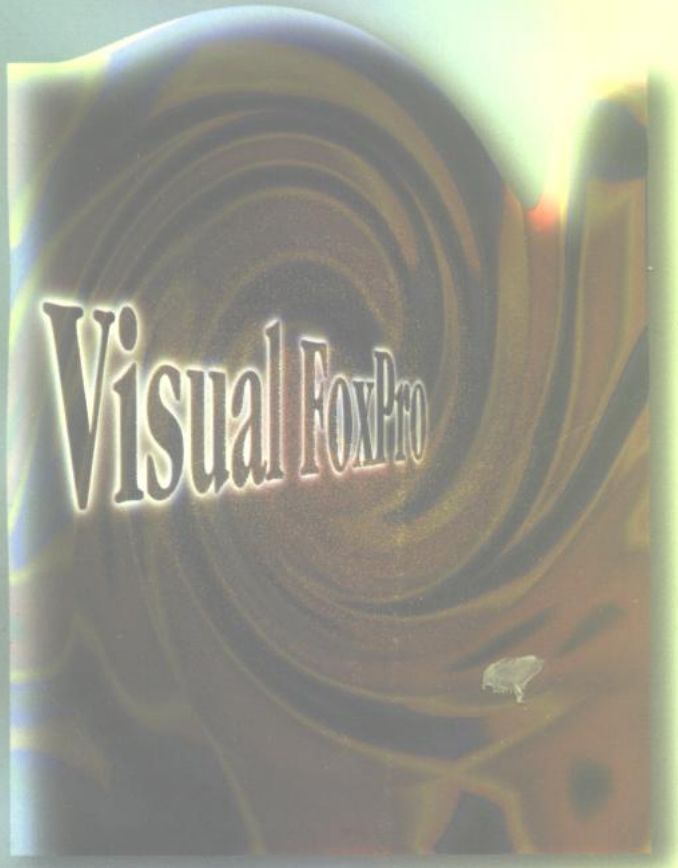
计算机语言函数应用丛书

Visual FoxPro

计算机语言函数应用

王 博 童长武 杨 柳 王广云 编著



科学出版社

计算机语言函数应用丛书

Visual FoxPro

计算机语言函数应用

王 博 童长武 杨 柳 王广云 编著

科学出版社
2000

内 容 简 介

近年来，FoxPro系列产品作为一种开发信息管理系统的强大工具而深受开发人员乃至广大用户的喜爱。从最初的FoxBASE发展到今天的Visual FoxPro（VPF）6.0 for Windows，已经逐渐将可视化编程、面向对象程序设计以及ODBC技术融入其中。本书详细介绍了VPF 6.0 for Windows的所有命令和函数的功能、语法及其参数的意义和用法，并针对比较重要的命令和函数给出相应的应用实例。附录中还给出系统菜单名称、系统变量等有用信息。同时本书还涵盖FoxPro新、旧版本的所有命令与函数。本书是开发关系型数据库人员的一本有用的工具书。

图书在版编目（CIP）数据

Visual FoxPro计算机语言函数应用/王 博等编.-北京：科学出版社，2000.1
ISBN 7-03-007768-7

Ⅰ.计… Ⅱ.王… Ⅲ.关系数据库-数据库管理系统，FoxPro-库函数
Ⅳ.TP311.13

中国版本图书馆CIP数据核字（1999）第30854号

科学出版社出版
北京东黄城根北街16号
邮政编码：100717
北京双青印刷厂 印刷
新华书店北京发行所发行 各地新华书店经售
*
2000年1月第 一 版 开本：787×1092 1/16
2000年1月第一次印刷 印张：32 3/4
印数：1—5 100 字数：776 600

