面向对象数据库教程

需要全本请访问 www.ertongbook.

第 1 章　Visual FoxPro 6.0 数据库的基本概念

在这一章中将介绍数据、信息和数据处理的基本概念，关系型数据库，Visual FoxPro 的特点以及 Visual FoxPro 的启动和退出操作。

1.1　数据、信息和数据处理

随着商品经济的发展，科学技术的进步和激烈的市场竞争，企业经营管理中的信息量倍增，决策难度也随之加大。人们越来越重视生产和经营活动中的信息作用，把信息视为企业的又一重要资源，十分重视信息的收集、加工和使用，从而也促进了信息科学的诞生和发展。数据库管理技术是信息科学的重要组成部分。下面就从数据、信息和数据处理基本概念开始介绍。

为了记载信息，人们使用了各种物理符号及其组合来表示信息，这些符号及其组合就是数据。数据的形式多种多样，如数字数据、文字数据、声音数据和图像数据等。信息是一种已被加工为特定形式的数据，这种数据形式对于使用者来说是有意义的，而且对当前和将来的决策具有明显的实际价值。

所谓数据处理，包括对数据的收集、记载、分类、排序、存储、计算或加工、传输等项内容。

当今的时代，百分之七十的计算机用于数据处理。数据处理方法和技术的研究已成为计算机科学的重要课题。其中，数据库技术已成为数据处理的最主要的方法和工具。

1.2　关系型数据库

数据库技术的基本思想是对数据实行集中的、统一的、独立的管理，用户最大限度地共享数据资源。数据库中的数据是有结构的数据集合，如记录教师档案的数据集合，如下表所示：

记录号	编号	姓名	性别	职称	基本工资	出生日期	籍贯	简历	相片
1	1101	刘文学	.T.	副教授	870	09/01/62	北京市	Memo	Gen
2	1102	张文良	.F.	讲师	780	08/01/70	天津市	Memo	Gen
3	2101	刘金山	.F.	教授	1200	01/06/60	上海市	Memo	Gen
4	2102	赵志伟	.T.	讲师	780	01/07/73	沈阳市	Memo	Gen
5	3101	徐志博	.T.	副教授	1000	01/02/42	山东青岛市	Memo	Gen
6	3102	何志秀	.F.	副教授	820	01/05/65	天津市	Memo	Gen
7	4101	徐志宏	.T.	副教授	1000	03/01/43	河北德州市	Memo	Gen
8	4102	许文	.T.	讲师	870	02/04/70	辽宁大连市	Memo	Gen
9	5101	李宏	.F.	教授	1300	05/04/41	天津市	Memo	Gen
10	5102	周仁良	.T.	讲师	850	09/08/69	北京市	Memo	Gen

这是一张记录教师数据的二维表，表的每一行都记录了一名教师的相关数据，在数据库中称作记录。而表的每一列称作项，是描述教师属性的同类型数据项，如：编号、姓名、性别等。

这张由相关数据记录构成的二维表，记载了某校教师的档案数据。采用同样的方法也可以构成描述教师任课的二维表。对于学生同样可以建立学生档案及学习成绩档案的二维表。为存储各表，就须给每个表都起个名字，可称作关系名或表名。对于教师档案表和教师任课表可以通过教师编号建立起对应关系；同样地，对于学生的档案表和成绩档案表也可以通过学号建立起对应关系。此外，为了检索符合要求的教师或学生记录，也应在记录间建立关系。这种关系在数学上就称作关系模型。关系模型应满足如下要求：

(1) 表中每一项代表一个数据项，不允许有重复项。

(2) 表中每一列里所有数据应具有相同类型。

(3) 表中各列不能同名。

(4) 表中不允许有重复行。

(5) 表中行和列的位置改变不影响它们的信息内容。

通过关系模型建立的数据库就被称作关系数据库。关系数据库应具有筛选、投影和连接功能。筛选就是从表中能筛选出符合要求条件的所有记录，例如从考生档案表中筛选出满足录取条件的学生记录。投影是从记录的所有数据项中选出指定的数据项，如只从教师档案表中选定编号、姓名与籍贯 3 项数据。连接是从两个表中，选取满足连接条件的指定数据项构成新表中诸条记录。如从学生档案表中选取档案数据，从成绩档案表中选取学习成绩数据。依两表中学号相同的连接条件，组成每名学生的档案数据和成绩数据于同一记录的新表。能完成以上所述功能的软件就是关系数据库管理系统。最新的关系型数据库管理系统就是面向对象的由事件驱动的数据库管理系统 Visual FoxPro。

计算机引进数据库后，由用户、数据库管理系统和数据库共同构成数据库系统。用户使用数据库进行数据处理，目的是获取信息；而数据库管理系统是帮助用户达到这一目的的工具和手段。

1.3 Visual FoxPro 6.0 数据库管理系统的特点

微机数据库管理系统，已由最初的 dBASE 经 FoxBASE、FoxPro 发展到 Visual FoxPro，功能日益强大，操作更加灵活。从数据库应用程序的设计方面看，正在经历一次程序设计思想方面的革命，就是从已被广泛采用的面向过程的结构化程序设计方法，发展到面向对象由事件驱动的程序设计方法。下面就来介绍 Visual FoxPro 6.0 数据库管理系统的主要特点。

1. 面向对象由事件驱动的应用程序设计方法

以前广泛采用面向过程的结构化程序设计方法设计应用程序，它要求程序设计人员必须具有较高的程序设计技巧。此外，采用这种方法设计的数据库应用程序很难实现多任务操作。为此，Visual FoxPro 6.0 提供了面向对象由事件驱动的全新程序设计方法。采用该方法开发数据库应用软件不仅简化了设计，并且用户界面操作灵活、样式美观。

2. 提供可视设计工具

为提高应用程序设计效率,减轻设计人员劳动强度，Visual FoxPro 6.0 提供了用于应用开发的各种设计器、向导、工具栏、菜单和生成器。这些设计工具不仅减轻了设计人员劳动量，而且其可视性为不具备应用程序设计技术的广大用户，获得了开发应用程序的能力。

3. 增强了项目及数据库管理功能

Visual FoxPro 在创建项目的同时，生成了该项目的项目管理器。由它全面管理项目中数据库、应用程序及文档等，使数据库的应用和开发更加方便。此外，其数据库的管理功能也更加强大，提供了过去只在大型计算机的数据库管理系统中才具有的功能，如设置表字段的默认值、字段及记录的有效性规则及表间记录的参照完整性规则等，因此也极大地提高了数据的安全性。

4. 增强了网络功能

Visual FoxPro 6.0 所提供的视图和表单，不仅可访问和更新本地数据库中的数据，还可以访问和更新网络中就近服务器中的数据。

此外，Visual FoxPro 6.0 还增强了许多其他方面的功能，在此就不一一列举了。我们相信，Visual FoxPro 还会进一步丰富它的功能，从而成为用户首选的数据库软件。

1.4 启动和退出 Visual FoxPro 6.0

在使用 Visual FoxPro 6.0 之前，首先要启动 Visual FoxPro 6.0 ，用后要关闭退出。

1.4.1 启动 Visual FoxPro 6.0

在视窗下，启动 Visual FoxPro 6.0 的步骤如下所述：

(1) 单击屏幕左下角的“开始”按钮。

(2) 将鼠标指针指向“开始”菜单的“程序”项，然后指向 Microsoft Visual FoxPro 6.0 组中的 Visual FoxPro 6.0 选项。

(3) 单击 Visual FoxPro 6.0 选项，开始启动 Visual FoxPro 6.0。当在屏幕上出现命令窗口，并在其中显示光标时，标志启动已经完成。此后可以接受并执行 Visual FoxPro 6.0 的任何命令或操作。如果在安装后初次启动 Visual FoxPro 6.0，则在出现命令窗口之前，还会显示 Visual FoxPro 6.0 的操作选择对话框。该框用于选择启动后直接进入的操作，它设有 5 个选择按钮和 1 个复选框。如果要新建项目应单击第一个按钮；查找示例程序单击第二个按钮；查询 Visual FoxPro 联机帮助文档单击第三个按钮；要打开已有项目单击第四个按钮；要关闭此屏进入 Visual FoxPro 主窗口，应单击第五个按钮。如果今后启动不再需要显示此对话框时，选择该复选框，此后在启动 Visual FoxPro 时就将直接进入其主窗口。

1.4.2 退出 Visual FoxPro 6.0

退出 Visual FoxPro 6.0 的步骤如下所述：

(1) 切换到 Visual FoxPro 6.0 的命令窗口。

(2) 单击“文件菜单”中的“退出”项或在命令窗口中键入并执行 QUIT 命令。 必须指出的是：按正常操作步骤退出 Visual FoxPro 6.0，都将自动保存在缓冲区中尚未存入数据库的数据，并完成表、数据库及项目的关闭操作。但如果非正常或意外退出就有可能丢失数据或破坏数据库。望读者一定要按正常步骤退出 Visual FoxPro 6.0，切忌直接关断主机电源。

1.5 思考与练习

(1) 以实例说明数据、信息和数据处理。

(2) 应满足哪些要求的数据库可称作关系型数据库？

(3) Visual FoxPro 6.0 关系数据库有哪些特点？

(4) 举例说明关系型数据库中的筛选、投影和连接功能。

(5) 练习 Visual FoxPro 6.0 的启动和正常退出。

需要全本请访问 www.ertongbook.

第 2 章　Visual FoxPro 6.0 的常量、变量、表达式和函数

在这一章中将介绍在数据库操作及应用软件开发中经常涉及到的一些十分重要的概念和基本知识，它包括：数据类型、常量、变量、表达式和函数。只有正确地理解和掌握这些重要的概念和基本的知识，才能准确地使用命令和开发数据库应用程序。

2.1　数 据 类 型

记载信息的符号组合称为数据，人们用数据来描述实体的对象及其属性。数据类型是简单数据的基本属性，是一个重要的概念，因为只有相同类型的数据之间才能直接运算，否则就会发生数据类型不匹配的错误。

Visual FoxPro 是一种关系型数据库管理软件，在关系型数据库中把描述每一实体集合的数据表示成一张二维表。例如，表示教师档案的一张二维表如下所示。

记录号	编号	姓名	性别	职称	基本工资	出生日期	籍贯	简历	相片
1	1101	刘文学	.T.	副教授	870	09/01/62	北京市	Memo	Gen
2	1102	张文良	.F.	讲师	780	08/01/70	天津市	Memo	Gen
3	2101	刘金山	.F.	教授	1200	01/06/60	上海市	Memo	Gen
4	2102	赵志伟	.T.	讲师	780	01/07/73	沈阳市	Memo	Gen
5	3101	徐志博	.T.	副教授	1000	01/02/42	山东青岛市	Memo	Gen
6	3102	何志秀	.F.	副教授	820	01/05/65	天津市	Memo	Gen
7	4101	徐志宏	.T.	副教授	1000	03/01/43	河北德州市	Memo	Gen
8	4102	许文	.T.	讲师	870	02/04/70	辽宁大连市	Memo	Gen
9	5101	李宏	.F.	教授	1300	05/04/41	天津市	Memo	Gen
10	5102	周仁良	.T.	讲师	850	09/08/69	北京市	Memo	Gen

表中，共有 10 名教师的档案记录数据。记录有 9 个字段。第一行是描述实体集合的记录型，即记录结构。其中 9 个字段名分别为编号、姓名、性别、职称、基本工资、出生日期、籍贯、简历及相片。

Visual FoxPro 定义 13 种字段类型和 7 种数据类型。13 种字段类型是：字符型、数值型、浮动型、双精度型、整型、货币型、日期型、日期时间型、逻辑型、备注型、通用型、二进制字符型和二进制备注型。而 7 种数据类型是：字符型、数值型、货币型、日期型、日期时间型、逻辑型和通用型。字段属表文件所特有，而数据既可作为数据表文件中的字段内容，也可以作为内存变量的内容或常量使用。下面就来比较详细地介绍常用字段和数据类型。

1. 字符型字段和字符型数据

字符型字段用作存放字符型数据。字符型数据是指一切可印刷的字符，其中包括英文字

母、阿拉伯数字、各种符号、汉字及空格。

上述教师档案中的编号和姓名字段就属于字符型字段，而存储的编号和姓名数据就属于字符型数据。字符型字段宽度为 1~254 个字节。

2. 数值型、浮动型、双精度型和整型字段与数值型数据

数值型字段按每位数一个字节存放数值数据，而浮动型字段存放浮点数值数据。这二者最大宽度为 20 位。整型字段存放整数，最大和最小整数为±2 147 483 647。用该类型字段存放较大整数时可节省存储容量，因为它只占用 4 个字节。双精度字段用于存放双精度数，常用于科学计算，可得 15 位精度，但只占用 8 个字节容量。这些字段中存放的数据统称为数值型数据。

3. 货币型字段和货币型数据

货币型字段用于存放货币型数据，只占用 8 个字节，但可存±922 337 203 685 477.8087 的数，且可有 4 位小数。

4. 日期型字段和日期型数据

日期型字段用作存放日期型数据。常用日期格式为“年.月.日”和“年/月/日”。在教师档案中的出生日期字段就属于日期型字段，其出生年月日数据就是日期型数据。日期型字段有固定宽度，占用 8 个字节，其中年、月、日各占两个字节。

5. 日期时间型字段和日期时间型数据

日期时间字段存放日期时间数据，它占用 8 个字节。其格式为：年.月.日 时:分:秒 AM 或 PM。

6. 逻辑型字段和逻辑型数据

逻辑型字段用作存放逻辑型数据。逻辑型数据只有两个值，即“真”和“假”，常用于作逻辑判断。逻辑型字段有固定宽度，占用 1 个字节。在输入逻辑型数据时可用 T、t、Y、y 中任何一个字符代表“真”，而用 F、f、N、n 中的任何一个字符代表“假”。教师档案中的性别字段就用了逻辑型字段，可用“真”表示男性，而用“假”表示女性。

7. 备注型字段

备注型字段可以存放多种信息，如字符文本、源程序代码等，使其得到了广泛应用。它常用于记录信息可有可无，可长可短的情况，如教师档案中的简历一项，有些人的简历内容可能长一些，而有些人的简历内容可能短一些。此外，备注型字段还可用于提供运行程序时的帮助信息。

记录在备注项中的信息，实际上并不存放在表文件中，而是存放在与表文件同名，但扩展名为“.FPT”的文件中。当创建表文件时，如果定义了备注型字段，则相应的备注文件就会自动生成；当其建成后也会随表文件自动打开。

8. 通用型字段和通用型数据

通用型字段可存放图片、电子表格、声音、设计分析图及字符型数据等。有了这种类型

字段就使得 Visual FoxPro 成为全方位的数据库。和备注字段一样，通用字段数据也存入与表文件同名而扩展名为.FPT 的文件中。

综上所述，FoxPro 定义了 13 种类型的字段，但只有 7 种类型的数据。

2.2 常量与变量

常量在程序执行的过程中不改变其值，而变量在程序执行过程中却允许随时改变其值。FoxPro 定义了常量、用户内存变量、系统内存变量和字段变量。

2.2.1 常量

FoxPro 定义了 5 种类型常量：数值型常量、字符型常量、逻辑型常量、日期型常量和日期时间型常量。

1. 数值型常量

数值型常量可以是整数或实数，例如：76、12.34 等在程序中都是数值型常量。

2. 字符型常量

字符型常量是用定界符括起来的由字符、空格或数字所组成的字串。定界符可以是单引号、双引号或方括号。当某一种定界符本身是字符型常量的组成部分时，就应选用另一种定界符。例如："abcde"、'副教授'、[张志秀]、" " 等在程序中都是正确写法的字符型常量，而{ASD}、'abcde'fg'hi'都是不正确写法的字符型常量。

3. 逻辑型常量

逻辑型常量只有两个值:“真”与“假”。用.T.、.t.、.Y.、.y.表示“真”，用.F.、.f.、.N.、.n.表示“假”，注意两边的小圆点不能丢掉但可以用空格代替。

4. 日期型常量

日期型常量必须用花括号括起来，例如：{^2000-07-12}、{^2000.07.12}、{^2000/07/12}都是日期型常量的正确写法。

5. 日期时间型常量

日期时间型常量也必须用花括号括起来，例如:{^2000-07-12 11:12:32pm}、{^2000.07.12 11:12:32pm}、{^2000/07/12 11:12:32pm}都是日期时间型常量的正确写法。注意：日期和时间数据间必须有空格。

2.2.2 变量

FoxPro 定义了 3 种类型变量：字段变量、用户内存变量和系统内存变量。用户内存变量简称为内存变量。前两种变量的名称用 1～10 个字母、下划线和数字表示，但必须以字母打头；而后一种变量名称由系统规定。

1. 字段变量

字段变量是表文件结构中的数据项。FoxPro 定义了 13 种类型字段变量，这在上一节已经全面介绍过。这里再补充一点，就是字段变量是一种“多值”变量，如教师档案这个表文件有多条记录，则表的各个字段就有多个值。移动记录指针的位置到所需记录，就可以找出各字段变量的当前值。

2. 内存变量

内存变量是表结构之外独立存在于内存中的变量，一般随程序运行结束或退出 FoxPro 而释放。内存变量常用于存储程序运行的中间结果或用于存储控制程序执行的各种参数。FoxPro 定义 6 种类型内存变量，即字符型内存变量、数值型、逻辑型、日期型、日期时间型和屏幕型内存变量。对屏幕型内存变量，可用 SAVE SCREEN TO <MemVarName>命令存放当前屏幕上的信息，用 RESTORE SCREEN FROM <MemVarName>命令从屏幕内存变量恢复屏幕信息。Visual FoxPro 最多允许定义 65 000 个内存变量。此外，FoxPro 还提供一维和二维内存变量数组。

3. 系统内存变量

系统内存变量是 Visual FoxPro 自动生成和维护的变量，用于控制 FoxPro 的输出和显示信息的格式。为与一般内存变量相区别，在系统内存变量名前加一下划线“_”。例如，系统内存变量_PEJECT 用于设置打印输出时的走纸方式。系统默认值是“BEFORE”，即打印前走纸换页。用户可将其设置为“NON”，即打印前后都不换页走纸。

2.2.3 内存变量赋值命令

Visual FoxPro 提供多种命令定义内存变量和给内存变量赋值，常用的赋值命令格式如下所示：

STORE <表达式> TO <内存变量表>｜<变量数组>

和

<内存变量>｜<变量数组>=<表达式>

这两条命令都用于定义内存变量并给变量赋值，不同之处是前一个命令可定义多个内存变量并赋以同值，而后一个命令却只能定义单个内存变量或变量数组。

例 2-1　定义常用类型的内存变量并将其赋值。

在 Visual FoxPro 命令窗口中键入如下命令：

```
store 12.34 to a
store "张志伟" to b
store .t. to c
store {^1975.02.11} to d
dimension e(2,3)
e(1)=12.34
e(2)="张志伟"
e(3)=.t.
```

```
e(4)={^1975.02.11}
list memory like ?
```

屏幕显示执行结果如下：

```
A          N         12.34  (       12.34000000)
B          C  "张志伟"
C          L  .T.
D          D  75.02.11
E          A
  (   1,   1)    N    12.34  (       12.34000000)
  (   1,   2)    C  "张志伟"
  (   1,   3)    L  .T.
  (   2,   1)    D  95.02.11
  (   2,   2)    L  .F.
  (   2,   3)    L  .F.
```

在例 2-1 **中有关** Visual FoxPro **命令简介如下：**

`store 12.34 to a`	**定义内存变量** a**，数值型，其值为** 12.34**。**
`store "张志伟" to b`	**定义内存变量** b**，字符型，其值为张志伟。**
`store .t. to c`	**定义内存变量** c**，逻辑型，其值为真。**
`store {^1975.02.11} to d`	**定义内存变量** d**，日期型，其值为** 75 **年** 2 **月** 11 **日。**
`dimension e(2,3)`	**定义二维变量数组** e **，该数组有二行三列，共** 6 **个元素：** e(1,1)**、** e(1,2)**、** e(1,3)**、** e(2,1)**、** e(2,2)**、** e(2,3)**。这些元素被定义作逻辑型并赋以初值“假”。二维数组元素还可以用单下标相应表示为：**e(1)**、**e(2)**、**e(3)**、**e(4)**、**e(5)**、**e(6)**。**
`e(1)=12.34`	**数组元素** e(1)**，即** e(1,1)**，赋值** 12.34**，其类型已转换为数值型。**
`e(2)="张志伟"`	**数组元素** e(2)**，即** e(1,2)**，赋值张志伟，其类型已转换为字符型。**
`e(3)=.t.`	**数组元素** e(3)**，即** e(1,3)**，赋“真”值，其类型不变，仍为逻辑型。**
`e(4)={^1975.02.11}`	**数组元素** e(4)**，即** e(2,1)**，赋一日期值，其类型已转换为日期型。**
`list memory like ?`	**内存变量显示命令。其中子句“**like ?**”中的问号为通配符代表任一字符。此命令只显示以一字符为名的所有内存变量的属性，否则还将显示系统内存变量的有关属性。**

系统规定：如果内存变量与打开的当前表文件的字段同名，则字段名优先。此时如若使用同名内存变量，则必须加写“M->**”或“**M.**”以限定，即：**M->**内存变量名或** M.**内存变量。其中，限定由** M**、减号和大于号或由** M **和圆点组成。**

2.3 表 达 式

表达式是 Visual FoxPro 命令的重要组成部分，它是由常量、变量、函数用运算符连结而成的有意义的式子。

2.3.1 运算符

根据运算符的不同功能划分为 4 类：算术运算符、关系运算符、 逻辑运算符和字符串连接运算符，下面做详细介绍。

(1) 算术运算符。算术运算符包括以下几种：

()	分组优先运算符
**、^	乘幂运算符
*、/	乘除运算符
%	求余运算符
+、-	加减或单目运算符

算术运算符主要用于数值数据间的算术运算，其运算结果也是数值型数据。运算符的优先级排序为：括号、单目、乘幂、乘除、求余、加减。其中括号优先级最高，加减运算符优先级最低。所谓单目运算符，即运算符只在其后面有数据的运算符。

例如，用户在命令窗口中键入显示运算结果的命令：

```
? -2**2+5
```

其显示在屏幕上的结果为：9.00 而不是 1.00。该命令的操作符是“?”，它命令计算机计算和显示表达式的值。例如：

```
? 3*3%2^3+1
```

其结果为 2。

(2) 关系运算符。关系运算符包括以下几种：

<	小于
>	大于
=	相等比较
==	字符串全等比较
<=	小于或等于
>=	大于或等于
<>、#、!=	不等于
$	包含运算

关系运算符用于比较运算，即比较同类型表达式的值。如果其值使关系成立，则比较运算的结果取逻辑值“真”，否则取逻辑值“假”。各关系运算符的优先级相同。

例 2-2 将变量赋值并进行比较运算，其命令及命令执行结果如下所示：

需要全本请访问 www.ertongbook.

命令及执行结果	说明
a=12.34	定义数值型变量
b=23.45	定义数值型变量
? a<b	数值比较
.T.	关系成立取真值
? a>b	数值比较
.F.	关系不成立取假值
x="a"	定义字符型变量
y="b"	定义字符型变量
? x>y	比较 a 和 b 字符的 ASCII 码
.F.	a 的 ASCII 码是 97，b 的 ASCII 码是 98，关系不成立取逻辑值假
? "bc"$"abcd"	比较$左边字串是否是其右边字串的子串
.T.	是其子串，取逻辑值真
? "ac"$"abcd"	比较$左边字串是否是其右边字串的子串
.F.	不是其子串，取逻辑值假
? "abcd"="abc"	比较等号右边字串是否是其左边字串的子串
.T.	是其子串，取逻辑值真
? "abc"="abcd"	比较等号右边字串是否是其左边字串的子串
.F.	不是其子串，取逻辑值假
? "abcd"=="abc"	字串等长并完全相同比较
.F.	不等长取逻辑值假

(3) 逻辑运算符。

逻辑运算符常用作表达式之间的逻辑运算。参加运算的表达式应有逻辑值，其运算结果也是逻辑值。常用的逻辑运算符有如下 3 种：

.NOT.、！ 逻辑非

.AND. 逻辑与

.OR. 逻辑或

逻辑运算符的意义是：

A.AND.B 只当 A 和 B 都为真时，逻辑运算的结果才取真值，否则取假值。

A.OR.B 当 A 和 B 中至少有一个取真值时，逻辑运算结果取真值。

.NOT.A 当 A 取真值时，结果取假值，反之结果取真值。

3 种逻辑运算的真假值如表 2-1 所示。

表 2-1 逻辑运算真值表

A	.T.	.T.	.F.	.F.
B	.T.	.F.	.T.	.F.
A.AND.B	.T.	.F.	.F.	.F.
A.OR.B	.T.	.T.	.T.	.F.
.NOT.A	.F.	.F.	.T.	.T.

逻辑运算符的优先级由高到低的次序为：逻辑非、逻辑与、逻辑或。例如：

```
xb=.f.
a=0
b=5
? a<=b.and..not.xb
.T.
```

其逻辑运算结果取真值。

(4) 字符串连接运算符。

字符串连接运算符用于字符串连接操作，有如下两种：

```
+
-
```

其中，“+”运算符用作将加号前后的字符串连接起来组成一个新字符串，这种连接又称作完全连接；而“-”号运算符，用作将减号前面字符串中的尾空格移到减号后面字符串的尾部，然后相连接组成一个新的字符串。

例 2-3　字符串连接。

```
a="abc   "                &&abc 后面有两个空格
b="def   "                &&def 后面有两个空格
c="ghi"
? a+b+c
abc   def   ghi           完全连接
x=a-b-c
? x
abcdefghi                 空格移到连接后生成串的尾部
```

在以上各类运算符之间的优先级是：算术运算符和字符串连接运算符的优先级最高，关系运算符次之，逻辑运算符最低，同级运算符按从左至右顺序执行。

2.3.2　表达式

表达式是 FoxPro 命令和函数的重要组成部分，它通常由常量、变量、函数通过运算符连接而成。表达式通过运算得出表达式的值。Visual FoxPro 命令和函数对表达式的类型有明确要求。不同类型的表达式，要求有相应类型的常量、变量、函数和运算符。一定要严格遵守相应的规定，否则将出现语法错误以至计算机拒绝执行。Visual FoxPro 定义如下 4 种表达式：

(1) 数值型表达式。

它由数值型常量、变量、函数及算术运算符组成。表达式的值也为数值型数据，例如：

```
178*3+(56-12)/6
```

(2) 字符型表达式。

字符型表达式由字符型常量、字符型变量、字符型函数及字符串连接运算符构成。其运算结果也为字符型数据，例如：

```
'姓名="何志秀"'+'.and.'+'出生日期={^1965.01.05}'
```

其运算结果的字符串如下所示:

```
姓名="张志伟".and.出生日期={^1965.01.05}
```

(3) 逻辑型表达式。

逻辑型表达式由逻辑型常量、逻辑型变量、逻辑型函数、关系式用逻辑运算符连接而成。通过运算，逻辑表达式取逻辑值真或假。例如，下列表达式:

```
籍贯="北京市".and..not.性别
.t.
.not.eof()
```

都是正确写法的逻辑表达式。其中，第一个逻辑表达式用作选取籍贯是北京市，并且是女性记录的条件；第二个逻辑表达式只包含一个逻辑常量，逻辑表达式永取真值；在第三个逻辑表达式中的 eof()为文件记录指针测试函数，当指针指向文件尾时返回真值，否则返回假值。所以表达式.not.eof()当指针停在非文件尾时取真值，常用作控制循环结束条件。

还有一点需要说明，就是在逻辑表达式定义中所提到的关系式，有些著作中称它为关系表达式。它由数值型、字符型或日期型常量、变量、函数及其表达式，通过关系运算符连接而成。其运算结果取逻辑值，即当关系成立时取真值，否则取假值，例如:

```
姓名= "何志秀"
基本工资>=630
```

都是正确写法的关系表达式。其实关系表达式，是一种没有逻辑运算符的逻辑表达式特例。

(4) 日期表达式。

日期表达式，由日期型常量、日期型变量和日期型函数通过“+、-”算术运算符构成。

例如:

```
{^1995.4.13}+30
{^1995.4.13}-30
```

都是正确写法的日期表达式，其结果分别为：95.05.13 和 95.03.14，但下面的表达式却是数值型的:

```
{^1995.4.13}-{^1995.4.7}
```

结果为 6。其实也不难理解，因为表达式的值为日期间隔天数，亦非日期量。

2.4 常 用 函 数

Visual FoxPro 定义的函数十分丰富，共有 200 余个。灵活运用这些函数，不仅可以简化许多运算，而且能够加强 Visual FoxPro 的许多功能。在学习和使用这些函数时需要注意下面 3 点:

(1) 准确地掌握函数功能。

(2) 函数的返回值有确定的类型，因而在组成表达式时特别要注意类型匹配。

(3) 函数对其参数的类型也有确定要求，否则将产生类型不匹配语法错误。

函数既可按其返回值类型分类，也可以按其功能分类。按返回值类型可将函数划分为：数值型、字符型、日期型和逻辑型四类函数。而按函数功能可将函数划分为：字符处理函数、数学运算函数、转换函数、日期函数、测试函数、环境函数、键处理函数、数组函数、窗口函数、菜单函数和其他类型共 11 种函数。下面就按功能分类介绍常用函数。

2.4.1 字符及字符串处理函数

字符及字串处理函数的处理对象均为字符型数据，但其返回值类型却各异。

1. 取子串函数

格式：SUBSTR(cExpression,nStartPosition[,nCharactersReturned])

说明:

该函数用于选取字符串表达式或备注型字段的部分字符。

cExpression 指定字符表达式或备注字段。

nStartPosition 指定取子串起始位置。

nCharactersReturned 指定取子串的字节数。该参数的默认设置指从起始位置取到串尾。很明显函数的返回值是字符型。

例 2-4 取字符串中三个子串，备作行号用。

```
stor "1995.03.12" to cRQ
? subs(cRQ,1,4), subs(cRQ,6,2), subs(cRQ,9,2)
```

该例的结果如下所示：

```
1995 03 12
```

2. 删除空格函数

有如下三个函数可以删除字符数据中的多余空格，其格式如下所示：

```
TRIM(cExpression)
ALLTRIM(cExpression)
LTRIM(cExpression)
```

说明:

cExpression 指定拟删除多余空格的字符表达式。

第一个函数的功能是删除字符表达式值的尾部空格；第二个函数的功能是删除字符表达式值的前后空格；最后一个函数的功能则是删除字符表达式值前面的空格。显然，函数的返回值是字符型。

例 2-5 去掉第一个字符串的尾空格后与第二个字符串连接。

```
Stor "abcd    " to x
Stor "efg" to y
```

```
? trim(x)+y
abcdefg
```

3. **空格函数**

格式： SPACE(nSpaces)

说明：

nSpaces　**指定空格数。**

该函数的功能是产生指定长度的空格字符串。函数的返回值为字符型。

例 2-6　定义 xm(**姓名**)**变量，其初值赋予** 8 **个空格。**

```
stor spac(8) to xm
```

2.4.2　数学运算函数

Visual FoxPro **提供了** 20 **余种数学运算函数，极大地增强了其数学运算功能。数学函数的返回值皆为数值型。下面介绍几种常用的数学函数。**

1. **取整函数**

格式： INT(nExpression)

说明：

nExpression　**指定取整表达式。**

函数的功能是取指定数值表达式计算结果的整数部分。

例 2-7　取整并显示其结果。

```
? int(-14.74)
  -14
```

2. **四舍五入函数**

格式： ROUND(nExpression,nDecimalPlaces)

说明：

nExpression　**指定拟做四舍五入的数值表达式。**

nDecimalPlaces　**指定四舍五入的小数位数。**

函数的功能是：依据给出的四舍五入小数位数，对数值表达式的计算结果做四舍五入处理。

例 2-8　对下面给出的数做四舍五入处理并显示其结果。

```
? round(3.14159,4),round(2048.9962,0),round(2048.9962,-3)
      3.1416      2049      2000
```

3. **金融函数**

金融函数用于计算到期的本息和、投资现值及每次应偿还的贷款金额数。

(1) **计算每次偿还贷款数函数。**

格式： PAYMENT(nPrinciple,nInterestRate,nPaments)

说明:

nPrinciple **指定贷款本金。**

nInterestRate **指定年利率。**

nPaments **指定贷款总期数。**

该函数用于计算固定利率贷款每期需缴纳的贷款金额数。

(2) **计算投资现值函数。**

格式： PV(nPament,nInterestRate,nTotalPaments)

说明:

nPament **指定每期付款金额。**

nInterestRate **指定年利率。**

nTotalPaments **指定总期数。**

该函数用于计算投资金额的现有值。

(3) **计算本息和函数。**

格式： FV(nPament,nInterestRate,nPeriods)

说明:

nPament **指定固定期数的付款金额。**

nInterestRate **指定年利率。**

nPeriods **指定固定期数值。**

该函数用于计算投资金额的未来本利和。

例 2-9 年利率为 11%，**借期一年，总贷款数是** 24000 **元，每月应付款是多少**?

```
? payment(24000,0.11/12,12)
    2121．16
```

2.4.3 转换函数

转换函数常用于转换某些数据的类型。

1. **数值转数字字串函数**

格式： STR(nExpression,[nLength[,nDecimalPlaces]])

说明:

nExpression **指定拟转换的数值表达式。**

nLength **指定转换后的宽度，包括小数位和小数点。**

nDecimalPlaces **指定小数位数。**

该函数用于将数值转换为数字字串。函数的返回值当然为字符型数据。

例 2-10 将数值型数据转换为字符型数据并显示其转换结果。

需要全本请访问 www.ertongbook.