第一章　dBASE 数据库系统

1.1　dBASE 系统的发展

60年代后期，随着计算机技术的发展与更新，随着计算机应用领域的扩大，人们对计算机管理提出了新的要求：要求管理的范围不仅限于某个局部，而是整体管理，如管理一个部门或一个单位的全部数据；管理的数据量也急剧增加；管理的限制也增多，例如要求数据的保密，数据的一致，数据使用的权限等等。因此，计算机技术的一个新的分支——数据库技术随之产生和发展。70年代初，关系数据库的产生与发展使数据库技术发展到一个新的阶段。关系数据库简单易学，为数据库的广泛应用奠定了技术基础。70年代后期，微型计算机日益普及，几乎渗透到了各个领域。如果使微型计算机配有关系数据库，那么它对计算机应用与普及的影响是极为可观的。1982年美国Ashton-Tate公司研制并推出了适用于微型计算机的关系数据库系统dBASE Ⅱ。尽管dBASE Ⅱ还相当粗糙，功能上有很多限制，有许多不尽人意的地方，但它却大大地拓宽了微型计算机的应用领域。尤其在我国，有不少单位与部门使用了dBASE Ⅱ，进行了某些管理与应用。但是，由于 dBASE Ⅱ 本身的一些固有的缺点，例如运行速度慢，数值精度低，文件字段数的限制，可同时使用的数据库文件数仅限于2个，这些都远远不能满足多方面使用的要求，限制了dBASE Ⅱ进一步广泛的应用与普及。

dBASE Ⅲ是在dBASE Ⅱ应用的基础上，充分了解与发现dBASE Ⅱ在应用中的不足，对dBASE Ⅱ作了较大改进而开发出来的dBASE第二代产品。它充分地保留并发展了dBASE Ⅱ简单易学等优点，同时增加了许多新的命令和函数，扩充了系统的功能和使用范围，弥补了dBASE Ⅱ功能上的不足，并着重解决了运行速度慢、同时允许打开的数据库文件少、数据库文件字段少、数据精度低等问题。从而使dBASE Ⅲ比dBASE Ⅱ更完善，功能更强，运行速度更快。

尽管 dBASE Ⅲ解决了 dBASE Ⅱ 的许多问题，但是，由于它还是作为在单个计算机上运行的系统，在使用上仍然受着相当的限制，从而限制了 dBASE 系统更广泛地普及与应用。因此，1986年Ashton-Tate 公司推出了 dBASE Ⅲ PLUS 系统。dBASE Ⅲ PLUS在 dBASE Ⅲ 的基础上又增加了许多新的命令与函数，扩充了 dBASE Ⅲ 的功能，并着重解决了数据共享问题。dBASE Ⅲ PLUS可以运行在局部网络系统上，使多个用户能同时访问服务器共享盘中的数据，并提供相应的安全保密措施。此外，dBASE Ⅲ PLUS还可对程序进行伪编译加密，加快程序运行速度，防止程序遭到破坏与窃取。因此，dBASE Ⅲ PLUS 比 dBASE Ⅲ 更完善，功能更强，适应性更广，实用性更强。

1988年10月Ashton-Tate公司推出了dBASE Ⅳ系统。dBASE Ⅳ是 dBASE 系统的最近版本，是dBASE系统的第四代产品。dBASE Ⅳ总结了dBASE 系统多年应用的经验，对 dBASE 系统进行了全面分析，广泛地引进了其它关系数据库系统的优点。它是在dBASE Ⅲ PLUS的基础上进行研制的。dBASE Ⅳ 不但对 dBASE Ⅲ PLUS 已有的命令进行了扩充，以增强其功

能，还增加了许多新的命令、函数以及数组、自定义函数等。它还引进了控制中心（Control Center），以菜单驱动方式操作数据，菜单形式灵活多样，非常适用于dBASE的初学者。尤其是dBASE Ⅳ引进了SQL语言。SQL语言在关系数据库的发展中具有举足轻重的地位，它有许多优点，它已经广泛地被计算机系统作为关系数据库系统的用户接口。SQL语言的引进无疑对增强其系统功能大有益处。

