

第三篇 实用软件篇

第一章 数据库

§1.1 dBASE III

1.1.1 概述

1. 简介

(1) 发展概况。美国 Ashton-Tate 公司的 dBASE 系列的数据库管理系统, 自 1978 年问世以来, 很快便风靡世界。自 80 年代初引入我国以后, 以其优越的性能, 灵活方便的使用方法, 简单易学的特点赢得了广大科技人员及企事业管理人员的喜爱, 已成为在微机上使用最广泛的实用软件之一。

美国 Ashton-Tate 公司用 10 年的时间, 勾勒出了 dBASE 系列软件的发展轨迹。dBASE 家族第一个成功的产品 dBASE II, 只有 96 条命令(包括 28 条 SET 命令); 它所能管理的数据库文件最多只能有 32 个字段, 65535 条记录; 同时只能对两个数据库工作区操作。到了 dBASE III, 就具有了 118 条命令(包括 34 条 SET 命令); 所能管理的数据库文件可以有 128 个字段, 10 亿条记录; 同时可访问的工作区数达到了 10 个之多。1986 年问世的 dBASE III PLUS 除增加了编目管理、现场管理、屏幕格式文件的自动生成等一系列新功能之外, 更重要的是增加了网络功能。由于它较好地解决了在微机局部网络上共享数据库时产生的数据安全性及完整性问题, 使之成为在微机局部网络上可以成功运行的为数不多的优秀数据库管理系统之一。1988 年诞生的新一代数据库管理系统 dBASE IV, 被认为是 dBASE 家族的骄子。它除了保留 dBASE III PLUS 的全部功能之外, 增加了非过程性接口 Control Center, 增加了举例查询功能 QBE, 提供了方便程序员的应用程序生成器。除此之外, 还支持结构化查询语言 SQL。dBASE IV 所能管理的数据库文件可以有 255 个字段; 能打开的文件总数增加到 99 个; 命令条数增加到 245 条(包括 75 条 SET 命令); 增加了数组功能及自定义函数功能; 计算与统计功能也增加了许多, 运行速度是 dBASE III PLUS 的 9 倍。所有这些优越的性能, 是其在世界市场竞争中取胜的保证。

(2) 用途。dBASE III 适宜于中小型事务管理, 例如: 销售管理、财务管理、仓库管理、人事管理、图书资料管理、病房管理、饭店管理等。

(3) 特点。

① 简单易学, 灵活方便。dBASE III 每条命令的功能从命令名上就能一目了然, 易学易记。每条命令的基本功能很简单, 很容易掌握。而各命令又有许多选择项, 将

这些选择项灵活地组合起来,可以实现十分复杂的功能,使用起来极为方便。

② 结构严谨,命令丰富。dBASE III 的系统盘只是一片 $5\frac{1}{4}$ 的 360K 软盘。如果不使用汉字,只要求 256 K 字节的内存。但它却提供了一系列功能齐全的命令,并支持投影、选择及连接这三种基本的关系运算,是一个使用方便的关系型数据库管理系统。

2. 主要特性

(1) 技术规格。

数据库文件的指标:

每个库记录条数	$\leq 1 \times 10^9$ 条
每个库字节总数	$\leq 2 \times 10^9$ byte
每条记录最大长度	在 DBF 文件中为 4000 字节
字段个数	≤ 128 个

字段宽度的指标:

字符型字段	≤ 254 个字节
日期型字段	= 8 个字节
逻辑型字段	= 1 个字节
备注型字段	≤ 4096 个字节
数值型字段	≤ 19 个字节

文件操作的指标:

最多能打开 15 个各种类型的文件。

同时可打开 10 个数据库文件。如果在数据库中使用备注型字段,则该数据库按两个文件计算,一个为数据库文件本身,另一个为备注文件。

每个现行的数据库文件可以打开 7 个相应的索引文件。

每个现行的数据库文件可以打开一个屏幕格式文件。

数值精度:

数字最大位数为 19 位(包括小数点),精度为 16 位。

最大数: $1 \times 10^{+308}$ 。

最小正数: 1×10^{-307} 。

有关内存变量的指标:

现行内存变量总数 ≤ 256 个。

内存变量占用的总字节数 ≤ 6000 。

以上所有指标都受硬件配置的限制。

(2) 文件类型。dBASE III 可以管理以下 9 种类型的文件:

① 数据库(.dbf)文件。由文件头和文件内容两部分组成。文件头存放结构,文件内容存放数据记录。

② 数据库备注(.dbt)文件。备注文件用来存放数据库文件中备注型(MEMO)字段的值。是数据库文件的辅助文件。

③ 索引(.ndx)文件。索引文件实际上是对数据库文件按某个关键字建立的索引表,使用该索引表,数据库文件中的记录就可以按逻辑顺序(例如,字母顺序或数值大小)而不是物理顺序排列。

④ 命令(.prg)文件。命令文件中存放的是用 dBASE III 的命令编写的程序。

⑤ 格式(.fmt)文件。格式文件生成用户需要的输入和输出屏幕格式。

⑥ 标签(.lbl)文件。标签文件中存放用 LABEL 命令打印标签时所需要的信息。

⑦ 报表格式(.frm)文件。报表格式文件中包含用 REPORT 命令打印报表时需要的信息。

⑧ 内存变量(.mem)文件。内存变量文件用来存放用户生成的内存变量。

⑨ 数据(文本输出)(.txt)文件。文本输出文件提供了 dBASE III 与其它应用软件间相互连接的手段。该类文件只包含可打印的 ASCII 字符。

上述 9 类文件的文件名生成规则与 DOS 下的各类文件的文件名规则一致,扩展名是系统给定的默认值,用户可以选用自己定义的扩展名。

(3) 数据库文件结构。数据库文件的结构是通过对各字段的定义而建立的。结构包含如下内容:

① 字段名。字段名最多可由 10 个字符组成。可以包括字母、数字、汉字和下画线,打头的必须是字母,不允许有空格。

② 字段类型。dBASE III 包含如下五种类型的字段:

③ 字符型(c 型)。字符型字段的内容可以是任何从键盘输入的可打印的字符,其中包括字母、汉字、数字、特定符号和空格。

④ 数值型(N 型)。数值型字段中存放的是可进行算术运算的数据值。包括整数型和小数型(即通常所说的实数型)两类数据值。当为整数时,最多可为 19 位正整数或 18 位负整数;当为小数时,小数位数不能超过 15 位,整数部分不能超过 16 位。

⑤ 逻辑型(L 型)。逻辑型字段只接受代表逻辑值的单个字母。代表逻辑“真”值的字母有:T,t,Y,y;代表逻辑“假”值的字母有:F,f,N,n。

⑥ 日期型(D 型)。日期型字段用来存放日期,默认日期格式是 mm/dd/yy (即月/日/年,月、日、年各自只允许用两位数字表示)。日期型字段的宽度规定为 8 个字符。

⑦ 备注型(M 型)。备注型字段中存放一个指针,该指针指向存放实际信息的 .dbt 文件的块。备注型字段的宽度规定为 10,但与之对应的 .dbt 文件的块中却可以存放最多至 4096 个字节的内容。定义备注型字段之后,同名的 .dbf 和 .dbt 文件必须同时存在,否则不能正常使用。

③ 字段宽度。字段宽度限制了字段中所能存放的数据的字符个数或数字位数。C 型字段最多可达 254 个字符,若使用汉字,最多可达 127 个汉字;N 型字段若存放小数,则小数点也占一位;L 型和 D 型是定宽的,用户不能自行定义;M 型在 .dbt 文件中的宽度是可变的,当没有数据输入时,宽度为 0,只要有数据输入,则宽度至少是一个块(512 个字节),最多可达 8 个块,共计 4096 个字节。

(4) 内存变量。内存变量是独立于数据库结构而存在的一种有名字的数据项。它是一种临时存放数据的方便手段。例如,对字段求和之后,可将所得的和存放在内存变量之中,以便输出或再次计算。另外,在程序中,内存变量更是不可缺少的。

内存变量的名字与字段名规则相同。为了避免与字段名混淆,引用时可在前面加上 M ->。

dBASE III 最多允许有 256 个内存变量同时存在于内存中,且要求它们所占用的内存空间必须少于 6000 个字节。

内存变量可以以 .mem 文件的形式存盘,使用时可以恢复。

在程序方式下,内存变量分为全局型(PUBLIC)和局部型(PRIVATE)两种,但在命令方式下定义的内存变量只有全局型一种。

内存变量有四种数据类型:字符型,数值型,日期型和逻辑型。各种类型的有关规定与数据字段相同。

(5) 操作规则。使用 dBASE III 的命令时必须遵循如下规则:

① 每条命令必须以一个命令动词开头,前面不能有空格。

② 每条命令必须符合手册中给出的句法格式:

VERB[<范围>][<表达式清单>][FOR/WHILE<条件>]

每条命令都由命令动词(VERB),必选项和可选项组成。方括号中的内容是可选项。其中<范围>选择项包括三种选择:ALL, NEXTn 和 RECORD n。FOR 和 WHILE 之间的斜杠表示二者只能选择其一。选用 FOR<条件>时, dBASE III 将对所有记录或范围限定内的记录进行判断,直到记录指针达到文件末尾或范围末尾为止。选用 WHILE<条件>时, dBASE III 在遇到第一条不满足条件的记录时便停止。因此,前者适用于各种数据库文件,而后者只适用于排过序或建立了索引的数据库文件。命令行中的选择项可以以任何顺序出现。

③ 命令行的最大长度不得超过 254 个字符。由于屏幕行宽有限,当命令行超过屏幕行宽时,可以用分号“;”换行,也可以不输入换行符而连续输入, dBASE III 会自动插入换行符。

④ 命令的各部分之间可以用一个以上空格分隔,但空格要算在命令行长度之内。

⑤ 命令动词和命令短语可以缩写为前 4 个字母。

⑥ 命令行中的所有内容既可以用大写字母,也可以用小写字母,还可以大小写混

合。其中，字段名、内存变量名及文件名可以用汉字，但必须是半角汉字。

⑦ 应尽量避免使用与命令动词相同的字段名、内存变量名或文件名。

⑧ 在程序中，要注意 DO WHILE/ENDDO、DO CASE/ENDCASE、IF/ENDIF 的嵌套关系。

(6) 表达式

表达式可以是一个字段、一个内存变量、一个常数或函数，也可以是用合法运算符将它们正确连接起来的式子。在后一种情况下，要注意运算符两端的数据类型要一致(日期运算是一种特殊情况)。

(7) 运算符。dBASE III 提供 5 类运算符，可以进行如下 5 种运算：

① 算术运算：算术运算产生数值型的结果。可进行的算术运算有：

+	加
-	减
*	乘
/	除
** 或 ^	乘方
()	括号操作

② 比较运算：比较运算产生逻辑型的结果。参与比较运算的数据可以是字符型的、数值型的或日期型的，但要求运算符两端的数据类型要一致。可进行的比较运算有：

<	小于
>	大于
=	等于
<> 或 #	不等于
<=	小于等于
>=	大于等于

③ 逻辑运算：逻辑运算产生一个逻辑型的结果。可进行的逻辑运算有：

.AND.	逻辑与
.OR.	逻辑或
.NOT.	逻辑非
()	括号分组

④ 字符串运算：

\$ # 子串比较。在被比较的两个字符串中，若后一个包含前一个，结果为 .T.，反之，结果为 .F.。例如，“ABC”\$“NABCDE”结果为 .T.。

+ 将两个字符串连接起来形成一个新串。

- 字符串连接运算。与“+”连接不同之处在于可将前一个字符串的尾部空格移

到新生成的字符串的末尾。例如, "AB" + "CD" 结果为 "ABCD"。而 "AB" - "CD" 结果为 "ABCD"。

⑤ 日期运算:

- + 一个日期与一个整数相加产生另一个日期。例如, 若A的值为日期 01/01/88, 则 A+365 得 01/01/89。
- 日期与日期相减得到一个整数, 表示两个日期之间相隔的天数; 日期与整数相减得到另一个日期。

(8) 运算优先级。算术运算的优先级如下:

- ① 一元运算(+ 或 -)
- ② 指数运算:
- ③ 乘或除
- ④ 加或减

逻辑运算优先级如下:

- ① 逻辑非
- ② 逻辑与
- ③ 逻辑或

当几类运算混合在一个表达式中时, 优先顺序如下:

- ① 算术运算或字符串连接运算
- ② 比较运算
- ③ 逻辑运算

在上述所有运算中, 括号优先级最高。同级操作则以左至右执行。

(9) 全屏幕操作。当使用 APPEND, BROWSE, CHANGE, CREATE, EDIT, INSERT, MODIFY 和 READ 命令时, 可以用下列屏幕编辑键输入或修改数据。

