

# 第一章 FoxPro 数据库语言简介

## 1.1 FoxPro 的特点

FoxPro 是 FoxSoftware 公司自 FoxBASE+1.12,2.00,2.10 等软件之后,推出的 Fox 数据库应用系列的又一新产品,它具有对信息的分类、检索和处理等功能。FoxPro 具备了一个优秀的微机数据库系统所应该具备的几乎所有的特点。

- 易于使用,初学者也能自如地使用及进行工作。
- 提供足够的和强有力的编程特点。
- 有一个良好的图形界面和窗口功能,平滑且反应敏捷,且速度很快。
- 具有一个功能很强的内部文本编辑器。
- 用户已有的全部程序(包括FoxBASE+和dBASEIII程序)能在其环境下完整地工作。
- 使用户能访问数千个预先编写好的程序。
- 具有较为完善的真编译功能,可编译为直接在操作系统下运行的.exe文件。
- 提供操作方便且处理范围广泛的纠错机制。
- 具有较强的硬件适应性,只需要标准较低的硬件配置。
- 能根据用户要求重新定义键盘。
- 具有一个内部报表生成器,它比独立的报表生成器更易使用且功能更强。
- 用户真正可以用来处理大量的数据:数十万甚至数百万条记录。
- 比任何其它的微机数据库管理系统都快(比dBASEIV快8倍,比dBASEIII Plus快16倍)。

此外, FoxPro 还具备了这样一些特点:

- FoxPro与FoxBASE+百分之百地兼容,并且与dBASEIV兼容。
- FoxPro是目前DOS系统中唯一能支持鼠标器的微机数据库管理系统,它对鼠标器提供自动透明的支持,且没有鼠标时其工作也一样好。
- FoxPro对FoxBASE+语句做了大量的扩展,例如, FoxPro 2.00比FoxBASE+2.00多120余条命令,多44条set命令,多142个函数和增加了42个系统内存变量等。
- FoxPro可在最小和最经济的硬件配置上进行操作,可在最小配置为512KB的PC/XT机上运行,也可充分利用用户机器的高速度扩展内存,扩充显示卡模式等全部优点,还能运行在一些复杂的新机器上,即使是在小配置的情况下,它还具有最大和最强硬件配置的全部优点。
- 采用一种称为“Rushmore”(快速查找)的优化技术,可以使对较大的数据库的查找速度比采用普通的查找方法快几百倍,数据库越大,速度优势越明显, FoxPro

还采用一种“Compact”（紧缩）类型的索引方法，建立的索引文件长度缩短 20% 以上，这对于大型数据库来说是十分可观的，在 FoxPro 中，可以建立“Compound”（复合）索引文件，把对应库的所有索引文件都放在唯一的索引文件中，这样在修改数据库时，可以自动修改与之关联的索引文件。

- FoxPro 提供了高级外部程序接口 API，可以方便地实现与 C 语言和汇编语言的连接，这对于 FoxPro 编写大型，高水平，多功能软件提供了十分便利的条件。
- SQL（结构化查询语言）是日趋流行的一种标准关系型数据库管理系统语言，FoxPro 的关系举例检索(RQBE)的功能就是通过 SQL 的 Select 命令处理的，其数据库检索方便灵活，不过目前的 FoxPro 还不全面地支持 SQL。

FoxPro 所具有的强大功能和优异的特点，使它一问世就受到了广大软件开发人员和计算机用户的高度重视，FoxPro 的应用前景十分美好。

FoxPro 2.5 包含一种全新的编码基数，并有许多新的特点。Microsoft 公司声称，他们已经开发成功一种通用核心编码基数，而且 DOS 下和 Windows 下的 FoxPro 2.5 有百分之八十的源码采用了这种编码。这个核心编码基数同时也是将要投放市场的 Macintosh 和 UNIX FoxPro 的基础。然而这种改进很像小汽车里的发动机，当你使用 DOS 下的 FoxPro 2.5 时，它们的卓越性能可能不会轻易地表现出来。Windows 下的 FoxPro 是 Microsoft Windows 环境下的新产品，但 DOS 下的 FoxPro 用户和开发员也应当对 Windows 产品很在行。

尽管乍一看，DOS 下的 FoxPro 2.5 与 FoxPro 2.0 并没有什么明显的不同，事实上 FoxPro 2.5 比 FoxPro 2.0 的运行速度更快，它以更快的速度进行数据库的管理。写屏速度也比 2.0 版本快，并且许多在网络环境下的操作要更快一些。足以让人兴奋的是，如果你至少有 8 兆内存，则 Windows 下的 FoxPro 2.5 与 DOS 下的运行速度几乎一样快。

Windows 下的 FoxPro 和 DOS 下的 FoxPro 2.5 的增强版，现在可支持多达 225 个工作区。不幸的是，DOS 下的 FoxPro 2.5 的标准版仍只支持 25 个工作区。下表提供的是 DOS 下的 FoxPro 2.0 与 FoxPro 2.5 系统参数的比较。

|             | FoxPro 系统指标    |                | Windows 下的 FoxPro |
|-------------|----------------|----------------|-------------------|
|             | FoxPro 2.5 标准版 | FoxPro 2.5 增强版 |                   |
| 每个数据库中最大记录数 | 10 亿个          | 10 亿个          | 10 亿个             |
| 每条记录中所含字符数  | 65000 个        | 65000 个        | 65000 个           |
| 每条记录中最多字段个数 | 255            | 255            | 255               |
| 每个字段的最多字符数  | 254            | 254            | 254               |
| 最多工作区个数     | 25             | 225            | 225               |
| 每个 IDX 索引关  |                |                |                   |