1.2 dBASE命令的使用方法

dBASE Ⅳ 的命令具有两种工作方式，一种是单命令工作方式，也称立即执行方式；另一种是批命令工作方式，也称程序执行方式。

1.2.1 立即执行方式

立即执行方式是在dBASE Ⅳ圆点提示符下输入dBASE命令，然后按回车（↵）键，通知计算机，命令立即被执行。例如，在当前磁盘目录下，显示数据库文件清单，在dBASE Ⅳ圆点提示符下输入

```
.DIR↵
```

执行的结果，dBASE Ⅳ显示数据库文件的清单；如果在当前磁盘目录中没有数据库文件，则显示信息“None”。

按回车键之前，可以编辑已输入的命令。可以使用向左（←）和向右（→）键移动光标，使用插入（Ins）键和删除（Del）键插入和删除字符，还可以按Esc键撤消或删除整个命令。

在圆点提示符下，命令行最多可输入254个字符，如果命令行超过254个字符，可按Ctrl-Home键打开编辑窗口，在编辑窗口中输入命令行。按回车（↵）键，结束命令行的输入，关闭编辑窗口，命令立即被执行。在编辑窗口中，命令行最大可含有1024个字符。

如果输入的命令语法有错或者丢掉了必要的参数项，dBASE Ⅳ将出现提示信息。此提示信息提供三种选择：

1. Cancel：取消执行，并消除命令，返回到圆点提示状态。
2. Edit：返回到圆点提示状态，重新编辑，编辑后按回车（↵）键，再执行该命令。
3. Help：需要dBASE Ⅳ提供帮助信息。

dBASE Ⅳ在圆点提示符下有一个内存缓冲区，它自动保存所输入的命令，这个内存缓冲区称为历史缓冲区。当需要重复执行一条命令时，不须重新输入该命令，而是通过按上移（↑）键和下移（↓）键，显示以前所输入的命令（以相反的次序，每次只显示一条命令。），寻找所需要的命令，按回车键进行执行。在按回车键执行之前，可以利用标准编辑键对历史缓冲区命令进行编辑。

启动dBASE Ⅳ时，历史缓冲区默认最多保存20条输入命令。但可以通过环境设置命令SET HISTORY TO随时改变这个默认值，最大可设置保存16000条输入命令。有关历史缓冲区的设置及管理将在第十一章“调试技术”中详细讨论。

1.2.2 程序执行方式

把要执行的命令组织在一起，命名一个名字，以文件的形式存入磁盘，该磁盘文件称为程序文件（或命令文件）。这个程序文件是用 DO 命令调用执行的。

例如，执行下列一组环境设置命令：

```
. SET TALK OFF
. SET ESCAPE OFF
. SET DEVICE TO PRINTER
. SET AUTOSAVE ON
```

以上每条命令均在圆点提示符下，是采用单命令执行方式进行的，即每键入完一条命令，dBASE Ⅳ就立即执行该命令。若采用程序执行方式，将上述四条命令组织在一起，以文件 ENVI.PRG 的形式保存到磁盘上，则键入程序调用 DO 命令，会得到同样的结果：

```
. TYPE ENVI.PRG
ENVI.PRG 10/17/89
SET TALK OFF
SET ESCAPE ON
SET DEVICE TO PRINTER
SET AUTOSAVE ON
. DO ENVI
```

