第99頁的『數學函數』。
- **字串操作函數** 修改字串和字元。請參閱第110頁的『字串函數』。
- **字組操作函數** 變更字組。請參閱第126頁的『字組函數』。
- **表格操作函數** 協助您自行設定表格。請參閱第134頁的『表格函數』。
- **純本文檔案介面函數** 執行檔案輸入和輸出。請參閱第149頁的『純本文檔案介面』。
- **Web 登記函數** 執行在 Web 登記上的作業。請參閱第167頁的『Web 登記函數』。

許多函數具有一種或多種下列的格式：

- 以 DTW_、DTWF_、和 DTWR_ 作為開頭的函數，將它們的結果以一個輸出參數來傳回。您必須指定輸出參數。這個範例會顯現伺服器的時間：

```
@DTW_TIME(nowTime)
現行區域時間是 $(nowTime)。
```

- 以 DTW_r、DTWF_r、和 DTWR_r 作為開頭的函數，將它們的結果傳回至函數呼叫，所以它們並不會擁有輸出參數。這個範例會顯現伺服器時間，就如同上一個範例一般：

```
現行區域時間是 @DTW_rTIME()。
```

- 以 DTW_m 作為開頭的函數在多重參數上執行函數。每個參數都是輸入參數也都是輸出參數。結果會被傳回至個別的輸入參數。這個範例會將 3 個輸入參數轉換為大寫，以使其看起來可整齊劃一：

```
@DTW_mUPPERCASE(model, style, shipNo)
Shipment $(shipNo) contains $(quantity) of model $(model) $(style).
```

函數參數必須以正確的順序來排列。在指定最後的輸入參數之前，必須指定所有的輸入參數，或將其指定為空值 ("") 以接受預設值。例如，您可以這個方式來呼叫 DTW_TB_INPUT_TEXT：

```
@DTW_TB_INPUT_TEXT(myTable, "1", "2", "", "", "32")
```

第四個和第五個參數使用預設值。它們必須被併入空值以表示 "32" 在所產生的 HTML 中是 MAXLENGTH 的值。並未指定最終的參數，所以所使用的是預設值。如果您選取接受使用預設值來當作 MAXLENGTH 以及之前的兩個參數的話，就請將它們省略：

```
@DTW_TB_INPUT_TEXT(myTable, "1", "2")
```

當有後續的非空值輸入參數的時候，您必須為輸入參數在參數列示中指定中間空值。在您指定您最終的輸出參數之前，您不需要指定中間空值輸入參數。

以下的說明所描述的函數參數即是字串、整數、浮點數，和陣列（陣列代表表格）。所有 Net.Data 變數都是字串類型，但是整數、浮點數、或陣列這些項目是分別用來代表整數、浮點數或陣列值的字串。

一般函數

這些是您可以使用的一般目的函數：

- 第87頁的『DTW_ADDQUOTE』
- 第88頁的『DTW_DATE』
- 第89頁的『DTW_GETENV』
- 第90頁的『DTW_GETINIDATA』
- 第91頁的『DTW_HTMLENCODER』
- 第93頁的『DTW_QHTMLENCODER』
- 第95頁的『DTW_SETENV』
- 第96頁的『DTW_TIME』
- 第98頁的『DTW_URLESCSEQ』

DTW_ADDQUOTE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將在輸入字串中的單引號置換為兩個單引號。字串中是否包含單引號，對 SQL 陳述式可否順利的完成是必須的。

語法

- @DTW_ADDQUOTE(stringIn, stringOut)
- @DTW_rADDQUOTE(stringIn)
- @DTW_mADDQUOTE(stringMult, stringMult2, ..., stringMultn)

參數

表 1. DTW_ADDQUOTE 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。DTW_mADDQUOTE 可以有多個輸入字串。
字串	<i>stringOut</i>	OUT	一個來自 <i>stringIn</i> 修改後之表格頁面的變數。
字串	<i>stringMult</i>	INOUT	<ul style="list-style-type: none">• 在輸入上：包含字串的變數。• 在輸出上：包含輸入字串的一個變數，該字串的每一個單引號 (') 字元都被兩個單引號字元所置換。

範例

範例 1:

```
@DTW_ADDQUOTE(string1,string2)
```

- 輸入：string1 = "John's Web page"
- 傳回：string2 = "John''s Web page"

範例 2:

```
@DTW_rADDQUOTE("The article's title is 'Once upon a time'")
```

- 傳回：The article"s title is 'Once upon a time''

範例 3:

```
@DTW_mADDQUOTE(string1,string2)
```

- 輸入：string1="Joe's bag", string2=""to be or not to be""
- 傳回：string1="Joe''s bag", string2=""'to be or not to be'""

DTW_DATE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

在所指定的格式中傳回現行系統日期。

語法

- @DTW_DATE(format, stringOut)
- @DTW_DATE(stringOut)
- @DTW_rDATE(format)
- @DTW_rDate()

參數

表 2. DTW_DATE 參數

資料類型	參數	使用	說明
字串	<i>format</i>	IN	一個指定資料格式的變數或文字字串。有效格式包括： <ul style="list-style-type: none">• D--一年中的第幾天 (001-366)• E--歐式日期格式 (dd/mm/yy)• N--正常日期格式 (dd mon yyyy)• O--照順序的日期格式 (yy/mm/dd)• S--標準日期格式 (yyyymmdd)• U--美式日期格式 (mm/dd/yy) 預設值是 N。
字串	<i>stringOut</i>	OUT	一個包含具有所指定格式之日期的變數。

範例

範例 1:

```
@DTW_DATE(results)
```

- 傳回：results = "25 04 1997"

範例 2:

```
@DTW_DATE("E", results)
```

- 傳回：results="25/04/97"

範例 3:

```
%HTML(report){  
<P>這個報表於 @DTW_rDATE("U") 建立的。
```

- 傳回：04/25/97

DTW_GETENV

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將所指定之環境變數的值傳回。您也可以使用 `ENVVAR` 來取得環境變數的值。有關詳細資訊，請參閱第13頁的『`ENVVAR` 陳述式』。

語法

- `@DTW_GETENV(envVarName, envVarValue)`
- `@DTW_rGETENV(envVarName)`

參數

表 3. `DTW_GETENV` 參數

資料類型	參數	使用	說明
字串	<i>envVarName</i>	IN	一個變數或文字字串。
字串	<i>envVarValue</i>	OUT	在 <i>envVarName</i> 上所指定的環境變數值。若該值找不到的話將會傳回一個空字串。

範例

範例 1:

```
@DTW_GETENV(myEnvVarName, myEnvVarValue)
```

- 輸入：`myEnvVarName = "PATH"`
- 傳回：`myEnvVarValue = "/usr/path"`

範例 2:

```
@DTW_rGETENV(myPath)
```

- 輸入：`myPath = "PATH"`
- 傳回：`"/usr/path"`

範例 3:

```
伺服器是 @DTW_rGETENV("SERVER_NAME")。
```

- 傳回：`"www.software.ibm.com"`

DTW_GETINIDATA

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回所指定之架構變數的值。若找不到值的話將會傳回一個空字串。

註：特殊的架構路徑變數，MACRO_PATH、EXEC_PATH、和 INCLUDE_PATH，以及 ENVIRONMENT 陳述式無法由這個呼叫取回。

語法

- @DTW_GETINIDATA(iniVarName, iniVarValue)
- @DTW_rGETINIDATA(iniVarName)

參數

表 4. DTW_GETINIDATA 參數

資料類型	參數	使用	說明
字串	<i>iniVarName</i>	IN	一個變數或文字字串。
字串	<i>iniVarValue</i>	OUT	在 <i>iniVarName</i> 中所指定的架構變數值。

範例

範例 1:

```
@DTW_GETINIDATA(myEnvVarName, myEnvVarValue)
```

- 輸入：myEnvVarName = "FFI_PATH"
- 傳回：myEnvVarValue = "D:\FFI"

範例 2:

```
@DTW_rGETINIDATA("HTTP_COOKIE")
```

- 傳回："cookie.txt"

DTW_HTMLENCODE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

編碼字元將標準 HTML 十進位跳出碼 (escape code) 使用在大部份的字元上。您可以使用這個函數來對您不想要瀏覽器解譯為 HTML 的資料來進行編碼。例如，藉由使用適當的跳出字元 (escape character)，您可以顯示小於 (<) 及大於 (>) 記號，而它們通常是由 HTML 標籤所保留。

在第二個範例中，字串

"1 2 3" 在 HTML 中於每個數字之間只顯現一個空白。使用 DTW_HTMLENCODE 來確定所顯現的空白數目是否正確。

下列的字元是由 DTW_HTMLENCODE 函數所編碼：

表 5. HTML 十進位跳出字元

字元	名稱	字碼
SPACE	空白	
"	雙引號	"
#	號碼記號	#
%	百分比	%
&	& 記號	&
/	反斜線	\
:	冒號	:
;	分號	;
<	小於	<
=	等於	=
>	大於	>
?	問號	?
@	At 記號	@
[左方括弧	(
\	斜線	/
]	右方括弧)
^	Carat	^
{	左大括弧	{
	直線	|
}	右大括弧	}
~	波型符號	~

語法

- @DTW_HTMLENCODE(stringIn, stringOut)
- @DTW_rHTMLENCODE(stringIn)

參數

表 6. *DTW_HTML_ENCODE* 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
字串	<i>stringOut</i>	OUT	一個包含已修改之輸入字串的變數，在該輸入字串中有某些特定的字元已被 HTML 跳出字元編碼所置換。

範例

範例 1:

@DTW_HTML_ENCODE(string1,string2)

- 輸入：string1 = "Jim's dog"
- 傳回：string2 = "Jim's dog"

範例 2:

@DTW_rHTML_ENCODE("X <= 10")

- 傳回："X <= 10"

DTW_QHTMLENCODE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

執行如同 @DTW_HTMLENCODE 的函數，但亦將單引號字元 (') 編碼為 '。DTW_QHTMLENCODE 所使用的 HTML 十進位跳出字元將顯現在第91頁的表 5。

請考慮對所有的 SQL INPUT 陳述式使用這個函數。例如，如果您如同下列的範例一般，將 O'Brien 輸入為姓，則單引號將會造成一個錯誤：

```
INSERT INTO USER1.CUSTABLE (LNAME, FNAME)
VALUES ('O'Brien', 'Patrick')
```

使用 DTW_QHTMLENCODE 函數來變更 SQL 陳述式以防止錯誤發生：

```
INSERT INTO USER1.CUSTABLE (LNAME, FNAME)
VALUES ('O&#39;Brien', 'Patrick')
```

相關明細請參閱範例 3。

語法

- @DTW_QHTMLENCODE(stringIn, stringOut)
- @DTW_rQHTMLENCODE(stringIn)

參數

表 7. DTW_QHTMLENCODE 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
字串	<i>stringOut</i>	OUT	一個包含 <i>stringIn</i> 修改格式的變數，而在 <i>stringIn</i> 中有某些字元已被 HTML 跳出字元編碼所置換。

範例

範例 1:

```
@DTW_QHTMLENCODE(string1,string2)
```

- 輸入：string1 = "Jim's dog"
- 傳回：string2 = "Jim's dog"

範例 2:

```
@DTW_rQHTMLENCODE("John's & Jane's")
```

- 傳回："John's & Jane's"

範例 3:

```
%FUNCTION(DTW_SQL) insertName(){  
INSERT INTO USER2A.SURVEY  
(NAME) VALUES '@DTW_rQHTMLENCODE(fullname)'  
%}
```

- 輸入：fullname = "Patrick O'Brien"
- 傳回： 'Patrick 0'Brien'

DTW_SETENV

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

設定一個具有指定值之環境變數，並傳回上一個值。若找不到上一個值的話將會傳回一個空字串。

語法

- @DTW_SETENV(envVarName, envVarValue, prevValue)
- @DTW_rSETENV(envVarName, envVarValue)

參數

表 8. DTW_SETENV 參數

資料類型	參數	使用	說明
字串	<i>envVarName</i>	IN	一個代表環境變數的變數或文字字串。
字串	<i>envVarValue</i>	OUT	具有一環境變數已被設定之值的變數或文字字串。
字串	<i>prevValue</i>	OUT	一個包含環境變數之上一個值的變數。 DTW_rSETENV 如同函數傳回值一般的將值傳回。

範例

範例 1:

```
@DTW_SETENV("PATH", "myPath", prevValue)
```

- 輸入：myPath = "myPath"
- 傳回：prevValue = "myPreviousPath"

範例 2:

```
@DTW_rSETENV("PATH", "myPath")
```

- 輸入：myPath = "myPath"
- 傳回："myPreviousPath", PATH = "myPath"

DTW_TIME

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

以所指定的格式將現行系統時間傳回。

語法

- @DTW_TIME(stringIn, stringOut)
- @DTW_TIME(stringOut)
- @DTW_rTIME(stringIn)
- @DTW_rTIME()

參數

表 9. DTW_TIME 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	指定時間格式的變數或文字字串。有效的格式為： <ul style="list-style-type: none">• C 常用時間 (hh:mmAM/PM 以 12 小時時間顯示)• L 地區時間 (hh:mm:ss)• N 正常時間 (hh:mm:ss 以 24 小時時間顯示)• H 從午夜 12 點起算的小時數• M 從午夜 12 點起算的分鐘數• S 從午夜 12 點起算的秒數 預設值是 N。
字串	<i>stringOut</i>	OUT	一個包含以所指定格式顯示之時間的變數。

範例

範例 1:

@DTW_TIME(results)

- 傳回：results = "10:30:53"

範例 2:

@DTW_TIME("C", results)

- 傳回：results = "10:30AM"

範例 3:

@DTW_rTIME("M")

- 傳回："630"

範例 4:


```
%REPORT{  
<P>這個報表於 @DTW_rTIME(), @DTW_rDATE() 建立的。  
%}
```

- 傳回：這個報表於 15:04:39, 01 May 1997 建立的。

DTW_URLESCSEQ

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

在 URL 中不允許以它們的跳出值來置換字元。您必須使用這個函數來將任何列示在第 98 頁的表 10 中的字元傳送至其他的巨集檔案或 HTML 區塊。

表 10. 在 URL 不允許的字元

字元	名稱	字碼
SPACE	空白	
"	雙引號	
#	號碼記號	
%	百分比	
&	& 記號	
/	反斜線	F
:	冒號	A
;	分號	B
<	小於	C
=	等於	D
>	大於	E
?	問號	F
@	At 記號	(
[左方括弧	B
\	斜線	C
]	右方括弧	D
^	Carat	E
{	左大括弧	B
	直線	C
}	右大括弧	D
~	波型符號	E

語法

- @DTW_URLESCSEQ(stringIn, stringOut)
- @DTW_rURLESCSEQ(stringIn)

參數

表 11. DTW_URLESCSEQ 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。

表 11. DTW_URLESCSEQ 參數 (繼續)

資料類型	參數	使用	說明
字串	<i>stringOut</i>	OUT	一個包含輸入字串的變數，該輸入字串具有在 URL 中不允許由它們的十六進位跳出值來置換的字元。

範例

範例 1:

```
@DTW_URLESCSEQ(string1,string2)
```

- 輸入：string1 = "Guys & Dolls"
- 傳回：string2 = "Guys%20%26%20Dolls"

範例 2:

```
@DTW_rURLESCSEQ("Guys & Dolls")
```

- 傳回："Guys%20%26%20Dolls"

範例 3：這個範例使用在 ROW 區塊中的 DTW_rURLESCSEQ。當應用程式使用者在名稱上按一下時，名稱和電子郵件位址會被傳送至 Net.Data 巨集 fullrpt.mac 的輸入區塊，而該 fullrpt.mac 具有 name 以及 email。

```
%ROW{
<P><a href="fullrpt.mac/input?name=@DTW_rURLESCSEQ(V1)&email=@DTW_rURLESCSEQ(V2)">
$(V1)</a>
%}
```

- 輸入：V1="Patrick O'Brien", V2="obrien@ibm.com"
- 傳回：

```
<P><a href="fullrpt.mac/input?name=Patrick%20'O'Brien&email="obrien%40ibm.com">
Patrick O'Brien</a>
```

數學函數

這些函數可讓您作大部分的算數計算。

- 第100頁的『DTW_ADD』
- 第101頁的『DTW_DIVIDE』
- 第102頁的『DTW_DIVREM』
- 第103頁的『DTW_FORMAT』
- 第106頁的『DTW_INTDIV』
- 第107頁的『DTW_MULTIPLY』
- 第108頁的『DTW_POWER』
- 第109頁的『DTW_SUBTRACT』

DTW_ADD

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

新增兩個參數的值。

語法

- @DTW_ADD(number1, number2, precision, result)
- @DTW_rADD(number1, number2, precision)

參數

表 12. DTW_ADD 參數

資料類型	參數	使用	說明
浮點數	<i>number1</i>	IN	一個代表某個數字的變數或文字字串。
浮點數	<i>number2</i>	IN	一個代表某個數字的變數或文字字串。
整數	<i>precision</i>	IN	一個代表指定結果之精確度之正整數的變數或文字字串。預設值是 9。
浮點數	<i>result</i>	OUT	包含 <i>number1</i> 和 <i>number2</i> 總合的變數。

範例

範例 1:

```
@DTW_ADD(NUMB1, NUMB2, "2", result)
```

- 輸入：NUMB1 = "105", NUMB2 = "3"
- 傳回：result = "1.1E+2"

範例 2:

```
@DTW_rADD("12", NUMB2, "5")
```

- 輸入：NUMB2 = "7.00"
- 傳回："19.00"

DTW_DIVIDE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將第一個參數的值除以第二個參數的值。

語法

- @DTW_DIVIDE(number1, number2, precision, result)
- @DTW_rDIVIDE(number1, number2, precision)

參數

表 13. DTW_DIVIDE 參數

資料類型	參數	使用	說明
浮點數	<i>number1</i>	IN	一個代表某個數字的變數或文字字串。
浮點數	<i>number2</i>	IN	一個代表某個數字的變數或文字字串。
整數	<i>precision</i>	IN	一個代表指定結果之精確度之正整數的變數或文字字串。預設值是 9。
浮點數	<i>result</i>	OUT	包含將 <i>number1</i> 除以 <i>number2</i> 之結果的變數。

範例

範例 1:

```
@DTW_DIVIDE("8.0", NUMB2, result)
```

- 輸入：NUMB2 = "2"
- 傳回：result = "4"

範例 2:

```
@DTW_rDIVIDE("1", NUMB2, "5")
```

- 輸入："1", NUMB2 = "3"
- 傳回："0.33333"

範例 3:

```
@DTW_rDIVIDE(NUMB1, "2", "5")
```

- 輸入：NUMB1 = "5"
- 傳回："2.5"

DTW_DIVREM

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將第一個參數除以第二個參數並傳回餘數。如果餘數的非零，則其正負號與第一個參數的正負號是相同的。如果結果不能以一完整的數字來表示，則運算將會錯誤或失敗。

語法

- @DTW_DIVREM(number1, number2, precision, result)
- @DTW_rDIVREM(number1, number2, precision)

參數

表 14. DTW_DIVREM 參數

資料類型	參數	使用	說明
浮點數	<i>number1</i>	IN	一個代表某個數字的變數或文字字串。
浮點數	<i>number2</i>	IN	一個代表某個數字的變數或文字字串。
整數	<i>precision</i>	IN	一個代表指定結果之精確度之正完整數字的變數或文字字串。預設值是 9。
浮點數	<i>result</i>	OUT	包含將 <i>number1</i> 除以 <i>number2</i> 之餘數的變數。

範例

範例 1:

@DTW_DIVREM(NUMB1, NUMB2, result)

- 輸入：NUMB1 = "2.1", NUMB2 = "3"
- 傳回：result = "2.1"

範例 2:

@DTW_rDIVREM("10", NUMB2)

- 輸入：NUMB2 = "0.3"
- 傳回："0.1"

範例 3:

@DTW_rDIVREM("3.6", "1.3")

- 傳回："1.0"

範例 4:

@DTW_rDIVREM("-10", "3")

- 傳回："-1"

DTW_FORMAT

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

自行設定數字的格式。如果只有指定數字參數，則結果將會以如同已執行 `@DTW_rADD(number,"0")` 的狀況來進行格式化。如果指定了任何其他的選項，則會依據下列的規則來對數字進行格式化：

- 前面和後面的選項將說明有多少字元會分別讓結果的十進位部分與整數部分來使用。如果您省略的這兩項或其中一項，則使用作為該部分的字元的數目將會等於所需要的數量。
- 如果前面的並不足以大到包含數字的整數部分（加上負數之前的正負號），則將是個錯誤的結果。如果前面的大於該部分之所需，則在數字的左邊將是空白。如果後面的大小與數字之十進位的大小不相同，則數字會以整數（或以零擴充）填入。指定 0 會導致數字被四捨五入為整數。
- 此外，`expp` 和 `expt` 會控制結果的指數部分。`expp` 設定數字之指數部分的位置，預設值是設定為如同所需的大小（其也可能是零）。`expt` 設定觸發點以供指數表示法使用。預設值是 `precision` 參數的預設值。
- 如果 `expp` 是 0，則將不支援指數，且數字將以簡單格式表示，如有必要的話會再加上零。如果 `expp` 不夠大以致於無法包含指數，則將會有錯誤的結果。
- 如果整數或十進位部分之數字的位置所需分別超過 `expt` 或是 `expt` 的兩倍，則將會使用指數表示法。如果 `expt` 是 0，則一定會使用指數的表示法，除非指數是 0。（如果 `expp` 是 0，將會以 `expt` 來置換 0 值。）當指數是 0 而 `expp` 被指定為非零的時候，`expp+2` 空白會作為結果之指數部分的支援。如果指數是 0，但並未指定 `expp` 時，將會使用簡單格式。

語法

- `@DTW_FORMAT(number, before, after, expp, expt, precision, result)`
- `@DTW_rFORMAT(number, before, after, expp, expt, precision)`

參數

表 15. *DTW_FORMAT* 參數

資料類型	參數	使用	說明
浮點數	<i>number</i>	IN	一個代表某個數字的變數或文字字串。
整數	<i>before</i>	IN	一個代表正完整數字的變數或文字字串。這是一個可選用的參數。您必須輸入一個空字串 ("")，以擁有額外的參數。
整數	<i>after</i>	IN	一個代表正完整數字的變數或文字字串。這是一個可選用的參數。您必須輸入一個空字串 ("")，以擁有額外的參數。
整數	<i>expp</i>	IN	一個代表正完整數字的變數或文字字串。您必須輸入一個空字串 ("")，以擁有額外的參數。

表 15. DTW_FORMAT 參數 (繼續)

資料類型	參數	使用	說明
整數	<i>expt</i>	IN	一個代表正完整數字的變數或文字字串。您必須輸入一個空字串 ("")，以擁有額外的參數。
整數	<i>precision</i>	IN	一個代表指定結果之精確度之正完整數字的變數或文字字串。預設值是 9。
浮點數	<i>result</i>	OUT	一個含所指定之整數化和格式化之數字的值。

範例

範例 1:

@DTW_FORMAT(NUMB, BEFORE, result)

- 輸入：NUMB = "3", BEFORE = "4"
- 傳回：result= " 3"

範例 2:

@DTW_FORMAT("1.73", "4", "0", result)

- 傳回：result = " 2"

範例 3:

@DTW_FORMAT("1.73", "4", "3", result)

- 傳回：result = " 1.730"

範例 4:

@DTW_FORMAT(" - 12.73", "", "4", result)

- 傳回：result = "-12.7300"

範例 5:

@DTW_FORMAT("12345.73", "", "", "2", "2", result)

- 傳回：result = "1.234573E+04"

範例 6:

@DTW_FORMAT("1.234573", "", "3", "", "0", result)

- 傳回：result = "1.235"

範例 7:

@DTW_rFORMAT(" - 12.73")

- 傳回：" - 12.73"

範例 8:

@DTW_rFORMAT("0.000")

- 傳回："0"

範例 9:

@DTW_rFORMAT("12345.73", "", "", "3", "6")

- 傳回："12345.73"

範例 10:


```
@DTW_rFORMAT("1234567e5", "", "3", "0")
```

- 傳回："123456700000.000"

範例 11:

```
@DTW_rFORMAT("12345.73", "", "3", "", "0")
```

- 傳回："1.235E+4"

DTW_INTDIV

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將第一個參數除以第二個參數，並傳回結果的整數部分。

語法

- @DTW_INTDIV(number1, number2, precision, result)
- @DTW_rINTDIV(number1, number2, precision)

參數

表 16. DTW_INTDIV 參數

資料類型	參數	使用	說明
浮點數	<i>number1</i>	IN	一個代表某個數字的變數或文字字串。
浮點數	<i>number2</i>	IN	一個代表某個數字的變數或文字字串。
整數	<i>precision</i>	IN	一個代表指定結果之精確度之正完整數字的變數或文字字串。預設值是 9。
浮點數	<i>result</i>	OUT	一個包含將 <i>number1</i> 除以 <i>number2</i> 所得之商之整數部分的變數。

範例

範例 1:

```
@DTW_INTDIV(NUMB1, NUMB2, result)
```

- 輸入：NUMB1 = "10", NUMB2 = "3"
- 傳回：result = "3"

範例 2:

```
@DTW_rINTDIV("2", NUMB2)
```

- 輸入：NUMB2 = "3"
- 傳回："0"

DTW_MULTIPLY

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將兩個參數相乘。

語法

- @DTW_MULTIPLY(number1, number2, precision, result)
- @DTW_rMULTIPLY(number1, number2, precision)

參數

表 17. DTW_MULTIPLY 參數

資料類型	參數	使用	說明
浮點數	<i>number1</i>	IN	一個代表某個數字的變數或文字字串。
浮點數	<i>number2</i>	IN	一個代表某個數字的變數或文字字串。
整數	<i>precision</i>	IN	一個代表指定結果之精確度之正完整數字的變數或文字字串。預設值是 9。
浮點數	<i>result</i>	OUT	一個包含 <i>number1</i> 和 <i>number2</i> 乘積的變數。

範例

範例 1:

```
@DTW_MULTIPLY(NUM1, NUMB2, result)
```

- 輸入：NUMB1 = "4", NUMB2 = "5"
- 傳回：result = "20"

範例 2:

```
@DTW_rMULTIPLY("0.9", NUMB2)
```

- 輸入：NUMB2 = "0.8"
- 傳回："0.72"

DTW_POWER

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將第二個參數作為第一個參數的乘幂。

語法

- @DTW_POWER(number1, number2, precision, result)
- @DTW_rPOWER(number1, number2, precision)

參數

表 18. DTW_POWER 參數

資料類型	參數	使用	說明
浮點數	<i>number1</i>	IN	一個代表某個數字的變數或文字字串。
浮點數	<i>number2</i>	IN	一個代表某個數字的變數或文字字串。
整數	<i>precision</i>	IN	一個代表指定結果之精確度之正完整數字的變數或文字字串。預設值是 9。
浮點數	<i>result</i>	OUT	一個包含number1的number2次方。

範例

範例 1:

@DTW_POWER(NUMB1, NUMB2, result)

- 輸入：NUMB1 = "2", NUMB2 = "-3"
- 傳回：result = "0.125"

範例 2:

@DTW_rPOWER("1.7", NUMB2, precision)

- 輸入：NUMB2 = "8", precision = "5"
- 傳回："69.758"

DTW_SUBTRACT

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將第二個參數的值從第一個參數的值中減去。

語法

- @DTW_SUBTRACT(number1, number2, precision, result)
- @DTW_rSUBTRACT(number1, number2, precision)

參數

表 19. DTW_SUBTRACT 參數

資料類型	參數	使用	說明
浮點數	<i>number1</i>	IN	一個代表某個數字的變數或文字字串。
浮點數	<i>number2</i>	IN	一個代表某個數字的變數或文字字串。
整數	<i>precision</i>	IN	一個代表指定結果之精確度之正完整數字的變數或文字字串。預設值是 9。
浮點數	<i>result</i>	OUT	一個包含 <i>number1</i> 和 <i>number2</i> 之差的變數。

範例

範例 1：這個範例顯現一個比較數值的方式，其為在 Net.Data 中的字串。

```
@DTW_SUBTRACT(NUM1, NUMB2, comp)
%IF(comp > "0")
<P>$(NUM1) 大於 $(NUM2) 。
%ENDIF
```

- 輸入：NUMB2 = "2.07"
- 傳回："-0.77"

範例 2：

```
@DTW_SUBTRACT(NUMB1, NUMB2, result)
```

- 輸入：NUMB1 = "1.3, NUMB2 = "1.07"
- 傳回：result = "0.23"

範例 3：

```
@DTW_rSUBTRACT("1.3", NUMB2)
```

- 輸入：NUMB2 = "2.07"
- 傳回："-0.77"

字串函數

下列是由 Net.Data 所支援的標準字串函數組：

- 第111頁的『DTW_ASSIGN』
- 第112頁的『DTW_CONCAT』
- 第113頁的『DTW_DELSTR』
- 第114頁的『DTW_INSERT』
- 第116頁的『DTW_LASTPOS』
- 第117頁的『DTW_LENGTH』
- 第118頁的『DTW_LOWERCASE』
- 第119頁的『DTW_POS』
- 第120頁的『DTW_REVERSE』
- 第121頁的『DTW_STRIP』
- 第122頁的『DTW_SUBSTR』
- 第123頁的『DTW_TRANSLATE』
- 第125頁的『DTW_UPPERCASE』

警告: 除了在 OS/400 的平台之外，在其他任何的平台上，Net.Data 均不支援多重位元組字串操作。

DTW_ASSIGN

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

請指定一個輸入變數給輸出變數。因為 $\$(Vn)$ ，其中 n 是一個數字，其在 **REPORT** 區塊之外將無法辨識，但若您想要參考位於 **ROW** 區塊之外的值的話，您可以使用這個函數將值指定到一個不同的變數。

您亦可以使用這個函數在巨集中變更變數。例如，您可以變更 **HTML** 區塊的 **DATABASE**。（請參閱在 第74頁的『**DATABASE**』中的範例。）

語法

- @DTW_ASSIGN(stringOut, stringIn)

參數

表 20. DTW_ASSIGN 參數

資料類型	參數	使用	說明
字串	<i>stringOut</i>	OUT	一個包含與 <i>stringIn</i> 相同之文字字串的變數。
字串	<i>stringIn</i>	IN	一個變數或文字字串。

範例

範例 1:

```
@DTW_ASSIGN(RC, "0")
```

- 將 RC 設定為 "0"。

範例 2:

```
@DTW_ASSIGN(string1, string2)
```

- 將 string1 設定為 string2 的值。

DTW_CONCAT

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

連接兩個字串。

語法

- @DTW_CONCAT(stringIn1, stringIn2, stringOut)
- @DTW_rCONCAT(stringIn1, stringIn2)

參數

表 21. DTW_CONCAT 參數

資料類型	參數	使用	說明
字串	<i>stringIn1</i>	IN	一個變數或文字字串。
字串	<i>stringIn2</i>	IN	一個變數或文字字串。
字串	<i>stringOut</i>	OUT	包含 'stringIn1 + stringIn2' 之字串的變數。

範例

範例 1:

```
@DTW_CONCAT("This", " is a test.", result)
```

- 傳回: result = "This is a test."

範例 2:

```
@DTW_CONCAT(string1, "Jose!", result)
```

- 輸入: string1 = "No way "
- 傳回: result = "No way Jose!"

範例 3:

```
@DTW_rCONCAT("This", " is a test.")
```

- 傳回: "This is a test."

DTW_DELSTR

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

從第 *n* 個字元起刪除長度字元所指定字串的子字串。

語法

- @DTW_DELSTR(stringIn, *n*, length, stringOut)
- @DTW_DELSTR(stringIn, *n*, stringOut)
- @DTW_rDELSTR(stringIn, *n*, length)
- @DTW_rDELSTR(stringIn, *n*)

參數

表 22. DTW_DELSTR 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
整數	<i>n</i>	IN	要開始刪除子字串的字元位置。如果 <i>n</i> 大於 <i>stringIn</i> 的長度，則 <i>stringOut</i> 會被設定為 <i>stringIn</i> 的值。
整數	長度	OUT	所要刪除之子字串的長度。預設值是刪除所有的字元，直到 <i>stringIn</i> 的尾端。
字串	<i>stringOut</i>	OUT	一個包含 <i>stringIn</i> 修改格式的變數。

範例

範例 1:

```
@DTW_DELSTR("abcde", "3", "2", result)
```

- 傳回：result = "abe"

範例 2:

```
@DTW_rDELSTR("abcde", "6", "1")
```

- 傳回："abcde"

DTW_INSERT

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將一個字串插入至另一個在第 *n* 個字元之後開始的字串。

語法

- @DTW_INSERT(stringIn1, stringIn2, n, length, pad, stringOut)
- @DTW_INSERT(stringIn1, stringIn2, n, length, stringOut)
- @DTW_INSERT(stringIn1, stringIn2, n, stringOut)
- @DTW_INSERT(stringIn1, stringIn2, stringOut)
- @DTW_rINSERT(stringIn1, stringIn2, n, length, pad)
- @DTW_rINSERT(stringIn1, stringIn2, n, length)
- @DTW_rINSERT(stringIn1, stringIn2, n)
- @DTW_rINSERT(stringIn1, stringIn2)

參數

表 23. DTW_INSERT 參數

資料類型	參數	使用	說明
字串	<i>stringIn1</i>	IN	一個被插入 <i>stringIn2</i> 的變數或文字字串。
字串	<i>stringIn2</i>	IN	一個變數或文字字串。
整數	<i>n</i>	IN	字元是位於插入 <i>stringIn1</i> 之後的 <i>stringIn2</i> 中。如果 <i>n</i> 大於 <i>stringIn2</i> 的長度，則將會使用填補字元， <i>pad</i> 來填補不足的字元。預設值是在 <i>stringIn2</i> 的開頭插入。
整數	<i>length</i>	IN	所要插入之 <i>stringIn1</i> 字元的數目。如果這個參數大於 <i>stringIn1</i> 的長度，則將會以填補字元， <i>pad</i> 來填滿字串。預設值是 <i>stringIn1</i> 的長度。
整數	<i>pad</i>	IN	如同說明般，使用作為 <i>n</i> 以及 <i>length</i> 的填補字元。預設 <i>pad</i> 字元是空白。
字串	<i>stringOut</i>	OUT	一個包含由插入部分或所有 <i>stringIn1</i> 所修改的 <i>stringIn2</i> 。

範例

範例 1:

```
@DTW_INSERT("123", "abc", result)
```

- 傳回：result = "123abc"

範例 2:

```
@DTW_INSERT("123", "abc", "5", result)
```

- 傳回：result = "abc 123"

範例 3:

```
@DTW_INSERT("123", "abc", "5", "6", result)
```

- 傳回：result = "abc 123"

範例 4:

```
@DTW_INSERT("123", "abc", "5", "6", "_", result)
```

- 傳回：result = "abc__123__"

範例 5:

```
@DTW_rINSERT("123", "abc", "5", "6", "+")
```

- 傳回："abc++123++"

DTW_LASTPOS

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回在另一個字串中之最後一個字串所出現的位置，從第 *n* 個字元開始往回算。

語法

- @DTW_LASTPOS(stringIn1, stringIn2, n, position)
- @DTW_LASTPOS(stringIn1, stringIn2, position)
- @DTW_rLASTPOS(stringIn1, stringIn2, n)
- @DTW_rLASTPOS(stringIn1, stringIn2)

參數

表 24. DTW_LASTPOS 參數

資料類型	參數	使用	說明
字串	<i>stringIn1</i>	IN	一個搜尋 <i>stringIn2</i> 的變數或文字字串。
字串	<i>stringIn2</i>	IN	一個變數或文字字串。
整數	<i>n</i>	IN	位於 <i>stringIn2</i> 中開始搜尋 <i>stringIn1</i> 的字元位置。預設值是從第一個字元開始搜尋。
整數	<i>position</i>	OUT	在 <i>stringIn2</i> 中出現最後一個 <i>stringIn1</i> 的位置。如果找不到的話，將會傳回 0。

範例

範例 1:

```
@DTW_LASTPOS(" ", "abc def ghi", result)
```

- 傳回：result = "8"

範例 2:

```
@DTW_LASTPOS(" ", "abc def ghi", "10", result)
```

- 傳回：result = "8"

範例 3:

```
@DTW_rLASTPOS(" ", "abc def ghi", "7")
```

- 傳回："4"

DTW_LENGTH

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回字串的長度。

語法

- @DTW_LENGTH(stringIn, length)
- @DTW_rLENGTH(stringIn)

參數

表 25. DTW_LENGTH 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
整數	<i>length</i>	OUT	包含 <i>stringIn</i> 長度的符號。

範例

範例 1:

```
@DTW_LENGTH("abcdefgh", result)
```

- 傳回：result = "8"

範例 2:

```
@DTW_rLENGTH("")
```

- 傳回："0"

DTW_LOWERCASE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回一個全部都是小寫的字串。

語法

- @DTW_LOWERCASE(stringIn, stringOut)
- @DTW_rLOWERCASE(stringIn)
- @DTW_mLOWERCASE(stringMult1, stringMult2, ..., stringMultn)

參數

表 26. DTW_LOWERCASE 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個有大小寫字元或文字字串。
字串	<i>stringOut</i>	OUT	一個包含所有都是小寫字元之 <i>stringIn</i> 的變數。
字串	<i>stringMult</i>	INOUT	<ul style="list-style-type: none">• 在輸入上：包含字串的變數。• 在輸出上：一個包含被轉換為大寫之輸入字串的變數。

範例

範例 1:

```
@DTW_LOWERCASE("This", stringOut)
```

- 傳回：stringOut = "this"

範例 2:

```
@DTW_rLOWERCASE(string1)
```

- 輸入：string1 = "Web Pages"
- 傳回："web pages"

範例 3:

```
@DTW_mLOWERCASE(string1, string2, string3)
```

- 輸入：string1 = "THIS", string2 = "IS", string3 = "LOWERCASE"
- 傳回：string1 = "this", string2 = "is", string3 = "lowercase"

DTW_POS

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回在另一個字串中第一個出現字串的位置。

語法

- @DTW_POS(stringIn1, stringIn2, n, nOut)
- @DTW_POS(stringIn1, stringIn2, nOut)
- @DTW_rPOS(stringIn1, stringIn2, n)
- @DTW_rPOS(stringIn1, stringIn2)

參數

表 27. DTW_POS 參數

資料類型	參數	使用	說明
字串	<i>stringIn1</i>	IN	一個所要搜尋的變數或文字字串。
字串	<i>stringIn2</i>	IN	一個所要搜尋的變數或文字字串。
整數	<i>n</i>	IN	在 <i>stringIn2</i> 中開始進行搜尋的字元位置。預設值是開始在 <i>stringIn2</i> 的第一個字元進行搜尋。
整數	<i>nOut</i>	OUT	一個包含在 <i>stringIn2</i> 中第一個出現 <i>stringIn1</i> 之位置的變數。如果找不到的話，將會傳回 0。

範例

範例 1：

```
@DTW_POS("day", "Saturday", result)
```

- 傳回：result = "6"

範例 2:

```
@DTW_POS("a", "Saturday", "3", result)
```

- 傳回：result = "7"

範例 3:

```
@DTW_rPOS(" ", "abc def ghi", "5")
```

- 傳回："8"

DTW_REVERSE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將輸入字串反轉。

語法

- @DTW_REVERSE(stringIn, stringOut)
- @DTW_rREVERSE(stringIn)

參數

表 28. DTW_REVERSE 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個要反轉的變數或文字字串。
字串	<i>stringOut</i>	OUT	一個包含 <i>stringIn</i> 之反轉格式的變數。

範例

範例 1:

```
@DTW_REVERSE("This is it.", result)
```

- 傳回：result = ".ti si sihT"

範例 2:

```
@DTW_rREVERSE(string1)
```

- 輸入：string1 = "reversed"
- 傳回："desrever"

DTW_STRIP

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

除去在輸入字串中的前導空白、尾隨之空白，或兩者。

語法

- @DTW_STRIP(stringIn, option, stringOut)
- @DTW_STRIP(stringIn, stringOut)
- @DTW_rSTRIP(stringIn, option)
- @DTW_rSTRIP(stringIn)

參數

表 29. DTW_STRIP 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
字串	<i>option</i>	IN	指定要從 <i>stringIn</i> 中所除去的空白。預設值是 B。 <ul style="list-style-type: none">• B 或 b: 將前導與尾隨空白均除去• L 或 l: 只除去前導空白• T 或 t: 只除去尾隨空白
字串	<i>stringOut</i>	OUT	一個具有由選項所指定來除去空白之 <i>stringIn</i> 的變數。

範例

範例 1:

```
@DTW_STRIP(" day ", result)
```

- 傳回: result = "day"

範例 2:

```
@DTW_STRIP(" day ", "T", result)
```

- 傳回: result = " day"

範例 3:

```
@DTW_rSTRIP(" a day ", "L")
```

- 傳回: "a day "

DTW_SUBSTR

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

將輸入字串的子字串，與選用的填補字元一起傳回。

語法

- @DTW_SUBSTR(stringIn, n, length, pad, stringOut)
- @DTW_SUBSTR(stringIn, n, length, stringOut)
- @DTW_SUBSTR(stringIn, n, stringOut)
- @DTW_rSUBSTR(stringIn, n, length, pad)
- @DTW_rSUBSTR(stringIn, n, length)
- @DTW_rSUBSTR(stringIn, n)

參數

表 30. DTW_SUBSTR 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個所要搜尋的變數或文字字串。
整數	<i>n</i>	IN	一個子字串開始的字元位置。預設值是從 <i>stringIn</i> 的開始進行啟動
整數	<i>length</i>	IN	子字串的字元數目。預設值是其餘的字串。
字串	<i>pad</i>	IN	若是 <i>n</i> 大於 <i>stringIn</i> 的長度，或如果長度位於 <i>stringIn</i> 的終點之後，則將會使用填補字元。預設值是空白。
字串	<i>stringOut</i>	OUT	一個包含 <i>stringIn</i> 子字串的變數。

範例

範例 1:

```
@DTW_SUBSTR("abc", "2", result)
```

- 傳回：result = "bc"

範例 2:

```
@DTW_SUBSTR("abc", "2", "4", result)
```

- 傳回：result = "bc "

範例 3:

```
@DTW_SUBSTR("abc", "2", "4", ".", result )
```

- 傳回：result = "bc.."

範例 4:

```
@DTW_rSUBSTR("abc", "2", "6", ".")
```

- 傳回："bc...."

DTW_TRANSLATE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

在輸入字串中使用輸入和輸出轉換表格、`tableI` 和 `tableO` 的轉換字元。如果在參數列示中沒有 `tableI`、`tableO`、和 `pad`，則將會把 `stringIn` 轉換為大寫。如果 `tableI` 和 `tableO` 都在列示中，但沒有 `pad`，則將會搜尋 `tableI` 中的輸入字串，以及將其轉換為在 `tableO` 中的相符字元。如果 `pad` 在參數列示中，則 `stringIn` 和 `stringOut` 的長度是相同的。如果在輸入字串中的字元不在 `tableI` 中，則會使用 `pad` 字元。轉換表格可以是任何長度。如果沒有提供轉換表格，則輸入字元會被轉換為大寫。

語法

- `@DTW_TRANSLATE(stringIn, tableO, tableI, pad, stringOut)`
- `@DTW_TRANSLATE(stringIn, tableO, tableI, stringOut)`
- `@DTW_TRANSLATE(stringIn, tableO, stringOut)`
- `@DTW_TRANSLATE(stringIn, stringOut)`
- `@DTW_rTRANSLATE(stringIn, tableO, tableI, pad)`
- `@DTW_rTRANSLATE(stringIn, tableO, tableI)`
- `@DTW_rTRANSLATE(stringIn, tableO)`
- `@DTW_rTRANSLATE(stringIn)`

參數

表 31. *DTW_TRANSLATE* 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
字串	<i>tableO</i>	IN	使用作為轉換表格的變數或文字字串。請使用空值 ("") 來指定 <i>tableI</i> 或填塞；否則這個參數將是可選用的。
字串	<i>tableI</i>	IN	一個搜尋 <i>stringIn</i> 的變數或文字字串。請使用空值 ("") 來指定填塞；否則這個參數將是可選用的。
字串	填塞	IN	所使用的填補字元。預設值是空白。
字串	<i>stringOut</i>	OUT	一個包含 <i>stringIn</i> 子字串的變數。

範例

範例 1:

```
@DTW_TRANSLATE("abbc", result)
```

- 傳回：result = "ABBC"

範例 2:

```
@DTW_TRANSLATE("abbc", "R", "bc", result)
```

- 傳回：result = "aRR "

範例 3:

```
@DTW_rTRANSLATE("abcdef", "12", "abcd", ".")
```

- 傳回："12..ef"

範例 4:

```
@DTW_rTRANSLATE("abbc", "", "", "")
```

- 傳回："abbc"

DTW_UPPERCASE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

以大寫來傳回字串。

語法

- @DTW_UPPERCASE(stringIn, stringOut)
- @DTW_rUPPERCASE(stringIn)
- @DTW_mUPPERCASE(stringMult1, stringMult2, ..., stringMultn)

參數

表 32. DTW_UPPERCASE 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個有大小寫字元之變數或文字字串。
字串	<i>stringOut</i>	OUT	一個包含全部都是大寫字元之 <i>stringIn</i> 的變數。
字串	<i>stringMult</i>	INOUT	<ul style="list-style-type: none">• 在輸入上：包含字串的變數。• 在輸出上：包含輸入字串的一個變數，該字串的每一個單引號 (') 字元都被兩個單引號字元所置換。

範例

範例 1:

```
@DTW_UPPERCASE("Test", result)
```

- 傳回：結果 = "TEST"

範例 2:

```
@DTW_rUPPERCASE(string1)
```

- 輸入：string1 = "Web pages"
- 傳回："WEB PAGES"

範例 3:

```
@DTW_mUPPERCASE(string1, string2, string3)
```

- 輸入：string1 = "This", string2 = "is", string3 = "upper case"
- 傳回：string1 = "THIS", string2 = "IS", string3 = "UPPER CASE"

字組函數

這些函數支援字串函數並修改字組或字組集。Net.Data 將字組解譯為一個以空白做分隔的字串，或者一個在兩端均具有空白的字串。這裡將顯示幾個範例：

字串值	字組數
one two three	3
one , two , three	5
Part 2: Internet Sales Grow	5

- 第127頁的『DTW_DELWORD』
- 第128頁的『DTW_SUBWORD』
- 第129頁的『DTW_WORD』
- 第130頁的『DTW_WORDINDEX』
- 第131頁的『DTW_WORDLENGTH』
- 第132頁的『DTW_WORDPOS』
- 第134頁的『DTW_WORDS』

DTW_DELWORD

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回輸入字串的子字串。字組會從字組 *n* 中刪除由長度所指定的字組數目。

語法

- @DTW_DELWORD(stringIn, n, length, stringOut)
- @DTW_DELWORD(stringIn, n, stringOut)
- @DTW_rDELWORD(stringIn, n, length)
- @DTW_rDELWORD(stringIn, n

參數

表 33. DTW_DELWORD 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
整數	<i>n</i>	IN	要被刪除的第一個字組的位置。
整數	<i>length</i>	IN	要刪除的字組數目。預設值是刪除從 <i>n</i> 到 <i>stringIn</i> 尾端的所有字組。選用性的參數。
字串	<i>stringOut</i>	OUT	一個包含 <i>stringIn</i> 修改格式的變數。

範例

範例 1:

```
@DTW_DELWORD("Now is the time", "5", result)
```

- 傳回：result = "Now is the time"

範例 2:

```
@DTW_DELWORD("Now is the time", "2", result)
```

- 傳回：result = "Now"