定价：46.00 元

（如有印装质量问题，我社负责调换(环伟)）

前　言

Visual FoxPro 6.0（VFP 6.0）是微软公司自 1993 年发行 FoxPro 2.5 for Windows 之后版本最新、功能最全的 Xbase 类数据库，是一个全新的 32 位数据库开发系统，支持 Windows 95 和 Windows NT，也支持 Windows 98，是 Microsoft Visual Studio 98 的一部分。相对于前期的版本，Visual FoxPro 6.0 增加了许多新特性，尤其在支持 Internet 方面的功能更加显著。例如，FoxPro 6.0 支持活动文档。活动文档是基于 Windows 并可以内嵌到 Internet 浏览器中的非 HTML 应用程序，它既可以运行表单、报表、类和代码，也可以内嵌到 Internet Explorer 等浏览器中。VFP 6.0 已全面支持 GIF（Graphics Interchange Format）和 JPEG（Joint Photographic Electronic Group）图象格式。通过 VFP 6.0 开发的应用软件可支持 HTML 格式的帮助文档，使得用户获得帮助更加方便、快捷。VFP 6.0 还可以让开发者建立自动服务器，更加完全地支持 Client/Server 体系结构。VFP 6.0 已支持 OLE 拖放技术，在开发的应用程序中以及与其他支持拖放技术的程序之间可以很方便地拖放数据。另外，通过 VFP 6.0 开发的程序已解决了 2000 年问题。最后，VFP 6.0 继续强化了一些重要的功能，如 ODBC 技术、面向对象的特性等。通过 ODBC，在 VFP 6.0 中可以访问几乎任何类型的数据库，如 SQL Server，Oracle 等。在面向对象方面，除更加完善对抽象性、继承性、多态性的支持外，VFP 6.0 还支持用户自定义接口技术。

为了便于广大数据库程序开发人员更好地使用 VFP 6.0，特编写了这本命令与函数专集。本书共分为两篇，具体结构安排如下：第一篇为命令篇，书中将命令按功能分为数据库操作命令、界面控制命令、窗口命令、菜单命令、索引排序命令、变量与数组处理命令、程序流程控制命令、键盘与鼠标命令、错误处理与调试命令、SQL 命令、打印命令、环境设置命令、程序管理命令、文件管理命令以及网络命令；第二篇为函数篇，包括字符函数、数值函数、类型转换函数、日期和时间函数、数据库操作函数、菜单函数、窗口函数、键盘与鼠标输入函数、环境设置函数、SYS（）函数、变量处理函数、打印函数、程序管理函数、文件管理函数、网络函数以及 DDE 函数等。

由于作者水平有限，再加上时间仓促，错误及不妥之处在所难免，敬请广大读者批评指正。

目　录

命 令 篇

函 数 篇

命 令 篇

第一章 数据库操作命令

ADD TABLE

功能：将一个自由表添加到当前数据库中。

语法：ADD TABLE TableName|? [NAME LongTableName]

参数：TableName 添加到数据库中的自由表的名称。

? 可显示“打开”对话框，从中选择添加到数据库中的表。

NAME LongTableName 为添加的表指定一个长名（最长 128 个字符），可以代替扩展名为 .dbf 的短文件名。

注释：自由表添加到数据库中后，就不再是自由表，但其操作方式不会发生变化，而且还可以通过 REMOVE TABLE 命令重新释放为自由表。

自由表必须满足以下几个条件才能添加到数据库中：

(1) 必须是合法的 .DBF 文件；

(2) 不能与数据库中已有的表同名；

(3) 必须是自由表。

(4) 所添加到的数据库不能处于事务处理当中。

参阅：CLOSE DATABASES, CREATE DATABASE, DISPLAY TABLES, FREE TABLE, OPEN DATABASE, REMOVE TABLE

APPEND

功能：在当前表的末尾添加一条或多条新录。

语法：APPEND [BLANK]
[IN nWorkArea | cTableAlias]
[NOMENU]