1.3 dBASE 命令的构成和基本规则

1.3.1 dBASE 命令的构成

先举例说明典型的 dBASE 命令的形式。

假定现有一个学生数据库文件 student，它含有每个学生的姓名和数学成绩的数据，求数学不及格的人数，并打印出他们的姓名和成绩。

解决此问题可用下列一组 dBASE 命令实现：

```
. 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，以备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）由打印机输出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。

在dBASE Ⅳ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，但往往又比语句更精炼，功能更强。例如，如果使用一般的高级语言（如 BASIC、FORTRAN 等语言），则解决上例所提的问题就需要很多的语句，以便用来描述“打开数据

库文件”、“统计”、“打印”和“关闭数据库文件”等操作的详细过程。

从上例可以看出，dBASE Ⅳ的数据库操作命令都是以命令动词开头，后面可以跟一个或多个子句。例如，第二条命令的命令动词为 COUNT，子句为 ALL 和 FOR。

下面列出 dBASE Ⅳ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dBASE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。

例如，上面例子中第二、三条命令的范围均为 ALL。

4. FOR〈条件〉。它告诉 dBASE Ⅳ，命令仅对满足条件的记录进行操作。如果使用 FOR 子句，dBASE Ⅳ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。

在FOR子句和WHILE子句中，〈条件〉必须是逻辑型表达式，也就是说，〈条件〉返回值必须是真（.T.）值或假（.F.）值。例如上面例子中的第二、三条命令，〈条件〉为“数学成绩<60”。当数学成绩低于60分时，条件“数学成绩<60”返回真（.T.）值；当数学成绩高于或等于60分时，条件“数学成绩<60”返回假（.F.）值。

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。但是，应该注意，允许操作结果向内存变量输出的命令必也允许操作结果向一个数组元素输出。

7. ALL [LIKE/EXCEPT〈通配符〉]。它告诉 dBASE Ⅳ，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。在通配符中可使用符号？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。

下面举两个例子，并详细说明每条命令中子句的作用。

例 1

DISPLAY	姓名，职称，工资	NEXT 6	FOR 工资 > 200.00	TO PRINTER
命令动词	表达式表	范围	FOR〈条件〉	[TO…]

命令动词“DISPLAY”指示计算机要完成“显示”数据操作。表达式表“姓名、职称、工资”指示计算机对每条记录“显示”出它的姓名、职称、工资字段数据。范围“NEXT 6”表明从当前记录开始连续对6条记录进行“显示”操作。FOR〈条件〉“FOR工资>200.00”表示对工资字段大于200.00的记录进行“显示”操作。[TO…]“TO PRINTER”控制“显示”的结果在打印机输出。由此可知，上述DISPLAY命令的功能是：打印输出从当前记录开始连续6条记录中工资高于200.00元人的姓名、职称、工资。

例 2

命令动词“SUM”指示计算机要完成“求和”数据操作。表达式表“工资”指示计算机对工资字段进行“求和”操作。FOR〈条件〉“FOR 教研室 = '计算机'”表示对教研室字段为“计算机”的记录进行“求和”操作。〔TO…〕“TO mgz”控制“求和”结果存放在内存变量 mgz 中。由此可知，上述SUM命令的功能是：计算“计算机”教研室所有人的工资总和，将结果存入内存变量 mgz 中保存。

1.3.2 基本规则

dBASE Ⅳ命令有以下规则：

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。如下边三条命令是等价的：

```
LTST 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 教研室 = “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. 在单命令工作方式下，一条命令的最大字符个数可达254个。如果一条命令的长度超过254个字符，可通过按 Ctrl-Home 键打开编辑窗口。在编辑窗口中，命令行的总长度最大可达1024个字符。
5. 命令动词和 dBASE Ⅳ 保留字均可用4个以上字母简写。例如，下边三条命令是等价的：

```
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. dBASE Ⅳ中没有绝对不能使用的单词，但最好不使用dBASE Ⅳ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 dBASE Ⅳ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 dBASE Ⅳ的保留字，作为别名 (ALIAS)。

9. 程序设计时，DO WHILE/ENDDO, DO CASE/ENDCASE, IF/ENDIF, SCAN/ENDSCAN, 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“;”。

1.3.3 符号约定

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。若选择该项目，不要写方括号本身。

〈 〉：角括号，表示括号内的项目是必须要选的，但不要写角括号本身。

/：斜线号，表示两个项目中选择其中一个项目，但不要写斜线号本身。

…：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。

1.4 数据显示命令(?/??)和赋值命令(STORE)

在 dBASEⅣ中，显示命令?/??和赋值命令 STORE 是用的最多的命令。特别是显示命令? /? ?，在程序设计中几乎每个程序中都要用到，应很好掌握。

1.4.1 数据显示命令 ?/? ?

数据显示命令 ? /? ? 可以完成如下操作：

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。

例