控制键	等价键	功 能
↑	^E	将光标上移一行或一个字段
↓	^X	将光标下移一行或一个字段
←	^S	将光标向左移一步。在菜单选择方式下, 向左移一个选择项
→	^D	将光标向右移一步。在菜单选择方式下, 向右移一个选择项
^→	^B	在 BROWSE 命令下, 将光标向右平移一个字段; 在 MODIFY REPORT 命令下, 向上翻滚显示文件结构; 在 MODIFY COMMAND 命令下, 将光标移到行末; 在 CREATE(或 MODIFY) STRUCTURE 命令下, 将光标移到尾部新定义字段开始处
^←	^Z	在 BROWSE, MODIFY REPORT 和 MODIFY COMMAND 命令中与 "→" 作用相似, 仅方向相反。在 CREATE 命令中, 将光标移回到本字段开始处

续表

控制键	等价键	功 能
Del	^G	删去光标指出的字符
End	^F	将光标向右移一个字
^End	^W	编辑完, 结果存入磁盘, 退出
ESC	^Q	不保存修改结果而返回到命令方式(注意, 在 APPEND 和 BROWSE 命令中保存已输入或修改完的记录, 只有按此键时的当时记录不保存)
Home	^A	将光标向左移一个字(在 CREATE 命令中, 光标移到上一字段开始处)
^Home		进入和退出菜单选择方式的反复开关
INS	^V	反复控制插入模式的开关。当打开时, 键盘上输入的字符将插在光标之前; 关闭时, 键盘输入字符将取代光标指向的字符
^KW		在 MODIFY COMMAND 命令中, 将整个文件写入到另外的文件中
^KR		在 MODIFY COMMAND 命令中, 把另外一个文件读进来
^N		插入一个新的行或新的字段
PgUp ~	^R	在分页显示时, 回到前一页。对 APPEND, EDIT 等命令相当于回到前一记录(如果一个记录不超过一页)或前一页(如果一个记录超过一页)。对 BROWSE 命令相当于回到前一页(每页 17 个记录, 每个记录一行)
PgDn	^C	与 PgUp 键功能相反
^PgDn		当光标停在备注型字段时按下此键, 便进入输入和编辑该字段内容的字处理方式
RETURN (回车)		将光标移到下一行或下一字段。在 APPEND 命令中, 当光标处于一空记录的第一个字符时敲入此键, 则存盘并退出; 在 EDIT 命令中, 当光标处于最后一个记录的最后一个字段时敲入此键, 也是存盘并退出; 在 MODIFY COMMAND 命令中, 若处于插入方式, 按此键就插入一行; 在菜单选择时, 按下此键则选中当前标有光标的项
^T		自光标处开始向右抹去一个字
^U		在 BROWSE 或 EDIT 命令下, 给记录做删除标记, 工作于反复模式下; 在 CREATE 和 MODIFY 命令中, 删去一个字段
^Y		自光标处起抹到字段末或该行末

3. dBASE III 的运行和退出

(1) 运行环境。

- IBM PC, PC/XT, COMPAQ, 长城机及其它PC 的完全兼容机
- 不使用汉字时至少要求 256K 字节的内存。使用汉字时要把汉字系统占用的内存加上, 通常要求 512K 以上的内存。
- 要求 MS DOS 或PC DOS 2.0 以上版本的操作系统。使用汉字时, 要求相当于 DOS2.0 以上版本的汉字操作系统。
- 要求两个 360K 软盘驱动器, 或一个 360K 软盘驱动器和一个硬盘驱动器。
- 列宽至少为 80 的打印机。

(2) CONFIG.SYS 文件。用来启动系统的 DOS 盘上应有一个 CONFIG.SYS 文件, 它至少应包含如下两条语句:

FILES=20

BUFFERS=24

第一条语句规定了 dBASE III 可同时打开的文件数(通常为 15 个, 除去系统所占用的 5 个), 用户可在 10—20 之间选择; 后一条语句规定了 dBASE III 在内存中可开辟的缓冲区的数目。若内存不够用, 可适当减少 BUFFERS 的数目, 例如令 BUFFERS=12。

可以用 DOS 的 COPY CON:命令或其它编辑程序生成 CONFIG.SYS 文件。

(3) CONFIG.DB 文件。dBASE III 系统盘上应有一个 CONFIG.DB 文件, 它的主要内容:

- 功能键设置。
- 部分 SET <命令字>TO <参数>形式的 SET 命令。
- 所有 ON/OFF 形式的 SET 命令。
- dBASE III 使用何种编辑软件。
- 内存变量可使用的最大内存空间。
- 进入 dBASE III 后首先执行哪条命令。

CONFIG.DB 文件中的语句格式是:

<命令字>=<值> (例如: DEFAULT=B:)

CONFIG.DB 文件可识别的项有:

命令名	参 量	说 明
ALTERNATE	文件名	
BELL	ON/OFF	
CARRY	ON/OFF	
COLOR	颜色定义项	
COMMAND	dBASE III 命令	定义引导 dBASE III 后立即执行的命令

续表

命令名	参 量	说 明
CONFIRM	ON/OFF	
CONSOLE	ON/OFF	
DEBUG	ON/OFF	
DECIMALS	整数(0—14)	
DEFAULT	驱动器标识符	
DELETED	ON/OFF	
DELIMITER	ON/OFF/1—2个字符	
DEVICE	SCREEN/PRINTER	
ECHO	ON/OFF	
ESCAPE	ON/OFF	
EXACT	ON/OFF	
F _n	字符串	n = 2—10
HEADINGS	ON/OFF	
HELP	ON/OFF	
INTENSITY	ON/OFF	
MARGIN	整数(0—254)	
MAXMEM	整数(200—720)	代表内存字节数, 以 K 为单位
MENUS	ON/OFF	
MVARSIZ	整数(1—31)	代表内存字节数, 以 K 为单位
PATH	路径名清单	
PRINT	ON/OFF	
PROMPT	字符串	定义系统提示符
SAFETY	ON/OFF	
STEP	ON/OFF	
TALK	ON/OFF	
TEDIT	程序名	用作编辑器的字处理程序
UNIQUE	ON/OFF	
WP		用于编辑备注型字段的字处理程序

CONFIG .DB 文件的建立方法与CONFIG .SYS 相同。

(4)运行。用存有 CONFIG .SYS 的DOS 盘启动系统, 然后把装有 CONFIG.DB 的 dBASE III 系统盘插入 A 驱动器, 在 A > 提示符下键入 dBASE 便可。也可以把 dBASE III 系统盘的内容复制到硬盘之中, 在 C > 提示符下键入 dBASE。

(5)退出。在 dBASE III 的圆点提示符下键入 QUIT 便退出了 dBASE III, 回到 DOS 提示符下。

1.1.2 命令

1. 命令分类

(1)建立文件的命令。下列命令建立数据库文件和其它有关文件:

COPY	复制当前的数据库文件
COPY FILE	复制各类文件
CREATE	建立新的数据库文件
INDEX	建立索引文件
JOIN	将两个数据库文件连接起来生成第三个数据库文件
MODIFY COMMAND	生成命令文件、屏幕格式文件和过程文件
MODIFY LABEL	建立标签格式文件
MODIFY REPORT	建立报表格式文件
SAVE	将当前内存变量存到磁盘文件之中
SORT	为当前数据库文件生成一个按关键字值顺序排列的新的数据库文件
TOTAL	生成一个分类求和文件

(2)追加及插入数据记录的命令。以下命令为数据库文件追加新记录或将新记录插入到数据库文件之中:

APPEND	在数据库文件末尾追加新记录
BROWSE	可在数据库文件末尾追加新记录
INSERT	在数据库文件中插入新记录

(3)数据记录的修改及编辑命令。下列命令编辑修改数据库文件中的记录:

BROWSE	用全屏幕窗口显示方式编辑数据记录
CHANGE	编辑修改数据库文件中的指定字段或记录的内容
DELETE	给数据记录加删除标记
EDIT	修改数据库文件中指定字段或记录的内容
PACK	将加了删除标记的记录从数据库文件中清除
READ	与@...GET一起使用,可编辑修改字段或变量的内容
RECALL	去掉记录的删除标记
REPLACE	用指定的值替换字段的内容
UPDATE	更新数据库文件的某些内容
ZAP	删除数据库文件的全部记录

(4)数据记录的列表及显示命令。下列命令有选择地显示数据库文件中的数据记录,

无论 SET TALK ON/OFF 为何种状态:

@...SAY	在屏幕上或打印机上灵活地输出字段内容
?	显示字段内容
BROWSE	用全屏幕窗口方式显示数据记录
DISPLAY	显示记录最常用的方法
LIST	列表显示记录,是最常用的方法
REPORT	以报表形式反映数据库文件内容

(5) 对数据记录做数值计算的命令。下列命令对数据库文件的记录计数或对数值型字段求和、求平均数或分类求和:

AVERAGE	求指定的数值型字段的平均值
COUNT	求指定范围内的记录条数
TOTAL	对数据记录分类求和
SUM	求指定的数值型字段的和

(6) 移动记录指针及查找记录的命令。下面的命令可重新定位记录指针或查找数据记录:

COUNTINUE	与 LOCATE 命令一起使用以查找另一条满足条件的记录
FIND	在索引过的数据库文件中查找记录
GOTO	将指针定位到指定记录
LOCATE	查找满足条件的记录
SEEK	在索引过的数据库文件中查找记录
SKIP	使记录指针在数据库文件中相对移动

(7) 对整个数据库文件进行操作的命令。以下命令将影响到整个数据库文件:

APPEND FROM	将指定文件的内容转移到当前数据库文件之中
CLOSE	关闭指定的文件
ERASE	删除数据库文件
MODIFY STRUCTURE	修改数据库文件的结构
REINDEX	给现存的索引文件重建索引
RENAME	给文件改名
SELECT	选择数据库文件工作区
USE	打开新的数据库文件,并关闭旧的

(8) 对其他类型的文件进行操作的命令。下列命令用于建立命令文件、屏幕格式文件、标签文件及报表格式文件:

MODIFY COMMAND	建立命令文件、屏幕格式文件及其它 ASCII 码文件
CREATE/MODIFY LABEL	建立或编辑标签文件

CREATE/MODIFY REPORT	建立或编辑报表文件
LABEL FORM	输出标签
REPORT FORM	输出报表
(9) 对内存变量进行操作的命令。下面的命令都涉及到内存变量:	
@ ... SAY ... GET	显示或编辑内存变量的内容
?	显示或计算表达式的值
AVERAGE	将求出的平均值存入内存变量之中
CLEAR ALL	初始化 dBASE III, 关闭所有文件
CLEAR MEMORY	清除当前内存中的内存变量
COUNT	将记录计数的结果存入内存变量
DISPLAY MEMORY	显示当前内存变量
READ	与 @ ... GET 一起使用, 可将数据读入内存变量之中
RELEASE	清除内存变量
RESTORE	将磁盘文件中的内存变量恢复到内存之中
SAVE	将当前内存变量保存到磁盘文件之中
STORE	给内存变量赋值
SUM	将求出的和存入内存变量之中
PRIVATE	将内存变量定义为局部型的
PUBLIC	将内存变量定义为全局型的
(10) 程序设计命令。以下命令专用于程序设计之中:	
ACCEPT	暂停程序的执行, 接受键盘输入的字符型数据
CANCEL	撤消命令文件的执行
DO	执行程序或过程
DO WHILE ... ENDDO	执行一段循环程序
DO CASE ... ENDCASE	从多种情况中选择执行一种
IF ... ELSE ... ENDIF	从两种选择中执行一种
EXIT	跳出循环
LOOP	使循环短路
INPUT	暂停程序的执行, 接受键盘输入的数据
MODIFY COMMAND	建立和编辑修改命令文件
PARAMETERS	为过程传递参数
PROCEDURE	开始一个过程
RETURN	程序结束返回

WAIT	暂停程序的执行, 接受键盘输入的单个字符
(11) 辅助类命令:	
ASSIST	菜单驱动的 dBASE III 命令
DIR	显示当前磁盘文件目录
DISPLAY STATUS	显示 dBASE III 当前系统状态
CLEAR	清屏幕
DISPLAY STRUCTURE	显示数据库文件的结构
EJECT	打印机换页
HELP	给出帮助信息
SET	设置系统参数
QUIT	退出 dBASE III

2. 命令细述

命令: ? / ? ?

格式: ? / ? ? <表达式清单>

说明: 该命令计算和显示表达式的值。

单问号表示先回车换行再计算及显示表达式的值, 而双问号不回车换行。当在程序中使用, 双问号在当前光标处或当前打印机头位置输出表达式的值。当使用单问号且后面没有表达式时, 将输出一个空行。

举例:

(1) 数据库文件 BOOKS 具有书名、价格、出版日期、卖完否及内容摘要等五个字段, 欲知第一条记录的书名, 可使用命令:

.USE BOOKS

.? 书名

《红与黑》

欲知买 5 本《红与黑》要花多少钱 (每本价格 5.10 元)

.? 价格 * 5

25.50

(2) 求表达式的值:

.x = 5

.? x + 2 - 5

2

命令: @

格式: @ <行, 列> [SAY <表达式> [PICTURE <短语>]]

[GET<变量>][PICTURE<短语>][RANGE<表达式,表达式>]
[CLEAR]