参数：BLANK 在当前表末尾添加一条空记录。执行 APPEND BLANK 命令时不打开编辑窗口。

IN nWorkArea 指定追加新记录的表所在的工作区号。

IN cTableAlias 指定追加新记录的表的别名。

如果省略工作区号和别名，新记录将追加到当前工作区中打开表的末尾。

NOMENU 指定该参数可将菜单标题“表”从菜单栏中删除，从而防止对编辑

窗口格式的修改。

注释：如果指定 APPEND 或 APPEND BLANK 命令时当前工作区中无表打开，会显示“打开”对话框，从中可以选择追加记录的表。

APPEND 命令执行时打开编辑窗口，从中可以输入记录数据。追加一条新记录后，Visual FoxPro 自动更新所有打开的索引。

注意：添加空记录时，编辑窗口不会打开。

范例：用 APPEND BLANK 命令创建一个包含随机数的 10 条记录的表，显示表中数据的最大值和最小值。

```
CLOSE DATABASES
CREATE TABLE Random (cValue N(3))
FOR nItem = 1 TO 10 && 追加 10 条记录
    APPEND BLANK
    REPLACE cValue WITH 1 + 100 * RAND( ) && 插入随机数
ENDFOR
CLEAR
LIST && 显示数值
gnMaximum = 1 && 初始化最小值
gnMinimum = 100 && 初始化最大值
SCAN
   gnMinimum = MIN(gnMinimum, cValue)
   gnMaximum = MAX(gnMaximum, cValue)
ENDSCAN
? '最小值为:', gnMinimum
? '最大值为:', gnMaximum
```

参阅：APPEND FROM ARRAY，BROWSE，CHANGE，EDIT，INSERT-SQL，REPLACE

APPEND FROM

功能：将另一个文件中的数据追加到当前表的末尾。

语法：APPEND FROM FileName|? [FIELDS FieldList] [FOR lExpression][[TYPE]
[DELIMITED[WITH TAB|WITH Delimiter|WITH BLANK||
WITH CHARACTER Delimiter]
|DIF|FW2|MOD|PDOX|RPD|SDF|SYLK|WK1|WK3|
WKS|WR1|WRK|XLS|XL5]][AS nCodePage]

参数：FileName 取得数据的源文件，若未给出扩展名，系统会默认为 .dbf，即表文件。而且无论 SET DELETED 的设置如何，表中标记为删除的记录也被追加过来。

?：显示打开对话框，从中可选择要追加记录的表。

FIELDS FieldList 指定要追加数据的字段。

FOR lExpression　只对满足 lExpression 条件的记录有效。

TYPE　指定源文件的类型。虽然当源文件不是表时，必须指定文件的类型，但 TYPE 关键字是可选的。源文件类型十分广泛，包括带分界符的 ASCII 文本文件在内，都可以用做源文件

DELIMITED　指定追加到当前表中的文件是一个分隔数据文件，该文件为 ASCII 文本文件。每一条记录的结尾都跟有一个回车符和一个换行符，各字段内容默认由逗号分隔，而且字符型数值还要加上双引号。

WITH Delimiter　指定文件中的字符字段不是由引号分隔，而由特定字符分隔，如 with _。

WITH TAB　指定文件中的字段不是由逗号分隔，而是由 TAB 分隔。

WITH BLANK　指定文件中的字段由空格分隔。

WITH CHARACTER Delimiter　指定文件中的字段由 Delimiter 指定的字符分隔，如 with ?。

DIF　指定可以从 VisiCalc.DIF（数据交换格式）文件导入数据，文件中的列成为当前表中的字段，而行则成为记录。

FW2　指定可从 Framework II 创建的文件导入数据。

MOD　指定可从 Microsoft Multiplan version 4.01 创建的文件中导入数据。

PDOX　指定可导入 Borland 公司 Paradox v3.5，v4.0 数据库中的数据。

RPD　指定可从 RapidFile 1.2 创建的文件中导入数据。

SDF　指定可从系统数据格式文件中导入数据。该文件为 ASCII 文本文件，其记录有固定的长度,并以回车符和换行符结尾,其中各个字段不做分隔,扩展名为 .TXT。