|                                                   |                 |              |         |
|---------------------------------------------------|-----------------|--------------|---------|
| 键表达式长度                                            | 100 个字符         | 100 个字符      | 100 个字符 |
| 每个 CDX 索引关键字                                      |                 |              |         |
| 表达式中字符数                                           | 240             | 240          | 240     |
| 每个数据库可打开索引文件的最大个数                                 | 只受可用内存与可用文件柄的限制 |              |         |
| 关系表达式的                                            |                 |              |         |
| 最大长度                                              | 不限              | 不限           | 不限      |
| 字符型字段的最大长度                                        | 254             | 254          | 254     |
| 字段名中最多字符个数                                        | 10              | 10           | 10      |
| 数值计算的精度                                           |                 |              |         |
| (小数位)                                             | 16              | 16           | 16      |
| 缺省内存变量个数                                          | 256             | 256          | 256     |
| 最多内存变量个数                                          | 3600            | 65000        | 65000   |
| 数组的最多个数                                           | 3600            | 65000        | 65000   |
| 每个数组中元素的最多个数                                      | 3600            | 65000        | 65000   |
| 源文件中最多程序行数                                        | 不限              | 不限           | 不限      |
| 单个过程或自定义函数的长度                                     | 64K             | 64K          | 64K     |
| 每个文件中过程或自定义函数的最多个数                                | 不限              | 不限           | 不限      |
| DO 命令的最多嵌套层数                                      | 32              | 32           | 32      |
| READ 命令的最多嵌套层数                                    | 5               | 5            | 5       |
| 结构化程序命令的最多嵌套层数(如 IF / ENDIF, DO WHILE / ENDDO...) | 64              | 64           | 64      |
| 可打开窗口的最大个数                                        | 只受可用内存和文件柄的限制   |              |         |
| 可打开的 BROWSE 窗口的最多个数                               | 25              | 225          | 225     |
| 每个程序行最多字符个数                                       | 2048            | 2048         | 2048    |
| 可打开文件的最多个数                                        | 99              | 只受 MS-DOS 限制 |         |
| 每个键盘宏的最多击键次数                                      | 1024            | 1024         | 1024    |
| 一个 SQL SELECT 命令的最多可选字段数                          | 255             | 255          | 255     |
| 每个颜色组中颜色模式的最大个数                                   | 24              | 24           |         |
| 颜色组的最多个数                                          | 不限              | 不限           | 不限      |

|                    |               |     |     |
|--------------------|---------------|-----|-----|
| 每个颜色模式中最<br>多颜色数   | 10            | 10  |     |
| 用户可定义的颜色<br>模式个数   | 8             | 8   |     |
| 报表格式定义中的<br>最多目标个数 | 只受可用内存和文件柄的限制 |     |     |
| 报表格式文件中<br>多程序行数   | 255           | 255 | 255 |
| 报表格式的最大层数          | 20            | 20  | 20  |

注：如果在单用户方式下或在网络环境的独占方式下运行，则一个数据库的实际长度不能超过 20 亿个字节。这个限制也适用于采用结构化 CDX 索引的共享 DBF 文件。那些没有索引或采用 IDX 索引的共享文件的长度不能超过 10 亿个字节。

FoxPro 2.5 与 FoxPro 2.0 的 FOXUSER 资源文件的格式是有所不同的。为了在 FoxPro 2.5 下使用 FoxPro 2.0 的颜色集，用户应当首先运行由 Microsoft 公司提供的 FIXUSER.APP 文件。这个文件可在你的 DOS 下的 FoxPro 2.5 根目录下的子目录 \GOODIES\MISC 中找到。一旦运行 FIXUSER.APP，上述颜色集就不能在 FoxPro 2.0 下运行了，因此，在你进行转换之前一定要作 FOXUSER 文件的备份。

DOS 下和 Windows 下的 FoxPro 2.5 引进了一种称为通用(general)类型的新的字段类型。其目的是为了存储 Windows 下的 FoxPro 所使用的位图(BMP)文件，OLE 目标文件和 WAV 声音文件。不过，你也可以创建一个包含通用字段的数据库文件，包含通用字段的数据库也可以被 FoxPro 2.0 存取，但它将通用字段当作一种未知字段类型来处理。

DOS 下的 FoxPro 2.5 增强版对内存管理作了本质上的重要改进。DOS 下的 FoxPro 2.5 增强版或已编译为增强版 EXE 文件的用户程序，可以在 Microsoft Windows 下运行。DOS 下的 FoxPro 2.5 增强版与 2.0 相比，在运行时只需要更少的可用扩展内存，而且不需要进入请求式页面调度处理。DOS 下的 FoxPro 2.5 增强版不再需要转换 API 库的外部文件，因为这种转换已可在它内部完成。

## 1.2 FoxPro 2.5 的几个版本

FoxPro 2.5 软件（不论单用户版本还是局部网版本）实际上均可以再划分为两种不同的版本（标准与增强型），但每一种版本都只能是在微机上是或局部网中使用，具体应使用哪种版本的 FoxPro 2.5 取决于用户使用的微机（或局部网）的型号和设置情况。

FoxPro 2.5 共有以下几种版本：

- 标准FoxPro 这种版本是为单用户专门设计的、它适用于装有8086/8088或80286处理器的PC如标准的PC或PC/AT等。
- 增强型FoxPro 这种版本适用于32位且具有大容量内存的机器（如80386/80486），该版本要求PC应该具有80386或更高级别的微处理器，另外，建议用户的PC最好还应该至少有3MB的内存(RAM)。
- 标准FoxPro/Lan 这是一种专门供微处理器为8086/8088或80286的局域网工作站

使用的标准 FoxPro 版本。

- 增强型 FoxPro / LAN 这是一种适用于局域网的增强型版本，要求网络文件服务器或工作站至少应配置有：a) 80386 或更高级别的微处理器，b) 至少 3Mb 的内存，只有这样，用户才能在局域网中使用该 FoxPro 版本所提供的网络文件服务功能。

### 1.3 FoxPro 2.5 与 FoxPro 1.xx 的差异

尽管在 FoxPro 1.0 下建立的应用程序不需任何改变就可以在 FoxPro 2.5 下运行，但如下的差异还是需要注意的。

- 键盘宏——在早期的 FoxPro 版本下定义的键盘宏在 FoxPro 2.5 版本下也许不能工作因为一些菜单结构和人机对话界面已作了改进。
- 使用数组的源程序——在以前的 FoxPro 版本中，dimension 命令用来建立数组并将其所有元素都初始化为 (.F.)，FoxPro 2.5 版本中，数组元素保持原样，但要把 FoxPro 1.0 版本中的数组说明命令作一些改变，即先释放掉数组，然后重新作说明。
- FXP 文件——在 FoxPro 1.02 中编译过的文件，若放在 FoxPro 2.5 中运行，会自动重新进行编译，但 FoxPro 1.xx 不会自动重新编译 FoxPro 2.5 的目标文件，因此，若需将 FoxPro 2.5 的程序拿到 FoxPro 1.xx 中运行，最好删除目标文件或进行重新编译。
- 标签文件——FoxPro 1.xx 下的标签文件可以在 FoxPro 2.5 下运行，但在 FoxPro 2.5 下生成的标签文件，若拿到 FoxPro 1.xx 中运行，会产生如下的错误信息：