说明:按给定坐标显示或打印数据项。可用它生成用户屏幕格式。

命令中行坐标和列坐标可以是整数,也可以是能计算出整数值的数值表达式。

对于一般的 24 行 80 列的显示器。使用英文 dBASE III 时,行坐标取值从 0 到 23,列坐标取值从 0 到 79;使用汉字 dBASE III 时,若为 11 行汉字显示系统,行坐标取值从 0 到 10。若是 24 行汉字显示系统,行坐标取值从 0 到 23。无论前一种或后一种,列坐标取值均从 0 到 79。dBASE III 通常将第 0 行留作状态行,应避免使用。

当使用打印机时,行坐标的取值从 0 到 255,超出后会空走纸。列坐标也是从 0 到 255。注意,后一条 @ 命令的行坐标不能小于前一条 @ 命令的行坐标,否则将会导致打印机换页。当两条 @ 命令的行坐标相同时,后一条 @ 命令的列坐标要大于前一条,否则也会导致换页。

SAY 选择项的表达式必须是合法的 dBASE III 表达式,当与 GET 子句一起使用时,SAY 子句的表达式常常是提示字符串。

GET 子句常用于显示或编辑(与 READ 一起使用时)字段或内存变量的内容。注意,备注型字段的内容不能被编辑。

RANGE 选择项用来限制 GET 变量输入的数值型或日期型数据的范围。前一个表达式是下限,后一个表达式是上限。注意,该选择项只能与 GET 及 READ 一起使用。

@<行,列>不带任何选择项,用来清除从给定列开始的指定行。

@<行,列>CLEAR 用于清除从给定坐标开始直到屏幕右下角的部分屏幕。

PICTURE 选择项为用户提供格式化输出的能力以及对输入数据的强制性限制。<短语>可以由功能符和格式符组成。

(1)功能符:

C—— 在一个正数后面显示CR(表示收入)

X—— 在一个负数后面显示DB(表示支出)

(—— 将一个负数用括号括起

B —— 将数值型数据移到最左边显示

Z —— 将数字值0作为一个空白字符串显示

D —— 美国日期格式

E —— 欧洲日期格式

A —— 只适用于字母字符

! —— 将小写变大写显示

R —— 使要加入的字符不是去代替原来的字符,而是插入到原来的字符串中间显示

在这 10 个功能符中, C, X, (, B 和 Z 只适用于数值型数据。其中 C, X, (仅能与 SAY 子句联用。D, E 适用于日期型、字符型和数值型数据。A, !, R 只适用于字符型数据。

使用功能符时, 要在前面加上 @。

(2) 格式符。dBASE III 为用户提供了 11 种类型的标准格式符。其中, 有些格式符可以与 SAY 子句一起使用, 以实现格式化输出; 有些可以与 GET 及 READ 一起使用, 以实现格式化输入。同一种格式符用于格式化输入或格式化输出时的作用略有不同。

① 可与 GET 子句及 READ 一起使用的格式符:

9 —— 只允许字符型数据的数字及数值型数据的数字及符号

—— 只允许数字、空格和符号

A —— 只允许字母

L —— 只允许逻辑型数据

N —— 只允许字母和数字

X —— 允许任何字符

! —— 将字母转换成大写形式, 而对其它字符无影响

. —— 指定小数位置

以上 8 种格式符用于对 GET 子句被 READ 语句编辑的变量所输入的内容加以限制。

② 可与 SAY 子句一起使用的格式符:

A, L, N, X, #, 9 —— 可显示任何字符

\$ —— 在显示数值型数据时, 数据的前导 0 用 \$ 代替

* —— 在显示数值型数据时, 用 * 代替数据的前导 0

. —— 指定小数位置

, —— 如果逗号左边有数字时, 逗号才被显示出来。当被显示的数值型数据位数多于三位时, 才将其三位一组地分开

! —— 将小写字母转换成大写字母显示

<其它符号> —— 对于字符型数据, 只有使用了 R 功能符, 才能将<其它符号>插入字符串中间显示; 否则, 只能用<其它符号>去代替相应位置上的字符。对于数值型数据, 有无 R 功能符都能将<其它符号>插入显示。

功能符和格式符可以结合起来使用, 这时, 应在它们中间置入一个空格。功能符和格式符大小写均可。

当在打印机上输出信息时, 只能选用 SAY 子句, 而不能选用 GET 及 CLEAR 选择项。打印机的行坐标列坐标最好使用函数 PROW () 和 PCOL ()。

举例:

例 1 下面的程序说明@命令的基本功能:

```
@ 5,7 SAY "这是一本书"
@ 3,5 SAY "3*5+2="
@ ROW( ), COL( ) SAY 3*5+2
@ ROW( )+1,1 SAY "下一个数据是:"
```

第一条语句在屏幕的 5 行 7 列开始的位置上显示出字符串: 这是一本书。第二条语句在屏幕的 3 行 5 列开始的位置上显示出字符串: 3*5+2=。第三条语句在 3 行 11 列显示 3*5+2 的值 17。第四条语句从 4 行 1 列开始显示字符串: 下一个数据是:。

例 2 有屏幕格式文件 format1.fmt, 其内容如下:

```
@ 4,2 SAY "  学生姓名: " GET 学生姓名
@ 5,2 SAY "语文考试成绩: " GET 语文成绩
@ 6,2 SAY "数学考试成绩: " GET 数学成绩
@ 7,2 SAY "物理考试成绩: " GET 物理成绩
@ 8,2 SAY "化学考试成绩: " GET 化学成绩
@ 9,2 SAY "英语考试成绩: " GET 英语成绩
```

现使用如下命令对其操作:

```
.USE student          { 打开学生成绩数据库文件 }
.SET FORMAT TO format1.fmt
.EDIT
```

屏幕显示如下:

```
学生姓名: 张明
语文考试成绩: 84.00
数学考试成绩: 85.40
物理考试成绩: 76.30
化学考试成绩: 90.00
英语考试成绩: 89.00
```

例 3 有内存变量如下:

```
.list memory
CD1      pub C      " 04/27/84 "
CD2      pub C      " 042784 "
RD1      pub D      04/27/84
ND1      pub N      42784          (42784.00000000)
```

使用命令改变 GET 变量的显示方式:

```
.SET DELIMITER TO '[ ]'
```

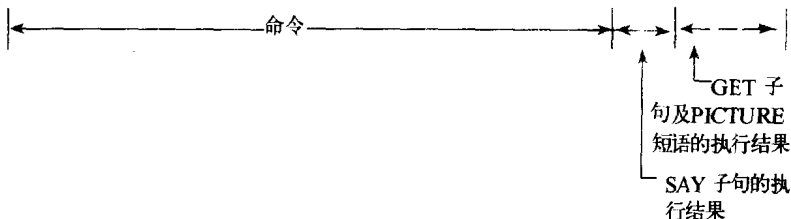
.SET DELIMITER ON

清屏幕:

.CLEAR

请看下面的命令及它们的执行结果:

```
.@ ROW ( ), 45 SAY ' 1 ' GET cd1 PICTURE "@ e " 1 [27/04/84]
.@ ROW ( ), 45 SAY ' 2 ' GET cd2 PICTURE "@ d " 2 [04/78/]
.@ ROW ( ), 45 SAY ' 3 ' GET cd2 PICTURE "@ rd " 3 [04/27/84]
.@ ROW ( ), 45 SAY ' 4 ' GET rd1 4 [04/27/84]
.@ ROW ( ), 45 SAY ' 5 ' GET rd1 PICTURE "@ d " 5 [04/27/84]
.@ ROW ( ), 45 SAY ' 6 ' GET rd1 PICTURE "@ e " 6 [27/04/84]
.@ ROW ( ), 45 SAY ' 7 ' GET nd1 PICTURE "@ d " 7 [ 4/27/84]
.@ ROW ( ), 45 SAY ' 8 ' GET nd1 PICTURE "@ rd " 8 [ 4/27/84]
.@ ROW ( ), 45 SAY ' 9 ' GET nd1 PICTURE "@ e " 9 [27/ 4/84]
.@ ROW ( ), 45 SAY ' 10 ' GET nd1 PICTURE "@ re " 10 [27/ 4/84]
```



命令: ACCEPT

格式: ACCEPT [<提示>] TO <内存变量>

说明: ACCEPT 要求一个键盘输入, 输入以回车结束。输入的内容被作为字符串存入给定的内存变量之中。

<提示>可以是一个字符型变量或一个字符串。若是后者, 应将其用单撇号、双撇号或方括号[]括起。

如果用户仅从键盘上键入了一个回车符, 内存变量的内容将是一个空字符(ASCII 0)。该命令最多能接收 254 个字符。

例如: 提示用户输入一个学生的姓名并将其存入内存变量 Mname 之中:

. ACCEPT "请输入学生的姓名:" TO Mname

屏幕上将显示:

请输入学生的姓名: