dBASE III

汇编语言

关系数据库管理系统

上海电子计算机厂

目　　录

附　　录

1.0 使用数据库

为了执行数据库程序，将有 dBASE的磁盘(最好复制一块这样的磁盘)放到任何一个可用的磁盘驱动器中。置该驱动器为默认驱动器（例：若磁盘被放到"B"驱动器，打入"B:"跟一回车）然后键入如下的内容：

DBASE

程序被装入内存，并且以请求输入日期为执行的开始：

ENTER DATE AS MM/DD/YY OR RETURN FOR NONE:

这个日期将记入到在这次运行期间被修改的任何一个数据库中，同时也将打印在该次运行中产生的任一个 REPORT 的报表标题中。对日期进行核对，以确定其准确性。警告，对于1900年与2100年的 2 月29日，使用日历检查是无效的。斜线或任何专用字符（句号除外）可以用来分界数字。

有效的日期例子：

1,1,81

02 01 82

3/17/83

然后屏幕显示信息如下：

*** dBASE II VER 2.xx ***

.

第二行的句点是dBASE提示，表明 dBASE准备接收命令。dBASE命令通常是命令式句子：动词后可跟一个短语，该短语更进一步指明命令功能。执行任何一部分命令之前，dBASE扫描整个命令行。若 dBASE在命令中查出错误，则通过控制台告诉用户错误信息。一般地，用户可修改错误命令或重新输入整个命令。当 dBASE发现一个明显地不能描述的错误时，就假定是句法错，则显示错误行，并在引起错误的短语前显示问号。

错误修改实例：

```
. DISPRAY MEMORY
*** UNKNOWN COMMAND
DISPRAY MEMORY                  列出错误命令
CORRECT AND RETRY? Y            是的，要修改
CHANGE FROM  :PR                修改字母PR
CHANGE TO    :PL                改为PL
DISPLAY MEMORY                  修改后的命令行
MORE CORRECTIONS? (cr)          回车=不再修改
. STORE (2+2 TO X
*** SYNTAX ERROR ***
      ?                         指出串（2+2有错
STORE (2+2 TO X
```

```
CORRECT AND RETRY? Y
CHANGE FROM  :+2
CHANGE TO  :+2)
STORE  (2+2)  TO X
MORE CORRECTIONS? N          不再修改
   4                         结果
. SUM TO X
NO EXPRESSION TO SUM         说明错误原因
SUM TO X
CORRECT AND RETRY? N         不修改，终止该命令
```

也可用下列方法执行程序：

DBASE 〈文件名〉

这将装载 dBASE 到内存中，处理〈文件名〉的命令文件，并开始直接执行该命令文件。当在 SUBMIT 文件中使用 dBASE，或当使用dBASE的QUIT命令的链接可选项时，这种形式是特别有用的。

控制字符

ctl-P -联打印机开关（见SET PRINT命令）

ctl-U -删当前行

ctl-X -删当前行（全屏幕编辑除外）

Rubout -删最后输入的字符

ctl-H （或退格键） -删最后输入的字符

ESC -终止某个运行时间较长的命令。例如：DISPLAY，COUNT，DELETE，INPUT，LIST，LOCATE，RECALL，REPLACE，SKIP，和 SUM。ESC也可终止象 ACCEPT，INPUT，REPORT(对话)，以及 WAIT 等命令。在任何情况下，ESC能返回到人机对话控制，并显示圆点提示。

当命令文件在运行过程中，且当每一个命令行执行前，dBASE检查有无ESC字符。（该字符不显示）注：ESC的功能可用SET ESCAPE OFF 命令取消。

2.0 系统要求

为使dBASE能正常运行，系统必须具备以下特点：

a）以8080或 z-80为基础的微处理器系统；

b）包括 CP/M的48k（或更多）字节的内存。（dBASE 最多用到 A400H)。注：在某些机器中，包括 Apple，Heath，及Northstar，需要大于48k的存贮，因为它们有一个过大的 CP/M模块；

c）CP/M操作系统（1.4或2.x版）；

d）在CP/M下使用一个或多个大容量存贮设备。（通常是软盘或硬盘驱动器）；

e) 若使用全屏幕操作，则需要光标可寻址的CRT。（最好是24行乘80列的CRT）；

f) 可任意选择的文本打印机（为某些命令）。

3.0 文　件

基本上，一个文件是存在存贮设备中的信息的集合，该设备含有用户的数据。信息可从文件中存取。文件可分为六个类型，每一种类型都与特定的操作有关，或由 dBASE 建立。

所有的dBASE文件是标准的CP/M文件，其名最多为 8 个字符，文件类型为 3 个字符，以下是 dBASE 所用的默认文件类型。对于存取文件的各个命令，文件类型可以不用提供，由 dBASE 为该命令假定默认的类型。例如：若一文件已用 .DBF作为它的文件类型，则在任何文件操作命令中不需指明。

数据库文件　－.DBF
存贮器文件　－.MEM
命令文件　　－.CMD
报表格式文件－.FRM
文本输出文件－.TXT
索引文件　　－.NDX
格式文件　　－.FMT

任何合法的CP/M文件名都可被dBASE文件所用。请记住，若在任何文件的存取期间，用户没有提供文件类型，dBASE 将假定是上面这些文件类型。对于更进一步与文件名，类型有关的信息，请参考Digital Research 公司出版的“CP/M用户指南”。

3.1 数据库文件(.DBF)

dBASE 的一切都离不开数据库。dBASE 的数据库文件由结构记录及 0～65535 个数据记录组成。结构记录实际上是数据记录格式图。结构最多含有32个不同项目。结构中每一项称为数据记录的一个数据场。结构包括下列数据：

* 数据场名
* 数据场中的数据类型
* 数据场的大小
* 记录中数据的位置

数据场名－　名字最多10个字符。在dBASE 运行期间的所有操作将用该名引用该数据场。场名必须是字母数字型（可有冒号）。必须以字母开始，冒号必须嵌入在名中。见下例。

数据场名的例子：

A
A123456789
ABC:DEF

A:B:C:D:E

ABCD:　　无效，冒号未嵌入

ABC, DEF　　无效，逗号是非法的

数据类型－　dBASE对于指明数据场的内容允许使用三个数据类型。他们是：字符串('ABCD')，数字（2或5.18），及逻辑型（真/假）。

场的大小－　这是要放入场中的所需数据的字符位数（宽度）。字符串场和数字场可以是1～254个字符长。数字场的小数点也占1位。逻辑场总是占1位。而且，对于数字场，小数点右边的位数也可以包含在结构中。

一旦确定了结构，用户就能往场中输入数据，送入记录的多少，由需要所定。通常，在任何时候仅有一个结构数据文件给用户使用（这称之为USE文件或文件在USE中）。然而，有一种方法可以一次使用两个数据库。见SELECT和JOIN命令。

3.2 存贮器文件(.MEM)

存贮器文件是静态的存贮器文件，该存贮器被划分为与记录变量相似的变量。这些变量被认为是存贮器变量且限制在64个以内。

存贮器变量的值与打开的数据库无关。也就是说，USE文件的记录位置与在存贮器文件中的变量无关。被使用的存贮器变量含有常量、计算结果及符号代替串(看第5部分)等等。存贮器变量的命名，类型的指定，大小的规则与上面所描述的场变量相同。

SAVE 命令将把当前所有存贮器变量写到存贮器文件中；RESTORE 命令可将存贮器文件读到内存中。

3.3命令文件(.CMD)

命令文件含有一系列的 dBASE 命令语句。这给用户提供了存贮一组使用频繁的命令序列的方法，并可方便地处理一个或多个数据库文件。

尽管dBASE 现在有能力用MODIFY COMMAND 命令建立/编辑命令文件本身，但仍可用文本编辑程序或字符处理程序建立或修改。命令文件用 DO 命令开始。命令文件包含任何 dBASE 命令，但是，由于一些命令（CREATE, INSERT, APPEND（来自键盘））需要用户输入超出命令文件范围的内容，所以要仔细。

命令文件可以嵌套。例如：命令文件在执行过程中可以包含 DO 命令，重复一遍，在使用嵌套结构时要仔细，dBASE 准许在任何给定的时间里最多打开16个文件。因此，若有一个USE文件则仅还有15个命令文件可嵌套。某些命令也可用于工作文件（例：SORT 使用两个附加文件；REPORT, INSERT, COPY, SAVE, RESTORE和PACK 使用一个附加文件）。例如：若从嵌套的命令文件最外层发出 SORT 命令，则仅可使用13个命令文件(即：一个USE文件，2个SORT工作文件和13个命令文件一共 =16）。无论何时命令文件发出RETURN命令或遇到命令文件的文件尾，则关闭命令文件。此方法对其它命令也有效。

3.4 报表格式文件(.FRM)