```
. STORE 34 TO Mvar
      34
. ? Mvar
      34
```

（2）用单问号? 可以显示表达式的值。

例 1

```
. STORE 4 TO Mvar 1
          4
. STORE 6 TO Mvar 2
          6
. ? (Mvar 1 + Mvar 2 )/ 2
          5
```

例 2

```
. 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 1 显示输出数字常数

```
. ? 4
        4
```

例 2 显示输出字符串常数

```
. ? "THIS IS A STRING"
THIS IS A STRING
```

应当注意，这里的引号是用来确定字符串起止和终止界限的，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。

例 1

```
. STORE "AB" TO Mvar 1
AB
. STORE "CD" TO Mvar 2
CD
. STORE "EF" TO Mvar 3
EF
. ? Mvar 1 ,Mvar 2 ,Mvar 3
AB CD EF
```

例 2

```
. STORE 4 TO Mnum 1
          4
. STORE 3 TO Mnum 2
          3
. ? Mnum 1 + Mnum 2 ,Mnum 1 - Mnum 2
          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。

例

```
. STORE 32 TO Mnum 1
        32
. STORE 12 TO Mnum 2
        12
. ? "Mnum 1 + Mnum 2 = " ,Mnum 1 + Mnum 2
Mnum 1 + Mnum 2 =          44
```

3. 用 ? 命令可以输出一个空行。

若 ? 命令后面没有任何表达式，则表示本行什么也不显示输出，相当于输出一个空行。

例

```
. ? " 3 * 4 = " , 3 * 4
. ?
. ? "3* 5 = " , 3 * 5
```

结果

```
3 * 4 = 12

3 * 5 = 15
```

4. 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。

例

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

结果

```
A B CD E F
G H IJ  K L
M N QP Q R
.
```

1.4.2 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 命令也可用于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。假若内存变量 Mnum 原值为 100，则新值将变为200。星号表示乘号。

（3）一个 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、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
```

完全等价于

```
.STORE 10*Mnum TO Mnum
```

除此之外，STORE 还可用于对数组赋值。有关数组的内容将在第八章中介绍。

1.5 记录、字段和数据值

dBASE Ⅳ是关系数据库管理系统。关系数据库的处理对象是二维表。每一张二维表称为一个关系，而关系中所包含的是一条条记录，构成这些记录的基本单元是字段。字段表示实体某一方面的属性。对于每个具体记录来说，每个字段的具体值也就是该字段的数据值，这个数据值反映了所处理的实体所具有的属性的数据。

例如，学生关系

姓　名	性　别	年　龄
王　军	男	18
李　红	女	17
张　义	男	19

包含有三条记录，每条记录由三个字段（姓名、性别、年龄）构成。

1.5.1 记录

在 dBASE Ⅳ中，一个关系可看作为一个数据库文件，按照记录存储的先后顺序从小到大进行编号，这个编号称为记录号。每个数据库文件最多允许含有10亿条记录，即记录号最小为 1，最大为1,000,000,000。每个数据库文件最多可含有255个字段，记录各个字段总和最多不超过4000个字节。各记录总计允许使用的最多字节个数为20亿个。事实上，最终的限制因素将是计算机系统的磁盘容量。

1.5.2 字段和数据值

字段由字段名、字段类型和字段宽度三部分构成。

字段名必须由字母或汉字开头，可由汉字、字母、数字和下划线组成，最多不超过10个字符。例如，下面是合法的字段名：

```
CLIEN_ID  姓名  CITY  A1
```

