

目 录

第一章 FoxPro 语言概述	1
1.1 命令窗口	1
1.2 创建程序	1
1.2.1 保存程序	2
1.2.2 修改程序	2
1.2.3 其他程序信息	3
1.3 FoxPro 的联机帮助工具	3
1.3.1 FoxPro for MS-DOS 中的联机帮助	3
1.3.2 FoxPro for Windows 中的联机帮助	3
1.4 FoxPro 语言的组成	4
1.4.1 语法规则	4
1.4.2 表达式	5
1.4.3 缩写说明	10
1.4.4 用 Scope, FOR 和 WHILE 指定记录范围	10
1.4.5 表或数据库的工作区	12
1.4.6 函数和备注字段	14
第二章 命令和函数分类介绍	16
2.1 FoxPro 命令和函数关键字	16
2.2 命令和函数分类	18
2.3 数据类型	19
2.3.1 字符型函数	19
2.3.2 数值函数	21
2.3.3 日期和时间函数	22
2.3.4 数据转换	23
2.4 表/.DBF	23
2.4.1 字段处理	23
2.4.2 表/.DBF 处理	24
2.4.3 记录处理	25
2.4.4 索引	26
2.4.5 关系	27
2.5 编程	27
2.5.1 调试和错误处理	27
2.5.2 程序执行	28
2.5.3 结构化编程	28

2.5.4	低级文件函数.....	29
2.5.5	内存变量处理.....	29
2.5.6	数组处理.....	30
2.5.7	事件处理程序.....	31
2.5.8	DDE 和 OLE	31
2.6	输入输出.....	32
2.6.1	数据格式.....	32
2.6.2	打印.....	34
2.6.3	菜单条和弹出式菜单.....	34
2.6.4	窗口.....	36
2.6.5	键盘输入和鼠标输入.....	37
2.6.6	报告和标签.....	38
2.7	FoxPro 环境	38
2.7.1	环境.....	38
2.7.2	文件管理.....	41
2.8	多用户环境中的 FoxPro	42
2.8.1	网络命令和函数.....	42
第三章 命令和函数详解		44
3.1	2.5 版的 FoxPro	44
3.2	命令和函数详解.....	48
第四章 系统内存变量.....		714
附录 A 命令、函数与系统内存变量清单		738

第一章 FoxPro 语言概述

FoxPro 是为数据管理而开发的软件产品,它具有对信息的分类、检索和处理等功能。尽管通过 FoxPro 的友好界面能够实现许多非过程化的数据管理操作,但为了充分发挥 FoxPro 的强大威力,读者还须掌握一些 FoxPro 语言的编程知识。

本书是一本 FoxPro 语言的参考手册,详细地列出了 FoxPro 在 MS-DOS 和 Microsoft Windows 这两种平台上的用法。本手册中所指的 FoxPro 既代表 FoxPro for MS-DOS,又代表 FoxPro for Windows。如果某些特性是专门针对 FoxPro for MS-DOS 或 FoxPro for Windows 的,我们将加以注明。我们在第三章开始部分列出一张表,该表以字母顺序完整地列出了在 FoxPro 2.5 版本中新提供的和功能增强的语言特点。

本章讨论的内容比较多,这些内容将会对用户使用 FoxPro 编程语言有所帮助。

1.1 命令窗口

当用户使用 FoxPro 用户界面时,FoxPro 将在命令窗口中生成对应于用户动作的相应命令。命令窗口是 FoxPro 用户界面与 FoxPro 语言之间的桥梁。

用户同 FoxPro 进行交互式对话操作期间产生的所有命令都将在命令窗口中出现,这就提供了一种学习 FoxPro 语言的手段。读者只要认真学习在使用用户接口期间产生的命令,就能很快地学会 FoxPro 语言。

在命令窗口中生成的命令能够再次执行,只要将光标移到该命令所在行并按回车键即可。如果 FoxPro 进行的交互式对话时间较长的话,可将这些命令拷贝或粘贴到一个 FoxPro 程序中。程序可以重复运行,即使程序中有成千上万条命令,重复执行它们也很容易。