REPORT命令既能产生一个格式文件，也能使用已存在的格式文件。格式文件包含报表生成者对题目，标题，求和及列内容的说明。格式文件由 dBASE的REPORT 对话生成。他们可用文本编辑或字符处理程序修改，从一开始就定义新的报表格式通常是较容易的。

3.5 文本输出文件(.TXT)

当指定“SET ALTERNATE TO ‹filename›”和“SET ALTERNATE ON”命令时，文本输出文件被建立。更详细的说明见SET命令。同样每当SDF（系统数据格式）或DELIMITED可选项使用时，COPY及APPEND命令也能产生文本（.TXT）文件。

3.6 索引文件(.NDX)

索引文件由 dBASE 的 INDEX 命令产生。它们含有寻找数据库文件记录的关键字和指针，索引是dBASE在一个大数据库中对数据快速定位的一个方法。见 INDEX命令。

3.7 格式文件(.FMT)

格式文件可含有“@”语句和“*”注释。该命令由“SET FORMAT TO ‹filename”命令来识别，且与后面的READ命令启动。象命令文件一样（格式文件类似），格式文件的建立和修改可用任何好的文本处理程序或 MODIFY COMMAND 命令来进行。然而格式文件不是必须的。“@”和“*”语句通常用在需要它们的命令文件中。

4.0 表达式

dBASE 中的表达式是一些简单项和运算符的组合，它能被计算并形成新的简单值。例如“2 + 2”是一个能被计算为“4”的表达式。表达式不一定总是数。表达式'abc' + 'def'计算的结果为'abcdef'(字符串连结)，或表达式 1 > 2 计算结果为逻辑值(布尔量)“.F.”(假)。

dBASE 表达式由下列成份构成：

* 数据库字段变量
* 存贮器变量
* 命令中的常数（字面值）
* 函数
* 运算

变量 - dBASE 中的变量是能改变其值的数据场。在 dBASE 文件中涉及到的记录的当前场名就是变量。它们的内容由移动文件指针或编辑当前记录来改变。变量也可用下

列命令建立或改变：STORE，RESTORE，COUNT，SUM，WAIT，ACCEPT，或INPUT。还有存贮器变量。

变量有三种类型

- 字符串型
- 数字型
- 逻辑型

常量 - 常量（或字面值）是一恒定的数据项，自定义其值。例如：1，'abc'，和.T.都是常量，不管数据库位置或任何存贮器变量命令如何，它们都有一恒定值。由于它们的值就由它们本身所代表，故称为字面值（相对于变量名而言）。它们代表的相应值是：数字1，字符串(含字母“a”，“b”和“c”)，和逻辑（布尔量）值为真（“.T.”）。

字符串常量必须用单引号（'），双引号（''），或方括号（[,]）括起来。若字符串含有其中一种“界限符”，则该界限符应当用另一种界限符括起来。例如：串'abc[def]ghi' 和[abc'ghi] 是有效的而'abc'def'ghi'是无效的。逻辑常量（真/假）：对于真值由“T”，“t”，“Y”，“y”代表（表示真或是），对于假值由“F”，“f”，“N”，“n”代表（表示假或不是）。

4.1 函数

函数是用在表达式中以执行那些难于或不可能用有规律的表达式来表达的特殊运算。在dBASE 中，有三个基本的函数类型：数字型，字符型和逻辑型。函数类型由其产生的值来决定。

取整函数：

INT（〈数字表达式〉）

该函数用于计算数学表达式并舍弃小数部分（若有的话）产生一整数值，数学表达式的内部值被INT函数截去尾数。

例：

```
. ? INT(123.456)
  123
. STORE 123.456 TO X
  123.456
. ? INT(X)
  123
```

记录号函数：

*

该函数的值是整数，对应当前记录号。

例：

```
. ? *
  6
```

```
4          (假定数据库已USE，且记
           录指针在4)
. SKIP
. ? *
  5
```

串函数

STR (〈数学表达式〉，〈长度〉，〔〈小数〉〕)

此函数用于计算数学表达式，并产生字符串。STR函数值是一个长度为〈长度〉的字符串。若指定了〈小数〉，它代表小数点右边数字的位数。命令后的全部说明可以是字面值，变量或表达式。

警告：当使用此函数产生一个索引关键字时，这些说明必须是字面值。

例：

```
. ? STR(123.456,9,3)
  123.456
```

子串函数：

$(〈字符表达式〉，〈开始位置〉，〈长度〉)

此函数可由串的指定部分形成另一个字符串。子串函数的值是以〈长度〉为字长，由取自字符串中的字符形成字符串，从表达式中取字符的起始位置为〈开始位置〉，字长为〈长度〉。〈开始位置〉和〈长度〉可以是字面值，变量或表达式。

若〈长度〉超过〈字符表达式〉的长度，或如果〈长度〉超过了〈字符表达式〉的〈起始位置〉后的剩余字符数，则结果仅是这些字符。见下面的例子。

警告：当使用此函数产生一个索引关键字时，说明必须是字面值。

例：

```
. ? $('abcdefghi',3,3)
  cde
. store 3 to m
  3
. store 3 to n
  3
. ? $('abcdefghi',m,n)
  cde
. ? $('abcdefghi',6,7)
  fghi
. DISPLAY FOR '8080'$TITLE
```

串变整数函数：

VAL(〈字符串〉)

该函数可从一个由数字、符号直至小数点组成的字符串产生一个整数。整数的长度等于

串中字符数。若字符串以数字开始，但串里有非数字字符，则 VAL 函数产生的值是数字打头的值。

另一种把字符数转变成数字的方法是使用“&”（见5.0宏命令）。当遇见代换命令时，“&”将把串转变成数（包括小数点）。

例：

```
. ? VAL('123')
  123
. ? VAL('123××× ')
  123
. ? VAL('123.456')
  123
. STORE '123.456' TO NUM
  123.456
. ? 14 + &NUM
  137.456
```

串长函数：

LEN(<字符串>)

该函数产生一整数，其值为被命名串的字符个数。

例：

```
. STORE 'abc' TO STRING
. ? LEN(STRING)
  3
```

删除记录函数：

*

这是一个逻辑函数，若当前记录已做删除标志，则值为 .TRUE.，否则为 .FALSE.。

例：

```
. ? *
.T.    （假定数据库已USE且当前记录已使用DELETE命令删除）
```

文件结束函数：

EOF

这是一个逻辑函数，若遇到 USE 文件尾（当前记录在数据库中为最后一个记录）则为 .TRUE.。例：

```
. ? EOF
.F.          （假定数据库已USE且不在最后一个记录位置）。
. GOTO BOTTOM
```

```
. ? EOF
.F.
. SKIP
. ? EOF
.T.
```

找子串函数：

@(〈字符串1〉，〈字符串2〉)

该函数产生一个整数值，该值为〈字符串1〉在〈字符串2〉中首次出现的位置。若串2中不含串1，则@函数值为零。注意：除了@函数告诉你在第二个串中出现第一个串的位置外，它与子串运算符“$”类似，而且能回答“串1在串2中的什么位置”。

例：

```
. ? @('def', 'abcdefghi')
  4
```

小写变大写函数：

!(〈字符串表达式〉)

该函数产生一个与字符串表达式相同的串，除非所有的小写字母已被变成大写字母。

例：

```
. ? !('abc')
ABC
```

数变字符函数：

CHR(〈数字表达式〉)

该函数产生与数学表达式等值的ASCII字符。也就是，若表达式是数13，则CHR（13）产生一回车ASCII字符。当用户需要向外部设备直接送控制时，（常常是向打印机送），该函数是很有用的。

例：

```
. ? 'abcd' + CHR(13) + '____'
____
abcd
____
```

日期函数：

DATE()

此函数产生一个字符串，该串以MM/DD/YY的格式包含着系统日期。这个字符串总是8个字符长。括弧间不需任何东西，它们仅表明这是函数（使用变量名“DATE”是无效的）。

dBASE 系统日期在调用 dBASE 时输入或随时使用SET DATE TO 命令输入。

例：

```
. ? DATE( )
09/15/81
. STORE DATE( ) TO MEMVAR
09/15/81
. SET DATE TO 4 1 82
. ? DATE( )
04/01/82
```

文件函数

FILE(〈串表达式〉)

这是一个逻辑函数，若〈串表达式〉存在，则为 .TRUE.，否则为 .FALSE.。

例：

```
. ? FILE('TRACE')
.T.
. USE TRACE
```

类型函数

TYPE(〈表达式〉)

这个函数产生一个单字符串，若〈表达式〉分别为字符、数字和逻辑型，则该串为 'C'，'N'或'L'。

例：

```
. STORE 1 TO X
. ? TYPE(X)
N
```

删尾空格函数：

TRIM(〈字符串〉)