Label file invalid

- 标签文件——在 FoxPro 1.xx 下生成的报表文件可以不加改变地在 FoxPro 2.5 下运行，但反过来就将产生如下的错误信息：  
report file invalid
- 若通过传地址的方式传送数组，则整个数组都可传过去，若在 FoxPro 1.xx 中通过传值的方式传递数组，则只能传递数组的第一个元素。
- FoxPro 1.xx 的一个以 "@" 开头的过程或程序若在 FoxPro 2.5 中重新编译，会产生错误，也就是说，若想在 FoxPro 2.5 下编译你的 FoxPro 1.xx 过程或程序，则在该过程或程序的开头处不要使用 "@" 字符。

### 1.4 FoxPro 与 FoxBASE+ 的差异

尽管 FoxPro 与 FoxBASE+ 尽可能地完全兼容，但当你想在 FoxPro 下运行 FoxBASE+ 的应用程序时，必须考虑尽量避免使用如下范围内的命令，或使用做一些修改。

#### 1. 错误报告

FoxPro 可以探测到一些被 FoxBASE+ 忽略掉的错误，如果你的 FoxBASE+ 程序中

包含这些错误，它们在 FoxPro 和 FoxBASE+下的作用是不同的，但 FoxPro 将给出错误信息。

## 2. 许多新的函数和关键字

FoxPro 增加了许多新的函数，当然也增加了许多新的关键字，如果 FoxPro 的这些函数或关键字在 FoxBASE+应用程序中被作用用户自定义函数(UDF)名，则 FoxPro 会将其解释为自己的定义而不执行这个自定义函数。例如，Evaluate() 函数。

## 3. set color to 命令

在 FoxBASE+中，执行不带任何参数的 set color to 命令将使屏幕恢复到缺省状态（即黑底白字），而在 FoxPro 中，执行不带参数的 set color to 命令将不会引起屏幕上任何颜色上的变化。

## 4. set default 命令

在 FoxBASE+中，当你用 set default 命令指定缺省驱动器和目录时，实际上起作用的只是指定了缺省驱动器，如下命令在 FoxBASE+中给出的只是缺省驱动器为 C，并没有实现用户的要求。

set default to c:\foxBASE\Programs，而同样的命令若用在 FoxPro 中，就可精确地给出用户要求的缺省驱动器和缺省目录路径。

## 5. 空字符的使用

在 FoxBASE+中，用户想将一个字符串初始化为空字符串（不包含任何字符），则下面的两条语句将起到相同的作用。

```
mystring = chr(0)
mystring = ""
```

但这一点和 FoxPro 是不同的，在 FoxPro 中，前一个语句将产生一个包含一个字符（即空字符）的字符串，而后一个语句才能真正将一个字符串初始化为空字符串。

## 6. .VUE 文件

在 FoxBASE+中，.VUE 文件包含有如下信息：数据库、索引、别名、格式文件、关联、字段名、过滤器，及各种 Set 命令的 on/OFF 状态，在 FoxPro 中，.VUE 文件也包含上述信息，但还要加上缺省驱动器、缺省目录路径、过程文件、帮助文件、源文件、索引文件、时钟位置、货币的国别及各种 Set 命令的 on/OFF 状态。

# 1.5 FoxPro 与 FoxBASE+的兼容性

FoxPro 可以运行 FoxBASE+的大多数源程序，而不需作任何改动，不过当你需要在 FoxPro 下运行 FoxBASE+的程序时，最好使用下列命令：

```
Set compatible FoxPlus
```

以保证兼容性。

同时，FOXPRO 还提供一个特殊的系统配置文件 Config.Fox，它将保证 FoxPro 与 FoxBASE+几乎百分之百地兼容，这个系统配置文件放在 FoxPro 2.5\goodies\misc，为了使用这一文件，你可以简单地将这文件复制为 config.FD，并放在当前目录下，也可以将 config.Fox 的内容放在 config.FP 文件中，为了实现兼容，还应在 config.FP 文件中

包含如下语句:

```
Brstatus=on
mackey=<expc>
notify=OFF
Scoreboard=on
Status=on
```

## 1.6 将 FoxBASE+2.10 文件转换为 FoxPro 2.5 文件

FoxPro 认可 FoxBASE+的如下类型的文件:

- 源程序 (.prg文件) —— 这类程序将自动地编译为与FoxPro兼容的目标程序 (.FXP 文件)。
- 数据库文件 (.DBF文件)。
- 备注型文件 (.DBT文件) —— FoxPro识别.DBT这类备注型文件时, 只将它们转换为与 FoxPro 兼容的备注型文件, 并加上扩展名.FPT。
- 索引文件 (.IDX文件)。
- 内存变量文件 (.Mem文件)。
- 屏幕格式文件 (.FMT文件) —— 当你在应用程序中执行set format to 命令时, FoxPro 自动将.FMT 文件编译为.PRX 文件。
- 报表格式文件 (.FRM文件) —— FoxPro的报表生成器,以与FoxBASE+不同的格式生成报表格式文件, 不过, FoxPro 可以用 FoxBASE+的.Frm 格式文件来打印报表, FoxPro 的报表生成器也可以读入.Frm 文件进行修改, 但格式与原来的 .Frm 文件不同了。
- 标签定义文件 (.LBL文件)。

上述类型的文件在用 FoxPro 进行执行时, 不需用户作任何修改, 所需的转换工作由 FoxPro 自动完成。

但下列两类文件需由用户作适当处理, 即扩展名为.NDX的索引文件和扩展名为 .DBT 的备注型文件。

### 1. 扩展名为.NDX 的索引文件

FoxPro 对数据库建立的索引文件格式与 dBASE,FoxBASE 索引文件格式是不同的, 而且索引文件变得更小了, 这主要是因为 FoxPro 采用了新的索引技术。

如果你正在运行含有 dBASE 的 NDX 索引文件的应用程序, 你也不必担心必须将你的索引文件转换为 FoxPro 格式——FoxPro 的设计者已为你考虑到了, FoxPro 会立即自动重新索引数据库, 并生成 FoxPro 的索引文件, 并在屏幕上显示如下信息:

```
dBASE III index--rebuilding
```

即 dBASE III 索引文件已被重建。

缺省时, 这类新的索引文件将被冠以扩展名.IDX,但也许有的用户想使这些索引文件具有扩展名.NDX, 如果这样的话, 只要在 config.Fp 文件中加入以下语句:

```
INDEX=NDX
```

这将使 FoxPro 给所有由它建立的索引文件加上扩展名.NDX, 当然, 这样的话, 会在重新索引过程中, 破坏掉原有的 dBASE 索引文件, 因为这时它们有相同的扩展名.NDX.

## 2. 扩展名为 DBT 的备注型文件

FoxPro 的备注型文件 (.FPT 文件) 与 FoxBASE+, dBASE III Plus, 和 dBASE IV 所用的备注型文件 (.DBT 文件) 稍有不同, .DBT 文件只能存放 ASCII 码文本数据, 而 .FPT 文件中可以存放任意类型的数据, 不过, FoxPro 可以读写 .DBT 文件而不需要将其转换为新的格式.

在以下两种情况下, FoxPro 会根据已存在的 .DBT 文件来建立一个新的 .FPT 文件.

- (1) 根据一个已有的包含备注型字段且有备注型文件的数据库来建立一个新的数据库结构时, copy to 命令就是一例.
- (2) 当你修改一个已有备注型文件的数据库结构时, 当修改后的库结构存盘时, .DBT 文件被自动转换为新的 .FPT 文件.

若想为 FoxPro 的一个含有备注型字段的数据库作一个能被 FoxBASE+ 识别的备份, 需执行下列命令:

```
copy to <Filename> TYPE FoxPlus
```

而且此时, 你的这个 FoxPro 数据库中的备注型字段只能包含能为 FoxBASE+ 处理的那些字符, 不应包含像 Ctrl+chr((26)) 和 null (chr(o)) 这类字符.

## 1.7 Rushmore 技术

Rushmore 技术是 FoxPro 2.5 采用的一种数据快速存取技术, 它实现了对一系列记录的有效快速存取, 其速度相当于对已索引的单个记录的存取速度.

用户通过使用 Rushmore 技术, 可以使一些复杂的数据库操作的速度比原来提高成百上千倍, FoxPro 2.5 使得用户只需使用个人计算机便可处理真正大的包含上百万个记录的大型数据库, 其速度可与具有主机的数据库系统媲美.

Rushmore 技术采用标准的 FoxPro B-树索引, 而不需要任何新型的文件或索引. Rushmore 技术可用于任何的 FoxPro 索引: 如 FoxPro 1.xx 采用的标准索引(.NDX), 压缩索引(.IDX)或复合索引(.CDX).

Rushmore 技术特别不依赖于新的压缩索引格式, 无论是 .CDX 格式还是 .IDX 格式的压缩索引, 都采用一种压缩技术, 它产生的索引文件只有旧格式索引文件长度的六分之一. 压缩索引的执行速度是相当快的, 因为它在物理上是非常小的, 这意味着在进行索引时只需做少量的磁盘存取, 而且大量的索引可驻留于 FoxPro 的存储缓冲区.

尽管 Rushmore 技术得益于压缩索引的小的文件长度, 但它在处理旧格式索引时, 效果也非常显著.

当需要处理的数据库非常大时, 小机器有可能为 Rushmore 技术提供不了足够的内存, 这种情况下, 会出现如下的警告信息:

```
Not enough memory for optimization
```

(没有足够的内存来实现优化), 程序会继续运行, 但不进行优化. 尽管不会出现数据丢失, 且程序能正确运行, 但这种情况下, Rushmore 技术在查询时已无法发挥作用.

因此, 如果用户在处理大型数据库(总计超过五十万条记录)时, 建议用户使用 FoxPro 2.5 的增强型版本。

最简单的情况是: Rushmore 可加快对单个数据库进行操作的 FOR 子句的执行速度。采用 FOR 子句可根据已索引的数据库来限定符合条件的一组记录。当 SET FILTER 置为 ON, 且对已索引过的数据库建立过滤条件后, Rushmore 技术还可加快下表中介绍的包含 FOR 子句的潜在可优化命令的执行速度。

为了对多重数据库更好地发挥 Rushmore 技术的优势, 用户应当使用 SQL SELECT 命令。FoxPro 的 SQL 工具在实现多重数据库的优化查询时, 将 Rushmore 技术当作一个基本工具, 从而对已存在的索引实现 Rushmore 技术, 甚至建立新的特定的索引, 以便加快查询速度。

### 1. 多重数据库的 Rushmore 技术

如果处理多于一个的数据库时, 为了充分发挥 Rushmore 技术的优化作用, 用户应用使用 SQL SELECT 命令, SQL 在优化它的查询时, 将 Rushmore 当作一个基本工具。

用户一旦选择了 SELECT, 就没有必要牢记在通常情况下的那些规则了, SQL 会自行决定在什么时候需要对一个查询进行优化, 并自动完成优化查询, 用户甚至不需要打开数据库或相应的索引文件, 如果 SQL 觉得需要索引文件, 它会为自己建立临时的索引文件。

### 2. 单个数据库的 Rushmore 技术

若处理的是单个数据库, 用户可以在出现 FOR 子句的任何地方来充分发挥 Rushmore 技术的优势。Rushmore 技术的研制者使得它的速度与欲处理的记录数据成正比, 即记录数越多, 使用 Rushmore 技术的优势越明显。

| 潜在的包含 FOR 子句的可优化命令 |         |         |       |
|--------------------|---------|---------|-------|
| AVERAGE            | COUNT   | LIST    | SORT  |
| BROWSE             | DELETE  | LOCATE  | SUM   |
| CALCULATE          | DISPLAY | RECALL  | TOTAL |
| CHANGE             | EDIT    | REPLACE |       |
| COPY TO            | EXPORT  | REPORT  |       |
| COPY TO ARRAY      | LABEL   | SCAN    |       |

为了充分发挥 Rushmore 技术的优势, 除了包含可优化的 FOR 子句, 上表中的命令还必须含有一个带有 ALL 或 NEXT 的范围子句, 若用户允许系统采用默认的范围 ALL, Rushmore 也可很好地发挥作用。

在进行优化时, Rushmore 可以使用任何已打开的索引文件, 除了已过滤的和唯一的索引。

在进行优化时, 不必对数据库进行排序, 如果用户建立索引文件, 那么这种排序是自动进行的。如果用户既想对一个大型数据库最大限度地发挥 Rushmore 技术的优势, 又想要求其中记录按特殊要求排序, 那么最好使用命令 SET ORDER TO 关掉索引控制, 然后执行排序命令 SORT。

### 3. 基本可优化表达式

Rushmore 技术产生作用与否, 取决于在一个 FOR 子句中是否出现一个基本可优化表达式。一个基本可优化表达式可以形成一个完整的表达式, 或只作为表达式的一部分出现。组合基本可优化表达式的规则见第四部分。

一个基本可优化表达式取如下形式之一:

<index expression> <relational operator> <constant expression>

(<索引关键字表达式> <关系运算符> <常量表达式>)

或

<constant expression> <relational operator> <index expression>

(<常量表达式> <关系运算符> <索引关键字表达式>).

在一个基本可优化表达式中:

- <索引关键字表达式> 必须与建立索引文件时的索引关键字表达式精确匹配, 而且索引关键字表达式中不能包含别名。
- <关系运算符> 必须是下列运算符之一: <, >, =, <=, >=, <>, #, !=。
- <常量表达式> 可以是任何表达式, 可包括内存变量和其它无关数据库的字段变量。

例如, 若用户已根据如下表达式建立了索引文件:

```
FIRSTNAME  
CUSTNO  
UPPER(LASTNAME)  
HIREDATE  
ADDR
```

则如下表达式便是基本可优化表达式:

```
FIRSTNAME = 'Fred'  
CUSTNO >= 1000  
UPPER(LASNAME) = 'SMITH'  
HIREDATE < {12 / 30 / 90}
```

如果用户执行命令

STORE 'WASHINGTON AVENUE' TO X, 则下面的表达式也是基本可优化表达式:

式:

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

### 4. 组合基本可优化表达式

Rushmore 数据处理优化技术取决于 FOR 子句表达式, 通过一个简单的或复杂的 FOR 子句表达式, 数据处理速度可以得到加强。如果 FOR 子句表达式由基本表达式组合而成, 本节中将基本表达式组合成一个 FOR 子句表达式的原则。

基本表达式应当是可优化的, 基本表达式可以通过逻辑运算符 AND, OR 或 NOT 组合成一个复杂的 FOR 子句表达式。

通过可优化的基本表达式组合成的表达式是完全可优化的。若其中有一个或多个基本表达式不可优化，则组合成的表达式也许是部分可优化的或不可优化的。

如下规则则可决定由可优化或不可优化的基本表达式组成的表达式是完全可优化的，部分可优化的或不可优化的：

| 基本表达式的组合规则       |          |                  |                       |
|------------------|----------|------------------|-----------------------|
| Basic Expression | Operator | Basic Expression | Query Result          |
| Optimizable      | AND      | Optimizable      | Fully Optimizable     |
| Optimizable      | OR       | Optimizable      | Fully Optimizable     |
| Optimizable      | AND      | Not Optimizable  | Partially Optimizable |
| Optimizable      | OR       | Not Optimizable  | Not Optimizable       |
| Not Optimizable  | AND      | Not Optimizable  | Not Optimizable       |
| Not Optimizable  | OR       | Not Optimizable  | Not Optimizable       |
| —                | NOT      | Optimizable      | Fully Optimizable     |
| —                | NOT      | Not Optimizable  | Not Optimizable       |

下表给出组合基本表达式的一些例子：

| 组合基本表达式的例子                                       |                                                            |
|--------------------------------------------------|------------------------------------------------------------|
| Example                                          | Expression Types, Operator and Result                      |
| FIRSTNAME = 'FRED' AND HIREDATE < {12 / 30 / 89} | Optimizable AND Optimizable<br>= Fully Optimizable         |
| FIRSTNAME = 'FRED' OR HIREDATE < {12 / 30 / 89}  | Optimizable OR Optimizable<br>= Fully Optimizable          |
| FIRSTNAME = 'FRED' AND 'S' \$ LASTNAME           | Optimizable AND Not Optimizable<br>= Partially Optimizable |
| FIRSTNAME = 'FRED' OR 'S' \$ LASTNAME            | Optimizable OR Not Optimizable<br>= Not Optimizable        |
| 'FRED' \$ FIRSTNAME AND 'S' \$ LASTNAME          | Not Optimizable AND Not Optimizable<br>= Not Optimizable   |
| 'FRED' \$ FIRSTNAME OR 'S' \$ LASTNAME           | Not Optimizable OR Not Optimizable<br>= Not Optimizable    |
| NOT FIRSTNAME = 'FRED'                           | The NOT Operator with Optimizable<br>= Fully Optimizable   |
| NOT 'FRED' \$ FIRSTNAME                          | The NOT Operator with Not Optimizable<br>= Not Optimizable |

用户可以通过括号将基本表达式进行分组组合，如上规则也适用于将表达式通过分组方式来组合。

## 5. 组合成复杂表达式

用户可将复杂表达式组合成更复杂的表达式。这些更复杂的表达式可以是完全可优化

的，部分可优化的或不可优化的，就像上表所示，这些更复杂的表达式当然也可以再进行组合，组合的结果可见下表，下表的规则也适用于将复杂表达式进行分组组合：

| 复杂表达式的组合规则            |          |                       |                       |
|-----------------------|----------|-----------------------|-----------------------|
| Expression            | Operator | Expression            | Result                |
| Fully Optimizable     | OR       | Not Optimizable       | Not Optimizable       |
| —                     | NOT      | Fully Optimizable     | Fully Optimizable     |
| Partially Optimizable | AND      | Partially Optimizable | Partially Optimizable |
| Partially Optimizable | OR       | Partially Optimizable | Partially Optimizable |
| Partially Optimizable | AND      | Not Optimizable       | Partially Optimizable |
| Partially Optimizable | OR       | Not Optimizable       | Not Optimizable       |
| —                     | NOT      | Partially Optimizable | Partially Optimizable |
| Not Optimizable       | AND      | Not Optimizable       | Not Optimizable       |
| Not Optimizable       | OR       | Not Optimizable       | Not Optimizable       |
| —                     | NOT      | Not Optimizable       | Not Optimizable       |

下表给出了复杂表达式进行组合的例子：

| 组合复杂表达式的例子                                                                                              |                                                                           |
|---------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------|
| Example                                                                                                 | Expression Types, Operator and Result                                     |
| (FIRSTNAME = 'FRED' AND HIREDATE < {12 / 30 / 89})<br>OR (LASTNAME = ' ' AND HIREDATE > {12 / 30 / 88}) | Fully Optimizable OR Fully Optimizable<br>= Fully Optimizable             |
| (FIRSTNAME = 'FRED' AND HIREDATE < {12 / 30 / 89})<br>AND 'S' \$ LASTNAME                               | Fully Optimizable AND Not Optimizable<br>= Partially Optimizable          |
| (FIRSTNAME = 'FRED' AND 'S' \$<br>LASTNAME)<br>OR (FIRSTNAME = 'DAVE' AND 'T' \$<br>LASTNAME)           | Partially Optimizable OR Partially Optimizable<br>= Partially Optimizable |
| ('FIED' \$ FIRSTNAME OR 'S' \$<br>LASTNAME)<br>OR ('MAIN' \$ STREET OR 'AVE' \$<br>STREET)              | Not Optimizable OR Not Optimizable<br>= Not Optimizable                   |

## 6. Rushmore 技术不可使用的情况

在少数情况下，Rushmore 技术可能达不到加速数据处理的目的。

一旦它不能优化在一个潜在可优化命令中的 FOR 子句表达式，Rushmore 就失效了。一旦一个命令中包含 WHILE 子句，则 Rushmore 技术对该命令失效。

在 FoxPro 的标准版本下，一旦所有打开的数据库中记录总数超过五十万个，FoxPro 也许要关闭 Rushmore 技术，然而增强型版本的 FoxPro 可以加快命令的执行速度，无论记录超过五十万个，还是小于五十万个。

在内存较小的情况下，Rushmore 技术不能优化数据处理命令。

## 1.8 运行 FoxPro 2.5 所需要的软硬件环境

单用户的 FoxPro 2.5 for Dos 有两种版本可供用户选择，标准型版本，增强型版本，增强型版本是一个真正的 32 位产品，它可以充分使用 80386 和 80486 微处理器，并可使用所有可用的扩展内存，这意味着文件的运行速度会更快一些。

标准版本的 FoxPro 只要求 8088 / 8086 或 80286 的处理器。

FoxPro 2.5 for MS-DOS 的配置

|           | 标准版本            | 增强版本          |
|-----------|-----------------|---------------|
| 处理器       | 8088 或更高        | 386SX 或更高     |
| RAM       | 640K (建议使用 2MB) | 3MB           |
| 鼠标        | 建议使用            | 建议使用          |
| MS-DOS 版本 | MS-DOS3.1 或更高   | MS-DOS3.1 或更高 |

当 FoxPro 被正确地使用在合适的硬件平台上时，它可提供一种异常快的速度。不过，它也是一种复杂的软件产品，因为它要与扩展内存，EMS 内存，XMS 内存，高位存储单元，DOS 文件柄和缓冲器，局域网操作系统的文件柄进行大量的相互作用。为了高效地运行 FoxPro 2.5，需对计算机进行合理配置，不仅要建立最有效的 FoxPro 的系统配置文件 CONFIG.FP，而且要调整 DOS，内存管理器，磁盘超高速缓冲器，Microsoft Windows 和局域网操作系统的配置和选择。

配置调整得越精细，FoxPro 运行得越好越快。配置 FoxPro 时的重要工作之一是，为 FoxPro 和用户程序获得尽可能大的常规内存空间（即基本内存空间）。常规 DOS 内存空间增加 5K 到 10K，这显得并不很多，但即使增加这点小的常规 DOS 内存空间，也将使 FoxPro 的性能得到改进。

根据我们的经验，用户所碰到的大多数问题直接来自系统配置方面，而不是 FoxPro 本身。

我们将介绍如何有效地配置计算机系统，局域网系统，以及进行内存管理，从而有效地运行 FoxPro 2.5 / DOS。

### 1. 系统配置文件 CONFIG.SYS

无论你用的是 DOS 的哪种版本，也不管你的计算机是单机运行还是作为局域网(Local Area Network，缩写为 LAN) 上的一个工作站，只要机器启动，DOS 总要在计算机的缺省驱动器的根目录中寻找 CONFIG.SYS 文件。CONFIG.SYS 文件中有两项设置的值对 FoxPro 的正常运行至关重要，它们是 FILES 参数和 BUFFERS 参数。

没有 CONFIG.SYS 文件的计算机也可以运行。但像 FoxPro 这样的高级软件，若没

有一个适当的 CONFIG.SYS 文件,是很难充分发挥它的作用的。CONFIG.SYS 文件的内容可以有多种形式,也可以只包含两项参数的设置:

```
BUFFERS=nn
```

```
FILES=nn
```

这里的 nn 代表一个整数值,或者,nn 可初始化一系列的设备驱动器,内存管理,磁盘高速缓冲器,以及其它选项,而且也可以很复杂。

#### (1) 打开文件个数参数 FILES

CONFIG.SYS 中的 FILES 参数定义了同一时刻可以打开的文件柄的个数。DOS 要求 3 到 5 个文件柄,这取决于 DOS 的版本。FoxPro 2.5/DOS 和 FoxPro 2.0 要求同样的文件柄个数,它们比 FoxBASE+和 FoxPro 1.x 要求的文件柄个数要多。对 FoxBASE+或 FoxPro 1.0 来说已足够的 FILES 参数,若要运行 FoxPro 2.5/DOS 和用户程序,则应当增大 FILES 参数。随着可用文件柄个数的增大,常规 DOS 内存空间减少了。

如果一个 FOXUSER 资源文件和一个帮助文件在使用着,启动 FoxPro 2.5/DOS 的标准版将立即使用如下 10 个文件柄:

```
FOXPRO.EXE
```

```
FOXPRO.OVL
```

```
FOXPRO.DBF
```

```
FOXPRO.FPT
```

```
FOXPRO.DBF
```

```
FOXHELP.FPT
```

一个 TMP(临时)文件,它是由索引 FOXHELP 文件时产生的。三个由编程作业、排序作业和编辑作业产生的临时文件。

FoxPro 2.5/DOS 的增强版要求九个文件柄,因为它不使用覆盖文件,而在它的内部完成覆盖交换。

每打开一个数据库,就要增加一个文件柄。如果打开的数据库包含了一个或多个备注型和/或通用(general)字段,则 FPT 文件还需要一个附加的文件柄。如果正在使用 IDX 类型的索引文件,无论是标准的还是紧缩的,每个 IDX 文件都要有一个文件柄。包含结构式 CDX 索引的复合索引 CDX 文件只要求一个文件柄,而不管其中有多少个 TAG。SQL 和 SQL CURSOR 也需要额外的文件柄,但文件柄的精确个数取决于 SQL 的类型和它的复杂程度。将 FILES 参数设为 40 并不能保证你运行复杂的应用程序而不碰到“Too many files Open”(打开文件个数太多)的错误信息。

如果 CONFIG.SYS 文件中没有包含 FILES 参数,大多数 DOS 版本默认 45 个可用文件柄。不同的 DOS 版本支持 FILES 参数的不同取值范围。一般来讲, DOS3.3 及其以上版本支持的 FILES 参数范围为 1 到 255,但这个范围也可能因具体机器的不同而不同。DOS 3.3 以下的版本一般只支持最多 99 个文件柄。如果你指定一个大于 99 的参数值,而你的 DOS 版本不支持多于 99 个的文件柄, DOS 通常取其缺省值 45。当你在网络环境下工作,并为服务器考虑文件柄配置时,文件柄的配置变成了一种平衡动作。

没有哪一个 FILES 参数能适应于所有的应用程序,因为每个应用程序都是独特的。

一般来说,我们发现,将 FILES 参数取为 75 到 100 之间的值,基本上能适合大多数应用程序的需要。

## (2) 缓冲器参数 BUFFERS

一个缓冲器,或称为磁盘缓冲器,是一块内存单元(一般为 512 到 532 字节,具体值要取决于 DOS 版本)。DOS 用它来存储最近从磁盘上读出的或写入磁盘的数据。不过,有些 DOS 缓冲器并不存储最新已写入磁盘的信息。将数据读出或写入内存的速度要比不断地移动硬盘的读/写磁头以便读出硬盘上的信息或将信息写入硬盘的速度快得多。只当缓冲器已满, DOS 才会去读写硬盘。DOS 缓冲器在某种程度上类似于磁盘高速缓冲器。

如果 CONFIG.SYS 文件中没有包含缓冲器参数 BUFFERS, DOS 的大多数版本将使用该参数的缺省值。缺省值取决于所用的机器,计算机中内存的容量以及 DOS 的版本等。IBM-XT 的 BUFFERS 参数的缺省值是 2, IBM-AT 的 BUFFERS 参数的缺省值是 3。如果计算机有 512K 以上的内存,新的 DOS 版本将 BUFFERS 参数的缺省值取为 10 或 15。最终取哪个值,还要看你所用的 DOS 版本。

CONFIG.SYS 文件中的缓冲器参数设置得太大或太小都将减弱计算机的性能。如果该参数设得过大, DOS 将把大量的数据读入缓冲器,因此减弱了性能。因为 FoxPro 使用自己的缓冲器或高速缓冲器,因而在 CONFIG.SYS 中指定一个大的缓冲器参数只能起到反作用,因为在 FoxPro 高速缓冲器中的信息将被移入 DOS 的缓冲器,然后再写入硬盘。更进一步,大的缓冲器参数设置将减少 DOS 可为 FoxPro 提供的内存空间,最终降低了机器的性能。

讨论缓冲器参数的最佳设置是一件无止境的事, Microsoft 公司推荐的 BUFFERS 参数的范围是 20 到 40。我们的经验也表明,具有 1M 以上内存的计算机的最佳 BUFFERS 参数是 20 或 25。不过这仅是一个一般的看法,也许这种最佳值要随着机器的不同而改变。例如,若正在使用一个高速缓冲磁盘控制器,用户可能会发现,如将 BUFFERS 参数降到 8 或 10,机器的速度会明显改善。

DOS 4.x 为 BUFFERS 参数提供了一个新的可选项,它可把缓冲器装入高位内存:

**BUFFERS=20 /x**

尽管选项 /x 可增加可用的常规 DOS 内存空间,但也已证明,该选项将导致与 FoxPro, WordPerfect 等直接存取高位内存的软件的内存冲突。如果你使用的是 DOS 4.x, 作者建议不要采用 /x 缓冲器参数可选项。

## 2. 局域网与文件柄

与单机或单用户环境相比较,局域网环境下的文件柄参数配置是一个更复杂的问题,但对你的多用户应用程序的作用来说,它又是一个关键问题。各种网络操作系统所提供的文件柄参数配置各不相同,因而使问题更复杂化了。

### (1) Novell 286 和 Novell 386 局域网

除了 DOS 的 CONFIG.SYS 文件, Novell 286 或 Novell 386 局域网上的工作站还要求一个 Novell 配置文件。在早期的 Novell 286 和 Novell 386 版本中,这个文件被叫做 NET.CFG。后来的 Novell 版本用的是 SHELL.CFG 文件, Novell 网络操作系统的最新版本中又采用了名为 NET.CFG 的配置文件。不论用的是 Novell 286 或 Novell 386 网络操作系统的哪个版本,一旦启动 Novell LAN, 就要寻找 Novell 配置文件。如果

NET.CFG 和 SHELL.CFG 文件同时存在, 则 Novell 使用 NET.CFG 文件。Novell 配置文件应当和 Novell 驱动器放入同一个子目录。

SHELL.CFG 或 NET.CFG 文件中应至少包含有关 FILE HANDLES 参数的申明。FILE HANDLES 参数在概念上类似于 CONFIG.SYS 文件中的 FILES 参数; 不过 Novell 的 FILE HANDLE 参数控制的是定位于服务器上的可用文件柄的个数, 而 CONFIG.SYS 中的 FILES 参数控制的是定位于硬盘和驻留于工作站内存中的可用文件柄的个数。DOS 文件柄与 Novell 文件柄的个数之和不能超出 250。例如, 若 FILES = 130, 而且 FILE HANDLE = 130, 则当用户希望进网时, Novell 网络操作系统将会结出一个错误信息。在你减少上述参数的值之前, 你是不可能进入局域网的。如果你将 CONFIG.FP 中的 FILES 参数降得太小, 你最终会碰到来自 DOS 或 FoxPro 的错误信息 "Too many files"。

如果正在运行 FoxPro 2.5 / DOS 的增强版, 你不可能使用由 FoxPro 提供的 225 个工作区中每一个。你还必须作一个平衡工作, 以便决定哪种配置更符合你的情况。我们一般总将 DOS 的 FILES 参数取为 100, 将 Novell 的 FILE HANDLE 参数也取为 100。

如果没有 NET.CFG 或 SHELL.CFG 文件, 或者关于参数 FILE HANDLE 的申明没有包含在上述文件中, Novell 将默认为 40 个可用的文件柄。这个值对运行 FoxPro / DOS 来说是不够的。

你也许还需要向上调整 Novell 服务器上的文件柄数目。Novell 服务器默认 250 个可用文件柄。服务器上最多可允许有 1000 个文件柄。在 Novell 286 网络系统中, 你可以通过 Novell 的 Maintenance 菜单来增加服务器上的文件柄数目。选择系统配置菜单选项。Novell 386 网络操作系统使用一个位于服务器系统目录中的 STARTUP.NCF 文件。为了增加服务器上可用文件柄的数目, 必须修改该文件。你可以通过安装选项菜单或使用一个文本编辑器将下列语句加入 STARTUP.NCF 文件来完成这种修改:

```
FILE HANDLE = nnn
```

人们经常问的一个问题是, Novell 286 网络操作系统和 Novell 386 网络操作系统是否要求 NETBIOS 或 DOS SHARE。尽管 Novell 不提供 NETBIOS 文件, 在 Novell 网上运行 FoxPro 2.5 / DOS 或 FoxPro / Windows 时, 并不需要 NETBIOS 文件, 也不需要 DOS SHARE。

## (2) Novell Net Ware Lite 和 Novell ELS

Novell Net Ware Lite 将 DOS SHARE 用作一种管理工具, 以便实现文件共享。(有关使用 SHARE 的更多信息, 请参考(4))。局域网工作站和服务器上的 CONFIG.SYS 文件中的 FILES 参数控制着可用文件柄的数目。服务器上的 FILES 参数必须提供足够的文件柄, 以便所有的服务器文件可被所有的工作站打开。例如, 五个工作站同时使用同一个数据库文件, 并且没有使用索引文件 (包括 CDX 结构索引文件), 则它们使用的是 5 个文件柄。若该数据库文件含有一个或多个备注型或通用型字段, 则将使用 10 个文件柄。

Novell ELS 既不使用 SHELL.CFG 文件, 也不使用 NET.CFG 文件。相反, 用户必须通过 Novell 的某种途径来改变可用文件柄的数目。

## (3) 基于 DOS 的局域网

基于 DOS 的局域网, 像 Novell NetWare Lite, 要求装入 DOS SHARE。这种以及大

量其它基于 DOS 的局域网通过 CONFIG.SYS 中的 FILES 参数来决定可用文件柄的数目。这类局域网上的工作站只需将 FILES 设为 99 便可很好地工作。

除了工作站上的 FILES 参数设置,服务器上的 FILES 参数必须为所有打开的文件提供足够的文件柄。

#### (4) DOS SHARE

DOS SHARE 为文件共享提供了一个有力工具,它还提供高级磁盘支持,它还为 DOS 4.x 中大于 32M 的 DOS 分区提供支持。在 DOS 5.0 和 DOS 6.0 下的大容量硬盘分区时,DOS SHARE 是不需要的。

如果用的是一个完整的局域网,如 Novell 286,Novell 386,那么不需要装入 DOS SHAR。

当在单机上运行 FoxPro 2.5 / DOS 或 FoxPro / Windows 时,若不希望以独占方式打开文件,则应当装入 DOS SHARE。FoxPro 将去检查 DOS SHARE 或 LAN OS(局域网操作系统)是否被激活,如果没有激活的话,则只以独占方式打开文件;而且这时,FoxPro 的 SET EXCLUSIVE OFF 命令也将不起作用。如果你想在单机上通过使用 Microsoft Windows 来完成多任务,那也应当装入 DOS SHARE。

DOS 手册和许多有关 DOS 的书都建议通过 CONFIG.SYS 来装入 DOS SHARE。我们反对这种做法,因为它去掉了本可由 DOS SHARE 提供的高级磁盘支持。例如,如果从 CONFIG.SYS 中装入 SHARE,将会因写软盘而引出 DOS 错误,拿出这张磁盘,放入另外一张软盘,再试图向新的磁盘写入信息,如果 SHARE 是通过 AUTOEXEC.BAT 装入的,这种错误就不会发生了。SHARE 可为硬盘或软盘提供高级磁盘支持。

当 SHARE 是通过 AUTOEXEC.BAT 或在 DOS 提示符下装入,而且被装入高位内存空间,那么你不会碰到任何问题或冲突,因为若将 SHARE 装入高位内存,便会给 FoxPro 留下更多的常规内存,FoxPro 会执行得更快。然而,若已通过 CONFIG.SYS 将 SHARE 装入高位内存,则 FoxPro 并不能很成功地运行。在 DOS 5.0 下,将 SHARE 装入高位内存会省出将近 10K 的常规内存空间,必然会导致 FoxPro 的更快运行。

SHARE 支持两个参数。/F:<以字节为单位的数>代表用于共享的字节数。如果不指定 /F:参数,SHARE 默认为 2048 字节。每个打开的文件使用文件名的长度再加上 11 个字节。参数 /L:<数值>用于指定可用的锁定数。若不指定 /L:参数,SHARE 默认为 20,在某些情况下,这并不能为 FoxPro 提供足够的锁定数,特别当用户程序中使用了多用户锁定命令 MULTLOCK。

将这两个参数指定为小于默认值的数会节约内存,但用户也许会发现,这样会影响 FoxPro 的性能。如果看到了下列错误信息:

System Resources Exhausted

那应当加大 SHARE 中的 /F:或 /L:参数的值。

#### (5) DOS 下的文件快速打开工具 FASTOPEN

有些 DOS 经销商已开始将 FASTOPEN 包含于 DOS3.1。例外的是 DOS 4.0,所有它的经销商都包含了 FASTOPEN。FASTOPEN 通过 CONFIG.SYS 文件被装入。FASTOPEN 在内存中建立了有关文件和子目录位置的一张表。一旦它们被调用,