FoxPro 命令也能直接执行,具体方式是在命令窗口中直接输入一个 FoxPro 命令并按回车键。若想再次执行这一命令,同样是将光标移到该命令所在行并按回车键。

因为命令窗口是一个特殊的编辑窗口,所以可以用 FoxPro 提供的编辑工具编辑其中的命令。也就是说,可以编辑、插入、删除、剪切、拷贝或粘贴命令窗口中出现的命令。

读者如需了解命令窗口的更详细的介绍,请参看《FoxPro 用户手册》中“窗口菜单”这一章。

1.2 创建程序

FoxPro 程序是由一系列 FoxPro 命令组成的,其数据处理能力极强。

用户能够通过将一系列命令输入到一个 FoxPro 程序文件中来执行它们。

运行一个程序要比在命令窗口中逐条输入诸命令优越,例如:

- 程序可自动地执行多条命令,这就节约了大量的输入工作和时间;
- 程序可修改和重复执行;

- 程序可以运行另一个程序；
- 只有程序才能使用一些特定的结构化编程命令；
- 在 FoxPro 中运行的程序具有非并行的特点。

程序实际上就是包含一系列命令的文本文件。在 FoxPro 中创建一个程序有下述两种途径：

- 在 File 菜单中选择 New...，随后便会出现 New 对话框，在该框中选择 Program 单选钮，然后选择 OK（在 FoxPro for MS-DOS 中）或 New（在 FoxPro for Windows 中）；
- 在命令窗口中输入命令 MODIFY COMMAND，然后按回车。

此时，FoxPro 会打开一个叫 UNTITLED.PRG 的新窗口。现在，就可以在这个窗口中输入程序。

注意：如果读者已经交互式使用过 FoxPro 并且想把所生成的命令保存起来，可以直接将它们拷贝或粘贴到程序编辑窗口中。甚至还可以从帮助文件中拷贝示例程序，将它们粘贴到一个程序中。从帮助文件中拷贝示例程序是学习 FoxPro 的命令和函数的便捷方法之一。

1.2.1 保存程序

程序创建后，当然需要保存。如果保存的是一个未曾命名的程序，则应在 File 菜单中选择 Save As...，然后在 Save As 对话框出现时输入该程序的名字。程序一经保存，便可执行或修改。

1.2.2 修改程序

程序保存以后，还可以对它进行修改。首先必须打开该程序，这可以采取下述几种方法之一：

- 在 File 菜单中选择 Open...，此时，会出现一个对话框，其中列出了可用文件。在 Type 弹出式菜单中选择 Program，在文件列表中选择要修改的程序。然后选择 OK（在 FoxPro for MS-DOS 中）或 Open（在 FoxPro for Windows 中）。
- 在命令窗口中输入命令 MODIFY COMMAND，再输入一个空格和待修改的程序名，最后按回车键。随后程序就被打开以供编辑。例如，要修改一个名叫 MYPROGRAM 的程序，应在命令窗口中输入下述命令：

```
MODIFY COMMAND MYPROGRAM
```

- 在命令窗口中键入命令 MODIFY COMMAND ?，然后按回车。此时即可从文件列表中选择要修改的程序，最后，选择 OK（在 FoxPro for MS-DOS 中）或 Open（在 FoxPro for Windows 中）。

程序打开后，就可以执行它了。可以采取下述两种方法之一执行程序：

- 在 Program 菜单中选择 Do...，再从程序列表中选择待执行的程序，然后选择 OK（在 FoxPro for MS-DOS 中）或 Open（在 FoxPro for Windows 中）。
- 输入命令 Do，后跟一个空格和待执行的程序名，然后按回车键。例如，要执行名为 MYPROGRAM 的程序，应在命令窗口中输入下述命令：

```
DO MYPROGRAM
```

1.2.3 其他程序信息

如需了解关于程序文件的创建、编辑和保存的完整介绍,请参看《FoxPro 用户指南》中“用户接口基础知识”一章里“文本编辑”这一节的内容。至于程序文件执行方面的介绍,则请参看《FoxPro 用户指南》中“程序菜单”一章的内容,或本书中有关 DO 命令的说明。

1.3 FoxPro 的联机帮助工具

在 FoxPro 中,帮助工具随手可得。联机帮助工具能够提供有关每个 FoxPro 命令、函数和系统内存变量的特定信息,以及有关 FoxPro 的一般的信息。

在 FoxPro 系统的窗口、对话框和菜单选项中还可获得上下文有关的帮助。当某个系统窗口或对话框在屏幕上被激活时,按 F1 功能键即可得到相应的帮助信息。而在命令窗口或程序中,如想得到某一命令或函数的帮助信息,应先选择该命令或函数,然后再按 F1 功能键。

1.3.1 FoxPro for MS-DOS 中的联机帮助

在 FoxPro for MS-DOS 中,若想获得 FoxPro 的联机帮助,应在 System 菜单中选择 Help... 或按下 F1 功能键,此时弹出的 Help 窗口列出了许多主题,再选择相应的主题,得到有关该主题的详细帮助信息。

如果想得到当前帮助的相关主题的帮助信息,可在 See Also 弹出式菜单中选择正文并选择 LOOK UP 下推钮。如果所选择的文本是一个帮助主题,该主题的帮助信息就会在屏幕上显示出来;如果所选文本并不对应于某一帮助主题,则会弹出一个对话框,其中列有所有帮助主题。

有关 FoxPro for MS-DOS 中联机帮助的更详细介绍,请参见《FoxPro 用户指南》中“系统菜单”一章的内容,或本书中有关 SET HELP 命令的说明。

1.3.2 FoxPro for Windows 中的联机帮助

在 FoxPro for Windows 中,若想获得 FoxPro 的联机帮助,应在 HELP 菜单中选择 HELP... 或按 F1 功能键,这样就可以打开帮助窗口。

FoxPro for Windows 支持两种类型的帮助方式:

- DBF 型的帮助,适于 MS-DOS 用户;
- Windows 型帮助,适于 Windows 用户。

在缺省情况下,系统使用 Windows 型的帮助方式。

在 Command 窗口或程序中,可以来回切换这两种类型的帮助方式。具体方法如下:

- 为将 FoxPro 的 Windows 型帮助方式变为 DBF 型帮助方式,在 Command 窗口或程序中发出 SET HELP TO FOXHELP.DBF 命令。除了在想显示 Help 窗口时需要选择 Help 菜单中的 Control 外, FoxPro 的 DBF 型帮助等同于 FoxPro for MS-DOS 的联机帮助。
- 为将 DBF 型帮助方式变为 Windows 型的帮助方式,在 Command 窗口或程序中发出 SET HELP TO FOXHELP.HLP 命令。

在 Windows 型帮助方式下:

- 无论何时,按下 F1 功能键都可显示出有关当前窗口或对话框的帮助信息;
- 若想得到有关菜单选项的帮助,先按 Shift+F1 键,然后选择该菜单选项;
- 若想找到与某一单词或短语相关的主题的帮助信息,在 HELP 菜单中选择 Search for Help on...;
- 为加入某一帮助主题的说明或注释,在 Help 窗口的 Edit 菜单中选择 Annotate,然后输入注释内容。注释的主题前有一个剪纸符号,将鼠标移到该符号上并单击鼠标按钮,即可看到具体的注释;
- 给一个经常使用的帮助主题作上标记能够更快地访问它。如想这样做,应在 Help 窗口的 BookMark 菜单中选择 Define...,然后输入当前主题的书签名。所定义的书签名将被加到 BookMark 菜单的底端;
- 在 Help 窗口的 File 菜单中选择 PrintTopic,可将帮助主题打印出来;
- 如想将帮助主题的内容拷贝到剪辑板中,可在 Help 窗口的 Edit 菜单中选择 Copy...,然后再选择想拷贝的帮助主题。

若想更详细地了解 FoxPro for MS-DOS 的联机帮助工具,请参看《FoxPro for Windows 用户指南》中“帮助菜单”一章。如果是 MS-DOS 用户,请参看《FoxPro for MS-DOS 用户指南》中“系统菜单”一章,或本书中有关 SET HELP 命令的内容。

1.4 FoxPro 语言的组成

FoxPro 语言由下列两要素组成:

- 实现某种操作的命令。例如,使用 BROWSE 命令可以看到一张表或一个数据库的内容;
- 返回一个数值的函数。例如,函数 DATE() 返回计算机的系统日期。函数带一对圆括号,系统就是通过这对圆括号将函数同命令区别开来的。

命令和函数组合起来构成 FoxPro 的语句。函数不能单独使用,它们一般都组合在一个 FoxPro 命令中。

例如,问号(?)是一个向屏幕输出的命令,DATE()是一个返回系统当前日期的函数,它们组合起来能够向屏幕输出系统日期:

```
? DATE()
```

又如,STORE 是向内存变量赋值的命令,TIME()是返回当前系统时间的函数,它们组合后则将系统时间存入内存变量 BEGINTIME 中:

```
STORE TIME() TO begintime
```

1.4.1 语法规则

FoxPro 的命令和函数均由一个或多个成分组成,这些成分决定了该命令或函数的使用方式。

先看一个典型的 FoxPro 命令的语法形式。REPLACE 命令可用于修改一个表或数据库记录的字段,该命令的语法形式如下:

```

REPLACE <field1> WITH <expr1> [ ADDITIVE]
[, <field2> WITH <expr2> [ ADDITIVE]]
...
[ <scope> ][FOR <expL1> ][WHILE <expL2> ]
[NOOPTIMIZE]

```

下面我们列出了 REPLACE 命令的诸成分：

关键字	REPLACE, WITH, ADDITIVE, FOR, WHILE, NOOPTIMIZE
表达式	expr1, expr2, expL1, expL2
名	field1, field2

本书在给出命令和函数的语法形式时使用了特定的排印方面的约定,通过这些约定表明命令或函数的哪些成分是必需的或是可选的,哪些成分必须按照指定格式输入等等。

- 大写 大写形式表明它是一个 FoxPro 的关键字。每个 FoxPro 的命令或函数都至少有一个关键字。FoxPro 就是通过关键字来区别每一个命令或函数的。因为 FoxPro 为内部使用而保留了这些关键字,所以关键字也叫做保留字。在前述 REPLACE 命令中,关键字有 REPLACE (FoxPro 通过它知道这是一个命令), WITH, ADDITIVE, FOR 和 WHILE。其中 WITH, ADDITIVE, FOR 和 WHILE 也叫做子句,因为它们为主命令的附加部分。
- () 用尖括号括住小写字母文本,表示这些小写字母文本的具体内容必须由用户提供。这些文本信息可以是文件名、表达式或内存变量名等。文本和尖括号应被恰当的名称或表达式所取代。在输入命令的时候应去掉尖括号。
- [] 方括号表示所括住的命令或函数内容是可选的。同样,在输入的时候也不要带上方括号。例如, REPLACE 命令中就有相当多的可选项,具体选择或不选择这些项由用户决定。像 ADDITIVE, <scope>, FOR <expL1>, WHILE <expL2> 和 NOOPTIMIZE 等子句都括在方括号中,也就是说,它们都是可选项。
- | 用垂直短线分开的可选项表示这些可选项只能选择其中之一。输入时不要带上垂直短线。
- ... 省略号表示一个命令或函数或其中某一部分以类似方式延续下去。例如,函数 INLIST () 需要大量的参数,在它的语法形式中,就是通过省略号来表示它可以包含更多的参数,而并不只是下面所列的 3 个：
- ```
INLIST (<expr1>, <expr2> [, <expr3>, ...])
```

#### 1.4.2 表达式

一个 FoxPro 表达式由下述几项中的一个或多个组成:函数、字段值、内存变量值、数组元素、常量以及操作符。FoxPro 一共有四种类型的表达式:字符型、数值型、日期型和逻辑型。

在 FoxPro 命令和函数中,表达式按如下方式给出：

| 表达式         | 表达式类型       |
|-------------|-------------|
| <expC>      | 字符型         |
| <expN>      | 数值型         |
| <expD>      | 日期型         |
| <expL>      | 逻辑型         |
| <expr>      | 上述四种表达式之一   |
| <expr list> | 由逗号分隔的多个表达式 |

如果在同一个命令或函数中有多个相同类型的表达式,则在表达式后加上数字,以表明它在该命令或函数中的位置。例如,函数 CHRTRAN() 的语法形式如下:

CHRTRAN (<expC1>,<expC2>,<expC3>)

该函数就需要有 3 个字符型表达式作为参数:<expC1>,<expC2>,<expC3>。

当字段、函数、内存变量和数组元素组合成一个表达式时,它们必须具有相同的数据类型。如果不是相同类型,FoxPro 会显示“Operator/Operand type mismatch”(操作符或操作数类型不匹配)的错误信息。

#### 1. 字符表达式

字符表达式由下述内容组成:

- 字符型字段;
- 返回字符值的函数;
- 由字符数据组成的内存变量或数组元素;
- 字符型常量,也叫字符串文字量。

所谓字符串文字量,就是用单引号或双引号引起来的一个字符串。例如,‘washington’或“Washington”。引号必须匹配,也就是说,不能在字符串的开始用单引号而在结束时用双引号。

如果在字符串中也需要使用引号,那么串中用到的引号就不应该与标志这个字符串的引号相同。例如,“Don’t touch!”就是一个合法的字符串文字量,而‘Don’t touch!’则不行。