TRIM 函数从一字段中删去尾部空格。通常为了显示时使列对齐，dBASE 的所有变量带有尾部空格。

注：因关键字长度在 dBASE 内部运用中是一可计算的量，故INDEX不能使用这个函数 。

例：

```
. STORE 'ABC    ' TO S
. ? LEN(S)
  9
. STORE TRIM(S) TO S
. ? LEN(S)
  3
```

4.2 运 算

dBASE 有四种基本的运算：算术，比较，逻辑和串。各种运算的专用运算符列表如下，

下面举一个不常见的例子。

要知道，运算符“两边”的类型必须一致这一点很重要。即可以是整数加整数，或字符与字符的连结，一个整数与字符相加的结果在 dBASE 看来是一个句法错。

```
. STORE 3 TO A
  3
. STORE '3' TO B
  3
. ? A+B
*** SYNTAX ERROR ***
  ?
? A+B
CORRECT AND RETRY(Y/N)?
```

这个错误的产生是由于数字和字符在机器中被看作是类型不同的量；数字 3 是十六进制数 3，而字符 3 是 ASCII 值33（十六进制）。这使程序很为难，不知道是执行加法操作还是连结操作。象上面的例子中应使用同类型变量：

```
. ? A+VAL(B)
  6
```

串'3'被转变成整数并执行加法操作。

1. 算术运算符（产生算术结果）

+	=	加法
-	=	减法
*	=	乘法
/	=	除法
()	=	圆括号用于组合运算

例：

```
. ? (4+2)*3
 18
. ? 4+(2*3)
 10
```

这是一个使用圆括号进行成组计算的例子

2. 比较运算符（产生逻辑结果）

<	=小于
>	=大于
=	=等于
<>	=不等于

<= = 小于或等于
>= = 大于或等于
$ = 子串运算符 （例如，若A和B是字符串，当A串等于B串或包含在B串中，则A$B为真.TRUE.）

例：

```
. ? 'abc'$'abcdefghi'          子串运算符$的例子
.T.
. ? 'abcd'$'ghijkll'
.F.
. DISPLAY FOR '8080'$TITLE  此命令的结果是：TITLE场中，带有'8080'字符
                            的那些记录，全部显示在屏幕上。
```

3. 逻辑运算符（产生逻辑结果）

.OR. =布尔或
.AND. =布尔与
.NOT. =布尔非 （一元运算符）

例：

```
. store t to a
.T.
. store f to b
.F.
. ? a .or. b
.T.
. store .not. b to c
.T.
. ? a.and. c
.T.
```

4. 串运算符（产生串结果）

\+ = 串连结
\- = 压缩空格的串连结

例：

```
. STORE 'ABCD ' TO A          在一个串连结中，两个串彼此被联上。
ABCD
. STORE 'EFGH' TO B
EFGH
. ? A+B
ABCD EFGH
```

. STORE 'ABCDE ' TO A　　在一个压缩空格的串连结中，尾空格被移到串尾。打头的和嵌入的空格不变。

ABCDE

. STORE '1234 67' TO B

1234 67

. ? A-B

ABCDE1234 67

执行顺序

数字、串和逻辑运算符组合使用时有一固定的运算顺序。即什么运算在前什么运算在后。下面的表指明了三种运算的优先等级。同级运算（1，2等等）的顺序自左至右。

例：

. ? 4+2*3

算术运算优先级	串运算优先级	逻辑
1) 圆括号，函数	圆括号，函数	.NOT.
2) 一元 +，-（数前的正负号）	关系运算，$（子串操作）	.AND.
3) *，/	+，-（连结）	.OR.
4) +，-（加，减号）		
5) 关系运算		

5.0 宏代换

如果在命令中遇到&号后跟着字符串存贮器变量名，dBASE 将用存贮器变量中的字符串取代&和存贮器变量名。允许用户一次定义部分命令，并反复在各种命令中调用。

当频繁使用复杂的表达式时，该函数很有用。也准许在嵌套的命令文件中传送参数。在&和下一个指定字符（包括空格）之间的所有字符被认为是存贮器变量名。

若用户要求把字符附加到&符号替换处，则存贮器变量名应当用句号结束。在替换时句号象&一样被消去。

若&后面没有跟一个有效的存贮器变量名，则函数不展开，并且&仍然保留在命令行中。

例：

. ACCEPT "Enter data disk drive letter" to DR　在执行时，若对 ACCEPT的提示回答"B"，则将是 USE B:DATAFILE