SYLK　指定可从 MS-Multiplan 中使用的符号连接交换格式（Symbolic Link Interchange）文件中导入数据，无扩展名。

WK1　指定可从 Lotus 1-2-3 版本 2.x 电子表格中导入数据。

WK3　指定可从 Lotus 1-2-3 电子表格中导入数据。

WKS　指定可从 Lotus 1-2-3 版本 1－A 电子表格中导入数据。

WR1　指定可从 Lotus Symphony 版本 1.1 或 1.2 电子表格中导入数据。

WRK　指定可从 Lotus Symphony 版本 1.0 电子表格中导入数据。

XLS　指定可从 Microsoft Excel 电子表格中导入数据。

XL5　指定可从 Microsoft Excel 版本 5.0 电子表格中导入数据。

AS　nCodePage 指定源表或源文件的代码页。Visual FoxPro 在复制源表或源文件的同时，自动将数据转换到当前表的代码页。

如果指定的代码页不支持，Visual FoxPro 将产生一条错误信息。可以用 GETCP()函数显示“代码页”对话框，从中为追加数据的表或文件指定代码页。

如果 nCodePage 为 0，Visual FoxPro 假设源表或源文件的代码页与当前所选表的代码页相同，不需要进行代码页的转换。

注释：如果要追加的源表或源文件是 Visual FoxPro 表或 FoxPro 早期版本创建的表，默认扩展名为 .DBF。如果未带扩展名，则必须指定。如果是其他文件类型，还须

指定其类型。

源表中标记为删除的记录在添加到目标表后，删除标记去除。

通过 DBF（）函数还可以从 SELECT-SQL 命令创建的临时、只读游标中追加记录，所用命令如下：

APPEND FROM DBF（'<Cursor Name>'）

范例：打开 customer 表，将表结果复制到 backup 表中并打开 backup 表。然后 Visual FoxPro 向其追加 customer 表中国家为芬兰的所有记录。再将这些记录复制到新的分隔文件 TEMP.TXT 中。

```
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'Data\testdata')
USE customer
COPY STRUCTURE TO backup
USE backup
APPEND FROM customer FOR country = 'Finland'
COPY TO temp TYPE DELIMITED
MODIFY FILE temp.txt
USE
DELETE FILE backup.dbf
DELETE FILE temp.txt
```

参阅：COPY FILE，COPY TO，EXPORT，GETCP（），IMPORT

APPEND FROM ARRAY

功能：将数组中的每一行数据作为一条记录添加到当前表中。

语法：APPEND FROM ARRAY ArrayName
[FOR lExpression]
[FIELDS FieldList
| FIELDS LIKE Skeleton
| FIELDS EXCEPT Skeleton]

参数：ArrayName　指定作为数据源的数组的名称。

FOR lExpression　指定从数据中追加记录的条件，lExpression 必须在条件表达式中包含目标字段的名称。

将数组数据追加到表中之前，检查与 lExpression 中指定的目标字段对应的数组元素，看其是否满足 lExpression 指定的逻辑条件，如果满足条件，则追加该行记录。如果不能满足条件，该数组行数据不能追加，再检查下一行数据。

FIELDS FieldList　指定要增加数据的字段。第一个字段用数组中第一个元素更新，第二个字段由第二个数组元素来更新，依次类推。不带本参数则表示更新所有字段。

FIELDS LIKE Skeleton　指定只更新符合字段梗概 Skeleton 的字段的数据。

FIELDS EXCEPT Skeleton　指定只更新除符合字段梗概 Skeleton 之外的字段的

数据。

字段梗概支持通配符。例如，要指定用数组数据更新以字母 A 和 P 开头的所有字段，可使用如下命令：

APPEND FROM ARRAY aMyArray FIELDS LIKE A*，P*

LIKE 子句可以和 EXCEPT 子句配合使用，例如：

APPEND FROM ARRAY aMyArray FIELDS LIKE A*,P* EXCEPT PARTNO*

注释：APPEND FROM ARRAY 命令忽略备注型字段和通用型字段。如果表以共享方式打开，APPEND FROM ARRAY 命令在追加记录时还锁定表头。

如果数组时一维数组，APPEND FROM ARRAY 向表追加一条记录。数组中第一个元素填充新记录中第一个字段值，第二个元素值填充第二个字段值，依次类推。

如果一维数组中的元素数大于表中的字段数，则多出的元素将被忽略。如果表中的字段数多于数组元素数，则多出的字段被置为空值。下表是各个字段类型的缺省空值。

字段类型	缺省空值
字符型	空格
数值型	0
货币型	0
浮点型	0
整型	0
双字节型	0
日期型	空日期
日期时间型	空日期时间
逻辑型	假（.F.）
备注型	空

如果是二维数组，APPEND FROM ARRAY 将数组中每一行作为一条记录追加到表中。例如，如果数组由两行，则向表追加两条记录。

数组的第一列内容填充新加记录的第一个字段，第二列内容填充新记录的第二个字段，依次类推。如果二维数组的列数多于表中的字段数，多出的列将被忽略。反之，如果字段数多于数组的列数，则多出的字段同样被置为空值。

范例：新建一个表，然后用 APPEND FROM ARRAY 命令向表中追加记录。

```
LOCAL ARRAY aNewRec (3)
CREATE TABLE Test FREE (Object C (10), Color C (16), SqFt n(6,2))
SCATTER TO aNewRec BLANK && 从表创建新数组
aNewRec [1] ="Box"&& 给数组赋值
aNewRec [2] ="Red"
aNewRec [3] =12.5
APPEND FROM ARRAY aNewRec && 将数组元素追加到表中
```

参阅：APPEND，COPY TO ARRAY

APPEND GENERAL

功能：从文件输入 OLE 对象，放入通用型字段中。

语法：APPEND GENERAL GeneralFieldName [FROM FileName]
[DATA c]
[LINK]
[CLASS OLEClassName]

参数：GeneralFieldName　指定存放 OLE 对象的通用型字段的名称。

FROM FileName OLE　对象所在的源文件（必须包含全名及其扩展名），例如可能是一个 .WAV 声音文件，也可能是一个 .BMP 图象文件。

DATA cExpression cExpression　指定一个字符串表达式，该表达式被计算并作为字符串传递给通用型字段中的 OLE 对象。该 OLE 对象必须能够接收和处理该字符串。

LINK 在 OLE 对象和包含该对象的文件之间建立连接，这样该对象就不会嵌入通用型字段，而只是与该字段建立一连接关系。但是其内容仍会出现在此通用字段中。如果省略 LINK，OLE 对象将嵌入通用型字段中。

CLASS OLEClassName　指定 OLE 的类（非缺省类）。当包含 OLE 对象的文件的扩展名不同于默认扩展名，并且要强制类行为时，可以指定类名。如果默认扩展名可用于多个 OLE 服务器，则可用 CLASS 子句指定具体的服务器。

注释：如果该通用型字段中已经有一个 OLE 对象，执行该命令后会被新的 OLE 对象覆盖。若要从通用型字段中去除 OLE 对象，需执行 APPEND GENERAL GeneralFieldName（GeneralFieldName 是要清除的通用型字段的名称）命令。

参阅：@. . . SAY-图片与 OLE 对象，MODIFY GENERAL，OLE Bound 控件

APPEND MEMO

功能：将文本文件中的数据复制到备注字段中。

语法：APPEND MEMO MemoFieldName FROM FileName [OVERWRITE]
[AS nCodePage]

参数：MemoFieldName　数据所放入的备注型字段的名称。

FileName　指定作为数据源的文本文件的名称（包含全名和扩展名）。

OVERWRITE　指定用文件的内容覆盖备注型字段原有内容。

AS nCodePage　指定拷贝到备注型字段的文本文件的代码页。

注释：如果省略了 OVERWRITE 参数，文件中的数据将被追加到备注型字段的末尾。

范例：将备注字段 notes 的数据复制到文本文件 Test.txt 中，然后再将 Test.txt 的数据追加到备注字段的末尾，最后用 Test.txt 的数据替换备注字段的数据。

```
CLOSE DATABASES
OPEN DATABASE (HOME(2) + 'Data\testdata')
USE employee
MODIFY MEMO notes NOEDIT && 打开 notes 备注字段
COPY MEMO notes TO test.txt && 根据备注字段创建文本文件
WAIT WINDOW 'TEST.TXT 文本文件- 按 ESC 键' NOWAIT
MODIFY FILE test.txt NOEDIT && 打开文本文件
APPEND MEMO notes FROM test.txt && 追加文本文件数据
MODIFY MEMO notes NOEDIT && 再显示备注字段
APPEND MEMO notes FROM test.txt OVERWRITE && 替换 notes
MODIFY MEMO notes NOEDIT NOWAIT
```

DELETE FILE test.txt

参阅：COPY MEMO，GETCP（）

APPEND PROCEDURES

功能：将文本文件中的存储过程追加到当前数据库中的存储过程中。

语法：APPEND PROCEDURES FROM FileName [AS nCodePage]
[OVERWRITE]

参数：FileName　指定要追加的存储过程所在的文本文件的名称。

AS nCodePage　指定要追加的存储过程所在文本文件的代码页。Visual FoxPro复制文本文件的内容，同时自动将文本文件内容转换到指定的代码页。

如果指定的 nCodePage 的值无效，Visual FoxPro 会产生出错信息。可以用 GETCP（）函数显示代码页对话框，从中指定一个代码页。

如果省略 AS nCodePage，系统复制要追加的存储过程所在文本文件的内容，同时自动将文本文件的内容转换到 Visual FoxPro 当前代码页。Visual FoxPro 当前代码页可由 CPCURRENT（）函数确定。

如果 nCodePage 为 0，系统会认为要追加的存储过程所在的文本文件的代码页与当前数据库的代码页相同，从而不对 Visual FoxPro 当前代码页进行转换。

OVERWRITE　指定文本文件中的存储过程覆盖数据库中的存储过程。如果省略 OVERWRITE，则不会覆盖，而是将文本文件中的存储过程追加到当前存储过程。

注释：可以程序形式使用 APPEND PROCEDURES 命令来修改数据库中的存储过程。执行 APPEND PROCEDURES 命令时，数据库必须是打开的，并且是当前数据库。否则会出现出错信息。

范例：CLOSE DATABASES

OPEN DATABASE (HOME (2) + 'Data \ testdata') && 打开 testdata 数据库

CREATE TABLE mytable FREE (mProcedure M) && 创建自由表，mProcedure 为备注字段

APPEND BLANK && 添加一条空记录

REPLACE mProcedure WITH "PROCEDURE MyProcedure" + CHR(13) + CHR (10)

COPY MEMO mProcedure TO mytemp.txt && 将备注字段内容复制到临时文件中

USE && 关闭临时表

APPEND PROCEDURES FROM mytemp.txt && 将过程复制到数据库

CLEAR

DISPLAY PROCEDURES && 显示与当前数据库相关的过程

DELETE FILE mytable.dbf && 删除临时表

DELETE FILE mytable.fpt && 删除临时表备注字段

DELETE FILE mytemp.txt && 删除临时文本文件

参阅：COPY PROCEDURES，CREATE TRIGGER，DISPLAY PROCEDURES，MODIFY PROCEDURE，OPEN DATABASE，PROCEDURE，SET DATABASE

AVERAGE

功能：计算数据库中数值型字段的算术平均值。

语法：AVERAGE [ExpreList][Scope]
[FOR lExpression1][WHILE lExpression2]
[TO VarList|TO ARRAY ArrayName]
[NOOPTIMIZE]

参数：ExpreList　指定求平均值的表达式，该表达式可以是多个字段，中间以逗号分隔，也可以是与表中数值字段有关的表达式。

Scope　指定参加求平均值运算的记录或记录范围。Scope 的选项有 ALL，NEXT nRecords，RECORDS nRecordNumber 以及 REST。缺省为 ALL。

FOR lExpression1　只计算符合 lExpression1 逻辑条件的记录。

WHILE lExpression2　只有 lExpression2 为真时，记录被包括进求平均值运算中。在该选项中，系统将自动关闭 Rushmore 功能。

TO VarList　指定存放计算结果的变量或数组元素列表。

TO ARRAY ArrayName　指定存放计算结果的一维数组。如果该数组不存在，系统将自动创建，并自动调整其大小。

NOOPTIMIZE　关闭该命令的 Rusbmore 优化功能。

注释：如果 SET TALK 为 ON，计算结果显示在屏幕上。如果 SET HEADINGS 为 ON，字段名或与表达式有关的字段会显示在结果上方。

范例：use employees
AVERAGE age，salary FOR employeeID>20 TO ages，salarys
*计算编号大于 20 的员工的平均年龄与平均工资。

参阅：CACULATE，SUM

BLANK

功能：清除当前表记录中字段的数据。

语法：BLANK [FIELDS FieldList] [Scope]
[FOR lExpression1] [WHILE lExpression2]
[NOOPTIMIZE]

参数：FieldList　只清除该参数指定的字段列表中字段的数据。缺省为清除所有字段。

Scope　指定要清除的记录范围，可选项有 ALL，NEXT nRecords，RECORD nRecordNumber 以及 REST。缺省为当前记录。

FOR lExpression1　只清除满足该表达式条件的记录中的字段。

WHILE lExpression2　只清除满足该表达式条件的记录中的字段。

NOOPTIMIZE　禁止对该命令的优化。

注释：如果记录指针已指向当前工作区的末尾，那么 BLANK 不清除另一相关工作区中

记录的字段数据。要使该命令能够作用到其他相关记录的字段上，记录指针必须指向当前工作区中一个已排序的记录。

范例：CLOSE DATABASES
OPEN DATABASE (HOME ( ) + 'samples \ data \ testdata')
USE customer && 打开 customer 表
CLEAR
DISPLAY && 显示当前记录
SCATTER TO gaCustomer && 用记录内容创建数组
BLANK && 清除此记录
DISPLAY && 显示空记录
GATHER FROM gaCustomer && 然后恢复原始记录内容
DISPLAY && 再显示恢复后的记录

参阅：APPEND, EMPTY ( ), ISBLANK, REPLACE, SET OPTIMIZE

BROWSE

功能：打开浏览窗口，显示当前表或所选表中的记录。

语法：BROWSE [FIELDS FieldList]
[FONT cFontName [, nFontSize]]
[STYLE cFontStyle]
[FOR lExpression1 [REST]]
[FORMAT]
[FREEZE FieldName]
[KEY eExpression1 [, eExpression2]]
[LAST | NOINIT]
[LOCK nNumberOfFields]
[LPARTITION]
[NAME ObjectName]
[NOAPPEND]
[NODELETE]
[NOEDIT | NOMODIFY]
[NOLGRID] [NORGRID]
[NOLINK]
[NOMENU]
[NOOPTIMIZE]
[NOREFRESH]
[NORMAL]
[NOWAIT]
[PARTITION nColumnNumber [LEDIT] [REDIT]]
[PREFERENCE PreferenceName]

[SAVE]
[TIMEOUT nSeconds]
[TITLE cTitleText]
[VALID [:F] lExpression2 [ERROR cMessageText]]
[WHEN lExpression3]
[WIDTH nFieldWidth]
[WINDOW WindowName1]
[IN [WINDOW] WindowName2 | IN SCREEN]
[COLOR SCHEME nSchemeNumber]

参数：FIELDS FieldList　指定要浏览的字段列表。也可以包含其他相关表的字段，但需在这些字段前加上所在表的别名。缺省为浏览所有字段。

FONT cFontName [, nFontSize]　指定浏览窗口所使用的字体及字号。字符表达式 cFontName 指定字体名，数值表达式 nFontSize 指定字号。例如，下列子句指定浏览窗口使用 16 点的“Courier”字体：

FONT 'Courier', 16

如果使用 FONT 子句，但却省略 nFontSize 参数，则缺省使用该字体的 10 点字号。如果省略 FONT 子句，则缺省使用 8 点的 MS Sans Serif 字体。如果所指定的字体无法使用，系统会用一种最相近的字体来代替。

STYLE cFontStyle　指定浏览窗口的字体风格。如果省略 STYLE 子句，则使用“常规”风格字体。

字符	字体风格
B	黑体
I	斜体
N	常规
O	轮廓
Q	不透明
S	阴影
-	删除线
T	透明
U	下划线

如果所指定的字体风格无法使用，系统会用一种最相近的字体风格或“常规”风格来代替。上表是一些字符所代表的字体风格。可以用多个字符指定组合风格。

FOR lExpression1 [REST]　指定浏览窗口中只显示使表达式 lExpression1 为真的记录。如果 lExpression1 为可优化表达式，则 Rushmore 技术可以优化 BROWSE FOR 所指定的查询。FOR 子句可使记录指针移到满足条件的第一条记录，而如果使用 REST 可强制记录指针保持在当前位置。

FORMAT　指定使用格式文件控制浏览窗口的显示格式和数据输入格式。格式文件必须已用 SET FORMAT 命令打开。下面是应用于浏览窗口的格式文件的一部分内容：

要浏览的字段列表；所有 VALID 子句；所有 WHEN 子句；所有 RANGE 子句；字段宽度（PICTURE 子句指定）；所有 SAY 表达式。

下例用格式文件来验证输入浏览窗口的数据。

第一行创建一个浏览字段（cust_id），它有 5 个字符宽，并且只允许输入字母和数字。第二行创建一个浏览字段（company），其中不能有空格，最多可有 20 个字母字符。第三行创建一个浏览字段（contact），仅当字段为空时才允许输入数

据。

下面是 CUSTENTR.FMT 格式文件的内容，用它来验证输入 customer 表的数据的有效性：

```
@ 3, 0 GET cust _ id PICTURE 'NNNNN'
@ 3, 0 GET company VALID company ! = SPACE (40) ;
                    PICTURE 'AAAAAAAAAAAAAAAAAAAA'
@ 3, 0 GET contact WHEN contact = SPACE (40)
* 下面是使用格式文件的程序
CLOSE DATABASES
OPEN DATABASE (HOME ( ) + 'samples \ data \ testdata')
USE customer && 打开 customer 表
SET FORMAT TO custentr.fmt
BROWSE FORMAT
```

FREEZE FieldName　只允许对浏览窗口中一个字段进行修改，该字段用 FieldName 指定。其他字段只显示，但不能编辑。

```
CLOSE DATABASES
OPEN DATABASE (HOME( ) + 'samples \ data \ testdata')
USE customer && 打开 customer 表
BROWSE FIELDS phone :H = 'Phone Number:', ;
                    company :H = 'Company:' ;
                    FREEZE phone
```

KEY eExpression1 [, eExpression2]　限制浏览窗口中显示的记录的范围。可以用 KEY 子句指定一个索引关键值 (eExpression1) 或者一个关键值范围 (eExpression1, eExpression2)，使浏览窗口中只显示此范围内的记录。但条件是所浏览的表必须已建立索引，并且 KEY 子句中所使用的索引关键值的数据类型必须与主索引文件或标识的索引表达式一致。

LAST | NOINIT　保存对浏览窗口外观所做的更改。所做更改保存在 FOXUSER 文件中，其中可包括字段列表、各字段宽度以及浏览窗口的位置和大小的变更。

如果 BROWSE 命令使用 LAST 或 NOINIT 子句，而且 SET RESOURCE 为 ON，则浏览窗口将按 FOXUSER 文件中上一次存储的配置打开。这样就恢复了上一次执行 BROWSE 命令时所创建的浏览窗口的配置。如果上一次在命令窗口中执行的 BROWSE 命令带有一长串子句，则带 LAST 或 NOINIT 选项执行 BROWSE 命令可避免重新键入该命令。有关 FOXUSER 文件的详细情况，请参见 SET RESOURCE。

如果上一次打开浏览窗口时使用的 BROWSE 命令带 PREFERENCE 子句，则 BROWSE LAST 不能恢复上次的设置。如果按 CTRL + Q 退出浏览窗口，将会丢失本次会话中对浏览窗口所做的所有更改。LAST 和 NOINIT 子句作用相同，保留 NOINIT 的目的是为了与 dBASE 保持兼容。