

怎样使用汉字编译dBASE III

怎样使用 汉字编译dBASE III

张延瑞 编著

312
YR/2

出版社

上海科学普及出版社

IBM-PC软件

怎样使用汉字编译dBASEIII

张延瑞 编著



上海科学普及出版社

0020341

内 容 提 要

本书内容包括：编译dBASE III的概念和主要优点，编译dBASE III命令、函数，简化编译dBASE III及编程技巧，原盘编译dBASE III的步骤，编译dBASE III与解释dBASE III的主要差别，管理信息子系统程序等。附录中包括：解释dBASE III和编译dBASE III的命令和函数表，解释dBASE III全屏幕操作命令，编译dBASE III不支持的解释dBASE III命令和函数表等。本书每章后均有小结和练习题，十分适于自学和培训。

读者对象：IBM-PC用户，程序员，大中专院校有关专业师生。

组 稿 杭州《电子与仪表技术》编辑部

责任编辑 胡名正 叶绍华

封面设计 毛增南

JS264/07

怎样使用汉字编译 dBASE III

张延瑞 编著

上海科学普及出版社出版

(上海曹杨路 500 号 邮政编码 200063)

新华书店上海发行所发行 上海市印刷七厂一分厂印刷

开本 787×1092 1/16 印张 9 字数 216000

1991 年 2 月第 1 版 1991 年 2 月第 1 次印刷

ISBN 7-5427-0313-7/TP·29 定价：5.50 元

目 录

第一章 编译dBASEⅡ概述	(1)
§ 1.1 编译dBASEⅡ的概念.....	(1)
§ 1.2 编译dBASEⅡ的主要优点.....	(1)
§ 1.3 简例.....	(2)
§ 1.4 小结.....	(3)
习题	(3)
第二章 编译dBASEⅡ命令、函数	(4)
§ 2.1 关于使用符号的约定.....	(4)
§ 2.2 编译dBASEⅡ命令.....	(4)
§ 2.3 编译dBASEⅡ函数.....	(14)
§ 2.4 小结.....	(19)
习题	(21)
第三章 简化编译dBASEⅡ及编程技巧	(22)
§ 3.1 简化编译 dBASEⅡ	(22)
§ 3.2 编程技巧.....	(24)
§ 3.3 小结	(34)
习题	(36)
第四章 编译dBASEⅡ与解释dBASEⅡ主要差别	(38)
§ 4.1 解释dBASEⅡ的运行过程.....	(38)
§ 4.2 编译dBASEⅡ的运行过程.....	(39)
§ 4.3 编译dBASEⅡ的功能扩展.....	(48)
§ 4.4 编译dBASEⅡ与解释dBASEⅡ命令、函数的差异.....	(51)
§ 4.5 小结	(53)
习题	(54)
第五章 原盘dBASEⅡ编译的步骤	(55)
§ 5.1 系统设置.....	(55)

§ 5.2	原盘上文件的内容	(55)
§ 5.3	拷贝保护	(56)
§ 5.4	编译过程的一般步骤	(57)
§ 5.5	编译过程的命令格式	(57)
§ 5.6	连接过程的命令格式	(58)
§ 5.7	DOS连接程序	(59)
§ 5.8	编译实例	(60)
§ 5.9	dBASE III 编译的隐含步骤	(65)
§ 5.10	覆盖文件	(67)
§ 5.11	调试程序	(69)
§ 5.12	dBASE III PLUS 编译程序	(71)
§ 5.13	小结	(86)
习题		(89)
第六章	管理信息子系统程序	(90)
§ 6.1	系统分析、设计、实施、评价	(90)
§ 6.2	简易工资管理子系统	(93)
§ 6.3	简化的财务管理子系统	(100)
§ 6.4	仓库管理子系统简介	(112)
§ 6.5	小结	(120)
习题		(121)
附录(一)	解释dBASE III 命令和函数表	(122)
附录(二)	编译dBASE III 命令和函数表	(131)
附录(三)	EDLIN(行编辑)命令	(134)
附录(四)	解释dBASE III 全屏幕操作命令	(135)
附录(五)	编译dBASE III 不支持(或不同方式支持)解释dBASE III 命令函数表	(137)

第一章 编译dBASE III概述

§ 1.1 编译dBASE III的概念

什么是编译dBASE III？这是近年来，在原有dBASE III（称为解释dBASE III）基础上，将一些功能加以扩展，发展成为另一种完整的系统，称为dBASE III编译系统。编译dBASE III为开发应用程序创造了有利条件。

通常dBASE III程序，是属于解释程序工作的范围。这种解释型dBASE III在运行时，针对每一行语句扫描，进行查错翻译。当遇到语句的语法错误时，就显示出错信息。这种迅速反馈的信息，对程序设计人员进行程序调试工作是比较直观易懂的，并能通过人机对话方式，纠正语法错误。但是，解释型dBASE III程序的运行速度较慢。因为每次运行程序需要对每一行源代码进行检错、翻译等重复工作。如当遇到语句LOOP时，又进行DO WHILE循环语句的重复操作。这样，解释程序要对循环体内相同的程序进行多次重复的检错及翻译，从而增加许多的运行时间。

编译dBASE III程序通常经过三步完成。第一步是对源程序进行编译，转化为机器指令的目标程序。第二步对目标程序加上运行管理程序及库函数，转化为可执行文件（扩展名为EXE）。第三步为运行文件。编译后程序的运行速度，比解释dBASE III程序的运行速度要快2~20倍。因为编译程序可避免多余的检错和翻译过程，根据源代码一次性生成一个压缩后的可执行文件。这个可执行文件，不需要任何其它软件的支持（包括dBASE III、编译程序等），可直接在操作系统（DOS或CCDOS）下运行。

编译程序为应用程序的编制者提供可靠的保密方法。

编译dBASE III对系统无特殊要求，可在IBM PC机及其兼容机长城机、浪潮机等上运行，仅需256K以上的内存容量。

用加密软件SUPERLOK加密的编译dBASE III原盘，具有拷贝保护能力，只能生成三个用户盘。

原盘上有一个计数器，每制作一个备份，计数器的数值减1。每回收一个备份，计数器加1，不需要的拷贝就可进行回收。

§ 1.2 编译dBASE III的主要优点

编译后的程序比解释dBASE III程序有明显的优点：

一、提高运行速度

经过编译后的程序运行速度要比解释dBASE III程序的运行速度快得多，一般可提高2~

20倍。因为编译dBASE III程序避免多余的检错和翻译过程。编译程序根据源程序代码生成压缩后的可执行.EXE文件,能在DOS(或CCDOS)下直接运行,不需要文件管理的开销,使可执行文件更有效,所占存储空间也少。而解释型dBASE III程序的运行速度较慢,在运行时无法控制不必要的错误检查过程,不能加速翻译过程。运行时需对每行源程序代码检错、翻译等重复工作。尤其遇到循环操作,对循环体内相同的程序进行多次重复的检错和翻译,从而增加运行时间。

二、保密性

编译dBASE III对源程序起到严格的保密作用。源程序经过编译后得到目标程序.EXE文件,无需加密,任何解密程序也不能破密而得到源程序。而解释dBASE III的程序虽经加密,但仍可破密得到源程序。因此,编译dBASE III为应用程序的编制者提供良好的保密条件。

三、功能扩充

编译dBASE III增加许多新的功能。如:

1. 含有64000个变量(解释dBASE III为256个变量)。
2. 每个数据库可拥有1024个字段(解释dBASE III为128个字段)。
3. 用户可自己定义函数。
4. 可同时打开20个文件(解释dBASE III可同时打开15个文件)。
5. 对某一个父文件可建立8个关系过程文件。
6. 备注字段可作为字符值存入内存变量。
7. 数值可用指数形式表示。
8. DO WHILE循环语句中可使用宏替换。
9. 能嵌套使用宏替换。
10. 按一定方式写成的HELP.PRG程序,当在程序运行时,按一下F1键,可调用和显示帮助信息。
11. 当出现程序错误时,ERROR.PRG程序能保持数据的完整,保留调试用信息并显示出来。
12. 可调用外部程序。

§ 1.3 简 例

例: 设 $A=1$, $B=2$, $C=3$, 计算 $D=(A+B)/C$, 用编译程序编译、连接、输出结果。

首先用dBASE III下的MODIFY COMMAND或DOS(或CCDOS)下的EDLIN命令建立文件。设在硬盘驱动器上进行并打印出程序如下(具体步骤请参阅解释dBASE III):

```
C>TYPE Z.PRG
```

```
A = 1
```

```
B = 2
```

```
C = 3
```

D = (A + B) / C

WAIT

? D

RETURN

其次进行编译、连接、运行过程如下：

C>DB3C Z (编译)

屏幕显示：C—DBASEⅡ 编译程序VER. 1.0

COMPILING Z.PRG

LINE1 LINE2 LINE3 LINE4 LINE5 LINE6 LINE7

CONSTRUCTING OBJECT FILE...Z.PRG编译完成

DBIII/COMPILER 1.0a COMPILATION COMPLETE

C>DB3L Z (连接)

屏幕显示：C—DBASEⅡ 连接程序VER. 1.0

PHASE I

LINKING@Z.PRG

PHASE II

LINKING LIBRARY; DB3PC.LIB

C>Z (运行)

屏幕显示：1 (表示输出结果)

以上是用简化编译dBASEⅡ进行的，操作命令简单，仅需一张系统盘片和一张工作盘片。如果都在C盘上(硬盘)进行，就不需要任何盘片。最后把得到的Z.EXE文件(包括浮动文件)复制到软盘上，就不需要编译dBASEⅡ的系统盘片，也不必经过上述编译、连接步骤，可直接放在A驱动器上，键入文件名Z即可(在DOS或CCDOS操作系统下)。

即：A>Z↵

显示：1 (结果)。

(注：“↵”表示回车。)

§ 1.4 小 结

编译dBASEⅡ是在解释dBASEⅡ基础上发展起来的。它本身是一个完整的系统。

编译dBASEⅡ比解释dBASEⅡ具有运行速度快、保密性强、功能扩展的优点。

在解释dBASEⅡ的MODIFY COMMAND命令或DOS(或CCDOS)操作系统下用EDLIN编辑命令或WORDSTAR字处理建立程序，经过编译、连接后，得到可执行.EXE文件，不需要别的软件支持，直接可在计算机上运行。

习题

1. 编译dBASEⅡ比解释dBASEⅡ有哪些优点？
2. 编译dBASEⅡ的编译、连接过程是怎样的？

第二章 编译dBASEⅢ命令、函数

本章讨论编译dBASEⅢ的主要命令和函数。这些命令和函数与解释 dBASEⅢ有所不同或补充。

由于编译程序与解释程序的工作原理不同，两者的差别也是自然存在的。

命令和函数以格式、差异（与解释dBASEⅢ比较）、说明、例子的形式分别叙述如下。

§ 2.1 关于使用符号的约定

在命令和函数中使用的符号，需填入各种信息，或可选择的信息等，在本节作统一的约定。

一、[]（方括号）：表示命令的可选部分。方括号本身不要键入。

二、（）（圆括号）：表示必须键入的信息。如果在方括号中有圆括号，则这些信息是可选择项。圆括号本身也需要键入。

三、<>（尖括号）：表示必须提供的信息。如数据库名，字段名，内存变量名，文件名等。尖括号本身不需要键入。

四、^：表示键盘上的CTRL键（控制键）。操作时先按下控制键不放手，再按下其它的键。如^W说明先按下控制键不放手，同时再按下W键。

五、...：表示这中间有其它的语句。

如 DO WHILE...

...

DO WHILE...

...

ENDDO

ENDDO

ROUND (<expN>, <decimals>)

在说明一个命令或函数时，大写字母的词（如上述ROUND）是不可省略的键入信息。该函数表示舍入。小写字母的信息需要填入的信息。如<expN>和<decimals>表示表达式N和小数位数。其间用逗号分开。同时圆括号的信息是必需的。总的意思表示ROUND函数根据指定的小数位数进行舍入。

当然，小写字母的信息可完全用中文来表示，显得更清楚。

§ 2.2 编译dBASEⅢ命令

一、@命令

1. 格式：

@<行号, 列号>[SAY<表达式>][PICTURE<子句>]]

[GET<表达式>][PICTURE<子句>]]

[RANGE<表达式, 表达式>]

[VALID<表达式>]

[CLEAR]

2. 差异:

除可选项PICTURE和RANGE外, 并可选用VALID项。这项表示用户确认键入的正确性。当表达式的值为真, 或SET ESCAPE为ON时按ESC键, 才完成GET命令项。

3. 说明:

如没指定行和列的坐标值, 则不返回出错信息。

4. 例:

W = 0

@5, 5 SAY "W?" GET W VALID W>0 (W取得大于零的值)

二、@...BOX命令

1. 格式:

@<顶, 左, 底, 右>BOX<字符串>

2. 说明:

这是编译程序所具有的命令。

@...BOX命令用于在屏幕上画方框。

其中顶, 左, 底, 右为四个坐标, 数值范围分别为:

顶: 0~24

左: 0~79

底: 0~24

右: 0~79

框的大小和相对位置由键入的四个坐标决定。

<字符串>表示用8个字符代表4个角和4个边。如指定第9个字符, 用这个字符填充方框字符串中指定的字符。从左上角开始, 按顺时针方向显示。

3. 例:

C = CHR(201) + CHR(205) + CHR(187) + CHR(186) + CHR(188)
+ CHR(204) + CHR(200) + CHR(186) + CHR(176)

@1, 0, 24, 79 BOX C

表示每一个角和边用不同的字符画一方框, 并且用一个不同的字符CHR(176)填充该框。

如表示清除方框, 用空字符串:

@1, 0, 24, 79 BOX " "

其中1, 0, 24, 79分别表示

1~24行, 0~79列。

三、APPEND FROM

1. 格式:

APPEND[范围][<字段表>]FROM<文件名>[FOR/WHILE<条件>][SDF/DELIMITED]

2. 差异说明:

APPEND FROM可用字段表作为变量。当给定了字段表, 增加的数据就填入该字段中, 并且能增加记录的范围。

3. 例:

如要增加一个字段表, 命令形式为:

APPEND W, X, Y FROM AA FOR X=2

如使用范围指定一个记录

APPEND RECORD 4 FROM BB

四、AVERAGE

1. 格式:

AVERAGE<字段表>TO<内存变量表>[FOR/WHILE<条件>]

2. 差异说明:

其中的字段表和内存变量表不能省略的。

3. 例:

AVERAGE N1, N2 TO T1, T2 FOR UPPER FIELD NAME=YY

五、CALL

1. 格式: CALL<过程>[WITH<参数表>]

2. 差异: 调用编译或汇编过的程序。解释dBASE III无此命令。

3. 说明: 程序可用任意的编译语言编写, 如C语言、汇编语言等。被调用程序是“INTEL 8086可重定位的目标文件形式”。扩展名为·OBJ。必须符合C语言的调用和参数传递规则。

当调用别的高级语言写的程序时, 连接程序中需含有编译的程序库。

CALL命令的参数表的参数不能超过7个。被调用程序要定义为远(FAR)过程, 意指用远返回结束。

数据引用由四字节形式为段地址, 偏移量的指针组成, 依次压入堆栈。数据类型是引用传送, 在程序中保存BP, SS和DS寄存器。

字符串通过引用传送, 以空字节结束(字符串结尾是一个零字节)。

因数据区含有内存中顺序存放的数据项, 需要保存数据项的长度。如数据项太长, 可能写到别的数据中。

数字变量以 8 个字节浮点形式传送。由 53 位对数首数和偏加量为 1023 的 11 位指数组成。整数可传送到汇编程序中。先送至 C 语言程序内作为整数，或把整数作为 ASCII 字符传送，并转换成汇编一级的整数（每个数字减去十六进制的 30，产生等价的十进制数）。

4. 例：

例 1：

A = "123"

CALL TEST WITH A "ABC"

? A

RETURN

表示要调用一个名为 TEST 的 C 语言程序。

所调用的名字为 TEST 的 C 语言程序如下：

TEST (P1, P2)

CHAR * P1

CHAR * P2

WHILE (* P2)

* P1 ++ = * P2 ++

* 注意：按大模型编译 | * |。

例 2：

A = "123"

CALL TEST WITH A, ABC

? A

RETURN

表示调用产生相同结果的汇编语言程序。

（名字也是 TEST）

编译程序通过编译成等价的代码，产生下列参数传送：

栈地址	栈	命令
SP + 10	[SEGMENT OF ABC]	PUSH
SP + 8	[OFF SET OF ABC]	PUSH
SP + 6	[SEGMENT OF A]	PUSH
SP + 4	[OFF SET OF A]	PUSH
SP	[SEG OF RETURN ADDRESS]	CALL FAR TEST

于是汇编语言例行程序如下：

PUBLIC TEST, * 过程名

- PROG SEGMENT BYTE, * 编译代码段

ASSUME CS - PROG, * 放代码处

TEST PROG FAR, * 一个远过程

PUSH BP, * 栈 + 2

MOV BP, SP, * 取指针

```

PUSH DS
PUSH ES
CLD
LDS SI, DWORD PTR [BP = 10],
    "ABC"的段地址和偏移量,
LES DI, DWORD PTR [BP = 61],
    "123"(A)的段地址和偏移量
MOV CX, 3
REP MOVSB
POP ES
POP DS
POP BP
RET
TEST ENDP, * 过程结束
- PROG ENDS, * 段结束
END, * 程序结束

```

六、COUNT

1. 格式:

COUNT[<范围>][FOR/WHILE<条件>]TO<内存变量>

2. 差异说明:

需要有存放总计的内存变量。

3. 例:

USE N (打开库文件N)

COUNT TO M (统计库文件记录数)

? M (显示库文件记录数)

表示统计数据库N的记录个数。如总的记录个数为20, 则最后显示结果为20。

七、CREATE

1. 格式:

CREATE< .DBF文件名>

2. 差异:

形式与解释 dBASE III 相同, 但在编译程序中是建立一个有零个记录的结构扩充文件。

3. 说明:

上述命令和CREATE FROM联用能生成新的数据库。

4. 例:

CREATE S (建立零个记录的结构扩充文件S)

DO WHILE · T · (循环语句)
CLEAR (清屏)
APPEND BLANK (在数据库末尾添加空记录)
@3, 0 SAY"D1" GET D1 (字段名)
@4, 0 SAY"D2" GET D2 (字段类型)
@5, 0 SAY"D3" GET D3 (字段长度)
@6, 0 SAY"D4" GET D4 (字段小数)
READ (读入变量)
IF LEN (TRIM(D1)) = 0 (测试D1长度)
DELETE (暂时性删除)
PACK (永久性删除)
USE (关闭数据库)
EXIT (退出循环)
ENDIF (条件语句结束)
ENDDO (循环结束)
CREATE SS FROM S (生成新的数据库SS)

八、DISPLAY

1. 格式:

DISPLAY[OFF][范围]<字段表>[FOR/WHILE<条件>]

2. 差异说明:

字段表不能省略。字段名不能用于标题显示。

3. 例:

DISPLAY N1, N2 FOR N = 100

九、LIST

1. 格式:

LIST[OFF][范围]<字段表>
[FOR/WHILE<条件>]

2. 差异说明:

字段表不能省略。字段名不能用于标题显示。

3. 例:

LIST N1, N2 FOR N = 100

十、SUM

1. 格式:

SUM<字段表>TO<内存变量>[FOR/WHILE<条件>]

2. 差异说明:

每个字段表需含有相同的元素个数。内存变量和字段表不能省略。

在结构上，与相应解释 dBASE II 中相应命令不同。

3. 例:

```
SUM S1, S2 TO G
```

```
? G
```

十一、TYPE

1. 格式:

```
TYPE<文件名>[<文件名>/.PRN]
```

2. 差异说明:

表示用DOS的重新定向性能“>”，把TYPE命令的输出文件送到磁盘文件或打印机上。

通常，数据在屏幕上显示，如指定DOS的重新定向特性，数据被重新定向改在打印机上输出。

3. 例:

```
TYPE AA.PRG>PRN
```

表示把AA文件重新定向。这里使用DOS的重新定向性能“>”。

十二、FOR...NEXT

1. 格式:

```
FOR<内存变量>=<数字表达式1>TO<数字表达式2>[STEP<数字表达式3>]
```

```
<命令串>...
```

```
NEXT
```

2. 差异:

解释dBASE II 无此命令。

3. 说明:

内存变量和数字表达式需要给定。

STEP数字表达式（增量或步长）可正或负。

表示完成一个循环操作。可选择增加或减少循环中表达式的值。

4. 例:

```
FOR I=2 TO 10 STEP 2
```

```
? I
```

```
NEXT
```

执行结果显示为2,4,6,8,10。

如增量为负值时

```
FOR I=8 TO 2 STEP -2
```

```
? I
```

```
NEXT
```

显示的结果为 8 6 4 2。

十三、RESTORE SCREEN

1. 格式:

2. 差异:

解释dBASE III 无此命令。

3. 说明:

这命令和SAVE SCREEN 联用, 避免重画临时被HELP . PRG或其它命令置换掉原屏幕内容。

十四、SAVE SCREEN

1. 格式:

SAVE SCREEN

2. 差异:

解释dBASE III 无此命令。

3. 说明:

这命令和RESTORE SCREEN联用, 避免重画临时被HELP . PRG或其它命令置换掉原屏幕内容。

十五、FUNCTION

1. 格式:

FUNCTION<名字>...RETURN(< >)

2. 差异:

解释dBASE III 无此命令。

3. 说明:

表示用户自定义函数。像过程一样能传送参数。

用于任意类型的表达式、赋值语句和@...SAY等语句。

4. 例:

DO S (执行过程S)

RETURN (返回语句)

PROCEDURE S (过程S)

ACCEPT "请输入字符串值:" TO F

(把一个字符串值送到内存变量F中。)

CLEAR (清屏)

@12, 0 T(F) SAY F (调用自定义函数F)

RETURN (返回语句)

这里调用自定义函数T且传送参数F。

FUNCTION T (自定义函数T)

PARAMETERS R (参数传送)

P = LEN(R) (测试字符串R长度)

IF P < 80 (条件语句)


```

P = 1
ELSE
P = 0
ENDIF
RETURN(P) ( 返回 )

```

十六、SET FUNCTION

1. 格式:
SET FUNCTION<数字表达式>TO<字符串>
2. 差异:
SET FUNCTION 1与HELP.PRG相联系且不能改变的。
3. 说明:
用这条命令可设置功能键 2~40。
通过按下相应的功能键F1~F10表示功能键 2~10。
通过按下SHIFT键和相应的功能键F1~F10表示功能键11~20。
通过按下CTRL键和相应的功能键F1~F10表示功能键21~30。
通过按下ALT键和相应的功能键F1~F10表示功能键31~40。

十七、PARAMETERS

1. 格式:
PARAMETERS<内存变量表>
2. 差异说明:
可以调用程序传送数据项。此外, 能从DOS命令行将一个字符串送到程序中。
字符串是文件名中非空字符开始的, 每一个过程或函数仅有一个PARAMETER 语句。
如传送的参数是内存变量, 则是引用传送而不是数据传送。同时可修改PARAMETER, 以改变传送的数据。

```

3. 例:
C>MYPROG sample
表示在DOS命令行上, 文件名是在程序名后输入的。
程序名为MYPROG.PRG。
PARAMETERS file
file UPPER(TRIM(file)) + ".DBF"
IF NOT FILE("& file")
? file + "没找到"
RETURN
ENDIF
USE & file
RESTORE SCREEN

```