範例 3:

```
@DTW_DELWORD("Now is the time", "2", "2", result)
```

- 傳回：result = "Now time"

範例 4:

```
@DTW_rDELWORD("Now is the time.", "3")
```

- 傳回：result = "Now is"

DTW_SUBWORD

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回輸入字串的子字串。該子字串是從字組 *n* 開始且繼續延伸至所指定的字組數目的長度。

語法

- @DTW_SUBWORD(stringIn, n, length, stringOut)
- @DTW_SUBWORD(stringIn, n, stringOut)
- @DTW_rSUBWORD(stringIn, n, length)
- @DTW_rSUBWORD(stringIn, n)

參數

表 34. DTW_SUBWORD 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
整數	<i>n</i>	IN	子字串之第一個字組的字組位置。如果這個值大於在 <i>stringIn</i> 中的字組數目，則將會傳回一個空值。
整數	<i>length</i>	IN	在子字串中的字組數目。如果這個值大於從 <i>n</i> 到 <i>stringIn</i> 尾端的字組數目，則將會傳回所有到 <i>stringIn</i> 尾端的字組。預設值是傳回所有從 <i>n</i> 到 <i>stringIn</i> 尾端的字組。
字串	<i>stringOut</i>	OUT	一個包含由 <i>n</i> 和長度所指定之 <i>stringIn</i> 子字串的變數。

範例

範例 1:

```
@DTW_SUBWORD("Now is the time", "5", result)
```

- 傳回：result = ""

範例 2:

```
@DTW_SUBWORD("Now is the time", "2", result)
```

- 傳回：result = "is the time"

範例 3:

```
@DTW_SUBWORD("Now is the time", "2", "2", result)
```

- 傳回：result = "is the"

範例 4:

```
@DTW_rSUBWORD("Now is the time", "3")
```

- 傳回："the time"

DTW_WORD

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

從所指定之輸入字串的位置中傳回一個單一字組。

語法

- @DTW_WORD(stringIn, n, stringOut)
- @DTW_rWORD(stringIn, n)

參數

表 35. DTW_WORD 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
整數	<i>n</i>	IN	所要傳回字組的位置。如果這個值大於在 <i>stringIn</i> 中的字組數目，則將會傳回一個空值。
字串	<i>stringOut</i>	OUT	一個包含在字組位置 <i>n</i> 之字組的變數。

範例

範例 1:

```
@DTW_WORD("Now is the time", "3", result)
```

- 傳回：result = "the"

範例 2:

```
@DTW_WORD("Now is the time", "5", result)
```

- 傳回：result = ""

範例 3:

```
@DTW_rWORD("Now is the time", "4")
```

- 傳回："time"

DTW_WORDINDEX

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回在輸入字串之字組之第 n 個字組第一個字元的字元位置。

語法

- @DTW_WORDINDEX(stringIn, n, stringOut)
- @DTW_rWORDINDEX(stringIn, n)

參數

表 36. DTW_WORDINDEX 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
整數	<i>n</i>	IN	所要索引之字組的字組位置。如果If 這個值大於在輸入字串中的字組數目，則將會傳回 0。
字串	<i>stringOut</i>	OUT	一個包含 <i>stringIn</i> 之第 <i>n</i> 個字串位置的變數。

範例

範例 1:

```
@DTW_WORDINDEX("Now is the time", "3", result)
```

- 傳回：result = "8"

範例 2:

```
@DTW_WORDINDEX("Now is the time", "6", result)
```

- 傳回：result = "0"

範例 3:

```
@DTW_rWORDINDEX("Now is the time", "2")
```

- 傳回："5"

DTW_WORDLENGTH

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回輸入字串之第 n 個的長度。

語法

- @DTW_WORDLENGTH(stringIn, n, stringOut)
- @DTW_rWORDLENGTH(stringIn, n)

參數

表 37. DTW_WORDLENGTH 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
整數	<i>n</i>	IN	您想要知道的字組長度之字組位置。如果這個值大於在輸入字串中的字組數目，則將會傳回 0。
字串	<i>stringOut</i>	OUT	一個包含在 <i>stringIn</i> 中之第 <i>n</i> 個字組長度的變數。

範例

範例 1:

```
@DTW_WORDLENGTH("Now is the time", "1", result)
```

- 傳回：result = "3"

範例 2:

```
@DTW_rWORDLENGTH("Now is the time", "6")
```

- 傳回："0"

DTW_WORDPOS

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回在另一個字串中所找到之某一個字串之第一個找到的字組數目。多重空白將以一個單一空白來看待，以供比較。對比較而言，是會區分大小寫的。

語法

- @DTW_WORDPOS(stringIn1, stringIn2, n, stringOut)
- @DTW_WORDPOS(stringIn1, stringIn2, stringOut)
- @DTW_rWORDPOS(stringIn1, stringIn2, n)
- @DTW_rWORDPOS(stringIn1, stringIn2)

參數

表 38. DTW_WORDPOS 參數

資料類型	參數	使用	說明
字串	<i>stringIn1</i>	IN	一個變數或文字字串。
字串	<i>stringIn2</i>	IN	一個所要搜尋的變數或文字字串。
整數	<i>n</i>	IN	在 <i>stringIn2</i> 中要開始進行搜尋的字組位置。如果這個值大於在 <i>stringIn2</i> 中的字組數目，則將會傳回 0。預設值是從 <i>stringIn2</i> 的開始進行搜尋。
字串	<i>stringOut</i>	OUT	在 <i>stringIn2</i> 中之 <i>stringIn1</i> 的字組位置。

範例

範例 1:

```
@DTW_WORDPOS("the", "Now is the time", result)
```

- 傳回：result = "3"

範例 2:

```
@DTW_WORDPOS("The", "Now is the time", result)
```

- 傳回：result = "0"

範例 3:

```
@DTW_WORDPOS("The", "Now is the time", "5", result)
```

- 傳回：result = "0"

範例 4:

```
@DTW_WORDPOS("is the", "Now is the time", result)
```

- 傳回：result = " 2"

範例 5:

```
@DTW_rWORDPOS("be", "To be or not to be", "3")
```

- 傳回："6"

DTW_WORDS

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回在字串中的字組數目。

語法

- @DTW_WORDS(stringIn, stringOut)
- @DTW_rWORDS(stringIn)

參數

表 39. DTW_WORDS 參數

資料類型	參數	使用	說明
字串	<i>stringIn</i>	IN	一個變數或文字字串。
字串	<i>stringOut</i>	OUT	一個包含在 <i>stringIn</i> 中之字組數目的變數。

範例

範例 1:

```
@DTW_WORDS("Now is the time", result)
```

- 傳回：result = "4"

範例 2:

```
@DTW_rWORDS(" ")
```

- 傳回："0"

表格函數

這些函數可藉著 Net.Data 表格的使用來簡化，且比使用 REXX、C、或 PERL，更有效率的撰寫您自己的函數。這些函數都會被使用在範例巨集 `tbtest.mac` 中，該巨集是附隨於 Net.Data 工具套裝軟體。

- 第136頁的『DTW_TB_DLIST』
- 第138頁的『DTW_TB_DUMPH』
- 第139頁的『DTW_TB_DUMPV』
- 第140頁的『DTW_TB_HTML_ENCODE』
- 第141頁的『DTW_TB_INPUT_CHECKBOX』
- 第142頁的『DTW_TB_INPUT_RADIO』
- 第143頁的『DTW_TB_INPUT_TEXT』
- 第144頁的『DTW_TB_LIST』
- 第146頁的『DTW_TB_SELECT』

- 第147頁的『DTW_TB_TABLE』
- 第149頁的『DTW_TB_TEXTAREA』

DTW_TB_DLIST

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

從巨集表格中傳回 HTML 定義列示。

語法

- @DTW_TB_DLIST(table, term, def, termstyle, defstyle, anchor, anchor_u, image, image_u)

參數

表 40. DTW_TB_DLIST 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	一個指定巨集表格變數給輸出當作 HTML 定義列示的符號。
整數	<i>term</i>	IN	在包含詞彙名稱值（於 <DT> 標籤之後之文字）之表格的直欄號碼。預設值是使用第一直欄。
整數	<i>def</i>	IN	在包含詞彙定義值（於 <DD> 標籤之後的文字）之表格的直欄號碼。預設值是使用第二直欄。
字串	<i>termstyle</i>	IN	一個包含詞彙名稱值之 HTML 元素列示的變數或文字字串。預設值是使用沒有樣式的標籤。
字串	<i>defstyle</i>	IN	一個包含詞彙定義值之 HTML 元素列示的變數或文字字串。預設值是使用沒有樣式的標籤。
字串	<i>anchor</i>	IN	指定哪一個 HTML 元素是由錨參照所產生的。有效的值有 DT 和 DD。預設值是不產生錨參照。
整數	<i>anchor_u</i>	IN	在包含錨參照的 URL 之表格中的直欄號碼。預設值是不產生錨參照。
字串	<i>image</i>	IN	指定列入壓縮檔所產生的是哪一個 HTML 元素。有效的值有 DT 和 DD。預設值是不產生列入壓縮檔。
整數	<i>image_u</i>	IN	在包含列入壓縮檔之 URL 之表格中的直欄號碼。預設值是不產生列入壓縮檔。

範例

```
@DTW_TB_DLIST(Mytable,"3","4","b i","strong","DD","2","DT","1")
@DTW_TB_DLIST(Mytable,"","4","b","strong","DT","2","DT","1")
@DTW_TB_DLIST(Mytable,"3","4","","","DT","2")
@DTW_TB_DLIST(Mytable,"","","i","b","DD","2","DT","1")
@DTW_TB_DLIST(Mytable,"","3","","","DD","2","DT","1")
@DTW_TB_DLIST(Mytable,"3","4","EM","B U I","DT","2","DT","1")
@DTW_TB_DLIST(Mytable,"","4","","","DD","2","DT","1")
@DTW_TB_DLIST(Mytable,"3","","I","I","DT","2","DD","1")
```


範例 1: HTML 根據表格資料產生第一個看起來類似這個的範例。

```
<DL>
<DT>
<IMG SRC="http://www.mycompany.com/images/image1.gif" ALT=""><b><i>image1text</i></b>
<DD>
<A HREF="http://www.mycompany.com/anchor1.html"><strong>anchor1text</strong></A>
<DT>
<IMG SRC="http://www.mycompany.com/images/image2.gif" ALT=""><b><i>image2text</i></b>
<DD>
<A HREF="http://www.mycompany.com/anchor2.html"><strong>anchor2text</strong></A>
<DT>
<IMG SRC="http://www.mycompany.com/images/image3.gif" ALT=""><b><i>image3text</i></b>
<DD>
<A HREF="http://www.mycompany.com/anchor3.html"><strong>anchor3text</strong></A>
<DT>
<IMG SRC="http://www.mycompany.com/images/image4.gif" ALT=""><b><i>image4text</i></b>
<DD>
<A HREF="http://www.mycompany.com/anchor4.html"><strong>anchor4text</strong></A>
</DT>
</DL>
```

DTW_TB_DUMP

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回巨集表格變數的內容。每個表格的列都被列示在不同的行上。整個表格是由 `<PRE></PRE>` 標籤所包圍。

語法

- `@DTW_TB_DUMP(table)`

參數

表 41. *DTW_TB_DUMP* 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	一個指定巨集表格變數給輸出的符號。

範例

範例 1:

`@DTW_TB_DUMP(Mytable)`

由這個範例所產生的 HTML 看起來類似這個：

```
<PRE>
名稱      部門      職位
Jack Smith Internet 技術 軟體工程師
Helen Williams 資料庫 開發部經理
Alex Jones 製造 工業工程師
Tom Baker 採購 業務代表
</PRE>
```

DTW_TB_DUMPV

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回巨集表格變數的內容。每一列都在不同的行上。整個表格是由 `<PRE>``</PRE>` 標籤所包圍。

語法

- @DTW_TB_DUMPV(table)

參數

表 42. DTW_TB_DUMPV 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	一個指定巨集表格變數給輸出的符號。

範例

範例 1:

@DTW_TB_DUMPV(Mytable)

由這個範例所產生的 HTML 看起來類似這個：

```
<PRE>
http://www.mycompany.com/images/image1.gif
http://www.mycompany.com/anchor1.html
imagetext
anchor1text
http://www.mycompany.com/images/image2.gif
http://www.mycompany.com/anchor2.html
image2text
anchor2text
http://www.mycompany.com/images/image3.gif
http://www.mycompany.com/anchor3.html
image3text
anchor3text
http://www.mycompany.com/images/image4.gif
http://www.mycompany.com/anchor4.html
image4text
anchor4text
</PRE>
```

DTW_TB_HTMLENCODE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回具有這些 HTML 字元編碼的輸入巨集表格：

& 記號	&	&
雙引號	"	"
大於	>	>
小於	<	<

語法

- @DTW_TB_HTMLENCODE(table, collist)

參數

表 43. DTW_TB_HTMLENCODE 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	INOUT	所要修改的巨集表格變數。
字串	<i>collist</i>	IN	所要編碼的直欄號碼。預設值是對所有的直欄進行編碼。

範例

範例 1:

```
@DTW_TB_HTMLENCODE(Mytable, "3 4")
```

在指定的表格之直欄 3 和直欄 4 中的特殊字元被它們的編碼格式所置換。

DTW_TB_INPUT_CHECKBOX

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

從巨集表格變數中傳回一個或多個 HTML 核對框輸入標籤。

語法

- @DTW_TB_INPUT_CHECKBOX(table, prompt, namecol, valuecol, rows, checkedrows)

參數

表 44. DTW_TB_INPUT_CHECKBOX 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	顯示為核對框輸入標籤的巨集表格變數。
字串	<i>prompt</i>	IN	在表格中的欄號碼，或包含顯示在核對框旁之文字的字串。這個是所需的參數，但可以巨有一個空 ("") 值。當其是空值的時候，所使用的值是名稱直欄所定義的值。
字串	<i>namecol</i>	IN	在表格中的直欄號碼或包含輸入欄位名稱的字串。
字串	<i>valuecol</i>	IN	在表格中的直欄號碼或一個包含輸入欄位值的字串。預設值是 1。
字串	<i>rows</i>	IN	在產生輸入欄位之表格之列的列示。預設值是使用所有的列。
整數	<i>checkedrows</i>	IN	指定所要檢查之表格之列的列示。預設值是不檢查位。

範例

```
@DTW_TB_INPUT_CHECKBOX(Mytable,"","5")
@DTW_TB_INPUT_CHECKBOX(Mytable,"3","5")
@DTW_TB_INPUT_CHECKBOX(Mytable,"3","5","4")
@DTW_TB_INPUT_CHECKBOX(Mytable,"3","5","", "2 5")
@DTW_TB_INPUT_CHECKBOX(Mytable,"3","4","", "2 3 4", "1 3 4")
```

由 HTML 所產生的最後一個範例看起來類似這個：

```
<INPUT TYPE="CHECKBOX" NAME="anchor2text" VALUE="1">image2text<BR>
<INPUT TYPE="CHECKBOX" NAME="anchor3text" VALUE="1" CHECKED>image3text<BR>
<INPUT TYPE="CHECKBOX" NAME="anchor4text" VALUE="1" CHECKED>image4text<BR>
```

DTW_TB_INPUT_RADIO

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

從巨集表格變數中傳回一個或多個 HTML 圓鈕輸入標籤。

語法

- @DTW_TB_INPUT_RADIO(table, prompt, namecol, valuecol, rows, checkedrows)

參數

表 45. DTW_TB_INPUT_RADIO 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	顯示為圓鈕輸入標籤的巨集表格變數。
字串	<i>prompt</i>	IN	在表格中或包含顯示於圓鈕旁邊之文字字串的直欄號碼。所需的參數，但是可以包含一個空 (") 值。當是空值時，請使用值欄的值。
字串	<i>namecol</i>	IN	在表格中的直欄號碼或包含輸入欄位欄位名稱的字串。
字串	<i>valuecol</i>	IN	在表格中的直欄號碼或一個包含輸入欄位值的字串。
字串	<i>rows</i>	IN	在產生輸入欄位之表格之列的列示。預設值是使用所有的列。
整數	<i>checkedrows</i>	IN	在表格中的列號碼顯示所勾選之相對應的圓鈕。只有一個值是被允許的。

範例

```
@DTW_TB_INPUT_RADIO(Mytable,"","Radio1","5")
@DTW_TB_INPUT_RADIO(Mytable,"3","Radio2","3")
@DTW_TB_INPUT_RADIO(Mytable,"3","Radio3","4")
@DTW_TB_INPUT_RADIO(Mytable,"3","Radio4","4","2 3 4","4")
```

由 HTML 所產生的最後一個範例看起來類似這個：

```
<INPUT TYPE="RADIO" NAME="Radio4" VALUE="anchor2text">image2text<BR>
<INPUT TYPE="RADIO" NAME="Radio4" VALUE="anchor3text">image3text<BR>
<INPUT TYPE="RADIO" NAME="Radio4" VALUE="anchor4text" CHECKED>image4text<BR>
```

DTW_TB_INPUT_TEXT

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

從 Net.Data 巨集 TABLE 變數中傳回一個或多個輸入文字標籤。

語法

- @DTW_TB_INPUT_TEXT(table, prompt, namecol, valuecol, size, maxlen, rows)

參數

表 46. DTW_TB_INPUT_TEXT 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	顯示為文字輸入標籤的巨集 TABLE 變數。
字串	<i>prompt</i>	IN	在表格中的欄號碼或包含顯示在輸入欄旁之文字的字串。如果所顯示的是一個空 (") 值，則將不會有文字顯現。
字串	<i>namecol</i>	IN	在表格中的欄號碼或包含輸入欄位欄位名稱的字串。
字串	<i>valuecol</i>	IN	在表格中的欄號碼或一個包含為預設輸入欄位值得字串，其是為 VALUE 屬性所指定的。預設值是不產生 VALUE 屬性值。
整數	<i>size</i>	IN	輸入欄位的字元號碼，其是為 SIZE 屬性所指定的。預設值是最長預設輸入值的長度，或者如果沒有預設輸入時則是 10。
整數	<i>maxlen</i>	IN	輸入字串的最大長度，是為 MAXLEN 屬性所指定的。預設值是不產生 MAXLENGTH 屬性值。
整數	<i>rows</i>	IN	在產生輸入欄位之表格之列的列示。預設值是使用所有的列。

範例

```
@DTW_TB_INPUT_TEXT(Mytable,"","5")
@DTW_TB_INPUT_TEXT(Mytable,"","5","4")
@DTW_TB_INPUT_TEXT(Mytable,"Enter title:","5","4")
@DTW_TB_INPUT_TEXT(Mytable,"3","5","4")
@DTW_TB_INPUT_TEXT(Mytable,"3","5","4","30")
@DTW_TB_INPUT_TEXT(Mytable,"3","5","4","35","40")
@DTW_TB_INPUT_TEXT(Mytable,"3","3","4","35","40","1 2 3")
```

由 HTML 所產生的最後一個範例看起來類似這個：

```
<P>image1text
<INPUT TYPE="TEXT" NAME="image1text" VALUE="anchor1text" SIZE="35" MAXLENGTH="40">
<P>image2text
<INPUT TYPE="TEXT" NAME="image2text" VALUE="anchor2text" SIZE="35" MAXLENGTH="40">
<P>image3text
<INPUT TYPE="TEXT" NAME="image3text" VALUE="anchor3text" SIZE="35" MAXLENGTH="40">
```

DTW_TB_LIST

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回 HTML 列示。

語法

- @DTW_TB_LIST(table, listtype, listitem, itemstyle, anchor_u, image_u)

參數

表 47. DTW_TB_LIST 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	指定巨集表格變數給輸出當作 HTML 列示的符號。
字串	<i>listtype</i>	IN	所要產生的列示類型。可接受的值包括： <ul style="list-style-type: none">• DIR• MENU• OL• UL
整數	<i>listitem</i>	IN	在包含列示值（於 標籤之後的文字）之表格中的直欄號碼。預設值是使用第一直欄。
字串	<i>itemstyle</i>	IN	一個包含詞彙名稱值之 HTML 元素列示的變數或文字字串。預設值是使用沒有樣式的標籤。
整數	<i>anchor_u</i>	IN	在包含錨參照的 URL 之表格中的直欄號碼。如果並未指定這個值，則將不會產生錨。
整數	<i>image_u</i>	IN	在包含線內壓縮檔之 URL 之表格中的直欄號碼。如果並未指定這個值，則將不會產生列入壓縮檔。

範例

```
@DTW_TB_LIST(Mytable,"MENU")
@DTW_TB_LIST(Mytable,"MENU","3")
@DTW_TB_LIST(Mytable,"UL","3","","2")
@DTW_TB_LIST(Mytable,"UL","4","B","2","1")
@DTW_TB_LIST(Mytable,"DIR","3","b i")
@DTW_TB_LIST(Mytable,"OL","4","TT","U","2","1")
```

由 HTML 所產生的最後一個範例看起來類似這個：

```
<TT><U>
<OL>
<LI><A HREF="http://www.mycompany.com/anchor1.html">
<IMG SRC="http://www.mycompany.com/images/image1.gif" ALT="">anchor1text</A>
<LI><A HREF="http://www.mycompany.com/anchor2.html">
<IMG SRC="http://www.mycompany.com/images/image2.gif" ALT="">anchor2text</A>
```



```
<LI><A HREF="http://www.mycompany.com/anchor3.html">  
<IMG SRC="http://www.mycompany.com/images/image3.gif" ALT="">anchor3text</A>  
<LI><A HREF="http://www.mycompany.com/anchor4.html">  
<IMG SRC="http://www.mycompany.com/images/image4.gif" ALT="">anchor4txt</A>  
</OL>  
</U></TT>
```

DTW_TB_SELECT

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

傳回一個 HTML SELECT 功能表。

語法

- @DTW_TB_SELECT(table, name, optioncol, size, multiple, rows, selectrows)

參數

表 48. DTW_TB_SELECT 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	顯示為 SELECT 欄位的巨集表格變數。
字串	<i>name</i>	IN	SELECT 欄位之 NAME 屬性的值。
整數	<i>optioncol</i>	IN	在表格（該表格具有使用在 SELECT 欄位中之 OPTION 標籤中的值）中的直欄號碼。預設值是使用第一直欄。
整數	<i>size</i>	IN	在使用作為在 SELECT 欄位中之 OPTION 標籤之表格中的列號碼。預設值是使用所有的列。
字串	<i>multiple</i>	IN	指定是否允許所作的多重選擇。預設值是 N，表示不允許多重選擇。
字串	<i>selectedrows</i>	IN	從表格中選取要使用在選取之欄位中的列。預設值是使用所有的列。
字串	<i>rows</i>	IN	從已檢查過標籤之表格中選取列的列示。若要指定多於一行，您必須將多重參數設定為 Y。預設值是選取第一個項目。

範例

```
@DTW_TB_SELECT(Mytable,"URL0")
@DTW_TB_SELECT(Mytable,"URL1","3")
@DTW_TB_SELECT(Mytable,"URL1","3","9")
@DTW_TB_SELECT(Mytable,"URL3","3","","y")
@DTW_TB_SELECT(Mytable,"URL4","2","3","y","2 4")
@DTW_TB_SELECT(Mytable,"URL5","3","","y","","2 5")
@DTW_TB_SELECT(Mytable,"URL6","3","","y","1 2 4","1 4")
```

由 HTML 所產生的最後一個範例看起來類似這個：

```
<SELECT NAME="URL6" SIZE="3" MULTIPLE>
<OPTION SELECTED>image1text
<OPTION>image2text
<OPTION>image4text
</SELECT>
```

DTW_TB_TABLE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

從巨集表格變數中傳回 HTML 表格。

語法

- @DTW_TB_TABLE(table, options, collist, cellstyle, anchor_u, image_u, url_text, url_style)

參數

表 49. DTW_TB_TABLE 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	一個以 HTML 表格輸入的巨集表格變數。
字串	<i>options</i>	IN	位於 TABLE 標籤中的表格屬性。預設值示不使用屬性。有效的值包括： <ul style="list-style-type: none">• BORDER• CELSPACING• WIDTH
字串	<i>collist</i>	IN	使用在 HTML 表格中之表格的欄號碼。預設值是使用所有的欄。
字串	<i>cellstyle</i>	IN	HTML 樣式元素的列示，例如 B 和 I, 是圍繞在每個 TD 標籤旁的文字。預設值是不使用樣式標籤。
整數	<i>anchor_u</i>	IN	在表格中的欄號碼包含使用來建立錨參照的 URL。您必須也在 collist 中指定欄。預設值是不產生錨參照標籤。
整數	<i>image_u</i>	IN	在包含使用來建立線內壓縮檔之 URL 的表格中的欄號碼。您必須也在 collist 中指定欄。預設值是不產生壓縮檔標籤。
整數	<i>url_text</i>	IN	在表格中包含顯示錨參照或線內影像檔的文字的欄號碼。預設值是使用 URL 它本身。
字串	<i>url_style</i>	IN	指定在 url_text 中文字之 HTML 樣式元素的列示。預設值是不產生樣式標籤。

範例

```
@DTW_TB_TABLE(Mytable,"BORDER","4 2 1","i","2","1","4","b")
@DTW_TB_TABLE(Mytable)
@DTW_TB_TABLE(Mytable,"BORDER")
@DTW_TB_TABLE(Mytable,"","3 4")
@DTW_TB_TABLE(Mytable,"BORDER","2 3 4","em")
```

為第一個範例所產生的 HTML 看起來類似這個：

```
<TABLE BORDER>
<TR>
<TH>TITLE
<TH>ANCHORURL
```

```

<TH>IMAGEURL
<TR>
<TD><i>anchor1text</i>
<TD><A HREF="http://www.mycompany.com/anchor1.html"><b>anchor1text</b></A>
<TD><IMG SRC="http://www.mycompany.com/images/image1.gif" ALT=""><b>anchor1text</b>
<TR>
<TD><i>anchor2text</i>
<TD><A HREF="http://www.mycompany.com/anchor2.html"><b>anchor2text</b></A>
<TD><IMG SRC="http://www.mycompany.com/images/image2.gif" ALT=""><b>anchor2text</b>
<TR>
<TD><i>anchor3text</i>
<TD><A HREF="http://www.mycompany.com/anchor3.html"><b>anchor3text</b></A>
<TD><IMG SRC="http://www.mycompany.com/images/image3.gif" ALT=""><b>anchor3text</b>
</TABLE>

```

DTW_TB_TEXTAREA

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X	X	X	X	X

從巨集表格變數中傳回 HTML TEXTAREA 標籤。

語法

- @DTW_TB_TEXTAREA(table, name, numRows, numcols, valuecol, rows)

參數

表 50. DTW_TB_TEXTAREA 參數

資料類型	參數	使用	說明
陣列	<i>table</i>	IN	一個顯示為 TEXTAREA 標籤的巨集表格變數
字串	<i>name</i>	IN	文字區域的名稱。
整數	<i>numrows</i>	IN	所要顯示的列數。預設值是在表格中的列數。
整數	<i>numcols</i>	IN	所要顯示之直欄號碼。預設值是在表格中最長列的長度。
整數	<i>valuecol</i>	IN	表格（其值顯示在文字區域中）的直欄數。預設值是第一直欄。
字串	<i>rows</i>	IN	在表格中的列是使用來產生 TEXTAREA 標籤。預設值是使用所有的列。

範例

```
@DTW_TB_TEXTAREA(Mytable,"textarea1")
@DTW_TB_TEXTAREA(Mytable,"textarea2","3")
@DTW_TB_TEXTAREA(Mytable,"textarea3","3","40")
@DTW_TB_TEXTAREA(Mytable,"textarea4","2","80","3")
@DTW_TB_TEXTAREA(Mytable,"textarea5","3","70","4","1 3 4")
```

由 HTML 所產生的最後一個範例看起來類似這個：

```
<TEXTAREA NAME="textarea5" ROWS="3" COLS="70">
anchor1text
anchor3text
anchor4text
<TEXTAREA>
```

純本文檔案介面

純本文檔案介面是為 DB2 在大部分平台進行預先架構之語言環境。從純本文檔案資源中（純本文），您可以 開啟、讀取、並操作資料，並使用在 Net.Data 巨集中的函數來將資料儲存在純本文檔案中。

純本文檔案介面區隔字元

爲了增進執行效能，您可以保留來自純本文檔案中之 SQL 系列要求的 Net.Data 列表輸出。您可以在後續的要求中來取回純本文檔案，以置換 SQL 要求的再次發出。

Net.Data 純本文檔案可以從 Net.Data 表格來建立，而 Net.Data 表格可以從純本文檔案來建置。爲了製作表格和純本文檔案之間的轉換，您必須在表格和純本文檔案中的記錄中定義直欄之間的對映。區隔字元提供如何將部分之純本文檔案中的記錄分配與對映到表格中的之欄的方式，以及如何可將在表格中的欄對映到純本文檔案中的記錄中。

有兩種區隔字元：

新行字元 (ASCITEXT)

當您的表格只有一欄時，請使用這個轉換。Net.Data 會將每個在相對應純本文檔案中的記錄對映到在表格中的單一系列。在這個狀況下，將在純本文檔案中的記錄分開所使用的一般新行字元只有區隔字元。

新行字元和區隔字元字串 (DELIMITED)

當您的表格有多重欄時，請使用這個轉換。當 Net.Data 從表格中的列建立一個純本文檔案記錄時，它會將區隔字元字串以一個介於項目之間的區隔符號來置換。當 Net.Data 從純本文檔案來重新建立表格時，它會使用區隔字元字串來決定在表格的欄中要用置放多少列。在這個範例中，一般性的新線字元會分格在純本文檔案中的記錄，該純本文檔案將與表格中的列相對應，且區隔字元字串在一個單一記錄內會將項目分開。

您可以使用 DTWF_SEARCH 函數來取回某些特定保留在從 Net.Data 表格來建立之純本文檔案中的記錄。指定一個在 DTWF_SEARCH 中的字串來傳回所有包含在純本文檔案中當作 Net.data 表格列之字串的記錄。

純本文檔案介面函數

- 第151頁的『DTWF_APPEND』
- 第153頁的『DTWF_CLOSE』
- 第154頁的『DTWF_DELETE』
- 第154頁的『DTWF_DELETE』
- 第158頁的『DTWF_OPEN』
- 第159頁的『DTWF_READ』
- 第161頁的『DTWF_REMOVE』
- 第162頁的『DTWF_SEARCH』
- 第164頁的『DTWF_UPDATE』
- 第166頁的『DTWF_WRITE』

DTWF_APPEND

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

將表格的變數的目次寫入檔案尾端。

語法

- @DTWF_APPEND(filename, transform, delimiter, table, retry, rows)

參數

表 51. DTWF_APPEND 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	IN	新增表格變數目次之檔案的名稱。
字串	<i>transform</i>	IN	檔案的格式： <ul style="list-style-type: none">• ASCIITEXT -使用介於欄值間的新行字元來撰寫表格，並忽略區隔字元參數。• DELIMITED - 使用指定在區隔字元參數中的區隔字元來將表格寫入檔案中。
字串	<i>delimiter</i>	IN	一個指示值終止的字元字串。這個參數是區分大小寫的。若轉換是 ASCIITEXT 的話，則將被忽略。
字串	<i>table</i>	IN	一個從中讀取記錄的表格變數。
整數	<i>retry</i>	IN	如果檔案無法立刻附加的話，所要重試的次數。預設值是不重試。
整數	<i>rows</i>	IN	從表格中所附加之最大的列數目。預設值是附加所有的列。指定 0 將附加所有的列。

範例

範例 1:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
}%  
@DTWF_APPEND(myFile, "DELIMITED", " ;", myTable)
```

範例 2:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
}%  
@DTWF_APPEND(myFile, "ASCIITEXT", " ;", myTable)
```

範例 3:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
}%  
@DTWF_APPEND(myFile, "ASCITEXT", " ;", myTable, "0", "10")
```


DTWF_CLOSE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

關閉由 DTWF_OPEN 所開啓的檔案。

語法

- @DTWF_CLOSE(filename, retry)

參數

表 52. DTWF_CLOSE 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	IN	所要關閉之檔案的名稱。當檔案被關閉時，所傳回的是檔案的完全路徑名稱。
整數	<i>retry</i>	IN	如果檔案無法立刻關閉的話，所要重試的次數。預設值是不重試。

範例

範例 1:

```
@DTWF_CLOSE(myFile, 5)
```

DTWF_DELETE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

從檔案中刪除記錄。（並不刪除空檔案。）

語法

- @DTWF_DELETE(filename, transform, delimiter, retry, rows, startrow)

參數

表 53. DTW_DELETE 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	IN	所要刪除之記錄的檔案名稱。
字串	<i>transform</i>	IN	檔案的格式： <ul style="list-style-type: none">• ASCIITEXT -使用介於欄值間的新行字元來撰寫表格，並忽略區隔字元參數。• DELIMITED - 使用指定在區隔字元參數中的區隔字元來將表格寫入檔案中。
字串	<i>delimiter</i>	IN	一個指示值終止的字元字串。這個參數是區分大小寫的。若轉換是 ASCIITEXT 的話，則將被忽略。
整數	<i>retry</i>	IN	如果檔案無法立刻刪除的話，所要重試的次數。預設值是不重試。
整數	<i>列</i>	IN	所要刪除之最大的列數目。預設值是刪除所有的列。指定 0 將刪除所有的列。
整數	<i>startrow</i>	IN	開始刪除的列號碼。值 1 表示從第一列開始刪除。如果這個值大於在檔案中的列號碼，則該值會被變更為最後一個記錄並以錯誤傳回。預設值是從 1 開始。

範例

範例 1:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
    myWait = "5000"  
    myRows = "2"  
}%  
@DTWF_DELETE(myFile, "Delimited", "|", myWait, myRows)
```

範例 2:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
    myStart = "1"
```

```
        myRows = "2"  
    %}  
    @DTWF_DELETE(myFile, "AsciiText", "|", "0", myRows, myStart)
```

DTWF_INSERT

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

將記錄插入檔案。

語法

- @DTWF_INSERT(filename, transform, delimiter, table, retry, rows, startrow)

參數

表 54. DTWF_INSERT 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	INOUT	插入記錄的檔案名稱。傳回檔案的完全路徑名稱。
字串	<i>transform</i>	IN	檔案的格式： <ul style="list-style-type: none">• ASCIITEXT -使用介於欄值間的新行字元來撰寫表格，並忽略區隔字元參數。• DELIMITED - 使用指定在區隔字元參數中的區隔字元來將表格寫入檔案中。
字串	<i>delimiter</i>	IN	一個指示值終止的字元字串。這個參數是區分大小寫的。若轉換是 ASCIITEXT 的話，則將被忽略。
字串	<i>table</i>	IN	從該處將記錄插入表格的表格變數。
整數	<i>retry</i>	IN	如果檔案無法立刻被寫入的話，所要重試的次數。預設值是不重試。
整數	<i>rows</i>	IN	從表格所插入之最大的列數。預設值是插入所有的列。值 0 將會插入所有的列。
整數	<i>startrow</i>	IN	開始插入之列的號碼。指定 1 表示從第一列開始插入。如果這個值大於在檔案中的列號碼，則該值會被變更為最後一個記錄並以錯誤傳回。預設值是從 1 開始。

範例

範例 1:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
    myWait = "3000"  
}%  
@DTWF_INSERT(myFile, "Delimited", "|", myTable, myWait)
```

範例 2:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE
```

```
    myStart = "1"  
    myRows = "2"  
%}  
@DTWF_INSERT(myFile, "AsciiText", "|", myTable, "0", myRows, myStart)
```

DTWF_OPEN

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

開啓一個檔案。該檔案會保持開啓直到它被 `DTWF_CLOSE` 關閉，以讓諸如 `DTWF_READ` 和 `DTWF_WRITE` 等函數來在沒有呼叫 `DTWF_OPEN` 的狀況下，自動開啓檔案。

語法

- `@DTWF_OPEN(filename, mode, retry)`

參數

表 55. `DTWF_OPEN` 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	INOUT	所要開啓的檔案名稱。所傳回的路徑和檔案名稱。
字串	<i>mode</i>	IN	所要求的存取類型： <ul style="list-style-type: none">• <code>r</code> - 開啓舊有的檔案以供讀取。• <code>w</code> - 建立一個檔案以供撰寫。（將舊有之相同的檔案名稱銷毀，如果存在的話。）• <code>a</code> - 開啓一個檔案以供附加。如果找不到的話，<code>Net.Data</code> 會建立檔案。• <code>r+</code> - 開啓一個舊有的檔案以供讀取和寫入。• <code>w+</code> - 建立一個檔案以供讀取和寫入。（將舊有之相同的檔案名稱銷毀，如果存在的話。）• <code>a+</code> - 在附加模式中開啓檔案以供讀取或附加。如果找不到的話，<code>Net.Data</code> 會建立檔案。
整數	<i>retry</i>	IN	如果檔案無法立刻開啓的話，所要重試的次數。預設值是不重試。

範例

範例 1:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myMode = "r+"  
}%  
@DTWF_OPEN(myFile, myMode, "1000")
```

DTWF_READ

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

從檔案中將記錄讀取至表格變數。

語法

- @DTWF_READ(filename, transform, delimiter, table, retry, rows, startrow, columns)

參數

表 56. DTWF_READ 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	INOUT	將其記錄讀取至表格變數中的檔案名稱。所傳回的路徑和檔案名稱。
字串	<i>transform</i>	IN	檔案的格式： <ul style="list-style-type: none">• ASCIITEXT -使用介於欄值間的新行字元來撰寫表格，並忽略區隔字元參數。• DELIMITED - 使用指定在區隔字元參數中的區隔字元來將表格寫入檔案中。
字串	<i>delimiter</i>	IN	一個指示值終止的字元字串。這個參數是區分大小寫的。若轉換是 ASCIITEXT 的話，則將被忽略。
字串	<i>table</i>	IN	一個從中讀取檔案記錄的表格變數。
整數	<i>retry</i>	IN	如果檔案無法立刻讀取的話，所要重試的次數。預設值是不重試。
整數	<i>rows</i>	IN	所要讀取至表格中之檔案記錄的最大數目。預設值是讀取所有的記錄，或直到表格已滿。0 表示一直讀取至檔案的尾端。傳回在結果表格中的列數。
整數	<i>startrow</i>	IN	在檔案中，要開始使用的記錄。預設值是從第一個記錄開啓讀取。
整數	<i>columns</i>	OUT	傳回在表格中的欄數。

範例

範例 1:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
    myWait = "1000"  
}%  
@DTWF_READ(myFile, "DELIMITED", ";", myTable, myWait)
```

範例 2:

```
%DEFINE {
    myFile = "c:/private/myfile"
    myTable = %TABLE
    myWait = "0"
    myRows = "0"
    myStartrow = "1"
    myColumns = ""
}%
@DTWF_READ(myFile, "DELIMITED", ";", myTable, myWait, myRows,
            myStartrow, myColumns)
```

範例 3:

```
%DEFINE {
    myFile = "c:/private/myfile"
    myTable = %TABLE
}%
@DTWF_READ(myFile, "ASCITEXT", ";", myTable, myColumns)
@DTW_TB_TABLE(myTable, "BORDER", "")
```


DTWF_REMOVE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

刪除整個檔案。

語法

- @DTWF_REMOVE(filename, retry)

參數

表 57. DTW_REMOVE 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	INOUT	所要刪除的檔案名稱。所傳回的路徑和檔案名稱。
整數	<i>retry</i>	IN	如果檔案無法立刻刪除的話，所要重試的數目。預設值是不重試。

範例

範例 1:

```
%DEFINE myFile = "c:/private/myfile"  
@DTWF_REMOVE(myFile)
```

範例 2:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myWait = "2000"  
%}  
@DTWF_REMOVE(myFile, myWait)
```

DTWF_SEARCH

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

將字串搜尋的結果傳回給表格變數。

語法

- @DTWF_SEARCH(filename, transform, delimiter, table, searchFor, retry, rows, startrow)

參數

表 58. DTWF_SEARCH 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	INOUT	所要搜尋之檔案的名稱。傳回所找到之檔案的名稱和路徑。
字串	<i>transform</i>	IN	檔案的格式： <ul style="list-style-type: none">• ASCIITEXT -使用介於欄值間的新行字元來撰寫表格，並忽略區隔字元參數。• DELIMITED - 使用指定在區隔字元參數中的區隔字元來將表格寫入檔案中。
字串	<i>delimiter</i>	IN	一個指示值終止的字元字串。這個參數是區分大小寫的。若轉換 是 ASCIITEXT 的話，則將被忽略。
字串	<i>table</i>	IN	一個置放結果的表格變數。將會傳回欄 3，如果轉換 是 DELIMITED 的話： <ol style="list-style-type: none">1. 找到有相符者的列。2. 找到有相符者的欄。3. 來自檔案的相符欄。
字串	<i>searchFor</i>	IN	所要搜尋的字元字串。
整數	<i>retry</i>	IN	如果檔案無法立刻搜尋的話，所要重試的次數。預設值是不重試。
整數	<i>rows</i>	IN	要讀取入表格中之最大的列數。預設值是讀取所有的列，或直到表格已滿。指定 0 來讀取置檔案的尾端。在結果表格中的列數是由這個參數傳回的。
整數	<i>startrow</i>	IN	在檔案中開始進行搜尋的記錄。預設值是 1，其將從第一個記錄開始搜尋。

範例

範例 1:

```
%DEFINE {  
  myFile = "c:/private/myfile"  
  myTable = %TABLE
```

```

myWait = "1000"
mySearch = "0123456789abcdef"
@DTWF_SEARCH(myFile, "DELIMITED", ";",
             myTable, mySearch, myWait)

```

範例 2:

```

%DEFINE {
    myFile = "c:/private/myfile"
    myTable = %TABLE
    mySearch = "answer:"
    myWait = "0"
    myRows = "0"
    myStartrow = "1"
}%
@DTWF_SEARCH(myFile, "DELIMITED", ";", myTable,
             mySearch, myWait, myRows, myStartrow)

```

DTWF_UPDATE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

從表格變數中更新檔案的記錄。

語法

- @DTWF_UPDATE(filename, transform, delimiter, table, retry, rows, startrow)

參數

表 59. DTWF_UPDATE 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	IN	其記錄是從表格變數來更新的檔案名稱。
字串	<i>transform</i>	IN	檔案的格式： <ul style="list-style-type: none">• ASCIITEXT - 使用介於欄值間的新行字元來撰寫表格，並忽略區隔字元參數。• DELIMITED - 使用指定在區隔字元參數中的區隔字元來將表格寫入檔案中。
字串	<i>delimiter</i>	IN	一個指示值終止的字元字串。這個參數是區分大小寫的。若轉換是 ASCIITEXT 的話，則將被忽略。
字串	<i>table</i>	IN	檔案記錄從該處開始更新的表格變數。
整數	<i>retry</i>	IN	如果檔案無法立刻被寫入的話，所要重試的次數。預設值是不重試。
整數	<i>rows</i>	IN	從表格所要更新之記錄的最大數目。預設值是更新所有的記錄。0 表示更新所有在檔案中的列。
整數	<i>startrow</i>	IN	要更新的第一個檔案。預設值是 1，其表示在檔案的開始即進行更新。如果值大於檔案中記錄的數目，則該值會被變更為指示在檔案中最後一個記錄的號碼，並傳回錯誤。

範例

範例 1:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
    myWait = "1500"  
    myRows = "2"  
}%  
@DTWF_UPDATE(myFile, "Delimited", "|", myTable, myWait, myRows)
```

範例 2:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
    myStart = "1"  
    myRows = "2"  
%}  
@DTWF_UPDATE(myFile, "AsciiText", "|", myTable, "0", myRows, myStart)
```

DTWF_WRITE

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X	X		X	X	X

將表格變數的目次寫入至檔案中。

語法

- @DTWF_WRITE(filename, transform, delimiter, table, retry, rows, startrow)

參數

表 60. DTWF_WRITE 參數

資料類型	參數	使用	說明
字串	<i>filename</i>	IN	一個寫入表格變數之記錄的檔案名稱。所傳回的路徑和檔案名稱。
字串	<i>transform</i>	IN	檔案的格式： <ul style="list-style-type: none">• ASCIITEXT -使用介於欄值間的新行字元來撰寫表格，並忽略區隔字元參數。• DELIMITED - 使用指定在區隔字元參數中的區隔字元來將表格寫入檔案中。
字串	<i>delimiter</i>	IN	一個指示值終止的字元字串。這個參數是區分大小寫的。若轉換是 ASCIITEXT 的話，則將被忽略。
字串	<i>table</i>	IN	使用來將列匯出至檔案的表格變數。
整數	<i>retry</i>	IN	如果檔案無法立刻被寫入的話，所要重試的次數。預設值是不重試。
整數	<i>rows</i>	IN	所要撰寫之檔案記錄的最大數目。預設值是寫入整個表格。0 表示撰寫所有至檔案尾端的記錄。
整數	<i>startrow</i>	IN	開始撰寫至檔案中的記錄號碼。預設值是 1，其表示在第一個記錄開始進行。如果一個值位於所指定的檔案之後，檔案的最後一列將會與錯誤一起傳回。

範例

範例 1:

```
%DEFINE {  
    myFile = "c:/private/myfile"  
    myTable = %TABLE  
}%  
@DTWF_WRITE(myFile, "DELIMITED", ";", myTable)
```

範例 2:

```
%DEFINE {
    myFile = "c:/private/myfile"
    myTable = %TABLE
}%
@DTWF_WRITE(myFile, "ASCIITEXT", ";", myTable, "5000")
```

範例 3:

```
%DEFINE {
    myFile = "c:/private/myfile"
    myTable = %TABLE
}%
@DTWF_WRITE(myFile, "ASCIITEXT", ";", myTable, "5000", "10", "50")
```

Web 登記函數

Web 登記是一個讓您能輕易地新增、取回和刪除登錄而由 Net.Data 來維護鍵的檔案，您可以在一個單一系統上建立多重 Net.Data Web 登記。每個登記都具有一個名稱，並且可以包含多重登錄。Net.Data 提供函數來維護登記和其所包含之登錄。

- 第168頁的『DTWR_ADDENTRY』
- 第169頁的『DTWR_CLEARREG』
- 第170頁的『DTWR_CREATEREG』
- 第171頁的『DTWR_DELENTY』
- 第172頁的『DTWR_DELREG』
- 第173頁的『DTWR_LISTREG』
- 第174頁的『DTWR_LISTSUB』
- 第175頁的『DTWR_RTVENTRY』
- 第176頁的『DTWR_UPDATEENTRY』

註: 當使用 OS/2 時，對於 *registry*, *registryVariable*, 和 *registryData* 參數。請勿使用星號 (*)

DTWR_ADDENTRY

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X			X	X	X

將登錄新增至 Web 登記。

語法

- @DTWR_ADDENTRY(registry, registryVariable, registryData, index)

參數

表 61. DTWR_ADDENTRY 參數

資料類型	參數	使用	說明
字串	<i>registry</i>	IN	一個要新增登錄的登記名稱。
字串	<i>registryVariable</i>	IN	所要新增之登記 <i>registryVariable</i> 字串部分的值。
字串	<i>registryData</i>	IN	所要新增之登記 <i>registryData</i> 字串部分的值。
字串	<i>index</i>	IN	在所要新增之索引登錄中， <i>registryVariable</i> 字串索引部分的值。這個參數是可選用的。如果有指定的話，將會把一個索引的登錄新增至所指定的登記中。

範例

範例 1:

```
@DTWR_ADDENTRY("Myregistry", "Jones", "http://Advantis.com/~Jones/webproj")
```

範例 2:

```
@DTWR_ADDENTRY("URLLIST", "SMITH", "http://www.software.ibm.com/",  
"WORK_URL,")
```


DTWR_CLEARREG

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X		X	X	X	X

從 Web 登記清除登錄。

語法

- @DTWR_CLEARREG(registry)

參數

表 62. DTWR_CLEARREG 參數

資料類型	參數	使用	說明
字串	<i>registry</i>	IN	所要清除的登記名稱。

範例

範例 1:

```
@DTWR_CLEARREG("Myregistry")
```

DTWR_CREATEREG

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X			X	X	X

建立一個新的 Web 登記。

語法

- @DTWR_CREATEREG(registry, security)

參數

表 63. DTWR_CREATEREG 參數

資料類型	參數	使用	說明
字串	<i>registry</i>	IN	所要建立的登記名稱。
字串	<i>security</i>	IN	要建立登記所要有的保密類型。在 Unix 平台上，預設保密與建立登記的目錄是相同的。您為 3 個保密群組指定保密：使用者、群組、和公用。R 提供讀取許可權、W 提供寫入許可權、而 X 提供執行許可權。例如，若要提供所有 3 個群組完全的權限，請將這個參數指定為 *RWX, *RWX, *RWX。

範例

範例 1:

```
@DTWR_CREATEREG("myRegistry")
```

範例 2:

```
@DTWR_CREATEREG("URLLIST", "*RWX, *RWX, *R")
```

DTWR_DELENTY

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X			X	X	X

從 Web 登記中刪除登錄。

語法

- @DTWR_DELENTY(registry, registryVariable, index)

參數

表 64. DTWR_DELENTY 參數

資料類型	參數	使用	說明
字串	<i>registry</i>	IN	從中除去登錄的登記名稱。
字串	<i>registryVariable</i>	IN	所要除去之登錄的 registryVariable 字串部分的值。
字串	<i>index</i>	IN	在索引的登錄中， registryVariable 字串之索引部分的值。這是一個可選用的參數。如果指定的話，從登記中將索引的登錄除去。

範例

範例 1:

```
@DTWR_DELENTY("Myregistry", "Jones")
```

範例 2:

```
@DTWR_DELENTY("URLLIST", "SMITH", "WORK_URL")
```

DTWR_DELREG

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X			X	X	X

刪除 Web 登記

語法

- @DTWR_DELREG (登記)

參數

表 65. DTWR_DELREG 參數

資料類型	參數	使用	說明
字串	<i>registry</i>	IN	要刪除登記的名稱。

範例

範例 1:

```
@DTWR_DELREG("Myregistry")
```

DTWR_LISTREG

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X			X	X	X

列示整個 Web 登記。

它傳回由使用者所傳送之 OUT 表格變數中的有關登記的資訊。表格變數在被當作參數般傳送至 FUNCTION 區塊以供 LISTREG 登記作業使用之前，已在使用者巨集中定義過了。如果使用者為表格的最大列數而使用 ALL 選項來定義表格變數，則這個作業會列示所有在表格中可用的登記，每個表格列都有一個。換句話說，如果使用者將表格列的最大數目指定為值 X，然後若在所指定的登記中的登錄多於 X 個登錄，則將只有前 X 個登錄會被列示出來且將會傳回一個錯誤碼，以表示因為可用來列示其他登錄的表格列不足，所以只可完成部分的列示。如果值 X 超過在指定之登記中可用登錄的數目的話，則將會列示出所有的登記登錄。在表格中一定是 2 直欄。表格的「直欄」表頭將由「Web 登記」語言環境設定為 "REGISTRY_VARIABLE" 和 "REGISTRY_DATA"。

語法

- @DTWR_LISTREG(registry, registryTable)

參數

表 66. DTWR_LISTREG 參數

資料類型	參數	使用	說明
字串	registry	IN	所要列示的登記名稱。
字串	registryTable	OUT	放置登記之表格變數的名稱。

範例

範例 1:

```
%DEFINE RegistryTable = %TABLE(ALL)

@DTWR_LISTREG("URLLIST", RegistryTable)
```

DTWR_LISTSUB

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
							X

列示在 Web 登記中的立即子鍵。

它傳回有關由使用者傳回在 OUT 表格參數中的登記。在表格變數被當作參數傳送至 LISTSUB 登記作業之前，將會在巨集中對表格變數下定義。如果使用者為表格的最大列數而使用 ALL 選項來定義表格變數，則這個作業會列示在表格中所有可用的登記，每個表格列都有一個。換句話說，如果使用者將表格列的最大數目指定為值 X，然後若在所指定的登記中的登錄多於 X 個登錄，則將只有前 X 個登錄會被列示出來且將會傳回一個錯誤碼，以表示因為可用來列示其他登錄的表格列不足，所以只可完成部分的列示。如果值 X 超過在指定之登記中可用登錄的數目的話，則將會列示出所有的登記登錄。在表格中的直欄數一定是一。表格的直欄表頭將被設定為 "REGISTRY_SUBKEY"。

這個函數只有在「Windows95 系統登記」相容的作業系統上才有效。

語法

- @DTWR_LISTSUB(registry, registryTable)

參數

表 67. DTWR_LISTSUB 參數

資料類型	參數	使用	說明
字串	registry	IN	所要列示的登記名稱。
字串	registryTable	OUT	放置登記之表格變數的名稱。

範例

範例 1:

```
%DEFINE RegistryTable = %TABLE(ALL)

@DTWR_LISTSUB("URLLIST", RegistryTable)
```

DTWR_RTENTRY

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X			X	X	X

從 Web 登記中取回 registryData 字串。

語法

- @DTWR_RTENTRY(registry, registryVariable, registryData, index)
- @DTWR_rRTENTRY(registry, registryVariable, index)

參數

表 68. DTWR_RTENTRY 參數

資料類型	參數	使用	說明
字串	<i>registry</i>	IN	具有所要取回登錄之登記的名稱。
字串	<i>registryVariable</i>	IN	其 registryData 字串被取回之登記登錄之 registryVariable 字串部分的值。
字串	<i>registryData</i>	OUT	傳回符合 registryVariable 之登記之 registryData 字串部分的值。
字串	<i>index</i>	IN	其被傳回之 registryData 字串之索引登錄中的 registryVariable 字串索引部分的值。這是一個可選用的參數。如果有指定的話，將會傳回索引之登錄的 registryData 字串。

範例

範例 1:

```
%DEFINE RegistryData = ""
@DTWR_RTENTRY("Myregistry", "Jones", RegistryData)
```

範例 2:

```
@DTWR_RTENTRY("URLLIST", "SMITH", RegistryData, "WORK_URL")
```

範例 3:

```
@DTWR_rRTENTRY("Myregistry", "Jones")
```

範例 4:

```
@DTWR_rRTENTRY("URLLIST", "SMITH", "WORK_URL")
```

DTWR_UPDATEENTRY

目的

AIX	HP-UX	OS/2	OS/390	OS/400	SCO	SUN	Win NT
X	X	X			X	X	X

為所指定的登記而使用由呼叫者所指定的新值來置換舊有的 registryData 字串值。無法變更 registryVariable 字串。

語法

- @DTWR_UPDATEENTRY(registry, registryVariable, newData, index)

參數

表 69. DTWR_UPDATEENTRY 參數

資料類型	參數	使用	說明
字串	<i>registry</i>	IN	具有所要更新之登錄的登記名稱。
字串	<i>registryVariable</i>	IN	所要更新之登記的 registryVariable 字串部分的值。
字串	<i>newData</i>	IN	所要更新之登記的 registryData 字串部分的新值。
字串	<i>index</i>	IN	在所要更新之索引登錄中， registryVariable 字串索引部分的值。這是一個可選用的參數。如果指定的話，將會更新索引登錄。

範例

範例 1:

```
@DTWR_UPDATEENTRY("Myregistry", "Jones", "http://advantis.com/~Jones/personal")
```

範例 2:

```
@DTWR_UPDATEENTRY("URLLIST", "SMITH", "http://www.software.ibm.com/personal", "WORK_URL")
```

附錄A. DB2 WWW Connection

如果您擁有 DB2 WWW Connection，則您可以執行 Net.Data 版本 1 的現有的應用程式。為充分利用 Net.Data 的特性，建議更新您的應用程式。

%EXEC_SQL

目的

本行會呼叫一個 SQL 區塊。建議您將 SQL 陳述式以函數的方式來呼叫。有關詳細資訊，請參閱 第16頁的『FUNCTION 區塊』。

HTML_INPUT

目的

這個與稱為 INPUT 的 HTML 區塊相同。有關詳細資訊，請參閱 第23頁的『HTML 區塊』。

HTML_REPORT

目的

這個與稱為 REPORT 的 HTML 區塊相同。有關詳細資訊，請參閱 第23頁的『HTML 區塊』。

SQL

目的

這個與在 Net.Data 中以 FUNCTION(DTW_SQL) 呼叫的函數相同。

它可以包含 SQL_REPORT 和 SQL_MESSAGE 陳述式，其亦來自於 DB2 WWW Connection。DB2 WWW Connection 並不支援所指名的 %SQL 區塊。

範例

這是一個 DB2 WWW Connection 巨集的範例。

```
%SQL{
UPDATE $(dbtbl) SET URL='$(URL)' WHERE ID=$(ID)
%SQL_MESSAGE{
100: "<B>所選取的 URL 已不再存在於表格中
</B >。" : Continue
%}
%}

%HTML_INPUT{
<HTML>
...
%EXEC_SQL
</HTML>
%}

%HTML_REPORT{
<HTML>
...
</HTML>
%}
```

在 Net.Data 中相同的巨集如下所示：

```
%FUNCTION(DTW_SQL) URLQuery(){
UPDATE $(dbtbl) SET URL='$(URL)' WHERE ID=$(ID)
%MESSAGE{
100: "<B>所選取的 URL 已不再存在於表格中
</B >。" : Continue
%}
%}

%HTML(INPUT){
<HTML>
...
@URLQuery
</HTML>
%}

%HTML(REPORT){
<HTML>
...
</HTML>
%}
```

SQL_MESSAGE

目的

這與 Net.Data MESSAGE 陳述式相同。有關範例請參閱第37頁的『MESSAGE 區塊』。

SQL_REPORT

目的

這與 Net.Data REPORT 陳述式相同。有關範例，請參閱第40頁的『REPORT 區塊』。

SQL_CODE

目的

它來自 DB2 WWW Connect，且 Net.Data 將支援它以便能夠與其相容。它相等於 第 60頁的『RETURN_CODE』。

附錄B. Net.Data 訊息和訊息碼

當 Net.Data 偵測到不尋常的狀況，將會提供這些訊息。

-1002 函數 *function*。無法配置記憶體。

解說：伺服器無法處理來自 Net.Data 的儲存體要求。

使用者回應：請確定伺服器擁有足夠的記憶體。

-1001 函數 *function*。內部程式碼 *code*。

解說：一個對內部函數的呼叫失敗。這是一個 Net.Data 的內部錯誤。

使用者回應：請將問題告知您的軟體服務代表。

1000 函數 *function*。找不到函數。

解說：在函數呼叫上所要求的函數並不是 Net.Data 所支援的函數。

使用者回應：請確定所指定的函數是位於它被呼叫的巨集檔案中。函數名稱必須是您使用來呼叫函數的名稱。請檢查 FUNCTION 區塊的語法。

1001 函數 *function_name*。參數 *parm_name* 包含一個空值。

解說：某個輸入參數包含一個 NULL 值。

使用者回應：請確定在參數尚未被傳送至函數之前，它已被定義過了且不是一個空值。

1002 函數 *function_name*。參數 *parm_name* 包含一個空字串。

解說：某個輸入參數包含一個由空值終止字元所組成的字串值。

使用者回應：請確定所指定的參數包含一個非空值。

1003 函數 *function*。所傳遞的參數數目並不正確。

解說：在函數呼叫上所傳遞參數的數目要不是超過所允許的最大數目，就是少於被呼叫之函數必要的最小數目。

使用者回應：請檢查函數語法，並確定您傳遞了所有必要的參數，但不多於所指定的最大數目。

1004 函數 *function*。參數 *parm_name* 不是一個表格。

解說：在函數呼叫上被傳遞的參數應該為 Net.Data 巨集表格變數，但卻使用了字串變數。

使用者回應：請確定變數在 DEFINE 陳述式或區塊中被定義為一個 TABLE 變數。

1005 函數 *function*。參數 *parm_name* 不是一個字串。

解說：在函數呼叫上傳遞的參數應為 Web 巨集字串變數之必要參數，但卻使用了表格變數。

使用者回應：請確定您並未在 DEFINE 陳述式中將此變數定義為 TABLE 變數的。

1006 函數 *function*。參數 *parm_name* 不是一個輸出參數。

解說：在參數的函數呼叫上傳遞了一個文數字的字串，但應傳遞一個輸出參數。

使用者回應：請勿替輸出參數指定任何輸入值。參數的類型可能需要變更為 INOUT。

1007 函數 *function*。參數 *parm_name* 包含的值無效。

解說：可能存在下列其中的一個狀況：

- 所傳遞的值超過所支援的最大值。
- 所傳遞的值少於所支援的最小值。
- 所傳遞的值不是所支援的選項之一。
- 所傳遞之表格的列或直欄的值小於或等於零。

使用者回應：請確定值並沒有超過範圍或無效。

1008 函數 *function*。參數值 *parm_value* 超過表格的界限。

解說：可能存在下列其中的一個狀況：

- 程式試圖修改表格的列或直欄值，但是所接收到之列或直欄的值卻小於 0，或大於表格中所允許之列的最大數目。
- 某個列或直欄值被接收來作為內建函數的輸入，但所收到的值卻小於 0，或大於目前在表格中列或直欄的數目。

使用者回應：請確定所指定的值並沒有小於 0 或大於目前表格中的列數。

1009 函數`function`。變數字串`string`格式不正確。

解說： 由「系統」或 Perl 程式所返回之資料的語法不正確。可能存在下列其中的一個狀況：

- 找不到等號。
- 找不到開始的引號。
- 找不到終止的引號。
- 找不到介於值之間的區隔符號。

使用者回應： 請檢查由函數所傳回的資料是否有語法上的錯誤。

1010 函數`function`。並不是所有要求的資料均被傳回。

解說： 一個指定為輸出參數的表格，但是由語言環境所返回之資料的列數大於表格所允許之列數的最大值。資料會一直被寫入表格中直到表格額滿為止，且會捨棄剩餘的資料。

使用者回應： 您可以忽略已捨棄的資料，或增加表格大小並再次執行函數。

2000 函數`function`。所要求的檔案`filename` 找不到。

解說： 一般檔案介面內建函數在它所能尋找的目錄中找到所指定的檔案。

使用者回應： 請確定檔案是位於由起始設定檔案中之 `FFI_PATH` 陳述式所指定的路徑中。

2001 函數`function`。所要求的檔案`filename` 無法在所指定的模式中開啓。

解說： 一般檔案介面內建函數無法開啓所指定的檔案，因為該檔案被這個或另一個處理所使用，且在所指定的模式中無法共用。

使用者回應： 請確定另一個處理並未鎖定該檔案。

2002 函數`function`。所要求的檔案`filename` 尚未開啓。

解說： 一般檔案介面內建函數無法關閉所指定的檔案，因為該檔案並未由這個巨集呼叫所開啓。

使用者回應： 該檔案必須由開啓它的巨集來關閉。將會失去所作的變更。

2003 函數`function`。試圖讀取一個具有超過所支援之最大位元組數之資料的列。

解說： 一般檔案介面內建函數無法將資料列讀入表格變數中，因為在該列中的位元組數目超過所支援之位元組數的最大數目。

使用者回應： 表格太大，`Net.Data` 無法處理。

2004 函數`function`。在 `FFI_PATH` 中所指定的路徑超過所支援之位元組數的最大數目。

解說： 一般檔案介面內建函數試圖尋找檔案，但在 `FFI_PATH` 架構檔案變數中，發現一個大於最大支援之位元組數目的 (4095個位元組)的路徑。

使用者回應： 將 `FFI_PATH` 陳述式縮短為 `Net.Data` 為現行應用程式所需要的那些目錄即可。

2005 函數`function`。System error

解說： 對系統函數的呼叫失敗。這個報表至 `Net.Data` 的內部錯誤可能會需要使用者的互動，或它可能是一個 `Net.Data` 所不適合處理的暫時性系統錯誤。如果這個問題仍然存在，請將問題告知您的軟體服務代表。

使用者回應： 請檢查您的架構然後再試一次。如果這個問題仍然存在，請將問題告知您的軟體服務代表。

2006 函數`function`。所要求的檔案`filename` 無法在所指定的模式中存取。

解說： 一般檔案介面內建函數 無法存取所指定的檔案，因為該檔案被這個或另一個處理所使用，且在所指定的模式中無法共用。

使用者回應： 請將使用檔案的處理終止並重試一次。請考慮將 `RETRY` 值指定為當函數被呼叫時，如果檔案正在使用中則自動重試。

3001 函數`function`。登記`registry_name`已經存在。

解說： Web 登記內建函數無法建立 Web 登記，因為所指定的登記已經存在。

使用者回應： 請使用另一個 Web 登記名稱。

3002 函數`function`。登記`registry_name`正由另一個處理所使用或並未存在。

解說: Web 登記內建函數因下列這些狀況中的某一項而無法刪除所指定的登記:

- 此登記正由另一個處理所使用。
- 找不到登記。

使用者回應: 此登記正由另一個處理所使用, 請在關閉處理後重試。

3003 函數`function`。登記`registry_entry` 已經存在。

解說: Web 登記內建函數無法將登錄新增至所指定的登記中, 因為所指定的登錄已經存在。

使用者回應: 無法在 Web 登記中重複登錄。請修改登錄並再次提出函數, 或使用現存的登錄。

3004 函數`function`。登記登錄`registry_entry` 找不到。

解說: Web 登記內建函數無法從所指定的登記將登錄除去或取回, 因為所指定的登錄並未存在。

使用者回應: Net.Data 找不到所要求的登記登錄。

3005 函數`function`。登記`registry_name`找不到。

解說: Web 登記內建函數 無法使用所指定的登記, 因為找不到該登記。

使用者回應: 如果登記並不存在的話, 請將其建立。

3006 函數`function`。在登記`registry_name` 中的路徑並不存在。

解說: Web 登記內建函數 無法建立所指定的登記, 因為在登記名稱中並未指定路徑。

使用者回應: 當您建立登記時請指定路徑。

3007 函數`function`。您並未被授權執行所要求的登記作業。

解說: Web 登記內建函數 無法完成所指定的作業, 因為要求者對所指定之登記沒有適當的權限。

使用者回應: 請變更您 Web 登記函數中的安全性參數來授權給該作業。

3008 函數`function`。登記`registry_name` 無法建立。

解說: Web 登記內建函數無法建立所指定的登記, 但原因不明。

使用者回應: 請檢查您的架構, 然後再試一次。

4000 函數`function`。參數`parm_name`並不是一個整數數字或者它太大了。

解說: 可能存在下列其中的一個狀況:

- 某個輸入參數包含一個非整數的值。
- 某個輸入參數包含一個值, 該值大於所支援之最大值 999,999,999。
- 輸出無法以一個整數來表示。

使用者回應: 請確定值並沒有超過範圍或無效。

4001 函數`function`。參數`parm_name`不是一個有效的數。

解說: 可能存在下列其中的一個狀況:

- 某個輸入參數包含一個不是有效數字格式的值。
- 某個輸入參數包含一個值, 該值指定一個在所支援之 -999,999,999 到 +999,999,999 範圍之外的指數。

使用者回應: 請確定值並沒有超過範圍或無效。

4002 函數`function`。算數上溢或下溢。

解說: 算數運算的結果其指數超過所支援之 -999,999,999 到 +999,999,999 範圍之外。

使用者回應: 請確定該值在所支援的範圍之內。

5000 函數`function`。EXEC 陳述式是空的。

解說: 在函數 區塊之 EXEC 陳述式之內所指定的字串所包含的只有空白字元。

使用者回應: 請指定一個不是只有空白字元的字串。

6000 函數`function`。並不是在函數區段中指定 EXEC。

解說: EXEC 陳述式並不是在正被呼叫之函數的函數區塊中指定的。

使用者回應: 在函數區塊中新增 EXEC 陳述式。

附錄C. Net.Data 平台參考手冊

並非每一種平台都支援所有的 Net.Data 特性。本章節將顯示給您平台所支援的特性為何。一個**X** 表示有支援該特性。

表 70. Net.Data 語言環境

語言環境	AIX	HP	OS/2	OS/390	OS/400	SCO	SUN	Win NT
Java Applets	X	X	X	X		X	X	X
Java 應用程式	X		X			X	X	X
ODBC	X	X	X	X		X	X	X
Oracle	X					X		
Perl	X		X	X	X	X	X	X
Rexx	X		X	X	X	X	X	X
SQL	X	X	X	X	X	X	X	X
Sybase	X					X		X
系統	X		X	X	X	X		X

表 71. Net.Data 儲存程序資料類型

資料類型	AIX	HP	OS/2	OS/390	OS/400	SCO	SUN	Win NT
BLOB	X	X	X			X	X	X
CHAR	X	X	X	X	X	X	X	X
CLOB	X	X	X			X	X	X
DATE	X	X	X		X	X	X	X
DBCLOB	X	X	X			X	X	X
DECIMAL				X				
DOUBLE	X	X	X		X	X	X	X
DOUBLEPRECISION	X	X	X		X	X	X	X
FLOAT	X	X	X	X	X	X	X	X
INTEGER	X	X	X	X	X	X	X	X
GRAPHIC	X	X	X	X	X	X	X	X
LONGVARCHAR	X	X	X		X	X	X	X
LONGVARGRAPHIC	X	X	X		X	X	X	X
SMALLINT	X	X	X	X	X	X	X	X
TIME	X	X	X		X	X	X	X
TIMESTAMP	X	X	X		X	X	X	X
VARCHAR	X	X	X	X	X	X	X	X
VARGRAPHIC	X	X	X	X	X	X	X	X

表 72. Net.Data 變數

變數	AIX	HP	OS/2	OS/390	OS/400	SCO	SUN	Win NT
ALIGN	X	X	X	X	X	X	X	X
DATABASE	X	X	X		X	X	X	X
DB2PLAN				X				
DB_CASE	X	X	X	X	X	X	X	X
DB2SSID				X				
DTW_APPLET_ALTTEXT	X	X	X	X		X	X	X
DTW_CURRENT_FILENAME	X	X	X	X	X	X	X	X
DTW_CURRENT_LAST_MODIFIED	X	X	X	X	X	X	X	X
DTW_DEFAULT_REPORT	X	X	X	X	X	X	X	X
DTW_MACRO_FILENAME	X	X	X	X	X	X	X	X
DTW_MACRO_LAST_MODIFIED	X	X	X	X	X	X	X	X
DTW_MP_PATH	X	X	X	X	X	X	X	X
DTW_MP_VERSION	X	X	X	X	X	X	X	X
DTW_PRINT_HEADER	X	X	X	X	X	X	X	X
DTW_SAVE_TABLE_IN	X	X	X	X	X	X	X	X
DTW_SET_TOTAL_ROWS	X	X	X	X	X	X	X	X
DTW_HTML_TABLE	X	X	X	X	X	X	X	X
LOCATION				X				
LOGIN	X	X	X		X	X	X	X
N_columnName	X	X	X	X	X	X	X	X
N1, N2, Nn	X	X	X	X	X	X	X	X
NLIST	X	X	X	X	X	X	X	X
NUM_COLUMNS	X	X	X	X	X	X	X	X
PASSWORD	X	X	X		X	X	X	X
RETURN_CODE	X	X	X	X	X	X	X	X
ROW_NUM	X	X	X	X	X	X	X	X
RPT_MAX_ROWS	X	X	X	X	X	X	X	X
SHOWSQL	X	X	X	X	X	X	X	X
SQL_CODE	X	X	X	X	X	X	X	X
TOTAL_ROWS	X	X	X	X	X	X	X	X
TRANSACTION_SCOPE	X	X	X	X	X	X	X	X
V_columnName	X	X	X	X	X	X	X	X
Vn	X	X	X	X	X	X	X	X
VLIST	X	X	X	X	X	X	X	X
START_ROW_NUM	X	X	X		X		X	X

表 73. Net.Data 函數

函數	AIX	HP	OS/2	OS/390	OS/400	SCO	SUN	Win NT
DTW_ADDQUOTE	X	X	X	X	X	X	X	X
DTW_DATE	X	X	X	X	X	X	X	X
DTW_GETENV	X	X	X	X	X	X	X	X
DTW_GETINIDATA	X	X	X	X	X	X	X	X
DTW_HTMLENCODE	X	X	X	X	X	X	X	X
DTW_QHTMLENCODE	X	X	X	X	X	X	X	X
DTW_SETENV	X	X	X	X	X	X	X	X
DTW_TIME	X	X	X	X	X	X	X	X
DTW URLESCSEQ	X	X	X	X	X	X	X	X
DTW_ADD	X	X	X	X	X	X	X	X
DTW_DIVIDE	X	X	X	X	X	X	X	X
DTW_DIVREM	X	X	X	X	X	X	X	X
DTW_FORMAT	X	X	X	X	X	X	X	X
DTW_INTDIV	X	X	X	X	X	X	X	X
DTW_POWER	X	X	X	X	X	X	X	X
DTW_SUBTRACT	X	X	X	X	X	X	X	X
DTW_ASSIGN	X	X	X	X	X	X	X	X
DTW_CONCAT	X	X	X	X	X	X	X	X
DTW_DELSTR	X	X	X	X	X	X	X	X
DTW_INSERT	X	X	X	X	X	X	X	X
DTW_LASTPOS	X	X	X	X	X	X	X	X
DTW_LENGTH	X	X	X	X	X	X	X	X
DTW_LOWERCASE	X	X	X	X	X	X	X	X
DTW_MULTIPLY	X	X	X	X	X	X	X	X
DTW_POS	X	X	X	X	X	X	X	X
DTW_REVERSE	X	X	X	X	X	X	X	X
DTW_STRIP	X	X	X	X	X	X	X	X
DTW_SUBSTR	X	X	X	X	X	X	X	X
DTW_TRANSLATE	X	X	X	X	X	X	X	X
DTW_UPPERCASE	X	X	X	X	X	X	X	X
DTW_DELWORD	X	X	X	X	X	X	X	X
DTW_SUBWORD	X	X	X	X	X	X	X	X
DTW_WORD	X	X	X	X	X	X	X	X
DTW_WORDINDEX	X	X	X	X	X	X	X	X
DTW_WORDLENGTH	X	X	X	X	X	X	X	X
DTW_WORDPOS	X	X	X	X	X	X	X	X

表 73. Net.Data 函數 (繼續)

函數	AIX	HP	OS/2	OS/390	OS/400	SCO	SUN	Win NT
DTW_WORDS	X	X	X	X	X	X	X	X
DTW_TB_DLIST	X	X	X	X	X	X	X	X
DTW_TB_DUMPH	X	X	X	X	X	X	X	X
DTW_TB_DUMPV	X	X	X	X	X	X	X	X
DTW_TB_HTMLENCODER	X	X	X	X	X	X	X	X
DTW_TB_INPUT_CHECKBOX	X	X	X	X	X	X	X	X
DTW_TB_INPUT_RADIO	X	X	X	X	X	X	X	X
DTW_TB_INPUT_TEXT	X	X	X	X	X	X	X	X
DTW_TB_LIST	X	X	X	X	X	X	X	X
DTW_TB_SELECT	X	X	X	X	X	X	X	X
DTW_TB_TABLE	X	X	X	X	X	X	X	X
DTW_TB_TEXTAREA	X	X	X	X	X	X	X	X
DTWF_APPEND	X	X	X	X		X	X	X
DTWF_CLOSE	X	X	X	X		X	X	X
DTWF_DELETE	X	X	X	X		X	X	X
DTWF_INSERT	X	X	X	X		X	X	X
DTWF_OPEN	X	X	X	X		X	X	X
DTWF_READ	X	X	X	X		X	X	X
DTWF_REMOVE	X	X	X	X		X	X	X
DTWF_SEARCH	X	X	X	X		X	X	X
DTWF_UPDATE	X	X	X	X		X	X	X
DTWF_WRITE	X	X	X	X		X	X	X
DTWR_ADDENTRY	X	X	X			X	X	X
DTWR_CLEARREG	X	X	X			X	X	X
DTWR_CREATEREG	X	X	X			X	X	X
DTWR_DELENTY	X	X	X			X	X	X
DTWR_DELREG	X	X	X			X	X	X
DTWR_LISTREG	X	X	X			X	X	X
DTWR_LISTSUB	X	X	X			X	X	X
DTWR_RTVENTRY	X	X	X			X	X	X
DTWR_UPDATEENTRY	X	X	X			X	X	X

表 74. Net.Data 介面

介面類型	AIX	HP	OS/2	OS/390	OS/400	SCO	SUN	Win NT
CGI	X	X	X	X	X	X	X	X
Internet 連線 API	X		X					X
Internet 伺服器 API								X

表 74. Net.Data 介面 (繼續)

介面類型	AIX	HP	OS/2	OS/390	OS/400	SCO	SUN	Win NT
Netscape API		X					X	X
現場連線	X	X	X				X	X

附錄D. Net.Data for AIX

Net.Data 是用來取代 DB2 WWW Connection。有關 AIX 的詳細說明，請參閱隨附於 Net.Data 的 README 檔案。README 檔案包括這些資訊：

- 基本需求
- 安裝
- 架構
- 解除安裝

當在 AIX 平台建立語言環境時，您需要執行某些特定的步驟，因為 AIX 需要載入共用程式庫。在 AIX 上，需要語言環境來提供由 Net.Data 所呼叫的常式及傳回語言環境介面的常式的位址，如 `dtw_initialize()` 和 `dtw_execute()`。

Net.Data 使用 `dtw_fp` 結構來從 AIX 的語言環境中取回指向語言環境介面常式的指標，其格式如下：

```
typedef struct dtw_fp {
    int (* dtw_initialize_fp)(); /* dtw_initialize function pointer */
    int (* dtw_execute_fp)(); /* dtw_execute function pointer */
    int (* dtw_getNextRow_fp)(); /* dtw_getNextRow function pointer */
    int (* dtw_cleanup_fp)(); /* dtw_cleanup function pointer */
} dtw_fp_t;
```

當載入共用程式庫時，Net.Data 會將這個結構當成一個在 `dtw_getFp()` 常式中的參數傳遞至語言環境。

`dtw_fp` 結構為被傳遞來當作唯一的參數。這個結構包含每個支援介面的欄位，語言環境必須是設定這些欄位。如果語言環境有提供指定的介面，則它會將欄位設定為該介面的函數指標。如果它並未提供所指定的介面，則它會將欄位設定為 `NULL`。在程式模版中的 `dtw_getFp()` 常式將呈現這個常式的正確實作方式。

當載入共用程式庫時，為了讓 Net.Data 取得對這個常式的指標，`dtw_getFp` 常式必須是在共用程式庫的匯出檔案中所指定的第一個登錄點。稱為 `dtwsampshr.o` 的一個支援所有可用之語言環境介面常式的程式庫之匯出檔案的樣本看起來類似於：

```
#!dtwsampshr.o
dtw_getFp
dtw_initialize
dtw_execute
dtw_getNextRow
dtw_cleanup
```

如果在您的 AIX 系統上有許多對 REXX 語言環境呼叫，則請考慮將 `RXQUEUE_OWNER_PID` 設定為 0。有許多對 REXX 語言環境呼叫的巨集，可輕易地產生出很多處理以及交換系統資源。例如，如果您的 Web 伺服器正在每秒處理 2 個要求。在一個呼叫中將 20 列傳回給在 ROW 區塊中之 REXX 函數的巨集，會強迫伺服器嘗試每秒帶出 80 (20 (列數) * 2 (REXX 要求) * 2 (HTTP 要求)) 個處理，而這將遠超過許多伺服器所能處理的。

將 `RXQUEUE_OWNER_PID` 設定為 0，使之在每秒有不多於兩個的處理，而這即使對一個小 AIX 方框而言，也是可處理的。使用 `DTW_rSETENV` 來設定此值：

```
@DTW_rSETENV("RXQUEUE_OWNER_PID", "0")
```

上述之範例(即呼叫在 ROW 區塊中的 REXX 語言環境)的另一個方式為使用表格變數來將整個擷取的表格傳送至 REXX 函數，讓處理發生在對語言環境的單一呼叫中。

附錄E. Net.Data for OS/2

Net.Data 是用來取代 DB2 WWW Connection 。有關 OS/2 的詳細資訊，請參閱隨附於 Net.Data 的 README 檔案。README 檔案包括這些資訊：

- 基本需求
- 安裝
- 架構
- 解除安裝

附錄F. Net.Data for OS/390

此附錄是供給負責程式安裝和維護的系統程式設計師使用。它包含了下列的各項資訊：

- 軟體需求
- 架構與安裝概觀
- 安全性與身份驗證的建議
- 架構存取 DataJoiner 之 Net.Data 的步驟

參閱「Net.Data for OS/390 程式目錄」可取得有關安裝和架構步驟的明細。

如果您安裝使用「MVS Custom-Built Product Delivery Offering (CBPDO*) (5751-CS3)」的 Net.Data，則請使用由 CBPDO 磁帶所提供的軟體拷貝程式目錄。您的 CBPDO 包含此產品的升級版之禁止軟體拷貝服務程式規劃 (PSP)。

若要判斷在 OS/390 上是否支援特定之 Net.Data 特性的話，請參閱 第189頁的『附錄 C. Net.Data 平台參考手冊』。

必要的軟體

若要安裝並執行 OS/390 的 Net.Data，則您需要所有下列的軟體：

- 具有 PTFs UN88194、UN88461、UN87133、和 UN88416 的 Language Environment/370 版本 1.5
- RACF 版本 2.1 或具有必須支援之相等安全性產品
- 下列其中一項作業系統產品：
 - 具有 OpenEdition MVS System Services 1.1.0 的 MVS/ESA 版本 5.2.2
 - 具有 OpenEdition MVS System Services 1.2.0 的 MVS/ESA 版本 5.2.2
 - 具有 OpenEdition MVS System Services 1.3.0 的 MVS/ESA 版本 5.2.2
 - OS/390 版次 1 或更新的版次
- 下列其中一項伺服器產品：
 - Internet Connection Secure Server for OS/390 版本 2 或更新的版本
 - Internet Connection Server for MVS/ESA 版本 1.1
- 下列其中一項資料庫產品：
 - DB2 for OS/390 版本 5
 - DB2 for MVS/ESA 版本 4

可選用的軟體

對 IMS 應用程式和資料存取而言：IMS/ESA 版本 4 或更新的版本

若要經由 DataJoiner 來存取不同種類的資料來源：具有 PTF U447593 的 DataJoiner for AIX 版本 1.2 或 DataJoiner for HP-UX 版本 1.1

安裝與架構概觀

這個章節提供一個 Net.Data for OS/390 之安裝與架構步驟的概觀以供規劃。它並不包含完整的程序。有關安裝與架構程序的明細，請參閱「Net.Data for OS/390 程式目錄」。

1. 安裝 Net.Data :
 - a. 安裝 Net.Data 軟體。
 - b. 自行設訂 Net.Data 起始設定檔案。
 - c. 安裝 Net.Data 文件和圖示。
 - d. 自行設訂 Net.Data 併入檔案。
 - e. 設定 REXX、SQL 和 ODBC Net.Data 語言環境。
2. 執行安裝驗證處理：
 - a. 建立並載入 Net.Data 樣本 DB2 表格。
 - b. 安裝樣本 DB2 儲存 程序。
 - c. 自行設定 Net.Data 樣本巨集。
 - d. 連結 Net.Data DBRM。
 - e. 準備測試環境，包括設定 Web 伺服器。
 - f. 執行 Net.Data 樣本巨集。

安全性與身份驗證

您可以對 Net.Data for OS/390 使用三種身份驗證方法的類型。所有這三種類型都將根據「IBM Internet Connection 伺服器」在它的架構檔中所提供作為指引的選項：

以從屬站為基礎的身份驗證：

在從屬站中使用者被提示要輸入使用者 ID 和通行碼。在將控制權傳遞至 Net.Data 以及任何如資料庫的資源之前要先驗證使用者 ID 和通行碼。

以伺服器為基礎的身份驗證：

使用了在 Web 伺服器上的使用者 ID，且並未提示使用者輸入使用者 ID 或通行碼。並不建議您使用這個選項且應謹慎使用。

代理身份驗證：

代理使用者 ID 是被配置為一組專為身份驗證的使用者群組，並管理如資料庫等資源的存取。

請勿在 Net.Data 架構中使用 Net.Data DATABASE、LOGIN 以及 PASSWORD 選項，也請勿在 OS/390 平台中使用巨集檔案。使用上述的三個方式之一來配置使用者 ID (LOGIN)。

將 Net.Data 架構為存取 DataJoiner

您可以使用具有 DataJoiner 的 Net.Data for OS/390 來存取遠端資料庫，如 DB2/6000、Oracle、以及 Sybase。這個部份是說明要如何將您的系統架構成與具有 PTF U447593 的 DataJoiner for AIX 版本 1.2 或 DataJoiner for HP-UX 版本 1.1 一起使用。

架構步驟：

1. 在通信資料庫 (CDB) 中輸入所需的資訊以便對 DataJoiner 執行遠端連接。CDB 上的資訊是在 *IDB2 Installation Guide*。
2. 使用 BIND PACKAGE 命令來將 Net.Data DBRM 連結至安裝 DataJoiner 的遠端位置。
3. 使用 BIND PLAN 命令來將 Net.Data DBRM 連結至 DB2。使用 PKLIST 選項來併入在遠端位置所建立的套裝軟體。
4. 在位於 Web 伺服器的文件根目錄中的 Net.Data 架構檔中，將 LOCATION 變數修改成一個對 SQL 函數的輸入變數。新的 DTW_SQL 環境陳述式如下所示：

```
ENVIRONMENT (DTW_SQL) dtwsq1 (IN LOCATION)
```

使用 DataJoiner 來存取遠端資料的 Net.Data 巨集，必須指定一個 LOCATION 的值。在這個範例中，Net.Data 巨集經由 DataJoiner 來查詢遠端的資料庫：

```
%{ ***** 定義區塊 ***** %}
%DEFINE {
    DB2SSID="NDA1"
    LOCATION="QMFDJ00"
    DTW_DEFAULT_REPORT="YES"
}%

%{ ***** 函數定義區塊 ***** %}
%FUNCTION(DTW_SQL) selectall() {
    SELECT * FROM $(tabnam)
}%

%{ ***** HTML 區塊：Table_Input ***** %}
%HTML(Table_Input) {
<Title>DJ 測試 #1</Title>
<Body>
<h1 align=center>表格選擇</h1>
<br>
<form method="post" action="Column_Output">
<p>輸入表格名稱：<input type="text" name="tabnam"></p>
<p><input type="submit"></p>
</form>
</Body>
}%

%{ ***** HTML 區塊：Column_Output ***** %}
%HTML(Column_Output) {
<Title>DJ 測試 #1</Title>
<Body>
@selectall()
</Body>
}%
```

程式材料

IBM 程式是由程式號碼和特性號碼所定義。Net.Data 的程式號碼是 5655-DB2。

基本的「機器可讀取材料」為在基本授權程式和特性程式碼之下提供的，也是使用產品時所必須。可選用的「機器可讀取材料」是在個別特性程式碼之下提供的，它是可訂購的，但在產品運作上並不是必要的。

程式宣告材料說明由 Net.Data 所支援的特性。若您尚未接到這份副本的話，請聯絡您的 IBM 服務代表洽詢相關資訊。

機器可讀取材料

這個程式的分送媒體是 9 個磁軌的磁帶以（6250 BPI 來撰寫）、3480 磁帶匣、3590 磁帶匣或 4mm 磁帶匣。不論磁帶或磁帶匣都包含所有安裝時所需的程式和資料。它是使用 SMP/E 來安裝的，且其格式為 SMP/E RELFILE。

第202頁的表 75 會說明磁帶或磁帶匣。

註：如果您安裝使用 MVS Custom-Built Product Delivery Offering (CBPDO*) (5751-CS3) 的 Net.Data，則在這個圖表的某些資訊可能是無效的。有關其真正的值，請查閱 CBPDO 文件。

表 75. 基本材料：程式磁帶

媒體	特性號碼	實體容體	外部標籤識別字	VOLSER
6250 磁帶	5841	1 之 1	Net.Data base	24C110
3480 磁帶	5842	1 之 1	Net.Data base	24C110
4mm 卡匣	6290	1 之 1	Net.Data base	24C110

Net.Data FMID

Net.Data 是由下列的 FMID 所組成：

H24C110

附錄G. Net.Data for OS/400

除非新版中另外指出，否則本版將引用於：

- IBM Operating System/400 (Program 5716-SS1), 版本 3 版次 7 修訂版 0；
- IBM TCP/IP Connectivity Utilities for AS/400 (Program 5716-TC1)，版本 3 版次 7 修訂版 0；
- IBM Operating System/400 (Program 5763-SS1)，版本 3 版次 2 修訂版 0；
- IBM TCP/IP Connectivity Utilities for AS/400 (Program 5763-TC1)，版本 3 版次 2 修訂版 0；

以及所有後續的版次和修訂版。

請務必配合產品層次，使用適當的版本。

Net.Data for OS/400 簡介

Net.Data 可以取代 DB2 WWW Connection。如果您仍在使用 DB2 WWW Connection，而希望移轉到 Net.Data，請參閱第216頁的『從 DB2 WWW Connection 移轉到 Net.Data』。

Net.Data 與後續版本的 DB2 WWW Connection 相容，它是建構在強大的資料庫存取功能，以及 DB2 WWW Connection 的報表製作功能上。其功能經過大幅增強之後，現在已成為理想的 Web 開發環境，可以建立簡單的動態 Web 畫面，也可以建立複雜的 Web 應用程式。

Net.Data 將免費檢附在下列各版的 OS/400 作業系統上：

- 版本 3 版次 7
- 版本 3 版次 2

與 DB2 WWW Connection 版本 1. A PTF for V3R2 一起出廠的 OS/400 作業系統的 V3R2 版次，可以啓用 Net.Data。

V3R2 和 V3R7 所強調的重點，是 Internet Connection for AS/400；它可以將 AS/400 轉換成 Web 伺服器。這個 Internet Connection 支援提供了許多新的特性，可讓您輕鬆的在 AS/400 上，用 Web 存取資料和應用程式，而 Net.Data 就是其中一個新的 Internet Connection for AS/400 特性。

Internet Connection for AS/400 中含有 Net.Data，是 OS/400 V3R7 TCP/IP 軟體的標準組件。(OS/400 V3R2 使用者必須有 PTF 才能安裝 Net.Data)。您就不需要再多買其他的東西，同時，也不需要下載和安裝任何 Net.Data 軟體。

您所需要的 AS/400 TCP/IP 軟體會檢附在 OS/400 中，但它不一定要安裝。下述軟體應該安裝在您的系統上：

- V3R7
 - IBM OS/400 V3R7 (5716-SS1)
 - IBM TCP/IP Connectivity Utilities/400 V3R7 (5716-TC1)
- V3R2
 - IBM OS/400 V3R2 (5763-SS1)
 - IBM TCP/IP Connectivity Utilities/400 V3R2 (5763-TC1)

如需 Net.Data for the OS/400 的最新資訊，請進入下述 URL：

<http://www.as400.ibm.com/netdata>

一般的 OS/400 概念

在您使用 Net.Data 之前，必須先了解一些基本概念。尤其是您所用的 HTTP 伺服器 and 整合式檔案系統。

HTTP 伺服器

Net.Data 在 AS/400 上，只能被當作一般閘道介面 (CGI) 程式加以呼叫。由於本身是 CGI 程式，因此 Net.Data 是採用標準 CGI 介面：從屬站 (亦即瀏覽器使用者) 資料 (如格式或 URL 資料) 是由 CGI 程式從 stdin 接收，而由 CGI 所產生的輸出資料，則藉由寫入 stdout 而傳回該從屬站。

如果您要將 Net.Data 和 HTTP 伺服器一起執行，主要需擔心下列兩點：

- HTTP 伺服器是否支援 CGI 程式
- HTTP 伺服器是否使用標準 CGI 介面來支援 CGI 程式

如果上述問題的答案為是，則 Net.Data 可以與您所選的 HTTP 伺服器一起使用。

最後還要注意一點。在下述各節當中，我們將告訴您，如何利用 Internet Connection for AS/400 來啟動和執行 Net.Data，前者是 V3R2 和 V3R7 (以及後續版次) 之 OS/400 作業系統標準組件的 HTTP 伺服器。如果您不使用 Internet Connection for AS/400，便需要決定如何用 HTTP 伺服器，來呼叫 Net.Data 作為 CGI 程式使用。

整合式檔案系統

在使用 Net.Data 時所能指定的所有物件參照，都假設是以**路徑名稱**的格式來顯示。路徑名稱可以讓系統知道如何尋找物件。路徑名稱是由連續的目錄名稱所組成，後面再接著該物件的名稱。個別目錄和物件名稱，是由斜線 (/) 字元加以分隔；例如：

`/directory1/directory2/file`

路徑名稱與**整合式檔案系統**密不可分。

整合式檔案系統是 OS/400 的一部份，它可以支援類似個人電腦和 UNIX 作業系統的資料流輸入/輸出和儲存體管理，並針對存放在 AS/400 上的所有資訊，而提供整合結構。

整合式檔案系統的其中幾個重要特性如下：

- 階層式目錄結構，可以將物件像樹枝上的水果一樣組織起來。您可以經由通往物件的目錄來指定路徑，而存取該物件。
- 一般介面，可讓使用者和應用程式存取資料流檔案，以及資料庫檔案、文件、以及存放在 AS/400 上的其他物件。
- 存放在區域 AS/400 上，由不同**檔案系統**管理之資料流檔案的一般察看畫面。

檔案系統可以提供支援，讓使用者和應用程式存取某些區段的儲存體，這些儲存體皆組織為邏輯單元，而且都是檔案、目錄、程式庫和物件。

每一個檔案系統，都有一組與儲存體資訊相互作用的邏輯結構和規則。這些結構檔案可能會因檔案系統的不同而不同。其實，從結構和規則的觀點來看，OS/400 對經

由程式庫來存取資料庫檔案和其他各種物件類型的支援，就可視為一種檔案系統。同樣的，OS/400 對經由資料夾結構來存取文件 (真正的資料流檔案) 的支援，也可以視為一種獨立的檔案系統。

整合式檔案系統的確將程式庫支援和資料夾支援，視為各自獨立的檔案系統。具有不同功能的其他類型的檔案管理支援，也被視為各自獨立的檔案系統。下面是一些可在 AS/400 上使用的檔案系統：

"root" / 檔案系統。root 檔案系統具有磁碟作業系統 (DOS) 和 OS/2 檔案系統兩種特性。

路徑名稱的格式如下：

/Directory/Directory . . . /Object

路徑名稱的每一個構成要素最多可達 255 個字元長，遠比在 QSYS.LIB 或 QDLS 檔案系統還多。而完整的路徑名稱上限相當長，最多可達 16 個 MB。程式和系統空間限制之外的目錄階層，其深度則無限制。

檔案系統保留了物件名稱輸入時的大小寫格式，但在系統搜尋名稱時，大小寫則無區別。

QOpenSys

開放式系統的檔案系統。這個檔案系統與 UNIX 開放式系統標準 (如 POSIX 和 XPG) 相容。

路徑名稱的格式如下：

/Directory/Directory/ . . . /Object

路徑名稱的每一個構成要素最多可達 255 個字元長。完整的路徑名稱最多可達 16 MB 長。程式和系統空間限制之外的目錄階層，其深度則無限制。

與 "root" (/) 檔案系統不同的是，QOpenSys 檔案系統在搜尋物件名稱時，要區分大小寫。例如，全大寫的字元字串，與全小寫的同一個字元字串，就不能算相符。

QSYS.LIB

程式庫檔案系統。這個檔案系統支援 AS/400 程式庫結構。它可以讓您存取資料庫檔案，以及其他由該程式庫支援所管理的所有 AS/400 物件類型。

路徑名稱的每一個構成要素，都必須含有物件名稱，後接該物件的物件類型。例如：

/QSYS.LIB/CGI.LIB/MYPPGM.PGM

/QSYS.LIB/EMP.LIB/PAY.FILE/TAX.MBR

物件名稱和物件類型是由一點 (.) 加以分隔。程式庫中的物件若是類型不同，便可以使用相同名稱，因此物件類型絕不能重複指定，才能識別該物件。每一個構成要素的物件名稱，最多可達 10 個字元長，而物件類型最多可達 6 個字元長。QSYS.LIB 中的目錄階層可以為二或三層深 (路徑名稱中的二或三個構成要素)，端視所存取的物件類型而定。如果該物件是一個資料庫檔案，則階層中可含有三層 (程式庫、檔案、成員)；否則，只能有兩層 (程式庫、物件)。如果 / 和 QSYS.LIB 是前兩層，則 QSYS.LIB 的目錄階層最多可達五層深。

一般說來，QSYS.LIB 檔案系統的物件名稱都沒有大小寫分別。無論物件名稱的字元是大寫還是小寫，在搜尋物件名稱時，結果都一樣。但是，如果名稱括在引號內，則其中每一個字元的大小寫，就將全部保留。因此，如果要搜尋用引號括住的名稱，就得注意引號內的字元大小寫。

如需深入了解整合式檔案系統 (包括上面未列出的其他檔案系統)，請參閱整合式檔案系統簡介。

啓動和執行 Net.Data

本節將告訴您，啓動和執行 Net.Data 的基本步驟。這些步驟如下：

1. 將 Net.Data 程式物件複製到您的 CGI-BIN 程式庫中。
2. 在架構檔中加上特定的指令，來架構 HTTP 伺服器。
3. 建立 Net.Data 起始設定 (INI) 檔 (可選用的)。
4. 建立 Net.Data Web 巨集。
5. 授與 Net.Data CGI 程式所存取之物件的使用者設定檔權限。
6. 呼叫 Net.Data Web 巨集。

如果您依照本節步驟進行之後，仍然無法讓 Net.Data 處理要求，請參閱第216頁的『問題分析』，以尋找問題解析的密訣。

步驟 1 -- 將 Net.Data 程式物件複製到 CGI-BIN 程式庫中

需要複製到 CGI-BIN 程式庫的 Net.Data 程式物件是 DB2WWW，位於 QTCP 程式庫中。請用「建立重複的物件」(CRTDUPOBJ) 命令，來複製程式物件。

DB2WWW 程式物件對 *PUBLIC 使用者的物件授權，設為 *EXCLUDE。變更 CGI-BIN 目錄中的 DB2WWW 程式物件，讓 CGI 程式所執行的使用者設定檔，有權存取該程式物件。要做到這一點，有兩種方法：一種是將程式物件對 *PUBLIC 使用者的權限改成 *USE，另一種是特別授權給使用者設定檔，來存取 DB2WWW 程式物件。

在 V3R2 和 V3R7 中，Internet Connection for AS/400 只在 QTMHHTTP1 使用者設定檔下執行 CGI 程式。

步驟 2--在 HTTP 架構檔加入 Net.Data 指令

您可以使用「使用 HTTP 架構」(WRKHTTPCFG) 命令的選項 1 (新增) 或選項 13 (插入)，來執行下列各項：

1. 確定 Enable GET 和 Enable POST 指令在架構檔中。
2. 為 Net.Data 新增 Map 和 Exec 指令。

如果它們不在其中，請在架構檔中，啓用 Enable GET Y1" 和 Enable POST Y2" 兩種方法的那一節中，加上這兩個指令。當檔案中有了這兩個指令之後，便會出現如圖表 1 所示的顯示畫面：

```
+-----+
                                使用 HTTP 架構
                                系統：  SYSNAM01
請輸入選項，然後按 Enter 鍵。
    1=新增   2=變更   3=複製   4=除去   5=顯示y   13=插入

選項   序號   登錄
```

```

00010      # * * * * *
00020      # HTTP CONFIGURATION FOR NET.DATA TESTING
00030      #
00040      HostName sysnam01.location.company.com
00050      Port 80
00060      #-----
00070      # Methods Enabled
00080      #
00090      Enable GET  Y1"
00100      Enable POST Y2"
00110      #
00120      #-----

```

尚有...

F3=跳出 F5=復新 F6=列印列示 F12=取消 F17=頂端 F18=底端
F19=編輯順序

+-----
圖表 1. 使用 HTTP 架構 (WRKHTTPCFG) -- 顯示畫面 1

新增圖表 2 中所顯示的 Map 和 Exec 陳述式。這個顯示畫面會顯示已經加入的指令。

+-----

使用 HTTP 架構

系統： SYSNAM01

請輸入選項，然後按 Enter 鍵。

1=新增 2=變更 3=複製 4=除去 5=顯示 13=插入

選項	序號	登錄
	00130	#-----
	00140	# Mapping/Pass Rules + Executables
	00150	#
	00160Y1"	Map /cgi-bin/db2www/* /QSYS.LIB/CGI.LIB/DB2WWW.PGM/*
	00170Y1"	Map /CGI-BIN/DB2WWW/* /QSYS.LIB/CGI.LIB/DB2WWW.PGM/*
	00180	#
	00190	#
	00200Y2"	Exec /QSYS.LIB/CGI.LIB/*
	00210	Pass /WWW/html/*
	00220	#-----

尚有...

F3=跳出 F5=復新 F6=列印列示 F12=取消 F17=頂端 F18=底端
F19=編輯順序

+-----
圖表 2. 使用 HTTP 架構 (WRKHTTPCFG) -- 顯示畫面 2

Map 指令 Y1" 可以將格式 "/cgi-bin/db2www/*" 中的登錄，對映到系統上 Net.Data 程式所在的程式庫。(字串後面的星號 (*) 代表字串後面所接的任何資料)。大寫和小寫的 map 陳述式都有，因為指令是有大小寫區分的。在這個範例當中，Map 陳述式會對映到同一個位置。

Exec 指令 Y2" 可以讓 HTTP 伺服器執行 CGI 程式庫中的任何 CGI 程式。您可以在指令上，指定程式所在的程式庫 (而不是程式本身)。若要避免 CGI 程式庫中的其他 *PGM 物件被執行，請將 *PUBLIC 和 QTMHHTP1 排除在物件存取之外。

您必須使用 STRTCPSVR CL 命令重新啟動 HTTP 伺服器，讓架構檔中所做的變更生效：

```
STRTCPSVR *HTTP RESTART(*HTTP)
```


請注意，Pass 指令不是 Net.Data 所用。如果您要簡化 URL，請在 Net.Data 起始設定檔中使用 MACRO_PATH 陳述式。請參閱下一個步驟。

步驟 3--建立 Net.Data 起始設定檔

Net.Data 起始設定檔不一定要建立。使用起始設定檔的好處，是程式的 URL 和參照較短，而且可將檔案併入 Web 巨集檔中。但是，如果您決定建立自己的語言環境，必須具備起始設定檔。

如果沒有建立起始設定檔，則 Net.Data 的執行方式，就好像已經架構了起始設定檔一樣(檔案中只有支援的語言環境陳述式)(請參閱第211頁的『所支援的語言環境』)。所有的巨集、併入和可執行檔參照，都必須完整。

如果起始設定檔已經建立，而且更新過，就不必一定要結束或重新啟動 HTTP 伺服器，才能讓您所做的變更生效。Net.Data 會在 HTTP 伺服器工作初次呼叫時，讀取一次起始設定檔。架構資料會儲存起來，這樣一來，再進行後續的 Net.Data 呼叫時，Net.Data 就不必讀取起始設定檔。但是，如果起始設定檔已經做了變更，Net.Data 會偵測到這個變更，而再次讀取該起始設定檔。

您可以使用「建立來源實體檔」(CRTSRCPF) 命令，來建立起始設定檔。由於架構陳述式中的文字，必須全數放在同一行上，因此，最好能建立一個記錄長度為 240 的起始設定檔。該檔必須在 DB2WWW 程式物件所在的程式庫中建立。檔名必須是 "INI"。成員名稱必須是 "DB2WWW"。您可以使用「來源登錄公用程式」(SEU)，在檔案中加上架構陳述式。

圖表 3 是一個起始設定檔的範例。每一個架構陳述式中的文字，必須全部放在同一行上(為了閱讀方便，因此將 ENVIRONMENT 陳述式分數行顯示)。請注意，下面這個 Net.Data 起始設定檔中，一共有 6 個架構陳述式(包括空行在內)。同時也請注意，如果您不打算在 Web 巨集中使用任何語言環境，就不需要 ENVIRONMENT 陳述式。也不需要指定任何路徑架構陳述式。

```
+-----  
MACRO_PATH      /WWW/MACRO;/QSYS.LIB/WWW.LIB/MACRO.FILE  
INCLUDE_PATH     /WWW/MACRO;/QSYS.LIB/WWW.LIB/MACRO.FILE  
EXEC_PATH        /QSYS.LIB;/QSYS.LIB/WWW.LIB  
  
ENVIRONMENT(DTW_REXX) /QSYS.LIB/QTCP.LIB/QTMRHREXX.SRVPGM ( )  
ENVIRONMENT(DTW_SQL)  /QSYS.LIB/QTCP.LIB/QTMSHSQL.SRVPGM (IN DATABASE,  
    LOGIN, PASSWORD, TRANSACTION_SCOPE, SHOWSQL, DB_CASE, DTW_SET_TOTAL_ROWS,  
    OUT DTWTABLE, SQL_CODE, TOTAL_ROWS )  
ENVIRONMENT(DTW_SYSTEM) /QSYS.LIB/QTCP.LIB/QTMSHSYS.SRVPGM ( )  
+-----
```

圖表 3. Net.Data 起始設定 (INI) 檔的內容

步驟 4--建立 Net.Data Web 巨集

在建立 Web 巨集之前，您必須先決定 Web 巨集要常駐在哪一個檔案系統。Web 巨集是不是存放在類似 UNIX 的檔案系統(如 QOpenSys)上，或者是不是存放在程式庫檔案系統(QSYS.LIB)中，對於 Net.Data 而言並不重要。甚至您可能會希望將 Web 巨集儲存在多個檔案系統上。

您必須根據您選擇要儲存 Web 巨集的檔案系統，來建立目錄或程式庫。下面是幾個範例：

- 在 "root" 檔案系統中，請使用「建立目錄」(CRTDIR) CL 命令來建立目錄樹：

```
CRTDIR DIR('/WWW')
CRTDIR DIR('/WWW/macro')
```

- 在程式庫檔案系統中 (QSYS.LIB)，請使用「建立程式庫」(CRTLIB) CL 命令來建立程式庫，而使用「建立來源實體檔案」(CRTSRCPF) CL 命令來建立來源實體檔案：

```
CRTLIB LIB(WWW)
CRTSRCPF FILE(WWW/MACRO) RCDLEN(240)
```

一旦目錄或程式庫建立起來，就需要建立 Web 巨集，並将它複製到目錄或程式庫中。您可以使用來源登錄公用程式，在 AS/400 系統上建立來源實體檔案成員。您可以使用「複製到資料流檔案」(CPYTOSTMF) CL 命令，將該成員從來源實體檔案成員複製到檔案系統（亦即 "root"）目錄上。例如，假設我們在程式庫 WWW、來源實體檔案 MACRO、以及含有下述文字的成員中，建立了來源實體檔案成員 MACSAMP：

```
%HTML(HelloWorld) {
<P>Hello World
%}
```

如果我們希望將該成員複製到 "root" 檔案系統的目錄 /WWW/macro 中，則需要使用 CPYTOSTMF，如下所示：

```
CPYTOSTMF FROMMBR('/qsys.lib/www.lib/macro.file/MACSAMP.mbr')
TOSTMF('/WWW/macro/MACSAMP') STMFOP(*REPLACE) ENDLINFMT(*LF)
```

步驟 5--授與使用者設定檔對物件的權限

CGI 程式所執行的使用者設定檔，必須具備對 Web 巨集中參照的任何物件，以及對 URL 參照的巨集，具備適當的存取權。

在 V3R2 和 V3R7 中，Internet Connection for AS/400 只在 QTMHHTP1 使用者設定檔下，執行 CGI 程式。

您必須根據您選擇要將 Web 巨集儲存在哪一個檔案系統，而授權讓執行 Net.Data CGI 程式的使用者設定檔來存取 Web 巨集。下面是對 QTMHHTP1 使用者設定檔授權的幾個範例：

- 在 "root" 檔案系統中，請使用「變更權限」(CHGAUT) CL 命令，來授權給使用者設定檔（請注意，您必須授權讓它存取路徑中所有的物件）：

```
CHGAUT OBJ('/WWW') USER(QTMHHTP1) DTAAUT(*RX)
CHGAUT OBJ('/WWW/macro') USER(QTMHHTP1) DTAAUT(*RX)
CHGAUT OBJ('/WWW/macro/*') USER(QTMHHTP1) DTAAUT(*RX)
```

- 在程式庫檔案系統中 (QSYS.LIB)，請使用「授與物件權限」(GRTOBJAUT) CL 命令，來授權給使用者設定檔（請注意，您只需要授權讓它存取程式庫和來源實體檔案即可）：

```
GRTOBJAUT OBJ(WWW) OBJTYPE(*LIB) USER(QTMHHTP1) AUT(*USE)
GRTOBJAUT OBJ(WWW/MACRO) OBJTYPE(*FILE) USER(QTMHHTP1) AUT(*USE)
```

您也可以使用 CHGAUT CL 命令，來授與 QSYS.LIB file 系統中的物件存取權限，如下所示：

```
CHGAUT OBJ('/QSYS.LIB/WWW.LIB') USER(QTMHHTP1) DTAAUT(*RX)
CHGAUT OBJ('/QSYS.LIB/WWW.LIB/MACRO.FILE') USER(QTMHHTP1) DTAAUT(*RX)
```

語言環境專用的權限注意事項，都記錄在 第211頁的『所支援的語言環境』中的每一個語言環境節中。

步驟 6--呼叫 Net.Data Web 巨集

Net.Data for the OS/400 只能被當作一般開道介面 (CGI) 程式加以呼叫。Net.Data 可以從下列幾種方法呼叫：

- 錨點參照

```
<A HREF="http://{<I>web-server}/cgi-bin/db2www/{macro-file}/
{HTML-block} ?name=val&... ">any text</A>
```

- HTML 表格頁面

```
<FORM METHOD={method} ACTION=http://{web-server}/cgi-bin/db2www/
{macro-file}/{HTML-block} ?name=val&... >any text</FORM>
```

- URL

```
http://{<I>web-server}/cgi-bin/db2www/{<I>macro-file}/{<I>HTML-block}
<I>?name=val&...
```

表 76. 巨集呼叫元素

method	您可以指定 get 或 post，這兩種功能在 HTML 2.0 中都有指定。由於 get 方法有其限制，因此較不被採用。
web-server	這是系統管理者所定義的 Web 伺服器名稱。例如，www.ibm.com。如果巨集存在於區域機器上，則只需要相對的 URL 即可，不需要 http:/www.ibm.com/。
macro-file	這是 Net.Data 應用程式軟體開發者所定義的 Web 巨集名稱。您在此處所指定的，直接關係到您的 Net.Data 起始設定檔中是否指定 MACRO_PATH 陳述式。如果沒有指定 MACRO_PATH，則需要對 Web 巨集指定完整的路徑名稱。比方說，如果起始設定檔中沒有指定 MACRO_PATH，則巨集檔為 /QSYS.LIB/WWW.LIB/MACRO.FILE/MACSAMP.MBR。但是，如果起始設定檔中有 MACRO_PATH 陳述式，則巨集檔為 MACSAMP.MBR。
HTML-block	這是您所呼叫的 Web 巨集中的 HTML 區塊名稱。
?name=val&	這些是可在您應用程式中傳送的選用性參數。例如，您可以傳送使用者 ID，這樣就不必一再輸入；或者，也可以傳送呼叫另一個巨集的 Web 巨集名稱，方便讓您退出。

需要呼叫先前所建立之範例巨集的 URL (假設沒有 MACRO_PATH 陳述式) 為：

- "root" 檔案系統

```
http://server/cgi-bin/db2www/WWW/MACRO/MACSAMP/HelloWorld
```

- 程式庫檔案系統 (QSYS.LIB)

```
http://server/cgi-bin/db2www/QSYS.LIB/WWW.LIB/MACRO.FILE/MACSAMP.MBR/HelloWorld
```

在 AS/400 上使用 Net.Data

下列是 Net.Data 之 OS/400 實作方法的明顯特性，以及一些有幫助的提示：

- Net.Data 巨集可以儲存在任何檔案系統中。您可以使用「複製到資料流檔案」(CPYTOSTMF) 命令，將 Net.Data 巨集從 QSYS.LIB 檔案系統複製到另一個檔案系統。
- 如果您要建立自己的語言環境，則支援 Net.Data 語言環境介面的 API，是在 QTMHLE 服務程式中。ILE C 語言表頭檔可提供呼叫 ILE C 中之 Net.Data API 所需的原型。您可以在 Net.Data 的 URL 中，找到該介面的相關文件。若要取得必要的原型，請依下述方法併入表頭檔：

```
#include <dtwle.h>
```

QTMHLE *SRVPGM 物件已併入於 QTCP 程式庫中。

QSYSINC 程式庫含有表頭檔，它可以安裝，但不一定非安裝不可。在編譯使用這些表頭檔的程式之前，務必將 QSYSINC 放在您的系統上。

- 如果您無法取得語言環境來正常運作，請察看 HTTP 伺服器工作中的工作日誌，看看是否有任何訊息發出，告訴您問題的癥結所在。當您針對剛建立的語言環境加以除錯時，最好將伺服器工作的最小值和最大值設為 2，這樣，Net.Data 要求就可以經由一個 HTTP 伺服器工作而集中發出。此外，每一次語言環境服務程式更新時，您都必須結束 HTTP 伺服器，然後再將它重新啟動，讓 Net.Data 得以啟動更新後的服務程式。
- Net.Data 的 OS/400 實作方法，並不支援下列可能在其他平台上被支援的 Net.Data 特性：
 - INCLUDE_URL 陳述式。當它發生時，這個陳述式不予處理。
 - 使用「一般閘道介面」緒管理的現場連線。但是，透過 URL 呼叫，與資料庫的連線都保持在穩定狀態下，因此，在存取遠端資料庫時，不會降低執行效能。
 - IBM Internet Connection Servers (ICS)、NetScape Servers (NS) 和 Microsoft 的 Internet Server (IS) 的應用程式設計介面支援。

所支援的語言環境

Net.Data 可讓新的語言和資料庫介面，以「可插入式」的方法加入。這些語言環境都被當作服務程式加以存取。服務程式的名稱，是在 Net.Data 起始設定檔中架構，而且與語言環境名稱有關。每一個語言環境都必須支援一組由 Net.Data 定義的介面。

Net.Data for OS/400 支援下列各語言環境：

- REXX：可讓外部 REXX 程式或列入 (巨集函數區塊中的 REXX 陳述式) 由 REXX 直譯器加以解譯。您可以在 *AS/400 REXX/400 參考手冊* 中，找到 REXX 的相關資訊。
- SQL：可讓 SQL 陳述式由 DB2 處理。您可以在 *DB2 for OS/400 SQL 參考手冊* 中，找到 SQL 的詳細資訊。
- SYSTEM：可讓外部程式 (例如：C、C++、RPG、COBOL) 執行。

下列各節將為您說明上述語言環境。

REXX (DTW_REXX) 語言環境

REXX 語言環境可以解譯在 Net.Data 巨集之 FUNCTION 區塊中指定的內部 REXX 程式，或者，可以執行儲存在另一個檔案的內部 REXX 程式。REXX 語言環境的部份特性如下：

- QREXX() REXX 應用程式介面 (API) 是語言環境用來啟動 REXX 程式之 REXX 直譯器的工具。
- QREXVAR() REXX API 是 REXX 語言環境用來將資料傳給 REXX 程式，並從 REXX 程式讀取資料所用的工具。因此，REXX 程式可以直接操縱 FUNCTION 區塊中所指定的 Net.Data 參數變數。

- REXX 語言環境可以在預設的命令環境 COMMAND (CL 命令環境) 中，啟動 REXX 直譯器。如果您要執行另一個命令環境，如 EXEC SQL (SQL 環境)、CPICOMM (CPI 通信環境)、或是使用者定義的語言環境，就必須使用 ADDRESS 內建 REXX 函數，在 REXX 程式中指定語言環境。
- REXX 語言環境要求內部 REXX 程式必須常駐在 QSYS.LIB 檔案系統中。
- 除了 REXX 程式所用的任何資源之外，QTMHHTTP1 使用者設定檔必須具有適當的權限，來存取和讀取含內部 REXX 程式的檔案。

REXX 程式會將 Net.Data 巨集表格參數的值，當作 REXX stem 變數加以存取。對於 REXX 程式而言，表格 T 的欄位標題是 T_N.i，而欄位值是 T_V.i.j。

對外部 REXX 程式的呼叫，是由下述格式的陳述式，在 FUNCTION 區塊中識別：

```
%EXEC{ REXX-file-name Yoptional parameters" %}
```

下面是一個簡單的範例，說明內部 REXX 程式和參照至外部 REXX 程式的巨集 (REXXM)。

```
%define a = "3"
%define b = "0"

%function(DTW_REXX) func1(IN inp1, OUT outp1){
%EXEC{ /QSYS.LIB/REXX.LIB/REXXSRC.FILE/TREXX.MBR %}
%}

%function(DTW_REXX) func2(IN inp1, OUT outp1){
outp1 = 2*inp1
%}

%HTML(REPORT){
@func1(a, b)
b=$(b)
@func2(a, b)
b=$(b)
%}
```

在範例當中，@func1 會產生由 REXX 直譯器解譯的 REXX 程式 TREXX.MBR；而 @func2 則產生負責解譯陳述式 "outp1 = 2*inp1" 的 REXX 直譯器。在這兩種案例中，都設定了 REXX 變數儲存池，讓 REXX 直譯器得以存取變數 "a" 和 "b"。當 @func2 完成之後，"b" 便設為 "6" (假設 REXX 程式 TREXX.MBR 不修改 "a")。

這是一個參照該巨集的 URL 範例，其前提是假設下列各項成立：

- 巨集位於 /WWW/巨集目錄
- 已經架構 HTTP 伺服器來呼叫 Net.Data CGI-BIN 程式 DB2WWW
- CGI-BIN 程式所執行的使用者設定檔 QTMHHTTP1，已被授權存取該巨集檔，以及含有 REXX 程式的檔案

```
http://hostname/cgi-bin/db2www/WWW/macro/REXXM/report
```

如果您選擇不建立 Net.Data 起始設定檔，則 REXX 語言環境將採用預設值。但是，如果建立了起始設定檔，而且您希望使用 REXX 語言環境，則下述架構陳述式就必須在起始設定檔中：

```
ENVIRONMENT(DTW_REXX) /QSYS.LIB/QTCP.LIB/QTMHREXX.SRVPGM ( )
```

SQL (DTW_SQL 或 SQL) 語言環境

SQL 語言環境是用 DB2 來執行 SQL 陳述式。下面是 SQL 語言環境的其中一些特性：

- SQL 語言環境要求區域資料庫的目錄登錄，一定要在關聯式資料庫目錄中，（也就是說，遠端位置為 *LOCAL 的目錄登錄）。您可以用「新增關聯式資料庫目錄登錄」(ADDRDBDIRE) 命令，來新增登錄。
- 任何有效的 SQL 陳述式，都可以傳到 SQL 語言環境中。SQL 陳述式必須是有效的 DB2 for OS/400 命令。
- SQL 語言環境是在確定控制下執行。因此，所有由規則掌控的確定控制，都必須加入。所有經由 DTW_SQL 存取的檔案或表格，也都必須記錄下來（但是 SQL 陳述式為 SELECT 的案例則除外）。此外，使用者設定檔 QTMHHTPI 必須有足夠的權限來執行必要的作業。如果您要存取集合中的一個 SQL 表格，它會經由 AS/400 上的 native SQL 支援，自動為您記錄下來，而不需要採取任何明確的動作。如果您要執行 DTW_SQL 語言環境 FUNCTION 區塊中的 "CREATE TABLE" 陳述式，必須先授與使用者設定檔 QTMHHTPI 適當的權限，來存取正在建立表格的集合中的日誌，讓表格得以加入日誌中。如果沒有這麼做，"CREATE TABLE" 要求便會失敗。
- 可以使用 SQL 或系統命名模式。其預設值是 SQL 命名模式。如果您要使用系統命名模式，請在 Net.Data 起始設定檔中插入下面這一行：

```
DTW_SQL_NAMING_MODE = SYSTEM_NAMING
```

同時，也可以將 DTW_SQL_NAMING_MODE 設為 SQL_NAMING，與 SQL 命名模式的預設值相同。

當您與遠端 AS/400 建立連線時，「SQL 呼叫層次介面」會在程式庫 QGPL 中，尋找名為 QSQLPKG 的 *SQLPKG 物件。如果它存在的話，就採用它，否則，就建立它。這個 SQL 套裝軟體含有存取 native SQL 時所遵從的規則。因此，由第一個連線的屬性設定規則，後續的連線都必須遵守。這些屬性的其中一項是命名模式。如果您設定的 DTW_SQL_NAMING_MODE，與遠端 AS/400 上現有 QGPL/QSQLPKG 的命名模式相衝突，Net.Data 巨集中的 SQL 陳述式便會產生 SQLCODE -5016。為了避免這種情況，請選定一個命名模式之後，就不再變更。如果 QGPL/QSQLPKG 物件與您所選的命名模式相衝突，請將它刪除，並再次發出 Net.Data 要求。再新建一個符合您要求之命名模式的 QGPL/QSQLPKG。

- 您不可以與同一個遠端資料庫建立並行連線。如果目前已經使用一個使用者 ID (LOGIN SQL 語言環境參數) 與遠端資料庫連線，但又被要求使用第二個使用者 ID，與同一個遠端資料庫連線，這時候，SQL 語言環境必須先切斷目前的連線，待確定之後，再用「新」的使用者 ID 和通行碼，重新建立連線。必須進行確定的原因是，如果連線斷了，而稍後巨集又發生錯誤時，就完全沒有辦法進行回復作業了。但如果您是使用 "TRANSACTION_SCOPE=SINGLE"，就不成問題了。如果您是使用 "TRANSACTION_SCOPE=MULTIPLE" (此為預設值)，而「新」的使用者 ID 與先前用來建立資料庫和遠端系統連線的使用者 ID 不同，則 SQL 語言環境會自動回復，並傳回 SQL_CODE -752，指出該連線不變。
- 您最多可以存取 50 個資料庫，包括區域和遠端在內。與資料庫的連線，一直都被 SQL 語言環境保持在作用中的狀態，使得 Net.Data 所執行的 HTTP 伺服器工作得以維持。這樣一來，在初次連接資料庫之後，就可以迅速存取資料庫了。

當 SQL 語言環境與遠端系統建立連線時，就建立了使用者 ID 與該連線的關聯。如果在後續的 Net.Data 查詢中，該使用者 ID 與該連線相關的使用者 ID 不相符，則連線會結束，而另外再與該資料庫建立新的連線（只有當異動範圍是 SINGLE 時，才會發生這種情況）。因此，如果要考慮到執行效能的話，Web 巨集撰寫者應該在發出 SQL 陳述式給遠端資料庫時，將它編入或使用同一個使用者 ID。在進行區域資料庫存取時，則不處理使用者 ID 和通行碼。

- 如果您所建立的語言環境，是使用資料庫存取類別程式庫或 SQL 呼叫層次介面，而且是在巨集中參照，則不能使用該 SQL 語言環境。
- QTMHHTTP1 使用者設定檔必須具備適當的權限，可以存取 HTTP 伺服器所在機器上的資料庫。對於遠端資料庫而言，使用者 ID 和通行碼是用來決定可以存取哪些資料庫資源。
- SQL 陳述式無法傳到 EXEC 陳述式上的 SQL 語言環境。

下面是一個簡單的範例，說明一個發出單 SQL 命令的巨集 (SQLM)：

```
%define DATABASE="HOSTNAME"

%FUNCTION(DTW_SQL) sql1 (){
select * from custinfo.customer
%}

%HTML(REPORT){
@sql1()
%}
```

參照該巨集的 URL，與 REXX 語言環境的 URL 範例很類似，只是巨集檔名 REXXM 換成 SQLM。

如果您選擇不建立 Net.Data 起始設定檔，則 SQL 語言環境採用預設值。但是，如果建立了起始設定檔，而且您希望使用 SQL 語言環境，則下述架構陳述式就必須在起始設定檔中：

```
ENVIRONMENT(DTW_SQL) /QSYS.LIB/QTCP.LIB/QTMSQL.SRVPGM
( IN DATABASE, LOGIN, PASSWORD, TRANSACTION_SCOPE, SHOWSQL, DTW_SET_TOTAL_ROWS,
  DB_CASE, OUT DTWTABLE, SQL_CODE, TOTAL_ROWS )
```

這個環境陳述式的文字，在起始設定檔中必須全部置放在同一行。此處分成數行顯示，則是為了方便您閱讀。

上述架構陳述式中的 SQL 語言環境參數，是傳到語言環境，我們將說明如下：

- **DATABASE**：SQL 語言環境與此變數中所指定的資料庫建立連線 (如果尚未建立連線的話)。下面這個範例，將告訴您如何在 Net.Data 巨集中設定這個變數：

```
%DEFINE DATABASE="HOSTNAME"
```

- **LOGIN** 和 **PASSWORD**：如果 DATABASE 參數是參照遠端資料庫的話，則在連接資料庫時，會使用 LOGIN 和 PASSWORD 變數中所指定的使用者 ID 和通行碼。但在存取區域資料庫時，這些參數則不被處理，而 CGI-BIN 程式所執行的使用者設定檔 QTMHHTTP1，必須被授權存取任何被存取的檔案。

```
%DEFINE{ LOGIN="MYUSERID"
          PASSWORD="DB2WWW"
%}
```

- **TRANSACTION_SCOPE**：指定 SQL 命令的異動範圍。如果沒有定義該變數，則預設動作是 "MULTIPLE"，也就是說，只有在 %HTML 區塊中所有的 SQL 命令都順利完成之後，才進行 COMMIT。SQL 命令若不成功，將使該區塊中所有先前執行的 SQL 命令全部回復。若指定 "SINGLE"，則表示 %HTML 區塊中的每一個 SQL 命令順利完成時，都要進行一次 COMMIT。

```
%DEFINE TRANSACTION_SCOPE="SINGLE"
```

- **SHOWSQL**：隱藏或顯示已經執行的 SQL 命令。其預設值是不顯示已經執行的 SQL 命令。

```
%DEFINE SHOWSQL="YES"
```

- **DB_CASE**：指定在 SQL 命令執行之前的大小寫。其預設值是不轉換。您可以指定 "UPPER" 或 "LOWER"，強迫將 SQL 命令中所有的字元都設為大寫或小寫。

```
%DEFINE DB_CASE="UPPER"
```

- **DTW_TABLE**：如果沒有傳來使用者定義的表格，則 SQL 查詢的結果將儲存在這個表格中。
- **SQL_CODE**：這個變數將含有 SQL 警告或錯誤碼。若 SQL 查詢成功，則 SQL_CODE 為零。

查詢
完成，其 SQL 碼為 \$(SQL_CODE)。

系統 (DTW_SYSTEM) 語言環境

SYSTEM 語言環境可以容許呼叫在 FUNCTION 區塊中的 EXEC 陳述式所識別的外部程式。

SYSTEM 語言環境可以使用 C 語言 system() 函數呼叫，將指定的程式名稱和參數，傳到作業系統等待執行，以解譯 EXEC 陳述式。由於這個方法不容許 Net.Data 變數像 REXX 語言環境一樣，直接傳送或取回到可執行檔陳述式，因此 SYSTEM 語言環境是以下述方法傳送和取回變數：

- 輸入參數是使用 putenv() 函數，當作系統的「環境變數」來傳送，而且可以用語言專用的方法來執行程式，而加以取回。
- 輸出參數則是用語言專用的 putenv() 函數，傳回 SYSTEM 語言環境。

SYSTEM 語言環境程式會根據 Net.Data 巨集表格參數值的 Net.Data 名稱，來存取它們。表格 T 的欄位標題是 T_N_i，而欄位值是 T_V_i_j。

SYSTEM 語言環境預期該可執行檔為命令或程式。除了可執行檔所用的任何資源之外，QTMHHTP1 使用者設定檔必須具備適當的權限來執行可執行檔。

下面是一個簡單的巨集範例 (SYSM)，該巨集指定程式為可執行檔，並將它傳給一個 Net.Data 參數：

```
%define var1 = "OriginalValue"

%FUNCTION(DTW_SYSTEM) test(INOUT parm1){
%EXEC{ /QSYS.LIB/PGM.LIB/TSYS0001.PGM %}
%}

%HTML(REPORT){
<PRE>
在函數呼叫之前的 var1 值：$(var1)
@test(var1)
在函數呼叫之後的 var1 值： $(var1)
</PRE>
%}
```

參照該巨集的 URL，與 REXX 語言環境的 URL 範例很類似，只是巨集檔名 REXXM 換成 SYSM。

如果您選擇不建立 Net.Data 起始設定檔，則 SYSTEM 語言環境採用預設值。但是，如果建立了起始設定檔，而且您希望使用 SYSTEM 語言環境，則下述架構陳述式就必須在起始設定檔中：

```
ENVIRONMENT(DTW_SYSTEM) /QSYS.LIB/QTCP.LIB/QTMSYS.SRVPGM ( )
```

從 DB2 WWW Connection 移轉到 Net.Data

如果您要使用 DB2 WWW 版本 1，而且要移轉到 Net.Data (DB2 WWW 版本 2)，請閱讀 第206頁的『啟動和執行 Net.Data』。如果在閱讀本節之後，您覺得需要一個 Net.Data 起始設定檔 (一如剛剛讀過的部份所說明，如果參照 Net.Data 巨集的 URL，沒有完全表明 Net.Data 巨集位置的話，就需要建立一個起始設定檔)，請建立一個。如果您決定建立一個起始設定檔，就必須在起始設定檔中加入一個 SQL 語言環境陳述式 (有關 SQL 語言環境陳述式語法的相關資訊，請參閱第212頁的『SQL (DTW_SQL 或 SQL) 語言環境』)。

使用 DB2 WWW Connection 版本 1 時，每一個巨集檔所在的程式庫中，都需要有一個起始設定檔。但是 Net.Data 就不一定了。如果您選擇建立 Net.Data 起始設定檔，而且在 DB2 WWW 版本 1 起始設定檔中使用 MACRO_PATH 架構陳述式，則必須將這些路徑陳述式組合成一個路徑陳述式，然後將組合後的路徑陳述式，插入 DB2WWW 程式物件所在的程式庫的起始設定檔中。一旦 Net.Data 開始為 Net.Data 巨集執行作業之後，就應該刪除所有不用的 DB2 WWW 版本 1 起始設定檔。

問題分析

下面這些問題，是假設 Net.Data CGI-BIN 程式物件 DB2WWW 已經移到 CGI-BIN 程式所在的程式庫 WWWCGI 中。

- 症狀：錯誤 500 (後接訊息內容)

```
script 要求不正確 -- '/QSYS.LIB/WWWCGI.LIB/DB2WWW.PGM/  
QSYS.LIB' 不能執行
```

原因：Exec 規則不正確

```
Exec /QSYS.LIB/WWWCGI.LIB/DB2WWW.PGM/*  
Exec /qsys.lib/wwwcgi.lib/db2www.pgm/*
```

解決方法：指定一個 Exec 規則，只提供路徑給 DB2WWW 程式。例如：

```
Exec /QSYS.LIB/WWWCGI.LIB/*  
Exec /qsys.lib/wwwcgi.lib/*
```

- 症狀：錯誤 404 (後接訊息內容)

```
找不到 - 檔案不存在，或者有讀取保護  
Yeven tried multi"
```

原因：漏了 Exec 規則。

解決方法：以大寫和小寫指定 Exec 規則，提供通往 DB2WWW 程式的路徑。例如：

```
Exec /QSYS.LIB/WWWCGI.LIB/*  
Exec /qsys.lib/wwwcgi.lib/*
```

- 症狀：錯誤 403 (後接訊息內容)

禁止 - 按照規定

原因：Map 或 Exec 規則漏了或不正確。

解決方法：以大寫和小寫指定 DB2WWW 程式的 Map 和 Exec 規則。例如：


```
Map /cgi-bin/db2www/* /QSYS.LIB/WWWCGI.LIB/DB2WWW.PGM/*
Map /CGI-BIN/DB2WWW/* /QSYS.LIB/WWWCGI.LIB/DB2WWW.PGM/*
Exec /QSYS.LIB/WWWCGI.LIB/*
```

- **症狀：**架構陳述式正確，但是 Net.Data 無法運作。當所有的架構都在而且都正確，但 Net.Data 卻仍然不能妥善處理資料或文件 (或者根本不能) 時，就會發生這個問題。可能的原因和解決方法有下列幾種：

- **原因：**Map、Exec 或 Pass 規則的順序不對。

解決方法：當 URL 要根據 Map、Exec 或 Pass 規則加以評估時，是根據第一個相符的規則來執行。因此，您必須非常小心，以確保要評估的陳述式在達到要求的規則之前，沒有被重複對映或變更。同時，也必須確定使用者的架構檔中，沒有 "Pass /*"。

- **原因：**使用者設定檔 QTMHHTTP1 沒有適當的權限，可以存取 Net.Data 巨集。

解決方法：所有的 CGI-BIN 程式都在使用者設定檔 QTMHHTTP1 下執行。QTMHHTTP1 使用者設定檔必須獲得授權，可以存取 Net.Data 在處理 Net.Data 巨集時，所要存取的所有物件。

- **原因：**Net.Data 起始設定檔中的路徑陳述式不正確。

解決方法：Net.Data 會使用起始設定檔中的路徑陳述式 (如果有的話)，形成目前正在處理之 Net.Data 巨集中的任何 Net.Data 巨集或可執行檔參照。如果物件參照不完整，而且起始設定檔中的路徑陳述式不正確，則 Net.Data 會告訴您，它找不到所參照的物件。因此，您必須確定讓物件參照完整，或者讓 Net.Data 起始設定檔具有適當的路徑陳述式。

附錄H. Net.Data for Windows NT

Net.Data 是用來取代 DB2 WWW Connection 。有關 Net.Data for Windows NT 的詳細資訊，請參閱隨附於 Net.Data 的 README 檔案。README 檔案包括這些資訊：

- 基本需求
- 安裝
- 架構
- 解除安裝
- 樣本應用程式
- 常問的問題

名詞解釋

API. 應用程式設計介面。Net.Data 為增進透過 CGI 處理的執行效能，因而支援三項專賣的 API。

applet. 併入於 HTML 頁面中的一個 Java 程式。Applet 是與可使用 Java 的瀏覽器（如 Netscape）一起使用，且是在當 HTML 頁面被載入時載入。

應用程式設計介面 (API). 為一功能性介面，由作業系統或由一個可分開訂購的授權程式所提供，讓您可以使用高階語言撰寫應用程式來使用作業系統或授權程式之特定資料或函數。

BLOB. 二進位大型物件。

CGI. 通用閘道介面。

cliette. 一個服務 Web 伺服器所提出之要求的長期執行的處理。連線管理程式會排程 cliette 處理來服務這些要求。

CLOB. 字元大型物件(Character large object)。

通用閘道介面. Web 伺服器傳送控制權給應用程式以及接收回資料的一個標準路徑。

連線管理程式. 一個名稱為 dtwcm.exe 的可執行檔，dtwcm.exe 在 Net.Data 中，該檔案是用來支援現場連線。

資料庫. 一個含有表格，或含有表格空間和索引空間的集合體。

資料庫管理系統 (DBMS). 一個控制資料庫的建立、組織、和修改以及存取儲存於其內之資料的軟體系統。

資料類型. 直欄和文字的屬性。

DBMS. 資料庫管理系統(Database management system)。

防火牆. 具有防火牆軟體的電腦可保護內部網路不受外界的侵害。

一般檔案介面. 一個讓您可讀取與撰寫純本文檔案的 Net.Data 語言環境。

html. 超本文標示語言 (hypertext markup language)

http. 超本文轉送通信協定 (hypertext transfer protocol)

超本文標示語言. 一個使用來撰寫 Web 文件的標示語言。

超本文轉送通信協定. 一個使用在 Web 伺服器和瀏覽器之間的通信協定。

ICAPI. nternet 連線 API (Internet Connection API)

ICS. Internet 連線伺服器 (Internet Connection Server)

ICSS. Internet 連線保密伺服器 (Internet Connection Secure Server)

Internet. 一個國際公用的 TCP/IP 電腦網路。

Internet 連線伺服器. IBM 之非保密 Web 伺服器。

Internet 連線保密伺服器. IBM 的保密 Web 伺服器。

內部網路. 一個位於公司防火牆內的 TCP/IP 網路。

ISAPI. Microsoft 的 Internet 伺服器 API。

Java. 一個對 Internet 應用程式而言特別有用的與平台無關的物件導向的程式設計語言。

語言環境. 一個 Net.Data 用來提供對外部資料來源（如 DB2），或程式設計語言之存取的可插式模組。

現場連線. 與連線管理程式及 Web 伺服器 API 一起運作的 Net.Data 架構。現場連線可讓資料庫連線被再次使用。

LOB. 大型物件 (Large object)

NSAPI. Netscape API。

空值. 一個表示沒有資訊之特殊值。

路徑. 用來尋找檔案的搜尋路徑。

PERL. 已直譯的程式設計語言。

埠. 一個 16 位元的號碼，用來供 TCP 和高階通信協定或應用程式通信用。

TCP/IP. 傳輸控制通信協定 / Internet 通信協定 (Transmission Control Protocol/Internet Protocol)

傳輸控制通信協定 / Internet 通信協定. 一組支援區域及廣域網路之對等連接功能的通信協定。

URL. 通用資源位置 (Uniform resource locator)

通用資源指位器. 一個，用來指出 http 伺服器以及選擇性地指出目錄以及檔案名稱的位址，例如
<http://www.software.ibm.com/data/net.data/index.html>.

Web 伺服器. 執行 http 伺服器軟體(如 Internet Connection)的電腦。

索引

索引順序以中文字，英文字，及特殊符號之次序排列。

〔四劃〕

內含檔 27
內建函數 85

〔五劃〕

平台, AIX 195
平台, NT 219
平台, OS/2 196
平台, OS/390 199
平台, OS/400 203

〔六劃〕

字串
 函數參數 85

〔八劃〕

函數
 一般 85
 字串 110, 85
 字組 126
 表格 134
 浮點數 85
 純本文檔案 149
 陣列 85
 說明 85
 數學 99
 整數 85
 Web 登記 167
呼叫
 外部程式 13
 函數 20
表頭 27

〔十劃〕

浮點數
 函數參數 85
陣列
 函數參數 85

〔十一劃〕

條件邏輯 24

〔十二劃〕

備註 8

〔十四劃〕

語法圖，閱讀 1

〔十五劃〕

標底 27

〔十六劃〕

整數
 函數參數 85
錯誤處理 37

〔十七劃〕

環境變數 12

〔二十三劃〕

變數
 報表 65
 預先定義 49
 隱含 55
 SQL 73
 TABLE 陳述式 48
變數名稱 6
變數參照 7

A

AIX 平台 195
ALIGN 66

D

DATABASE 73
DB2PLAN 75
DB2SSID 76
DB2WWW
 語法 180
DB_CASE 74
DEFINE 區塊 9
DEFINE 陳述式 9
DTWF_APPEND 150
DTWF_CLOSE 152

DTWF_DELETE 153
DTWF_INSERT 155
DTWF_OPEN 157
DTWF_READ 158
DTWF_REMOVE 160
DTWF_SEARCH 161
DTWF_UPDATE 163
DTWF_WRITE 165
DTWR_ADDENTRY 167
DTWR_CLEARREG 168
DTWR_CREATEREG 169
DTWR_DELENTY 170
DTWR_DELREG 171
DTWR_LISTREG 172
DTWR_LISTSUB 173
DTWR_TRVENTRY 174
DTWR_UPDATEENTRY 175
DTW_ADD 99
DTW_ADDQUOTE 86
DTW_APPLET_ALTTEXT 66
DTW_ASSIGN 110
DTW_CONCAT 111
DTW_CURRENT_FILENAME 49
DTW_CURRENT_LAST_MODIFIED 50
DTW_DATE 87
DTW_DEFAULT_REPORT 67
DTW_DELSTR 112
DTW_DELWORD 126
DTW_DIVIDE 100
DTW_DIVREM 101
DTW_FORMAT 102
DTW_GETENV 88
DTW_GETINIDATA 89
DTW_HTMLENCODER 90
DTW_HTML_TABLE 68
DTW_INSERT 113
DTW_INTDIV 105
DTW_LASTPOS 115
DTW_LENGTH 116
DTW_LOWERCASE 117
DTW_MACRO_FILENAME 51
DTW_MACRO_LAST_MODIFIED 52
DTW_MP_PATH 53
DTW_MP_VERSION 54
DTW_MULTIPLY 106
DTW_POS 118
DTW_POWER 107
DTW_PRINT_HEADER 69
DTW_QHTMLENCODER 92
DTW_REVERSE 119
DTW_SAVE_TABLE_IN 77
DTW_SETENV 94

DTW_SET_TOTAL_ROWS 70
DTW_SQL_NAMING_MODE 213
DTW_STRIP 120
DTW_SUBSTR 121
DTW_SUBTRACT 108
DTW_SUBWORD 127
DTW_TB_DLIST 135
DTW_TB_DUMPV 137
DTW_TB_DUMPV 138
DTW_TB_HTMLENCODER 139
DTW_TB_INPUT_CHECKBOX 140
DTW_TB_INPUT_RADIO 141
DTW_TB_INPUT_TEXT 142
DTW_TB_LIST 143
DTW_TB_SELECT 145
DTW_TB_TABLE 146
DTW_TB_TEXTAREA 148
DTW_TIME 95
DTW_TRANSLATE 123
DTW_UPPERCASE 124
DTW_URLESCSEQ 97
DTW_WORD 129
DTW_WORDINDEX 129
DTW_WORDLENGTH 130
DTW_WORDPOS 131
DTW_WORDS 133

E

ENVVAR 陳述式 12
EXEC 陳述式 13
EXEC_SQL 177

F

FUNCTION 區塊 15

H

HTML IF 陳述式 24
HTML 區塊 22
HTML_INPUT 區塊 178
HTML_REPORT 區塊 179

I

INCLUDE 陳述式 27
INCLUDE_URL 陳述式 29

L

LIST 陳述式 31
LIST 變數 47

LOCATION 78
LOGIN 79

M

Macro IF 區塊 33
MESSAGE 區塊 37

N

NLIST 57
Nn 56
NT 平台 219
NUM_COLUMNS 58
N_columnName 55

O

OS/2 平台 196
OS/390 平台 199
OS/400 平台 203

P

PASSWORD 80

R

REPORT 區塊 39
RETURN_CODE 59

ROW 區塊 42
ROW_NUM 60
RPT_MAX_ROWS 71

S

SHOWSQL 81
SQL 命名模式 213
SQL 區塊 180
SQL_CODE 183
SQL_MESSAGE 區塊 181
SQL_REPORT 區塊 182
START_ROW_NUM 72

T

TABLE 陳述式 43, 48
TOTAL_ROWS 61
TRANSACTION_SCOPE 82

V

VLIST 64
Vn 63
V_columnName 62