注意:在 FoxPro 的文档中经常会提及“空串”。空串就是长度为零的字符串,表明它不含有任何字符。空串用中间没有空格的一对引号表示:‘’或“”。

下述操作符可用在字符表达式中:

#### 字符表达式操作符

| 操作符 | 对应的操作                         |
|-----|-------------------------------|
| +   | 字符表达式连接,将两个字符串简单地拼在一起         |
| —   | 字符表达式连接,将第一字符串的尾部空格去掉后连接第二字符串 |
| \$  | 字符表达式比较                       |

使用字符串表达式操作符可完成如下工作:

- 连接字符串,例如:  
‘Good ’ + ‘Morning!’
- 连接字符串和字段,例如:

"Please respond by " + invoice.date

- 连接两个字段并去掉尾部空格,例如:

Customerfirst—Customer.last

- 连接字符串和内存变量,例如:

"Dear " + mfirstname

## 2. 数值表达式

数值表达式由下述内容组成:

- 数值型字段;
- 返回数值型值的函数;
- 由数值型数据组成的内存变量和数组元素;
- 数值型常量。

数值型常量是一些数值,如 0,1,-1,3.1415 等等。

数值型表达式操作符(按优先级顺序排列)

| 操作符  | 对应的操作       |
|------|-------------|
| ( )  | 用以组合表达式的圆括号 |
| *,^  | 指数运算        |
| *,/  | 乘法和除法       |
| %    | 取模(余数)      |
| +, - | 加法和减法       |

## 3. 日期表达式

日期表达式由下述内容组成

- 日期型字段;
- 返回日期值的函数;
- 由日期型数值构成的内存变量和数组元素;
- 日期型常量。

日期型常量由数值构成,如 1,2,3,100 等等。日期型常量描述的是天数,可以对一个给定的日期加上或减去一个日期型常量(天数)。

定义一个日期时,日期必须用花括号括起来。例如,下述命令定义了一个日期型内存变量 XMAS,并将日期 12/25/92 存在其中:

```
STORE {12/25/92} TO xmas
```

下述两个命令都定义了一个空的日期型内存变量 BLACKDATE:

```
STORE{ } TO blankdate
```

```
STORE{ } TO blankdate
```

FoxPro 视无效日期为空日期。

#### 4. 逻辑表达式

逻辑型表达式的值是真或假之一。在 FoxPro 中,用 .T. 和 .F. 来定义真和假这两个值。

逻辑型表达式由下述内容组成:

- 逻辑型字段;
- 返回逻辑值的函数;
- 由逻辑值组成的内存变量或数组元素;
- 由特殊的关系运算符分隔开的其它表达式类型(如字符型、数值型或日期型)。例如:  
?.voice.date>={06/01/92}AND invoice amount>15000

FoxPro 按从左到右的顺序求逻辑表达式的值,并且只进行必要的计算。也就是说,并不是每一项都计算。在上述例子中,通过 AND 操作符定义了一个逻辑表达式,只要表达式中有一个值是假(.F.),则整个表达式的值就是假。因此,当 FoxPro 发现第一个为假的逻辑值时,它就不会再去计算这个表达式了:

.T. AND .T. AND .F. AND .T. AND .T. AND .T.

#### 逻辑表达式操作符(按运算优先级顺序)

| 操作符   | 对应的操作       |
|-------|-------------|
| ( )   | 用以组合表达式的圆括号 |
| !,NOT | 逻辑非         |
| AND   | 逻辑与         |
| OR    | 逻辑或         |

逻辑表达式的关系操作符可用于所有的数据类型,并返回一个逻辑值。下面中的关系操作符均能完成这一功能:

#### 逻辑表达式关系操作符

| 操作符       | 对应的操作 |
|-----------|-------|
| <         | 小于    |
| >         | 大于    |
| =         | 等于    |
| <>, #, != | 不等于   |
| <=        | 小于等于  |
| >=        | 大于等于  |
| ==        | 字符串比较 |

下述命令表明了关系操作符是怎样用于数值型、日期型和字符型数据的:

```
? 1>2 && 显示 .F.
? {01/01/92}<={01/01/92} && 显示 .T.
```

#### 5. 名

有些 FoxPro 的命令和函数需要用户提供一个名。可在 FoxPro 中使用的名有:

- 表/.DBF 文件名；
- 表/.DBF 别名；
- 表/.DBF 字段名；
- 索引文件名；
- 文件名；
- 内存变量名和数组名；
- 窗口名；
- 菜单名；
- 弹出式菜单名。

在下表中，左列是在命令和函数的语法说明中给出的名，右列说明了应在命令和函数中提供的对应内容：

| 名             | 说明         |
|---------------|------------|
| <file>        | 文件名        |
| <field>       | 字段名        |
| <field list>  | 字段名列表      |
| <array>       | 内存变量数组名    |
| <memvar>      | 内存变量或数组元素名 |
| <memvar list> | 内存变量名列表    |
| <menu name>   | 菜单名        |
| <pad name>    | 菜单笺小键盘名    |
| <popup name>  | 弹出菜单名      |
| <popup list>  | 弹出菜单名列表    |
| <window name> | 窗口名        |
| <window list> | 窗口名列表      |

使用名列表时，名与名之间必须用逗号分隔。名不是表达式，也不是内存变量、数组元素或字段。名不能用引号包起来。

例如，在 REPLACE 命令中，必须至少包含一个字段名，例如 field1。

## 6. 名表达式

尽管名不是内存变量或数组元素，但是可以定义一个名表达式，以代替同名的字符内存变量或数值元素的值。

名表达式为 FoxPro 的命令和函数提供了灵活性。

将名存放在内存变量(或数组元素)中，以后就可以在命令或函数中用内存变量来代替该名，只要将内存变量括在圆括号中就行了。

例如，REPLACE 命令需要一个字段名，可以将这个字段名存在一个内存变量中，然后在 REPLACE 命令字段名出现的地方用名表达式来代替它：

```
STORE 'city' TO mvarcity
REPLACE (mvarcity) WITH 'Redmond'
```

这里,字段名 CITY 被存在内存变量 MVARCITY 中,在使用 REPLACE 命令时,名表达式 (mvarcity)将用字段名代替内存变量。

借助于称为宏代换的处理,可以用内存变量来代替名。在宏代换中,内存变量前应加上一个“&”符号,以通知 FoxPro 该内存变量的值将作为一个名使用。如需了解更多的有关宏代换的信息,请参看本书中有关 & 主题的内容。含有名表达式的命令或函数比含有宏代换的命令或函数执行得更快,因此,可能的话应尽量使用名表达式,而不要使用宏代换。

#### 1.4.3 缩写说明

下表给出了描述 FoxPro 命令和函数时常用的缩写和术语及其含义:

| 缩写             | 含义              |
|----------------|-----------------|
| alias          | 工作区别名           |
| array          | 数组名             |
| borderstring   | 边界定义字符串         |
| column         | 屏幕或窗口列          |
| command        | FoxPro 命令       |
| delimiter      | 定界符(分隔符)        |
| dir            | 目录              |
| drive          | 磁盘驱动器名          |
| key            | 索引表达式           |
| list           | 表达式、字段等的集合      |
| macro name     | 键盘宏名            |
| memo field     | 备注字段名           |
| memvar         | 内存变量或数组元素       |
| pad name       | 菜单笔名            |
| parm list      | 参数列表            |
| path           | 磁盘目录说明          |
| port           | 串行或并行输出端口名      |
| procedure name | 过程文件名           |
| row            | 屏幕或窗口行          |
| scope          | 表或、DBF 记录的范围    |
| skel           | 文件说明框架          |
| statements     | FoxPro 命令集合     |
| textlines      | 非 FoxPro 命令的正文  |
| topic          | FoxPro 帮助文件中的主题 |
| work area      | 表或、DBF 的工作区     |

#### 1.4.4 用 Scope, FOR 和 WHILE 指定记录范围

当某一命令对一个表或数据库中的记录进行操作时,通常可以在命令中指明该命令作

用的范围。可通过 Scope, FOR 和 WHILE 子句指定记录范围。

当 Scope 命令中包含 Scope 子句时, 该命令只对表或数据库中指定的一部分记录起作用。可以使用下述子句之一来指明记录的范围:

- ALL——命令对表或数据库中所有记录起作用;
- NEXT(expN)——命令作用于从当前记录开始到当前记录后(expN)条记录为止的这一范围内的记录。NEXT1 表示仅作用于当前记录。
- RECORD(expN)——命令作用于从当前记录开始直到表或数据库的最后一个记录为止的这一范围内的记录。

还可以用 FOR 和 WHILE 子句来指定命令 FOR(expL)作用的记录范围。当使用 FOR 子句时, 命令作用于每个符合指定条件(expL)的记录。

Rushmore 技术和 FOR 子句

Rushmore 技术是一种数据存取技术, 它可以高效地访问许多记录的集合, 而速度可以与存取带索引的单个记录相媲美。借助于这种技术, 一些复杂的数据库操作能快上百上千倍。FoxPro 使得个人计算机能够管理含有数百万条记录的巨型数据库, 而速度却同大型机数据库系统相仿。

在最简单的形式下, Rushmore 技术通过 FOR 子句根据现有的索引提高单个表或数据库命令的执行速度。在任何 FOR 子句出现的地方都可以利用这一技术, 它的运行速度同被检索的记录个数成正比。

可对 FOR 子句进行优化的命令

|               |         |         |        |
|---------------|---------|---------|--------|
| AVERAGE       | COUNT   | JOIN    | REPORT |
| BROWSE        | DELETE  | LABEL   | SCAN   |
| CALCULATE     | DISPLAY | LIST    | SORT   |
| CHANGE        | EDIT    | LOCATE  | SUM    |
| COPY TO       | EXPORT  | RECALL  | TOTAL  |
| COPY TO ARRAY | INDEX   | REPLACE |        |

Rushmore 技术提供的数据库检索优化技术的优劣取决于 FOR 子句表达式。FOR 子句表达式可以很简单也可以很复杂, 而数据检索的速度可以通过优化的 FOR 子句表达式来得到提高。

可优化表达式可通过 AND, OR 和 NOT 等逻辑操作符进行组合。在构成复杂的表达式时, 这个复杂表达式也可能是可优化的。

基本的可优化表达式是下述两种形式之一:

(index expression) (relational operator) (constant expression)

或

(constant expression) (relational operator) (index expression)

在一个基本的可优化表达式中:

- $\langle \text{index expression} \rangle$ , 即 $\langle \text{索引表达式} \rangle$ , 必须同用来构建索引的表达式完全相匹配, 并且  $\langle \text{index expression} \rangle$  不应该包含别名。
- $\langle \text{relational operator} \rangle$ , 即 $\langle \text{关系操作符} \rangle$ , 应该是下述几述符号之一:  
 $<, >, =, <=, >=, <>, \#, ! =$ 。
- $\langle \text{constant expression} \rangle$ , 即 $\langle \text{常量表达式} \rangle$ , 可以是任何表达式, 包括内存变量和不相关的表或库中的字段。

例如, 如果在下述表达式上建有索引:

```
firstname
custno
UPPER (lastname)
hiredate
addr
```

那么下面列出的就是基本的可优化表达式:

```
firstname = 'Fred'
custno >= 1000
UPPER (lastname) = 'SMITH'
hiredate = {12/30/90}
```

如果发出命令:

```
STORE 'WASHINGTON AVENUE' TO X
```

那么下述表达式亦是基本的可优化表达式:

```
addr = X
addr = SUBSTR (X, 8, 3)
```

有关 `rushmore` 技术的详细介绍, 请参见《FoxPro 开发指南》中“优化应用程序”一章。

#### WHILE $\langle \text{expL} \rangle$ 表达式

WHILE 表达式将导致命令作用于逻辑表达式 $\langle \text{expL} \rangle$ 值为真的每一个记录。一碰到表达式的值为假, 就立即停止执行命令, 不再理会后面的记录。这种表达式一般用于经过排序或索引的数据库。

Scope, FOR 和 WHILE 表达式能够同时用于同一个 FoxPro 命令。当同时有 FOR 和 WHILE 子句存在时, WHILE 表达式有较高的优先权, 而 FOR 子句用来过滤由 WHILE 挑选出的记录。

#### 1.4.5 表或数据库的工作区

在 FoxPro for Windows 和 FoxPro for MS-DOS 的 32 位扩充版本中, 可以在 255 个工作区中打开和处理数据库。工作区可用数字 1 到 255 或在其中打开的表/.DBF 的别名来区分, 其中, 头 10 个工作区还可利用字母 A 到 J 来命名。

在 FoxPro for MS-DOS 的标准版本中, 可以在 25 个工作区中打开和处理数据库。每一个工作区可用数字 1 到 25 或其中打开的表/.DBF 的别名来区分, 其中, 头 10 个工作区还可利用