USE &DR:DATAFILE

. STORE 'DELETE RECORD ' TO T

&T 5　　在执行时将是DELETE RECORD 5

更好的例子见附录 A。

6.0 与非数据库处理程序的接口

dBASE 可从 dBASE 以外的处理程序（如：BASIC，FORTRAN，PASCAL）建立

的文件中读数据，并能产生被其它处理程序可以接受的文件。

当用指定的 SDF(SYSTEM DATA FORMAT) 可选项时，APPEND 能读标准的 ASCII 文本文件（使用CP/M 文本行后用回车换行的规定）。类似地，使用 SDF 可选项，使COPY命令产生标准的ASCII 格式文件。除非不使用可选项，否则用SDF和DELIMITED 可选项，所产生的文件类型将是 .TXT。

某些处理程序及语言按限定的格式读写文件。该形式是所有的场以逗号分隔，且字符串被引号括起来。当 DELIMITED 关键字包含在命令中时，dBASE能 APPEND 和 COPY 这些文件。若使用DELIMITED这个特点，则也假定了SDF被使用。

由于一些处理程序用单引号，而有的用双引号去定界字符串，APPEND承认任何一种。COPY命令能产生标准的单引号，但可输出任何一种由 DELIMITED项的WITH短语定义的字符。建议你最好使用单引号或双引号。

当一个“，”被用在COPY命令的WITH 短语中时，出现特殊情况。所有字符串中的尾空白及数字中打头的空白被删去。同样，字符串将不用单引号或其它字符括起来。

例：

```
. USE <FILENAME>.DBF
. COPY TO <FILENAME>.TXT DELIMITED WITH ''
. USE <FILENAME>.DBF
. APPEND FROM <FILENAME>.DAT SDF
```

7.0 命令分类

在 dBASE 正常使用期间，各种命令的组合使用，能完成一种特殊的目的。如下面所展示的每组命令。一些 dBASE 命令与多数“现代化”计算机所使用的语言在结构上类似。这些命令列在 COMMAND FILE类中。控制这些命令的使用有一些特殊规则，见 9.0 部分的说明。

文件的建立 - 下面的命令建立数据库文件和有关文件：

- CREATE - 建立一个新的数据库文件结构。
- COPY - 复制现存的数据库以建立备份。
- MODIFY - 改变数据库结构。
- REPORT - 建立一个报表格式文件。
- SAVE - 把存贮器变量存储到大容量存贮器中。
- INDEX - 建立一个索引文件。
- REINDEX - 重新组合一个老索引文件。
- JOIN - 连结两个数据库并输出。
- TOTAL - 输出统计过的数据库记录。

添加数据 - 下面的命令把新数据记录加到数据库中：

- APPEND - 在文件尾加数据。

* CREATE －允许在建立时加数据。
* INSERT －将数据插入到文件中。

编辑数据 －下面的命令在数据库中编辑数据：

* CHANGE －编辑场列。
* BROWSE －全屏幕查看和编辑数据
* DELETE －为欲删除的记录作标志。
* EDIT －修改数据库中指定的数据场。
* PACK －删去带删除标志的记录。
* RECALL －撤消删除标志。
* REPLACE －用值取代数据场。
* READ －从用户定义的全屏幕置换数据。
* UPDATE －成批修改数据库

数据显示命令 －下面的命令显示数据库中所选择的数据：

* @ －在CRT或打印机上显示用户格式数据。
* BROWSE －尽可能多的场显示在屏幕上，最多可显示19个记录。
* COUNT －统计满足表达式条件的记录数。
* DISPLAY －显示记录，场及表达式。
* READ －在全屏幕方式下显示数据和提示信息。
* REPORT －有格式地显示数据报表。
* SUM －对一组数据库记录计算并显示表达式的和。
* ? －显示一个表达式清单。

定位命令 －下面的命令用于把当前记录指针置于指定的记录：

* CONTINUE －定位于下一个满足LOCATE命令中指定条件的记录处。
* FIND －定位于满足索引文件中关键字的记录处。
* GOTO －定位于指定的记录处。
* LOCATE －找满足条件的记录。
* SKIP －向前或向后定位。

文件操作命令 －下列命令改变整个数据库文件

* APPEND －合并dBASE文件或系统数据格式文件（SDF）。
* COPY －将数据库复制成另一数据库或SDF文件。
* DELETE －删文件。
* DO －指定一个后面要执行的命令文件。
* RENAME －改文件名。
* SELECT －选择USE文件。
* SORT －建立一个按某个数据场分类的数据库备份。