第一章 概 述

美国 Fox Software 公司推出的 FoxBASE+在国内有非常多的用户。该公司为了与美国 Ashton-Tate 公司的 dBASE(dBASE II,dBASE III,dBASE III PLUS 和 dBASE IV)抗衡,不断推陈出新,先后推出 FoxBASE、FoxBASE一、FoxPro 2.0,FoxPro 2.5 等各种版本。自 FoxPro 2.0 以后加入独有的 Rushmore 优化技术,使速度显著提高,大大高于其它产品。

FoxPro 2.5 是最新推出的,保持了与 dBASE 相兼容,使用 dBASE 和 FoxBASE+编写的程序无需作任何改动,可直接在 FoxPro 2.5 中运行。除此之外,它扩大已有的命令,还增加许多新的命令和函数,减少对用户的限制(如,工作区数量增至 255 个,记录长度增加到 65500 个字节)。采用 90 年代的先进技术,支持鼠标器,使整个系统更易使用和操作。尤其是可将程序伪编译为非常接近机器语言的中间代码程序,增快其执行的速度。如果程序经 Fox Distribution Kit 编译,可生成以.EXE 为扩展名的执行文件,脱离 FoxPro,而直接在 DOS 或 Windows 下运行。

FoxPro 2.5 按照支持的环境可分为:DOS 版 FoxPro 2.5 和 Windows 版 FoxPro 2.5。DOS 版 FoxPro 2.5 与 Windows 版 FoxPro 2.5 命令和函数的格式及功能基本相同,只是极少数有些差别,本书将详细介绍 FoxPro 2.5,并且指出这些差异。

1.1 启动 FoxPro

1.1.1 DOS 版 FoxPro 2.5 的启动

启动 DOS 版 FoxPro 2.5 比较容易,在操作系统提示符下输入:

FOXPRO

屏幕显示如图 1.1 所示的画面,表示已完成 FoxPro 的启动,等待用户的操作。

在图 1.1 所示的屏幕上,最顶上一行为系统菜单(System Menu),用户使用这个菜单可直接进行数据的操作。屏幕中间显示 FoxPro 的版本号、商标和系列号,位于屏幕右下方的窗口称为命令窗口(Command),启动后光标停在此处。

1.1.2 Windows 版 FoxPro 2.5 的启动

Windows 版 FoxPro 2.5 可通过以下两种方法在 Windows 中启动:

(1) 双击 FoxPro 图标(FoxPro for Windows)或移动光标到 FoxPro 图标并按回车键;

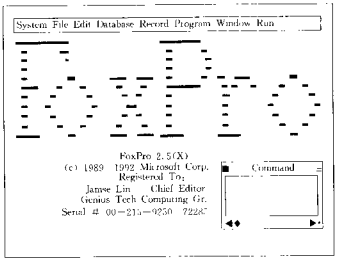

图 1.1 启动 FoxPro 2.5 的屏幕

(2) 在 Windows 的菜单 File 中选择 Run 选择项，在出现 Run 对话框时，键入带有驱动器符和目录名的 FOXPROW.EXE。假定 FoxPro 2.5 安装在 D 盘 FOXPROW 目录中，输入：

D:\FOXPROW\FOXPROW

稍候，屏幕显示如图 1.1 所示的画面，表示已启动 FoxPro。

1.2 命令窗口

在图 1.1 右下角显示一个命令窗口，用户在命令窗口中输入 FoxPro 命令，按回车键后，表示命令输入完成，FoxPro 执行该命令。当命令执行完成后，FoxPro 重新等待用户输入下一条命令。

在命令窗口中，用户可以利用←，→和 Del 键修改输入的命令行，也可以使用→，←，↑，↓，PgUp，PgDn，Home，End 等键找出先前执行过的命令，经过编辑后，按下回车键来重新执行该命令，可用 Ins 键改变当前覆盖/插入的状态。

用户也可以使用鼠标器点击窗口底边和右边的箭头(▲，▼，◀，▶)进行上、下、左、右的滚动显示。

命令窗口通常约占 5 行 20 个字符宽，用户可以利用键盘、鼠标移动和改变命令窗口的大小。

1. *移动窗口*

使用鼠标器移动窗口，将鼠标器指针指向窗口的顶行(含有 Command)，然后按住鼠标器按钮，拖动窗口到希望的位置，释放鼠标器按钮。

使用键盘移动窗口，按下 Ctrl-F7 键使命令窗口的边界变成闪烁，按→，←，↑，↓，

PgUp，PgDn，Home 和 End 键移动窗口到希望的位置，然后按回车键。

2. 改变窗口的大小

使用鼠标器改变窗口大小，将鼠标器指针指向命令窗口的右下角，按住鼠标器按钮进行拖动，命令窗口将随之改变其大小，待调整到希望的大小时释放鼠标器按钮。

使用键盘改变窗口大小，按下 Ctrl-F8 键使命令窗口的边界变成闪烁，按方向键后，命令窗口将随之改变其大小，待调整到希望的大小时，按回车键。

1.3 FoxPro 语言的成分

每一种计算机语言都有它自身的基本组成成分。FoxPro 含有数百种命令或语句，通过它们实现建立数据库和对数据的各种查询等操作。与其它高级语言相似，FoxPro 还支持多种数据文件。它能够定义和处理不同类型的数据，并可通过程序实现对数据文件中的数据的操作。同时它还拥有数百种功能丰富、使用方便的函数。

下面介绍 FoxPro 的主要组成成分。

1.3.1 命令

在 FoxPro 语言中，对数据的操作都是由命令来完成的。命令相当于一般高级语言中的语句，但比高级语言的语句更精炼，功能更强。例如，有下列一组 FoxPro 命令：

```
USE student
COUNT ALL FOR 数学成绩<60
LIST ALL 姓名,数学成绩 FOR 数学成绩<60 TO PRINTER
CLOSE ALL
```

这四条命令的功能分别是：

(1) 打开名字为 student 的数据库文件，以便对该数据库文件进行操作。

(2) 在已打开的数据库文件 student 中，统计数学成绩在 60 分以下的人数。

(3) 用打印机输出 60 分以下同学的姓名和数学成绩。

(4) 关闭数据库文件 student，保存在磁盘上。

如果使用一般的高级语言(如 BASIC，FORTRAN 等语言)，完成上述功能需要很多的语句才能完成描述“打开数据库文件”、“统计”、“列表显示”和“关闭数据库文件”等操作的详细过程。

下面列出 FoxPro 数据操作命令的一般语法形式：

```
<命令动词> [<表达式表>] [<范围>] [FOR <条件>] [WHILE <条件>]
           [TO FILE <文件名> | TO PRINTER | TO ARRAY <数组表> |
           TO <内存变量>]
           [ALL [LIKE | EXCEPT <通配符>]] [IN <别名>]
```

可以看出，它可分成八个部分。下面分别进行说明。

1. 命令动词。它是 FoxPro 的命令名，用来指示计算机要完成的操作。例如上面例子中的“USE”，“COUNT”等。

2. 表达式表。它是一个或多个由逗号分隔开的表达式，用来指示计算机执行该命令所操作的结果参数。例如，在上面例子的第三条命令中，表达式表为"姓名，数学成绩"。

3. 范围。它表明命令可以操作的记录。范围可有下列四种选择：

RECORD <n> —— 仅对指定的 n 号记录进行操作。

NEXT <n> —— 对从当前记录开始的 n 条记录进行操作。

ALL —— 对数据库文件中所有记录进行操作。

REST —— 对从当前记录开始到数据库文件结尾的记录进行操作。

4. FOR <条件>。它告诉 FoxPro，命令仅对满足条件的记录进行操作。如果使用 FOR 子句，FoxPro 将记录指针重新指向数据库文件顶，并且用 FOR 条件与每条记录进行比较。

5. WHILE <条件>。在数据库文件中，从当前记录开始，按记录顺序从上向下处理，直到不满足条件为止。

6. TO ...。它控制命令操作结果的输出。有些命令允许操作结果向文件输出，有些命令允许操作结果向打印机输出，有些命令允许操作结果向内存变量输出。

7. ALL [LIKE|EXECPT <通配符>]。它告诉 FoxPro，包括或不包括与通配符相匹配的文件、字段或内存变量。通配符是与文件名、字段名或内存变量名相符合的一般形式。在通配符中可使用 * 和 ?。? 代表任何单一的字符，* 代表一组任意字符。

8. IN ...。它允许在当前工作区下操作其它工作区中的数据库文件。IN 子句可含有别名、字母、数字或赋给别名、字母、数字的表达式。

为了方便用户，FoxPro 命令作了以下一些规定：

1. 每条命令以命令动词开头，必须严格符合本书所介绍的命令语法格式。

2. 命令中的子句可按任意次序排放。如下面三条命令是等价的：

```
LIST 姓名,性别,职称 NEXT 5 FOR 教研室 = '计算机'
LIST NEXT 5 姓名,性别,职称 FOR = '计算机'
LIST FOR 教研室='计算机' NEXT 5 姓名,性别,职称
```

3. 命令中的子句可由若干个空格隔开，每个空格也算一个字符。

4. 命令行的总长度最大可达 2048 个字符。

5. 命令动词和FoxPro保留字均可用4个以上字母简写。例如，下面三条命令是等价的：

```
DISPLAY MEMORY
DISPLA MEMO
DISP MEMO
```

6. 可以以小写、大写和大小写混合的形式输入命令行。

7. FoxPro 中没有绝对不能使用的单词，但最好不使用 FoxPro 命令动词和保留字作为字段名、内存变量名、数组名以及文件名，否则将有可能带来不必要的混乱。

例如，以 status 命名的字段是无法由 LIST status 命令显示的。因为该命令将显示当前 FoxPro 的系统状态，而不是显示 status 字段的内容。

8. 不能使用单个字母 A 到 J 作为数据库文件名，因为它们是 FoxPro 的保留字，作为

别名(ALIAS)。

9. 设计程序时，DO WHILE/ENDDO，FOR/ENDFOR，SCAN/ENDSCAN，DO CASE/ENDCASE，IF/ENDIF，PRINTJOB/ENDPRINTJOB 以及 TEXT/ENDTEXT 命令必须配对使用。

10. 程序设计时，如果命令太长，一行写不下，可分几行写，但每行(最后一行除外)末尾要使用一个分行符";"。

在命令语法格式中，本书对所使用的符号作如下约定：

[ ]：方括号，表示是可选的项目。

< >：角括号，表示必须要选的项目。

|：竖线号，表示两个项目中选择其中一个项目。

...：省略号，表示前项可继续重复多次选择。项与项之间用逗号隔开。

1.3.2 文件

FoxPro 支持多种类型的文件，文件名遵守操作系统的文件命名规则，必须以字母开头，可含有字母(大小写均可)、数字和下划线(_)，由不超过 8 个字符的标识名加上 3 个字符的类型名(或扩展名)组成。

例如：PEOPLE.DBF 是一个有效的文件名。其中 PEOPLE 为文件的标识名；DBF 为类型名，也称为扩展名或文件名后缀，表示文件的类型。

在 FoxPro 文件中，数据库文件是最基本的文件，它存放用户关心的数据。在关系数据库系统中，数据库文件是以表格的形式表现出来的，二维表格的每一竖栏(列)为一个"字段"，每一横行为一个"记录"。数据库文件的扩展名为 DBF。

数据库明细文件是数据库文件的辅助文件。当数据库文件含有明细型字段时，系统自动为此数据库文件生成一个明细文件，用于保存明细型字段中大块的数据。数据库明细文件的扩展名为 FPT。

1.3.3 变量

FoxPro 使用两种变量：字段变量和内存变量。字段变量是存在数据库文件(.DBF)中的，每个数据库文件都包含有若干字段变量。

内存变量不同于字段变量，它存在内存之中，独立于数据库文件。它的设置为用户使用数据库系统带来极大的方便。内存变量是一种临时工作单元，通常用来保存中间结果或保存对数据库进行某种分析处理后得到的数据结果，或者用于控制流程，要用时可以随时定义，不用时又可以释放。

FoxPro 变量由变量名、变量类型和变量宽度三部分组成。

变量名由汉字、字母、数字或下划线组成，最多不超过 10 个字符，必须以字母或汉字开头。

例：下面是合法的变量名：

CLIEN_ID　姓名　CITY　A1

下面是非法的变量名：

1A　姓名　ZIP-CODE

FoxPro 的字段变量有七种类型，它们分别是：

字符型字段
数字型字段
浮点型字段
日期型字段
逻辑型字段
明细型字段
通用型字段

内存变量有四种类型，它们分别是：

字符型内存变量
数字型内存变量
日期型内存变量
逻辑型内存变量

1. 字符型(C 型)

字符型变量的数据值是由可显示的汉字和字符(包括空格)构成的。用字母 C 表示。系统允许一个字符型字段宽度为 1 到 254 个字符，字符型内存变量宽度可以是 0 到 64K(1K=1024)个字符。字符型内存变量可以存储来自明细型字段的数据。

2. 数字型(N 型)

数字型变量的数据值是由正负号、小数点和 0 到 9 的数字构成的，可进行算术运算的数据，用字母 N 表示。

数字型变量宽度为该变量最大整数位、最大小数位之和，小数点和正负号分别算作一个数位。小数宽度为最大小数位。数字型变量只有前 16 位数字有意义。

3. 浮点型(F 型)

浮点型变量与数字型变量完全相同，它只是为了保持兼容 dBASE IV 而保留，内存变量不存在浮点型。

4. 日期型(D 型)

日期型变量是用于存放日期的数据，用字母 D 表示。通常采用美国格式：月/日/年，其中：月、日、年均为两位数字。

日期型变量宽度固定为 8 个字节。

5. 逻辑型(L 型)

逻辑型变量的数值是由逻辑真(.T.)和假(.F.)值所构成的，用字母 L 表示。接受 T，t，Y，y 为逻辑真(.T.)值；接受 F，f，N，n 为逻辑假(.F.)值。

逻辑型变量宽度固定为 1 个字节。

6. 明细型(M 型)

内存变量没有明细型，只有字段变量有明细型。明细型字段变量是用于存放大块的 ASCII 码数据的，数据被存放在数据库文件的明细文件(.FPT)中。它在数据库文件中用 memo 一词指明，用字母 M 表示。明细型字段变量在数据库文件中的宽度固定为 10 个

字节。

明细型字段存放数据的宽度是可变的，最大长度取决于可用磁盘空间的大小。

7. 通用型(G 型)

通用型变量用于保存二进制、图象等数据，用字母 G 表示。

DOS 版 FoxPro 忽略通用型变量，Windows 版 FoxPro 没有通用型内存变量。

1.3.4 系统内存变量

系统内存变量是 FoxPro 自动建立和管理的内存变量，用户是不能建立和删除的，它主要用于控制打印机和屏幕的输出格式，以及保存打印机的设置。

所有系统内存变量都是以下划线(_)开头，以区别于普通的内存变量。

启动 FoxPro 时，FoxPro 系统自动为各个系统内存变量赋予默认的值，用户可在命令窗口下和程序中随时使用赋值命令(STORE)改变系统内存变量的值。

1.3.5 常数

常数是在命令和程序运行过程中不改变的数据，共有四种类型：字符型、数字型、日期型和逻辑型。

1. 字符型常数

字符型常数是由可打印的字符和汉字构成的，必须用定界符括起来。有三种定界符：

(1) 单引号 ' '

(2) 双引号 " "

(3) 方括号 [ ]

应当注意，定界符必须成对使用。当字符型常数本身含有定界符时，应该选择另外一对异于该定界符的定界符作为真正的定界符。

下面都是合法的字符型常数：

"12.35"、"红的"、'Y'、[abc"def"ghi]

2. 数字型常数

数字型常数是由正负号、0 至 9、小数点构成的。例如，下面是合法的数字型常数：

12.34，150，0.000025，140000

3. 日期型常数

日期型常数一般格式为{月/日/年}。月、日、年分别为二位数字。大括号{ }表示日期型数据，等同于 CTOD()函数。例如：

{12/20/59}

等价于

CTOD("12/20/59")

空日期常量为：

{ / / }

或

{ }

4. 逻辑型常数

逻辑型常数是由逻辑真、假值构成的。逻辑值两边必须有圆点。例如：

.T.、.t.、.Y.、.y.代表“真”

.F.、.f.、.N.、.n.代表“假”

1.3.6 表达式

FoxPro 使用四种类型的运算符：算术、比较、逻辑和字符串运算符。

1. 算术运算

算术运算符对表达式进行算术运算，产生数值结果。它包括七种运算符：

加号+	余数%
减号-	乘方**（或^）
乘号*	括号（）
除号/	

运算规则：

先乘除后加减；乘方优先于乘除；函数优先于乘方，括号最优先；同级运算符自左至右顺序运算。

括号无大、中、小之分，一律用圆括号。圆括号()内可再套用圆括号()。

2. 比较运算

比较运算符对两个表达式进行比较运算，产生逻辑型(真、假值)结果。它包括八种运算符：

小于<	小于或等于<=
大于>	大于或等于>=
等于=	子字符串比较$
不等于<>、!=或#	字符串比较==

使用比较运算符时，应注意以下几点：

(1) 比较运算符可用在字符、数字和日期型表达式中，但用于比较的两个表达式数据类型必须相同。

(2) 数字型数据是按其数值大小进行比较；字符型数据是按其 ASCII 码值的顺序进行比较；日期型数据是按年、月、日的先后进行比较。

例：

```
? 123<=456                          && 数字型比较，结果为.T.
? 'AB'<'BB'                         && 字符型比较，结果为.T.
? {01/01/60}<={01/07/86}            && 日期型比较，结果为.T.
```

(3) 子字符串比较$。如果 A 和 B 都是字符串，而且 A 与 B 相同或 A 是 B 的子串，则 A$B 的结果是真值。

例：

命　令	执行结果
? "this"¥"this is a string"	.T.
? "this"¥"THIS IS A STRING"	.F.
? "this is a string"¥"this is a string"	.T.

(4) 比较运算符"="用于字符型数据比较,它与 SET EXACT 的状态有关。

执行 SET EXACT OFF,SET EXACT 设置为 OFF 状态,当比较两个字符串时,从字符串的第一个字符到最后一个字符逐个比较,到达第二个字符串最后一个字符时,比较结束。如果前面的字符相同,则认为两个字符串相同。

执行 SET EXACT ON,SET EXACT 设置为 ON 状态时,只有两个字符串的字符个数和每个字符完全相同,才认为两个字符串相同。SET EXACT 默认为 OFF 状态。

命　令	执行结果
SET EXACT OFF	
? "abcd"="abcd"	.T.
? "abcd"="abcd "	.F.
? "abcd"="abcd"	.T.
SET EXACT ON	
? "abcd"="abcd"	.F.
? "abcd"="abcd "	.F.
? "abcd"="abcd"	.T.

(5) 字符串比较符"=="与比较符"="不同,它与 SET EXACT 的状态无关,只有当比较符"=="两边字符串完全相同时,返回结果才为真值。

例:

命　令	执行结果
SET EXACT OFF	
? "abcd"=="abcd"	.F.
? "abcd"=="abcd "	.F.
? "abcd"=="abcd"	.T.
SET EXACT ON	
? "abcd"=="abcd"	.F.
? "abcd"=="abcd "	.F.
? "abcd"=="abcd"	.T.

3. 逻辑运算

逻辑运算符对一个或两个逻辑型表达式进行逻辑运算,产生逻辑型(真、假值)结果。它包括四种运算符:

逻辑与 .AND.

逻辑非 .NOT.(或!)

逻辑或 .OR.

括号（）

运算规则：

先逻辑与后逻辑或；逻辑非优先于逻辑与；括号最优先。括号无大、中、小之分，一律用圆括号()，可以在圆括号内再套用圆括号()。

各种逻辑运算符运算结果归纳如下：

逻辑型表达式 A	逻辑型表达式 B	A.AND.B	A.OR.B	.NOT.A（! A）
T	T	T	T	.F.
T	F	F	T	.F.
F	T	F	T	.T.
F	F	F	F	.T.

4. 字符串运算

字符串运算符对两个字符型数据进行连接运算，产生一个新的字符型数据。它包括两个运算符：

连接运算符　　+

压缩空格运算符　　-

例如：

若：变量 A 的内容为"this　"

　　变量 B 的内容为"is　a　string"

则：A + B = "this　is　a　string"

字符串运算符-，其作用是将两个字符串连成一个字符串。与"+"不同的是，它将运算符前字符串尾部的空格移加到运算符后字符串的尾部。

例如：

若：变量 A 的内容为："this　"

　　变量 B 的内容为："is　a　thing"

则：A - B = "this is　a　string　"

5. 表达式

在 FoxPro 中，由运算符和括号将常数和函数连接起来的有意义的式子称为表达式。当不同类型的运算符在同一表达式中出现时，运算规则为：

比较运算符优先于逻辑运算符；算术、字符串运算符优先于比较运算符；括号最优先，同级自左至右顺序运算。括号无大、中、小之分，一律用圆括号()，可以在圆括号内再套用圆括号。

同变量一样，表达式按运算结果可分为：字符型表达式(C 型)、数字型表达式(N 型)、逻辑型表达式(L 型)和日期型表达式(D 型)。

例：

字符型表达式：

"姓名"+姓名

数字型表达式：

88－52＊63　　2＊SQRT(4)

逻辑型表达式：

0<2　　mselect¥"Aa1"

日期型表达式：

CTOD('01/01/90')

1.3.7 函数

函数用来实现某些特定的运算或操作。FoxPro 提供 100 多种函数，从概念来讲，FoxPro 函数与数学中的函数没有什么根本区别。

注意以下几点：

(1) 所有函数必须跟随有圆括号，无论函数是否需要参数(除宏代换函数 & 外)。

(2) 每个函数可有一个返回值。

(3) 传送给函数的参数有一定的数据类型，必须按要求的数据类型传送参数值。

表 1.1 仅列出部分返回值为数字型的函数。

表 1.1　FoxPro 函数

函数名	形　式	功　能
AT	AT(<字符型表达式 1>,<字符型表达式 2>)	子字符串查找
ABS	ABS(<数字型表达式>)	求绝对值
EXP	EXP(<数字型表达式>)	求指数
INT	INT(<数字型表达式>)	求整
LOG	LOG(<数字型表达式>)	求自然对数
SQRT	SQRT(<数字型表达式>)	求平方根
MAX	MAX(<数字型表达式 1>,<数字型表达式 2>, ...)	求最大值
MIN	MIN(<数字型表达式 1>,<数字型表达式 2>, ...)	求最小值
ROUND	ROUND(<数字型表达式 1>,<数字型表达式 2>)	四舍五入
MOD	MOD(<数字型表达式 1>,<数字型表达式 2>)	求模

FoxPro 中有些函数返回值无实际意义，对于这些函数的调用可用下面的格式：

=函数名

例：

=ADIR(asubdir,"E:*.*","D")

此函数功能是将 E 驱动器的根目录下所有文件和子目录名存入数组中。

对于不关心返回值的函数，也可用上述格式调用。

1.4 赋值和输出命令

在 FoxPro 中，显示命令？/?? 和赋值命令 STORE 是用得最多的命令，应很好掌握。

1.4.1 赋值命令 STORE

赋值命令 STORE 的主要功能是给内存变量提供数据，下面对它作一些说明。

(1) STORE 命令可以用于建立内存变量，并赋予内存变量初值。例如：

```
STORE 100 TO mnum
```

它的作用是：建立内存变量 mnum，赋予初值 100。

(2) STORE 命令也可用为已经建立的内存变量重新赋值。例如：

```
STORE 2 * mnum TO mnum
```

它的作用是：为已经建立的内存变量 mnum 重新赋值。假定内存变量 mnum 原值为 100，则新值将变为 200。星号表示乘。

(3) 一条 STORE 命令可以同时建立若干个内存变量或同时为若干个内存变量重新赋值。例如，建立内存变量 ma，mb，mc 和 md，初始化为 0：

```
STORE 0 TO ma, mb, mc, md
```

(4) 当 STORE 命令给单个内存变量提供数据时，可以写成＜内存变量＞＝＜表达式＞形式。

例如：

```
mnum = 10 * mnum
```

等价于

```
STORE  10 * mnum  TO  mnum
```

1.4.2 数据显示命令？/??

数据显示命令？/?? 可以完成以下操作：

1. 显示变量、表达式和常数的值

(1) 如某一变量已经赋值，则可用单问号？显示该变量的值。

命　令	执行结果
STORE 34 TO mvar	34
? mvar	34

(2) 用单问号？可以显示表达式的值。

命　令	执行结果
STORE 4 TO mvar1	4
STORE 6 TO mvar2	6
? (mvar1+mvar2)/2	5

STORE "ABC" TO ma	ABC
STORE "DEF" TO mb	DEF
? ma+mb	ABCDEF

可以看出，?命令具有计算和输出显示的双重功能，在遇到表达式时，它先计算后显示输出计算的结果。

(3) 用? 命令可以原样显示输出常数。

例：显示输出数字常数和字符串常数。

命　令	执行结果
? 4	4
? "THIS IS A STRING"	THIS IS A STRING

应当注意，这里有引号是用来确定字符串起始和终止界限的，它本身不作为字符串中的字符。

2. 显示若干个变量、表达式和常数的值

(1) 用一条? 命令可以显示若干个变量的值。用一条? 命令可以显示若干个表达式的值，也可用一条? 命令显示若干个常数。变量之间、表达式之间、常数之间需用一个逗号隔开。

命　令	执行结果
STORE "AB" TO mvar1	AB
STORE "CD" TO mvar2	CD
STORE "EF" TO mvar3	EF
? mvar1,mvar2,mvar3	AB CD EF
STORE 4 TO mnum1	4
STORE 3 TO mnum2	3
? mnum1+mnum2,mnum1−mnum2	7　　1

(2) 用一条? 命令可以同时显示若干变量、若干表达式、若干常数的值。

命　令	执行结果
STORE 32 TO mnum1	32
STORE 12 TO mnum2	12
? "mnum1+mnum2=", mnum1+mnum2	Mnum1+Mnum2=　44

3. 用? 命令可以输出一个空行

命　令	执行结果
? "3 * 4 =", 3*4	3 * 4 = 12
?	(空一行)
? "3 * 5 =", 3*5	3 * 5 = 15

4. 双问号?? 命令的使用

双问号?? 命令的使用等同于单问号? 命令的使用。它们的区别是:单问号? 命令从下一行开始显示输出,双问号?? 命令在当前光标位置开始显示输出。

```
? "A","B","C"
?? "D","E","F"
? "G","H","I"
?? "J","K","L"
? "M","N","Q"
?? "P","Q","R"
```

结果:

```
A  B  C  D  E  F
G  H  I  J  K  L
M  N  Q  P  Q  R
```

? 和?? 命令后面还可以跟许多子句和选择项,有关? 和?? 命令的更详细介绍参见第十章。

1.5 退出 FoxPro

在完成 FoxPro 操作后,必须在关闭计算机之前退出 FoxPro。在命令窗口中输入 QUIT,并按回车键即可。

用 QUIT 命令正常退出 FoxPro 后,自动关闭所有打开的文件,将变化的数据存入磁盘保存。

注意:如果没有正常退出 FoxPro,如电源掉电、计算机死机等,则有可能造成数据丢失或损坏。

第二章　建立数据库和数据录入

从本章起，将陆续介绍 FoxPro 的命令。

FoxPro 将数据保存在数据库文件中。一个数据库应用管理系统可以有多个数据库文件，数据库文件的多少取决于管理的数据和用户处理的需求。数据库文件是构成数据库应用管理系统的重要部分。

在介绍数据库文件建立之前，先讨论数据库文件的工作区及其操作。

2.1　数据库文件的工作区

2.1.1　工作区和别名

在对数据库文件进行操作之前，必须先打开该数据库文件。打开数据库文件时，FoxPro 为其分配一个工作区，每个工作区可以包含一个打开的数据库文件，必须在不同的工作区中，同时打开不同的数据库文件。每个数据库文件占用一个工作区。

FoxPro 提供了 225 个工作区，各个工作区是相互独立的，均有各自的记录指针。在同一时刻，每个工作区只能打开一个数据库文件。

注意，虽然 FoxPro 允许同时打开多至 225 个数据库文件，但实际上是不可能的。因为，同时允许打开的数据库文件数目要受到操作系统的限制，如：可用内存的数量和 CONFIG. SYS 中 FILES 的设置。

FoxPro 规定 225 个工作区分别为 1 到 225。为了便于工作区间的访问，系统还为前 10 个工作区规定了别名 A 到 J，工作区 11 到 225 别名为 W11 到 W225。FoxPro 最初的当前工作区为 1。

2.1.2　工作区选择

可用 SELECT 命令选择所需的当前工作区。SELECT 命令的一般格式为：

SELECT <工作区号>|<别名>

工作区号：为数字型表达式，取值范围是 1—225。

别名：为字符型表达式。有三种等效的别名，一种是系统固定的别名，分别为 A—J、W11—W225；另一种是使用 USE 命令打开数据库文件时所定义的；第三种别名是在未规定数据库文件别名的情况下，默认数据库文件（没有扩展名）名为别名。

使用 SELECT 命令选择当前工作区，对每个工作区的记录指针和打开的数据库文件没有任何影响。

例：下面一组命令

```
SELECT  1
USE  people
SELECT  2
USE wife
SELECT  A
```

第一条命令将工作区 1 设置为当前工作区。

第二条命令在工作区 1 打开数据库文件 people。

第三条命令将工作区 2 设置为当前工作区。

第四条命令在工作区 2 打开数据库文件 wife。

第五条命令将工作区 1(别名为 A)设置为当前工作区。

如果将最低未使用的工作区设置为当前工作区，可输入下面的 SELECT 命令：

```
SELECT  0
```

一般情况下，如果访问当前工作区以外的其它工作区打开的数据库文件中的某个字段，而又不想用 SELECT 命令转换工作区，仍保留当前工作区，则可通过下列形式指定其它工作区中数据库文件的字段：

<别名>－><字段名>

或

<别名>.<字段名>

2.2 定义新的数据库

数据库文件由两部分组成，结构定义部分和数据部分，结构定义部分描述了数据存放的形式以及存放的顺序，包括数据库文件名称，数据库文件由哪些字段构成，每个字段的名字、类型、宽度和小数位位数等。

数据部分是数据库文件所要保存的主体，是按照数据结构的描述进行有序存放的，数据部分由记录构成，字段是构成记录的基本单元。一个数据库文件最多可有 255 个字段。字段可以是字符型、数字型、浮点型、日期型、逻辑型、备注型和通用型。

2.2.1 CREATE 命令

确定了数据库文件中需要包含的字段之后，就可以用 CREATE 命令来定义一个新的数据库文件。

CREATE 命令是一条全屏幕编辑命令，它的一般格式为：

CREATE <文件名>

下面举例说明这一条命令的用法。

假定建立数据库文件 people.dbf，它需要反映以下基本信息：编号、姓名、性别、出生年月、婚否、所在部门、职务、月收入和工作成绩。各数据的要求如下：

1. 编号，字符型数据，4 位数字，占 4 个字节。

2. 姓名，字符型数据，最大宽度为 6 个字节(3 个汉字)。

3. 性别，字符型数据，最大宽度为 2 个字节(1 个汉字)。

4. 出生年月，日期型数据，宽度为 8 个字节。

5. 婚否，逻辑型数据，宽度为 1 个字节，已婚者为真(.T.)，未婚者为假(.F.)。

6. 所在部门，字符型数字，最大宽度为 12 个字节(6 个汉字)。

7. 职务，字符型数据，最大宽度为 6 个字节(3 个汉字)。

8. 月收入，数字型数据，工薪阶层月收入不超过 10000.00 元，可设宽度为 7 个字节，其中小数位位数占 2 个字节。

9. 工作成绩，明细型数据。

建立数据库文件 people，在 FoxPro 命令窗口中输入：

```
CREATE people.dbf
```

屏幕显示图 2.1 所示的对话框。

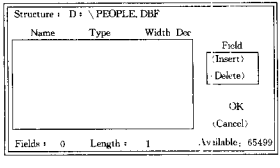

图 2.1　建立数据库文件

对话框中最下一行显示数据库文件中的字段数(Fields)，占用字节总数(Length)和剩余的字节数(Available)。开始时已定义的字段数为 0，占用字节数为 0，剩余字节数为 65500，光标这时在第一行的 Name 列上。

首先键入“编号”，然后按回车键，表示字段名输入完毕，光标移动到字段类型 Type 列。输入代表字段类型的字母 C，完成字段类型的输入，光标移动到宽度 Width 列，键入数字 4，并按回车键，光标跳过小数位 Dec 列，自动又移动到下一行的 Name 列，表示第一个字段定义完成。FoxPro 准备接收第二个字段的定义。

按照上述过程反复输入，直至 9 个字段定义完成为止。

当输入逻辑型字段时，FoxPro 自动填写字段宽度(Width)为 1；当输入日期型字段时，FoxPro 自动填写字段宽度(Width)为 8；当输入明细型和通用型字段时，FoxPro 自动填写字段宽度(Width)为 10。

输入完上述 9 个字段后，屏幕显示如图 2.2 所示的画面。

当确认定义的字段没有错误时，使用 Tab 键将光标移动到 OK 功能选择项，并按回车键，或者撤击 OK 功能选择项，屏幕显示如图 2.3(a)所示的对话框。