字母 A 到 J 来命名。

如果要在非当前工作区中打开一个表/.DBF,则必须指定或者选择相应的工作区。当一个表/.DBF 打开时,该表/.DBF 就有了一个别名,可以用它来识别这个数据。除非另外指定,否则别名就是这个表/.DBF 的名(不包括.DBF 后缀)。

#### 缺省的别名

如果用下述命令在工作区 A 中打开名为 CUSTOMER.DBF 的文件,则 FoxPro 将自动指定别名为 CUSTOMER。

```
SELECT A
USE customer
```

随后即可在命令或函数中用这个别名指定该表/数据库。

#### 用户定义的别名

用户可以在打开数据库文件的同时为其定义一个别名。例如可用下述命令打开数据库 CUSTOMER.DBF(在工作区 A 中),并给它定义一个别名 people:

```
SELECT A
USE customer ALIAS people
```

随后即可通过别名 PEOPLE 来指定该数据库。一个别名由最多 10 个字母、数字或下划线组成,并且必须以字母或下划线打头。如果在用户指定的别名中有无效字符,FoxPro 会自动地创建一个别名。

详情请参看本书中有关 USE 命令的说明。

#### FoxPro 定义的别名

在某些情况下,FoxPro 将自动地给表/.DBF 定义一个别名。例如,如果同时在多个工作区中打开一个数据库,并且是用带关键字 AGAIN 的 USE 命令打开的,而在每个工作区中打开时又没有指定别名的话,FoxPro 就会自动地定义一个缺省的别名。

在 FoxPro for Windows 和 FoxPro for MS-DOS 的 32 位扩充版本中,在头 10 个工作区里指定的缺省别名是字母 A 到 J,在从 11 到 255 的工作区里指定的缺省别名是 W11 到 W255。

在 FoxPro for MS-DOS 的标准版本中,在头 10 个工作区里指定的缺省别名字母 A 到 J,而在从 11 到 25 的工作区里指定的缺省别名 W11 到 W25。

如果别名发生了冲突,则代之以缺省别名。

#### 使用别名

如果 CUSTOMER.DBF 在工作区 C(或工作区 3)中打开,并且采用缺省的别名 CUSTOMER,则发出下述命令之一即可从其他任何工作区转到该工作区:

```
SELECT C
SELECT 3
SELECT customer
```

用户还可以在其他工作区中通过在字段名前加上别名指针来引用该工作区中的字段。具体方法就是在字段名前加上相应的别名或代表工作区的字母,然后跟一个句点或一个“—>”符号。

如果在工作区 C 中想访问在工作区 A 中打开的 CUSTOMER.DBF 的 CONTACT 字段,可使用下述内容之一:

```
customer.contact
```

```
A.contact
```

如果当前不在工作区 A 中,而 CUSTOMER.DBF 是在工作区 A 中以 PEOPLE 作为别名打开的,则可使用下述命令之一:

```
people.lastname
```

```
A.lastname
```

引用一个工作区

在打开一个数据库之前可以先引用(即进入其中)一个工作区,在 SELECT 后而写上工作区的字母或数字即可,如下所示:

```
SELECT A
```

```
SELECT 1
```

内存变量

有时候内存变量名会偶然地同字段名相同。如果出现这种情况,FoxPro 一般使字段名具有更高的优先权。在指定一个同某字段同名的内存变量时,可在变量名前加上一个由 M 后跟一个句点或—>符号的前缀,例如:

```
M.lastname
```

#### 1.4.6 函数和备注字段

FoxPro 的备注字段能够包含任何数据,可以为空,也可以是汇编源代码、打印机驱动程序等等,并且这些内容可以是任意长度。但并不是所有的字符串和字符函数都可用于备注字段。仅有下述函数可以直接作用于不限长的备注字段:

```
$
```

```
LEN()
```

```
AT()
```

```
ATC()
```

```
ATLINE()
```

```
ATCLINE()
```

```
MEMLINES()
```

```
MLINE()
```

```
SUBSTR()
```

另外,利用 REPLACE 命令,可用一个任意长的备注字段或其子串来代替另一个备注字段。

备注字段能够以内存驻留字符串的形式存储在此之后,就可用所有的字符处理函数对这些字符串进行处理。

在 FoxPro for MS-DOS 的标准版本中,字符串长度最大是 64K,而在 FoxPro for Windows 和 FoxPro for MS-DOS 的 32 位扩充版本中,其最大长度则只受可用内存的限制。