下面是非法的字段名：

```
1A  姓␣␣名  ZIP; CODE
```

在 dBASE Ⅳ中，字段共有六种类型：字符型（C型），数字定点型（N型），数字浮点型（F型），日期型（D型），逻辑型（L型），明细型（M型）。

1. 字符型字段的数据值和宽度

字符型字段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。宽度为该字段中最大的字符数，系统允许最大宽度为254个字符。

2. 数字定点型字段的数据值和宽度

数字定点型字段的数据值是由正负号、数字0至9、小数点构成的，可以进行算术运算。它是以BCD (Binary Coded Decimal) 码形式存放的，用字母N表示。宽度包括数字宽度和小数宽度两部分。数字宽度为该字段最大整数位与最大小数位之和，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。系统允许最大数字宽度为20个数位。

3. 数字浮点型字段的数据值和宽度

数字浮点型字段的数据值是由正负号、数字0至9、小数点构成的，可以进行算术运算。它是以浮点二进制数形式存放的，与dBASE Ⅲ PLUS的存放形式完全相同，用字母F表示。宽度包括数字宽度和小数宽度两部分。数字宽度为该字段最大整数位与最大小数位之和，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。系统允许最大数字宽度为20个数位。

数字N型和F型都是用于存放数字型数据的，两者显示的形式完全一样，主要区别于dBASE Ⅳ内部的存放形式。N型采用BCD码形式存放数值，F型采用浮点二进制数形式存放数值；N型反映的数值比较精确，F型反映的数值是一个近似值；F型适用于描述很大或很小的数值，多用于科学计算，而N型主要用于商业和财务计算。

4. 日期型字段的数据值和宽度

日期型字段是用于存放日期型数据的，用字母D表示。通常格式是“月/日/年”，其中月、日、年均为两位数字。字段宽度固定为8个字节。例如，“09/05/88”表示1988年9月5日。

5. 逻辑型字段的数据值和宽度

逻辑型字段的数据值是由逻辑真（.T.）和假（.F.）值所构成的。用字母L表示。字段宽度固定为1个字节。在输入时，逻辑真（.T.）值也可用t、Y和y来表示，逻辑假（.F.）值也可用f、N和n来表示。

6. 明细型字段的数据值和宽度

明细型字段是用于存放大块的ASCII码数据的。数据被存放在数据库的辅助文件(.dbt)中，在数据库文件中用memo一词指明，用字母M表示。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，最大限制与所使用的编辑程序或字处理程序有关。

1.6 常数、变量和函数

1.6.1 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（N，F）型、日期型、逻辑型。

一、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。它有三种定界符：

（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，则应该选择另外一

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" ghl]

二、数字（N，F）型常数

数字型常数是由正负号、0至9、小数点构成的。默认情况下是N型，可以使用FLOAT（ ）函数将其转换成F型。例如，下面是合法的N型常数：

12.35，-150，0.000025，140000

下面是合法的F型常数：

FLOAT(0.000025)，FLOAT(140000)，FLOAT(-0.00123427)

三、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")

四、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"

1.6.2 变量

变量在命令和程序运行过程中其内容是可能要发生改变的。dBASE Ⅳ有三种类型变量：字段、内存变量和系统内存变量。

字段是构成数据库文件的基本单元，这已经在第五节详细讨论过。

内存变量存在于内存之中，它独立于数据库文件。它的设置为用户使用数据库系统带来极大的方便。内存变量可以用来保留中间结果，或保留对数据库进行某种分析处理后得到的数据结果，内存变量也经常用于控制程序流程。

内存变量是一种临时工作单元，需要用时可以随时定义，不用时又可以释放，也可以在磁盘上永久保存。

一、内存变量名

内存变量名必须以一个字母或汉字开头，由字母、数字、汉字和下划线组成，中间不允许含有空格，最长不超过10个字符。

应当注意，内存变量名一般不要与 dBASE Ⅳ命令动词、保留字同名，以免造成系统识别上的混淆。

二、内存变量生成

任何时候为内存变量赋值的过程，也可以看成是内存变量生成的过程。也就是说，当为某一内存变量赋值时，dBASE Ⅳ系统首先查看内存中该内存变量是否已经存在，如果不存在，则生成一个新的内存变量，并赋值。

最常用的生成内存变量命令是 STORE 命令和〈内存变量〉=〈表达式〉命令。

三、内存变量类型

内存变量有五种类型：字符（C）型、数字N型、数字F型、日期（D）型、逻辑（L）型。

字符型：字符型内存变量最多可含有254个字符。占用内存总共是内存变量的实际长度加2个字节。

数字 N 型：N型内存变量最大数值为$0.9\times E+308$，最小数值为$0.1\times E-307$。

数字 F 型：F型内存变量的精度达到15.9位数字（不计小数点），最大数值为$0.9\times E+308$，最小数值为$0.1\times E-307$。

日期型：日期型内存变量的宽度总是8个字节，但在内存中需要占用9个字节。通常采用美国格式：月/日/年。

逻辑型：逻辑型内存变量的宽度总是一个字节，但在内存中需要占用2个字节。接受T、t、Y、y为真，F、f、N、n为假。

四、内存变量限制

在默认情况下，内存中的内存变量同时最多只能有500个。

五、内存变量的使用

当引用的内存变量与数据库文件中的字段同名时，内存变量名前应加前缀M—>。例如

```
.?M—>Name
```

表明显示输出内存变量名Name的内容。

当内存变量不需要的时候，可以随时用 RELEASE 命令从内存中清除，释放占用的空间。例如，当上例的内存变量 Name 不需要时，可用下面的命令清除：

```
.RELEASE Name
```

内存变量也可用 SAVE 命令以内存变量文件形式保存在磁盘上，需要时用 RESTORE 命令将磁盘上的内存变量调回到内存中。例如，命令

```
.SAVE TO Smmm
```

将当前内存变量全部保存在磁盘内存变量文件 Smmm 中。而命令

```
.RESTORE FROM Smmm
```

将磁盘内存变量文件 Smmm 中的内存变量调回到内存中。

系统内存变量是dBASE Ⅳ自动建立和维护的内存变量，用户是不能建立和清除的，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。有关系统内存变量的使用将在第十章详细讨论。

1.6.3 函数

dBASE Ⅳ提供了128种函数。从概念上讲，dBASE Ⅳ函数与数学中的函数没有什么根本区别，但需要注意四点：

1. 所有函数必须跟随有圆括号，无论该函数是否需要参数。
2. 每个函数必须有一个返回值。
3. 返回值具有一定数据类型。根据返回值的数据类型可以将函数分为：数字N型函数、数字F型函数、字符型函数、日期型函数和逻辑型函数。

4. 传送给函数的参数也有一定的数据类型，一定要按照要求的数据类型传送参数值。下面列出三角函数、数学函数和它们的功能。

函　　数	功　　能
SIN (<数字型表达式>)	求正弦
COS (<数字型表达式>)	求余弦
TAN (<数字型表达式>)	求正切
ASIN (<数字型表达式>)	求反正弦
ACOS (<数字型表达式>)	求反余弦
ATAN (<数字型表达式>)	求反正切
ATN2(<数字型表达式1>, <数字型表达式2>)	求反正切
ABS (<数字型表达式>)	求绝对值
SIGN (<数字型表达式>)	求正负号
INT (<数字型表达式>)	取整
PI( )	求F型的π值
LOG (<数字型表达式>)	求以e为底的自然对数
LOG10 (<数字型表达式>)	求以10为底的对数
SQRT (<数字型表达式>)	求平方根
CEILING (<数字型表达式>)	求大于或等于指定值的最小整数
FLOOR (<数字型表达式>)	求小于或等于指定值的最大整数
MAX(<数字型表达式1>/<日期型表达式1>, <数字型表达式2>/<日期型表达式2>)	求两者中较大者
MIN (<数字型表达式1>/<日期型表达式1>, <数字型表达式2>/<日期型表达式2>)	求两者中较小者
EXP (<数字型表达式>)	求自然指数
FIXED (<数字型表达式>)	F型转换成N型
FLOAD (<数字型表达式>)	N型转换成F型
ROUND (<数字型表达式1>, <数字型表达式2>)	四舍五入
RAND ([<数字型表达式>])	随机函数
MOD (<数字型表达式1>, <数字型表达式2>)	求模

1.7 运算符、运算规则和表达式

dBASE Ⅳ 提供四种类型的运算符：算术、比较、逻辑和字符串运算符。根据运算性质可以划分成四种运算：算术、比较、逻辑和字符串运算。

1.7.1 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 6 种运算符：

加号　+　　除号　/

减号　-　　乘方　**或^

乘号　*　　括号　( )

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（ ），可以在圆括号（ ）内套用圆括号（ ）。

说明：对于日期型数据，可以使用加（+）、减（-）运算符进行如下形式的运算，产生数值或日期结果：

1. 一个表示天数的数值可与日期型数据相加，产生日期结果。

例

```
. STORE {05/24/88} TO Ma
05/24/88
. STORE 10 TO Mnum
        10
. ? Ma + Mnum
06/03/88
. ? Ma + 101
09/02/88
. ? {10/20/88} + 30
11/19/88
```

2. 一个表示天数的数值可与日期型数据相减，产生日期结果。

例

```
. STORE {05/24/88} TO Ma
05/24/88
. STORE 10 TO Mnum
        10
. ? Ma - Mnum
05/14/88
. ? Ma - 101
02/13/88
. ?  {10/20/88} - 30
09/20/88
```

3. 两个日期型数据可以相减，产生数值结果。该数值为两个日期相差的天数。

例

```
. STORE {10/24/88} TO Ma
10/24/88
. STORE {10/10/88} TO Mb
10/10/88
. ? Ma - Mb
          14
. STORE {08/24/88} TO Ma
08/24/88
. STORE {05/07/88} TO Mb
05/07/88
```

```
. ? Ma - Mb
        109
. ? Mb - Ma
        - 109
. STORE {08/24/88} TO Ma
08/24/88
. STORE {05/07/87} TO Mb
05/07/87
. ? Ma - Mb
        475
```

1.7.2 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 7 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 #		

说明：

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的数据类型必须相同。例如，下列是合法的比较运算：

```
. ? 'This' >= 'That'
.T.
. ? 4366<= 7624
.T.
. STORE {11/13/88} TO Ma
11/13/88
. STORE {10/07/88} TO Mb
10/07/88
. ? Ma>Mb
.T.
. ? Ma<= Mb
.F.
```

下列是不合法的比较运算：

```
.? '123' <  >123
.?  {09/27/88)<  >'09/27/88' }
.STORE '24.6' TO Ma
24.6
.? Ma>100
```

2. 字符型数据按 ASCII 码值的顺序进行比较，数字型数据按数值大小进行比较，日期型数据按年、月、日的先后进行比较。例如

```
. ? 123<=137
.T.
. ? "ab" >"AB"
.T.
. ? {02/21/89}<={01/08/89}
.F.
. ? " CF" >=" GFS"
.F.
```

3. 子字符串比较 $。如果 A 和 B 是字符串，而且 A 与 B 相同或 A 是 B 的子字符串，则 A$B 的结果是逻辑真值。例如

```
. ? "this" $"this is a string。"
.T.
. ? "this" $" THIS IS A STRING."
.F.
. ? "this is a string" $"this"
.F.
. ? "this is a string" $"this is a string"
.T.
```

1.7.3 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）结果。它包括 4 种运算符：

逻辑与　.AND.　逻辑非　.NOT.

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（ ）。

各种逻辑运算符的运算结果归纳如下：

逻辑与 .AND.

逻辑型表达式 A	逻辑型表达式 B	A.AND.B
T	T	T
T	F	F
F	T	F
F	F	F