返回总目录

SAVE MACROS 命令

把一组键盘宏保存到键盘宏文件或备注字段中。

语法

SAVE MACROS TO FileName | TO MEMO MemoFieldName

参数描述

TO FileName

指定保存宏的文件。键盘宏文件名最多可以包括 8 个字符，并且起始字符必须是一个字母或下划线，不能用数字作为文件名的第一个字母。除文件名第一个字符外，键盘宏文件名的其他字符可以为数字、字母和下画线的任意组合。一般应指定键盘宏文件扩展名为 .FKY 。如果指定的扩展名不是 .FKY ，则在用 RESTORE MACROS 命令恢复键盘宏时，必须包含指定的扩展名。

TO MEMO MemoFieldName

指定保存宏的备注字段。包括备注字段的表必须是打开的。但是，该表不必在选定的工作区中。要将宏保存到另一个工作区内的表中，在指定备注字段

时，应加上表的别名

说明

退出 Visual FoxPro 时，如果没有使用 SAVE MACROS 命令把创建的宏保存到键盘宏文件或备注字段中，就会丢失创建的宏。.

示例

下面的示例中把当前的一组宏保存到一个名为 MYMACROS.FKY 的文件中。

```
SAVE MACROS TO mymacros
```

请参阅

CLEAR, PLAY MACRO, RESTORE MACROS

SAVE SCREEN 命令

把 Visual FoxPro 主窗口或活动的用户自定义窗口的图像保存到屏幕缓冲区、变量或数组元素中。

语法

SAVE SCREEN [TO VarName]

参数描述

TO VarName

指定保存屏幕图像的变量或数组元素。

说明

使用 RESTORE SCREEN 可以重新显示保存在屏幕缓冲区、变量或数组元素中的图像。

当使用 DISPLAY 或 LIST MEMORY 命令查看变量和数组元素时，保存 Visual FoxPro 主窗口或用户自定义窗口图像的变量或数组元素具有 S 数据类型。

不带 TO *VarName* 子句执行此命令时，SAVE SCREEN 把 Visual FoxPro 主窗口或用户自定义窗口保存到屏幕缓冲区中。

请参阅

RESTORE FROM, RESTORE SCREEN, SAVE TO

SAVE TO 命令

把当前变量和数组保存到变量文件或备注字段中。

语法

SAVE TO FileName | MEMO MemoFieldName

[ALL LIKE Skeleton | ALL EXCEPT Skeleton]

参数描述

FileName

指定保存变量和数组的变量文件。变量文件的默认扩展名是 .MEM。

MEMO MemoFieldName

指定保存变量和数组的备注字段。

ALL LIKE Skeleton

指定保存所有满足指定梗概的变量和数组。其中的梗概可以包含问号(?)和星号(*)通配符。

ALL EXCEPT Skeleton

指定保存所有不满足指定梗概的变量和数组。其中的梗概可以包含问号(?)和星号(*)通配符。

说明

使用 RESTORE FROM 命令可从变量文件或变量备注字段中将变量和数组重新放回内存。请注意对象类型变量不能存入变量文件或变量备注字段中。

使用当前代码页标记变量文件或内存字段。

示例

在以下示例中，创建了两个变量。将它们保存在变量文件中并从变量文件中恢复，没有清除已有的变量。

```
gnVal1 = 50
gcVal2 = 'Hello'
```

```
SAVE TO temp
CLEAR MEMORY

gdVal3 = DATE ( )
RESTORE FROM temp ADDITIVE
CLEAR
DISPLAY MEMORY LIKE g*
```

请参阅

PRIVATE, PUBLIC, RELEASE, RESTORE FROM

SAVE WINDOWS 命令

把所有窗口定义或指定的窗口定义保存到窗口文件或备注字段中。

语法

SAVE WINDOWS WindowNameList | ALL
TO FileName | TO MEMO MemoFieldName

参数描述

WindowNameList

指定一个或多个要保存的窗口。窗口名称之间用逗号隔开。

ALL

把所有窗口定义保存到窗口文件或备注字段中。

TO FileName

指定保存窗口定义的窗口文件。

如果命名文件时没有给出扩展名，则指定默认扩展名 .WIN 。如果将窗口定义保存到一个文件时，指定了其他的扩展名，那么将来从文件中恢复窗口定义时，也必须包括该扩展名。

TO MEMO MemoFieldName

指定保存窗口定义的备注字段。包含备注字段的表必须是打开的，但它不必在当前选定工作区中。要把窗口定义保存到另一个工作区内的表中时，应在备注字段前加上表的别名。

说明

使用 RESTORE WINDOW 命令可以从窗口文件或备注字段中恢复窗口定义。保存窗口定义时，也保存各个窗口的状态。例如，如果一个窗口在保存到文件或备注字段中时是隐藏的，那么恢复后它仍然是隐藏的。

示例

在以下示例中，创建了一个名为 wOutput1 的窗口，并将窗口定义保存在文件 Temp.win 中，清除所有窗口并从文件中恢复并激活 wOutput1。

```
CLEAR
DEFINE WINDOW wOutput1 FROM 2,1 TO 13,75 TITLE 'Output' ;
    CLOSE FLOAT GROW ZOOM
ACTIVATE WINDOW wOutput1
```

```
@ 1,1 SAY 'This is the contents of the window'

SAVE WINDOWS wOutput1 TO temp
CLEAR WINDOWS
WAIT WINDOW 'The window has been saved - Press a key'

RESTORE WINDOW wOutput1 FROM temp
ACTIVATE WINDOW wOutput1
WAIT WINDOW 'The window has been restored - Press a key'

DEACTIVATE WINDOW wOutput1
RELEASE WINDOW wOutput1
DELETE FILE temp.win
```

请参阅

DEFINE WINDOW, RESTORE SCREEN, RESTORE WINDOW, SAVE SCREEN

SaveAs 方法

把一个对象作为 .SCX 文件保存起来。

语法

Object.SaveAs(cFileName [, ObjectName])

参数描述

cFileName

指定保存对象的 .SCX 文件。

ObjectName

指定对数据环境对象的引用。

说明

可以使用 SaveAs 方法创建表单或表单集，并把它保存为一个 .SCX 文件。SaveAs 方法只在 Visual FoxPro 交互工作期可用。

在使用 SaveAs 方法的同时，也保存了与对象有关的所有属性、事件和方法。注意，只能保存根据 Visual FoxPro 基类创建的对象。不能保存用户自定义类。

应用于

DEFINE CLASS, 表单, 表单集, _SCREEN

请参阅

SaveAsClass 方法, SET CLASSLIB

SaveAsClass 方法

把对象的实例保存为类库中的类定义。

语法

object.SaveAs.(ClassLibName, ClassName [, Description])

参数描述

ClassLibName

指定保存类定义的 .VCX 文件名。

ClassName

给类指定一个类名。

Description

指定类的 Description 。此参数可选。

说明

当使用 SaveAsClass 方法时，所有与对象有关的属性、事件和方法都作为类定义的一部分保存起来。

应用于

ActiveDoc，复选框，组合框，命令按钮，命令组，容器对象，控件对象，自定义，编

辑框，表单，表单集，表格，图像，标签，线条，列表框，OLE 绑定型控件，OLE 容器控件，选项按钮，选项组，页框，项目挂接对象，_SCREEN，形状，微调，文本框，计时器，工具栏

请参阅

SaveAs 方法，SET CLASSLIB

SAVEPICTURE（） 函数

由一个图片的对象创建一个位图文件 (.BMP)。

语法

SAVEPICTURE(oObjectReference, cFileName)

返值类型

逻辑值

参数描述

oObjectReference

指定一个图片的 object。指定一个图片的 object。

cFileName

指定生成的文件。如果 *cFileName* 中包含了路径，则在该路径中创建一个新文件。如果已有一个同名文件存在，它将被覆盖，而且即便您设置 SET SAFETY 为 ON，在覆盖时也不会有警告信息提示。

说明

图片对象通常由函数 LOADPICTURE() 创建。但是，有些 OLE 对象的属性值也是图片对象，比如：OLE Outline 控件的 PictureOpen 属性。这些图片对象都可用来创建位图文件。

请参阅

LOADPICTURE()

ScaleMode 属性

使用图形方法或定位控件时，指定对象坐标的度量单位。设计时与运行时可用。

语法

object.ScaleMode = nMode

参数描述

nMode

ScaleMode 属性的设置有：

设置	说明
0	Foxel（Visual FoxPro 术语，相当于当前字体字符的最大高度和平均宽度。）
3	像素（默认为显示器或打印机分辨率的最小单位。）

说明

如果您为跨平台使用创建了一个自定义可视类，请确定在可视类中表单的 ScaleMode 属性设置为像素。

应用于

表单，_SCREEN，工具栏

请参阅

DrawMode 属性，DrawStyle 属性

SCAN ... ENDSCAN 命令

在当前选定的表中移动记录指针，并对每一个满足指定条件的记录执行一组命令。

语法

```
SCAN [NOOPTIMIZE]
    [Scope] [FOR lExpression1] [WHILE lExpression2]
        [Commands]
    [LOOP]
    [EXIT]
ENDSCAN
```

参数描述

NOOPTIMIZE

防止 SCAN 的 Rushmore 优化。

有关详细内容，请参阅稍后的“SET OPTIMIZE 命令”和《Microsoft Visual FoxPro 6.0 中文版程序员指南》第十五章“优化应用程序”中的“掌握 Rushmore 技术”。

Scope

指定扫描记录的范围。只有范围之内的记录才可能扫描到。Scope 子句有 ALL、NEXT *nRecords*、RECORD *nRecordNumber* 和 REST。

SCAN 的默认范围是所有记录（ALL）。

FOR lExpression1

只有使表达式 *lExpression1* 计算为“真”(.T.)的记录，才对其执行命令。包含 FOR 子句可以筛选出不想扫描的记录。

当表达式 *lExpression1* 是可优化表达式时，Rushmore 将优化 SCAN ... FOR 命令创

建的查询。为使系统获得最佳性能，应在 FOR 子句中使用可优化的表达式。有关详细内容，请参阅稍后的“SET OPTIMIZE 命令”和《Microsoft Visual FoxPro 6.0 中文版程序员指南》第十五章“优化应用程序”中的“掌握 Rushmore 技术”。

WHILE lExpression2

指定一个逻辑表达式 *lExpression2* 作为执行命令的条件。只要逻辑表达式计算为“真”，就对记录执行命令，直至遇到使表达式不为“真”(.T.) 的记录为止。

Commands

指定要执行的 Visual FoxPro 命令。

LOOP

把控制权直接返回到 SCAN 命令。LOOP 可以放在 SCAN 到 ENDSCAN 之间的任何地方。

EXIT

把程序的控制权从 SCAN ... ENDSCAN 循环语句中交给 ENDSCAN 下面的命令。EXIT 可以放在 SCAN 到 ENDSCAN 之间的任何地方。

ENDSCAN

标志 SCAN 过程的结束。

说明

SCAN 命令自动将记录指针移到下一条满足指定条件的记录，并执行相应的命令块。

可以在 ENDSCAN 同一行的后面加一些注释。注释在程序编译和执行时不起作用。

示例

以下示例使用了 SCAN...ENDSCAN 循环语句显示位于瑞典 (Sweden) 的公司。

```
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'Data\testdata')
USE customer  && 打开 Customer 表
CLEAR

SCAN FOR UPPER(country) = 'SWEDEN'
   ? contact, company, city

ENDSCAN
```

请参阅

DO CASE ... ENDCASE, DO WHILE ... ENDDO, FOR ... ENDFOR

SCATTER 命令

从当前记录中，把数据复制到一组变量或数组中。

语法

```
SCATTER
    [FIELDS FieldNameList
```

| FIELDS LIKE Skeleton | FIELDS EXCEPT Skeleton] [MEMO]
TO ArrayName | TO ArrayName BLANK | MEMVAR | MEMVAR BLANK
| NAME ObjectName [BLANK]

参数描述

FIELDS FieldNameList

指定字段，命令将其内容传送到变量或数组中。如果省略 FIELDS *FieldNameList*，**则传送所有字段。如果在字段列表后放一个关键字** MEMO，**则字段列表中可以包含备注字段。**SCATTER **总是忽略通用和图片字段，即使包括了** MEMO **关键字也是如此。**

FIELDS LIKE Skeleton | FIELDS EXCEPT Skeleton

通过 LIKE **和** EXCEPT **子句，有选择地把字段中的内容传送到变量或数组中。如果包括** LIKE **子句，那么与** *Skeleton* **相匹配的字段被传送到变量或数组中。如果包括** EXCEPT *Skeleton*，**那么除了与** *Skeleton* **相匹配的字段外，其他所有字段都传送到变量或数组中。**

Skeleton **支持通配符。例如，如果要把所有以** A **和** P **字母开头的字段传送到变量或数组中去，可以使用如下命令:**

```
SCATTER FIELDS LIKE A*,P* TO myarray
```

可以同时使用 LIKE **和** EXCEPT **子句，如:**

```
SCATTER FIELDS LIKE A*,P* EXCEPT PARTNO* TO myarray
```

MEMO

指定字段列表中包含了备注字段。默认情况下，SCATTER 不处理备注字段。将很大的备注字段传送到变量或数组时，必须有足够的内存。如果缺乏足够的内存，Visual FoxPro 将产生相应的错误信息。如果某一备注字段太大，内存中装不下，那么该字段及字段列表中的其他备注字段的内容都不会传送。如果没有传送备注字段，那么对应的变量或数组元素设置为“假”(.F.)。

TO ArrayName

指定接受记录内容的数组。从第一个字段起，SCATTER 按顺序将每个字段的内容复制到数组的每个元素中。

如果指定数组的元素比字段数多，则多余数组元素的内容不发生变化。如果指定数组不存在，或者它的元素个数比字段数少，则系统自动创建一个新数组，数组元素与对应字段具有相同的大小和数据类型。

TO ArrayName BLANK

创建一个数组，它的元素与表中字段具有相同大小和数据类型，但没有内容。

MEMVAR

把数据传送到一组变量而不是数组中。SCATTER 为表中每个字段创建一个变量，并把当前记录中各个字段的内容复制到对应的变量中。新创建的变量与对应字段具有相同的名称、大小和数据类型。

如果 SCATTER 命令中包括字段列表，则为字段列表中每个字段都创建一个变量。

要引用与当前表中字段有相同名称的变量，应在变量名前加上限定符 M.。

重要提示 在使用 MEMVAR 时，不要加入 TO。如果加入了 TO，Visual FoxPro 创建一个名为 MEMVAR 的数组。

MEMVAR BLANK

创建一组空变量，每个变量与相应的字段有相同的名称、数据类型以及相同的大小。如果 SCATTER 中包含一字段列表，则为字段列表中的每一个字段创建一个变量。

NAME objectName [BLANK]

它创建一个对象，如果不包含 BLANK 关键字，则对象的属性与表中字段具有相同名称。每个对象属性的值都是表中对应字段的内容。如果包含 BLANK 关键字，则属性为空（请参阅 EMPTY() 函数，以了解不同字段类型所对应的空属性）。对于表中的备注或通用字段不能创建属性。

要引用与打开的表同名的对象属性，应在对象名前加上限定符 M.。例如：

```
USE customer
SCATTER NAME customer
? customer.company      && 返回表的值
? M.customer.company    && 返回对象属性的值
```

说明

SCATTER 和 COPY TO ARRAY 相似。所不同的是 COPY TO ARRAY 把多个记录复制到一个数组，而 SCATTER 只将一个记录复制到数组或一组变量，并且在数组或变量不存在时，SCATTER 能自动创建。

使用 GATHER 命令可以将变量或数组元素中的内容复制到表记录中。

示例

示例 1

此示例使用了 SCATTER 语句创建了一系列基于 test 表中字段的变量。然后为每个字段赋值并向表添加了一个空记录。使用 GATHER 命令将数据复制到表中。

```
CREATE TABLE Test FREE ;
    (object C(10), Color C(16), SqFt n(6,2))

SCATTER MEMVAR BLANK
m.object="Box"
m.Color="Red"
m.SqFt=12.5
APPEND BLANK
GATHER MEMVAR
BROWSE
```

示例 2

此示例使用 SCATTER 命令及其后跟着的 NAME 子句创建了一个具有基于表中字段属性的对象。然后为对象的属性赋值并向表添加一个空记录。使用 GATHER 命令及 NAME 子句将数据复制到表的新记录中。

```
CREATE TABLE Test FREE ;
    (Object C(10), Color C(16), SqFt n(6,2))

SCATTER NAME oTest BLANK
oTest.object="Box"
oTest.Color="Red"
oTest.SqFt=12.5
```

```
APPEND BLANK
GATHER NAME oTest
RELEASE oTest
BROWSE
```

请参阅

ALINES（）, APPEND FROM ARRAY, COPY TO ARRAY, DECLARE, DIMENSION, GATHER

SCCProvider 属性

项目的源代码管理器提供程序的名称。设计时只读。

语法

Object.SCCProvider

说明

可以在“选项”对话框的“项目”选项卡中为新项目指定一个源代码管理器提供程序。如果该项目不在源代码管理下，则 SCCProvider **属性包含空字符串。**

有关执行项目的源代码管理器的详细内容，请参阅《Microsoft Visual FoxPro 6.0 **中文版程序员指南》第二十九章“集体开发”中的“在** Visual FoxPro **中使用源代码管理**

软件”。

应用于

项目对象

请参阅

SCCStatus 属性

SCCStatus 属性

包含一个数值，该数值表明项目中一个文件的源代码管理状态。设计和运行时只读。

语法

Object.SCCStatus

说明

下表列出了 SCCStatus 属性可以包含的值：

值	FoxPro.h 常数	说明
0	SCCFILE_NOTcontrolLED	文件没有在源代码管理下。
1	SCCFILE_NOTCHECKED OUT	文件在源代码管理下，但是没有签出。

续表

2	SCCFILE_CHECKEDOUTCU	文件签出到当前用户。
3	SCCFILE_CHECKEDOUTOU	文件签出到非当前用户的某人。
4	SCCFILE_MERGECONFLICT	文件有一个合并冲突。
5	SCCFILE_MERGE	文件在冲突的情况下已经合并了。
6	SCCFILE_CHECKEDOUTMU	文件签出到多个用户。

应用于

文件对象

请参阅

SCCProvider 属性

_SCCTEXT 系统变量

指定 Visual FoxPro 中用于二进制文件和文本文件相互转换的程序。

语法

_SCCTEXT = *cProgramName*

参数描述

cProgramName

指定源代码控制转换程序。如果您的源代码控制转换程序不在当前默认的目录中，您还应指定文件路径。

您还可以如下在配置文件中包含一行指定源代码控制转换程序。

_SCCTEXT = cProgramName

说明

默认地，_SCCTEXT 中保存 SCCTEXT.PRG。该程序完成 Visual FoxPro 中二进制文件和文本文件的相互转换。在源代码控制中，这些文本文件作为比较和合并不同版本的二进制文件的标准。

请参阅

Projects Tab, 选项对话框

SCHEME( ) 函数

返回指定配色方案中的颜色对列表或单个颜色对。

语法

SCHEME(nSchemeNumber [, nColorPairNumber])

返值类型

字符型

参数描述

nSchemeNumber

指定需要颜色对列表的配色方案编号。SCHEME（） 返回十个颜色对。

NColorPairNumber

指定配色方案中一个颜色对的位置。SCHEME（） 只返回这个颜色对。例如，如果 *nColorPairNumber* 等于 4，则 SCHEME（） 返回配色方案中第四个颜色对。

示例

下面的示例返回第四个配色方案中的第三个颜色对：

```
? SCHEME(4,3)
```

请参阅

颜色概览

SCOLS（） 函数

返回 Visual FoxPro **主窗口中可用列数。**

语法

SCOLS（）

返值类型

数值型

说明

该函数的返回值取决于当前显示模式。显示模式可以使用 SET DISPLAY **命令更改。**

请参阅

COL（）,ROW（）,SET DISPLAY,SROWS（）,WCOLS（）,WROWS（）

_SCREEN 系统变量

指定 Visual FoxPro 主窗口的属性和方法。

语法

_SCREEN.*property*Name [= eValue]

–或者–

_SCREEN.methodName

参数描述

PropertyName
为主窗口指定一个属性。

EValue
指定属性的值。

MethodName 指定 Visual FoxPro 主窗口要执行的方法。

说明

_SCREEN 允许把 Visual FoxPro 主窗口作为一个对象来处理。但不能为 _SCREEN 创建事件过程。

_SCREEN 是一个对象类型的系统变量。

属性

Activecontrol	ActiveForm	Application
AutoCenter	BackColor	BaseClass
BorderStyle	BufferMode	Caption
Class	ClassLibrary	Clipcontrols
Closable	Comment	ControlBox
ControlCount	Controls	CurrentX
CurrentY	DataSession	DataSessionID
DefOLELCID	Desktop	DrawMode
DrawStyle	DrawWidth	Enabled
FillColor	FillStyle	FontBold
FontItalic	FontName	FontOutline
FontShadow	FontSize	FontStrikeThru
FontUnderline	Forecolor	FormCount
Forms	HalfHeightCaption	Height
HelpContextID	Icon	KeyPreview
Left	LockScreen	MaxButton
MaxHeight	MaxLeft	MaxTop
MaxWidth	MDIForm	MinButton
MinHeight	MinWidth	MousePointer

续表

Movable	Name	ParentClass
Picture	ReleaseType	RightToLeft
ScaleMode	ShowTips	TabIndex
TabStop	Tag	Top
Visible	Width	WindowState
WindowType		

方法

Addobject	Addproperty	Box
Circle	Cls	Draw
Hide	Line	Move
Point	Print	Pset
Refresh	Release	Removeobject
SaveAs	SaveAsClass	SetAll
Show	TextHeight	TextWidth
ZOrder		

示例

下例演示了如何用_SCREEN **命令来指定** Visual FoxPro window **主窗口的属性。**

```
* 保存现设置的变量
Local oldScreenLeft
```

```
Local oldScreenTop
Local oldScreenHeight
Local oldScreenWidth
Local oldScreenColor
WITH _Screen
    oldScreenLeft=.Left          && 保存现位置和尺寸
    oldScreenTop=.Top
    oldScreenHeight=.Height
    oldScreenWidth=.Width
    oldScreenColor = .Backcolor
    .LockScreen=.T.          && 使屏幕刷新不可用
    .BackColor=rgb(192,192,192)    && 把背景颜色改为灰色
    .BorderStyle=2           && 把边框改为双线
    .Closable=.F.            && 删除 window 控件按扭
    .ControlBox=.F.
    .MaxButton=.F.
    .MinButton=.T.
    .Movable=.T.
    .Height=285
    .Width=550
    .Caption="Custom Screen"      && 设置说明
    .LockScreen=.F.          && 使屏幕刷新不可用
ENDWITH
=MESSAGEBOX("Return to normal   ",48,WTITLE())
With _Screen
    .Left = oldScreenLeft       && 重设初始值
    .Top = oldScreenTop           && 位置和大小
    .Height = oldScreenHeight
    .Width   = oldScreenWidth
    .BackColor=oldScreenColor       && 把背景颜色改为白色
```

```
    .LockScreen=.T.          && 使屏幕刷新不可用
    .BorderStyle=3           && 把边框改为随机改变大小
    .Closable=.T.            && 重设 window 控件按扭
    .ControlBox=.T.
    .MaxButton=.T.
    .MinButton=.T.
    .Movable=.T.
    .Caption="Microsoft Visual FoxPro"   &&      重设说明
    .LockScreen=.F.          && 恢复屏幕刷新
Endwith
```

请参阅

MODIFY WINDOW, Form **对象**

SCROLL **命令**

将 Visual FoxPro **主窗口或用户自定义窗口的某一区域向上、向下、向左或向右滚动。**

语法

```
SCROLL nRow1, nColumn1, nRow2, nColumn2, nRowsScrolled
     [, nColumnsScrolled]
```

参数描述

nRow1, nColumn1, nRow2, nColumn2

指定在 Visual FoxPro 主窗口或活动用户定义的窗口中可以滚动的长方形区域。nRow1, nColumn1 指定了区域的左上角，nRow2, nColumn2 指定区域的右下角。

nRowsScrolled

指定在长方形区域上下滚动的行数。如果数值表达式 nRowsScrolled 为正值，指定 Visual FoxPro 向上滚动的行数。如果 nRowsScrolled 为负值，指定 Visual FoxPro 向下滚动的行数。如果 nRowsScrolled 为 0 并省略了 nColumnsScrolled 参数，Visual FoxPro 清除长方形区域。

nColumnsScrolled

指定在长方形区域左右滚动的列数。如果数值表达式 nColumnsScrolled 为正值，指定 Visual FoxPro 向右滚动的列数。如果 nColumnsScrolled 为负值，指定 Visual FoxPro 向左滚动的列数。如果您包含 nRowsScrolled 和 nColumnsScrolled 这两个参数，Visual FoxPro 斜向滚动区域。

示例

以下命令滚动一个小的长方形区域。

```
CLEAR
@ 4, 1 FILL TO 10, 8 COLOR GR+/B
WAIT WINDOW 'Press key to scroll left top corner'
SCROLL 0, 0, 5, 5, -2, 1
```

请参阅

MOVE WINDOW

ScrollBars 属性

指定编辑框、表单或表格所具有的滚动条类型。设计时可用，运行时可读写。

语法

[Form.]control.ScrollBars[= nType]

参数描述

nType

对表格， ScrollBars **的设置有：**

设置	说明
0	无（默认值）
1	水平滚动条
2	垂直滚动条
3	水平滚动条和垂直滚动条

对表单， ScrollBars **的设置有：**

设置	说明
0	无
1	水平滚动条
2	垂直滚动条
3	（默认值）水平滚动条和垂直滚动条

对编辑框，ScrollBars 的设置有：

设置	说明
0	无
2	垂直滚动条

说明

如果启用了此属性，则当表格、表单或编辑框显示不下它所包含的内容时自动出现滚动条。如果一个表单是 Active Document，则当 Active Document 容器比表单小时，自动显示滚动栏。

如果一个 Active Document 显示了一个具有滚动栏的表单时，当视口比包围表单上空间的矩形小时，会显示滚动栏。_SCREEN 忽略 Scrollbars 属性。

应用于

编辑框，表格

请参阅

ContinuousScroll **属性**, DoScroll **方法**, Scrolled **事件**

Scrolled **事件**

在表格控件或表单中，单击水平或垂直滚动条，或移动滚动条中的滚动块时，此事件发生。

语法

```
PROCEDURE object.Scrolled
LPARAMETERS [nIndex], nDirection
```

参数描述

nIndex

唯一标识控制数组中的某个控制。

nDirection

指定用户如何滚动表格控制中的内容。可能的参数有：

参数	**说明**
0	**使用上箭头键**

续表

1	使用下箭头键
2	单击垂直滚动条滚动块上面的区域
3	单击垂直滚动条滚动块下面的区域
4	使用左箭头键
5	使用右箭头键
6	单击水平滚动条滚动块左边的区域
7	单击水平滚动条滚动块右边的区域

说明

如果在程序代码中调用 DoScroll 方法，也发生 Scrolled 事件。Scrollbars 属性决定了一个表单是否具有滚动栏。

应该避免在 Scrolled 事件中制造等待状态（例如，WAIT WINDOW），因为当滚动表格或表单时会出现屏幕刷新问题。

应用于

表单，表格

请参阅

ActiveColumn 属性，ActiveRow 属性，DoScroll 方法，RelativeColumn 属性，RelativeRow 属性，ScrollBars 属性

SEC（） 函数

返回日期时间表达式中的秒。

语法

SEC(tExpression)

返值类型

数值型

参数描述

tExpression

指定的日期时间型表达式，SEC（） 函数返回该表达式的秒钟部分。 如果 *tExpression* 包含的只是日期而没有时间，Visual FoxPro 把默认的午夜时间（12: 00: 00 A.M.）加入到 *tExpression* 中。

示例

下面的示例显示当前时间和指定时间的秒钟部分。

```
CLEAR
? SEC(DATETIME（）)
? HOUR({1998-08-21，10:42:16am})  && 显示 16
```

请参阅

DATETIME（），SET SECONDS

SECONDS（） 函数

以秒为单位，返回自午夜以来经过的时间。

语法

SECONDS（）

返值类型

数值型

说明

SECONDS（） **以十进制格式返回一个数值，精度为** 1 **毫秒。如果在** Windows NT **上运行，则精度为** 10 **毫秒。**

示例

```
CLEAR
?SECONDS（）
?SECONDS（） /(60*60)
```

请参阅

SYS(2), TIME()

Seconds 属性

指定在文本框中是否显示一个日期时间值的第二部分。设计和运行时可用。

语法

object.Seconds[= nValue]

参数描述

nValue

取下列设置之一：

设置	说明
0	不。不显示一个日期时间值的第二部分。
1	是。显示一个日期时间值的第二部分。

续表

2	（默认值）SET SECONDS 的设置决定是否显示一个日期时间值的第二部分。如果 SET SECONDS 是 ON，则显示一个日期时间值的第二部分。如果 SET SECONDS 为 OFF，则不显示一个日期时间值的第二部分。

说明

如果 DateFormat 属性的设置为 Short 或 Long，则忽略 Seconds 属性的设置。

应用于

文本框

请参阅

Century 属性，DateFormat 属性，DateMark 属性，Hours 属性，SETSECONDS，StrictDateEntry 属性

SEEK 命令

SEEK 在表中搜索首次出现的一个记录，这个记录的索引关键字必须与指定的表达式匹配。

语法

SEEK eExpression
[ORDER nIndexNumber | IDXIndexFileName
| [TAG] TagName [OF CDXFileName]
[ASCENDING | DESCENDING]]
[IN nWorkArea | cTableAlias]

参数描述

eExpression
指定 SEEK 搜索的索引关键字。*eExpression* 可以是空字符串。

ORDER nIndexNumber
指定用来搜索关键字的索引文件或索引标识编号。*nIndexNumber* 指出了索引文件在 USE 和 SET INDEX 命令中列出的编号。首先，按照 USE 或 SET INDEX 中的顺序对打开的 .IDX 文件进行编号。然后，对结构 .CDX 文件（如果存在）中的标识进行编号，其顺序与创建它们的顺序相同。最后，对所有打开的独立 .CDX 文件进行编号，其顺序也与创建的顺序相同。有关索引编号的详细内容，请参阅 SET ORDER。

ORDER IDXIndexFileName
指定一个用来搜索关键字的 .IDX 文件。

ORDER [TAG] TagName [OF CDXFileName]
指定用来搜索索引关键字的 .CDX 文件中的标识。标识名称可能在一个结构 .CDX 文件中，也可以在任何其他打开的独立 .CDX 文件中。

如果在几个打开的独立 .CDX 文件中存在相同标识名称，则应使用 OF CDXFileName 指出包含所用标识的 .CDX 文件。

注意 如果存在相同的 .IDX 文件和标识名称时，.IDX 文件具有优先权。

ASCENDING

指定按升序搜索表。

DESCENDING

指定按降序搜索表。

IN nWorkArea

指定要搜索的表所在的工作区编号。

IN cTableAlias

指定要搜索的表的别名。

如果省略了 IN *nWorkArea* 和 IN *cTableAlias*，则在当前选定的工作区中搜索。

说明

只能在索引过的表中使用 SEEK 命令，并且只能搜索索引关键字。除非 SET EXACT 的设置为 OFF，否则匹配指的是完全匹配。

如果 SEEK 找到了与索引关键字相匹配的记录，则 RECNO() 返回匹配记录的记录号；FOUND() 返回“真”(.T.)；EOF() 返回“假”(.F.)。

如果找不到相匹配的关键字，则 RECNO() 将表中记录的个数加 1，然后返回。FOUND() 返回“假”(.F.)；EOF() 返回“真”(.T.)。

如果 SET NEAR 设置为 ON，则记录指针指向与索引关键字最相匹配的那个记录的后面一个记录。如果 SET NEAR 设置为 OFF，则记录指针指向文件尾。在这两种

情况下，RECNO（） 都返回与关键字最匹配的记录号。

示例

在下列示例中，打开了 customer 表并基于 company 字段建立了索引。使用 SEEK 命令查找索引表达式，该表达式匹配包含在变量 gcSeekVal 中的值。

```
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'Data\testdata')
USE customer ORDER company   && 打开 Customer 表

SET EXACT OFF
STORE 'B' TO gcSeekVal

SEEK gcSeekVal

IF FOUND（）
   DISPLAY FIELDS company, contact
ENDIF
```

请参阅

EOF（），FOUND（），INDEX，INDEXSEEK（），LOCATE，RECNO（），SEEK（），SETEXACT ，SET NEAR

SEEK（） 函数

在一个已建立索引的表中搜索一个记录的第一次出现位置，该记录的索引关键字与指定表达式相匹配。SEEK（） 函数返回一个逻辑值，指示搜索是否成功。

语法

SEEK(eExpression [, nWorkArea | cTableAlias
[, nIndexNumber | cIDXIndexFileName | cTagName]])

返值类型

逻辑值

参数描述

eExpression

指定 SEEK（） 函数搜索的索引关键字表达式。

nWorkArea

指定要在其中搜索索引关键字的表所在工作区的编号。

cTableAlias

指定要搜索的表的别名。

如果省略了 *nWorkArea* 或 *cTableAlias* ，则在当前工作区中搜索表。

nIndexNumber

指定用来搜索关键字的索引文件或索引标识编号。*nIndexNumber* 指出了索引文件在 USE 和 SET INDEX 命令中列出的编号。首先，按照 USE 或 SET INDEX 中的顺序对打开的 .IDX 文件进行编号。然后，对结构 .CDX 文件（如果存在）中的标识进行编号，其顺序与创建它们的顺序相同。最后，对所有打开的独立 .CDX 文件进行编号，其顺序也与创建的顺序相同。有关索引编号的详细内容，请参阅 SET ORDER。

cIDXIndexFileName

指定用来搜索索引关键字的 .IDX 文件。

cTagName

指定用来搜索索引关键字的 .CDX 文件的标识。标识名称可以来自结构文件 .CDX，也可以来自任何打开的独立 .CDX 文件。

注意 如果存在相同的 .IDX 文件和标识名称，优先使用 .IDX 文件。

说明

只能对设置了索引排序的表使用 SEEK() 函数，并且只能搜索索引关键字。除非 SET EXACT 的设置为 OFF 时，否则所指的匹配是完全相匹配。

如果找到了匹配的记录，则 SEEK() 函数返回“真”(.T.)，记录指针指向找到的记录。如果找不到匹配记录，SEEK() 函数返回“假”(.F.)。此时记录指针移到文件尾。执行 SEEK() 函数同先执行 SEEK 命令然后执行 FOUND() 函数的结果一样。

如果省略了 *nIndexNumber*，*IDXIndexFileName* 和 *cTagName* 参数，那么 SEEK()

函数使用主控索引或索引标识来搜索索引关键字。

示例

```
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'Data\testdata')
USE customer ORDER cust_id   && 打开 Customer 表
? SEEK('CHOPS')   && 返回 .T., 找到记录
```

请参阅

EOF（）,FOUND（）,INDEXSEEK（）,LOCATE,SEEK,SET ORDER

SELECT 命令

激活指定工作区。

语法

SELECT nWorkArea | cTableAlias

参数描述

nWorkArea

指定要激活的工作区。如果 *nWorkArea* 为零，则激活尚未使用的工作区中编号最小的那一个。

cTableAlias

指定要激活的、包含打开表的工作区。*cTableAlias* 是打开表的别名。也可以用从 A 到 J 中的一个字符作为 *cTableAlias* 来激活前 10 个工作区中的一个。

说明

默认情况下，启动 Visual FoxPro 时打开编号为 1 的工作区。

注意 在任何工作区中打开的表字段均可以在 Visual FoxPro 的命令和函数中使用。访问非当前工作区中打开表的字段时，应使用如下格式：

alias.field 或者 alias -> field。

示例

以下示例演示了选择工作区的方法。

```
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'Data\testdata')

SELECT 1      && 工作区 1
USE customer  && 打开 Customer 表

SELECT 2      && 工作区 2
USE orders    && 打开 Orders 表

SELECT customer      && 工作区 1
BROWSE

SELECT B      && 工作区 2
BROWSE
```

请参阅

ALIAS（），SELECT（），USE

SELECT – SQL 命令

从一个或多个表中检索数据。

语法

```
SELECT [ALL | DISTINCT] [TOP nExpr[PERCENT]]
    [Alias.] Select_Item [AS Column_Name]
    [, [Alias.] Select_Item [AS Column_Name] ...]
FROM [FORCE]
[DatabaseName!]Table [[AS] Local_Alias]
    [[INNER | LEFT [OUTER] | RIGHT [OUTER] | FULL [OUTER] JOIN
        DatabaseName!]Table [[AS] Local_Alias]
        [ON JoinCondition ... ]
[[INTO Destination]
    | [TO FILE FileName [ADDITIVE] | TO PRINTER [PROMPT]
```

```
    | TO SCREEN]]
 [PREFERENCE PreferenceName]
 [NOCONSOLE]
 [PLAIN]
 [NOWAIT]
 [WHERE JoinCondition [AND JoinCondition ...]
    [AND | OR FilterCondition [AND | OR FilterCondition ...]]]
 [GROUP BY GroupColumn [, GroupColumn ...]]
 [HAVING FilterCondition]
 [UNION [ALL] SELECTCommand]
 [ORDER BY Order_Item [ASC | DESC] [, Order_Item [ASC | DESC] ...]]
```

参数描述

SELECT

在 SELECT 子句中指定在查询结果中包含的字段、常量和表达式。

ALL

查询结果中包含所有行（包括重复值）。ALL 是默认设置。

DISTINCT

在查询结果中剔除重复的行。

注意　每一个 SELECT 子句只能使用一次 DISTINCT。

TOP nExpr[PERCENT]

在符合查询条件的所有记录中，选取指定数量或百分比的记录。TOP 子句

必须与 ORDER BY 子句同时使用。ORDER BY 子句指定查询结果中包含的列上由 Top 字句决定的行数，TOP 子句根据此排序选定最开始的 *nExpr* 个或 *nExpr*% 的记录。

您可以指定选取 1 到 32767 个记录。使用 ORDER BY 子句指定的字段进行排序，会产生并列的情况，比如，可能有多个记录，它们在选定的字段上相同；所以，如果您指定 *nExpr* 为 10，在查询结果中可能多于 10 个记录，因为可能有几个记录位置并列。

如果包含 PERCENT 关键字指定查询结果中的记录数，得到记录数的可能是小数，这时进行取整。包含 PERCENT 关键字时，*nExpr* 的范围是 0.01 到 99.99。

Alias.

限定匹配项的名称。*Select_Item* 指定的每一项在查询结果中都生成一列。如果多个项具有相同的名称，则应在这些项名前加上表的别名和一个句点，以防止出现重复的列。

Select_Item 指定包括在查询结果中的项。一个项可以是：

FROM 子句所包含的表中的字段名称。

一个常量，查询结果中每一行都出现这个常量值。

一个表达式，可以是用户自定义函数名。

AS Column_Name

指定查询结果中列的标题。当 *Select_Item* 是一个表达式或一个字段函数时，如果要给此列取一个有含义的名称，一般可以使用这个子句。*Column_Name*

可以是一个表达式，但不能包含那些表字段名称中不允许出现的字符（如空格）。

FROM

列出所有从中检索数据的表。如果没有打开表，Visual FoxPro 显示“打开”对话框以便指定文件位置。表打开以后，直到查询结束时才关闭。

如果您包含 FORCE 关键字，Visual FoxPro 在建立查询时会严格按照您在 FROM 子句中声明的顺序连接表；若不包含 FORCE 关键字，Visual FoxPro 会试图对查询进行优化。使用 FORCE 子句，避免了优化过程，可能加快查询执行的速度。

DatabaseName!

当包含表的数据库不是当前数据库时，*DatabaseName*! 指定这个数据库的名称。如果数据库不是当前数据库，就必须指定包含表的数据库名称。应在数据库名称之后表名之前加上感叹号（!）分隔符。

[AS] Local_Alias

为 *Table* 中的表指定一个临时名称。如果指定了本地别名，那么在整个 SELECT 语句中必须都用这个别名代替表名。本地别名不影响 Visual FoxPro 环境。INNER JOIN 只有在其他表中包含对应记录（一个或多个）的记录才出现在查询结果中。

LEFT [OUTER] JOIN 在查询结果中包含：JOIN 左侧表中的所有记录，以及 JOIN 右侧表中匹配的记录。OUTER 关键字可被省略；包含 OUTER 强调这是一个外连接 (outer join)。

RIGHT [OUTER] JOIN 在查询结果中包含：JOIN 右侧表中的所有记录，以及 JOIN 左侧表中匹配的记录。OUTER 关键字可被省略；包含 OUTER 强调这是一个外连接接 (outer join)。

FULL [OUTER] JOIN 在查询结果中包含：JOIN 两侧所有的匹配记录，和不匹配的记录；包含 OUTER 强调这是一个外连接 (outer join)。

ON JoinCondition 指定连接条件。

INTO Destination

指定在何处保存查询结果。如果在同一个查询中同时包括了 INTO 子句和 TO 子句，则 TO 子句不起作用。如果没有包括 INTO 子句，查询结果显示在“浏览”窗口中。也可以用 TO 将查询结果定向输出到打印机或文件。

Destination 可以是下列子句之一：

ARRAY *ArrayName* ，将查询结果保存到变量数组中。如果查询结果中不包含任何记录，则不创建这个数组。

CURSOR *CursorName* 将查询结果保存到临时表中。如果指定了一个已打开表的名称，则 Visual FoxPro 产生错误信息。执行完 SELECT 语句后，临时表仍然保持打开、活动但只读。一旦关闭临时表，则自动删除它。临时表作为 SORTWORK 指定驱动器上的一个临时文件存在。

包含 NOFILTER 是为了创建一个能用在后来的询问中的指针。在 Visual FoxPro 以前的版本需要一个额外的常量或表达式作为过滤器去创建一个能用在后来的询问中的指针。

SELECT *, .T. FROM customers INTD CURSOR myquery

包含NOFILTER能够减少询问的发生是因为临时表是建立在磁盘上的，当临时表被删除，指针就要被关闭。

DBF *TableName* | TABLE *TableName*，将查询结果保存到一个表中。如果指定的表已经打开，并且 SET SAFETY 设置为 OFF，则 Visual FoxPro 在不给出警告的情况下改写该表。如果没有指定扩展名，Visual FoxPro 指定表的扩展名是 .DBF。SELECT 语句执行结束后，表仍然保持打开活动状态。

包含 DATABASE DatabaseName 以指定添加了表的数据库。包含 NAME LongTableName 可以为该表命一个最多可包括128个字符的并且可以在数据库中代替短名字的长名。

TO FILE FileName

如果命令中包括了 TO 子句，但没有包括 INTO 子句，则查询结果定向输出到名为 *FileName* 的 ASCII 码文件、打印机或 Visual FoxPro 主窗口。

ADDITIVE 把查询结果定向输出到由TO FILE FileName*指定的文本文件的现存目录上*。

TO PRINTER [PROMPT] 使查询结果定向输出到打印机。在打印开始之前，使用可选的 PROMPT 子句显示一个对话框。您可以根据当前安装的打印机驱动程序调整打印机的设置。将 PROMPT 子句放置在紧跟 TO PRINTER 之后。

TO SCREEN 使查询结果定向输出到 Visual FoxPro 主窗口或活动的用户自定义窗口中。

PREFERENCE PreferenceName

如果查询结果送往浏览窗口，就可以使用 PREFERENCE 保存浏览窗口的属性和选项以备后用。PREFERENCE 把特征属性或参数选项长期保存在 FOXUSER 的资源文件中，任何时侯都可以对它们进行检索。

第一次执行有 PREFERENCE *PreferenceName* 的 SELECT 命令时创建参数选项。以后执行有相同参数选项名的 SELECT 命令时便将浏览窗口恢复到原来的参数选项状态。当浏览窗口关闭时，更新参数选项。

如果您按下 CTRL+Q+W 键退出“浏览”窗口，您对“浏览”窗口所做的更改不会保存到资源文件中。

NOCONSOLE

不显示送到文件、打印机或 Visual FoxPro 主窗口的查询结果。

PLAIN

防止列标题出现在显示的查询结果中。不管有无 TO 子句都可使用 PLAIN 子句。如果 SELECT 语句中包括 INTO 子句，则忽略 PLAIN 子句。

NOWAIT

打开浏览窗口并将查询结果输出到这个窗口后继续程序的执行。程序并不等待关闭浏览窗口，而是立即执行紧接在 SELECT 语句后面的程序行。

SELECT 命令中包括 TO SCREEN 可以把查询结果定向输出到 Visual FoxPro 主窗口或用户自定义窗口。如果显示时 Visual FoxPro 主窗口或用户自定义窗口中写满了一屏，就暂停输出。按任意键可以查看查询结果后面的内容。但是，如果命令中包括了 NOWAIT 子句，显示查询结果时就不会暂停，等待按键，而是在 Visual FoxPro 主窗口或用户自定义窗口中连续滚过所有内容。如果命令中包含有 INTO 子句，忽略 NOWAIT 子句。

WHERE

通知 Visual FoxPro 在查询结果中仅包含一定数目的记录。如果要从多个表中检索数据，WHERE 子句是必需的。

JoinCondition

指定一个字段，该字段连接 FROM 子句中的表。如果查询中包括不止一个表，就应该为第一个表后的每一个表指定连接条件。

连接多个查询条件必须使用操作符 AND。每个连接条件都有下面的形式：

FieldName1 Comparison FieldName2

其中 *FieldName1* 是一个表中的字段名，*FieldName2* 是另一表中的字段名，*Comparison* 是下表中列出的某一操作符。

操作符	比较关系
=	相等
==	完全相等
LIKE	SQL LIKE
<>, !=, #	不相等
>	大于
>=	大于等于
<	小于
<=	小于等于

对字符串使用 = 操作符时，所得结果与 SET ANSI 的设置有关。当 SET ANSI 设置为 OFF 时，Visual FoxPro 比较字符串的方式与 Xbase 的用户相同。当 SET ANSI 的设置为 ON 时，Visual FoxPro 比较字符串时遵守 ANSI 标准。有关 Visual FoxPro 字符比较方法的详细内容，请参阅 SET ANSI 和 SET EXACT 。

WHERE 子句支持 ESCAPE 操作符，允许执行对包含有 SELECT – SQL % 和通配符的数据的有意义的查询。

ESCAPE 子句允许指定一个可以看作文字字符的 SELECT – SQL 通配符。 ESCAPE 子句允许指定一个字符，一旦它被放到通配符字符之前，就表示这个通配符被看作一个文字字符。

FilterCondition

指定将包含在查询结果中记录必须符合的条件。使用 AND 或 OR 操作符，您可以包含随意数目的过滤条件。您还可以使用 NOT 操作符将逻辑表达式的值取反，或使用 EMPTY（） 函数以检查空字段。

FilterCondition 可以是下面示例中的任何一种形式：

示例 1

事例 1 显示 *FieldName1 Comparison FieldName* 窗体中的 Fliter Condition

```
customer.cust_id = orders.cust_id
```

示例 2

事例 1 显示 *FieldName Comparison Expression* 窗体中的 Fliter Condition

```
payments.amount >= 1000
```

示例 3

事例 3 显示 *FieldName Comparison* ALL (*Subquery*) 窗体中的 Fliter Condition

当筛选条件包括 ALL 时，只有指定字段满足所有子查询结果后，它所在的记录才能添加到查询结果中。

```
company < ALL ;

(SELECT company FROM customer WHERE country = “UK")
```

示例 4

事例 4 显示 *FieldName Comparison* ANY | SOME (*Subquery*) **窗体中的** Fliter Condition

当筛选条件包含 ANY **或** SOME **时，字段必须至少满足一个由子查询产生的值所决定的比较条件。**

```
company < ANY ;
(SELECT company FROM customer WHERE country = “UK")
```

示例 5

事例 5 **显示** *FieldName* [NOT] BETWEEN *Start_Range* AND *End_Range* **窗体中的** Fliter Condition

```
customer.postalcode BETWEEN 90000 AND 99999
```

上面的示例检查字段中的值是否在指定范围内。

示例 6

事例 6 **显示** [NOT] EXISTS (*Subquery*) **窗体中的** Fliter Condition

```
EXISTS ;
(SELECT * FROM orders WHERE customer.postalcode =
orders.postalcode)
```

上面的示例检查是否至少有一行满足子查询中的条件。当筛选条件包括 EXISTS **时，只要子查询不为空集，筛选的条件就为“真”** (.T.)。

示例 7

事例 7 **显示** *FieldName* [NOT] IN *Value_Set* **窗体中的** Fliter Condition

```
customer.postalcode NOT IN ("98052”,”98072”,”98034")
```

当筛选条件中包含 IN **时，把字段所在记录添加到查询结果中的条件是字段必**

须包含值集合的一个元素。

示例 8

事例 8 显示 *FieldName*[NOT] IN (*Subquery*) 窗体中的 Fliter Condition

```
customer.cust_id IN ;
(SELECT orders.cust_id FROM orders WHERE
orders.city="Seattle")
```

这里，记录包含在查询结果中的条件是：字段必须包含一个子查询的返回值。

示例 9

事例 9 显示 *FieldName*[NOT] LIKE *cExpression* 窗体中的 Fliter Condition

```
customer.country NOT LIKE“UK”
```

这个筛选条件查找每个与字符串表达式相匹配的字段。在字符串表达式中可以使用百分号 (%) 和下划线 (_) 通配符。下划线表示字符串中一个任意字符。

GROUP BY GroupColumn[, GroupColumn ...]

按列的值对查询结果的行进行分组。*GroupColumn* 可以是常规的表字段名，也可以是一个包含 SQL 字段函数的字段名，还可以是一个数值表达式，指定查询结果表中的列位置（最左边的列编号为 1）。

HAVING FilterCondition

指定包括在查询结果中的组必须满足的筛选条件。HAVING 应该同 GROUP BY 一起使用。它能包含数量不限的筛选条件，筛选条件用 AND 或 OR 连接，还可以使用 NOT 来对逻辑表达式求反。

FilterCondition 不能包括子查询。

使用 HAVING 子句的命令如果没有使用 GROUP BY 子句，则它的作用与

WHERE 子句相同。可以在 HAVING 子句中使用本地别名和字段函数。如果 HAVING 子句不包含字段函数的话，使用 WHERE 子句可以获得较快的速度。

[UNION [ALL] SELECT 命令]

把一个 SELECT 语句的最后查询结果同另一个 SELECT 语句最后查询结果组合起来。默认情况下，UNION 检查组合的结果并排除重复的行。要组合多个 UNION 子句，可使用括号。

ALL 防止 UNION 删除组合结果中重复的行。

UNION 子句遵守下列规则：

不能使用 UNION 来组合子查询。

两个 SELECT 命令的查询结果中的列数必须相同。

两个 SELECT 查询结果中的对应列必须有相同的数据类型和宽度。

只有最后的 SELECT 中可以包含 ORDER BY 子句，而且必须按编号指出所输出的列。如果包含了一个 ORDER BY 子句，它将影响整个结果。

ORDER BY Order_Item

根据列的数据对查询结果进行排序。每个 *Order_Item* 都必须对应查询结果中的一列。

它可以是下列之一：

FROM 子句中表的字段，同时也是 SELECT 主句（不在子查询中）的一个选择项。

一个数值表达式，表示查询结果中列的位置（最左边列编号为 1）。

ASC 指定查询结果根据排序项以升序排列。它是 ORDER BY 的默认选项。

DESC 指定查询结果以降序排列。

如果不使用 ORDER BY 指定查询结果的排列顺序，则查询结果不排序。

说明

同其他 Visual FoxPro 命令一样，SELECT 是 Visual FoxPro 的一个内部 SQL 命令。当使用 SELECT 进行查询时，Visual FoxPro 先解释查询要求，然后从表中查询并检索指定数据。可以在下列环境内建立 SELECT 查询：

- 命令窗口
- 一个 Visual FoxPro 程序（与使用其他 Visual FoxPro 命令一样）
- 查询设计器

如果执行了 SET TALK ON 命令后再执行 SELECT 命令，则 Visual FoxPro 显示出执行查询所用的时间和查询结果中记录的数目。_TALLY 包含了查询结果中记录的数目。

SET FILTER 设置的筛选条件对 SELECT 命令不起作用。

下面部分将多次提到子查询 (subquery)，子查询是指在 SELECT 命令中包含的 SELECT 命令。在 SELECT 命令的 WHERE 子句中可以包含最多两个平级的（非嵌套）的子查询。子查询中可以有多个连接条件 (join conditions)

创建查询输出时，列的命名遵循如下规则：

- 如果选择项是具有唯一名称的字段，则用字段名作为输出列名。
- 如果多个选择项具有相同名称。例如，如果名为 Customer 的表有一个 STREET 字段，而名为 Employees 的表也有一个 STREET 字段，则输出

列命名为 *Extension*_A 和 *Extension*_B（STREET_A 和 STREET_B）。如果选择项名称有 10 字符长，可以将名称截短后再加下划线和字母。例如，DEPARTMENT 变为 DEPARTME_A。

- 如果选择项是表达式，它的输出列命名为 EXP_A。其他表达式分别命名为 EXP_B、EXP_C，依此类推。
- 如果选择项包含诸如 COUNT() 这样的字段函数，则输出列命名为 CNT_A。如果另一个选择项包含 SUM()，它的输出列命名为 SUM_B。

SELECT 子句中的用户自定义函数：

在 SELECT 子句中使用用户自定义函数有明显优点，但使用时应考虑以下限制：

- SELECT 子句的运行速度会受用户自定义函数执行速度的影响。因此，如果使用户自定义函数的操作量很大，则这些函数的功能最好调用 C 语言或汇编语言编写的 API 或用户自定义函数来完成。
- 在 SELECT 激活的用户自定义函数中，很难预测 Visual FoxPro 输入/输出（I/O）和表的环境。一般来说，不知道选择的工作区是哪一个，不知道当前表的名称，甚至不知道正在处理的字段名。这些变量的值完全取决于用户自定义函数在优化过程的什么地方激活。
- 在 SELECT 子句调用的用户自定义函数中修改 Visual FoxPro I/O 或表的环境是很不安全的。一般来说，这样做的结果难以预料。
- 从 SELECT 将值传递给用户自定函数唯一可靠的方法，是激活用户自定义函数时以参数的形式传递。

- 经过实践，有可能发现某种被认为是违法的操作在某种 FoxPro 版本中运行正确，但这并不保证它在以后的版本中也能正确运行。

抛开这些限制不说，用户自定义函数在 SELECT 语句中还是可接受的。但不要忘记使用 SELECT 可能要降低性能。

下列字段函数可以与选定项一起使用，选定项可以是一个字段或包含字段的表达式：

- AVG(*Select_Item*)，计算列中数值的平均值。
- COUNT(*Select_Item*)，计算列中选定项的数目。计算查询输出的行数。
- MIN(*Select_Item*)，确定列中 *Select_Item* 的最小值。
- MAX(*Select_Item*)，确定列中 *Select_Item* 的最大值。
- SUM(*Select_Item*)，计算列中数值的和。

字段函数不能嵌套使用。

连接 (Join)

Visual FoxPro 支持 ANSI SQL '92 连接 (Join) 语法，通过比较两个或多个表中的字段，将它们的记录连接到一起，生成查询。例如，内部连接 (inner join) 是将两个表中连接字段 (joined field) 值相同的记录选取到查询中。Visual FoxPro 支持嵌套连接 (nested joins)

重要事项

创建连接条件时，请记住如下几点：

- 如果在查询中包括两个表，又没有指定连接条件，那么第一个表中每一个记录同第二个表中每一记录之间，只要满足筛选条件，就连接起来。这种查询

产生的结果很长。

- 在连接条件中使用 DELETED（），EOF（），FOUND（），RECCOUNT（），和 RECNO（） 等支持可选别名或工作区的函数时要小心。在这些函数中包括别名或工作区可能导致不可预料的结果。SELECT 不使用工作区，它执行与 USE ... AGAIN 相同的操作。在不带可选别名和工作区的情况下使用这些函数进行单表查询时，可得到正确的结果。但是，使用这些函数进行多表查询时，即使不带可选别名或工作区，也可能得到一些莫明其妙的结果。
- 当连接含有空字段 (empty field) 的表时也要注意。这是因为 Visual FoxPro 匹配空字段。例如，连接 CUSTOMER.ZIP 和 INVOICE.ZIP 两个表。如果 CUSTOMER 表中的 100 条记录中没有给出邮政编码，INVOICE 表中也有 400 条记录没有给出邮政编码，那么查询结果中将会由于空字段的匹配而出现 40000 条多余的记录。要去除查询结果中的空记录，可以使用 EMPTY（） 函数。

有关连接的其它情况，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》第八章“创建视图”。

示例

以下示例说明了使用 SELECT – SQL 用户自定义函数的用法:

示例 1

示例 1 显示了 customer 表中所有的公司名称（来自一个表的一个字段）。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT customer.company ;
    FROM customer
```

示例 2

示例 2 显示了来自两个表中三个字段的内容及两个表基于 cust_id 字段的连接。两个表均使用了本地别名。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT a.company, b.order_date, b.shipped_on ;
    FROM customer a, orders b ;
    WHERE a.cust_id = b.cust_id
```

示例 3

示例 3 显示了在指定字段只有唯一数据的记录。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT DISTINCT a.company, b.order_date, b.shipped_on ;
    FROM customer a, orders b ;
    WHERE    a.cust_id = b.cust_id
```

示例 4

示例 4 以升序次序显示了 country、postalcode 和 company 字段。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT country, postalcode, company ;
   FROM customer ;
   ORDER BY country, postalcode, company
```

示例 5

示例 5 将来自两个表的字段内容存储在第三个表中。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT a.company, b.order_date, b.shipped_on ;
   FROM customer a, orders b ;
   WHERE a.cust_id = b.cust_id ;
   INTO TABLE custship.dbf
BROWSE
```

示例 6

示例 6 显示订单 (order) 日期早于 1994 年 2 月 16 日的记录。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT a.company, b.order_date, b.shipped_on ;
   FROM customer a, orders b ;
```

```
    WHERE a.cust_id = b.cust_id ;
    AND b.order_date < {^1994-02-16}
```

示例 7

示例 7 显示 customer 表中邮政编码与 orders 表的邮政编码相匹配的所有公司名。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT company FROM customer a WHERE ;
    EXISTS (SELECT * FROM orders b WHERE a.postalcode = b.postalcode)
```

示例 8

示例 8 显示 customer 表中公司名以大写的 C 开头，但长度未定的所有记录。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT * FROM customer a WHERE a.company LIKE "C%"
```

示例 9

示例 9 显示 customer 表中国家名以大写的 U 开头其后跟着未知字母的所有记录。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT * FROM customer a WHERE a.country LIKE "U_"
```

示例 10

示例 10 以大写方式，输出列为 CityList 显示所有城市名。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT UPPER(city) AS CityList FROM customer
```

示例 11

示例 11 演示了您可以执行在包含百分比符号 (%) 数据上查询的方式。在百分比之前放置的反斜线 (\) 表示应该将百分比符号当作字符处理，而 ESCAPE 子句中的反斜线指定为转义字符。

因为在 Visual FoxPro 的示例表中不包含百分比符号，所以查询没有返回结果。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')
SELECT * FROM customer;
WHERE company LIKE "%\%%" ESCAPE "\"
```

示例 12

示例 12 演示了您可以执行在包含下划线符号 (_) 数据上查询的方式。在下划线之前放置的反斜线 (\) 表示应该将下划线符号当作字符处理，而 ESCAPE 子句中的反斜线指定为转义字符。

因为在 Visual FoxPro 的示例表中不包含下划线符号，所以查询没有返回结果。

```
CLOSE ALL
```

```
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')
SELECT * FROM customer;
WHERE company LIKE "%\_%" ESCAPE "\"
```

示例 13

在示例 13 中，转义字符使用它自身的含义。短划线既是转义字符同时也是具有意义字符。查询返回公司名包含百分号后跟着短划线的所有行。

因为在 Visual FoxPro 的示例表中不包含百分比符号，所以查询没有返回结果。

```
CLOSE ALL
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')
SELECT * FROM customer;
WHERE company LIKE "%-%--%" Escape "-"
```

请参阅

CREATE QUERY, CREATE TABLE – SQL, INSERT – SQL, MODIFY QUERY, SETANSI , SET EXACT, SET PATH, _TALLY

SELECT（） 函数

返回当前工作区编号或未使用工作区的最大编号。

语法

SELECT([0 | 1 | cTableAlias])

返值类型

数值型

参数描述

0

指定 SELECT （） 返回当前工作区的编号。

1

指定 SELECT（） 返回未使用工作区的最大编号。

cTableAlias

指定表别名， SELECT（） 返回其所在工作区编号。

说明

如果 SET COMPATIBLE 设置为 OFF，则 SELECT() 返回当前工作区的编号。如果 SET COMPATIBALE 设置为 ON，SELECT（） 返回未用工作区编号最大的工作区。

使用 SELECT 可以选择（激活）工作区。

示例

```
CLOSE DATABASES
SET COMPATIBLE ON
OPEN DATABASE (HOME(2) + 'data\testdata')

SELECT 0   && 取消使用工作区
USE customer   && 打开 Customer 表

SELECT 0   && 取消使用工作区
USE orders   && 打开 Orders 表

CLEAR
? SELECT（）    && 返回 3，最小可用工作区
```

请参阅

ALIAS（）, SELECT, SET COMPATIBLE

Selected 属性

指定组合框或列表框控件中的一项是否被选中。设计时不可用，运行时可读写。

语法

[Form.]Control.Selected(nIndex)[= lExpr]

参数描述

nIndex

指定组合框或列表框中项的索引。

lExpr

Selected 属性的设置有：

设置	说明
“真” (.T.)	选中了该项。
“假” (.F.)	（默认值）没有选中该项。

说明

Selected 属性对于需要多重选择的用户特别有用。从列表中可以快速地确定究竟选择了哪些项，也可以在代码中使用 Selected 属性从列表中选择某项或删除原有的选择。

例如，要检查是否选择了列表框中的第三项，可执行下列命令：

```
IF MyList.Selected(3)
   WAIT WINDOW “It’s selected!”
ELSE
   WAIT WINDOW “It’s not!”
ENDIF
```

示例

以下示例创建了一个列表框。显示在列表框中的数据项的来源是一个数组，使用

RowSource 属性指定数组名。将 RowSourceType 属性设置为 5（数组）以指定列表框中数据项的来源为数组。

将 MultiSelect 属性设置为 (.T.)，允许您从列表框中进行多个选择。通过使用 ListCount、Selected 和 List 属性显示数据项或在列表框中选取的数据项，以确定在列表框中的数据项的数目和选取的数据项。

```
CLEAR

DIMENSION gaMyListArray(10)
FOR gnCount = 1 to 10   && 使用字母填充数组
    STORE REPLICATE(CHR(gnCount+64),6) TO gaMyListArray(gnCount)
NEXT

frmMyForm = CREATEobject('Form')   && 创建表单
frmMyForm.Closable = .f.   && 禁用窗口的弹出菜单
frmMyForm.Move(150,10)   && 移动表单

frmMyForm.Addobject('cmbCommand1','cmdMyCmdBtn')   && 添加“Quit”命令按钮
frmMyForm.Addobject('lstListBox1','lstMyListBox')   && 添加 ListBox 控件

frmMyForm.lstListBox1.RowSourceType = 5   && 指定数组
frmMyForm.lstListBox1.RowSource = 'gaMyListArray' && 包含列表框数据项的数组
frmMyForm.cmbCommand1.Visible =.T.   && 使“Quit”命令按钮可视
frmMyForm.lstListBox1.Visible =.T.   && 列表框可视
frmMyForm.SHOW   && 显示表单
READ EVENTS   && 开始事件处理

DEFINE CLASS cmdMyCmdBtn AS CommandButton   && 创建命令按钮
    Caption = '\<Quit'   && 命令按钮上的标题
    Cancel = .T.   && 默认“取消”命令按钮(Esc)
    Left = 125   && 命令按钮列
```

```
    Top = 210   && 命令按钮行
    Height = 25   && 命令按钮高
    PROCEDURE Click
        CLEAR EVENTS   && 停止事件处理，关闭表单
        CLEAR   && 清除 Visual FoxPro 主窗口
ENDDEFINE

DEFINE CLASS lstMyListBox AS ListBox   && 创建 ListBox 控件
    Left = 10   && 列表框列
    Top = 10   && 列表框行
    MultiSelect = .T.   && 允许选取一个以上数据项
PROCEDURE Click
    ACTIVATE SCREEN
    CLEAR
    ? "Selected items:"
    ? "--------------"
    FOR nCnt = 1 TO ThisForm.lstListBox1.ListCount
        IF ThisForm.lstListBox1.Selected(nCnt)   && 选取数据项吗？
            ? SPACE(5) + ThisForm.lstListBox1.List(nCnt) && 显示数据项
        ENDIF
    ENDFOR
ENDDEFINE
```

应用于

组合框，列表框

请参阅

AddItem **方法**, Clear **方法**, List **属性**, ListCount **属性**, ListItemID **属性**, MultiSelect **属性**, NewItemID **属性**, RemoveItem **方法**, RemoveListItem **方法**,

SelectedBackColor, SelectedForeColor 属性

指定选定文本的前景色和背景色。

语法

Control.SelectedBackColor[= nColor]

– 或 –

Control.SelectedBackColor = RGB(nRedValue, nGreenValue, nBlueValue)

Control.SelectedForeColor[= nColor]

– 或 –

control.SelectedForeColor = RGB(nRedValue, nGreenValue, nBlueValue)

参数描述

nColor

指定一个表示颜色的编号。

注意 在属性窗口中，双击任何颜色属性都可以显示“颜色”对话框，在这个对话框中可以选择或定义颜色。在关闭该对话框后，与所选颜色对应的红、绿和蓝的深度成为这些属性的设置。

有关详细内容，请参阅 BackColor, ForeColor 属性中的颜色表。

应用于

组合框，编辑框，微调，文本框

请参阅

BackColor, ForeColor 属性，RGB（），SelectedItemBackColor, SelectedItemForeColor 属性

SelectedID 属性

指定组合框或列表框控件中的一个项是否选中。设计时不可用，运行时可读写。

语法

control.SelectedID(nItemID)[= lExpr]

参数描述

nItemID

指定组合框或列表框中一个项的 ID 号。

lExpr

SelectedID 属性的设置有：

设置	说明
“真” (.T.)	选中了该项。
“假” (.T.)	（默认值）没有选中该项。

说明

SelectedID 属性在需要允许用户做多重选择时特别有用。从列表中可以快速地确定选择了列表中的哪一项。也可以在代码中使用 Selected 属性从列表中选择某个项，或删除原有的选择。

例如，要检查是否选择了列表框中的第三项，可执行如下命令：

```
IF MyList.SelectedID(3)
   WAIT WINDOW “It’s selected!”
ELSE
   WAIT WINDOW “It’s not!”
ENDIF
```

应用于

组合框，列表框

请参阅

AddItem 方法, Clear 方法, List 属性, ListCount 属性, ListItemID 属性, MultiSelect 属性, NewItemID 属性, RemoveItem 方法, RemoveListItem 方法, Selected 属性, TopItemID 属性

SelectedItemBackColor, SelectedItemForeColor 属性

指定组合框或列表框中选中项的前景色或背景色。

语法

Control.SelectedItemBackColor[= nColor]

– 或 –

Control.SelectedItemBackColor RGB(nRedValue, nGreenValue, nBlueValue)

Control.SelectedItemForeColor[= nColor]

– 或 –

Control.SelectedItemForeColor RGB(nRedValue, nGreenValue, nBlueValue)

参数描述

nColor

指定一个表示颜色的编号。

注意 在属性窗口中，双击任何颜色属性都可以显示“颜色”对话框，在这个对话框中可以选择或定义颜色。在关闭该对话框后，与所选择的颜色对应的红、绿和蓝的深度成为这些属性的设置。

有关详细内容，请参阅稍前的“BackColor，ForeColor 属性”。

应用于

组合框，列表框

请参阅

BackColor, ForeColor 属性, RGB( ), SelectedBackColor, SelectedForeColor 属性

SelectOnEntry 属性

指定当用户单击列单元时，是否选定这个单元中的内容。

语法

Object.SelectOnEntry[= lExpr]

参数描述

SelectOnEntry 属性的设置有：

设置	说明
“真”(.T.)	选定列单元。
“假”(.F.)	（默认值）不选定列单元。

说明

设计时和运行时可用。如果 HighLight 属性的设置为“真”(.T.)，与列的 SelectOnEntry 属性结合起来使用可以确定是否选定整个单元。如果 HighLight 属性的设置为“假”(.F.)，则忽略 SelectOnEntry 属性。

应用于

列，EditBox 控件，TextBox 控件

请参阅

Highlight 属性，HighlightRow 属性

SelLength 属性

返回用户在控件的文本区域中选定的字符数目，或指定要选定的字符数目。设计时不可用，运行时可读写。

语法

[Form.]Control.SelLength[= nLength]

参数描述

nLength

指定选定字符的数目，并突出显示选定的文本。设置值的有效范围是 0 到控件中字符的总数。把 SelLength 设置为 0 会引起运行错误。

说明

将这个属性同 SelStart 和 SelText 属性结合起来使用可完成以下任务：

- 在字符串中设置插入点。
- 建立一个插入范围，限制插入点的位置。
- 在控件中选择一组指定的字符（子字符串）。
- 清除文本。

当使用上述属性时，要注意下列操作：

- 把 SelLength 设置为小于 0 的值时会引起运行错误。
- 如果将 SelStart 设置为大于文本的长度，则该属性实际设置为已有文本的长度。如果改变 SelStart，则实际上是将原有选择改变为一个插入点，并把 SelLength 设置为 0。
- 把 SelText 设置为一个新值时，相当于重置 SelLength 为 0，并用新字符串代替原来选定的文本。

应用于

组合框，编辑框，微调，文本框

请参阅

SelStart 属性，SelText 属性

SelStart 属性

返回控件的文本输入区域中用户选择文本的起始点。当没有选定文本时，指示插入点的位置。另外，它还可以指定控件的文本输入区域中选择的文本起始点。 设计时不可用，在运行时读/写。

语法

[Form.]control.SelStart[= nStart]

参数描述

nStart

指出选定文本的起始点，或在没有选定文本时指定插入点的位置。选定的文本带底纹显示。设置值的有效范围是 0 到控件的编辑区中字符的总数。

说明

把本属性同 SelLength 和 SelText 属性结合起来使用，可完成以下任务：

- 在字符串中设置插入点。
- 建立一个插入范围，限制插入点的位置。
- 在控件中选择一组指定的字符（子字符串）。
- 清除文本。

当使用上述属性时，要注意下列操作：

- 把 SelLength 设置为小于 0 时会引起运行错误。
- 如果将 SelStart 设置为大于文本的长度的值，则实际上是把该属性设置为现有文本的长度。改变 SelStart 则将选择区改变为一个插入点，并把 SelLength 设置为 0 。
- 将 SelText 设置为新值时，SelLength 会设置为 0 ，并用新字符串代替选择的文本。

应用于

组合框，编辑框，微调，文本框

请参阅

SelLength 属性，SelText 属性

SelText 属性

返回用户控件的文本区中选定的文本，如果没有选定任何文本，则返回空字符串 ("") 。指定包含选定文本的字符串。设计时不可用，运行时可读写。

语法

[Form.]control.SelText[= cString]

参数描述

cString

指定包含选定文本的字符串。当没有选定文本时，指定空字符串。选定的文本带底纹显示。

说明

把本属性同 SelLength 和 SelText 属性结合起来使用，可完成以下任务：

- 在字符串中设置插入点。
- 建立一个插入范围，限制插入点的位置。
- 在控件中选择一组指定的字符（子字符串）。
- 清除文本。

当使用上述属性时，要注意下列操作：

- 把 SelLength 设置为小于 0 的值时会引起运行错误。
- 如果将 SelStart 设置为大于文本长度的值，则实际上是将该属性设置为现有文本的长度。改变 SelStart 则将选择区改变为一个插入点，并把 SelLength 设置为 0 。
- 将 SelText 设置为新值时，SelLength 会设置为 0 ，并用新字符串代替选择的文本。

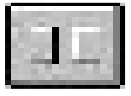
应用于

组合框，编辑框，微调，文本框

请参阅

SelLength 属性，SelStart 属性

Separator 对象



创建一个分隔符对象，用于工具栏中的控件之间加入空隙。

语法

Separator

说明

如果工具栏中的控件之间不包括 Separator 对象，控件将紧挨着放在工具栏中。Separator 对象是一个特殊的对象，它用来增加控件之间的空间并分隔控件组。

有关创建工具栏和分隔符的其他内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》第十一章“设计菜单和工具栏”。

属性

Application	BaseClass	Class
ClassLibrary	Comment	Enabled
Name	Parent	ParentClass
Tag		

事件

Destroy	Error	Init

方法

Addproperty	ReadExpression	ReadMethod
ResetToDefault	SaveAsClass	WriteExpression
ZOrder		

请参阅

Server 对象

对项目中一个服务程序的对象引用。

语法

Server

说明

在根据项目连编了一个可执行文件 (.exe) 或动态链接库 (.dll) 之后，会为项目中的每个服务程序创建和实例化一个服务程序对象。服务程序对象提供了对项目中一个服务程序的对象引用，并且允许您确定有关该服务程序的信息，通过该服务程序对象的属性管理它。

一个项目的服务程序集合包含了项目中的所有服务程序对象。

注意，一个服务程序对象是一个 COM 对象，所以将一个服务程序对象引用分配给一个变量，会创建一个 “Unknown Type” 的变量。

属性

CLSID	Description	HelpContextID
Instancing	ProgID	ServerClass

续表

ServerClassLib
rary

请参阅

File 对象,文件集合,Project 对象,项目集合,项目挂接对象,Servers Collection

ServerClass 属性

包含项目中一个服务程序类的名称。设计和运行时只读。

语法

object.ServerClass

说明

项目中服务程序类的名称也显示在“项目信息”对话框的“服务程序”选项卡中。

Automation 服务程序是在 Visual FoxPro 中创建的，需要将定义为 OLEPUBLIC 的类添加到项目中，然后根据该项目连编一个可执行文件 (.exe) 或动态链接库 (.dll)。

在项目中可以有任意多的 OLEPUBLIC 类，并且可以在程序文件 (.prg) 或类库 (.vcx) 中定义它们。

有关创建自定义 Automation 服务程序的详细内容，请参阅《*Microsoft Visual FoxPro 6.0* 中文版程序员指南》中的第十六章“添加 OLE”。

应用于

Server 对象

请参阅

CREATE CLASS, DEFINE CLASS, ServerClassLibrary 属性

ServerClassLibrary 属性

包含类库或程序的名称，该类库或程序中包含一个服务程序类。设计和运行时只读。

语法

object.ServerClassLibrary

说明

包含一个服务程序类的类库或程序的名称也显示在“项目信息”对话框的“服务程序”选项卡中。

Automation 服务程序是在 Visual FoxPro 中创建的，需要将定义为 OLEPUBLIC 的类添加到项目中，然后根据该项目连编一个可执行文件 (.exe) 或动态链接库 (.dll)。在项目中可以有任意多的 OLEPUBLIC 类，并且可以在程序文件 (.prg) 或类库 (.vcx) 中定义它们。

有关创建自定义 Automation 服务程序的详细内容，请参阅《*Microsoft Visual FoxPro 6.0* 中文版程序员指南》中的第十六章“添加 OLE”。

应用于

Server 对象

请参阅

CREATE CLASS, DEFINE CLASS, ServerClass 属性

ServerHelpFile 属性

为项目中服务程序类创建的类型库的帮助文件。

语法

object.ServerHelpFile[= cHelpFileName]

参数描述

cHelpFileName

指定类型库的帮助文件的名称。在默认情况下，cHelpFileName 包含空字符串。

说明

当在类或对象浏览器中查看一个类型库时，它的帮助文件通常用来提供有关服务程序类中的属性或方法的信息。Visual FoxPro 提供了“类浏览器”，用来查看一个类型库。类型库是当连编一个包含服务程序类的 .dll 或 .exe 文件时创建的。如果您使用 ServerHelpFile 属性为该类型库指定了帮助文件，就可以在类或对象浏览器中提供有关服务程序类中的属性或方法的帮助。

应用于

Project 对象

请参阅

HelpContextID 属性, Server 对象, Servers Collection, ServerProject 属性

ServerName 属性

包含一个 Automation 服务程序的完整路径和文件名。运行时只读。

语法

Application.object.ServerName

说明

ServerName 属性允许您确定一个内部处理 .dll 或外部处理 .exe Automation 服务程序的启动目录，以方便引用分布式应用程序中的其他文件。您也可以使用 ServerName 确定一个 Visual FoxPro .exe 应用程序的启动目录。

对于一个在交互工作期启动的 Visual FoxPro ，ServerName 属性包含与 FullName 属性相同的值。

应用于

Application 对象, _VFP

请参阅

FullName 属性, StartMode 属性

ServerProject 属性

包含服务程序类的项目的名称。

语法

object.ServerProject[= cName]

参数描述

cName

指定包含服务程序类的项目的名称。默认值是包含服务程序类的项目的名称，或者同样地是项目的 Name 属性值。

说明

您指定的 ServerProject 名称是 ProgID (Programmatic Identifier) 的第一部分，这个 ProgID 唯一地标识了服务程序。例如，如果您将 ServerProject 属性设置为 “MyApplication”，并且服务程序的一个 OLEPublic 类名是 “Server1”，您可以使用 “MyApplication.Server1” 这样的语法访问 Server1 类。

该属性对应于“项目信息”对话框的“服务程序”选项卡中的项目名。

应用于

Project 对象

请参阅

Name 属性, Server 对象, Servers Collection, ServerHelpFile 属性

Servers Collection

一个服务程序对象集合。

语法

Servers

说明

一个 Servers **集合提供了对服务程序对象的访问，允许您管理项目中的服务程序。有关服务程序集合和项目的详细内容，请参阅《***Microsoft Visual FoxPro 6.0* **中文版程序员指南》中的第三十二章“应用程序开发和开发者效率”的“项目管理器挂接”。有关创建自定义** Automation **服务程序的详细内容，请参阅《***Microsoft Visual FoxPro 6.0* **中文版程序员指南》中的第十六章“添加** OLE**”。**

属性

Count

方法

Item

请参阅

File 对象, 文件集合, Project 对象, 项目集合, ProjectHook 对象, Server 对象

SET 命令

打开查看窗口。

语法

SET

说明

该命令打开查看窗口，这是 Visual FoxPro 中最有效的工具之一。查看窗口可以方便地打开数据库、表，还可以轻松地建立表的关系，设置许多 Visual FoxPro 选项参数。

请参阅

SET VIEW

SET ALTERNATE 命令

将显示内容或打印输出定向到一个文本文件中。

语法

SET ALTERNATE ON | OFF

– 或 –

SET ALTERNATE TO [FileName [ADDITIVE]]

参数描述：

ON

将输出定向到文本文件。

OFF

（默认值）不将输出定向到文本文件。

TO FileName

创建文本文件。如果创建时没有明确指出文件的扩展名，则此文件具有 .TXT 扩展名。

如果不带文件名 *FileName* 使用 SET ALTERNATE TO 命令，则关闭使用 SET ALTERNATE TO 创建的最后一个文件。

ADDITIVE

在 *FileName* 指定的文件尾部追加输出内容。如果省略了 ADDITIVE，则覆盖指定文件原有内容。

请参阅

CLOSE, SET PRINTER

返回总目录

SET ANSI 命令
SET ASSERTS 命令
SET AUTOSAVE 命令
SET BELL 命令
SET BLOCKSIZE 命令
SET BORDER 命令
SET BROWSEIME 命令
SET BRSTATUS 命令
SET CARRY 命令
SET CENTURY 命令
SET CLASSLIB 命令
SET CLEAR 命令
SET CLOCK 命令
SET COLLATE 命令
SET COLOR OF 命令
SET COLOR OF SCHEME 命令
SET COLOR SET 命令
SET COLOR TO 命令

SET COMPATIBLE 命令
SET CONFIRM 命令
SET CONSOLE 命令
SET COVERAGE 命令
SET CPCOMPILE 命令
SET CPDIALOG 命令
SET CURRENCY 命令
SET CURSOR 命令
SET DATABASE 命令
SET DATASESSION 命令
SET DATE 命令
SET DEBUG 命令
SET DEBUGOUT 命令
SET DECIMALS 命令
SET DEFAULT 命令
SET DELETED 命令
SET DELIMITERS 命令
SET DEVELOPMENT 命令
SET DEVICE 命令
SET DISPLAY 命令
SET DOHISTORY 命令

SetData 方 法
SET ECHO 命 令
SET ESCAPE 命 令
SET EVENTLIST 命 令
SET EVENTTRACKING 命 令
SET EXACT 命 令
SET EXCLUSIVE 命 令
SET FDOW 命 令
SET FIELDS 命 令
SET FILTER 命 令
SET FIXED 命 令
SET FORMAT 命 令
SET FULLPATH 命 令
SET FUNCTION 命 令
SET FWEEK 命 令
SetFormat 方 法
SET HEADINGS 命 令
SET HELP 命 令
SET HELPFILTER 命 令
SET HOURS 命 令
SET INDEX 命 令

SET ANSI 命令

确定 SQL 字符串比较方式。

语法

SET ANSI ON | OFF

参数描述

ON

在较短的字符串后面加入空格以使它与较长的字符串具有相同的长度。然后，对两个字符串的每个字符进行一个字符一个字符的比较。

考虑以下比较：

'Tommy' = 'Tom'

如果将 SET ANSI 设置为 ON，因为填充空格的缘故，结果为 (.F.)。“Tom”变成了“Tom ”，字符串“Tom ”与“Tommy”不是逐字匹配的。

Visual FoxPro SQL 命令中，操作符 == 使用的就是这种比较方法。

OFF

指定较短的字符串不要填充空格。两个字符串比较到较短的字符串结束就可以

了。

考虑以下比较：

'Tommy' = 'Tom'

当 SET ANSI 设置为 OFF 时，因为比较在“Tom.”之后就停止了，比较结果为 (.T.)。

说明

SET ANSI 命令决定进行 SQL 字符串比较时是否在两个字符串中较短的一个后面加上空格。SET ANSI 对操作符 == 没有任何影响。当使用操作符 == 时，较短的字符串总是在其后加上空格后再进行比较。

SET ANSI 的作用域是当前数据工作期。

SET ANSI 命令与查询设计器

当创建查询时，Visual FoxPro 在查询设计器中建立一个 SELECT - SQL 命令。当创建连接和筛选条件时，如果选择了 Equal 或 Exactly Like 选项的话，操作符 = 或 == 包括在产生的 SELECT 命令中。SET ANSI 设置会影响在查询设计器中创建和执行的查询结果。

字符串顺序

在 SQL 命令中，比较时，两个字符串的左右顺序没有影响。把字符串从操作符 =（或 ==）的一边移到另一边不影响比较结果。

请参阅

CREATE QUERY, MODIFY QUERY, SELECT – SQL, SET DATASESSION,

SET ASSERTS 命令

指定 ASSERT **命令是否被忽略。**

语法

SET ASSERTS ON | OFF

参数描述

ON

Visaul FoxPro **遇到** ASSERT **命令时执行它。**

OFF

Visaul FoxPro **忽略** ASSERT **命令。**

说明

在 ASSERTS **命令的消息框中选择“全部忽略”按钮将自动** SET ASSERTS **为** OFF**。**

请参阅

ASSERT

SET AUTOSAVE 命令

当退出 READ 命令或返回到命令窗口时，决定是否把数据缓冲区中的数据保存到磁盘上。

语法

SET AUTOSAVE ON | OFF

参数描述

ON

无论什么时侯退出 READ 命令或者返回到命令窗口，指定缓冲区的数据都要保存到磁盘上。

OFF

只有当退出 READ 命令或返回到命令窗口，并且自上次保存之后五分钟时，才将缓冲区的数据保存到磁盘上。OFF 是 SET AUTOSAVE 的默认设置。

说明

将缓冲区的内容保存到磁盘上可以减少由于计算机突然断电丢失数据的可能性。SET AUTOSAVE 的作用域是当前数据工作期。

请参阅

SET BELL 命令

关掉或打开计算机铃声，也可以设置铃声属性。

语法

SET BELL ON | OFF

– 或 –

SET BELL TO [cWAVFileName]

参数描述

ON

（默认设置）打开铃声。

OFF

关掉铃声。

TO cWAVFileName

在铃响的时候指定铃声的波形。cWAVFileName 还包括到波形的路径。

不设置 cWAVFileName 而使用 SET BELL TO 命令，恢复默认的声音波形。

说明

在编辑期间，当到达一个字段末尾或输入无效数据时，SET BELL 可开始或停止响铃。

示例

在下面的示例中，播放了声波文件 TIME.WAV。

```
SET BELL TO 'C:\WINDOWS\TIME.WAV', 0
?? CHR(7)
```

SET BLOCKSIZE 命令

指定 Visual FoxPro 为备注字段如何分配磁盘空间。

语法

SET BLOCKSIZE TO nBytes

参数描述

nBytes

指定给备注字段分配磁盘空间的块的大小。如果 *nBytes* 等于 0，磁盘空间按单个字节（一块 1 字节）分配；如果 nBytes 为 1 到 32 之间的一个整数，分配磁盘空间的块大小为 *nBytes* 乘以 512；如果 *nBytes* 大于 32，分配的磁盘空间块大小为 *nBytes* 字节。确定 BLOCKSIZE 值大于 32，可以显著地节省磁盘空间。

说明

SET BLOCKSIZE 的默认值是 64。当文件创建后需重置 BLOCKSIZE 的值时，首先将 BLOCKSIZE 设置为一个新值，然后使用 COPY 命令创建一个新表，此表便具有新设定的长度。SET BLOCKSIZE 的作用域是当前数据工作期。

请参阅

COPY FILE, MODIFY STRUCTURE, PACK, SET DATASESSION

SET BORDER 命令

包含此命令是为了提供向后兼容性。请使用 BorderStyle 属性。

SET BROWSEIME 命令

指定当在“浏览”窗口中定位到一个文本框时，是否显示输入法生成器 (Input Method Editor)。

语法

SET BROWSEIME ON | OFF

参数描述

ON
（台湾版和中国大陆版 Windows 的默认值）

OFF
（韩国版 Windows 的默认值）当在“浏览”窗口中定位到一个文本框时，不显示输入法生成器。

说明

SET BROWSEIME 对应于“选项”对话框“常规”选项卡的“浏览 IME 控制”复选框。

该命令只用于 DBCS 环境中。.

请参阅

IMEMode 属性, IMESTATUS()

SET BRSTATUS 命令

包含此命令是为了提供向后兼容性。请使用 StatusBarText 属性。

SET CARRY 命令

当使用 INSERT、APPEND 和 BROWSE 命令创建新记录时，决定是否将当前记录数据复制到新记录中。

语法

SET CARRY ON | OFF

–或者–

SET CARRY TO [FieldList [ADDITIVE]]

参数描述

ON

从当前记录中，将所有工作区的所有字段的内容复制到新记录中。

OFF
（默认设置）防止将任何字段的数据复制到新记录中。
TO [FieldList [ADDITIVE]]
复制由 *FieldList* 指定的字段的数据，字段名之间用逗号隔开。
ADDITIVE 指定将字段列表中的字段添加到新记录的字段中。
执行 SET CARRY TO 命令隐含执行了 SET CARRY ON 命令。
不带参数 *FieldList* 执行 SET CARRY TO 可恢复默认设置（复制所有的字段）。

说明

使用 SET CARRY 命令可以确定是否把当前记录中的数据复制到新记录中。在编辑期间，那些总保持不变的字段可以将其复制到新记录中。例如，包含当前日期的字段可以复制到每个新记录中，这样就不必为每个新记录再输入日期。请注意备注字段和通用类型字段的数据不能复制到新记录中。
SET CARRY 只影响当前选择的工作区中打开的表。
SET CARRY 的作用域是当前数据工作期。

请参阅

INSERT, SET DATASESSION

SET CENTURY 命令

确定 Microsoft Visual FoxPro 是否显示日期表达式当前世纪部分并确定 Visual FoxPro 只有两位年份值的日期的解释方式。

语法

SET CENTURY ON | OFF | TO [nCentury [ROLLOVER nYear]]

参数描述

ON

指定在包含 10 个字符（包括日期分隔符）的日期格式中，用四位数字表示年。

注意 为与 2000 年兼容，建议您通常将 SET CENTURY 设置为 ON。有关 2000 年兼容性的详细内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》第三十三章“对编程的改进”中的“对 2000 年日期的支持”。

OFF

（默认值）指定在包括 8 个字符的日期格式中，用两位数字表示年。在进行日期计算时假定它表示的是 20 世纪。

TO nCentury

是 1 到 99 的数值，指定当前的世纪。当一个日期值有两个表示年的数字

时，nCentury 确定了该年的世纪。ROLLOVER 值确定了该年是否在 nCentury 世纪，还是在 nCentury 之后的世纪。

ROLLOVER nYear

是 0 到 99 的数值，指定年数，大于等于该年数的年份被认为是当前世纪，小于该年数的年份被认为是下个世纪。nYear 的默认值是当前年加 50 年后的最后两个数字，例如，如果当前年是 1998，nYear 是 48。

注意，翻转值只决定一个在输入时不带世纪部分的日期（不建议这样做）的世纪。

例如，如果当前年是 1998，并且 nYear 是默认值 (48)，在输入时不带世纪部分的日期，并且大于等于 48 的年份被认为是在本世纪（20 世纪）。在输入时不带世纪部分的日期，并且小于 48 的年份被认为是在下一个世纪（21 世纪）。

说明

使用 SET CENTURY 可以指定日期变量和函数的显示格式。

不带任何参数地发出 SET CENTURY TO 命令，会将默认世纪恢复为当前世纪，并且将 ROLLOVER 恢复为当前年加 50 年后的最后两个数字。在 Visual FoxPro 5.0 中，不带任何参数地发出 SET CENTURY TO 命令，会将默认世纪恢复为 19 世纪，并且将 ROLLOVER 恢复为零。

SET CENTURY 的作用域是当前数据工作期。新的数据工作期初始化为当前世纪，忽略 SET CENTURY 为当前数据工作期的设置值。

请参阅

DATE(), SET DATASESSION, SET STRICTDATE, SET SYSFORMATS, YEAR()

SET CLASSLIB 命令

打开包含类定义的 .VCX 可视类库。

语法

SET CLASSLIB TO ClassLibraryName [IN APPFileName | EXEFileName] [ADDITIVE] [ALIAS AliasName]

参数描述

TO ClassLibraryName

指定要打开的 .VCX 可视类库的名称。如果 *ClassLibraryName* 没有包含完整的路径名，Visual FoxPro 首先在默认的 Visual FoxPro 目录中查找可视类库，然后在 Visual FoxPro 路径下各目录中查找。

如果不带 *ClassLibraryName* 执行 SET CLASSLIB TO 命令，则 Visual FoxPro 将关闭所有的可视类库。用 RELEASE CLASSLIB 命令也可以关闭一个可视类库。

IN APPFileName | EXEFileName
指定类库所在的应用程序文件 (.APP) 或可执行文件 (.EXE)。

ADDITIVE
打开 .VCX 可视类库时，不关闭任何当前打开的 .VCX 可视类库。如果省略这个子句，将关闭所有 .VCX 可视类库。

ALIAS AliasName
指定可视类库的别名，可以通过它的别名来引用可视类库。例如，以下命令打开了一个名为 MyClass 的 .vcx 可视类库。将它赋予别名 MyCntrls，然后创建一个名为 MyButton 的控件。

```
SET CLASSLIB TO MyClass ALIAS MyCntrls
mMyButton = CREATEOBJ('MyCntrls.MyButton')
```

说明

当执行 CREATE 对象()、DEFINE CLASS 中的 ADD 对象 或 Add 对象 Method 时，Visual FoxPro 在下列位置按下列顺序查找类定义，这个类定义定义了这些命令中指定的对象。

1. Visual FoxPro 基类。
2. 按装入顺序查找内存中的类定义。
3. 当前程序中的类定义。
4. 由 SET CLASSLIB 打开的 .VCX 类库中的类定义。

5. 由 SET PROCEDURE 打开的过程文件中的类定义。
6. 在 Visual FoxPro 程序执行链中的类定义。
7. 注册屏幕。

如果没有找到包含对象的类定义，Visual FoxPro 产生一条错误信息。

示例

以下示例使用了 CREATE CLASSLIB 命令创建了一个名为 myclslib 的可视类库。基于 Visual FoxPro 表单 (Form) 基类创建了一个名为 myform 的类，并将其保存在 myclslib 可视类库中。使用 SET CLASSLIB 命令打开 myclslib 可视类库，以便使用其中的类。

```
CREATE CLASSLIB myclslib    && 新建 .VCX 可视类库
CREATE CLASS myform OF myclslib AS "Form"  && 新建类
SET CLASSLIB TO myclslib ADDITIVE    && 打开 MyClsLib.VCX
```

请参阅

ADD CLASS, Addobject 方法, CREATE CLASS, CREATE CLASSLIB, CREATEobject(), MODIFY CLASS, RELEASE CLASSLIB, SET PROCEDURE, SET OLEobject

SET CLEAR 命令

包含此命令是为了向后兼容性。请使用 Refresh 方法。

SET CLOCK 命令

语法

SET CLOCK ON | OFF | STATUS

– 或 –

SET CLOCK TO [nRow, nColumn]

参数描述

ON

在 Visual FoxPro 主窗口的右上角显示时钟。

OFF

（默认值）从 Visual FoxPro 主窗口中或状态栏中移去时钟。

STATUS
在图形状态栏中显示时钟。执行 SET STATUS BAR ON 命令可显示图形状态栏。

TO [nRow, nColumn]
使用行和列的坐标指定时钟在 Visual FoxPro 主窗口中的显示位置。不带坐标参数使用 SET CLOCK TO 时，时钟显示在 Visual FoxPro 主窗口中右上角的默认位置上。

在 Visual FoxPro 中，如果使用 SET CLOCK STATUS 命令将时钟设置在图形状态栏上，又使用 TO *nRow, nColumn* 在 Visual FoxPro 主窗口中给时钟指定一个位置，Visual FoxPro 将时钟放在指定的位置上，而不放在图形状态栏中。

请参阅

SET BRSTATUS, SET HOURS, SET STATUS

SET COLLATE 命令

指定在后续索引和排序操作中，字符型字段的排序顺序。

语法

SET COLLATE TO cSequenceName

参数描述

cSequenceName

指定排序顺序。

有效的排序顺序有：

选项	语言
DUTCH	荷兰语
GENERAL	英语、法语、德语、当代西班牙语、葡萄牙语和其他西欧语言
GERMAN	德国电话序列（DIN）
ICELAND	冰岛语
MACHINE	机器语言（早期 FoxPro 版本中排序顺序的默认设置）
NORDAN	挪威语、丹麦语
SPANISH	古典西班牙语
SWEFIN	瑞典语、芬兰语
UNIQWT	唯一重量

注意 当指定 SPANISH 选项时，“ch”是排在“c”和“d”之间的单个字母；“ll”排在“l”和“m”之间。

如果指定的排序序列是原义字符串，应该用引号把选项括起来：

```
SET COLLATE TO "SWEFIN"
```

MACHINE 是默认的排序顺序选项，也是 Xbase 用户很熟悉的排序方法。字符是按它们

在当前代码页中出现的顺序排序的。
美国和西欧用户可能更喜欢使用 GENERAL。在这个选项中，字符的排序顺序与它们在当前代码页中出现的顺序相同。在 FoxPro 的 2.5 以前的版本中，可以在创建索引时使用 UPPER( ) 或 LOWER( ) 来操作字符字段。在 FoxPro 的 2.5 以后的版本中，可以指定 GENERAL 排序顺序选项而省略 UPPER( ) 转换。

注意如果指定的排序顺序选项不是 MACHINE，并创建了 .IDX 文件，则创建的 .IDX 文件是压缩的 .IDX 文件。

使用 SET("COLLATE") 可以返回当前的排序顺序。

如果在 Visual FoxPro 的配置文件中加入如下一行，那么启动 Visual FoxPro 时将自动指定一种排序顺序。

```
COLLATE = cSequenceName
```

也可以执行如下命令，结果相同。

```
SET COLLATE TO cSequenceName
```

说明

SET COLLATE 可以对任何 FoxPro 支持语言中包含重音字符的表进行排序。改变 SET COLLATE 设置不会影响以前打开的索引排序顺序。Visual FoxPro 自动维护已经存在的索引，保持提供创建多种不同类型索引的灵活性，甚至对于同一字段也是如此。例如，如果在 SET COLLATE 设置为 GENERAL 时创建了一个索引，然后把 SET COLLATE 的设置改变为 SPANISH，这时，原来创建的索引仍然保持 GENERAL 排序

顺序。

SET COLLATE 的作用域是当前数据工作期。

有关代码页及 Visual FoxPro 的国际化支持的其他内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》第十八章“开发国际化应用程序”中的“Visual FoxPro 支持的代码页”。

有关配置 Visual FoxPro 的详细内容，请参阅“帮助”中“安装指南”第三章“配置 Visual FoxPro ”。

请参阅

IDXCOLLATE(), SET DATASESSION

SET COLOR OF 命令

包含此命令是为了提供向后兼容性。可用 SET COLOR OF SCHEME 代替。

SET COLOR OF SCHEME 命令

指定配色方案中的颜色，也可以将一个配色方案中的颜色复制到另一个配色方案。

语法

SET COLOR OF SCHEME nScheme1 TO
[SCHEME nScheme2 | ColorPairList]

参数描述

nScheme1

指定要更改的配色方案的编号。这个值可以是一个 1 到 24 之间的数。

注意 在 Visual FoxPro 中，配色方案 13 到 15 留作内部使用；在 FoxPro for Windows 中，配色方案 13 和配色方案 14 留作内部使用；在 FoxPro for Macintosh 中，配色方案 13 到 16 留作内部使用。请不要使用这些配色方案。

TO [SCHEME nScheme2]

指定改变 *nScheme1* 后的配色方案的编号。

TO [ColorPairList]

指定配色方案中想改变的颜色对数目，最多为 10 对。如果在某一个颜色对相应的位置上放置一个逗号，则可以指定不更改这个颜色对，这样便可以有

选择地改变配色方案中的颜色对。例如，如果要在方案 1 中配置第三个颜色对以加入白和蓝，其他颜色设置不变，可是使用后面的命令：

```
SET COLOR OF SCHEME | TO， ， W+/B*
```

颜色对也可以用 6 个用逗号隔开的 RGB（红、绿和蓝）颜色值来表示。如果要在方案 1 中配置第三个颜色对，以加入白和蓝，其他颜色设置不变，可以使用以下命令：

```
SET COLOR OF SCHEME 1 TO , , RGB(255,255,255,0,0,255)
```

说明

在 Visual FoxPro 中，不是所有的界面元素都能由配色方案控制。例如，查看窗口和命令窗口等系统窗口、系统菜单栏等等，它们的颜色由控制面板的颜色设置控制。执行不包括可选子句的 SET COLOR OF SCHEME nScheme1 TO 命令，可从当前配色方案中恢复颜色。

请参阅

颜色概述, CREATE COLOR SET, SET COLOR SET

SET COLOR SET 命令

装入预先定义好的颜色集合。

语法

SET COLOR SET TO [ColorSetName]

参数描述

ColorSetName

指定要装入的颜色集合。

说明

可以使用 SET COLOR OF SCHEME 命令创建颜色集合，并用 CREATE COLOR SET 命令将颜色集合保存起来。

在 Visual FoxPro 中，如果发出不带可选子句的 SET COLOR SET 命令，则仍然使用当前颜色集合。

有关颜色设置的详细内容，请参阅稍前部分的语言参考“颜色概述”。

请参阅

颜色概述, CREATE COLOR SET, SET COLOR OF SCHEME

SET COLOR TO 命令

包含此命令是为了提供向后兼容性。可用 SET COLOR OF SCHEME 命令代替。

SET COMPATIBLE 命令

控制与 FoxBASE+ 以及其他 Xbase 语言的兼容性。

语法

SET COMPATIBLE FOXPLUS | OFF | DB4 | ON
[PROMPT | NOPROMPT]

参数描述

FOXPLUS | OFF

（默认值）这两个关键字可以相互替换使用，它们都允许在 Visual FoxPro 环境下运行 FoxBASE+ 创建的程序而无需作任何改动。

DB4 | ON

包括这两个关键字中的任何一个都会影响一些命令和函数的操作。

PROMPT | NOPROMPT

在打开包含备注字段的 dBASE 表时，这两个选项决定 Visual FoxPro 是否显示一个对话框。

包含 PROMPT 选项可显示“转换备注字段”对话框。如果打开一个包含备注字段的 dBASE 表，默认情况下，Visual FoxPro 显示一个“转换备注字段”对话框。这一对话框可以用来将 dBASE 备注文件转换为 Visual FoxPro

格式。要在 Visual FoxPro 中打开此备注文件，必须先将它转换为 Visual FoxPro 格式。在此之后，可以在 COPY 命令中包含 TYPE FOXPLUS 选项，将备注文件转换为 dBASE 格式。

如果包含 NOPROMPT 选项，打开一个包含备注字段的 dBASE 表时，将不显示“转换备注字段”对话框。在这种情况下，dBASE 备注文件自动转换成 Visual FoxPro 格式。

说明

受 SET COMPATIBLE 影响的命令和函数包括：LIKE()，PLAY MACRO，SELECT() 以及 STORE（当 STORE 用来存储数组对时）。

SET COMPATIBLE 不支持那些 FoxPro 本身不支持的其他数据库语言的命令、函数或特性。例如，不能使用报表设计器来打开由其他数据库语言编写的报表文件。

下表列出了受 SET COMPTIBLE 影响的命令。

命令

带 RANGE 子句的 @ ... GET

@ ... SAY 滚动命令

ACTIVATE SCREEN

APPEND MEMO

DIMENSION

FSIZE( )

带 @ ... GET VALID 子句的 READ

嵌套的 READ

ACTIVATE WINDOW

DECLARE

GO | GOTO with SET

TALK ON

INKEY( )

续表

LASTKEY( )

菜单命令

带 @ ... GET VALID 子句的 READ 命令

READ

SET COLOR TO

SET FIELDS

SET MEMOWIDTH

STORE

带数值 PICTURE 子句的

TRANSFORM( )

SYS(2001, “COLOR”)

带数值 PICTURE 子句的

TRANSFORM( )

PLAY MACRO

嵌套的 READ

RUN | !

SET BORDER

SET MESSAGE

SET PRINTER TO <file>

SUM

SELECT( )

请参阅

SET KEYCOMP

SET CONFIRM 命令

设置是否可以通过键入最后一个字符来退出文本框。

语法

SET CONFIRM ON | OFF

参数描述

ON

指定在文本框中不能用键入最后一个字符的方法退出文本框。要退出文本框，可以按 ENTER，TAB 键或其他任何箭头键，从一个文本框移到另一个控件。

SET CONFIRM ON 也影响用 DEFINE BAR 和 DEFINE PAD 创建的菜单项和菜单标题。当键入菜单项或菜单标题的第一个字母时，该菜单项和菜单标题被选择，但是并不执行。要执行已选择了的菜单项或标题，应键入 ENTER 键或 SPACEBAR 键

OFF

（默认值）指定插入点在到达文本框中的最后一个字符时，移到下一个控件上，并且响铃（如果 SET BELL 设置为 ON）。

SET CONFIRM OFF 也影响菜单项和菜单标题。如果 SET CONFIRM 设置为 OFF，可以

通过键入与菜单项或菜单标题的第一个字母对应的键来从菜单栏中的菜单或菜单标题中执行一项（当 SET CONFIRM 设置为 ON 时，这种操作只能选择菜单项或标题）。

说明

SET CONFIRM 对菜单项和菜单标题的访问键没有任何影响。如果菜单项和菜单标题创建时带有访问键，就可以通过按对应的访问键来执行菜单项或菜单标题。

SET CONFIRM 在当前数据工作期有效。

请参阅

文本框控件，SET DATASESSION

SET CONSOLE 命令

设置是否将输出送到 Visual FoxPro 主窗口或活动的用户自定义窗口。

语法

SET CONSOLE ON | OFF

参数描述

ON
（默认值）将所有的输出送到 Visual FoxPro 主窗口或活动的用户自定义窗口中。

OFF
不将输出送到 Visual FoxPro 主窗口或活动的用户自定义窗口中。

说明

当交互式使用 Visual FoxPro 时，SET CONSOLE 设置为 ON，而且不能从命令窗口将设置值更改为 OFF。只能在程序中把设置改为 OFF。

SET CONSOLE 影响一些 Visual FoxPro 交互式对话框。例如，如果在 SET CONSOLE 设置为 OFF 时执行 BROWSE 命令，而当前又没有打开的表或数据库，则 Visual FoxPro 显示一条错误信息。在同样条件下，当 SET CONSOLE 设置为 ON 时，Visual FoxPro 会弹出一个“打开”对话框。

SET CONSOLE 不影响 @ ... SAY 的输出。@ ... SAY 的输出由 SET DEVICE 的设置控制。

重要提示 当 SET CONSOLE 设置为 ON 时，会出现一些错误。请使用 SYS(100) 检查 SET CONSOLE 的设置。有关这方面的详细内容，请参阅 SYS(100)

请参阅

SYS() 函数概述

SET COVERAGE 命令

开启或关闭编辑日志，或指定一个文本文件，编辑日志的所有信息将输出到其中。

语法

SET COVERAGE TO [FileName [ADDITIVE]]

参数描述

TO FileName

指定编辑日志信息输出到的文本文件。若执行 SET COVERAGE TO 不包含文件名，则关闭已打开的文件。

如果您指定的文件不存在。Visual FoxPro 将自动创建并打开它。

ADDITIVE

编辑日志信息追加到指定文件的尾部。如果您省略 ADDITIVE 关键字，当前的内容将覆盖原有的内容。

说明

您可以在“编辑日志”对话框中设置上面提到的所有选项。

有关代码范围分析器的详细内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》第三十二章“应用程序开发和开发者的生产率”中的“代码范围分析器应用程序”。

请参阅

_COVERAGE, SET eventTRACKING

SET CPCOMPILE 命令

指定编译程序的代码页。

语法

SET CPCOMPILE TO [nCodePage]

参数描述

nCodePage

使用 *nCodePage* **指定编译代码页。**

有关代码页及 Visual FoxPro **的国际化支持的其他内容，请参阅《**Microsoft Visual FoxPro 6.0 **中文版程序员指南》第十八章“开发国际化应用程序”中的“**Visual FoxPro **支持的代码页”。**

要重置编译代码页为当前代码页，请使用不带 *nCodePage* **的** SET CPCOMPILE TO **命令。使用** CPCURRENT( ) **可以确定当前代码页。**

说明

要使用指定代码页编译程序，请使用 SET CPCOMPILE 命令。用 SET CPCOMPLIE 指定的代码页用于 Visual FoxPro 自动编译的程序，也可用于从“编译”对话框中编译的程序或使用 COMPILE 命令编译的程序。不过，可以在 COMPILE 命令中使用 AS 子句来忽略由 SET CPCOMPILE 指定的代码页。

请参阅

COMPILE, CPCURRENT()

SET CPDIALOG 命令

指定打开表时是否显示“代码页”对话框。

语法

SET CPDIALOG ON | OFF

参数描述

ON

当打开一个表时，若下列条件为真，则显示“代码页”对话框：

表以独占方式打开。

表没有标明代码页。

OFF
当表打开时，不显示“代码页”对话框。

说明

“代码页”对话框允许为 FoxPro 的早期版本所创建的表指定代码页，也可以为创建 Visual FoxPro 表的其他产品指定代码页。如果选定了代码页，则用选定的代码页标记表。

当建立应用程序时，使用 SET CPDIALOG ON 可以确保包含在应用程序中的表使用正确的代码页标记。在最终的应用程序中，要确保 SET CPDIALOG 设置为 OFF。也可以在“选项”对话框的“数据”选项卡中，应用“提示代码页”复选框交互地指定是否显示“代码页”对话框。要显示“选项”对话框，可从“工具”菜单中选择执行“选项”命令。

注意 有关代码页及 Visual FoxPro 的国际化支持的其他内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》第十八章“开发国际化应用程序”中的“Visual FoxPro 支持的代码页”。

请参阅

CPCURRENT( ), CPCONVERT( ) , CPDBF( ) , GETCP( ) , USE

SET CURRENCY 命令

定义货币符号，并指定其显示位置。

语法

SET CURRENCY TO [cCurrencySymbol]

– 或 –

SET CURRENCY LEFT | RIGHT

参数描述

cCurrencySymbol

指定作为货币符号的字符串，最长可以是 9 个字符。使用不带 *cCurrencySymbol* 的 SET CURRENCY TO 命令可以将货币符号重置为默认的美元符（$）。

LEFT

（默认值）将货币符号放在货币值的左边。

RIGHT

将货币符号放在货币值的右边。

说明

在 FUNCTION 或 PICTURE 子句中包括 $ 代码时，在 @ ... SAY 生成的输出中以及在

@ ... GET 生成的文本框中都要显示货币符号。

SET CURRENCY 在当前数据工作期有效。

示例

下面的示例中，在货币值的两边显示了货币符 DM。如果使用 PICTURE 来显示货币，要确保在美元符前面加上 @。

```
STORE SET('CURRENCY') TO gcCurrPosit
STORE 1234.56 TO gnDollarAmnt
CLEAR
SET CURRENCY TO 'DM'
@ 2,2 SAY gnDollarAmnt PICTURE '@$99,999.99'
IF gcCurrPosit = 'LEFT'
    SET CURRENCY RIGHT
ELSE
    SET CURRENCY LEFT
ENDIF
@ 4,2 SAY gnDollarAmnt FUNCTION '$99,999.99'
```

请参阅

SET DATASESSION, SET DECIMALS, SET SEPARATOR, SET SYSFORMATS

SET CURSOR 命令

在 Visual FoxPro 等待输入时，确定是否显示插入点。

语法

SET CURSOR ON | OFF

参数描述

ON

（默认值）在执行 @ ... GET，@ ... EDIT，WAIT 或 INKEY( ) 期间显示插入点。

OFF

在执行 @ ... GET, @ ... EDIT，WAIT 或 INKEY( ) 期间不显示插入点。

说明

SET CURSOR 同 SYS(2002) 相似，能够控制是否显示插入点。

请参阅

INKEY( ), SET( ), SYS(2002), WAIT

SET DATABASE 命令

指定当前数据库。

语法

SET DATABASE TO [DatabaseName]

参数描述

DatabaseName

指定一个打开的数据库的名称，使它成为当前数据库。如果省略 *DatabaseName*，则打开的数据库都不会成为当前数据库。

说明

同时可以打开多个数据库，但是只有一个可能成为当前数据库。用以操作打开的数据库的命令和函数，例如 ADD 和 DBC()，只对当前数据库操作。

可以从“常用”工具栏上的数据库下拉列表中，选择一个打开的数据库作为当前数据库。

请注意当执行查询或表单时，Visual FoxPro 可以自动打开数据库。

SET DATABASE 在当前数据工作期有效。

示例

下面的示例创建了两个名为 mydbc1 和 mydbc2 的数据库和一个名为 table1 的表。用 SET DATABASE 将 mydbc1 变成当前数据库，并且在创建表时把表加到数据库 mydbc1 中。然后，关掉表并把它从数据库 mydbc1 中移去。用 SET DATABASE 将 mydbc2 变成当前数据库，然后用 ADD TABLE 将表添加到 mydbc2 中。最后用 RENAME TABLE 命令将表名由 table1 改为 table2。

```
CREATE DATABASE mydbc1
CREATE DATABASE mydbc2
SET DATABASE TO mydbc1
CREATE TABLE table1 (cField1 C(10), n N(10)) && 将表添加到 mydbc1
CLOSE TABLES    && 一个表要从数据库中移去时，必须先关闭
REMOVE TABLE table1
SET DATABASE TO mydbc2
ADD TABLE table1
RENAME TABLE table1 TO table2
```

请参阅

ADD TABLE, CLOSE DATABASES, DBC( ), DBGETPROP( ) , DBSETPROP( ) , DELETE DATABASE, DISPLAY TABLES, MODIFY DATABASE, OPEN DATABASE, REMOVE TABLE, SET DATASESSION

SET DATASESSION 命令

激活指定的表单数据工作期。

语法

SET DATASESSION TO [nDataSessionNumber]

参数描述

nDataSessionNumber

指定要激活的表单数据工作期。如果省略 *nDataSessionNumber*，将激活数据工作期 1（全局数据工作期）。

说明

默认情况下，当启动 Visual FoxPro 时，数据工作期 1（全局数据工作期）是活动的。

创建表单时，表单对象的 DataSession 属性决定表单是否有它自己唯一的数据工作期。如果表单的 DataSession 属性置为“真”（.T.），则表单有它自己的数据工作期；否则，不创建表单的数据工作期。可以使用表单的 DataSessionId 只读属性来确定表单数据工作期编号。

SET DATASEESION 一般用于调试表单对象，并且不应该在表单活动时使用。

下列 SET 命令的作用域为当前数据工作期：

SET 命令

SET ANSI
SET AUTOSAVE
SET BLOCKSIZE
SET CARRY
SET CENTURY
SET COLLATE
SET CONFIRM
SET CURRENCY
SET DATABASE
SET DATE
SET DECIMALS
SET DELETED
SET DELIMITERS
SET EXACT
SET EXCLUSIVE
SET FIELDS
SET FIXED
SET HOURS
SET LOCK
SET MARK TO
SET MEMOWIDTH
SET MULTILOCKS
SET NEAR
SET NULL
SET POINT
SET REPROCESS
SET SAFETY
SET SECONDS
SET SEPARATOR
SET SYSFORMATS
SET TALK
SET UNIQUE

请参阅

AUSED(), CREATE FORM, DataSession 属性, DataSessionId 属性

SET DATE 命令

指定日期表达式和日期时间表达式的显示格式。

语法

SET DATE [TO] AMERICAN | ANSI | BRITISH | FRENCH | GERMAN | ITALIAN | JAPAN | TAIWAN | USA | MDY | DMY | YMD | SHORT | LONG

说明

下表中列出了有效的设置值及其对应的日期格式：

设置	Format
AMERICAN	mm/dd/yy
ANSI	yy.mm.dd
BRITISH/FRENCH	dd/mm/yy
GERMAN	dd.mm.yy
ITALIAN	dd-mm-yy
JAPAN	yy/mm/dd
TAIWAN	yy/mm/dd
USA	mm-dd-yy

续表

MDY	mm/dd/yy
DMY	dd/mm/yy
YMD	yy/mm/dd
SHORT	日期的短格式由 Windows 中“控制面板”的“短日期样式”决定。（在“控制面板”中打开“区域设置”，再选择“日期”选项卡）
LONG	日期的短格式由 Windows 中“控制面板”的“长日期样式”决定。（在“控制面板”中打开“区域设置”，再选择“日期”选项卡）。注意，当 SET DATE 设置为 LONG 时，{^1601-01-01} 之前的转化为字符串的日期返回为空日期。

日期的默认设置是 AMERICAN。

SET DATE 设置也决定日期在日期时间表达式中的格式。

SET DATE 在当前数据工作期有效。

请参阅

DATE(), DATETIME(), SET CENTURY, SET DATASESSION, SET MARK TO, SETSYSFORMATS

SET DEBUG 命令

设置能否操作调试窗口和跟踪窗口。

语法

SET DEBUG ON | OFF

参数描述

ON

（默认值）能够从 Visual FoxPro 菜单系统中打开调试窗口和跟踪窗口。

OFF

不能从 Visual FoxPro 菜单系统中打开调试窗口和跟踪窗口。不过，当 SET DEBUG 设置为 OFF 时，可以通过 SET ECHO ON 或 ACTIVATE WINDOW DEBUG 打开调试窗口。同样，也可以通过 SET STEP 或 ACTIVATE WINDOW TRACE 打开跟踪窗口。

说明

有关使用“调试”和“跟踪”窗口的详细内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》第十四章“测试和调试应用程序”。

请参阅

SET DEBUGOUT 命令

将调试结果输出到一个文件。

语法

SET DEBUGOUT TO [FileName [ADDITIVE]]

参数描述

FileName

指定一个文件，调试结果将被输出到其中。如果指定的文件不存在，系统会自动创建它；若该文件已存在，而且您没有声明 ADDITIVE 子句，新的内容将覆盖文件中原有的内容。

执行 SET DEBUGOUT TO 命令将停止将调试结果输出到文件，并且关闭该文件。

ADDITIVE

将调试结果追加到指定文件 *FileName* 的尾部，而不覆盖原有的内容。

说明

调试结果中包含 ASSERT 信息，DEBUGOUT 命令的输出，以及在 SET 事件 LIST 命

令中或“事件跟踪”对话框中指定的事件。

请参阅

ASSERT, DEBUG, DEBUGOUT, SET EVENTLIST, SET EVETNTRACKING

SET DECIMALS 命令

指定数值表达式的小数显示位数。

语法

SET DECIMALS TO [nDecimalPlaces]

参数描述

nDecimalPlaces

指定被显示结果的最少的小数显示位数。默认值是 2 位小数。最大为 18 位小数，最少是 0。

说明

SET DECIMALS 指定最少的小数显示位数，用来控制除法、乘法、三角函数和金融函数计算结果的显示。

SET DECIMALS 在当前数据工作期有效。

请参阅

SET DATASESSION, SET FIXED, SET SYSFORMATS

SET DEFAULT 命令

设置默认的驱动器和目录。

语法

SET DEFAULT TO [cPath]

参数描述

cPath

指定下列参数：

驱动器指示符。

含目录名称的驱动器指示符。

子目录名称。

- 使用 MS-DOS 速记符号（\ 或 ..）的上述各项。

说明

SET DEFAULT 将默认目录设置为指定的目录。

Visual FoxPro 在默认目录中搜索文件。默认目录就是启动 Visual FoxPro 的目录，可以在启动程序或 Visual FoxPro 配置文件中指定不同的默认目录。如果 Visual FoxPro 不能在默认目录中找到文件，Visual FoxPro 接着搜寻 Visual FoxPro 路径（如果指定的话）。要指定 Visual FoxPro 路径，可使用 SET PATH。

如果创建文件时没有指定保存位置，则该文件会保存到 Visual FoxPro 的默认目录中。

退出 Visual FoxPro 时，会返回到 Windows。当退出 Windows 时，会返回到启动 Windows 时的驱动器和目录。

注意 SYS(5) 返回默认驱动器，SYS(2003) 返回没有驱动器指示符的默认目录，SYS(5)+SYS(2003) 返回默认的驱动器及目录。

示例

下面命令可以将默认驱动器设置为驱动器 A。

```
SET DEFAULT TO A
SET DEFAULT TO A:
```

下面命令可以指定一个特定的目录。

```
SET DEFAULT TO A:\sales
SET DEFAULT TO C:\sales\data
```

可以指定子目录。如果在驱动器 C 上的根目录是默认的 Visual FoxPro 目录，发出下面命令能够将默认目录设置为 C:\SALES。

```
SET DEFAULT TO sales
```

可以使用 MS-DOS 速记符号。如果当前目录是 C:\SALES\DATA，发出下面命令可将根目录设置为默认目录。

```
SET DEFAULT TO \
```

也可以使用下面命令将默认目录设置为上一层父目录。

```
SET DEFAULT TO ..
```

请参阅

CD | CHDIR, MD | MKDIR, RD | RMDIR, SET PATH, SYS(5), SYS(2003)

SET DELETED 命令

指定 Visual FoxPro 是否处理标有删除标记的记录，以及其他命令是否可以操作它们。

语法

SET DELETED ON | OFF

参数描述

ON

使用范围子句处理记录（包括在相关表中的记录）的命令忽略标有删除标记的记录。

OFF

（默认值）使用范围子句处理记录（包括在相关表中的记录）的命令可以访

问标有删除标记的记录。

说明

如果表索引是根据 DELETED( )命令建立的，那么，使用 DELETED( ) 测试记录状态的查询可以用 Rushmore 技术优化。

可以执行 DELETE – SQL 或 DELETE 命令标记要删除的记录，也可以在浏览窗口或编辑窗口中选择“表”菜单中的“删除记录...”命令标记要删除的记录。

可以执行 RECALL 命令恢复记录，也可以在浏览窗口或编辑窗口中选择“表”菜单中的“恢复记录...”命令恢复记录。

重要提示 如果默认的作用域是当前记录或者只含单个记录，SET DELETED 命令被忽略。INDEX 和 REINDEX 命令总是忽略 SET DELETED 命令，并且给表中所有记录建立索引。

SET DELETED 的作用域是当前数据工作期。

请参阅

DELETE, DELETE-SQL, DELETED(), PACK, RECALL, SET DATASESSION

SET DELIMITERS 命令

指定是否分隔文本框。包含此命令是为了提供向后兼容性。请使用 Format 属性。

SET DEVELOPMENT 命令

在运行程序之前，Visual FoxPro 检测并比较目标文件的产生日期和源程序的创建日期。

语法

SET DEVELOPMENT ON | OFF

参数描述

ON

（默认值）在执行程序前，重编译那些比目标程序更新的源程序。这样可以确保执行程序的最新版本。

OFF
不比较程序的源版本和编译版本。如果 SET DEVELOPMENT 设置为 OFF，可能无法保证执行的程序总是最新的版本。

说明

使用 MODIFY 命令编辑、修改源程序时，将忽略 SET DEVELOPMENT 设置状态，总是执行修改后程序的最新版本。

只有当程序是在 Visual FoxPro 以外修改时，才需要将 SET DEVELOPMENT 设置为 ON。例如，使用 TSR 外部编辑器，执行修改过的程序前可能需要发出 CLEAR PROGRAM 命令。有关详细内容，请参阅 CLEAR PROGRAM。使用 SET DEVELOPMENT OFF 可以优化性能。

当 SET DEVELOPMENT 设置为 ON 时，在 READ 中，运行的程序可以终止。当 SET DEVLOPMENT 设置为 ON，并且 READ 操作被激活时，可以选择“程序”菜单上的“取消”命令，“取消”命令将取消程序的运行。如果 SET DEVELOPMEMT 设置为 OFF，那么在执行 READ 命令时不能选择“程序”菜单上的“取消”命令。

在 Visual FoxPro 中，SET DEVELOPMENT 的设置也决定了在表单运行发生错误时是否打开跟踪窗口。如果 SET DEVELOPMENT 设置为 ON，打开跟踪窗口，并选中引起错误的程序行。如果 SET DEVELOPMENT 设置为 OFF，当表单中发生错误时不打开跟踪窗口。

请参阅

COMPILE, MODIFY COMMAND, MODIFY FILE

SET DEVICE 命令

将 @ ... SAY 的输出结果定向到屏幕、打印机或文件中。

语法

SET DEVICE TO SCREEN | TO PRINTER [PROMPT] | TO FILE *FileName*

参数描述

TO SCREEN

将 @ ... SAY 的输出结果定向到 Visual FoxPro 主窗口或活动的用户自定义窗口中。

TO PRINTER [PROMPT]

将 @ .. SAY 的输出结果定向到打印机。如果 @ .. SAY 命令中指定的页面坐标位置高于前面的 @ .. SAY 命令中指定的页面坐标位置，则发出走纸命令。在 Visual FoxPro 中，包含可选的 PROMPT 子句可以在打印开始之前显示一个对话框，从中可以调整打印机设置，包括打印的副本数目和打印的页码等。当前安装的打印机驱动程序决定了可以调整哪种打印机的设置。PROMPT 应该紧跟在 TO PRINTER 之后。

TO FILE *FileName*

指定一个保存 @ ... SAY 输出结果的文件。

说明

可以将 @ ... SAY 的输出结果定向到 Visual FoxPro 主窗口、活动的用户自定义窗口、打印机和文件。

请参阅

SYS(101)

SET DISPLAY 命令

设置支持多种显示模式的显示器的显示模式。

语法

SET DISPLAY TO CGA | EGA25 | EGA43 | VGA25 | VGA50

参数描述

CGA

将 Visual FoxPro 主窗口中的字号大小设置为 9 磅。

EGA25

将 Visual FoxPro 主窗口中的字号大小设置为 9 磅，将 Visual FoxPro 主窗口的大小设置为 25 行。

EGA43

将 Visual FoxPro 主窗口中的字号大小设置为 7 磅，将 Visual FoxPro 主窗口的大小设置为 50 行。

VGA25

将 Visual FoxPro 主窗口中的字号大小设置为 9 磅，将 Visual FoxPro 主窗口的大小设置为 25 行。

VGA50

将 Visual FoxPro 主窗口中的字号大小设置为 7 磅，将 Visual FoxPro 主窗口的大小设置为 50 行。

说明

SET DISPLAY 可以改变 Visual FoxPro 主窗口中的字号大小。必要时，可根据指定的选项来增加显示的行数，以此增加 Visual FoxPro 主窗口的大小。如果发出 SET DISPLAY 命令，图形状态栏将被关掉。

如果视频硬件不支持指定的选项，Visual FoxPro 将产生错误信息。

无论什么时侯发出 SET DISPLAY 命令，SET MESSAGE 行都将被重置为 Visual FoxPro 主窗口的最后一行

请参阅

SET MESSAGE, SYS(2006)

SET DOHISTORY 命令

决定是否将程序中的命令显示在命令窗口或放在文本文件中。包含此命令是为了提供向后兼容性。请使用跟踪窗口代替此命令。

SetData 方法

将数据放在 OLE 拖放 Data 对象中。只在设计时可用。

语法

oDataobject.SetData(eData [, nFormat | cFormat])

参数描述

eData

指定放在 Data 对象中的数据。如果省略可选的 nFormat 和 cFormat 参数，Visual FoxPro 按 CF_TEXT 和 CFSTR_OLEVARIANT 格式将数据放在 Data

对象中。如果 eData 是一个数组，需要在 eData 前面加一个 @ 记号。数组是按 CFSTR_OLEVARIANTARRAY 格式放在 Data 对象中的。如果 eData 是一个对象引用或通用字段，或者是包含对象引用的数组，则会产生一条错误信息。

NFormat|cFormat

指定放在 Data 对象中的数据的格式。下表列出了一些常用数据格式的值，以及每种格式的说明。使用 cFormat 也可以指定自定义格式。这时，eData 必须是使用 CREATEBINARY() 创建的字符型或二进制型。

数据格式*	nFormat	说明
CF_TEXT	1	文本格式。
CF_OEMTEXT	7	包含 OEM 字符集中字符的文本格式。
CF_UNICODETEXT	13	Unicode 文本格式，只在 Windows NT 下可用
CF_FILES 或者 CF_HDROP	15	一个标识一组文件的句柄。例如从 Windows 资源管理器拖来的一组文件。

续表

CFSTR_OLEVARIANTARRAY	“OLE Variant Array”	一个 Visual FoxPro 数组。使用这个格式在一次拖放中可以传送多个值。例如，这个格式可以用于将列表框中的一些项拖动到另一个列表框中。
CFSTR_OLEVARIANT	“OLE Variant”	一个变量。Visual FoxPro 中的所有数据类型都可以用变量代表。这个格式可以用于拖放 Visual FoxPro 数据，并且不丢失数据类型。
CFSTR_VFPSOURCEOBJECT	“VFP Source Object”	对一个 Visual FoxPro 拖动源对象的引用。

* 在 FOXPRO.H 中定义的。

说明

SetData 方法只能在 OLEStartDrag 事件中执行。

应用于

Data 对象

请参阅

ClearData 方法, CREATEBINARY( ), GetData 方法, GetFormat 方法,

OLE Drag-and-Drop 概述, OLEStartDrag event, SetFormat 方法

SET ECHO 命令

为调试程序打开跟踪窗口。已包含向后兼容的特性。可以使用“跟踪”窗口替代。

语法

SET ECHO ON | OFF

参数描述

ON

在跟踪窗口中显示正在执行的程序源代码。突出显示当前正在运行的程序行。

OFF

（默认值）关闭 FoxPro 2.0 以前版本中的跟踪窗口。在程序中使用 DEACTIVATE WINDOW TRACE 命令可以关闭跟踪窗口。

说明

使用跟踪窗口还可以为挂起的运行程序设置断点。为了后兼容而提供了这个命令。请使用跟踪窗口。

有关使用“调试”和“跟踪”窗口的详细内容，请参阅《Microsoft Visual FoxPro 6.0

中文版程序员指南》第十四章“测试和调试应用程序”。

请参阅

RESUME, SET STEP, SUSPEND

SET ESCAPE 命令

决定是否可以通过按 ESC 键中断程序和命令的运行。

语法

SET ESCAPE ON | OFF

参数描述

ON

（默认值）当按 ESC 键时中断命令和程序的运行。

若在某命令或程序运行期间按 ESC 键，当插入点在命令窗口中时，将出现下列信息：

*** 中断 ***

如果命令或程序运行期间按 ESC 键，在完成处理当前程序行后，出现有如下三种选项的警告。

- （默认值）选择“取消”，立即中止程序的运行并返回到命令窗口。

- 选择“挂起”，暂停程序的运行并返回到命令窗口。这个选项对调试程序非常有用。从“程序”菜单中选择“继续执行”或在命令窗口中发出 RESUME 命令能够从暂停运行程序行处重新启动程序。
- 选择“忽略”，能够从暂停行处继续运行程序。

OFF

禁止运行的程序和命令在按 ESC 键后被中断。

请参阅

ON ESCAPE, ON KEY LABEL

SET EVENTLIST 命令

指定被跟踪的事件，这些事件将显示在“调试输出”窗口中或输出到由 SET 事件 TRACKING 命令指定的文件中。

语法

SET EVENTLIST TO [EventName1 [, EventName2 ...] [ADDITIVE]]

参数描述

Event*Name1* [,Event*Name2* ...]

指定被跟踪的事件。您可以指定任意数目的事件，事件之间用逗号分隔。

ADDITIVE

若当前已存在一个被跟踪的事件序列，并且您在命令中使用了 ADDITIVE **子句，则** Event*Name1*, Event*Name2* ... **将被追加到已有的事件序列中；若没有使用** ADDITIVE **子句，** Event*Name1*, Event*Name2* ... **将替代已有的序列。**

说明：

SET EVENTLIST TO **命令若不带任何参数，将删除所有事件序列。您还可以在“事件跟踪”对话框中编辑事件序列。**

请参阅

SET DEBUGOUT, SET EVENT TRACKING

SET EVENTTRACKING **命令**

允许或关闭事件跟踪，或者指定事件跟踪信息输出到一个文本文件中。

语法

SET EVENTTRACKING ON | OFF | PROMPT
TO [FileName [ADDITIVE]]

参数描述

ON

打开事件跟踪，如果指定了文本文件 *FileName*，则将事件跟踪的信息输出到该文件中。

OFF

关闭事件跟踪，不将跟踪信息输出到文本文件中。

PROMPT

显示“事件跟踪”对话框，允许用户交互式地编辑事件跟踪序列。

TO *FileName*

指定事件跟踪信息输出到文本文件 *FileName*。若要将事件跟踪信息输出到文本文件，SET 事件 TRACKING 必须设置为 ON。SET EVENTTRACKING TO 而不指定任何文件，将关闭以前打开的文本文件。

如果您指定的文本文件不存在，Visual FoxPro 会自动创建并打开它。

ADDITIVE

将事件跟踪信息追加到指定的文本文件 *FileName* 尾部。若省略 ADDITIVE 子句，文件中的原有信息将被覆盖。

说明：

您可以使用 SET EVENTLIST 命令或在“事件跟踪”对话框中指定被跟踪的事件。

请参阅

SET COVERAGE, SET EVENTLIST

SET EXACT 命令

设置不同长度字符串的比较规则。

语法

SET EXACT ON | OFF

参数描述

ON

指定相等的表达式必须是每个字符都相匹配。比较时，忽略表达式结尾的空格。比较两个表达式时，在较短一个的右边加上空格，以使它与较长表达式的长度相匹配。

OFF

（默认值）指定必须是右端表达式结尾前的每个字符都相匹配，才是相等的表达式。

说明

SET EXACT 设置值对于相等长度的字符串无效。

字符串比较

Visual FoxPro 有两个关系操作符用来测试字符串是否相等。

操作符“=”在两个相同类型的值之间进行比较。该操作符适用于比较字符、数值、日期和逻辑数据。

但是，当使用操作符“=”比较两个字符表达式时，比较结果可能与预期的不完全相同。字符表达式比较时是从左到右逐个字符进行比较，一直到两个表达式中的对应字符不相等，或者到达操作符右端表达式的末端 (SET EXACT OFF)，或者到达两个表达式的末端 (SET EXACT ON)。

当需要两个完全相同的字符数据时可以使用操作符“==”。如果两个字符表达式使用操作符“==”进行比较时，操作符两边的表达式必须包含完全一样的字符（包括空格），才认为是相等的。使用操作符“==”进行字符串比较时忽略 SET EXACT 设置。

下表说明选择的操作符和 SET EXACT 设置如何影响比较（下划线代表空格）。

比较	= EXACT OFF	= EXACT ON	== EXACT ON or OFF
“abc” = “abc”	匹配	匹配	匹配
“ab” = “abc”	不匹配	不匹配	不匹配
“abc” = “ab”	匹配	不匹配	不匹配
“abc” = “ab_”	不匹配	不匹配	不匹配
“ab” = “ab_”	不匹配	匹配	不匹配

续表

“ab_” = “ab”	匹配	匹配	不匹配
“” = “ab”	不匹配	不匹配	不匹配
“ab” = “”	匹配	不匹配	不匹配
“__” = “”	匹配	匹配	不匹配
“” = “___”	不匹配	匹配	不匹配
TRIM(“___”)= “”	匹配	匹配	匹配
“” =TRIM(“___”)	匹配	匹配	匹配

SET EXACT 的作用域是当前数据工作期。

请参阅

SET ANSI

SET EXCLUSIVE 命令

指定 Visual FoxPro **在网络上以独占方式还是共享方式打开表文件。**

语法

SET EXCLUSIVE ON | OFF

参数描述

ON
（全局数据工作期的默认方式）网络上表的访问权限只给打开表的用户。网络上的其他用户不能存取该表。与 FLOCK( ) 不同，SET EXCLUSIVE ON 还防止所有其他用户以只读方式访问该表。在 USE 命令中加入 EXCLUSIVE 子句使文件在网络上以独占方式打开，在此方式下，不必锁定该表的记录和文件。
以独占方式打开表，能够确保其他用户不能更改文件的内容。有些命令只能处理以独占方式打开的表。这些命令有 INSERT，INSERT BLANK，MODIFY STRUCTURE，PACK，REINDEX 和 ZAP。

OFF
（私有数据工作期的默认方式）允许网络上的任何用户共享和修改网络上打开的表。

有关文件和记录锁定及网络上的共享表的其他内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》第十七章“共享访问程序设计”。

说明

改变 SET EXCLUSIVE 的设置并不改变已经打开表的状态。例如，如果一个表是在 SET EXCLUSIVE 设置为 ON 时打开的，当把 SET EXCLUSIVE 的设置改变为 OFF

时，表仍然保持原来的独占状态。

SET EXCLUSIVE 的作用域是当前数据工作期。

请参阅

FLOCK(), RLOCK() , SET DATASESSION, USE

SET FDOW 命令

指定一周中的第一天。

语法

SET FDOW TO [*nExpression*]

参数描述

nExpression

指定一周中的第一天。下表列出了 *nExpression* 可以取的值，以及对应的星期几将被设置成一周中的第一天。

NExpression	星期几
1	星期日

续表

2	星期一
3	星期二
4	星期三
5	星期四
6	星期五
7	星期六

如果省略 *nExpression*，一周中的第一天重新设置为星期天 (1)。

说明

一周中的第一天也可以通过选择“选项”对话框的国际选项卡中的“星期开始于”复选框来设置。

示例

```
STORE SET('FDOW') TO gnFdow && 保存当前值
SET FDOW TO 1 && 默认值，将星期天设置为一周的第一天
SET FDOW TO 7 && 将星期六设置为一周的第一天
SET FDOW TO &gnFdow &&还原初始值
```

请参阅

CDOW(), DAY(), DOW(), SET FWEEK, SYS()函数概述, WEEK()

SET FIELDS 命令

指定可以访问表中的哪些字段。

语法

SET FIELDS ON | OFF | LOCAL | GLOBAL

– 或 –

SET FIELDS TO [[FieldName1 [, FieldName2 ...]]
| ALL [LIKE Skeleton | EXCEPT Skeleton]]

参数描述

ON

指定只能访问出现在字段列表中的字段。

OFF

（默认值）指定可以访问当前表中的所有字段。

LOCAL

只能访问字段列表中当前工作区的字段。

GLOBAL

可以访问字段列表中的所有字段，包括在其他工作区中的字段。

SET FIELDS GLOBAL **允许在不发出** SET COMPATIBLE TO DB4 **命令时，**

也可以访问其他工作区中的字段。

TO [*FieldName1* [, *FieldName2* ...]]

指定当前表中可访问的字段的名称。在下列情况下，必须包括字段名称的别名：

- 当字段所在的表不在当前工作区中时。
- 当在两个或两个以上表中字段的名称相同时。

在字段名称的前面加上表的别名，可以在字段列表中包括其他工作区中打开的表的字段。但是，这些字段只有在发出了 SET FIELDS GLOBAL 或 SET COMPATIBLE DB4 命令后才能访问。

字段列表可以包含创建计算结果字段的语句。计算结果字段包含由表达式创建的只读数据。这个表达式可以是任何形式，不过它必须是有效的 Visual FoxPro 表达式。在发出了 SET FIELDS GLOBAL 或 SET COMPATIBLE DB4 命令之后，才能访问计算结果字段。

用于创建计算结果字段的语句格式如下：

<calculated field name> = <expr>

下面的示例创建一个名为 LOCATION 的计算结果字段：

```
CLOSE DATABASES
USE customer
SET FIELDS TO LOCATION = ALLTRIM(city) + ', ' + state
```

CITY 和 STATE 都是选定表中的字段名称。

ALL

允许访问当前表中的所有字段。

ALL LIKE *Skeleton* | EXCEPT *Skeleton*

可以单独或组合使用 LIKE 和 EXCEPT 子句，有选择地访问字段。如果使用 LIKE *Skeleton*，可以访问与 *Skeleton* 相匹配的字段。如果使用 EXCEPT *Skeleton*，可以访问不与 *Skeleton* 相匹配的字段。

Skeleton 支持 * 和 ? 通配符。例如，要访问所有以字母 A 和字母 P 开头的字段，可发出如下命令：

```
SET FIELDS TO ALL LIKE A*,P*
```

LIKE 子句可以和 EXCEPT 子句联合起来使用：

```
SET FIELDS TO ALL LIKE A*,P* EXCEPT PARTNO*
```

说明

SET FIELDS TO 可以添加字段。发出带字段列表的 SET FIELDS TO 命令，可以使指定的字段也成为当前可访问的字段。

发出 SET FIELDS TO 命令隐含地执行了 SET FIELDS ON 命令。发出既不包含字段列表也不包含 ALL 的 SET FIELDS TO 命令将从当前表中移去字段列表上的所有字段，使每个字段都不能被访问。

SET FIELDS 的作用域是当前数据工作期。

请参阅

SET FILTER, SET DATASESSION

SET FILTER 命令

指定访问当前表中记录时必须满足的条件。

语法

SET FILTER TO [*lExpression*]

参数描述

lExpression

指定记录必须满足的条件。

如果当前表在 *lExpression* 中指定的一个或多个字段上建立了索引，Visual FoxPro Rushmore 技术会优化基于这个或这些字段的查询。

说明

如果使用 SET FILTER 命令，则表中只有满足逻辑表达式 *lExpression* 指定的条件的记录才可以被访问。所有访问该表的命令都必须遵守 SET FILTER 指定的条件。对于每个打开的表，都可以设置单独的筛选条件。

只有当记录指针在表中移动时，才计算由 SET FILTER 指定的条件。

不带 *lExpression* 发出 SET FILTER TO 命令将关闭当前表的选择器。

注意 SELECT - SQL 不遵守当前的筛选条件。

请参阅

FILTER(),SELECT – SQL

SET FIXED 命令

在显示数值时，指定小数的显示位置。

语法

SET FIXED ON | OFF

参数描述

ON

要确定在结果中显示的小数位数，应使用 SET DECIMALS **设置。默认的小数位数是** 2。

OFF

（默认值）允许根据数值表达式中特定的常数、变量和操作符来决定显示结果中保留的小数位数。字段中的内容以声明的小数位数显示。

说明

SET FIXED **的作用域是当前数据工作期**

请参阅

SET DECIMALS, SET DATASESSION

SET FORMAT 命令

包含此变量是为了提供向后兼容性，可用 Format **属性代替。**

SET FULLPATH 命令

指定 CDX()，DBF()，MDX() **和** NDX() **等函数是否返回一个文件的路径。**

语法

SET FULLPATH ON | OFF

参数描述

ON

（默认值）指定 CDX()、DBF()、MDX() **和** NDX() **等函数返回驱动器号、**

路径和文件名。

OFF

指定只返回驱动器号和文件名。

请参阅

CDX( ), DBF( ) , MDX( ) , NDX( )

SET FUNCTION 命令

定义功能键。

语法

SET FUNCTION *nFunctionKeyNumber* | *KeyLabelName* TO [*eExpression*]

参数描述

nFunctionKeyNumber

指定功能键的编号，将向该功能键分配宏。

KeyLabelName

指定组合键，此函数将向该组合键分配宏。Visual FoxPro 支持包含功能键的组合键。可以把 CTRL 或 SHIFT 键同功能键组合起来创建其他可编程键。有关

键标记表达式的列表，请参阅 ON KEY LABEL 命令。

TO [*eExpression*]

指定保存到功能键或组合键中的一系列击键字母。Visual FoxPro 把表达式中的分号理解为回车键。可以用 CLEAR MACROS 命令清除功能键的定义。

请参阅

CLEAR MACROS, FKLABEL( ), FKMAX( ) , ON KEY LABEL

SET FWEEK 命令

指定一年的第一周要满足的条件。

语法

SET FWEEK TO [*nExpression*]

参数描述

nExpression

指定一个值，它决定了一年的第一周要满足的条件。下表列出了 *nExpression* 可以取的值，及其要满足的条件。

nExpression	第一周要满足的条件
1	（默认值）包含 1 月 1 日的那一周
2	第一周的多数（含 4 天）日期在当前年中
3	第一个整周

如果省略了 *nExpression*，一年的第一周被重新设置为 1（这周包含 1 月 1 日）。

说明

一年的第一周可以通过选择“选项”对话框的国际选项卡中的“一年的第一周”复选框来设置。

示例

```
STORE SET('FWEEK') TO gnFweek && 保存当前值
SET FWEEK TO 1 && 第一周包含 1 月 1 日
SET FWEEK TO 3 && 第一周有 7 天
SET FWEEK TO &gnFweek && 还原初始设置
```

请参阅

CDOW(), DAY(), DOW(), SET FDOW, SYS() 函数概述, WEEK()

SetFormat 方法

指定 OLE 拖放 Data 对象的数据格式。只在设计时可用。

语法

oDataobject.SetFormat(nFormat | cFormat)

参数描述

nFormat | cFormat

指定 Data 对象的数据格式。下表列出了一些常用数据格式的值，以及每种格式的说明。使用 cFormat 也可以指定自定义格式。

数据格式*	nFormat	说明
CF_TEXT	1	文本格式。
CF_OEMTEXT	7	包含 OEM 字符集中字符的文本格式。
CF_UNICODETEXT	13	Unicode 文本格式，只在 Windows NT 下可用。

续表

CF_FILES 或 CF_HDROP	15	一个标识一组文件的句柄。例如从 Windows 资源管理器拖来的一组文件。
CFSTR_OLEVARIANTARRAY	“OLEVariant Array”	一个 Visual FoxPro 数组。使用这个格式在一次拖放中可以传送多个值。例如，这个格式可以用于将列表框中的一些项拖动到另一个列表框中。
CFSTR_OLEVARIANT	“OLE Variant”	一个变量。Visual FoxPro 中的所有数据类型都可以用变量代表。这个格式可以用于拖放 Visual FoxPro 数据，并且不丢失数据类型。
CFSTR_VFPSOURCEOBJECT	“VFP Source Object”	对一个 Visual FoxPro 拖动源对象的引用。

* 在 FOXPRO.H 中定义的。

说明

可以在向 Data 对象中放置相应数据之前，向 Data 对象中放置数据格式。如果在 Data 对象中放置了数据格式，而不包含相应的数据，并且在 OLEDragDrop 事件中激活了 SetData Method，则拖动源发生 OLESetData 事件。然后，该拖动源在 OLESetData 事

件中使用 SetData Method 将数据放在 Data 对象中。

当需要将大量数据放在 Data 对象时，当使用 Visual FoxPro 在本地不支持的数据格式时，或者当使用大量数据格式时，只在 Data 对象上放置数据格式，会提高 OLE 拖放的性能。

SetFormat Method 只能在 OLEStartDrag 和 OLESetData 事件中执行。

应用于

DataObject 对象

请参阅

ClearData 方法, GetData 方法, GetFormat 方法, OLE Drag-and-Drop 概述, OLEDragDrop 事件, OLESetData 事件, OLEStartDrag 事件, SetData 方法

SET HEADINGS 命令

指定用 TYPE 显示文件内容时，是否显示字段的列标头，并指定是否包含文件信息。

语法

SET HEADINGS ON | OFF

参数描述

ON
（默认值）指定显示字段的名称。
如果发出 TYPE 命令来显示文件内容时，Visual FoxPro 在显示结果的前面插入换页符、文件的路径和名称以及日期。
OFF
指定不显示字段的名称。
如果发出 TYPE 命令来显示文件的内容，Visual FoxPro 不在显示的结果中插入其他信息。

说明

指定 AVERAGE、CALCULATE、DISPLAY、LIST 和 SUM 的输出结果在显示时是否将字段名显示为列标头。

请参阅

AVERAGE, CALCULATE, DISPLAY, LIST, SUM, TYPE

SET HELP 命令

激活或废止 Visual FoxPro **联机帮助或指定的帮助文件。**

语法

SET HELP ON | OFF

— 或者 —

SET HELP TO [*FileName*]

参数描述

ON

（默认值）当按下 F1 **键或在命令窗口中执行** HELP **命令时，显示帮助窗口。**

OFF

使 Visual FoxPro **联机帮助不可用。**

TO [*FileName*]

指定当按 F1 **键或发出** HELP **时显示的帮助文件。可以指定一个** .Dbf **型帮助文件、一个** Winhelp (.Hlp) **文件，或一个** HTML (.Chm) **帮助文件。**

如果执行不包含文件名的 SET HELP TO **命令，** Visual FoxPro **会搜索** MSDN **帮助文件** Msdnvs98.col。**在** Visual FoxPro 6.0 **中，使用** SET HELP TO **命令和**

一个帮助文件名，不关闭任何已打开的帮助文件。

说明

使用 SET HELP 可以为一个自定义应用程序提供简洁的联机帮助文件，或者在 Visual FoxPro 中在不同的帮助文件之间切换。

如果在安装 Visual FoxPro 时安装了 MSDN (Microsoft Developer's Network)，则默认的帮助文件是 MSDN 帮助文件 Msdnvs98.col。

如果完全安装了 MSDN (Microsoft Developer's Network) 库，或者在定制安装 MSDN 库时指定安装 Visual FoxPro 文档，则安装 Visual FoxPro 帮助文件 Foxhelp.chm。

如果不安装 MSDN 库，则不安装任何帮助文件。

也可以使用“选项”对话框“文件位置”文件夹的“帮助文件”选项，交互地指定一个帮助文件。

请参阅

HELP, SET HELPFILTER, SET TOPIC

SET HELPFILTER 命令

在帮助窗口中，让 Visual FoxPro 显示 .DBF 样式帮助主题的一个子集。

语法

SET HELPFILTER [AUTOMATIC] TO [*lExpression*]

参数描述

AUTOMATIC

在关闭帮助窗口后，自动删除使用 SET HELPFILTER 命令指定的条件。在命令中包含 AUTOMATIC 等效于在关闭帮助窗口后紧接着发出 SET HELPFILTER TO 命令。AUTOMATIC 必须直接放在 TO *lExpression* 前面。

lExpression

指定用于筛选“帮助”主题的逻辑表达式，只显示 *lExpression* 计算值为“真”(.T.) 的帮助主题。通常，*lExpression* 是一个字段名。

说明

只能对 .DBF 样式的帮助设置过滤器，对于图形方式帮助不能设置。

在 Visual FoxPro 中，只在专业版中才包含 .DBF 样式帮助。有关详细内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》。

请参阅

HELP, SET HELP, SET TOPIC

SET HOURS 命令

将系统时间设置为 12 小时制或 24 小时制。

语法

SET HOURS TO [12 | 24]

参数描述

TO 12

（默认值）指定为 12-小时时间格式。

TO 24

指定为 24-小时时间格式。

说明

使用不带 12 或 24 的 SET HOURS TO 命令，将返回到默认的 12-小时时间格式。

TIME( ) **总是返回 24 小时时间格式的时间值，并且不受 SET HOURS 的影响。**

DATETIME( ) **函数的返回值格式由当前 SET HOURS 的设置决定。**

SET HOURS 的设置只对当前数据工作期有效。

请参阅

DATETIME(), SECONDS() , SET CLOCK, SET SYSFORMATS, TIME()

SET INDEX 命令

打开一个或多个索引文件，供当前表使用。

语法

```
SET INDEX TO [IndexFileList | ? ]
  [ORDER nIndexNumber | IDXIndexFileName
  | [TAG] TagName [OF CDXFileName] [ASCENDING | DESCENDING]]
  [ADDITIVE]
```

参数描述

IndexFileList

指定要打开的一个或多个索引文件。在索引文件列表中，应使用逗号分隔多个索引文件。索引文件列表中，可包含任意个 .IDX 或 .CDX 索引文件名。除非有相同名字的 .IDX 或 .CDX 文件存在，否则没有必要包括文件扩展名。

索引文件列表中，第一个索引文件将成为主控索引文件，它控制记录的访问和显示。如果第一个索引文件是 .CDX 文件，并且没有发出 SET ORDER TO TAG 命令，则按记录的物理顺序显示和访问记录。

?

显示“打开”对话框，从这个对话框中可以打开单个的 .IDX 文件。

ORDER nIndexNumber

指定一个主控索引文件或标识，数值表达式 *nIndexNumber* 指定在索引文件列表中出现的索引文件。首先，按索引文件列表中出现的顺序给 .IDX 文件编号；然后，按照创建标识的先后顺序，对结构化的 .CDX 文件（如果存在的话）中的标识进行编号；最后，按照创建文件的顺序，对独立的 .CDX 文件中的标识进行编号。有关对索引文件和标识编号的详细内容，请参阅 SET ORDER。

如果 *nIndexNumber* 为 0，表中记录以物理顺序显示和访问，而索引文件保持打开状态。以物理顺序访问记录时，ORDER 0 能够更新打开的索引文件。

不带参数的 ORDER 命令等价于 ORDER 0。

如果 *nIndexNumber* 大于 .IDX 文件和 .CDX 文件标识的数目，Visual FoxPro 出现出错信息。

ORDER IDXIndexFileName

指定一个 .IDX 索引文件作为主控索引文件。

ORDER [TAG] TagName*e* [OF CDXFileName*e*]

指定 .CDX 文件中的一个标识 (*TagName*) 作为主控标识。标识名来自结构

化 .CDX 文件或打开的独立 .CDX 文件。在打开的独立 .CDX 文件中，如果存在相同名称的标识，请使用 OF *CDXFileName* 指定标识所在的 .CDX 文件。

ASCENDING | DESCENDING

指定显示和访问表记录时，是以升序还是以降序进行。索引文件或索引标识不作任何变化，只改变记录显示和访问的顺序。请将 ASCENDING 或 DESCENDING 关键字紧跟在 ORDER 子句后面。

ADDITIVE

指定前面打开的一个索引文件，除了结构化复合索引，在发出 SET INDEX 命令打开另一个索引文件或表文件时关闭。没有 ADDITIVE 子句，会关闭所有前面打开的文件。

说明

在有索引文件的表中，记录的显示顺序和访问顺序可由某个索引文件来决定。使用 SET INDEX 可以打开单索引 (.IDX) 和复合索引 (.CDX) 文件。如果一个表有结构化的 .CDX 文件，打开表时该文件自动打开。只有一个 .IDX 文件（主控索引文件）或 .CDX 中的标识（主控标识）控制表中记录的显示和访问顺序。某些命令（例如，SEEK）使用主控索引文件或主控索引标识来搜索记录。

执行不带参数的 SET INDEX TO 命令，会关闭当前工作区中所有打开的索引文件（结构化 .CDX 文件除外）。

请参阅

CLOSE INDEXES, INDEX, SET ORDER, USE

SET INTENSITY 命令

确定 Visual FoxPro 是否使用增强屏幕颜色属性显示字段。包含此命令是为了提供向后兼容性。可使用 SET COLOR OF SCHEME 代替。

SET KEY 命令

根据索引关键字，指定访问记录的范围。

语法

SET KEY TO [eExpression1 | RANGE eExpression2 [, eExpression3]]
[IN cTableAlias | nWorkArea]

参数描述

eExpression1

允许使用相同的索引关键字访问一组记录。*eExpression1* 是单个索引关键字的值，所有索引关键字与 *eExpression1* 相匹配的记录都是可访问的。

RANGE [,*eExpression2 eExpression3*]

可以访问关键字值在一定范围内的记录。*eExpression2* 允许访问关键字大于等于 *eExpression2* 的记录，*eExpression3*（前面用逗号分开）允许访问关键字值小于等于 *eExpression3* 的记录。对于同时包含 *eExpression2* 和 *eExpression3*（用逗号分隔它们）的 SET KEY 命令，其允许访问大于等于 *eExpression2* 且小于等于 *eExpression3* 的所有记录。

例如，CUSTOMER 表中有一个包含美国邮政编码的字符字段。如果该表在此字段上建立索引，就可以使用 SET KEY 来指定邮政编码的范围。

在下例中，只有邮政编码在 40000 到 43999 范围内的记录才出现在窗口中。

```
CLOSE DATABASES
USE customer
SET ORDER TO postalcode
SET KEY TO RANGE '40000', '43999'
BROWSE
```

IN *cTableAlias* | *nWorkArea*

在指定工作区内打开的表中，允许访问一定范围的记录。*cTableAlias* 指定工作区的别名，*nWorkArea* 指定工作区的编号。如果没有指定的表别名，Visual

FoxPro 产生错误信息；如果省略工作区别名和编号，SET KEY 仅对当前选定工作区中的表进行操作。

说明

使用 SET KEY 限制表中可以访问的记录范围。表必须建立了索引，并且指定的索引关键字值必须与主索引文件或主标识的索引表达式有相同的数据类型。

发出不带任何参数的 SET KEY TO 命令，可以恢复访问表中所有记录。

请参阅

INDEX, KEY(), SET FILTER

SET KEYCOMP 命令

控制 Visual FoxPro 的键击定位。

语法

SET KEYCOMP TO DOS | WINDOWS

说明

SET KEYCOMP 决定用来访问 Visual FoxPro 界面上控件（诸如按钮、列表框、菜单等）的键击和键击组合，SET KEYCOMP 的作用与控件有关。

当想使用自己熟悉的键击时，请使用 SET KEYCOMP。

用 MS-DOS 的按键在 Microsoft 窗口中浏览，使用如下命令：

```
SET KEYCOMP TO DOS
```

SET KEYCOMP TO DOS 可以指定 DOS 或 WINDOWS（默认值）选项。

可以在 Visual FoxPro 的配置文件 CONFIG.FPW 中指定 SET KEYCOMP 启动设置。

以下部分描述 DOS 和 WINDOWS 选项如何影响 Visual FoxPro。

默认按钮	
DOS	对话框中的默认按钮具有焦点，并且它的外观不变化。当按 CTRL+ENTER 时，会选定它。
WINDOWS	当在控件中移动活动焦点时，对话框中的默认按钮外观可能变化：它可能变暗，或拥有焦点（四周是粗体黑边）来指示它是当前默认按钮。按 ENTER 键可以选定默认按钮，并且总是执行默认按钮的操作。 为了说明对话框中默认按钮是如何改变外观的，请发出 SET KEYCOMP TO WINDOWS 命令，然后按 TAB 键进入“打开”对话框。

访问键	
DOS	控件的访问键是一个单键。如果不在一个有键盘控制的控件（组合框或列表框）中，可以按快捷键来选择它。

续表

WINDOWS	控件的访问键既可以是单键，也可以是一个组合键。如果当前控件有键盘控制（组合框或列表框），可以按 Alt 加上快捷键来选择该控件。要选择其他控件，可以按快捷键或 Alt 加上快捷键。
组合框	
DOS	当组合框有焦点时，可以按 ENTER 键或 SPACEBAR 键打开它，在组合框没有打开之前，组合框中的键盘控制不可用。
WINDOWS	当组合框有焦点时，可以通过按 SPACEBAR、ALT+UP ARROW 或 ALT+DOWN ARROW 来打开它。当组合框有焦点或已打开时，可以使用组合框的键盘控制。例如，选定的组合框包含一列可用的驱动器。如果驱动器 B 是当前驱动器，且驱动器 A、B 和 C 可以使用，可以在没有打开组合框时，按 C 或 DOWN ARROW 键来选择驱动器 C。
选项按钮	
DOS	当选中了一组选项按钮，想在选项按钮之间移动时，按 Tab 键。
WINDOWS	当选中了一组选项按钮时，想从选项按钮移动到其他控件时，按 Tab 键。要在选项按钮组中移动，按 Up Arrow 和 Down Arrow 键。
浏览窗口	
DOS	当输入字段时，不选中字段。

续表

WINDOWS	当输入字段时，自动选中字段。

请参阅

SET COMPATIBLE

SET LIBRARY 命令

打开一个外部的 API（应用程序接口）库文件。

语法

SET LIBRARY TO [*FileName* [ADDITIVE]]

参数描述

FileName

指定要打开的 API 库文件名或过程文件名。

Visual FoxPro **假定库文件有 .FLL 扩展名。如果库文件有 .FLL 扩展名，就不必在文件名中写入扩展名；如果库文件的扩展名不是 .FLL，就必须在文件名中写上扩展名。**

要点 使用该命令时，请注意以下几点：

在一个平台上，不能使用在其他平台上建立的 API 库。例如，为 FoxPro for MS-DOS 建

立的库，就不能在 Visual FoxPro 中使用；为 Visual FoxPro 建立的库，则不能在 FoxPro for MS-DOS 中使用。

在一个版本中，不能使用另一个版本的 API 库。例如，在 Visual FoxPro 中，不能使用 FoxPro for Windows 2.6版建立的库，必须用 Visual FoxPro 专业版重新编译并链接。

Visual FoxPro 假定过程文件的扩展名为 .PRG。

当使用 DO *ProcedureName* 执行一个过程时，Visual FoxPro 以如下顺序在下列文件中查找过程：

1. 在包含 DO *ProcedureName* 命令的文件中进行查找。
2. 使用 SET PROCEDURE 命令打开的过程文件（如果存在的话）。
3. 在执行链中的程序。Visual FoxPro 从最近执行的程序中开始查找程序文件，一直到第一个执行程序。
4. 使用 SET LIBRARY 命令打开的过程文件（如果设置过）。
5. 单独的程序文件。如果 Visual FoxPro 找到一个与 DO 指定的文件名相同的程序文件时，执行这个程序；如果找不到匹配的程序文件名，Visual FoxPro 产生出错信息。

ADDITIVE

打开附加的 API 库。ADDITIVE 放在 SET LIBRARY 命令中文件名后面。

当使用 SET LIBRARY 打开一个过程文件时，Visual FoxPro 忽略 ADDITIVE 子句。

说明

使用 SET LIBRARY 打开外部应用程序接口 (API) 库或过程文件。

API 例程库可以增强 Visual FoxPro 语言能力和用户界面。如果打开一个外部 API 库，

就可以像使用 Visual FoxPro 函数一样使用 API 函数。要显示库中可用的函数，请使用 DISPLAY STATUS 或 LIST STATUS 命令。

在 Visual FoxPro 专业版中，可以使用已有 API 函数，也可以创建自己的 API 库。

要从内存中移去所有 API 函数，请使用不包括 *FileName* 和 ADDITIVE 的 SET LIBRARY TO 命令；要从内存中移去单个库，请使用 RELEASE LIBRARY *LibraryName* 命令。

如果指定过程文件，那么所有程序都可使用过程文件中的过程，并且也可以在命令窗口中，以交互方式使用这些过程。

注意 Visual FoxPro 使用 SET LIBRARY 打开过程文件的功能与 dBASE IV 兼容。使用 SET LIBRARY 打开过程文件会关闭所有打开的 API 库。使用 SET LIBRARY 打开 API 库会关闭用 SET LIBRARY 打开的过程文件，请使用 SET PROCEDURE 打开过程文件，防止关闭 API 库。

有关过程文件的详细内容，请参阅 PROCEDURE 和 SET PROCEDURE。

请参阅

CALL, DISPLAY STATUS, LIST, LOAD, RELEASE

SET LOCK 命令

激活或废止在某些命令中的自动锁定功能。

语法

SET LOCK ON | OFF

参数描述

ON

当执行下列命令时，指定具有自动锁定表的功能。这时，网络上的其他用户只能读该表，因此确保您使用的是最新的数据。

OFF

（默认值）使下列命令能以共享方式访问表。如果不必从表中获得最新的信息，请使用 SET LOCK OFF。

说明

在执行需要只读访问表的命令时，Visual FoxPro 不对文件加锁。这些命令如下：

AVERAGE	CALCULATE
COPY TO	COPY TO ARRAY
COUNT	DISPLAY（在一定范围内）

续表

INDEX	JOIN (多个文件)
LIST	LABEL
REPORT	SORT
SUM	TOTAL

当执行这些命令时，不改变表中的内容，并且访问表时，网络上的其他用户也可以访问该表。当使用这些命令时，表可以被其他用户更改。例如，在其他用户改变在报表中的记录前，您可以使用 REPORT 命令打印报表。但是，报表中就包含了过时信息。SET LOCK 的作用域是当前数据工作期。

请参阅

FLOCK( ), LOCK( ) , RLOCK( ) , SET DATASESSION, SET MULTILOCKS

SET LOGERRORS 命令

决定 Visual FoxPro 是否将编译错误信息送入文本文件。

语法

SET LOGERRORS ON | OFF

参数描述

ON
（默认值）创建与被编译的程序有相同文件名，扩展名为 .ERR 的编译错误信息日志文件。如果存在相同名称的日志文件，将改写它。

OFF
编译程序时，不创建编译错误信息日志文件。

说明

编译程序时，可使用 SET LOGERRORS 命令将编译错误信息保存到文本文件中。如果存在与被编译程序相同名称的日志文件，并且程序编译没有错误，删除日志文件。

请参阅

COMPILE

SET MACKEY 命令

指定单个键或组合键，激活“宏键定义”对话框。

语法

SET MACKEY TO [KeyLabelName]

参数描述

KeyLabelName

指定显示“宏键定义”对话框的单个键或组合键。有关如何使用键标记的详细内容，请参阅 ON KEY LABEL。

说明

使用 SET MACKEY **命令可以更改用于显示“宏键定义”对话框的默认组合键。从“工具”菜单中选择“宏”命令可以显示这个对话框。**

请参阅

CLEAR, ON KEY LABEL, PLAY MACRO, RESTORE MACROS, SAVE MACROS

SET MARGIN **命令**

设置打印的左页边距，对所有定向到打印机的输出结果都有效。

语法

SET MARGIN TO *nColumns*

参数描述

nColumns

以列为单位指定左页边距。默认的左页边距为 0 列，最大的左页边距为 256 列。

说明

如果用 SET MARGIN 调整左页边距，则在系统变量 _PLOFFSET 中保存 SET MARGIN 指定的值。也可通过直接在 _PLOFFSET 中保存值来设置左页边距。

系统变量 _LMARGIN 的值对左页边距也有影响。

重要提示 SET MARGIN 指定的左页边距不影响由“报表设计器”创建、用 REPORT 运行的报表。虽然只是在运行由“报表设计器”创建的报表时调整 _PLOFFSET，在运行报表后仍将它重置为最初值。“报表设计器”的“页面设置”对话框中的左页边距设置决定了纸的左边的偏移量。打开“报表设计器”时，从“文件”菜单中选择“页面设置”命令，会显示“页面设置”对话框。

请参阅

_LMARGIN, _PLOFFSET

返回总目录

SET MARK OF 命令
SET MARK TO 命令
SET MEMOWIDTH 命令
SET MESSAGE 命令
SET MULTILOCKS 命令
SET NEAR 命令
SET NOCPTRANS 命令
SET NOTIFY 命令
SET NULL 命令
SET NULLDISPLAY 命令
SET ODOMETER 命令
SET OLEOBJECT 命令
SET OPTIMIZE 命令
SET ORDER 命令
SET PALETTE 命令
SET PATH 命令
SET PDSETUP 命令
SET POINT 命令
SET PRINTER 命令

SET PROCEDURE 命令
SET READBORDER 命令
SET REFRESH 命令
SET RELATION 命令
SET RELATION OFF 命令
SET REPROCESS 命令
SET RESOURCE 命令
SET SAFETY 命令
SET SECONDS 命令
SET SEPARATOR 命令
SET SKIP 命令
SET SKIP OF 命令
SET SPACE 命令
SET STATUS 命令
SET STATUS BAR 命令
SET STEP 命令
SET STRICTDATE 命令
SET SYSFORMATS 命令
SET SYSMENU 命令
SET TALK 命令
SET TEXTMERGE 命令

SET TEXTMERGE DELIMITERS 命令
SET TOPIC 命令
SET TOPIC ID 命令
SET TRBETWEEN 命令
SET TYPEAHEAD 命令
SET UDFPARMS 命令
SET UNIQUE 命令
SET VIEW 命令
SET WINDOW OF MEMO 命令
SET() 函数
SetAll 方法
SETFLDSTATE() 函数
SetFocus 方法
SetMain 方法

SET MARK OF 命令

为菜单标题或菜单项指定标记字符，也能够指定显示还是清除该标记字符。

语法

SET MARK OF MENU MenuBarName1
TO lExpression1

– 或 –

SET MARK OF POPUP MenuName1
TO lExpression3

– 或 –

SET MARK OF BAR nMenuItemNumber OF MenuName2
TO lExpression4

参数描述

MENU MenuBarName1

菜单栏名称，此命令为之指定、显示或者清除标记字符。

- TO lExpression1 显示或清除菜单栏中每个菜单标题的标记字符。如果逻辑表达式 lExpression1 的值为真 (.T.)，那么标记字符在每个菜单标题的旁边显示；如果逻辑表达式 lExpression1 的值为假 (.F.)，则从每个菜单标题的旁边清除标记字符。

POPUP MenuName1

指定菜单名字，为此菜单栏指定、显示或者清除标记字符。

- TO lExpression3 显示或清除所有菜单项的标记字符。如果逻辑表达式

lExpres 的值为真 (.T.)，则显示标记字符；如果逻辑表达式 lExpression3 的值为假 (.F.)，则清除标记字符。

BAR nMenuItemNumber OF MenuName2

指定菜单项编号（以及包含该菜单项的菜单名），为此菜单栏指定、显示或者清除标记字符。

- TO lExpression4 显示或清除菜单项的标记字符。如果逻辑表达式 lExpression4 的值为真 (.T.)，显示标记字符；如果逻辑表达式 lExpression4 的值为假 (.F.)，清除标记字符。

说明

不能指定菜单标题或者菜单项的标记字符，它们的标记字符只能是复选标记。不过，可以使用 SET MARK OF 来显示或清除菜单标题和菜单项的标记。使用 DEFINE POPUP PROMPT 子句 (FIELD, FILES, 或 STRUCTURE) 创建的菜单项都不能标记。

使用 MRKPAD( ) 可确定菜单标题是否有一个显示的标记字符；使用 MRKBAR( ) 可确定菜单项是否有显示的标记字符。

例如，使用 SET MARK OF，请参阅前面的“语言参考”中的 CNTBAR( ) 函数。

请参阅

CNTBAR( ), CNTPAD( ), DEFINE BAR, DEFINE MENU, DEFINE PAD, DEFINE POPUP, MRKBAR( ), MRKPAD( )

SET MARK TO 命令

指定显示日期表达式时所使用的分隔符。

语法

SET MARK TO [*cDelimiter*]

参数描述

cDelimiter

指定想使用的日期分隔字符。

说明

SET MARK TO **用来分隔显示日期中的年、月、日的字符。**

不带 *cDelimiter* **参数使用** SET MARK TO **时，可将分隔符重置为默认的正斜杠符**(/)**。**

SET MARK TO **的作用范围是当前数据工作期。**

请参阅

SET DATASESSION, SET DATE, SET SYSFORMATS

SET MEMOWIDTH 命令

指定备注字段和字符表达式的显示宽度。

语法

SET MEMOWIDTH TO *nColumns*

参数描述

nColumns

指定宽度值。范围在 8 到 8192 列之间，默认的输出宽度是 50 列。如果执行 SET COMPATIBLE ON 或者 SET COMPATIBLE DB4 命令，默认的宽度改为 80 列。如果指定的 *nColumns* 大于 8192，则宽度将设置为 8192。

说明

SET MEMOWIDTH 指定 ?|??、DISPLAY 或 LIST 等命令在 Visual FoxPro 主窗口或用户自定义窗口中的输出宽度。它影响长度大于 8192 个字符的备注字段和字符表达式的输出，还影响 ATCLINE()、ATLINE()、MEMLINE() 和 MLINE() 等函数的返回值。

注意 ? 和 ?? 的显示宽度不能超过 256 个字符。

在 Visual FoxPro 中，如果输出定向到 Visual FoxPro 主窗口，输出的宽度由 Visual FoxPro 主窗口的字体决定；如果输出定向到用户自定义窗口，输出的宽度由用户自定义窗口的字体决定。SET MEMOWIDTH 的作用范围是当前数据工作期。

请参阅

ATCLINE( ), ATLINE( ) , MEMLINES( ) , MLINE( ) , SET DATASESSION

SET MESSAGE 命令

定义在 Visual FoxPro 主窗口或图形状态栏中显示的信息，或者指定有关用户自定义菜单栏和菜单命令的信息位置。

语法

SET MESSAGE TO [cMessageText]

– **或** –

SET MESSAGE TO [nRow [LEFT | CENTER | RIGHT]]

– **或** –

SET MESSAGE WINDOW [WindowName]

参数描述

TO [cMessageText]

指定要显示的信息。

TO [nRow *[LEFT | CENTER | RIGHT]]*

指定信息在 Visual FoxPro 主窗口中的位置。*nRow* 表示信息所在的行。如

果 *nRow* 为 0，则不显示信息。

LEFT、CENTER 和 RIGHT 指定信息在屏幕水平方向上的位置。

对于 Visual FoxPro，当图形状态栏处于活动状态（已显示出来）时，SET MESSAGE 指定的信息位置将被忽略。

WINDOW [WindowName]

指定显示信息的窗口。要从窗口中除去信息，同时在屏幕上显示信息，可发出 SET MESSAGE WINDOW 命令。

说明

可用 SET MESSAGE 建立提示信息，也可用它为一些命令指定显示信息的位置，如 DEFINE BAR、DEFINE MENU、DEFINE PAD 以及 DEFINE POPUP。

在 Visual FoxPro 中，如果显示的状态栏是基于字符的，信息默认显示在 Visual FoxPro 主窗口的最后一行；如果显示的状态栏是图形状态栏，那么信息显示在该状态栏中。

一旦发出 SET DISPLAY 命令，SET MESSAGE 行将重置为 Visual FoxPro 主窗口的最后一行。在 Visual FoxPro 中，不带任何参数的 SET MESSAGE TO 命令把信息显示在图形状态栏中。

请参阅

SET DISPLAY, SET STATUS, SET STATUS BAR

SET MULTILOCKS 命令

决定能否使用 LOCK ()或 RLOCK ( ) 锁定多个记录。

语法

SET MULTILOCKS ON | OFF

参数描述

ON

允许尝试锁定一组记录。如果在 LOCK ( ) 或 RLOCK ( ) 中包含一组记录号，就可以来锁定多个记录。

OFF

（默认情况）允许尝试用 LOCK ( ) 或 RLOCK ( ) 来锁定单个记录。

说明

在网络上以共享方式打开一个表时，可以锁定该表中的多个记录，SET MULTILOCKS 的设置决定锁定单个记录还是多个记录。可以用函数 LOCK( ) 或 RLOCK( ) 锁定记录。

注意 SET MULTILOCKS的设置由 ON 切换到 OFF，或由 OFF 切换到 ON，都隐含执行了 UNLOCK ALL 命令，即对所有工作区内的所有记录解除锁定。

SET MULTILOCKS 的作用范围是当前数据工作期。

对于 Visual FoxPro，在使用 CURSORSETPROP( ) 启用行缓冲或表缓冲以前，MULTILOCKS 必须是 ON。有关行缓冲和表缓冲的详细内容，请参阅 CURSORSETPROP( )。

如果选定了“工作区属性”对话框中的“允许数据缓冲”复选框（选择查看窗口中的“属性”按钮时，显示“工作区属性”对话框），那么当前数据工作期的 MULTILOCKS 将自动设置为 ON。但是，如果清除“允许数据缓冲”复选框，却不会对当前数据工作期的 MULTILOCKS 设置为 OFF。

有关网络上记录和文件锁定、共享表的详细内容，请参阅前面的“语言参考”中的 LOCK( ) 和 RLOCK( ) 函数，与《Microsoft Visual FoxPro 6.0 中文版程序员指南》第十七章“共享访问程序设计”。

请参阅

CURSORSETPROP( ), LOCK( ) , RLOCK( ) , SET DATASESSION

SET NEAR 命令

FIND 或 SEEK 查找记录不成功时，确定记录指针停留的位置。

语法

SET NEAR ON | OFF

参数描述

ON

当使用 FIND 或 SEEK 查找记录不成功时，将记录指针放在最相近的记录上。使用此设置时，RECNO( ) 将返回最邻近的记录号，FOUND( ) 返回“假” (.F.)，EOF( ) 也返回“假” (.F.)。

OFF

（默认值）当使用 FIND 或 SEEK 查找记录不成功时，将记录指针放在表的尾部。使用此设置时，RECNO( ) 将表中记录的数目加 1 再返回，FOUND( ) 返回“假” (.F.)，EOF( )返回“真” (.T.)。

说明

当没有记录满足查找条件时，查找不成功。

如果查找不成功，则不论 SET NEAR 的设置如何，使用 0 作为参数的 RECNO( ) 命令都将返回最邻近记录的记录号。

SET NEAR 的作用范围为当前数据工作期。

请参阅

EOF( ), FIND, FOUND( ) , RECNO( ) , SEEK, SET DATASESSION

SET NOCPTRANS 命令

防止把已打开表中的选定字段转换到另一个代码页。

语法

SET NOCPTRANS TO [*FieldName1* [, *FieldName2* ...]]

参数描述

TO [FieldName1 [, FieldName2 ...]]

指定不能转换到其他代码页的字段。

如果发出 SET NOCPTRANS TO 命令时未指定字段名系列，就将把表中所有字符字段和备注字段返回到默认的代码页转化方式（由 CODEPAGE 配置项建立）。使用 SET("NOCPTRANS") 可以返回最近一次发出的 SET NOCPTRANS 命令所指定的字段。使用 CHR() 函数可确保单个字符不被转化。

说明

可以设置 Visual FoxPro，使其将字符字段和备注字段自动转化到其他代码页，因此可以使用 SET NOCPTRANS 来防止自动转化包含二进制数据的字段。例如，一个备注字段可能包含 Microsoft Word 文档。当您访问这个文档时，希望文档以它原来的、没有转化的形式存在，这时可使用 SET NOCPTRANS 指定不能转化的备注字段。

如果字符或备注字段中包含的二进制数据没有转化，就没有必要在访问这些数据之前

使用 SET NOCPTRANS 命令。在 Visual FoxPro 配置文件中，省略 CODEPAGE 配置项可以确保字符字段和备注字段不被转化。

有关代码页和 Visual FoxPro 的国际支持的详细内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》的第十八章“开发国际化应用程序”中的“Visual FoxPro 支持的代码页”。

请参阅

CPCONVERT(), CPCURRENT(), CPDBF(), MODIFY COMMAND, MODIFY FILE

SET NOTIFY 命令

确定是否显示某种系统信息。

语法

SET NOTIFY ON | OFF

参数描述

ON

（默认值）启用某些系统信息的显示。

OFF
禁止某些系统信息的显示。

说明

SET NOTIFY 影响的系统信息例子有：

- “表达式生成器”对话框中的“表达式有效”。
- 当取消程序运行时出现的“执行已取消”。

在 Visual FoxPro 中，系统信息显示在 Visual FoxPro 主窗口底边的图形（不基于字符）状态栏中。

请参阅

SET MESSAGE, WAIT

SET NULL 命令

确定 ATLTER TABLE、CREATE TABLE 和 INSERT-SQL 命令如何处理 null 值。

语法

SET NULL ON | OFF

参数描述

ON
指定由 ALTER TABLE 和 CREATE TABLE 所创建表的所有列都允许 null 值。在列定义中加入 NOT NULL 子句使该列不接受 null 值。
另外，参数 ON 还指定 INSERT - SQL 语句把 null 值加入到 INSERT - SQL VALUE 子句中没有包括的列中。INSERT - SQL 只将 null 值插入到允许 null 值的列中。
注意 如果增加支持 null.值的栏的数目，则该表的栏目限制将缩减到 254-255 之间。
OFF
（默认值）指定由 ALTER TABLE 和 CREATE TABLE 所创建表的所有列都不允许 null 值。在列定义时可以在 ALTER TABLE 和 CREATE TABLE 中加入 NULL 子句使列接受 null 值。
另外，参数 OFF 还指定 INSERT - SQL 语句把空值加入到 INSERT - SQL VALUE 子句没有包括的列中。

说明

SET NULL 只影响 ALTER TABLE、CREATE TABLE 和 INSERT - SQL 语句是否支持 null 值。其他命令不受 SET NULL 的影响。
SET NULL 的作用范围是当前数据工作期。

示例

下面的示例说明了 SET NULL 如何影响 null 值。第一个表 employee 在 SET NULL ON 时创建，因此该表的各个字段支持 null 值。用 REPLACE 在 cLastName 字段中

加入 null 值。第二个表 stuff 在 SET NULL OFF 时创建，因此它的各字段不支持 null 值。用 REPLACE 在 cLastName 字段中加入 0 值。

```
CLOSE DATABASES
SET NULL ON  && 字段支持 null 值
CREATE TABLE employee (cLastName C(20), ySalary Y(12,2))
APPEND BLANK  && 添加一个新的空记录
REPLACE cLastName WITH .NULL.  && cLastName 支持 null 值
SET NULL OFF  && 字段不支持 null 值
CREATE TABLE stuff (cLastName C(20), ySalary Y(12,2))
APPEND BLANK  && 添加一个新的空记录
REPLACE cLastName WITH 0  && 不支持 null 值
```

请参阅

ALTER TABLE, CREATE TABLE, INSERT – SQL, ISNULL(), NVL() , SETDATASESSION

SET NULLDISPLAY 命令

指定 null 值显示时对应的字符串。

语法

SET NULLDISPLAY TO [*cNullText*]

参数描述

cNullText

指定 null 值显示时对应的字符串。如果省略 *cNullText*，".NULL." 将用来代表 null 值。

说明

默认时，Visual FoxPro 使用 .NULL. 表示 null。SET NULLDISPLAY 命令允许您指定一串文字代替 .NULL.。一个对象（如一个控件），如果它的 NullDisplay 属性为空，则在该对象中出现的 null 值使用 SET NULLDISPLAY 命令指定的字符串表示。如果对象的 NullDisplay 属性指定了一个字符串，则该对象中的 null 值由这个字符串表示。

请参阅

CREATE TABLE - SQL, NullDisplay 属性

SET ODOMETER 命令

设置命令状态的报告间隔。

语法

SET ODOMETER TO [*nRecords*]

参数描述

TO [nRecords]

以记录数为单位指定汇报间隔。*nRecords* 值的范围为 1 到 32767 个记录。*nRecords* 的默认值为 100 个记录。

说明

Visual FoxPro 可以每隔一段时间显示已处理记录数的信息，SET ODOMETER 用来改变这个汇报信息的间隔。

例如，在执行 COPY TO 命令时会显示拷入到新文件中的记录数。可以通过发出 SET TALK OFF 命令来关闭记录记数器。

请参阅

SET TALK, _TALLY, OBJECT

SET OLEOBJECT 命令

Visual FoxPro 找不到对象时，指定是否在 OLE Registry 中查找。

语法

SET OLEOBJECT ON | OFF

参数描述

ON
（默认值） 指定 Visual FoxPro 找不到对象时，在 OLE Registry 中查找。
OFF
指定 Visual FoxPro 找不到对象时，不在 OLE Registry 中查找。

说明

当对象是由 CREATEOBJECT() 命令或 GETOBJECT() 命令创建时，Visual FoxPro 在下列地点以下面的顺序查找对象：

1. Visual FoxPro 基类。
2. 在内存中的类定义，按其装入顺序查找。
3. 当前程序中的类定义。
4. 用 SET CLASSLIB 打开的 .VCX 类库中的类定义。
5. 用 SET PROCEDURE 打开的过程文件中的类定义。
6. Visual FoxPro 程序执行链中的类定义。
7. The OLE Registry。

Visual FoxPro 最后在 OLE Registry 中查找对象。在对 OLE Registry 进行查找之前需要装入 OLE 支持模块。这样，Visual FoxPro 所需的内存数量将增加，其他应用程序可得的内存数量相应减少。
如果开发的应用程序不需要 OLE 支持，可发出 SET OLEOBJECT OFF 命令来阻止

Visual FoxPro 在找不到对象时搜索 OLE Registry。

SET OLEOBJECT 不影响表单和通用字段中的 OLE 对象。当打开包含 OLE 对象的表单进行修改或运行该表单时，或者打开一个带有通用字段的表时，Visual FoxPro 总是装入 OLE 支持模块。

由于 GETOBJECT() 命令会激活 OLE OBJECT，所以如果在 SET OLEOBJECT 设置为 OFF 时发出 GETOBJECT() 命令，Visual FoxPro 会产生错误信息。

请参阅

CREATEOBJECT(), GETOBJECT()

SET OPTIMIZE 命令

设置是否使用 Rushmore 优化技术。

语法

SET OPTIMIZE ON | OFF

参数描述

ON

（默认值）启用 Rushmore 优化。

OFF

禁止 Rushmore 优化。

说明

Visual FoxPro 使用一种叫 Rushmore 的技术来优化数据的检索，支持 FOR 子句的表命令可以使用 Rushmore 技术提高性能。当发出一条可优化命令时，Rushmore 决定哪些记录与 FOR 条件相匹配。发出的命令是在表中与 Rushmore 记录集相匹配的记录上执行的。

很少有情况需要禁止 Rushmore 优化。如果一条可从 Rushmore 优化中受益的命令修改了查询索引关键字时，Rushmore 记录集可能变得过时。这时，为确保从表中得到最新的信息，应禁止 Rushmore 优化。

可用 SET OPTIMIZE 在全局范围内启用或禁止 Rushmore 技术。每条使用 Rushmore 的命令都有一个 NOOPTIMIZE 子句，可以用它来禁止 Rushmore 优化开关。

下面的命令可用 Rushmore 来优化其运行性能：

命令		
AVERAGE	BLANK	BROWSE
CALCULATE	CHANGE	COPY TO
COPY TO ARRAY	COUNT	DELETE
DISPLAY	EDIT	EXPORT
INDEX	LABEL	LIST
LOCATE	RECALL	REPLACE

REPLACE FROM ARRAY　　REPORT　　SCAN

SORT　　SUM　　TOTAL

请参阅

INDEX, SET ORDER

SET ORDER 命令

指定表的主控索引文件或标识。

语法

SET ORDER TO
[nIndexNumber | IDXIndexFileName | [TAG] TagName [OF CDXFileName]
[IN nWorkArea | cTableAlias]
[ASCENDING | DESCENDING]]

参数描述

nIndexNumber

指定主控索引文件或标识的编号。 *nIndexNumber* **指的是在** USE **或** SET INDEX **中列出索引文件的顺序号。首先按它们在** USE **或** SET INDEX **中**

出现的顺序为打开的 .IDX 文件编号；随后，按创建顺序为结构 .CDX 文件中的索引标识（如果存在的话）编号；最后，按创建顺序为所有打开的独立的 .CDX 文件中的索引标识编号。

下面示例说明了不同索引文件类型和标识的编号方式（文件名只是为了说明问题，并不需要真正存在）。表 video.dbf 打开时带有三个索引 (title.idx, costs.cdx 和 rating.idx)，使用下面命令将其在第一个工作区中打开：

```
USE video INDEX title.idx, costs.cdx, rating.idx IN 1
```

video 表有一个结构复合索引文件(video.cdx)，其中有两个索引标识(NUMBERSOLD 和 YEARSOLD)。打开 video 时，此结构 .CDX 文件自动打开。

由于 .IDX 文件首先被编号，因此发出命令 SET ORDER TO 1 将使 title.idx 成为主控索引，而发出 SET ORDER TO 2 则使 rating.idx 成为主控索引。

```
SET ORDER TO 1
主索引文件: C:\FOX30\TITLE.IDX
SET ORDER TO 2
主索引文件: C:\FOX30\RATING.IDX
```

其次对 video.cdx 中的索引标识进行编号：

```
SET ORDER TO 3
主索引文件: C:\FOX30\VIDEO.CDX  标识: NUMBERSOLD
SET ORDER TO 4
主索引文件: C:\FOX30\VIDEO.CDX  标识: YEARSOLD
```

独立文件 costs.cdx 中的索引标识最后被编号：

```
SET ORDER TO 5
主索引文件: C:\FOX30\COSTS.CDX  标识: RENTALCOST
SET ORDER TO 6
主索引文件: C:\FOX30\COSTS.CDX  标识: BUYCOST
```

nIndexNumber 也可以是 0。如果发出 SET ORDER TO 0 命令，则所有索引文件仍保持打开，并且在增加、删除或修改记录时更新。但是，表中所有记录的显示和访问顺序是记录编号顺序而不是索引顺序。不带其他参数的 SET ORDER TO 命令与 SET ORDER TO 0 命令完全一样。

如果 *nIndexNumber* 大于 .IDX 文件和 .CDX 文件的索引标识数，则 Visual FoxPro 产生错误信息。

IDXIndexFileName

指定作为主控索引文件的 .IDX 文件。

[TAG] TagName [OF CDXFileName]

指定 .CDX 文件中的一个标识作为主控索引标识。标识名可以来自结构 .CDX 文件或任何打开的独立 .CDX 文件。

如果在各打开的独立 .CDX 文件中存在相同的标识名，应使用 OF *CDXFileName* 来指定包含此标识的 .CDX 文件。

如果 .IDX 文件和标识名重复，.IDX 文件优先。

IN nWorkArea | cTableAlias

为在非当前选定工作区中打开的表指定主控索引文件或索引标识。*nWorkArea* 指定工作区编号，*cTableAlias* 指定表的别名。

ASCENDING | DESCENDING

以升序或降序显示或访问表记录。使用 ASCENDING 或 DESCENDING 不会改变索引文件或索引标识。

说明

一个表可以同时打开多个索引文件。但是，只有一个单索引 (.IDX) 文件（主控索引文件）或一个来自复合索引 (.CDX) 文件的索引标识（主控标识）决定表中记录的显示和访问顺序。可用 SET ORDER 来指定主控索引文件或标识。有些命令（例如 SEEK）使用主控索引或标识来查找记录。

可以在 USE 命令中包含 INDEX 子句随表打开索引文件。如果一个表有相关的结构 .CDX 文件，这个文件会随着表的打开而自动打开。在一个表被打开以后，可以使用 SET INDEX 命令为这个表打开或关闭索引文件。

默认情况下，SET ORDER 为当前工作区中打开的表指定主控索引或主控标识。

请参阅

INDEX, ORDER(), SET INDEX

SET PALETTE 命令

指定是否使用默认的调色板。

语法

SET PALETTE ON | OFF

参数描述

ON

（默认值）恢复默认的 Visual FoxPro 调色板。

OFF

用来自 .BMP 图形和 OLE 对象的调色板替代 Visual FoxPro 的默认调色板。

说明

在 Visual FoxPro 和 FoxPro for Windows 中，可以显示和操作位图 (.BMP)图形和 OLE 对象。.BMP 图形和 OLE 对象可以包含调色板，决定图形和对象的显示方式。第一个图形或对象的调色板将用于所有后续的图形或对象。由于所有图形和对象都使用单一的调色板，因此一些图形和对象可能发生无法预料的变化。

默认的 Visual FoxPro 调色板能够确保多个 .BMP 图形和 OLE 对象正确地显示。

请参阅

SET PATH 命令

指定查找文件的路径。

语法

SET PATH TO [*Path*]

参数描述

TO [Path*]*

用于指定查找文件的目录。用逗号或分号隔开不同的目录。

注意　如果驱动器或目录名包含叹号 (!)，Visual FoxPro 不能正确识别该路径名。

在所有 FoxPro 平台上，所有返回路径信息的函数，如 CURDIR( )、DBF( ) 和 SYS(2003) 等在其返回值中都使用 MS-DOS 路径命令习惯。

说明

不带参数 *Path* 的 SET PATH TO 命令把路径恢复为默认目录。SET DEFAULT 可以指定默认目录，CURDIR( ) 函数可以返回当前默认的目录。

SET PATH 的作用范围不是当前数据工作期。使用 SET PATH 对默认目录的更改将影响所有的数据工作期。

请参阅

CD | CHDIR, GETFILE( ), LOCFILE( ) , MD | MKDIR, RD | RMDIR, SET DEFAULT, SET DATASESSION

SET PDSETUP 命令

装入一个打印机驱动程序。

语法

```
SET PDSETUP TO [[cPrinterDriverSetup [, Parameter1[, Parameter2 ...]]]
    [WITH Parameter3 [, Parameter4 ...]]]
```

参数描述

cPrinterDriverSetup

指定要装入的打印驱动程序名称。

当装入打印机驱动程序设置时，设置的名称保存在 _PDSETUP 系统变量中，同时可以创建相应的特殊变量数组 _PDPARMS。

如果 *cPrinterDriverSetup* 指定的打印机驱动程序设置名称在资源文件中不存在，就执行当前打印机驱动程序的设置应用程序以便能创建具有该名称的设置。如果当前打印

机驱动程序的设置应用程序是 GENPD.APP，那么将显示“打印设置编辑”对话框，从中可以创建打印设置。

如果设置名称以短划线 (-) 开头，那么不会执行 _GENPD 程序，但把短划线之后的名称存入 _PDSETUP 中。

如果不带 *cPrinterDriverSetup* 参数发出 SET PDSETUP TO 命令，将清除当前打印机设置，并将空字符串保存在 _PDSETUP 中，同时还从内存中清除 _PDPARMS 数组。

在配置文件 (CONFIG.FP) 中加入如下一行，可以在启动 FoxPro for MS-DOS 时装入默认的打印机驱动程序设置：

PDSETUP = '*SetupName*' WITH *ParameterList*

SetupName 是启动 FoxPro for MS-DOS 时要装入的设置名称。确保用引号把设置名称括起来。配置文件 CONFIG.FP 中指定的默认打印机驱动程序设置优先于“打印机驱动程序设置”对话框中指定的默认打印机驱动程序设置。

Parameter1 *[*, Parameter2 ...*]*

指定任意数目的可选参数。这些参数传给打印机设置接口应用程序，可为任意类型（字符型、数值型、逻辑型等等）。打印机设置接口应用程序中的第一行必须是 LPARAMETERS 或 PARAMETERS 语句，以接受 SET PDSETUP 传递过来的参数。

如果使用 GENPD.APP，不要包括这些可选参数。GENPD.APP 不接受由 SET PDSETUP 传递来的参数，所以包括这些参数将产生错误。

WITH Parameter3 [, Parameter4 ...]

创建特殊的 _PDPARMS 打印机数组。由 *Parameter3*、*Parameter4* 等指定的每一

个参数都成为 _PDPARMS 中的一个元素。第一个参数(*Parameter3*) 保存在 _PDPARMS 的第一个元素中，第二个参数(*Parameter4*)保存在 _PDPARMS 的第二个元素中，等等。这些参数的类型可为任意类型（字符型、数值型、逻辑型等等）。

如果使用 GENPD.APP，这个应用程序将改写您指定的所有参数。

说明

对于 Visual FoxPro，在打印由 FoxPro for MS-DOS 创建的基于字符的报表时，需要用到打印机驱动程序设置。

打印机驱动程序设置由许多设置组合而成。它包括打印机驱动程序和诸如页面方向、默认字体大小和字体类型、页边距等信息。打印机驱动程序设置保存在 FoxPro for MS-DOS 的资源文件 FOXUSER.DBF 中，可以在“打印设置编辑”对话框中以交互方式创建打印机驱动程序设置并给它指定名称。

也可以使用 _PDSETUP 系统变量装入或清除打印机驱动程序设置。

发出 SET PDSETUP 命令时，执行当前的打印机设置接口应用程序。SET PDSETUP 中包含的打印机驱动程序设置将被传递给接口应用程序。接口应用程序也可以由 _GENPD 系统变量指定。默认的接口应用程序是 GENPD.APP，即包括在 FoxPro for MS-DOS 中的打印机设置接口应用程序。

请参阅

_GENPD, _PDRIVER, _PDSETUP, SET PRINTER

SET POINT 命令

显示数值表达式或货币表达式时，确定所用小数点字符。

语法

SET POINT TO [cDecimalPointCharacter]

参数描述

cDecimalPointCharacter

指定小数点字符。

说明

SET POINT 可用来改变默认的小数点（句点 .）。不带 *cDecimalPointCharater* 参数的 SET POINT TO 命令恢复小数点为句点符。您可以把显示的小数点设置为不同的字符，但在计算中必须使用句点作为小数点。

SET POINT 的作用范围为当前数据工作期。

示例

```
gnX = 1.25
gcNewPoint = '_'
SET POINT TO gcNewPoint
? gnX
SET POINT TO        && 把小数点设为句号
```

? gnX

请参阅

SET DATASESSION, SET DECIMALS, SET FIXED, SET SEPARATOR, SETSYSFORMATS

SET PRINTER 命令

打开或者关闭向打印机的输出，或将结果输出到一个文件、端口。

语法

SET PRINTER ON [PROMPT] | OFF

– **或** –

SET PRINTER FONT cFontName [, cFontSize]
[STYLE cFontStyle]

– **或** –

SET PRINTER TO [FileName [ADDITIVE] | PortName]

– **或** –

SET PRINTER TO [DEFAULT | NAME WindowsPrinterName]

–**或**–

SET PRINTER TO NAME \\ServerName\PrinterName

参数描述

ON [PROMPT]

允许输出到打印机。当 SET PRINTER 设置为 ON 时，@ ... SAY 的格式化输出结果并不立即发送到打印机。要使 @ ... SAY 直接输出到打印机，应使用 SET DEVICE TO PRINTER。

可以包含 PROMPT 以便在打印前显示一个对话框。在这个对话框中可以调整打印机设置。当前已安装的打印机驱动程序决定可以调节哪些打印机设置。

OFF

（默认值）不允许输出结果发送到打印机。

FONT cFontName [, cFontSize]

指定打印机输出的默认字体。*cFontName* 用以指定字体的名称，*cFontSize* 指定磅值大小。

如果指定的字体不可用，FoxPro 将用具有相似字体特征的字体代替。

STYLE cFontStyle

指定打印机输出的默认字形。如果省略 STYLE 子句，将使用常规字形。

如果指定的字形不可用，FoxPro 将用具有相似字体特征的字形代替。

可由 *cFontStyle* 指定的字形如下：

字符	字形
B	粗体
I	斜体
N	常规体
O	轮廓
Q	不透明
S	阴影
–	删除线
T	透明
U	下划线

可以在命令中包括多个字符指定组合字形。例如，下面的命令指定了 16 磅的 Courier 粗体加斜体：

```
SET PRINTER FONT 'Courier', 16 STYLE 'BI'
```

TO [FileNam*e* [ADDITIVE] | PortName]

指定定向输出到的文件或端口。

在 Visual FoxPro 中，可使用 SET PRINTER TO NAME 来指定打印机。

FileName 指定接受输出结果的文件名称。如果包括 ADDITIVE，那么输出

结果将追加到文件已有内容的后面。如果省略 ADDITIVE，将改写文件已有的内容。

PortName 把输出结果发送给本地另一台打印机。

TO [DEFAULT | NAME WindowsPrinterName]

把打印机输出传送到默认的 Windows 打印机或指定的 Windows 打印机。Windows 打印机的名称保存在 WIN.INI 中。

可用 GETPRINTER( ) 或 APRINTERS( ) 确定当前已安装的打印机名称。如下命令将显示 Windows 打印机对话框并且直接把打印输出传送到指定的打印机上:

```
SET PRINTER TO NAME GETPRINTER( )
```

TO [NAME \\ServerName\PrinterName]

仅用于 Windows NT。将脱机打印输出到网络打印机。

ServerName 是打印服务器的网络名称。这一名称由网络管理员指定，并且必须是唯一的。

PrinterName 指定打印机的名称。这一名称也是由网络管理员指定的。

说明

可使用 SET PRINTER TO 并指定不同参数将输出分别定向到文件、本地的不同打印机端口和网络打印机。

不带参数的 SET PRINTER TO 命令重置输出为默认的 MS-DOS PRN 打印设备。

当定向输出到网络打印机时，输出结果将打印或收集在打印池中，直到发出一个新的 SET PRINTER 命令为止。有关在网络中打印的详细内容，请参阅您的网络文档。

请参阅

APRINTERS( ), GETPRINTER( ) , PRINTSTATUS( ) , SET DEVICE

SET PROCEDURE 命令

打开过程文件。

语法

SET PROCEDURE TO [FileName1 [, FileName2, ...]]
[ADDITIVE]

参数描述

FileName1 [, FileName2, ...]

指定打开文件的顺序。SET PROCEDURE **可带有多个文件名，即可以立刻打开多个过程文件。也就是说允许您创建多个独立的函数库并分别打开它们。**

ADDITIVE

在不关闭当前已打开的过程文件的情况下，打开其他过程文件。

说明

不带任何文件名的 SET PROCEDURE TO **的命令关闭所有打开的过程文件。**RELEASE PROCEDURE **可用来关闭单个文件。**

当执行一个过程时，如果在当前执行程序中找不到该过程，就将查找过程文件。有关过程文件的详细内容，请参阅 PROCEDURE。

请参阅

PROCEDURE, RELEASE PROCEDURE

SET READBORDER 命令

打开过程文件。

语法

SET READBORDER ON | OFF

参数描述

ON

将 @...GET 创建的所有文本框的边框设置为单线。如果在创建第一个文本框时，SET READBORDER 为 ON，则随后在同级的 READ 中创建的文本框都有边框。

OFF

（默认）指定由 @...GET 创建的文本框没有边框。如果在创建第一个文本框时，SET READBORDER 为 OFF，则随后在同级的 READ 中创建的文本

框都没有边框。

说明

SET READBORDER 命令指定由 @...GET 指定的文本框的边框是否设置为单线。

示例

在下面的示例中，@ ... GET 创建的前三个文本框有边框。虽然在创建第三个文本框之前 SET READBORDER 设置为 OFF，但是它仍有边框。第四个文本框没有边框。这是因为 READBORDER 已设置为 OFF，并且它在与前三个文本框不同的 READ 命令范围中。

```
SET READBORDER ON
@ 2,2 GET gnW DEFAULT 1 && 1st READ
@ 4,2 GET gnX DEFAULT 1 && 1st READ
SET READBORDER OFF
@ 6,2 GET gnY DEFAULT 1 && 1st READ
READ
@ 8,2 GET gnZ DEFAULT 2 && 2nd READ
READ
```

请参阅

READ, READ 事件

SET REFRESH 命令

当网络上的其他用户修改记录时，确定是否更新浏览窗口。

语法

SET REFRESH TO *nSeconds1* [, *nSeconds2*]

参数描述

TO nSeconds1 [, nSeconds2]

指定是否更新或更新的频度。*nSeconds1* 指定两次更新浏览窗口或备注编辑窗口之间的时间间隔，以秒为单位。*nSeconds1* 的值可以从 0 到 3600。默认值为 0 秒。当 *nSeconds1* 为非零值且其他用户改变了您正在浏览的记录时，这些记录每隔指定的刷新时间间隔便更新一次。如果 *nSeconds1* 为零，则不更新正浏览的记录。

Visual FoxPro 在本地工作站上的内存中缓冲部分表。*nSeconds2* 指定使用网络上当前数据更新这些本地数据缓冲区的频度。*nSeconds2* 是两次更新数据缓冲区的时间间隔，以秒为单位。*nSeconds2* 值可以从 0 到 3600。它的默认值为 5。如果 *nSeconds2* 设置为 0，缓冲区永不刷新。

如果为 *nSeconds1* 指定了一个非零值但是没有指定 *nSeconds2*，那么 *nSeconds2* 设置为与 *nSeconds1* 一样的值。但是如果 *nSeconds1* 设置为 0 且

不指定 *nSeconds2*，那么 *nSeconds2* 设置为 5。增加 *nSeconds2* 的值可以提高程序的运行性能。

说明

在网络上能够以共享方式打开表。因此正在浏览窗口中查看的记录很可能会被网络上的其他用户编辑。SET REFRESH 影响由 BROWSE、CHANGE 或 EDIT 命令打开的浏览窗口中的记录。在浏览窗口中为进行编辑而打开的备注字段也将被更新。SET REFRESH 还可以指定本地工作站上缓冲数据的更新频度。

请参阅

BROWSE, CHANGE, EDIT, MODIFY MEMO

SET RELATION 命令

在两个打开的表之间建立关系。

语法

```
SET RELATION TO
    [eExpression1 INTO nWorkArea1 | cTableAlias1
    [, eExpression2 INTO nWorkArea2 | cTableAlias2 ...]
    [IN nWorkArea | cTableAlias]
    [ADDITIVE]]
```

参数描述

eExpression1

指定用来在父表和子表之间建立关系的关系表达式。关系表达式通常是子表主控索引的索引表达式。

子表的索引可以是单项索引 (.IDX)、多项结构复合索引 (.CDX) 或独立复合索引。如果是复合索引，应指定适当的索引标识来排序子表。SET ORDER 可用来指定排序子表的索引标识。

例如，考虑“说明”部分描述的 customer 和 orders 两表。假定子表 orders 已经建立索引，并且使用如下命令用顾客编号进行排序：

```
SET ORDER TO TAG cust_id
```

要在 customer 和 orders 两表的顾客编号上建立关系，应选择包含父表 customer 的工作区或使用 IN 子句来指定父表的工作区或别名。然后，发出 SET RELATION 命令，用如下关系表达式指定索引表达式：

```
SET RELATION TO cust_id INTO orders
```

除非关系表达式是数值型的，否则子表必须建立索引。如果发出包含非数值关系表达式的 SET RELATION 命令时，子表没有使用索引进行排序，Visual FoxPro 显示错误信息。如果 *eExpression1* 是数值型表达式，那么当记录指针在父表中移动时，将计算此表达式。然后，子表的记录指针移到记录编号为 *eExpression1* 的记录上。

INTO nWorkArea1 | cTableAlias1

指定子表的工作区编号 (*nWorkArea1*) 或子表的表别名 (*cTableAlias1*)。

eExpression2 INTO nWorkArea2 | cTableAlias2 ...

指定其他关系表达式 (*eExpression2*) 和子表，建立另一个父表和子表之间的关系。在一条 SET RELATION 命令中可以创建单个父表与多个子表之间的关系，但是各个关系之间要用逗号隔开。

nWorkArea2 指定工作区编号，*cTableAlias2* 指定子表的别名。

IN nWorkArea

指定父表的工作区。

IN cTableAlias

指定父表的别名。IN 子句允许创建关系时不必首先选择父表工作区。如果 SET RELATION 命令中省略了 *nWorkArea* 和 *cTableAlias*，父表必须在当前选定的工作区中打开。

ADDITIVE

保留当前工作区中所有已存在的关系并创建指定的新关系。如果命令中不包括 ADDITIVE 子句，将断开当前工作区中的所有关系，然后再创建指定的关系。

在两个打开的表之间建立关系。

说明

在建立关系之前，必须打开一个表（父表），而且还必须在另一个工作区内打开其他表（子表）。相关的各表通常有一个相同的字段。例如，假定表 customer 包含顾客信息，有顾客姓名、地址和唯一的顾客编号等字段。表 orders 包含

订货信息，也有顾客编号字段，同时还有日期和发货信息等字段。SET RELATION 通过相同的字段，即顾客编号字段来联系两个表。要建立这种关系，子表必须在公共字段上建立索引。建立关系后，不论何时在父表 customer 中移动记录指针到指定顾客编号的记录上时，子表 orders 中的记录指针也将同时移到相同顾客编号的记录上。如果子表中找不到相匹配的记录，子表中的记录指针将停在表尾。不带任何参数的 SET RELATION TO 命令将移去当前选定工作区中的所有关系。可用 SET RELATION OFF 移去指定的父子关系。

请参阅

INDEX, RELATION( ), SET ORDER, SET RELATION OFF, SET SKIP, TARGET( )

SET RELATION OFF 命令

解除当前选定工作区中父表与相关子表之间已建立的关系。

语法

SET RELATION OFF INTO *nWorkArea* | *cTableAlias*

参数描述

INTO nWorkArea | cTableAlias

指定子表工作区编号或表别名。

请参阅

RELATION(), SELECT, SET RELATION, SET SKIP, TARGET()

SET REPROCESS 命令

指定一次锁定尝试不成功后，Visual FoxPro 对文件或记录再次尝试加锁的次数或时间。

语法

SET REPROCESS TO *nAttempts* [SECONDS] | TO AUTOMATIC

参数描述

TO nAttempts [SECONDS]

指定初始加锁不成功后 Visual FoxPro 对文件或记录再次尝试加锁的次数。默认值为 0，最大值为 32,000。

SECONDS **指定 Visual FoxPro 对文件或记录尝试加锁的时间为** *nAttempts*

秒。只有当 *nAttempts* 大于 0 时此设置才可用。
例如，如果 *nAttempts* 为 30，Visual FoxPro 对记录或文件的尝试加锁次数为 30 次。如果还包括 SECONDS (SET REPROCESS TO 30 SECONDS)，则 Visual FoxPro 持续地对文件或记录尝试加锁的时间是 30 秒。
如果 SET STATUS 设置为 ON，则出现系统信息（“等待锁定 ...”）。
如果 ON ERROR 例程有效，并且一条命令对文件或记录加锁的尝试不成功，那么将执行 ON ERROR 例程。但如果是一个函数尝试加锁，那么将不执行 ON ERROR 例程且该函数返回“假”(.F.)。
如果 ON ERROR 无效，且命令试图对文件或记录加锁，但未成功时，将出现相应的警告信息（例如，“其他用户正在使用该记录”）。如果函数的加锁尝试不成功，则不显示警告信息，而是由函数返回“假”(.F.)。
如果在 *nAttempts* 为 0（默认值）时，发出命令或使用函数来尝试锁定记录或文件，那么 Visual FoxPro 将无限制地尝试锁定记录或文件。在尝试锁定记录或文件时，Visual FoxPro 显示系统信息“尝试锁定 ...按 ESC 键取消”。等待过程中，如果记录或文件可以加锁时，则锁定该记录或文件并清除显示的系统信息。一个函数尝试加锁成功时，该函数返回“真”(.T.)。
如果按 ESC 键响应系统信息，则出现相应的警告信息（例如，“记录正被其他用户使用”）。如果是一个函数尝试加锁，则不出现此类信息，而由函数返回“假”(.F.)。
如果 ON ERROR 例程有效且一条命令正在对文件或记录尝试加锁，则 ON ERROR 优先于对记录或文件再次的加锁尝试。这种情况下，ON ERROR 例

程立即执行。Visual FoxPro 不再尝试对文件或记录进行加锁，也不显示系统信息。

如果 *nAttempts* 为 –1，Visual FoxPro 将无限制地尝试锁定记录或文件。这时即使按 ESC 键也不能取消加锁尝试，并且不执行 ON ERROR 例程。

只有当 SET STATUS 设置为 ON 时，Visual FoxPro 才显示“等待锁定 ...”。

如果尝试锁定的记录或文件已被其他用户加锁，那么只有等到该用户解锁后才能加锁。

TO *AUTOMATIC*

指定 Visual FoxPro 无限制地尝试锁定记录或文件（Setting n*Attempts* to –2 与此命令等价。）。在没有退出尝试锁定记录或文件时，该语句与 nAttempts to –1 相近。

Visual FoxPro 在尝试锁定记录或文件时显示系统信息“尝试锁定 ...按 ESC 键取消”。如果要加锁的记录或文件可用，就加上锁并清除显示的系统信息。

如果使用函数来加锁，函数返回“真”(.T.)。

如果 ON ERROR 例程无效且按 ESC 键来响应系统信息，则出现相应的警告信息（例如，“其他用户正在使用该记录”）。这种情况下如果函数尝试加锁不成功，将不显示警告信息，而由函数返回“假”(.F.)。

如果 ON ERROR 例程有效且按下了 ESC 键，则执行 ON ERROR 例程。

如果是函数尝试加锁，则不执行 ON ERROR 例程，而且函数返回“假”(.F.)。

有关锁定文件或记录以及在网络中共享表的其它信息，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》的第十七章的“共享数据库程序”。

说明

第一次对记录或文件尝试加锁不可能总成功。在网络上，记录或文件经常被其他用户锁定。SET REPROCESS 确定初始加锁尝试不成功时 Visual FoxPro 是否继续尝试锁定记录或文件。既可指定继续尝试加锁的次数，也可指定继续尝试加锁的时间限制。ON ERROR 例程影响如何处理不成功的加锁尝试。

SET REPROCESS 的作用范围是当前数据工作期。

请参阅

FLOCK(), SET DATASESSION

SET RESOURCE 命令

指定或更新一个资源文件。

语法

SET RESOURCE ON | OFF

— 或者 —

SET RESOURCE TO [*FileName*]

参数描述

ON
（默认值）指定对 Visual FoxPro 环境所作的更改保存在资源文件中。
OFF
指定对 Visual FoxPro 环境所作的变化不保存在资源文件中。
TO [FileName]
指定对 Visual FoxPro 环境所作的变化保存在资源文件 *FileName* 中，而不是保存在默认的 FOXUSER.DBF 资源文件中。
不带资源文件名的 SET RESOURCE TO 命令打开默认的 FOXUSER.DBF 资源文件。发出 SET RESOURCE TO 命令执行隐含的 SET RESOURCE ON 命令。

说明

资源文件是一个 Visual FoxPro 表，它包含一些设置，诸如键盘宏、预选参数、系统窗口的位置和大小、日记入口等系统和用户自定义资源。

请参阅

SYS(2005)

SET SAFETY 命令

设置是否在决定改写已有文件之前显示警告。

语法

SET SAFETY ON | OFF

参数描述

ON

（默认值）指定改写已有文件之前显示一个对话框。该对话框将提供选项使您决定是否改写已有文件。

对于表设计器或 ALTER TABLE 命令，指定表结构被修改后重新计算表规则或字段规则、默认值以及错误信息等。保存表结构所做的修改之后，将使用新的或修改后的数据有效性规则进行数据有效性检查。如果有效性规则包含 UDF（用户自定义函数），将不计算 UDF 并且忽略有效性规则。

OFF

指定在改写已有文件时不显示对话框。注意，对于内部处理 .dll 自动服务程序，SET SAFETY 默认设置为 OFF。

对于表设计器，指定表结构被修改后不重新计算表规则或字段规则、默认值以及错误信息等。但是，保存表结构所做的修改之后，将使用新的或修改后的数据有效性规则进行数据有效性检查。对于 ALTER TABLE 命令，指定表结构被修改后不重新计算表规则或字段规则、默认值以及错误信息等。保存表结构所做的修改之后，也不使用新的或修改后的数据有效性规则进行数据有效性检查。

说明

SET SAFETY 的作用范围是当前数据工作期。

请参阅

ALTER TABLE-SQL, MODIFY STRUCTURE, SET DATASESSION, SET TALK

SET SECONDS 命令

当显示日期时间值时，指定是否显示时间部分的秒。

语法

```
SET SECONDS ON | OFF
```

参数描述

ON

（默认值）指定在日期时间值中显示秒。

OFF

指定在日期时间值中不显示秒。

说明

SET SECONDS 的设置只对当前的数据工作期有效

示例

下面的示例说明了 SET SECONDS 设置对 DATETIME( ) 函数返回值的影响。当 SET SECONDS 设置为 ON 时，显示的值有秒部分。当 SET SECONDS 设置为 OFF 时，显示的值没有秒部分。

```
SET SECONDS ON
CLEAR
? DATETIME( )  && 显示包含秒的时间

SET SECONDS OFF  && 显示不包含秒的时间
? DATETIME( )
```

请参阅

CTOT( ), DATE( ) , DATETIME( ) , DTOT( ) , HOUR( ) , MINUTE( ) , SEC( ) , SECONDS( ) , SET SECONDS, TIME( )

SET SEPARATOR 命令

指定用来分隔小数点左边每三位数字之间的分隔符。包含此命令是为了提供向后兼容性，请使用 Windows 控制面板。

SET SKIP 命令

创建表与表之间的一对多关系。

语法

SET SKIP TO [*TableAlias1* [, *TableAlias2*] ...]

参数描述

TO TableAlias1 [, TableAlias2] ...

指定多个子表的别名。这些子表用来与父表创建一对多关系。表别名之间用逗号分隔。在支持范围的命令（DISPLAY、LIST 等）中，对于子表中每一个对应记录都重复父表的记录。

不带参数的 SET SKIP TO 命令从当前选定工作区的已打开父表中删除该一对多关系，而任何一对一关系仍然有效。一对一关系可以用 SET RELATION TO 删除。

说明

使用 SET RELATION 可以在不同工作区中打开的表之间建立关系。当记录指针在父表中移动时，子表中的记录指针也移动到第一个对应的记录上。SET RELATION 中的关系表达式决定子表中的记录指针移到何处。如果对于父表中每一个记录都建立了一对一的关系，记录指针将移到子表中第一个相匹配的记录上。如果在子表中找不到相匹配的记录，则记录指针移到表尾。

很多情况下，父表中的一个记录通常在子表中有多个记录与之对应。SET SKIP 允许在父表的一个记录与子表的多个记录之间建立一对多关系。当浏览父表时，父表的记录指针将一直保持不动，直到记录指针移过子表中所有相关的记录为止。

要建立一对多关系，首先使用 SET RELATION 在父表与子表之间建立关系。然后发出 SET SKIP 命令创建一对多关系。

示例

下面示例在三个表中查找第一个字段内容相同的记录。首先使用 SCAN 扫描第一个表。这个表与第二个表有关系，而第二个表与第三个表有关系。然后第一个表为其他两表发出 SET SKIP 命令。请注意 SET SKIP 对第二个表没有影响。它只影响被扫描（替代，等等）的表。在这个示例中，查到了 8 条记录。

```
CLOSE DATABASES
CREATE TABLE Parent FREE (Name C(1), Val C(10))      && 父表有一个 a 和 b.

INSERT INTO Parent VALUES ('a', 'Parent.a1')
INSERT INTO Parent VALUES ('b', 'Parent.b1')

SELECT  0     && 子表 1 有两个 a 和 b
CREATE TABLE Child1 FREE (Name1 C(1), Val C(10))
INSERT INTO Child1 VALUES ('a', 'Child1.a1')
INSERT INTO Child1 VALUES ('b', 'Child1.b1')
INSERT INTO Child1 VALUES ('b', 'Child1.b2')
INSERT INTO Child1 VALUES ('a', 'Child1.a2')
INDEX ON Name1 TAG tagName   && 标识名是无关紧要的
```

```
SELECT 0    && Child2 有两个 a 和 b
CREATE TABLE Child2 FREE (Name2 C(1), Val C(10))
INSERT INTO Child2 VALUES ('b', 'Child1.b1')
INSERT INTO Child2 VALUES ('b', 'Child1.b2')
INSERT INTO Child2 VALUES ('a', 'Child1.a1')
INSERT INTO Child2 VALUES ('a', 'Child1.a2')
INDEX ON Name2 TAG tagName   && 标识名是无关紧要的
SELECT Child1
SET RELATION TO Name1 INTO Child2
SELECT Parent
SET RELATION TO Name INTO Child1
SET SKIP TO Child1, Child2     && 注意对两个子表都设置 SKIP.
                        && 否则，只有 4 条记录列出.
SCAN ALL   && 有 8 种情况:4 个 a 和 4 个 b
    ? Parent.Val, Child1.Val, Child2.Val
ENDSCAN
```

请参阅

RELATION(), SET RELATION, SKIP

SET SKIP OF 命令

启用或禁止用户自定义菜单或 Visual FoxPro 系统菜单的菜单、菜单栏、菜单标题或

菜单项。

语法

SET SKIP OF MENU MenuBarName1 lExpression1

–或–

SET SKIP OF PAD MenuTitleName OF MenuBarName2 lExpression2

–或–

SET SKIP OF POPUP MenuName1 lExpression3

–或–

SET SKIP OF BAR nMenuItemNumber | SystemItemName OF MenuName2 lExpression4

参数描述

MENU MenuBarName1 lExpression1

启用或禁止 Visual FoxPro 系统菜单栏或使用 DEFINE MENU 创建的自定义菜单栏。例如，可以使用以下命令禁止 Visual FoxPro 的系统菜单栏 _MSYSMENU：

```
SET SKIP OF MENU _MSYSMENU .T.
```

使用以下命令启用它：

```
SET SKIP OF MENU _MSYSMENU .F.
```

PAD MenuTitleName OF MenuBarName2 lExpression2

启用或禁止由 DEFINE PAD **创建的** Visual FoxPro **系统菜单标题或用户自定义菜单标题。可以用如下命令禁止** Visual FoxPro **的“编辑”菜单标题：**

SET SKIP OF PAD _MSM_EDIT OF _MSYSMENU .T.

可以用如下命令激活：

SET SKIP OF PAD _MSM_EDIT OF _MSYSMENU .F.

POPUP MenuName1 lExpression3

启用或禁止由 DEFINE POPUP **创建的** Visual FoxPro **系统菜单或用户自定义菜单。**

可以用如下命令禁止 Visual FoxPro **“编辑”菜单：**

SET SKIP OF POPUP _MEDIT .T.

可以用如下命令激活：

SET SKIP OF POPUP _MEDIT .F.

BAR nMenuItemNumber | SystemItemName OF MenuName2 lExpression4

启用或禁止由 DEFINE BAR **创建的** Visual FoxPro **系统菜单上的菜单项或用户自定义菜单项。**

可以用如下命令禁止 Visual FoxPro **“文件”菜单上的“新建”命令：**

SET SKIP OF BAR _MFI_NEW OF _MFILE .T.

这里， *SystemItemName* **指定菜单命令** _MFI_NEW，*MenuName2* **指定菜单** _MFILE，*lExpression4* **指定逻辑表达式** .T.。**此菜单命令可以用如下命令激**

活：
SET SKIP OF BAR _MFI_NEW OF _MFILE .F.

使用 *nMenuItemNumber* 可指定用 DEFINE BAR 创建的菜单项。

说明

有关 Visual FoxPro 系统菜单各部分内部名称的完整列表，请参阅系统菜单名称。也可使用 SYS(2013) 来返回系统菜单的内部名称。

如果逻辑表达式 *lExpression* 的计算值为“真”(.T.)，则 SET SKIP OF 命令中的菜单、菜单栏、菜单名或菜单项被禁止。禁止的项以灰色显示且不能选择。如果逻辑表达式 *lExpression* 的计算值为“假”(.F.)，则激活相应的菜单、菜单栏、菜单名或菜单项，可以选择。

请参阅

CREATE MENU, DEFINE BAR, DEFINE MENU, DEFINE PAD, DEFINE POPUP, System Menu Names 概览, SKPBAR( )

SET SPACE 命令

使用 ? 或 ?? 命令时，确定字段或表达式之间是否显示空格。

语法

SET SPACE ON | OFF

参数描述

ON

（默认值）在字段或表达式之间插入空格。

OFF

移去字段或表达式之间的空格，并且将它们连接在一起输出。

请参阅

? | ??

SET STATUS 命令

显示或移去基于字符的状态栏。

语法

SET STATUS ON | OFF

参数描述

ON

显示基于字符的状态栏。如果 SET STATUS 为 ON，则显示基于字符的状

态栏，显示当前执行的程序名（如果有）、活动表的别名、当前记录指针位置、表中的记录数量，以及 Insert、NumLock 和 CapsLock 键的状态。当共享打开一个表时，也显示该表的记录或文件锁定状态。当每次执行更改状态信息的命令时，都更新状态栏。

OFF
（默认值）发出 SET STATUS OFF 命令可以移去状态栏。.

请参阅

SET STATUS BAR, StatusBarText 属性

SET STATUS BAR 命令

显示或删除图形状态栏。

语法

SET STATUS BAR ON | OFF

参数描述

ON
显示图形状态栏。

OFF
移去图形状态栏。

请参阅

SET STATUS, StatusBarText 属性

SET STEP 命令

为程序调试打开跟踪窗口，并挂起程序。

语法

SET STEP ON

参数描述

ON
打开跟踪窗口，并挂起程序。

说明

SET STEP 用于调试程序。可在程序中需要逐条执行命令的地方插入 SET STEP ON 命令。其他相关信息，请参阅《 Microsoft Visual FoxPro 6.0 中文版程序员指南》第十四章“测试和调试应用程序”。

可以通过如下步骤来把参数传给程序并跟踪它的运行：

1. 打开跟踪窗口。

2. 从跟踪窗口的“程序”菜单中选择“打开”命令选择要跟踪的程序。
3. 在程序的第一个可执行行上设置断点。
4. 在命令窗口中，带参数运行程序（DO ... WITH... ）。

请参阅

SET ECHO

SET STRICTDATE 命令

指定不明确的日期和日期时间常数是否产生错误。

语法

SET STRICTDATE TO [0 | 1 | 2]

参数描述

0
（默认值）指定关闭严格的日期格式检查。这个设置提供了对以前版本的 Visual FoxPro 的兼容性。

1
指定所有的日期和日期时间常数必须符合严格的日期格式。任何不符合严格

日期格式的日期或日期时间常数，或任何无效值都会产生一个错误，不论是在编译时、运行时或在一个交互的 Visual FoxPro 工作期中。

2

对于将 STRICTDATE 设置为 1，但是也加强了 CTOD() 和 CTOT() 函数中字符串的严格日期格式。注意，要小心应用这个级别，因为 Visual FoxPro 6.0 以前版本的代码使用的日期格式可能不符合严格数据格式，有可能造成难以料的运行时错误。

因为 CTOD（）和 CTOT（）返回的数值依赖于 SET DATE 和 SETCENTURY 对日期设置的解释，所以有产生 2000 不兼容性的错误。

这个级别最适合用于调试时，捕捉可能引起 2000 兼容性错误的代码。

说明

注意，SET STRICTDATE 的设置不影响 StrictDateEntry 属性。

有关严格的日期格式的详细内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》的第三十三章“对编程的改进”中的“对 2000 年日期的支持”。

请参阅

COMPILE, CTOD( ), CTOT( ), SET LOGERRORS, StrictDateEntry 属性

SET SYSFORMATS 命令

指定是否用当前 Windows 系统设置值更新 Visual FoxPro 系统设置。

语法

SET SYSFORMATS ON | OFF

参数描述

ON

指定当 Windows 系统设置改变时更新 Visual FoxPro 系统设置。SET SYSFOMAT ON 与选择“选项”对话框中“国际”选项卡上的“使用系统设置”复选框效果一样。执行 SET SYSFORMAT ON 会自动使得 SET DATE 设为 SHORT。

本设置通常只对当前数据工作期有效，如果它是在默认数据工作期中设置的，则对整个 Virsual FoxPro 工作期都有效。

OFF

（默认值）指定当 Windows 系统设置改变时不更新 Visual FoxPro 系统设置。

说明

Windows 系统设置可以在 Windows “控制面板”中的“区域设置”选项中指定。

当 SET SYSFORMATS 为 ON 时，可以用下面的 SET 命令来改写当前系统设置。但是，当 SET SYSFORMATS 为 ON 时修改 Windows 系统设置也将改写这些 SET 命令。

- SET CENTURY
- SET CURRENCY
- SET DATE
- SET DECIMALS
- SET HOURS
- SET MARK TO
- SET POINT
- SET SEPARATOR

Visual FoxPro 启动时的系统设置是这些 SET 命令的默认设置：

要在启动 Visual FoxPro 时使用 Windows 系统设置，应在 Visual FoxPro 配置文件 CONFIG.FPW 中加入如下命令：

```
SYSFORMATS = ON
```

请参阅

SET CENTURY, SET CURRENCY, SET DATASESSION, SET DATE, SET DECIMALS, SET HOURS, SET MARK TO, SET POINT, SET SEPARATOR

SET SYSMENU 命令

在程序运行期间，启用或禁止 Visual FoxPro **系统菜单栏，并对其重新配置。**

语法

```
SET SYSMENU ON | OFF | AUTOMATIC
    | TO [MenuList]
    | TO [MenuTitleList]
    | TO [DEFAULT] | SAVE | NOSAVE
```

参数描述

ON

在程序执行期间，当 Visual FoxPro **等待诸如** BROWSE、READ **和** MODIFY **命令等命令的键盘输入时，启用** Visual FoxPro **主菜单栏。**

OFF

在程序执行期间，禁止 Visual FoxPro **主菜单栏。**

AUTOMATIC

使 Visual FoxPro **主菜单栏在程序执行期间可见。可以访问菜单栏，但菜单项是启用还是禁止则取决于不同的命令。**

在 Visual FoxPro **中，默认设置是** AUTOMATIC。

TO [MenuList]

TO [MenuTitleList]

指定 Visual FoxPro 主菜单栏中菜单或菜单标题的子集。这些菜单或菜单标题可以是主菜单中的菜单或菜单标题的任意组合，相互之间用逗号隔开。菜单和菜单标题的内部名称列在系统菜单名称中。

例如，下面的命令从 Visual FoxPro 主菜单栏中移去除“文件”和“窗口”菜单外的所有菜单：

```
SET SYSMENU TO _MFILE, _MWINDOW
```

使用 RELEASE BAR 可指定菜单中的可用菜单项。

TO [DEFAULT]

将主菜单栏恢复为默认设置。如果对主菜单栏或它的菜单做过修改，可发出 SET SYSMENU TO DEFAULT 命令恢复。使用 SET SYSMENU SAVE 可以指定默认设置。

SAVE

使当前菜单系统成为默认设置。如果在发出 SET SYSMENU SAVE 命令之后修改了菜单系统，可以通过发出 SET SYSMENU TO DEFAULT 命令来恢复前面的设置。

NOSAVE

重置菜单系统为默认的 Visual FoxPro 系统菜单。但是，只有当发出 SET SYSMENU TO DEFAULT 命令之后才显示默认的 Visual FoxPro 系统菜单。

说明

SET SYSMENU 控制程序运行期间的 Visual FoxPro 主菜单栏。它允许有选择地移去 Visual FoxPro 主菜单系统中的菜单标题和菜单，并可以将其恢复。

不带其他参数的 SET SYSMENU TO 命令禁止 Visual FoxPro 主菜单栏。

请参阅

系统菜单名, POP MENU, PUSH MENU, RELEASE, SYS(2013)

SET TALK 命令

决定 Visual FoxPro **是否显示命令结果。**

语法

SET TALK ON | OFF | WINDOW [*WindowName*] | NOWINDOW

参数描述

ON

（默认值）允许把对话结果发送到 Visual FoxPro 主窗口、系统信息窗口、图形状态栏或用户自定义窗口中。如果 SET TALK 设置为 OFF 后再改为 ON，那么对话结果将传送到发出 SET TALK OFF 命令之前的输出位置。

OFF
阻止对话结果传送到 Visual FoxPro 主窗口、系统信息窗口、图形状态栏或用户自定义窗口中。

WINDOW [*WindowName*]
WindowName 指定接收对话结果的用户自定义窗口。接收对话结果的用户自定义窗口必须在传送对话结果之前创建。如果指定的窗口不存在，对话结果将发送到 Visual FoxPro 系统窗口中。

NOWINDOW
将对话结果直接传送到 Visual FoxPro 主窗口中。

说明

有些表处理命令返回有关运行状态的信息（或“对话结果”）。这些命令有：

APPEND FROM	AVERAGE	CALCULATE
COPY TO	COUNT	DELETE
INDEX	PACK	REINDEX
REPLACE	SELECT – SQL	SORT
SUM	TOTAL	

在 Visual FoxPro 中，对话结果直接传送到 Visual FoxPro 主窗口、图形状态栏或用户自定义窗口中。对话也可以被关掉。

SET TALK 的汇报时间间隔可以用 SET ODOMETER 命令指定。SET ODOMETER 的默认设置是 100。请注意当 SET TALK 设置为 ON 时，程序的运行速度会降低。这

是因为 Visual FoxPro 主窗口或用户自定义窗口必须经常更新的缘故。如果只需知道某一命令处理记录的总数，就可以在程序执行完后就发出 SET TALK OFF 命令并显示 _TALLY 系统变量。

SET TALK 的作用范围是当前数据工作期。

请参阅

DEFINE WINDOW, SET DATASESSION, SET NOTIFY, SET ODOMETER, SYS(103), _TALLY

SET TEXTMERGE 命令

指定是否对文本合并分隔符括起的字段、变量、数组元素、函数或表达式进行计算，并允许指定文本合并输出。

语法

```
SET TEXTMERGE
    [ON | OFF]
    [TO [FileName] [ADDITIVE]]
    [WINDOW WindowName]
    [SHOW | NOSHOW]
```

参数描述

ON

指定对文本合并分隔符括起的字段、变量、数组元素、函数或表达式进行计算。而且当它们放在 \ 或 \\ 之后或 TEXT 和 ENDTEXT 之间时将其输出。

下面的程序示例演示了当 SET TEXTMERGE 设置为 ON 时，gcTodayDate 变量的内容和 DATE( ) 及 TIME( ) 函数是如何计算的。之所以计算 gcTodayDate 变量、DATE( ) 及 TIME( ) 函数，是因为它们用文本合并分隔符分隔了，并且 SET TEXTMERGE 设置为 ON。

```
CLEAR
SET TALK OFF
STORE 'Today is: ' TO gcTodayDate
SET TEXTMERGE ON
\<<gcTodayDate>>
\\<<DATE( )>>
\The time is:
\\ <<TIME( )>>
```

下面是在一月一日运行时，该程序的结果：

```
Today is: 01/01/1998
The time is: 10:55:19
```

OFF
（默认值）指定对文本合并分隔符括起的字段、变量、数组元素、函数或表达式同括起它们的文本合并分隔符一起不作变化地输出。

```
CLEAR
SET TALK OFF
STORE 'Today is: ' TO gcTodayDate
SET TEXTMERGE OFF
\<<gcTodayDate>>
\\<<DATE( )>>
\The time is:
\\ <<TIME( )>>
```

下面是程序输出：

```
<<gcTodayDate>><<DATE( )>>
The time is: <<TIME( )>>
```

TO [FileName]
指定 \、\\ 和 TEXT ... ENDTEXT 产生的结果不输出到默认的 Visual FoxPro 主窗口中，而是直接输出到文本文件中。可以指定 *FileName* 把结果输出到某一文本文件中。如果指定的文件不存在，则创建新文件。如果指定的文件已经存在且 SET SAFETY 设置为 ON 时，可以选择改写已有文件。

以低级方式打开文本文件，文件句柄保存在 _TEXT 系统变量中。可以用不带参数的 SET TEXTMERGE TO 命令来关闭该文件。如果另一个文件的句柄先前已保存在 _TEXT 中，则关闭此文件。

ADDITIVE

指定 \、\\ 和 TEXT ... ENDTEXT 产生的输出追加在已有文件的后面。有关将文本合并输出到文件的详细内容，请参阅 _TEXT 系统变量。

WINDOW WindowName

指定 \、\\ 和 TEXT ... ENDTEXT 产生的结果不输出到默认的 Visual FoxPro 主窗口中，而是直接输出到用户自定义窗口中。*WindowName* 指定接收输出结果的窗口名。这一窗口必须在接收输出结果前用 DEFINE WINDOW 创建，可以不是活动窗口或可见窗口。

SHOW | NOSHOW（默认值）

SHOW 显示文本合并结果。

NOSHOW 不显示文本合并结果。

默认情况下，由 \、\\ 和 TEXT ... ENDTEXT 产生的输出结果发送到 Visual FoxPro 主窗口或活动的用户自定义窗口。

说明

\、\\ 和 TEXT ... ENDTEXT 命令用表、变量、数组元素、以及函数和表达式的计算结果替换文本中的内容。如果字段、变量、数组元素、函数或表达式用文本合并分隔符（默认为 << 和 >>）括起来了，可以计算它们的值然后以文本形式替换其内容。这一文本合并功能可以用于产生信件、程序和创建程序的模板。

SET TEXTMERGE 确定用文本合并分隔符括起来的字段、变量、数组元素、函数或

表达式的计算方式。它还可以直接将替换的结果输出到 Visual FoxPro 主窗口、用户自定义窗口或文件中。

可以用备注字段嵌套被替换文本。如果备注字段中包含有当前文本合并分隔符括起来的字段名、变量、数组元素、函数或表达式，则计算它们的值，并随备注字段的内容一起输出。备注字段的名称也必须用文本合并分隔符括起来。

请参阅

\ | \\, _PRETEXT, SET TEXTMERGE DELIMITERS, _TEXT

SET TEXTMERGE DELIMITERS 命令

指定文本合并分隔符。

语法

```
SET TEXTMERGE DELIMITERS
    [TO cLeftDelimiter [, cRightDelimiter]]
```

参数描述

TO cLeftDelimiter [, cRightDelimiter]

指定分隔符。如果只使用 *cLeftDelimiter* 指定一个分隔符，那么左边和右边的分隔符都设置为 *cLeftDelimiter*。如果指定两个分隔符 *cLeftDelimiter* 和

cRightDelimiter，则左边的分隔符设置为 *cLeftDelimiter*，右边的分隔符设置为 *cRightDelimiter*。

说明

使用 SET TEXTMERGE DELIMITERS 可以指定不同于默认分隔符的一套文本合并分隔符。默认的文本合并分隔符为双尖括号符（<< 和 >>）。可用 DISPLAY STATUS 来显示当前的分隔符。

如果发出不带参数的 SET TEXTMERGE DELIMITERS 命令，文本合并分隔符将恢复为默认设置值。

有关文本合并分隔符的详细内容，请参阅 SET TEXTMERGE。

请参阅

\ | \\, _PRETEXT, SET TEXTMERGE, _TEXT

SET TOPIC 命令

指定激活 Visual FoxPro 帮助系统时，要打开的帮助主题。

语法

SET TOPIC TO [cHelpTopicName | lExpression]

参数描述

cHelpTopicName
指定欲显示的帮助主题名称。
lExpression
一个逻辑表达式，它是打开指定的主题或主题的基础。

说明

有关创建自己的帮助系统的详细内容，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》。

请参阅

HELP, SET HELP, SET HELPFILTER, SET TOPIC ID

SET TOPIC ID 命令

指定激活 Visual FoxPro 帮助系统时要显示的帮助主题。帮助主题根据主题的上下文 ID 确定。

语法

SET TOPIC ID TO *nHelpContextID*

参数描述

OnHelpContextID

指定要显示的帮助主题的上下文 ID。nHelpContextID **在帮助项目文件中指定** MAP **部分的上下文数字。**

请参阅

HELP, SET HELP, SET TOPIC

SET TRBETWEEN **命令**

在跟踪窗口的断点之间启用或禁止跟踪。

语法

SET TRBETWEEN ON | OFF

参数描述

ON

（默认值）执行程序，在跟踪窗口中突出显示每行代码，使用 SET TRBETWEEN ON **命令和跟踪窗口的“程序”菜单中激活“在两个断点之间跟踪”命令一样。**

OFF

在跟踪窗口中，只突出显示程序运行暂停时的最后一行。使用 SET TRBETWEEN OFF **命令与禁止“在两个断点之间跟踪”命令一样。**

说明

在程序执行时，跟踪窗口显示程序的源代码，突出显示正在执行的程序行。当打开跟踪窗口时，可以设置终止程序的断点。

也可以用 ACTIVATE WINDOW TRACE、SET ECHO ON 或 SET STEP ON 打开跟踪窗口。

请参阅

ACTIVATE WINDOW, SET ECHO, SET STEP

SET TYPEAHEAD 命令

指定键盘缓冲区中可以存储的最大字符数。

语法

SET TYPEAHEAD TO *nCharacters*

参数描述

nCharacters

指定键盘缓冲区中可以存储的最大字符数。

如果使用 SET TYPEAHEAD TO 0，则键盘缓冲区中不保存任何字符，这条

语句使 INKEY() 和 ON KEY 均无效。

说明

在进行字符处理之前，键盘缓冲区可存储多达 32,000 个字符，SET TYPEAHEAD 的默认值是 20。

请参阅

INKEY(), ON KEY

SET UDFPARMS 命令

Visual FoxPro 在向用户自定义函数 (UDF) 传递参数时，指定为按值传递还是通过引用传递。

语法

SET UDFPARMS TO VALUE | REFERENCE

参数描述

TO VALUE

向一个用户自定义函数按值传递一个变量。当按值传递变量时，在用户自定义函数中可以更改变量的值，但调用程序中变量的原值不变。

TO REFERENCE

向一个用户自定义函数通过引用传递一个变量。当通过引用传递变量时，如果在用户自定义函数中更改了变量的值，则调用程序中变量的原值也随之更改。

说明

默认情况下，变量以值传递方式传递给用户自定义函数（用 DO ... WITH 传递给过程的变量为通过引用传递）。

不管 SET UDFPARMS 如何设置，可以强制设定以值传递方式或引用传递方式向一个 UDF 传递参数：如果用括号括起一个变量，则设定按值传递；在一个变量前加 @ 符号，可以设定以引用传递方式传递。

提示　可以把整个数组传递给一个过程或用户自定义函数。如果使用 SET UDFPARMS TO REFERENCE 命令，或者在数组名前加 @，就可以传递整个数组；如果使用 SET UDFPARMS TO VALUE 命令，或者用括号把数组名括起来，可以按值传递数组的第一个元素。数组元素总是按值传递。

示例

下面的示例展示了按值传递和引用传递变量的区别。

```
*** 按值传递变量***
CLEAR
SET TALK OFF
WAIT '按下键按值传递 …' WINDOW
SET UDFPARMS TO VALUE
STORE 1 TO gnX
```

```
*** gnX 的值没有更改***
@ 2,2 SAY 'UDF value: ' + STR(plusone(gnX))
@ 4,2 SAY 'Value of gnX: ' + STR(gnX)

*** 通过引用传递变量***
WAIT '按下键按引用传递 …' WINDOW
CLEAR
SET UDFPARMS TO REFERENCE
STORE 1 TO gnX
*** gnX 的值改变了***
@ 2,2 SAY 'UDF value: ' + STR(plusone(gnX))
@ 4,2 SAY 'Value of X: ' + STR(gnX)
SET UDFPARMS TO VALUE

***这个 UDF 给一个数加 1 ***
FUNCTION plusone
PARAMETER gnZ
gnZ = gnZ + 1
RETURN gnZ
*** UDF 结束 ***
```

下面的示例和上面的一样，只是分别用括号和 @ 来表示按值和通过引用传递变量：

```
*** 按值传递变量***
CLEAR
SET TALK OFF
WAIT '按下键按引用传递 …' WINDOW
STORE 1 TO gnX
```

```
@ 2,2 SAY 'UDF value: ' + STR(plusone((gnX)))
@ 4,2 SAY 'Value of gnX: ' + STR(gnX)

*** 通过引用传递变量 ***
WAIT '按下键按引用传递 ...' WINDOW
CLEAR
STORE 1 TO gnX
*** gnX 的值改变了***
@ 2,2 SAY 'UDF value: ' + STR(plusone(@gnX))
@ 4,2 SAY 'Value of gnX: ' + STR(gnX)
*** 这个 UDF 给一个数加 1***
FUNCTION plusone
PARAMETER gnZ
gnZ = gnZ + 1
RETURN gnZ
*** 结束***
```

以下是对上面的示例中的变量，通过括号和 @ 的使用分别进行值传递和引用传递。

```
***按值传递内存变量***
CLEAR
SET TALK OFF
WAIT '按下键按值传递' WINDOW
STORE 1 TO gnX
@ 2,2 SAY 'UDF value: ' + STR(plusone((gnX)))
@ 4,2 SAY 'Value of gnX: ' + STR(gnX)
```

```
***通过引用传递内存***
WAIT '按下键按值传递' WINDOW
CLEAR
STORE 1 TO gnX
@ 2,2 SAY 'UDF value: ' + STR(plusone(@gnX))
@ 4,2 SAY 'Value of gnX: ' + STR(gnX)

*** 这个 UDF 给一个数加 1***
FUNCTION plusone
PARAMETER gnZ
gnZ = gnZ + 1
RETURN gnZ
*** UDF 结束***
```

请参阅

DO, LPARAMETERS, PARAMETERS, PARAMETERS(), PROCEDURE

SET UNIQUE 命令

指定具有重复索引关键字的记录是否保留在索引文件中。

语法

```
SET UNIQUE ON | OFF
```

参数描述

ON
指定全部具有重复索引关键字的记录都不保留在索引文件中，只保留具有原始索引关键字值的第一个记录。

OFF
（默认值）把具有重复索引关键字的记录保留到索引文件中。

说明

当使用 REINDEX 命令时，索引文件保留它的 SET UNIQUE 设置，有关详细内容，请参阅 INDEX 和 REINDEX。

SET UNIQUE 作用范围是当前数据工作期。

请参阅

INDEX, REINDEX, SET DATASESSION

SET VIEW 命令

打开或关闭查看窗口，或者从一个视图文件中恢复 Visual FoxPro 环境。

语法

SET VIEW ON | OFF

–或–

SET VIEW TO FileName | ?

参数描述

ON

打开查看窗口。

OFF

（默认值）关闭查看窗口。

TO FileName

把 Visual FoxPro 环境还原成视图文件创建时的状态。其中用 *FileName* 指定视图文件，用 CREATE VIEW 命令创建视图文件。

?

显示“打开”对话框，从对话框中打开一个视图文件。

说明

有关视图文件的详细内容，请参阅 CREATE VIEW。

请参阅

CREATE VIEW, SET

SET WINDOW OF MEMO 命令

指定可以编辑备注字段的窗口。包含此命令是为了提供向后兼容性。

SET() 函数

返回各种 SET 命令的状态。

语法

SET(cSET 命令 [, 1 | cExpression | 2 | 3])

返值类型

字符型, 数值型

参数描述

cSETCommand

一个字符表达式，指定要返回当前状态信息的 SET 命令。指定命令的当前设置作为一个字符串或数值串返回。

1 | cExpression | 2 | 3]

返回一个 SET 命令的附加信息。1 和 cExpression 的作用相同；cExpression 可以是取值为字符型的任何表达式。包含一个参数并不意味着对所有 SET 命令都返回附加信息，它只是返回下表所列的标有 1、2 和 3 的命令附加信息。

SET 命令	返回值
ALTERNATE	ON 或 OFF
ALTERNATE, 1	FileName
BELL, 1	cWAVFileName
CENTURY	ON 或 OFF
CENTURY, 1	nCentury
CENTURY, 2	ROLLOVER nYear
CLOCK	ON 或 OFF
CLOCK, 1	*nRow* 和 *nColumn*
COMPATIBLE	ON 或 OFF
COMPATIBLE, 1	PROMPT 或 NOPROMPT
COVERAGE,1	FileName
CURRENCY	LEFT 或 RIGHT
CURRENCY, 1	cCurrencySymbol
DATE	AMERICAN, ANSI, BRITISH/FRENCH, GERMAN, ITALIAN, JAPAN, USA, MDY, DMY, 或 YMD
DATE, 1	日期的格式： 0 - MDY 1 - DMY 2 - YMD

续表

DELIMITERS	ON 或 OFF
DELIMITERS, 1	cDelimiters
EVENTTRACKING	ON 或 OFF
EVENTTRACKING, 1	FileName
FIELDS	ON 或 OFF
FIELDS, 1	FieldName1, FieldName2, ...
FIELDS, 2	LOCAL 或 GLOBAL
HELP	ON 或 OFF
HELP, 1	FileName
KEY	eExpression2, eExpression3
KEY, 1	eExpression2
KEY, 2	eExpression3
MESSAGE	nRow
MESSAGE, 1	cMessageText
MOUSE	ON 或 OFF
MOUSE, 1	nSensitivity
PRINTER	ON 或 OFF
PRINTER, 1	FileName 或 PortName
PRINTER, 2	默认的 Windows 打印机名称

续表

PRINTER, 3	默认的 Visual FoxPro 打印机名称（是在 Visual FoxPro “打印机”或“打印设置”对话框中指定的。）
RESOURCE	ON 或 OFF
RESOURCE, 1	FileName
TALK	ON 或 OFF
TALK, 1	WINDOW, NOWINDOW 或 WindowName
TEXTMERGE	ON 或 OFF
TEXTMERGE, 1	cLeftDelimiter and cRightDelimiter
TOPIC	cHelpTopicName|lExpression
TOPIC, 1	nContextID

说明

SET( ) 可以识别所有 Visual FoxPro 中 SET 关键字的四字符缩写形式（HELPFILTER 例外，它可以缩写为五个字符）。

例如，STAT 和 PRIN 可以分别用来代替 SET STATUS 和 SET PRINTER。

SET( ) 函数与 SYS(2001) 的作用一样。

请参阅

DISPLAY STATUS, LIST, SET, SYS(2001)

SetAll 方法

为容器对象中的所有控件或某类控件指定一个属性设置。

语法

Container.SetAll(*c*Property,*Value* [, *cClass*])

参数描述

cProperty

要设置的属性。

Value

属性的新设置，*Value* 的数据类型取决于要设置的属性。

cClass

指定类名，该类为对象的基类，不能是 Visual FoxPro 基类。

说明

使用 SetAll 方法可为容器中的所有控件或某类控件设置一个属性。例如，为了把表格控件中列对象的 BackColor 属性设置为红色，可以使用下列命令：

```
Form1.Grid1.SetAll( BackColor, RGB(255, 0, 0), 'Column')
```

也可以设置容器中其他对象包含的对象属性。要把表格控件中每个列对象包含的标头的 ForeColor 属性设置为绿色，可以使用下列命令：

```
Form1.Grid1.SetAll('ForeColor', RGB(0, 255, 0), 'Header')
```

示例

下面的示例用 Set All 方法和 DynamicBackColor 属性为一个表格控件中的记录指定背景颜色。如果表格中显示的记录编号是偶数，则记录的 DynamicBackColor 为白色，否则它的颜色为绿色。

在示例中，表格控件放在一个表单内，customer 表是打开的，而且内容用表格控件显示。Caption 属性用来为 CUST_ID 字段指定一个标头标题 (Customer ID)。表单中放有一个命令按钮，用来关闭表单。

```
CLOSE ALL  && 关闭表和数据库
OPEN DATABASE (HOME(2) + 'data\testdata')
USE customer  IN 0  && 打开 Customer 表
frmMyForm = CREATEOBJECT('Form')  && 创建一个表单
frmMyForm.Closable = .f.  && 禁止控件菜单框
frmMyForm.Addobject('cmd Command1','cmdMyCmdBtn')  && 添加命令按钮
frmMyForm.Addobject('grdGrid1','Grid')  && 添加表格控件

frmMyForm.grdGrid1.Left = 25     && 调整表格位置

frmMyForm.grdGrid1.SetAll("DynamicBackColor", ;
    "IIF(MOD(RECNO( ), 2)=0, RGB(255,255,255) ;
    , RGB(0,255,0))", "Column")  && 动态设置白和绿记录

frmMyForm.grdGrid1.Visible = .T.  && 使表格控件可见
frmMyForm.cmd Command1.Visible =.T.  && 使“Quit”命令按钮可视
frmMyForm.grdGrid1.Column1.Header1.Caption = 'Customer ID'
frmMyForm.SHOW  && 显示表单
```

```
READ eventS   && 开始事件循环

DEFINE CLASS cmdMyCmdBtn AS CommandButton   && 创建命令按钮
   Caption = '\<Quit'   && 为命令按钮加标题
   Cancel = .T.   &&  默认的取消命令按钮(Esc)
   Left = 125   && 命令按钮起始点所在的列
   Top = 210   && 命令按钮起始点所在的行
   Height = 25   && 命令按钮的高度

   PROCEDURE Click
      CLEAR eventS   && 终止事件循环，关闭表单
      CLOSE ALL   && 关闭表和数据库
ENDDEFINE
```

应用于

列，命令组，容器对象，表单，表单集，表格，选项组，页面，页框，_SCREEN，工具栏

请参阅

SaveAs 方法，SaveAsClass 方法

SETFLDSTATE() 函数

为表或临时表中的字段或记录指定字段状态值或删除状态值。

语法

SETFLDSTATE(cFieldName | nFieldNumber, nFieldState
[, cTableAlias | nWorkArea])

返值类型

逻辑型

参数描述

cFieldName | nFieldNumber

指定的字段名或字段编号，对此字段指定编辑或删除状态，字段编号 *nFieldNumber* 为字段在表或临时表结构中的位置。DISPLAY STRUCTURE 或 FIELD( ) 可以用来确定一个字段的编号。要给记录设置删除状态，把字段编号设置为 0。

nFieldState

为字段状态或删除状态指定一个值。下表列出字段状态值或删除状态值以及相应的编辑或删除状态。

nFieldState	编辑或删除状态
1	字段没有被编辑或删除状态没有改变。
2	字段已被编辑或删除状态已改变。
3	追加记录的字段没有被编辑或追加记录的删除状态没有改变。
4	追加记录的字段已被编辑或追加记录的删除状态已改变。

cTableAlias

表或临时表的别名，在其中指定编辑状态或删除状态。

nWorkArea

表或临时表的工作区。使用 SETFLDSTATE( ) 时，如果不带可选的 *cTableAlias* 或 *nWorkArea* 参数，则为当前选定工作区中打开的表或临时表指定字段状态值或删除状态值。

说明

Visual FoxPro 用字段状态值确定表或临时表的哪个字段被更新。SETFLDSTATE( ) 允许您控制 Visual FoxPro 的字段状态，而不管表或临时表中哪个字段进行了编辑。

示例

下面示例演示如何用 SETFLDSTATE( ) 改变字段状态。将 MULTILOCKS 设置为 ON 是表缓冲的要求，然后打开 testdata 数据库中的 customer 表，并用 CURSORSETPROP( ) 把缓冲方式设置成开放式表缓冲 (5)。GETFLDSTATE( ) 用来获取字段状态，这时显示一个值 (1)，表明 cust_id 字段未修改。在 cust_id 字段用 REPLACE 修改之后，再次使用 GETFLDSTATE( )，这时显示一个值 (2)，它表明 cust_id 字段已修改。SETFLDSTATE( ) 用来把 cust_id 字段的状态改回 1 (未修改)。调用 GETFLDSTATE( ) 后，显示值为 1，这与用 SETFLDSTATE( ) 指定的 cust_id 字段状态相对应。TABLEREVERT( ) 用来恢复表的原始状态。

```
CLOSE DATABASES
SET MULTILOCKS ON  && 为表缓冲必须设置为 ON
SET PATH TO (HOME( 2 ) + data\')  && 设置数据库的路径
OPEN DATABASE testdata  && 打开 testdata 数据库
USE Customer  && 打开 Customer 表
= CURSORSETPROP('Buffering', 5, 'customer')  && 启用表缓冲
```

```
Clear
? GETFLDSTATE('cust_id')   && 显示 1，未修改
REPLACE cust_id WITH '***'   && 更改字段内容
? GETFLDSTATE('cust_id')   && 返回 2，字段已修改
= SETFLDSTATE('cust_id', 1)   && 更改字段状态
? GETFLDSTATE('cust_id')   && 显示 1，未修改
= TABLEREVERT(.T.)   && 放弃所有对表的修改
```

请参阅

DELETED(), DISPLAY STRUCTURE, FIELD(), GETFLDSTATE()

SetFocus 方法

为一个控件指定焦点。

语法

Control.SetFocus

说明

如果控件的 Enabled 或 Visible 属性设置为“假”(.F.)，或者控件的 When 事件返回“假”(.F.)，则不能给一个控件指定焦点；如果 Enabled 或 Visible 属性已设置为“假”(.F.)，则控件在使用 SetFocus 方法接受焦点之前，必须首先把它们设置为“真”(.T.)。

一旦控件获得了焦点，用户的任何输入都针对这个控件。

应用于

复选框，列，组合框，命令按钮，容器对象，控件对象，编辑框，表格，列表框，OLE 绑定型控件，OLE 容器控件，选项按钮，微调，文本框

请参阅

Enabled 属性，GotFocus 事件，LostFocus 事件，Visible 属性

SetMain 方法

设置项目的主文件。

语法

Object.SetMain([cFileName [, cActiveDocClass]])

参数描述

cFileName

将项目中的一个文件指定为主文件。主文件可以是程序、表单，或包含一个 Active Document 类的可视类库。在 cFileName 中一定要包含文件的扩展名。如果省略 cFileName，或者 cFileName 是空字符串，则不将项目中的任何一个文件指定为主文件。

cActiveDocClass

将一个 Active Document 类指定为主文件。为了将一个 Active Document 类指定为主文件，cFileName 必须是包含了这个 Active Document 类的可视类库的名称。

说明

如果所指定的文件或 Active Document 类设置成主文件，则 SetMain 方法返回“真”(.T.)，如果所指定的文件或 Active Document 类不在项目中，或者所指定的文件的类型不正确，则返回“假” (.F.)。

主文件是一个程序（.prg 文件）、表单（.scx 文件）或 Active Document 类，作为一个已编译应用程序的执行开始点。通常，主文件设置了应用程序的运行环境，运行菜单程序或表单，以显示应用程序的界面，并且使用 READ EVETNS 命令建立应用程序的事件循环。在根据项目创建一个应用程序 (.app) 或可执行文件 (.exe) 之前，必须在项目管理器中指定一个主文件。

注意 将项目的一个文件或 Active Document 类指定为主文件，也设置该项目的 MainFile 和 MainClass 属性。

应用于

项目对象

请参阅

MainClass 属性, MainFile 属性

返回总目录

SQLMORERESULTS() 函数
SQLPREPARE() 函数
SQLROLLBACK() 函数
SQLSETPROP() 函数
SQLSTRINGCONNECT() 函数
SQLTABLES() 函数
SQRT() 函数
SROWS() 函数
StartMode 属性
_STARTUP 系统变量
StatusBar 属性
StatusBarText 属性
STORE 命令

SetVar 方法

为 Visual FoxPro 自动服务程序的一个实例创建一个变量，并将一个值保存在该变量中。

语法

Applicationobject.SetVar(cVariableName, eValue)

参数描述

cVariableName

指定要创建的变量的名称。

eValue

指定保存在变量中的值。如果 cVariableName 指定的变量已经存在，则将新值保存在该变量中。

说明

虽然可以使用 DoCmd 方法将一个变量设置为字符型，但是应该使用 SetVar 方法将一个变量设置为其他数据类型。

应用于

Application 对象，_VFP 系统变量

请参阅

DoCmd 方法，Eval 方法，STORE

SetViewPort 方法

设置表单的 ViewportLeft 和 ViewportTop 属性的值。

语法

object.SetViewPort(nLeft, nTop)

参数描述

nLeft

指定表单的 ViewportLeft 属性的值。

nTop

指定表单的 ViewportTop 属性的值。

说明

如果成功设置了 ViewportLeft 和 ViewportTop 属性的值，则 SetViewport 方法返回“真” (.T.)；否则返回“假” (.F.)。对于不包含滚动条的表单，将忽略 SetViewport 方法。

ViewportLeft 和 ViewportTop 属性的度量单位由表单的 ScaleMode 属性设置决定——像素（默认值）或 foxels。

应用于

表单

请参阅

ScaleMode 属性, ScrollBars 属性, ViewPortLeft 属性, ViewPortHeight 属性, ViewPortTop 属性, ViewPortWidth 属性

Shape 控件

 创建一个可以显示框、圆或椭圆的形状控件。

语法

Shape

说明

形状控件是可以显示矩形、圆或椭圆的图形控件，这些图形不能直接修改。但是，因为形状控件包括很多其他控件具有的属性、事件和方法，所以形状控件能响应事件，并且在运行时可动态地修改。

Curvature 属性决定显示什么样的图形，它的变化范围是 0 到 99。0 表示无曲率，用来创建矩形；99 表示最大曲率，创建圆和椭圆。

有关创建图形的其它信息，请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》的第十章“使用控件”。

属性

Application	BackColor	BackStyle
BaseClass	BorderColor	BorderStyle
BorderWidth	Class	ClassLibrary
ColorScheme	ColorSource	Comment
Curvature	DragIcon	DragMode
DrawMode	Enabled	FillColor
FillStyle	Height	HelpContextID
Left	MouseIcon	MousePointer
Name	OLEDragMode	OLEDragPicture
OLEDropEffects	OLEDropHasData	OLEDropMode
Parent	ParentClass	SpecialEffect
Tag	ToolTipText	Top
Visible	WhatsThisHelpID	Width

事件

Click	DblClick	Destroy
DragDrop	DragOver	Error
Init	MiddleClick	MouseDown
MouseMove	MouseUp	MouseWheel

续表

OLECompleteDrag	OLEDragDrop	OLEDragOver
OLEGiveFeedBack	OLESetData	OLEStartDrag
RightClick	UIEnable	

方法

AddProperty	CloneObject	Drag
Move	OLEDrag	ReadExpression
ReadMethod	ResetToDefault	SaveAsClass
ShowWhatsThis	WriteExpression	WriteMethod
ZOrder		

示例

下面的示例演示了如何用形状控件在表单上显示圆、椭圆和长方形。创建一个表单，并在表单上放置一套选项按钮和一个命令按钮。单击一个选项按钮，会在表单上显示相应的形状。设置 Visible 属性为“真”(.T.)以显示形状；设置为“假”(.F.)，则在显示另一个形状前隐藏此形状。每个形状的 Height、Width 和 Curvature 属性决定了所创建的形状类型（圆、椭圆或长方形）。

```
frmMyForm = CREATEobject('Form') && 创建一个表单
frmMyForm.Closable = .F. && 取消控件菜单栏

frmMyForm.Addobject('cmdCommand1','cmdMyCmndBtn') && 增加命令按扭
frmMyForm.Addobject('opgOptionGroup1','opgMyOptGrp') && 增加选项组
frmMyForm.Addobject('shpCircle1','shpMyCircle') && 增加圆形
frmMyForm.Addobject('shpEllipse1','shpMyEllipse') && 增加椭圆形
frmMyForm.Addobject('shpSquare','shpMySquare') && 增加 BOX 形
```

```
frmMyForm.cmdCommand1.Visible =.T. && 取消“退出”命令

frmMyForm.opgOptionGroup1.Buttons(1).Caption = "\<Circle"
frmMyForm.opgOptionGroup1.Buttons(2).Caption = "\<Ellipse"
frmMyForm.opgOptionGroup1.Buttons(3).Caption = "\<Square"
frmMyForm.opgOptionGroup1.SetAll("Width", 100) && 设置选项组宽度
frmMyForm.opgOptionGroup1.Visible = .T. && 使选项组可见
frmMyForm.opgOptionGroup1.Click  && 显示圆形

frmMyForm.SHOW  && 显示表单
READ EVENTS  && 启动事件程序

DEFINE CLASS opgMyOptGrp AS OptionGroup  && 创建一个选项组
    ButtonCount = 3  && 三个选项按扭
    Top = 10
    Left = 10
    Height = 75
    Width = 100

    PROCEDURE Click
        ThisForm.shpCircle1.Visible = .F.  && 隐藏圆形
        ThisForm.shpEllipse1.Visible = .F.  && 隐藏椭圆形
        ThisForm.shpSquare.Visible = .F.  && 隐藏方形

        DO CASE
            CASE ThisForm.opgOptionGroup1.Value = 1
                ThisForm.shpCircle1.Visible = .T. && 显示圆形
            CASE ThisForm.opgOptionGroup1.Value = 2
                ThisForm.shpEllipse1.Visible = .T.  && 显示椭圆形
            CASE ThisForm.opgOptionGroup1.Value = 3
                ThisForm.shpSquare.Visible = .T.  && 显示方形
        ENDCASE
```

```
ENDDEFINE

DEFINE CLASS cmdMyCmndBtn AS Command Button  && 创建命令按扭
    Caption = '\<Quit'  && 给命令按扭增加说明
    Cancel = .T.  && 默认取消命令按扭(Esc键)
    Left = 125  && 命令按扭列
    Top = 210  && 命令按扭行
    Height = 25  && 命令按扭高

    PROCEDURE Click
        CLEAR EVENTS  && 终止事件程序，关闭表单
ENDDEFINE

DEFINE CLASS shpMyCircle AS Shape  && 创建一个圆
    Top = 10
    Left = 200
    Width = 100
    Height = 100
    Curvature = 99
    BackColor = RGB(255,0,0)  && 红色
ENDDEFINE

DEFINE CLASS shpMyEllipse AS Shape  && 创建一个椭圆
    Top = 35
    Left = 200
    Width = 100
    Height = 50
    Curvature = 99
    BackColor = RGB(0,128,0)  && 绿色
ENDDEFINE

DEFINE CLASS shpMySquare AS Shape  && 创建一个方形
    Top = 10
```

```
    Left = 200
    Width = 100
    Height = 100
    Curvature = 0
    BackColor = RGB(0,0,255) && 蓝色
ENDDEFINE
```

请参阅

CREATE CLASS, CREATE FORM, DEFINE CLASS

_SHELL 系统变量

指定一个程序外壳。

语法

_SHELL = *c* Command

说明

在 Visual FoxPro **正执行程序时，** _SHELL **系统变量阻止访问命令窗口。带有要执行程序名的** DO **命令通常存储于** _SHELL **中。**

把 SHELL **配置项放进** Visual FoxPro **配置文件中，可以在启动** Visual FoxPro **时指定执行的命令。**

下面的示例演示了 _SHELL **的典型用法。**

1. 一个名为 MYSTART.PRG 的启动程序用来启动另一个程序 MYAPP.PRG。MYSTART.PRG 把运行 MYAPP.PRG 的命令存储于 _SHELL。Visual FoxPro 显示命令窗口之前，在 _SHELL 中搜索命令。如果 _SHELL 包含一条命令，就执行它，然后 Visual FoxPro 把空串存储于 _SHELL 中。
2. 当成功执行 MYAPP.PRG 中的初始化代码后，启动 MYAPP.PRG 的命令再次存入 _SHELL。Visual FoxPro 并不执行该命令，也不把空串存储于 _SHELL，而且任何对命令窗口的访问都被阻止（当 _SHELL 包含除空串之外的任何命令时，命令窗口的访问都被禁止）。
3. MYAPP.PRG 在结束执行之前，把空串存入 _SHELL 以便恢复对命令窗口的访问。

```
***MYSTART.PRG ***
...
_SHELL = "DO MYAPP.PRG"

*** MYAPP.PRG ***
*** 码的初始化 ***
...
*** 是否已完成码的初始化? ***
_SHELL = "DO MYAPP.PRG" && 阻止通往命令路径
...
*** 清除码 ***
_SHELL = ""
```

请参阅

DO,RUN|!,_STARTUP 系统变量

SHOW GET 命令

重新显示指定到变量、数组元素或字段的控制。包含此命令是为了提供向后兼容性，可用 Refresh 方法代替。

SHOW GETS 命令

重新显示所有控制。包含此命令是为了提供向后兼容性，可用 Refresh 方法代替。

SHOW MENU 命令

显示一个或多个用户自定义菜单栏，但不激活它们。

语法

SHOW MENU MenuBarName1 [, MenuBarName2 ...] | ALL
[PAD MenuTitleName]
[SAVE]

参数描述

MenuBarName1 [, MenuBarName2 ...]
指定要显示的一个或多个菜单栏的名称。

ALL
显示所有当前定义的菜单栏。

PAD *MenuTitleName*
指定在菜单栏中突出显示的菜单标题。

SAVE
保留指定菜单栏的图像，但不激活它们，可以用 CLEAR 命令清除菜单栏的图像。

说明

此命令显示菜单栏，但不能使用。

示例

```
CLEAR
DEFINE MENU mnuExample BAR AT LINE 2
DEFINE PAD padConv OF mnuExample PROMPT '\<Conversions' COLOR SCHEME 3 ;
    KEY ALT+C, ''
DEFINE PAD padCard OF mnuExample PROMPT 'Card \<Info' COLOR SCHEME 3 ;
    KEY ALT+I, ''
SHOW MENU mnuExample
```

请参阅

ACTIVATE MENU, CREATE MENU, DEFINE MENU

Show 方法

显示一个表单，并且确定是模式表单还是无模式表单。

语法

[*FormSet.*] Objec.Show([*nStyle*])

参数描述

nStyle

确定如何显示表单。下列值有效：

值	说明
1	模式表单。只有隐藏或释放模式表单之后，用户的输入（键盘或鼠标）才能被其他表单或菜单接收。在接收其他用户输入之前，程序必须隐藏或释放模式表单（通常是响应某些用户动作）。当应用程序中的模式表单显示时，虽然其他表单被废止，但是其他应用程序并不被废止。
2	（默认值）无模式表单。遇到 Show 方法之后出现的代码时，就执行它们。

如果 *nStyle* 省略，表单按 WindowType 属性指定的样式显示。

说明

Show 方法把表单或表单集的 Visible 属性设置为“真”(.T.)，并使表单成为活动的对象。如果表单的 Visible 属性已经设置为“真”(.T.)，则 Show 方法使它成为活动对象。

如果激活的是表单集，则表单集中最近一个活动表单成为活动表单；如果没有活动表单，则第一个添加到表单集类定义中的表单成为活动表单。

表单集中包含的表单保留 Visible 属性设置。如果表单的 Visible 属性设置为“假”(.F.)，表单集的 Show 方法不显示这个表单。

所有表单集中的表单采取表单集的模式。例如，如果表单集为模式表单集，则所有的表单都为模式表单。

应用于

表单，_SCREEN，工具栏

请参阅

Activate 事件，Hide 方法，Visible 属性，WindowType 属性

SHOW OBJECT 命令

重新显示指定控制。包含此命令是为了提供向后兼容性，可用 Refresh 方法代替。

SHOW POPUP 命令

显示一个或多个用 DEFINE POPUP 定义的菜单，但不激活它们。

语法

```
SHOW POPUP MenuName1 [, MenuName2 ...] | ALL
  [SAVE]
```

参数描述

MenuName1 [, *MenuName2* ...]
要显示的一个或多个菜单。

ALL
显示所有当前定义的菜单。

SAVE
保留指定菜单的图像，但不激活它们。可以用 CLEAR 命令清除菜单的图像。

说明

此命令显示菜单，但不能使用菜单。在显示之前，必须首先用 DEFINE POPUP 创建菜单。

请参阅

ACTIVATE POPUP, DEFINE POPUP

SHOW WINDOW 命令

显示一个或多个用户自定义窗口或 Visual FoxPro 系统窗口，但不激活它们。.

语法

SHOW WINDOW WindowName1 [, WindowName2 ...] | ALL | SCREEN
[IN [WINDOW] WindowName3

[REFRESH]
[TOP | BOTTOM | SAME]
[SAVE]

参数描述

WindowName1 [, WindowName2 ...]
要显示的一个或多个窗口名。

ALL
显示所有用户自定义窗口。

SCREEN
当隐藏 Visual FoxPro 主窗口时，此子句用来显示主窗口。也可以从“窗口”菜单中选择“屏幕”命令显示 Visual FoxPro 主窗口。
单击 Visual FoxPro 主窗口的关闭框，或者使用下列命令：DEACTIVATE WINDOW SCREEN、HIDE WINDOW SCREEN 或 RELEASE WINDOW SCREEN 可以隐藏 Visual FoxPro 主窗口。

IN [WINDOW]WindowName*3*
在 *WindowName 3* 指定的父窗口中显示一个窗口。这个窗口不具有父窗口的特性，父窗口中显示的窗口不能移出父窗口。如果父窗口移动了，子窗口也随之移动。
必须首先用 DEFINE WINDOW 命令创建 *WindowName 3* 指定的父窗口。

IN SCREEN
明确指明在 Visual FoxPro 主窗口（而不是其他窗口）中显示一个窗口。默认

情况下，窗口都放在 Visual FoxPro 主窗口中。

REFRESH

重画一个浏览窗口，这有助于保证在网络上浏览的表是最新版本的表。浏览窗口中表的工作区是选定的。

当网络上其他用户更改了备注字段时，就会刷新备注编辑窗口。SET REFRESH 命令决定备注编辑窗口刷新的间隔时间。有关在网络共享的已打开表中，如何刷新数据的详细内容，请参阅 SET REFRESH 命令。

TOP

把指定窗口放在其他窗口之前。

BOTTOM

把指定窗口放在其他窗口之后。

SAME

把指定窗口放回一系列窗口间的原来位置，此位置是该窗口被废止之前所占据的位置。SAME 只影响先前显示或激活、然后用 DEACTIVATE WINDOW 命令从 Visual FoxPro 主窗口中清除的窗口。

SAVE

释放窗口之后，在 FoxPro 主窗口中或另一窗口中保留窗口的图像。通常，在释放窗口之后，就把它们从 FoxPro 主窗口移去。可以用 CLEAR 命令从 FoxPro 主窗口或另一个窗口中清除窗口图像。

说明

SHOW WINDOW 命令控制在窗口中的显示和从前向后的屏幕放置位置。如果是隐藏窗口或非激活窗口，SHOW WINDOW 命令显示窗口但不激活它；如果正在显示一个

或多个窗口，SHOW WINDOW 命令用来更改窗口从前向后的顺序。也可以显示系统窗口。

在 Visual ForPro 中，可以用 SHOW WINDOW 命令显示 Visual FoxPro 的工具栏，用 HIDE WINDOW 命令从 FoxPro 窗口中移去工具栏。在显示工具栏之前它们必须是活动的。下面的表中列出了 SHOW WINDOW 和 HIDE WINDOW 使用的 Visual FoxPro 工具栏名称。使用时应把工具栏名称用括号括起来。

工具栏名称

调色板	数据库设计器	表单控制
表单设计器	布局	打印预览
查询设计器	报表控制	报表设计器
常用	视图设计器	

要显示系统窗口，可把整个系统窗口名放进引号中。不能用 SHOW WINDOW 命令把输出定向到用户自定义窗口，可以用 ACTIVATE WINDOW 命令把输出定向到用 DEFINE WINDOW 命令创建的用户自定义窗口。

示例

下面的示例创建并显示一个名为 wOutputl 的窗口。 由于 SHOW WINDOW 命令用来显示窗口，因此只有激活该窗口之后．才能把输出定向到窗口中。

```
CLEAR
DEFINE WINDOW wOutput1 FROM 2,1 TO 13,75 TITLE 'Output' ;
    CLOSE FLOAT GROW SHADOW ZOOM
SHOW WINDOW wOutput1
```

请参阅

ACTIVATE WINDOW, DEFINE WINDOW

ShowDoc 事件

当定位到一个 Active Document **上时发生。**

语法

PROCEDURE *Object*.*Show*Doc

说明

当定位到一个 Active Document **上时发生** ShowDoc **事件，可以通过打开这个** Active Document，**或者通过从容器的缓冲或历史记录中返回到这个** Active Document。**与** Init **事件不同，只在** Active Document **完全位于容器中时，才发生** ShowDoc **事件。**

当 Active Document **容器最小化，或者当容器恢复原来大小时，不发生** ShowDoc **事件。**

应用于

ActiveDoc **对象**

请参阅

HideDoc **事件**, Init **事件**

ShowTips 属性

对于指定的表单对象或指定的工具栏对象，确定是否显示“工具提示”。

设计和运行时可用。

语法

Object.ShowTips = *lExpr*

设置

lExpr

决定是否为指定控件显示“工具提示”，ShowTips 属性的设置有：

设置	说明
“真”(.T.)	（对工具栏对象是默认值）当用户把鼠标放在控件上时显示“工具提示”。
“假”(.F.)	（对表单对象是默认值）当用户把鼠标放在控件上时不显示“工具提示”。

说明

可以用 ToolTipText 属性指定在每个“工具提示”中显示的文本。

应用于

表单，_SCREEN，工具栏

请参阅

ToolTipText 属性

ShowWhatsThis 方法

对于具有 WhatsThisHelpID 属性的对象，显示它的“这是什么”帮助主题。

语法

Object.ShowWhatsThis

参数描述

Object

指定对象，显示它的“这是什么”帮助主题。

说明

当按下 F1 键时，自动调用 ShowWhatsThis 方法。如果 WhatsThisHelp 属性设置为“真” (.T.)，则显示 WhatsThisHelpID 属性指定的“这是什么”帮助主题。如果 WhatsThisHelp 属性设置为“假” (.F.)，则显示 HelpContextID 属性指定的帮助主题。

应用于

复选框，组合框，命令按钮，命令按钮组，容器对象，控件对象，编辑框，表单，表格，图像，标签，线条，列表框，OLE 绑定型控件，OLE 容器控件，选项按钮，选项

按钮组，形状，微调，文本框，计时器，工具栏

请参阅

WhatsThisButton **属性**, WhatsThisMode **方法**, WhatsThisHelp **属性**, WhatsThisHelpID **属性**

ShowWindow **属性**

指定一个表单或工具栏是否是顶层表单或是子表单。设计时可用；运行时只读。

应用于：

表单，工具栏

语法

Form.ShowWindow[= nExpr]

参数描述

nExpr

ShowWindow **属性的设置有：**

设置	说明
0	在屏幕中（默认值）。该表单是位于 Visual FoxPro 主窗口中的子表单。
1	在顶层表单中。该表单是活动的顶层表单中的子表单，顶层表单可以是 Visual FoxPro 主窗口或另一个顶层表单。如果想让子表单位于活动的顶层表单中，可以使用这个设置。 当顶层表单是 Visual FoxPro 主窗口时，如果 nExpr 设置为 1，则 Visual FoxPro 自动将 nExpr 重新设置为 0。
2	作为顶层表单。该表单是顶层表单，其中可以放置子表单。注意，不管 WindowType 属性的设置如何，顶层表单总是无模式的。

说明

子表单是包含在另一个表单中的表单。子表单不能被移动到父表单之外；当最小化时，它们显示为父表单中的按钮。如果父表单最小化了，子表单也最小化。

顶层表单是无父表单的独立的无模式表单，用于创建一个 SDI（单文档界面）应用程序，或者作为其他子表单的父表单。顶层表单的级别与其他 Windows 应用程序相同，可以显示在这些 Windows 应用程序的前面或后面。顶层表单也显示在 Windows 的任务栏上。

Desktop 属性决定了一个子表单的行为。如果 Desktop 属性设置为“真” (.T.)，则该子

表单不限制在父表单的范围内，可以移动到 Windows 桌面的任何地方。子表单在显示在 Windows 的任务栏上。

请参阅

ACTIVATE WINDOW，AlwaysOnTop 属性，Desktop 属性

SIGN() 函数

当指定数值表达式的值为正、负或 0 时，分别返回 1、-1 或 0。

语法

SIGN(nExpression)

返值类型

数值型

参数描述

nExpression

指定用 SIGN( ) 函数进行求值的数值表达式。如果求出值是正数，则 SIGN( ) 函数返回 1；如果求出值是负数，返回 -1；如果求出值为 0，则返回 0。

示例

```
STORE 10 TO gnNum1
STORE -10 TO gnNum2
```

```
STORE 0 TO gnZero
CLEAR
? SIGN(gnNum1) && 显示数值 1
? SIGN(gnNum2) && 显示数值 -1
? SIGN(gnZero) && 显示数值 0
```

请参阅

ABS()

SIN() 函数

返回一个角度的正弦值。

语法

SIN(*nExpression*)

参数描述

nExpression

要用 SIN()函数返回正弦值的角度。*nExpression*可以取任何值，并且函数的返回值在 -1 和 1 之间。

*nExpression*以弧度为单位，用 DTOR()函数可以从角度转换为弧度。SIN()函数的小数点显示位数可以用 SET DECIMALS 命令指定。

返值类型：

数值型

示例

```
CLEAR
? SIN(0) &&显示 0.00
? SIN(PI( )/2) &&显示 1.00
? SIN(DTOR(90)) &&显示 1.00
```

请参阅

ACOS( ), COS( ) , DTOR( ) , RTOD( ) , SET DECIMALS

Sizable 属性

指定对象的大小是否可以改变。

语法

Object.Sizable = *lExpr*

设置：

lExpr

Sizable **属性的设置有：**

设置	说明
“真” (.T.)	(默认) 可以调整对象。
“假” (.F.)	不可以调整对象。

说明

设计时可用，运行时可读写。

对于绑定型和非绑定型 OLE 控件，Sizable 属性只影响控件包含的 OLE 对象。如果 Sizable 属性设置为“真” (.T.)，并且激活 OLE 对象，就可以调整 OLE 对象的大小；如果 Autosize 属性设置为“真” (.T.)，而 OLE 对象不活动时，OLE 控件自动调整大小，以适应 OLE 对象新的大小。

注意 要使表单大小可以调整，需要把 BorderStyle 属性设置为 3。

应用于：

OLE 绑定型控件，OLE 容器控件，工具栏

请参阅

AutoSize 属性, BorderStyle 属性

SizeBox 属性

指定表单是否包含“大小”对话框，本属性为以后使用而保留。

语法

object.SizeBox = lExpr

参数描述

lExpr

SizeBox 属性的设置如下：

设置	说明
True (.T.)	默认值。当表单的 BorderStyle 属性设置为默认值 3 时，大小框出现在表单的右下角。
False (.F.)	表单不含大小对话框。

说明

大小框为用户提供一种用鼠标调整表单大小的方法。只有当表单的 BorderStyle 属性采用默认设置值 3 时，大小框才会出现。

应用于

表单

请参阅

BorderStyle 属性, ZoomBox 属性

SIZE POPUP 命令

更改用 DEFINE POPUP 命令创建的菜单大小。

语法

SIZE POPUP *MenuName* TO *nRow1*, *nColumn1* | BY *nRow2*, *nColumn2*

参数描述

MenuName

需要更改大小的菜单名。

TO *nRow1*, *nColumn1*

把菜单更改为指定大小，*nRow1* 和 *nColumn1* 分别指定了菜单右下角的新行和新列的坐标。

BY *nRow2*, *nColumn2*

相对于当前菜单尺寸更改大小，*nRow2* 和 *nColumn2* 指定菜单的行和列的改变值，它们相对于菜单右下角的当前行和当前列坐标。

说明

如果创建了用户自定义菜单，就可以更改它的大小，菜单不必是活动或可见的。

示例

本示例创建了包含带有 .prg 扩展名文件的菜单，并且在关闭前移动、扩大和收缩菜

单。

```
CLEAR
DEFINE POPUP popMovIn FROM 2,2 TO 7, 14 PROMPT FILES LIKE *.PRG ;
    TITLE 'Programs'
ACTIVATE POPUP popMovIn NOWAIT
=CHRSAW(2)
MOVE POPUP popMovIn BY 5,5    && 下移 popup
=CHRSAW(2)
SIZE POPUP popMovIn BY 5,5  && 扩大 popup
=CHRSAW(2)
SIZE POPUP popMovIn BY -5,-5  && 缩小 popup
=CHRSAW(2)
MOVE POPUP popMovIn BY -5,-5  && 上移 popup
=CHRSAW(2)
DEACTIVATE POPUP popMovIn
RELEASE POPUP popMovIn
```

请参阅

ACTIVATE POPUP

SIZE WINDOW 命令

更改用 DEFINE WINDOW **命令创建的窗口大小，或者更改** Visual FoxPro **系统窗口的大小。**

语法

SIZE WINDOW *WindowName* TO *nRow1*, *nColumn1* | BY *nRow2*, *nColumn2*

参数描述

WindowName

需要更改窗口大小的窗口名称。

为了更改系统窗口的大小，需要把整个系统窗口名放在括号里。例如，若想把命令窗口的大小增加一行和一列，使用下列命令：

```
SIZE WINDOW '命令窗口' BY 1,1
```

注意，只能更改命令窗口、调试窗口和跟踪窗口的大小。

TO *nRow1*, *nColumn1*

把窗口更改为指定大小。*nRow1* 和 *nColumn1* 分别指定窗口右下角相对于窗口左上角的新行和新列坐标。

BY *nRow2*, *nColumn2*

相对当前窗口尺寸更改窗口的大小。*nRow2* 和 *nColumn2* 指定窗口的行和列的改变值，它们相对于窗口右下角的当前行坐标和列坐标。

说明

如果创建一个用户自定义窗口，就可以更改它的大小，窗口不必是活动或可见的。

请参阅

ACTIVATE WINDOW, DEFINE WINDOW, ZOOM WINDOW

SKIP 命令

使记录指针在表中向前移动或向后移动。

语法

SKIP
[*nRecords*]
[IN *nWorkArea* | *cTableAlias*]

参数描述

nRecords

指定记录指针需要移动的记录数。

使用不带 *nRecords* 参数的 SKIP 命令将使记录指针走到下一个记录。如果 *nRecords* 为正数，记录指针向文件尾移动 *nRecords* 个记录；如果 *nRecords* 为负数，记录指针将向文件头移动 *nRecords* 个记录。

如果记录指针指向表的最后一个记录，并且执行不带参数的 SKIP 命令时，RECNO( ) 函数返回值比表中记录总数大 1，EOF() 函数返回“真” (.T.)；如果记录指针指向表的第一个记录,并且执行 SKIP-1 命令，则 RECNO( ) 函数返回 1，BOF( ) 函数返回“真” (.T.)。

IN *nWorkArea* | *cTableAlias*

在指定工作区的表中移动记录指针。*nWorkArea* 指定工作区编号，

cTableAlias 指定一个表或工作区的别名。

说明

如果表有一个主控索引名或索引文件，使用 SKIP 命令将使记录指针移动到索引序列决定的记录上。

示例

```
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')
USE customer  && 打开 Customer 表
CLEAR

SKIP 4 IN 'customer'
? RECNO('customer') && 显示数值 5
GO BOTTOM
SKIP -5
? RECNO( )
```

请参阅

GO | GOTO, SET SKIP

SKPBAR() 函数

确定是否可以用 SET SKIP OF 命令启用或废止一个菜单项。

语法

SKPBAR(*cMenuName*, *MenuItemNumber*)

返值类型：

逻辑型

参数描述

cMenuName

包含这个菜单项的菜单名。

MenuItemNumber

要 SKPBAR( )函数返回菜单项状态（启用或废止）的菜单项编号。当用 DEFINE BAR 命令创建菜单项时，就已经指定了菜单项编号。

说明

如果菜单项是废止的，SKPBAR( )函数返回“真” (.T.)；如果菜单项是启用的，则返回“假” (.F.)。

请参阅

DEFINE BAR, SET SKIP OF, SKPPAD( )

SKPPAD() 函数

确定是否可以用 SET SKIP OF 命令启用或废止一个菜单标题。

语法

SKPPAD(*cMenuBarName*, *cMenuTitleName*)

返值类型：

逻辑型

参数描述

cMenuBarName

包含此菜单标题的菜单栏名。

cMenuTitleName

要 SKPPAD( ) 函数返回菜单标题状态（启用或废止）的菜单标题名称。

说明

如果菜单标题是废止的，则 SKPPAD( ) 函数返回“真” (.T.)；如果菜单标题是启用的，则返回“假” (.F.)。

请参阅

DEFINE BAR, SET SKIP OF, SKPBAR()

SORT 命令

对当前选定表进行排序，并将排过序的记录输出到新表中。

语法

```
SORT TO TableName
ON FieldName1 [/A | /D] [/C]
  [, FieldName2 [/A | /D] [/C] ...]
  [ASCENDING | DESCENDING]
  [Scope] [FOR lExpression1] [WHILE lExpression2]
  [FIELDS FieldNameList
  | FIELDS LIKE Skeleton
  | FIELDS EXCEPT Skeleton]
  [NOOPTIMIZE]
```

参数描述

TableName

存放经过排序的记录的新表名，Visual FoxPro 为表取 .DBF 文件扩展名。如果文件不包含扩展名，则自动为它指定 .DBF 扩展名。

ON *FieldName1*

在当前选定的、要排序的表中指定字段，字段的内容和数据类型决定了记录在新表中

的顺序。默认情况是按升序排序，不能对备注或通用字段排序。
下例为下表按 cust-id 字段排序。Customer表已打开并排序过，创建一个新 temp 表。Temp 表中的记录按 cust-id 排序。

```
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'data\testdata')
USE customer  && 打开 Customer 表
CLEAR
LIST FIELDS company, cust_id NEXT 3
SORT TO temp ON cust_id
USE temp
LIST FIELDS company, cust_id NEXT 3
WAIT WINDOW 'Now sorted on CUST_ID' NOWAIT
```

要进一步排序新表，可以包含附加字段名 (*FieldName2，FieldName3*)。第一个字段 *FieldName1* 是主排序字段，第二个字段 *FieldName2* 是第二级排序字段，…，依此类推。

[/A | /D] [/C]

对于排序中包含的每个字段，可以指定排序顺序（升序或降序）。/A 为字段指定了升序。/D 指定了降序，/A 或 /D 可以作用于任何类型的字段。
默认情况下，字符型字段的排序顺序区分大小写。如果在字符型字段名后包含 /C，则忽略大小写。可以把 /C 选项同 /A 或 /D 选项组合起来，例如 /AC 或 /DC。
如下例，创建新表 clients。Orders 表以定货日期升序和运费降序排列。

```
USE orders
```

SORT TO clients ON order_date/A,freight/D

ASCENDING

将所有不带 /D 的字段指定为升序排列。

DESCENDING

将所有不带 /A 的字段指定为降序排列。

如果省略 ASCENDING 或 DESCENDING 参数，则排序顺序默认为升序。

Scope

指定需要排序的记录范围。Scope 子句包括：ALL、NEXT *nRecords*、RECORD *nRecordNumber* 和 REST。

SORT 命令的默认范围是 ALL，即所有记录。

FOR *lExpression1*

在当前表中，指定排序中只包含逻辑条件 *lExpression1* 为“真”(.T.) 的记录。FOR 子句可以有条件地排序记录，筛掉不满足条件的记录。

如果 *lExpression1* 是可优化表达式，Rushmore 会优化这个 SORT ... FOR 命令。为达到最优性能，应在 FOR 字句中使用可优化表达式。

相关信息请参阅《Microsoft Visual FoxPro 6.0 中文版程序员指南》的第十五章。

WHILE *lExpression2*

指定一个条件，在当前表中，只要逻辑表达式 *lExpression2* 的计算值为“真”，则依据此条件，排序中包含这个记录。

FIELDS *FieldNameList*

指定用 SORT 命令创建的新表中要包含的原表中的字段。如果省略 FIELDS

字句，新表中将包括原表中所有字段。

FIELDS LIKE *Skeleton*

在新表中包含那些与字段梗概 *Skeleton* 相匹配的原表字段。

FIELDS EXCEPT *Skeleton*

在新表中包含那些不与字段梗概 *Skeleton* 相匹配的原表字段。

字段梗概 *Skeleton* 支持通配符。

例如，列出新表中所有以字母 A 和 P 开头的字段使用如下命令：

```
SORT TO mytable ON myfield FIELDS LIKE A*,P*
```

LIKE 子句可与 EXCEPT 子句同时使用：

```
SORT TO mytable ON myfield FIELDS LIKE A*,P* EXCEPT PARTNO*
```

NOOPTIMIZE

关闭 SORT 命令的 Rushmore 优化。

说明

当前表中的一个或多个指定字段决定了记录在新表中出现的顺序。

重要提示 要确保有足够的磁盘空间保存新表，以及存储在排序过程中创建的临时工作文件，排序所需的磁盘空间可能是原表的三倍。可以用 DISKSPACE( ) 和 SYS(2020) 函数确定可用磁盘空间大小。如果在排序过程中，磁盘空间不足，Visual FoxPro 会显示错误信息，并删除临时工作文件。

包含数字和空格的字符型字段可能不会按照所希望的那样排序。这是因为数值型字段从右向左填充，空格位于左边；而字符型字段是从左向右填充，空格位于右边。

例如，表的两个记录包含字符型字段：一个是 1724，另一个是 18，而表按该字段以升

序排序，则包含 1724 的记录会出现在包含 18 的记录的前面。这是因为 Visual FoxPro 从左向右读取字符型字段中的字符，由于 17（1724）比 18（18）小，则 1724 出现在前面。为避免这类问题，应在位数较小的数字前加零（0018），或者使用数值型字段。

请参阅

COPY FILE, DISKSPACE(), INDEX, SYS(2020)

Sorted 属性

在组合框和列表框中，指定列表部分的各项是否按字母顺序排序。设计和运行时可用。

应用于：

组合框，列表框

语法

[*Form*.] Control.Sorted[= *lExpr*]

参数描述

lExpr

Sorted 属性的设置有：

设置	说明
“真” (.T.)	列表项按字母顺序排序（区分大小写）。为保持字母顺序，Visual FoxPro 进行几乎所有必要的字符串处理，包括从列表中添加项或移去项时对索引号做必要的更改。
“假” (.F.)	（默认）列表项不按字母顺序排序。

说明

只有 RowSourceType 属性设置为零时，Sorted 属性才可用。

注意，当列表项按字母排序时，将 Sorted 属性设置为“假” (.F.)，不会将列表项恢复到原来顺序。为了将列表项恢复到原来顺序，可以将控件的 RowSource 属性设置为自己，如下所示：

```
MyForm.MyCombo1.Rowsource = MyForm.MyCombo1.Rowsource
```

请参阅

AddItem 方法, List 属性, ListCount 属性, ListItemID 属性, RemoveItem 方法, RowSource 属性, RowSourceType 属性

SOUNDEX() 函数

返回指定字符表达式的发音。

语法

SOUNDEX(cExpression)

返值类型：

字符型

参数描述

cExpression

要使用 SOUNDEX( ) 函数求值的字符表达式。

说明

SOUNDEX( ) 函数返回一个四字符的字符串。它通过比较两个字符表达式的返回值，可以确定两个字符表达式是否是语音相近，也即发音相似。在表中搜索重复记录时，可能会用到这个函数。SOUNDEX( ) 函数不区分大小写，并且一般不考虑元音。

示例

```
CLEAR
? SOUNDEX('ArtM') = SOUNDEX('AtM') &&显示.F.
? SOUNDEX('Computer') &&显示 C513
```

请参阅

DIFFERENCE()

SPACE() 函数

返回由指定数目空格构成的字符串。

语法

SPACE(nSpaces)

返值类型：

字符型

参数描述

nSpaces

SPACE()**函数返回的空格数目。在** Visual FoxPro **中，** *nSpaces* **最大值的唯一限制是内存容量。**

请参阅

PADC() | PADL() | PADR() , REPLICATE()

Sparse 属性

指定 CurentControl**属性是影响列对象中的全部单元，还是仅影响列对象中的活动单元。**

设计时可用可写。

语法

Column.Sparse [= *lExpr*]

参数描述

lExpr

Sparse **属性的设置是：**

设置	**说明**
真 (.T.)	**（默认值）只有列对象的活动单元使用** CurrentControl **属性设置接收和显示数据，其他的单元使用默认的文本框。**
假 (.F.)	**列对象的所有单元使用** CurrentControl **属性显示数据，活动单元使用** CurrentControl **属性接收数据，**

应用于：

列

请参阅

Bound **属性**, CurrentControl **属性**, DynamicCurrentControl **属性**

SpecialEffect 属性

指定控件的不同样式选项。设计和运行时可用。

应用于

复选框，组合框，命令按钮，命令组，容器对象，控件对象，编辑框，列表框，选项按钮，选项组，页框，形状，微调，文本框

语法

[*Form*.] Control SpecialEffect = *Expr*

参数描述

Expr

对于页框控件，SpecialEffect 属性设置有：

设置	说明
0	凸起 (除容器对象之外所有控件和对象的默认值)
1	凹下
2	平面 (仅是容器对象的默认值)

注意 对于页框控件来说，只有 Tabs 属性为假 (.F.) 时，SpecialEffect 属性才可用。同样，有当 *Expr* 设置成 2 (平面) 时，页框控件的 BorderColor 属性才可用。

对于所有其它的控件，SpecialEffect 属性设置是：

设置	说明
0	(除了容器对象外的所有控件和对象的默认值)三维。提升控件的边框以模拟一个三维外观。
1	三维
2	平面

注意 如果 Height 属性设置值太小，那么“三维”设置不起作用。对于 Visual FoxPro for Windows，命令按钮忽略 SpecialEffect。

请参阅

BackColor, ForeColor 属性, SelectedItemBackColor, SelectedItemForeColor 属性

_SPELLCHK 系统变量

为 Visual FoxPro 文本编辑器指定一个拼写检查程序。

语法

_SPELLCHK = *ProgramName*

参数描述

ProgramName

指定拼写检查程序。如果该拼写检查程序不在当前默认目录下，那么程序名中应包含路径。

也可以在 Visual FoxPro 配置文件中指定拼写检查器程序，只需加入下面一行：

```
_SPELLCHK = ProgramName
```

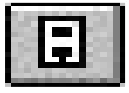
说明

默认情况下，_SPELLCHK 使用 SPELLCHK.APP 作为拼写检查程序。如果将 SPELLCK.APP 改名或移动到另一个目录下，则在 _SPELLCHK 中存入新的拼写检查程序名称和路径。

请参阅

MODIFY COMMAND, MODIFY FILE, MODIFY MEMO

Spinner 控件



创建一个微调控件。

语法

Spinner

说明

通过单击微调控件的上箭头或下箭头，或者在微调框内键入一个数值，可以实现微调控件在一个数值范围内进行选择。

KeyBoardHighValue 属性指定从键盘输入微调框的最大值，SpinnerHighValue 属性指定通过单击微调按钮输入的最大值。

KeyBoardLowValue 属性指定从键盘输入微调框的最小值，SpinnerLowValue 属性指定通过单击微调按钮输入的最小值。

有关创建微调控件的详细内容，请参阅帮助中的“表单设计器”，与《Microsoft Visual FoxPro 6.0 中文版中文版程序员指南》的第十章“使用控件”。

属性

Alignment	Application	BackColor
BaseClass	BorderColor	BorderStyle
Class	ClassLibrary	ColorScheme
ColorSource	Comment	ControlSource
DisabledBackColor	DisabledForeColor	DragIcon
DragMode	Enabled	FontBold
FontCondense	FontExtend	FontItalic
FontName	FontOutline	FontShadow

续 表

FontSize	FontStrikeThru	FontUnderline
ForeColor	Format	Height
HelpContextID	HideSelection	Increment
InputMask	KeyboardHighValue	KeyboardLowValue
Left	Margin	MouseIcon
MousePointer	Name	NullDisplay
OLEDragMode	OLEDragPicture	OLEDropEffects
OLEDropHasData	OLEDropMode	OLEDropTextInsertion
Parent	ParentClass	ReadOnly
RightToLeft	SelectedBackColor	SelectedForeColor
SelectOnEntry	SelLength	SelStart
SelText	SpecialEffect	SpinnerHighValue
SpinnerLowValue	StatusBarText	TabIndex
TabStop	Tag	TerminateRead
Text	ToolTipText	Top
Value	Visible	WhatsThisHelpID
Width		

事 件

Click
DblClick
Destroy
DownClick
DragDrop
DragOver
Error
ErrorMessage
GotFocus
Init
InteractiveChange
KeyPress
LostFocus
Message
MiddleClick
MouseDown
MouseMove
MouseUp
MouseWheel
OLECompleteDrag
OLEDragDrop
OLEDragOver
OLEGiveFeedBack
OLESetData
OLEStartDrag
ProgrammaticChange
RangeHigh
RangeLow
RightClick
UIEnable
UpClick
Valid
When

方 法

AddProperty
CloneObject
Drag
Move
OLEDrag
ReadExpression
ReadMethod
Refresh
ResetToDefault
SaveAsClass
SetFocus
ShowWhatsThis
WriteExpression
WriteMethod
Zorder

请 参 阅

CREATE CLASS, CREATE FORM, DEFINE CLASS

SpinnerHighValue, SpinnerLowValue 属性

指定单击上和下箭头时，微调控件所允许的最大值或最小值。设计和运行时可用。

语法

```
Spinner.SpinnerHighValue[ = nHigh]
Spinner.SpinnerLowValue[ = nLow]
```

参数描述

nHigh

在微调控件中用上箭头所能输入的最大值。

NLow

在微调控件中用下箭头所能输入的最小值。

应用于：

微调

请参阅

KeyboardHighValue, KeyboardLowValue 属性

SplitBar 属性

指定在表格控件中是否显示分隔栏。设计时可用；运行时只读。

语法

Grid.SplitBar[= lExpr]

参数描述

lExpr

取下列一个值：

lExpr	说明
“真” (.T.)	（默认值）在表格控件中显示分隔栏。允许将表格拆分为两个独立的窗格。
“假” (.F.)	在表格控件中不显示分隔栏。不允许将表格拆分为两个独立的窗格。

说明

SplitBar 属性优先于 the Partition 属性。

应用于

表格控件

请参阅

GridLines 属性, Partition 属性

SQL 命令概览

Visual FoxPro 支持 Structured Query Language (SQL) 命令。Visual FoxPro 的 SQL 命令利用 Rushmore 技术优化性能，并且一个 SQL 命令可以用来替换多个 Visual FoxPro 命令。

Visual FoxPro 支持下列 SQL 命令：

SELECT – SQL

指定查询的条件，并且执行查询。Visual FoxPro 解释该查询，并且从表中获取指定的数据。该 SELECT 命令象其他 Visual FoxPro 命令一样，已经是 Visual FoxPro 固有的了。您可以在下列地方创建一个 SELECT 命令查询：

- 在“命令”窗口中。
- 在 Visual FoxPro 程序中(同其他 Visual FoxPro 命令一样)。
- 在“查询设计器”中。

ALTER TABLE – SQL。

更改一个已有的表。您可以更改表中每个字段的名称、类型、精度、范围、对 null 值的支持以及参照完整性规则。

CREATE CURSOR – SQL

创建一个临时表。可以定义临时表中每个字段的名称、类型、精度、范围、对 null 值的支持，以及参照完整性规则。可以从命令本身也可以从一个数组获得这些定义。

CREATE TABLE – SQL

创建一个表。可以定义新表中每个字段的名称、类型、精度、范围、对 null 值的支持，以及参照完整性规则。可以从命令本身也可以从一个数组获得这些定义。

DELETE – SQL

利用 SQL 语法将表中的记录标记为删除。

INSERT – SQL

在一个已有表的末尾追加一个新记录。该新记录包含了 INSERT 命令中列出的或一个数组中的数据。

UPDATE – SQL

更新一个表中的记录。可以根据一个 SELECT – SQL 语句的结果更新记录。

请参阅

CREATECURSOR-SQL, CREATE QUERY, CREATE TABLE – SQL, DELETE – SQL, INSERT – SQL, MODIFY QUERY, SELECT – SQL, UPDATE – SQL

SQLCANCEL() 函数

请求取消一条正在执行的 SQL 语句。

语法

SQLCANCEL(*nConnectionHandle*)

返值类型：

数值型

参数描述

nConnectionHandle

指定活动的连接句柄，该连接句柄的 SQL 语句将被取消。

说明

如果成功取消了 SQL 语句，SQLCANCEL( )函数返回 1；如果发生连接级错误，返回 -1；如果发生环境级错误，返回 -2。

在异步方式下，SQLCANCEL( )取消执行 SQLCOLUMNS( )、SQLEXEC( )、SQLMORERESULTS( )和 SQLTABLES( )命令。可使用 SQLSETPROP( )来建立异步方式。

示例

下面的示例假定已经成功发出 SQLCONNECT( )，并且返回值存入名为 gnConnHandle

的变量。

SQLEXEC()将 SQL 语句送入数据源，并且将返回结果送入临时表，最后调用 SQLCANCEL()函数来终止查询。

```
= SQLSETPROP(gnConnHandle, 'asynchronous', .T.) && 停止 SQLEXEC()
= SQLEXEC(gnConnHandle, 'SELECT * FROM authors')
= SQLCANCEL(gnConnHandle) && 错误的 SELECT 语句，取消
```

请参阅

AERROR(), SQLCOLUMNS(), SQLEXEC(), SQLMORERESULTS(), SQLSETPROP(), SQLTABLES()

SQLCOLUMNS()函数

把指定数据源表的列名和关于每列的信息存储到一个 Visual FoxPro 临时表中。

语法

```
SQLCOLUMNS(nConnectionHandle,TableName
    [,"FOXPRO" | "NATIVE"] [, CursorName])
```

返值类型：

数值型

参数描述

nConnectionHandle

活动的连接句柄。

TableName

指定的远程表名称，从该远程表中返回列名。TableName 可包含通配符？和*。？可代替任意单个字符，*可代替任意个字符。

FOXPRO | NATIVE

指定在结果集合中列内容的格式，一定要用引号括起 FOXPRO 或 NATIVE。NATIVE 格式选项使用数据源格式存储表的列内容，FOXPRO 格式选项使用 Visual FoxPro 的表或临时表格式存储列内容，这些表或临时表在将数据源表导入 Visual FoxPro 时创建。如果省略 FOXPRO 或 NATIVE 参数，格式选项的缺省值是 FOXPRO。

下表显示了结果集合中的 FOXPRO 的列格式：

列名	说明
Field_name	列名
Field_type	列数据类型
Field_len	列长度
Field_dec	小数位数

下表显示结果集合中具有 NATIVE 格式的列。在 NATIVE 格式中，依据数据源的不同，下表中没有列出的一些附加列可能会包含在结果集合中。

列名	说明
Table_qualifier	表限定符标识
Table_owner	表拥有者标识
Table_name	表标识
Column_name	列标识
Data_type	列数据类型
Type_name	
Precision	列的精度
Length	数据的传送大小
Scale	列的宽度
Radix	数值类型的基准
Nullable	是否支持 null 值
说明	列的说明

如果 *TableName* 指定的表不存在，同时格式设置成 NATIVE，SQLCOLUMNS( )返回真 (.T.)，并创建一个空的表或临时表；如果 *TableName* 指定的表不存在，并且格式设置成 FOXPRO，SQLCOLUMNS( ) 返回假 (.F.)。

CursorName

为结果集合指定 Visual FoxPro 临时表的名称。如果不指定临时表名，Visual FoxPro 使用默认名 SQLRESULT。

说明

如果成功创建了临时表，SQLCOLUMNS()返回 1；如果 SQLCOLUMNS()仍在执行，返回 0；如果发生连接级错误，返回 -1；如果发生环境级错误，返回 -2。SQLCOLUMNS()是四个既可同步执行也可异步执行的函数之一，SQLSETPROP()的异步设置决定这些函数是同步执行还是异步执行。异步方式下，必须重复调用 SQLCOLUMNS()，直到返回一个非“假”(.F.)(仍在执行中)值为止。

示例

下面的示例假定已经成功地发出 SQLCONNECT()，并且其返回值存入名为 gnConnHandle 的变量中。SQLCOLUMNS()用来创建一个名为 MyCursor 的临时表，该临时表包含 authors 表中列出的内容。

```
= SQLCOLUMNS(gnConnHandle, 'authors', 'FOXPRO', 'MyCursor')
```

请参阅

AERROR()，SQLGETPROP()，SQLSETPROP()，SQLTABLES()

SQLCOMMIT() 函数

提交一个事务。

语法

SQLCOMMIT(*nConnectionHandle*)

返值类型

数值型

参数描述

nConnectionHandle

SQLCONNECT( )函数返回的、指向数据源的连接句柄。

说明

可使用 SQLCOMMIT( )提交一个事务，如果成功提交了事务，SQLCOMMIT( )返回 1；否则，返回 -1。如果 SQLCOMMIT( )返回 -1，可使用 AERROR( )函数确定该事务为什么不能提交。

如果人工事务有效（用 SQLSETPROP( )将 Transactions 属性设置成“人工”），就可以将多个更新送入远程表，并由 SQLCOMMIT( )提交所有的更新。

更新可以由 SQLROLLBACK( )函数回滚。

示例

下面的示例假定已经成功发出了 SQLCONNECT( )，并且其返回值存入名为 gnConnHandle 的变量中。示例中用 SQLSETPROP( )将 Transactions 属性设置成 2（人工），以允许使用 SQLCOMMIT( )和 SQLROLLBACK( )。

authors 表由 SQLEXEC( )修改，对表的更新由 SQLCOMMIT( )提交。

```
= SQLSETPROP(gnConnHandle, 'Transactions', 2) &&人工事务
= SQLEXEC(gnConnHandle, "INSERT INTO authors (au_id, au_lname);
   VALUES ('aupoe', 'Poe')") &&修改 authors 表
= SQLCOMMIT(gnConnHandle) &&提交更改
```

请参阅

AERROR( ), BEGIN TRANSACTION, END TRANSACTION,
SQLCONNECT( ) , SQLROLLBACK( ) , SQLSETPROP( )

SQLCONNECT() 函数

建立一个指向数据源的连接。

语法

SQLCONNECT([*DataSourceName*, *cUserID*, *cPassword* | *cConnectionName*])

返值类型

数值型

参数描述

DataSourceName

数据源的名称，该名称和 ODBC.INI **文件中的定义相同。**

cUserID

向数据源注册的用户标识。

cPassword

数据源的密码。

cConnectionName

用 CREATE CONNECTION **创建的命名连接。**

说明

如果成功地连接到数据源上，SQLCONNECT()函数返回一个正的、非零的句柄。应该将这个句柄存入变量中，在随后需要连接句柄的函数调用中，就能使用该变量。如果不能连接，SQLCONNECT()返回-2。

如果不带任何附加参数发出SQLCONNECT()，那么将显示选择连接或数据源对话框，供您选择数据源。

注意 为了支持Microsoft Transaction Server的SQL pass-through功能，必须废止ODBC登录对话框。使用SQLSETPROP(cConnectionHandle, 'DispLogin', 3)可以废止ODBC登录对话框（cConnectionHandle是SQLCONNECT返回的连接句柄）。也可以在连接设计器中废止ODBC登录对话框。

示例

下面的示例假设有名称为MyFoxSQLNT的ODBC数据源，并且此数据源的用户ID是“sa.”。发出SQLCONNECT()命令，并将返回值保存到变量gnConnHandle中。

如果成功地连接到数据源中，则SQLCONNECT()返回一个正值，显示对话框，并且用SQLDISCONNECT()从数据源中断开连接。

如果不能连接到数据源中，则SQLCONNECT()返回一个负值，并显示信息。

```
STORE SQLCONNECT('MyFoxSQLNT', 'sa') TO gnConnHandle
IF gnConnHandle <= 0
    = MESSAGEBOX('Cannot make connection', 16, 'SQL Connect Error')
ELSE
    = MESSAGEBOX('Connection made', 48, 'SQL Connect Message)
    = SQLDISCONNECT(gnConnHandle)
ENDIF
```

请参阅

AERROR( ), CREATE CONNECTION, SQLDISCONNECT( ), SQLGETPROP( ), SQLEXEC( ), SQLSETPROP( ), SQLSTRINGCONNECT( )

SQLDISCONNECT( ) 函数

终止与数据源的连接。

语法

SQLDISCONNECT(*nConnectionHandle*)

返值类型

数值型

参数描述

nConnectionHandle

由 SQLCONNECT( ) 返回、并指向数据源的连接句柄。*nConnectionHandle* 为 0 将终止所有活动的连接。

说明

如果成功终止了连接，SQLDISCONNECT( )返回 1；如果发生连接级错误，返回-1；如果发生环境级错误，返回 -2。

SQLDISCONNECT( )终止一个指向数据源的连接，在建立连接时，必须提供SQLCONNECT( )返回的连接句柄。

注意 在异步函数序列中或事务处理期间，如果执行 SQLDISCONNECT( )函数，SQLDISCONNECT( )将产生错误。

示例

下面的示例假定一个名为 MyFoxSQLNT 的 ODBC 数据可用，该数据源的用户标识是“sa”，发出 SQLCONNECT( )，其返回值存入名为 gnConnHandle 的变量。如果成功地连接到数据源上，SQLCONNECT( )返回一个正数，并显示一个对话框，再发出 SQLDISCONNECT( )解除和数据库的连接。如果不能连接到数据源上，SQLCONNECT( )返回一个负数，并显示一条信息。

```
STORE SQLCONNECT('MyFoxSQLNT', 'sa') TO gnConnHandle
IF gnConnHandle <= 0
    = MESSAGEBOX('Cannot make connection', 16, 'SQL Connect Error')
ELSE
    = MESSAGEBOX('Connection made', 48, 'SQL Connect Message)
    = SQLDISCONNECT(gnConnHandle)
ENDIF
```

请参阅

AERROR( ), SQLCONNECT( ), SQLSTRINGCONNECT( )

SQLEXEC() 函数

将一条 SQL 语句送入数据源中处理。

语法

SQLEXEC(*nConnectionHandle*, *cSQL*Command, [*CursorName*])

返值类型

数值型

参数描述

nConnectionHandle

由 SQLCONNECT() 返回的指向数据源的连接句柄。

*cSQL*Command

送入数据源的 SQL 语句。

SQL 语句中可以包含一个参数化的 WHERE 子句，该子句创建一个参数化的视图。所有 WHERE 子句中的参数必须在发出 SQLEXEC() 之前定义。例如，如果参数是变量，那么该变量必须在 SQLEXEC() 发出之前创建并初始化。

CursorName

Visual FoxPro 临时表的名称，结果集合将送入该临时表中。如果不包含临时表名，Visual FoxPro 使用默认名 SQLRESULT。对于多个结果集合，通过在

第一个临时表的名称后追加一个递增的数值构成新临时表名。

说明

如果存在多个结果集合，SQLEXEC()返回结果集合的数目。如果 SQLEXEC()仍在执行，SQLEXEC()返回 0；当 SQLEXEC()结束时，返回 1；如果发生连接级错误，SQLEXEC()返回 -1。

如果 SQL 语句产生一个结果集合，那么 SQLEXEC()将该结果集合存入指定的 Visual FoxPro 临时表中；如果 SQL 语句产生两个或多个结果集合，同时 SQLSETPROP()设置为 1(批处理方式)，可以通过 SQLSETPROP()设置 BatchMode 选项为 0，并且在每次调用 SQLMORERESULTS()时更改临时表的名称来命名每个结果集合。

SQLEXEC()是既能同步执行又能异步执行的四个函数之一。SQLSETPROP()异步设置决定它们是同步执行还是异步执行。在异步方式下，必须重复调用 SQLEXEC()直到返回一个非零值(0 表示仍在执行)。

示例

下面的示例假定已经成功地发出 SQLCONNECT()并且其返回值存入名为 gnConnHandle 的变量。SQLEXEC()用来执行一个查询，该查询将 authors 表中的全部内容返回到名为 MyCursor 的临时表中。

```
= SQLSETPROP(gnConnHandle, 'asynchronous', .F.)
= SQLEXEC(gnConnHandle, 'SELECT * FROM authors', 'MyCursor')
```

请参阅

AERROR()，SQLCANCEL()，SQLGETPROP()，SQLMORERESULTS()，SQLPREPARE()，SQLSETPROP()

SQLGETPROP() 函数

返回一个活动连接的当前设置或默认设置。

语法

SQLGETPROP(*nConnectionHandle*, *cSetting*)

返值类型：

字符型、数值型或逻辑型

参数描述

nConnectionHandle

由 SQLCONNECT() 返回的、指向数据源的连接句柄。如果 *nConnectionHandle* 为 0，SQLGETPROP() 返回环境设置。

cSetting

指定具体的设置，有关各设置的详细内容，请参阅 SQLSETPROP()。

说明

如果发生连接级的错误，返回 -1；如果发生环境级的错误，返回 -2。

示例

下面的示例用 SQLCONNECT() 显示“连接”对话框。选择一个数据源，则通过 SQLGETPROP() 和 DataSource c 设置显示结果。

```
* Clear environment
```

```
CLOSE ALL
CLEAR ALL
CLEAR

*显示已选择的联系和数据源对话框
* to choose a connection
nHandle=SQLCONNECT()

*检测连接，报告结果
IF nHandle > 0
    cSource= SQLGETPROP(nHandle, "datasource")
    =MESSAGEBOX("Current Data Source = "+cSource,0,"Connection Results")
ELSE
    =MESSAGEBOX("Connection Error = " + ;
    ALLTRIM(STR(nHandle)),0,"Connection Results")
ENDIF
=SQLDISCONNECT(nHandle)
```

请参阅

AERROR( ), SQLCONNECT( ), SQLSETPROP( )

SQLMORERESULTS() 函数

如果存在多个结果集合，则将另一个结果集合复制到 Visual FoxPro 临时表中。

语法

SQLMORERESULTS(*nConnectionHandle*)

返值类型：

数值型

参数描述

nConnectionHandle

由 SQLCONNECT( ) 返回的、指向数据源的连接句柄。

说明

在非批处理方式下，SQLMORERESULTS( ) 决定 SQLEXEC( ) 执行语句是否产生多个可用的结果集合。如果有多个结果集合可用，将依次将集合复制到 Visual FoxPro 临时表中。

如果仍在执行 SQL 语句，SQLMORERESULTS( ) 返回 0；如果终止执行 SQL 语句，返回 1；如果没有找到更多数据，返回 2。在非批处理方式下，应该在每次成功调用 SQLEXEC( ) 之后调用 SQLMORERESULTS( )，直到 SQLMORERESULTS( ) 返回 2 (没有找到更多的数据)。SQLSETPROP( ) 批处理设置决定 SQLEXEC( ) 以批处理方式或非批处理方式执行 SQL 语句。

如果发生连接级的错误，SQLMORERESULTS( ) 返回 -1；如果发生环境级的错误，返回 -2。

SQLMORERESULTS( ) 函数是既可同步执行也可异步执行的四个函数之一，SQLSETPROP( ) 的异步设置决定这些函数以同步方式执行还是以异步方式执行。在异步方式下，必须重复调用 SQLMORERESULTS( )，直到返回一个非零值 (0 表示仍在执行中)。

示例

下面的示例假定已经成功执行了 SQLCONNECT( )，并且将返回值存入名为 gnConnHandle 的变量中。SQLSETPROP( ) 将 BatchMode 属性设置为“假”(.F.)，这样就可以检索单个的结果集合。

SQLMORERESULTS( ) 执行了两次，创建两个包含 SQLEXEC( ) 查询结果的临时表。

SET 可用来显示“查看”窗口，并查看 SQLEXEC( ) 创建的临时表。

```
= SQLSETPROP(gnConnHandle, 'BatchMode', .F.) &&单个的结果集合
= SQLEXEC(gnConnHandle, 'SELECT * FROM authors;
    SELECT * FROM titles')
= SQLMORERES(gnConnHandle) && 第一个结果集合
= SQLMORERES(gnConnHandle) && 第二个结果集合
```

请参阅

AERROR( ), SQLCANCEL( ), SQLCONNECT( ), SQLEXEC( ), SQLGETPROP( ), SQLSETPROP( )

SQLPREPARE() 函数

在使用 SQLEXEC( ) 执行远程数据操作前，可使用本函数使远程数据为将要执行的命令做好准备。

语法

SQLPREPARE(*nConnectionHandle*, *cSQLCommand*, [*CursorName*])

返值类型：

数值型

参数说明

nConnectionHandle

指定连接句柄 (connection handle)，**这个句柄通常是由** SQLCONNECT( ) **函数返回的**。

*cSQL*Command

要传递给远程数据源，并由远程数据源执行的 SQL **语句**。

在这个 SQL **语句中的** WHERE **子句中可以包含参数，但所有** WHERE **子句中的参数都必须在** SQLPREPARE( ) **执行前事先定义。例如，如果您在** WHERE **子句中包含有一个变量，那么这个变量必须在** SQLPREPARE( ) **执行前已被初始化**。

CursorName

指定一个 Visual FoxPro **临时表。**SQL **语句执行后返回的查询结果将被保存在这个表中。如果您不指定表名，**Visual FoxPro **将用** SQLRESULT **作为默认表明**。

如果远程执行后的结果需要多个临时表，临时表名将由您指定的表名结尾加递增的数字编号组成。

说明

SQLPREPARE( ) **将** SQL **语句传送到数据源，在那里，**SQL **语句被事先编译。这样，**

当您执行 SQLEXEC( ) 函数时，速度会快得多。另外，如果您事先执行了 SQLPREPARE( )，那么在其后的 SQLEXEC( ) 所使用的参数中只须指定连接句柄 *nConnection*Handle 即可。

示例

```
gcAuthor = 'Smith'
= SQLPREPARE(gnConnHandle, 'SELECT * FROM authors; WHERE au_lname = ?gcAuthor')
= SQLEXEC(gnConnHandle)

gcAuthor = 'Jones'
= SQLEXEC(gnConnHandle)
```

请参阅

SQLCONNECT( ), SQLEXEC( )

SQLROLLBACK( ) 函数

取消当前事务处理期间所做的任何更改。

语法

SQLROLLBACK(*nConnectionHandle*)

返值类型

数值型

参数描述

nConnectionHandle

由 SQLCONNECT( ) 返回的、指向数据源的连接句柄。

说明

如果事务回滚成功，SQLROLLBACK( ) 返回 1；否则，返回 -1。如果 SQLROLLBACK( ) 返回 -1，可以使用 AERROR( ) 确定该事务不能回滚的原因。

如果人工事务处理有效（SQLSETPROP( ) 的 Transaction 属性设置为“人工”），就可以将多个更新发送到远程表中，这些更新可以用 SQLROLLBACK( ) 全部回滚。可以用 SQLCOMMIT( ) 提交更新。

示例

下面的示例假定已经成功发出 SQLCONNECT( )，并且其返回值存入一个名为 gnConnHandle 的变量。SQLSETPROP( ) 将 Transaction 属性设置成 2 (人工)，以允许使用 SQLCOMMIT( ) 和 SQLROLLBACK( )。

示例中用 SQLEXEC( ) 修改了表 authors，并用 SQLROLLBACK( ) 取消对该表的更改。

```
= SQLSETPROP(gnConnHandle, 'Transactions', 2) && 人工
= SQLEXEC(gnConnHandle, "INSERT INTO authors (au_id, au_lname);
    VALUES ('aupoe', 'Poe')")
= SQLROLLBACK(gnConnHandle)
```

请参阅

AERROR( ), SQLCOMMIT( ), SQLCONNECT( ), SQLSETPROP( )

SQLSETPROP() 函数

指定一个活动连接的设置。

语法

SQLSETPROP (*nConnectionHandle*, *cSetting* [, *eExpression*])

参数描述

nConnectionHandle

由 SQLCONNECT() 返回的、指向数据源的连接句柄。

cSetting

指定设置，下表列出了 *cSetting* 的值：

设置	说明
Asynchronous	指定结果集合是同步返回（默认值为“假”(.F.)），还是异步返回(“真”(.T.)）。可读写。
BatchMode	指定 SQLEXEC( )是一次返回全部结果集合（默认值为“真”(.T.)），还是用 SQLMORERESULTS( )单个返回结果集合（“假”(.F.)）。可读写。
ConnectBusy	如果共享连接繁忙，为“真”(.T.)；否则为“假”(.F.)。只读。
ConnectString	注册的连接字符串。只读。
ConnectTimeOut	指定返回一个连接超时错误之前的等待时间（以秒计算）。如果指定为 0（默认值），将无限期等待,而不会返回超时错误。ConnectTimeOut 可以是以 0 到 600。可读写。
DataSource	和 ODBC.INI 文件中相同的数据源文件名。可读写。

续表

DispLogin	包含一个数值，决定什么时候显示“ODBC 注册”对话框。DispLogin 可以设定为下列值： 1 或 DB_PROMPTCOMPLETE（在 FOXPRO.H 中定义）。1 是默认值。 2 或 DB_PROMPTALWAYS（在 FOXPRO.H 中定义）。 3 或 DB_PROMPTNEVER（在 FOXPRO.H 中定义）。 如果指定为 1 或 DB_PROMPTCOMPLETE，则当需要的信息未得到时，总是显示“ODBC 注册”对话框。 如果指定为 2 或 DB_PROMPTALWAYS，则总是显示“ODBC 注册”对话框，允许连接前更改设置。 如果指定为 3 或 DB_PROMPTNEVER，则不显示“ODBC 注册”对话框并且当所需要的注册内容不可用时，Visual FoxPro 产生错误。可读写。
DispWarnings	DispWarnings 指定要显示错误信息（“真”(.T.)），还是不显示错误信息（“假”(.F.)，默认值）。可读写。
IdleTimeout	以秒计算的空闲超时间隔。在指定的时间间隔之后，活动的连接成为不活动的，默认值是 0（无限期等待）。可读写。
ODBChdbc	内部 ODBC 连接句柄，外部库文件（FLL 文件）在调用 ODBC 时使用该句柄。只读。
ODBChstmt	内部 ODBC 语句句柄，外部库文件（FLL 文件）在调用 ODBC 时使用该句柄。只读。
PacketSize	连接所用的网络包的大小，调整该值可以改善性能。默认值是 4096 字节 (4K)。可读写。
Password	连接密码。只读。

续表

QueryTimeOut	指定在返回一般超时错误之前等待的时间（以秒计算）。如果指定为 0，将无限期等待而不会返回超时错误。QueryTimeOut 可以是 0 到 600，默认值是 15。可读写。
Transactions	包含一个数值，该数值决定连接如何管理远程表上的事务处理。Transactions 可以设定为下列值： 1 或 DB_TRANSAUTO（在 FOXPRO.H 中定义） .1 是默认值，自动处理远程表的事务。 2 或 DB_TRANSMANUL（在 FOXPRO.H 中定义）。通过 SQLCOMMIT( ) 和 SQLROLLBACK( ). 人工处理事务。可读写。
UserId	用户标识。只读。
WaitTime	在 Visual FoxPro 检查 SQL 语句是否结束执行之前延迟的以毫秒计的时间值。默认值是 100 毫秒。可读写。

eExpression

对 *cSetting* 标明的设置值。如果省略了 *eExpression* 参数，则还原成默认值。

返值类型：

数值型

说明

如果调用成功，SQLSETPROP( ) 返回 1；否则，如果发生连接级错误则返回 -1，发生环境级错误则返回 -2。

使用 SQLSETPROP( ) 在连接级上指定设置。若要在环境级指定 Visual FoxPro 默认设置，用 0 作为连接句柄。

ConnectTimeOut 选项只能在 Visual FoxPro 级上设置，而不能在连接级上使用。所有其

他选项既可在连接级，也可在 Visual FoxPro 级上进行设置，每个在 Visual FoxPro 级的设置作为随后连接的默认值。

可用 SQLGETPROP( ) 返回指定设置的当前值。

注意，为了支持 Microsoft Transaction Server 的 SQL pass-through 功能，必须废止 ODBC 登录对话框。使用 SQLSETPROP(cConnectionHandle, 'DispLogin', 3) 可以废止 ODBC 登录对话框（cConnectionHandle 是 SQLCONNECT 返回的连接句柄）。也可以在连接设计器中废止 ODBC 登录对话框。

示例

用 SQLSETPROP( ) 为当前连接设置数据包大小。

```
*清除环境*
CLOSE ALL
CLEAR ALL
CLEAR

*显示选择的连接和数据源对话框
* to choose a connection
nHandle=SQLCONNECT()

*检测连接，报告结果
IF nHandle > 0
    * Set PacketSize
    nSet=SQLSETPROP(nHandle, "PacketSize", 2048 )
    * Test 设置 and display results
    IF nSet > 0
        =MESSAGEBOX("PacketSize was set to 2048",0,"Connection Results")
    ELSE
        =MESSAGEBOX("Error Setting PacketSize",0,"Connection Results")
```

```
   ENDIF
ELSE
   =MESSAGEBOX("No Connection",0,"Connection Results")
ENDIF
=SQLDISCONNECT(nHandle)
```

请参阅

AERROR( ), SQLGETPROP( )

SQLSTRINGCONNECT() 函数

使用一个连接字符串建立和数据源的连接。

语法

SQLSTRINGCONNECT([*cConnectString*])

返值类型：

数值型

参数描述

cConnectString

指定一些 ODBC **驱动程序需要的数据源连接字符串，** Visual FoxPro **将该连接字符串传给** ODBC **驱动程序中。有关数据源连接字符串的详细内容，请参阅** ODBC **驱动程序的文档。**

如果不带 *cConnectString* 发出 SQLSTRINGCONNECT( )，则显示“SQL DATA SOURCE”（SQL 数据源）对话框，允许您选择一个数据源。

说明

如果成功连接到数据源上，SQLSTRINGCONNECT( )返回一个正的、非零句柄，应该将这个句柄存入一个变量，在随后需要连接句柄的函数调用中使用该变量。

示例

下面的示例假定一个名为“MyFoxSQLNT”的 ODBC 数据源可用，并且该数据源的用户标识是“sa”，密码是“FOXPRO”。发出 SQLSTRINGCONNECT( )，返回值存入一个名为 gnConnHandle 的变量。

如果成功连接到数据源上，SQLSTRINGCONNECT( )返回一个正数，然后显示一个对话框，并发出 SQLSTRINGCONNECT( )，解除与数据源的连接。

如果不能连接到数据源上，SQLSTRINGCONNECT( )返回一个负数，并显示一条信息。

```
STORE SQLSTRINGCONNECT('dsn=MyFoxSQLNT;uid=sa;pwd=FOXPRO');
    TO gnConnHandle
IF gnConnHandle < 0
    = MESSAGEBOX('Cannot make connection', 16, 'SQL Connect Error')
ELSE
    = MESSAGEBOX('Connection made', 48, 'SQL Connect Message')
    = SQLDISCONNECT(gnConnHandle)
ENDIF
```

请参阅

AERROR( ), SQLCONNECT( ), SQLDISCONNECT( )

SQLTABLES() 函数

把数据源中的表名存储到 Visual FoxPro 临时表中。

语法

SQLTABLES(*nConnectionHandle* [, *cTableTypes*] [, *cCursorName*])

返值类型：

数值型

参数描述

nConnectionHandle

由 SQLCONNECT( ) 返回的、指向数据源的连接句柄。

cTableTypes

指定一个或多个表类型。有效的表类型有：“TABLE”、“VIEW”、“SYSTEM TABLE”或任何有效的数据源特定表类型标识。*cTabletypes* 必须大写。如果要包含多个表类型列表，请用逗号分隔。

如果省略 *cTableTypes* 参数，或者 *cTableTypes* 是空字符串，则选定数据源中所有表名。

指定的表类型必须用单引号括起来。下面的示例演示了如何指定“VIEW”和“SYSTEM TABLE”表类型。

```
? SQLTABLES(handle, "'VIEW', 'SYSTEM TABLE'", "mydbresult")
```

cCursorName

指定 Visual FoxPro 临时表的名称，结果集合发送到该临时表。如果不包含临时表名，Visual FoxPro 将使用默认名“SQLRESULT”。

下表显示临时表中的列：

列名	说明
TABLE_QUALIFIER	表限定符标识
TABLE_OWNER	表拥有者标识
TABLE_NAME	数据字典中的表名。
TABLE_TYPE	数据字典中的表类型
REMARKS	表的说明

说明

如果成功创建临时表，SQLTABLES( )返回 1；如果仍在执行中，则返回 0。如果发生连接级的错误，返回 -1；发生环境级的错误，返回 -2。

SQLTABLES( )是既可以同步执行又可以异步执行的四个函数之一。SQLSETPROP( )的异步选项设置决定这些函数是同步执行还是异步执行。异步方式下，必须重复调用 SQLTABLES( )，直到返回一个非“假”(.F.)值，“假”(.F.)值表示函数仍在执行。

示例

下面的示例假定名为“MyFoxSQLNT”的数据源可用，并且该数据源的用户标识是“sa”。发出 SQLCONNECT( )，并且返回值存入名为 gnConnHandle 变量。如果不能连接到数据源上，SQLCONNECT( )返回一个负数，并且显示一条信息。

如果成功地连接到数据源上，SQLCONNECT( )返回一个正数并且显示一个对话框。

SQLTABLES( )用来创建一个名为 mycursor 的临时文件，该临时文件包含有关数据源

中表的信息，用 LIST 显示与表有关的信息。

```
STORE SQLCONNECT('MyFoxSQLNT', 'sa') TO gnConnHandle
IF gnConnHandle < 0
    = MESSAGEBOX('Cannot make connection', 16, 'SQL Connect Error')
ELSE
    = MESSAGEBOX('Connection made', 48, 'SQL Connect Message)
    STORE SQLTABLES(gnConnHandle, 'TABLES', 'mycursor') TO nTables
    IF nTables = 1
        SELECT Mycursor
        LIST
    ENDIF
ENDIF
```

请参阅

A E R R O R ( ), SQLCOLUMNS( ), SQLCONNECT( ), SQLGETPROP( ), SQLSETPROP( )

SQRT() 函数

返回指定数值表达式的平方根。

语法

SQRT(*nExpression*)

返值类型：

数值型

参数描述

nExpression

由 SQRT( ) 计算的数值表达式，*nExpression* 不能是负值。

说明

SQRT( ) 返回值中的小数位数是当前小数位数设置和 *nExpression* 中小数位数中的较大者，当前小数位数设置用 SET DECIMALS 指定。

示例

```
CLEAR
? SQRT(4) && 显示数值 2.00
? SQRT(2*SQRT(2)) && 显示数值 1.68
```

请参阅

SET DECIMALS

SROWS() 函数

返回 Visual FoxPro **主窗口中可用的行数。**

语法

```
SROWS( )
```

返值类型：

数值型

说明

SROWS( ) 返回值依赖于当前显示方式，显示方式可以用 SET DISPLAY 更改。

示例

```
CLEAR
? SROWS( )
```

请参阅

COL( ), ROW( ), SCOLS( ), SET DISPLAY, WCOLS( ), WROWS( )

StartMode 属性

包含一个数值，该数值表明 Visual FoxPro 的一个实例是如何启动的。运行时只读。

语法

Applicationobject.StartMode

说明

下表列出了 StartMode 属性可以包含的数值，以及 Visual FoxPro 的实例是如何启动的。

数值	说明
0	一个开发版的 Visual FoxPro 在一个交互工作期启动。
1	Visual FoxPro 启动为一个应用程序对象。例如，下面的命令创建了一个作为应用程序对象的 Visual FoxPro 实例： oMyObject = CREATEOBJECT('VisualFoxPro.Application')
2	Visual FoxPro 启动为一个外部处理 .exe 自动服务程序。
3	Visual FoxPro 启动为一个内部处理 .dll 自动服务程序。
4	Visual FoxPro 启动为一个可发布的 .app 或 .exe 文件。

有关使用 Visual FoxPro 创建自定义 automation 服务程序的详细内容，请参阅《Microsoft Visual FoxPro 6.0 中文版中文版程序员指南》的第十六章“添加 OLE”。

应用于：

Application 对象，_VFP 系统变量

请参阅

CREATE 对象(), ServerName 属性, SYS(2335)

_STARTUP 系统变量

指定启动 Visual FoxPro 时运行的应用程序名。

语法

_STARTUP = *ProgramName*

参数描述

ProgramName

启动 Visual FoxPro 时要运行的应用程序。必须在 Visual FoxPro 的配置文件中包含 _STARTUP。

也可以在配置文件中包含下列命令行之一，指定 Visual FoxPro 启动时运行的命令或程序：

COMMAND = cVisualFoxProCommand

– 或 –

COMMAND = DO ProgramName

_STARTUP 指定的启动程序总是比配置文件中 命令 指定的命令或程序先运行。

说明

_STARTUP 默认包含 Vfp6strt.app。也可以在“选项”对话框的“文件位置”选项卡指定一个启动应用程序。

请参阅

DO

StatusBar 属性

指定在 Visual FoxPro 一个实例的状态栏显示的文本。在运行时可读写。

语法

Application Object.StatusBar[= cMessageText]

参数描述

cMessageText

指定在状态栏显示的文本。

应用于

Application **对象，** _VFP **系统变量**

请参阅

SET MESSAGE

StatusBarText 属性

指定控件获得焦点时在状态栏中显示的文本。设计和运行时可用。

应用于

复选框，组合框，命令按钮，编辑框，表格，列表框，选项按钮，微调，文本框

语法

[*Form*.] Control StatusBarText[= *cText*]

参数描述

cText

指定文本，用该文本对获得焦点的控件加以说明。

请参阅

Caption 属性

STORE 命令

将数据存入变量、数组或数组元素中。

语法

STORE eExpression TO VarNameList | ArrayNameList

– 或 –

VarName | ArrayName = eExpression

参数描述

eExpression

指定一个表达式，该表达式的值将存入变量、数组或数组元素中。如果指定变量不存在，则创建该变量，并且将其初始化为 *eExpression*。数组必须先用 DIMENSION 命令定义，STORE 命令用新值替换现有的变量、数组或数组元素中的值。

VarNameList

指定变量或数组元素的列表，将 *eExpression* 存入这些变量或数组元素中。变量名或数组元素之间用逗号分隔。

ArrayNameList

指定 *eExpression* 所要存入的、已存在的数组名，数组名之间用逗号分隔。如果 SET COMPATILE 为 OFF，那么 STORE 命令用指定值初始化数组中的每一个元素；如果 SET COMPATIBLE 为 ON，STORE 命令将指定值存入指定名称的变量中，并改写任何已存在的同名数组。

说明

等号 (=) 赋值操作符可代替 STORE 命令。变量、数组或数组元素必须位于等号的左边，其值位于等号的右边。

使用大括号可以将日期直接存入变量、数组或数组元素：

```
STORE {12/25/99} TO gdXMas
```

有关创建日期和日期时间值的详细内容，请参阅《Microsoft Visual FoxPro 6.0 中文版中文版程序员指南》的第三十三章“对编程的改进”中的“对 2000 年日期的支持”。

帮助中的“系统容量”列出了能够创建的变量或数组的最大数。在 Visual FoxPro 配置文件中可以增加或减少此限制。有关配置 Visual FoxPro 的详细内容，请参阅帮助中

《安装指南》的第三章“配置 Visual FoxPro ”。

示例

```
STORE DATE( ) TO gdDate
STORE 50 TO gnNumeric
STORE 'Hello' TO gcCharacter
STORE .T. TO glLogical
STORE $19.99 TO gyCurrency

DIMENSION gaMyArray(2,2)
SET COMPATIBLE OFF
STORE 2 TO gaMyArray

CLEAR
DISPLAY MEMORY LIKE g*
```

请参阅

DIMENSION, SET COMPATIBLE