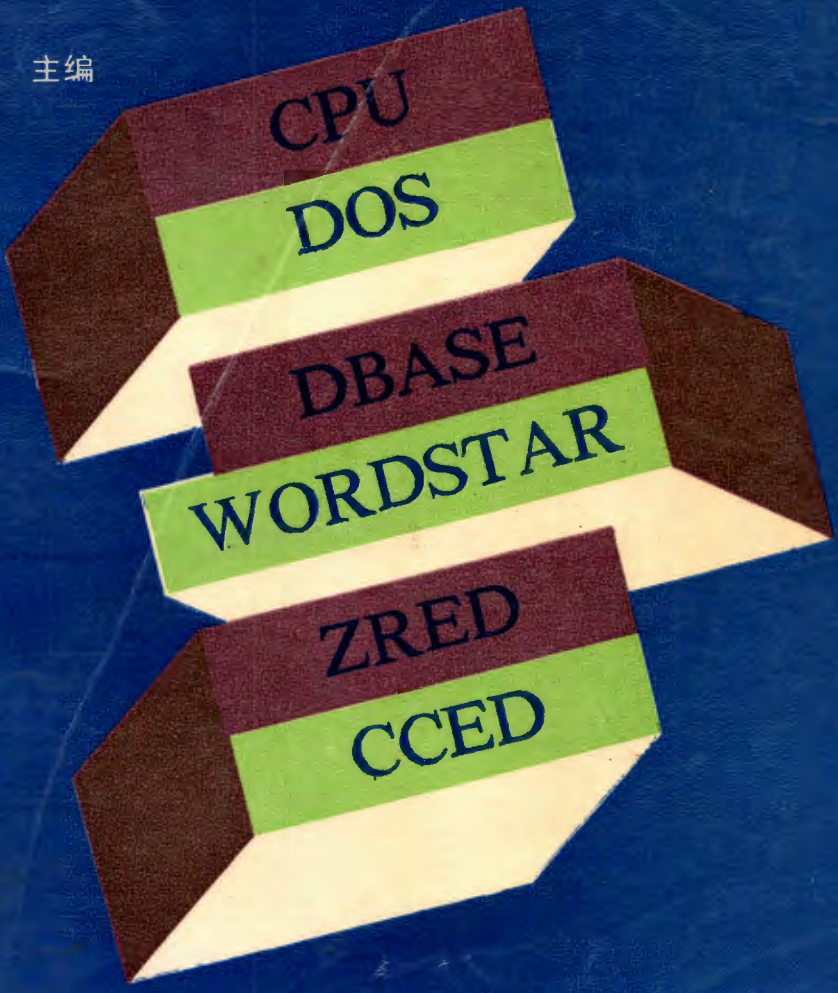
微机应用基础

——初学者上机必备

牛树长　主编



北京理工大学出版社

责任编辑：倪福卿
封面设计：何　欣

本书被评为'94第六批
全国优秀科技畅销书

ISBN 7—81013—712—3/TP·78
定价：7.45　元

微机应用基础

——初学者上机必备

牛树长　主编

北京理工大学出版社

（京）新登字 149 号

内 容 简 介

当今，微型计算机的应用已渗透到国民经济各个部门。本书旨在帮助初学者掌握微机的有关基础知识和使用技能，并用于解决实际问题。

全书共二十一章，内容包括计算机的基本概念，DOS 操作系统，DBASE Ⅲ 的基础知识，ZRED 和 CCED 文字编辑软件使用及自然码汉字输入方式等。书中有较多的实用例题和上机操作示范，可供初学者练习。

本书适合微机初学者、操作员及管理人员阅读，也可作为有关培训班的教材。

微机应用基础

——初学者上机必备

牛树长　主编

*

北京理工大学出版社出版发行

各地新华书店经售

一二〇一印刷厂印刷

*

850×1168 毫米　32 开本　10 印张　256 千字

1993 年 5 月第一版　1994 年 8 月第四次印刷

ISBN7－81013－712－3/TP・78

印数：35201—46200 册　定价：7.45 元

前　言

近年来，计算机技术特别是微型计算机技术，已得到广泛的发展和应用，计算机的使用实际上已渗透到国民经济的各个部门，已成为衡量一个部门技术水平的重要标志。

帮助初学者掌握微型计算机的有关知识和使用技能，并用来解决本单位、本部门的实际问题是本书编写的宗旨。在本书的编写中，力图内容通俗易懂，深入浅出，初学者只要一步一步地照着做就能学会微型计算机的基本使用方法. 而且对已经基本掌握计算机操作的读者也有一定的帮助。

全书共分二十一章：第一章介绍计算机的基本概念；第二至十章介绍 DOS 操作系统，详细地讲述操作系统内、外部命令，目录，批处理及管道功能；第十一至十七章系统地讲解了 DBASEIII 的基础知识，函数的概念，数据库的建立，数据的录入，数据库的基本操作，多工作区的操作及简单的编程，并且有一些简单的实验和例题便于初学者加深理解。第十八至二十一章介绍了一些实用的编辑软件以提高初学者用计算机处理问题的能力。

全书内容很丰富，在学完这些内容后不仅可以解决实际的应用问题，而且可以为今后的学习和进一步的提高打下基础。

本书也可以做为计算机学习班的培训教材使用。

全书由牛树长主编，参加本书编写工作的还有杜宪忠、韩焕、张新义、张庆前、张苏。在本书的编写过程中得到了北京铁路局人事处、北京铁路工程公司人事科和教育科的大力支持，在此表示感谢。

由于我们水平有限，错误和不当之处一定难免，希望广大读者指正。

1993 年 1 月

目　录

第一章　计算机的基本概念

目前计算机已经相当普及，作为中青年管理干部来说，了解计算机的知识、掌握计算机常用操作的基本技能不仅是非常必要的，应当说是事在必行。今后若不懂计算机操作就很难适应管理工作的需要。

1—1　计算机的常识

自 1946 年第一台计算机问世后的几十年，计算机经历了电子管、晶体管、小规模集成电路、大规模集成电路的过程，到目前已经发展到超大规模集成电路的时代。就其体积从整间的楼房缩小到桌式、台式甚至是膝式；就其价格从数千万美元降至数百美元；就其使用范围从国防科研渗向社会的各个层次和领域，现在已经普及到一般家庭和儿童。计算机的诞生和发展为人类生产力的变革起到了划时代的作用。

计算机已经成了人类社会的一大支柱，成了人们所不可缺少的基本特殊工具。在社会的军事、科研、生产、管理等领域，计算机正在发挥着越来越重要的作用。

计算机虽然能够适应如此广泛和复杂的使用环境，但其内部却只有 0、1 两个基本状态。这就是所谓的“二进制”计数方式。二进制有它独特的运算规律和运算方法。“逢二进一”就是其基本的特征之一。

例如：1＋1＝10　　11＋10＝101

1＊1＝1　　11＊10＝110

在计算机内部，所有运算都是用二进制运算完成，所有的控制指令也是用二进制表示。我们输入的十进制数据或字符数据都要经过转换，变为二进制的数据形式后才能被计算机处理，再通过相反的转换输出计算结果。

在二进制数中，每八个二进制位的长度叫做一个字节，更大的换算单位是 K 和 M 。换算关系如下：

1024 字节＝1K

1024K ＝1M

在计算机数据和设备的表示方法中，这些单位多被用来说明文件的大小或设备的容量。如

文件为 312 字节　　（表示文件的大小）

内存为 1M　　（设备容量）

360K 软盘　　（软盘容量）

认识这种表示方法，是了解和使用计算机的常识。很多计算机设备的指标和软件的运行环境都是用这些方法表示的。

1—2　计算机的组成

计算机作为一个完整的系统，由不同功能的专用电路板配合适当的外部设备组装而成。这些具备不同属性的物理构件统称计算机硬件。主要有以下几种：

1. 中央处理器和辅助电路　中央处理器又称 CPU，它是一块大规模集成电路芯片。如 8088，8086，80286，80386 等等。在处理器内包括运算器、控制器、标志器、寄存器、内部数据总线、内部控制总线等等。它是计算机的心脏，它只能处理 0 或 1 的两个状态的计算，只能完成由机器码定义的控制指令。

只有处理器是不够的，它必须在很多外围电路的支持下才能充分发挥作用和效率。计算机的主板基本包括了处理器的工作环境。为了解决主板与外部设备的配合，就要有各种专用电路配合

卡，每种卡都包括了一种或数种外部接口的规范。

2. 内存储器 大量的控制指令、原始数据要输入CPU进行计算，计算的中间结果和最终结果也要进行存储。为了与高速的CPU适应，就要使用高速的存储器，这部分存储器就是所谓的内存。目前微型计算机的内存一般在512K到2M甚至更多，但DOS操作系统能够指挥的内存空间仅为640K。内存一般在计算机主板上，因为这是一个非常有用的概念，必须特别强调。

3. 外部存储设备 内存中的数据只能暂存，断电后即全部丢失。为了能够使数据或指令文件能够长期保存并能够随时恢复或应用，就要借助于外部存储设备；保留和恢复数据的外部设备，有磁带机，磁盘机（又称驱动器）、光盘等，是计算机的重要组成部分。

磁盘有两大类，一种固定式的叫硬盘，它的存贮量在10M至200M甚至更多，反应速度快。另一种是软盘驱动器，它可更换磁盘，交换数据方便，可与硬盘交换数据。软盘驱动器把数据写到磁盘介质上，读写密度有区别，5.25英寸磁盘有360K和1.2M两种，3.5英寸磁盘有720K和1.44M两种。高密度驱动器可兼容低密度磁盘。低密度磁盘不能代替高密度磁盘使用。

在配备两个驱动器的计算机上，用设备符号A和B区分，分上下安装，启动指定设备时，相应的指示灯发光。可用这个现象分辨出驱动器的设备符号是A还是B。

4. 键盘和显示器 计算机的内部工作必须按照人们的要求进行，我们对计算机的干预是通过键盘和显示器进行的。是键盘和显示器沟通了操作员和处理器的联系。

显示器种类很多，在方式上有单色和彩色之分，在分辨率上有高、中、低之分，在配套的适配卡支持下才能工作。在管理应用上，用户关心的是屏幕实际显示的行数。早期的计算机以彩色、中分10行为主，现在多为彩显、高分25行。作为管理用，后者更实用。

5. 打印机 把计算机内存或磁盘的数据、信息印刷成文字文件的设备。

目前打印机发展很快，创新的机型层出不穷。就一般的办公使用，仍是以针式打印机为主。

针式打印机有自带字库的，如 AR3240、LQ1600K 、OKI 系列等多种，这类打印机有自己的驱动程序，可直接支持计算机的打印输出。还有一类不带字库的打印机，如 LM1724 、LQ1000 、NEC2024 等多种，它们依赖于计算机内部的字库和管理模式，计算机在启用这种打印机前必须要运行相应的支持程序，否则打印机就不能输出汉字。在实用中要注意到这一步骤。

以上这些由自身物理结构确定功能的设备统称硬件。为了区别运算器和内存，把驱动器、键盘、显示器、打印机等系统配制统称外部设备（简称外设）。这些提法是经常用到的。

1—3 计算机软件

1. 系统软件 操作系统是最基本的系统软件之一，它是计算机自身资源的管理系统。它可以合理的组织和协调内存与外设之间数据联络，为各外设安排工作环境，调动支持各外设完成指定的任务。

系统软件是给用户提供最基本的操作手段的软件，是应用计算机的基础，用户可通过操作系统规定的命令指挥和操作计算机的外部设备。微机上应用最广泛的操作系统是 DOS 操作系统。由于我们使用的汉字也需要相应的系统软件支持，一般在原有的操作系统前提下扩充汉字管理功能，这被叫做操作系统的汉化。如 UCDOS 、CCDOS 、2.13 等均属于汉字操作系统。它们应属于系统软件的范畴。

另一种系统软件是计算机语言系统。该系统能够把高级语言编写的程序翻译成机器语言。

一般的计算机语言都有完整的功能描述和具体的运算规定，是人们控制计算机运行的一种约定形式。这些语言是用户开发应

用程序所依赖的基础和工具。

2. 应用软件 应用软件是用户完成数据处理任务的程序或系统。一般是在系统软件的基础上进一步针对具体问题开发的实用系统，它具有使用方便和处理问题单一的特点。

所谓程序开发是指通过一种或几种计算机语言把人对计算机的要求编写成计算机指令序列，计算机在执行这些指令后，产生编程人员预期的结果，从而达到数据处理的目的。

由计算机操作员接触的程序系统基本都属于应用软件。

操作系统、语言系统和应用程序都是属于软件的范畴。由于人们思维形式的不同，对计算机的工作和要求亦不相同。要对计算机表达出这种要求，就要通过某种计算机指令，这些由人所赋予计算机的指令集合，就是所谓的软件。

硬件一经指定不会依赖于人的思维改变，而软件能。硬件是一种物质的存在，软件则是思维的表述形式。

1—4 计算机语言的发展过程

人们对计算机的控制是经历了一个由复杂到简便的过程。

1. 机器码 也称机器语言，是由计算机处理器可直接识别的机器指令，这是最原始的计算机语言，只能用 0、1 两个代码书写指令或对计算机提供数据。这种由机器码书写的程序执行速度最快、效率最高。但要把复杂的计算任务翻译成计算机机器码却非一般人所能胜任的工作。这样做不但非常枯燥和繁琐，出现问题时处理也很困难。由此造成软件开发周期长，维护极为困难，以至不能被一般人接受。

2. 汇编语言 为了加速软件的开发，人们规定了一些与机器语言相对应的代码。用代码书写计算机指令要比机器码简单，通过计算机可把这些代码翻译成机器语言。这种带有帮助记忆的符号被称为助记符，由助记符组成的集合叫汇编语言。它是与机器码最

接近的低级语言。使用汇编语言仍然要求对处理器的内部状态非常熟悉，学习起来要有相当的专业基础，除非是十分必要，一般人很少用它做实用程序。

3. 高级语言 随着计算机应用范围的扩大，要求不懂计算机内部结构的人同样能控制计算机完成指定的任务，高级计算机语言就诞生了。

高级语言在处理问题的描述上完全脱离了机器语言，开始与人们习惯的思维表达方式接近。编程人员不需知道计算机是怎样完成任务的，只要把任务按照指定的语言规范正确描述，就能够达到控制计算机运行的目的。

目前应用较广的语言有 BASIC 、COBOL 、FORTRAN 、PASICAL 等等。C 语言是介乎于汇编语言与高级语言之间的一种桥梁式语言，它兼容双方的特点，也有很多用户。

4. 工具语言 计算机从开始的数据处理扩展到管理后，与人联系角度也发生了相应的变化。尽管高级语言较以前有了很大的进步，应用于管理仍嫌其太复杂。为此，适应管理的工具型语言已被广泛的接受并迅速的发展。工具语言的应用方法进一步简化，与习惯思维的表达方式更接近。目前流行的 DBASE 、FOXBASE 等均属于工具类语言。

工具语言以其功能完整及易于掌握见长，能够较好适应从事管理工作人员的知识程度，即使不经过专业性的学习，也能应付一般管理工作的需要，只是速度上有些相形见拙。

1—5 计算机使用和保养

正确使用和保护计算机是对计算机操作员的基本要求，是保证设备和数据安全的实际需要。日常上机应注意以下几个问题：

1. 开关机 开机一般应遵循先外后内的顺序，即

电源设备→显示器→打印机→主机

如果有不间断电源和调压器，应满足电源设备的预热时间。

关机顺序相反，即

主机→打印机→显示器→电源设备

如果遇到死机，则需关机处理，此时切莫即时通断电源，这对硬盘非常不利，一般要等一、两分钟，再重新开机。

2. 防护 计算机中的软、硬驱动器都是很精密的，灰尘的侵入会给这些设备带来威胁。保持环境的清洁是必要的，不要在键盘上方吃东西，在关机之后应盖好防护罩。

3. 软盘的使用

软盘是数据的存放介质，对防尘、防磁、防潮、防硬性损伤等方面都要特别小心。由于对软盘的保护不当造成数据丢失将是后悔莫及。所以对于存有重要数据的软盘，既要多备份，又要妥善保存。

4. 不间断电源

不间断电源不仅是能源的后备，也是净化能源环境的工具。电源的污染会干扰计算机的正常工作，突然断电会使计算机内存数据全部丢失。对于有较高的数据要求或供电缺乏保证的情况，不间断电源是计算机的必要后援设备。

5. 病毒 计算机病毒的流行对计算机的工作和数据造成巨大威胁，每个操作员都要引起高度重视。防治计算机病毒的措施有：

(1)行政隔离，这是最有效的办法。对外来数据必须经过病毒检测，确认无病毒后方可使用。

(2)禁止非本机操作员使用计算机。禁止他人的软盘介入。

(3)用常驻内存的防病毒软件或使用防病毒硬件可在一定程度上减少病毒的侵害。

(4)发现计算机工作失常，要进行病毒检测，切不要盲目的变更文件，防止病毒危害的加剧。

(5)确认病毒存在后，要找专业人员协助处理。

1—6 关于操作系统的工具软件

目前流行一些操作系统的辅助软件，如 PCTOOLS 和 PCSHELL 等几个级别的版本。它们的共同特点是通过菜单选择的方式完成磁盘的有关操作。

这些工具都有详尽的使用说明，它们可对 DOS 功能做些有益的补充，但操作并不一定简单。这里仅对用户关心的文件恢复操作作简单的介绍。

误删除了有用的文件的事情时有发生，对于 DOS 本身，文件是不能被恢复的了，但借助于工具软件就有恢复的可能。

在删除文件的磁盘空间尚未被其它文件覆盖的情况下，文件可被完全恢复。否则恢复就很困难甚至不可能。

以 PCSHELL 为例：

用 DOS 的外部命令方式可启动 PCSHELL 工具，若 PCSHELL 存在于当前驱动器 C 的当前路径，则可输入

C>PCSHELL ↵

即可进入操作工具软件的菜单，此时屏幕分左右两部分，左侧显示目录结构，右侧显示文件名。

恢复工作的步骤是

(1)用 CTRL＋驱动器符，使操作指向所需驱动器。

(2)用 TAB 键可切换当前操作到目录部分，即白色双框在目录一侧。

(3)用箭头键移动光标到文件所在的目录项。

(4)顺序按 S、U、C、F 键(屏幕显示菜单从略)，屏幕出现被 DOS 删除的文件清单。

(5)文件名的第一位是?，表示此文件被 DOS 删除。如果文件名的最后有“@”符号，表示该文件被部分覆盖或全部覆盖。如果没有，文件是可完全恢复的。对于可完全恢复的文件，可按下列步

骤操作：

用箭头将光标调到欲恢复的文件后按 G 键，屏幕显示该文件的有关信息，光标停在文件名的?处，可输入一个重新命名的字符，然后回车两次，指定的文件可被恢复。

(6)一个文件被恢复后，返回到文件名清单，如继续恢复文件，可重复(5)的操作，否则按 ESC 键并回车可退出 PCSHELL 。

第二章　DOS 键盘说明

2—1　专用键

(1)ENTER 回车键:按此键表示该行命令输入结束,请求处理程序进行处理。在编辑中也用来结束一个输入行。

(2)ESC "ESCAPE"键:作废当前命令行,在屏幕上保留文字样板并在后面显示"\",光标下移一行。可开始构造一个新行或对原样板进行编辑。

(3)TAB→←键:制表定位键,只占一个文件字节长,但被 DOS 解释成八字节长度的空格。

(4)CTRL ("CONTROL"):控制键,该键总是与其它键组合后形成各种控制指令,详见后面的说明。

(5)SHIFT:字符换档键,一般有左右两个,它可控制字母键的大小写和所有双字符键的切换。按住此键后再按字母键,可切换翻转当前字母的大小写;如果按双字符键,则可输入上字符,否则是下字符。

(6)ALT ("ALTERNATE")键:与其它键组合使用,形成不同的辅助控制命令,详见后面的说明。

(7)NUM LOCK 键:该键为小键盘的状态控制键和锁定键,可完成编辑键盘和数字键盘翻转和锁定。按一次可锁定数字(0,1,2,…,9)状态,再按就是编辑键(→,←,↑,↓,行,页)状态。可往复使用。

(8)CAPS LOCK:字母大小写往复锁定切换键。按下此键一次,字母键成大写,再按就翻转到小写。

(9)屏幕打印键在101 键盘上只需按一个键 PRINT SCR -

EEN，在 IBM 键盘上需用两个键，PRTSC 键与↑键（或 * 键）一同按下，屏幕信息将在打印机上印出。

(10)功能键：F1，F2，…，F10，F11，F12，101型键盘上有12个功能键，IBM 普通键盘上功能键仅有10个，在 DOS 中定义了前5个功能键，其功能如下：

F1：从样板行中逐个复制字符，按一次复制一个。前一个命令行或用 ESC 键结束的行都属于 DOS 的样板行。

F2：从样板行中复制到指定字符。先按一下 F2键，再按某个字符键，样板行被复制到这个指定的字符为止。

F3：复制样板行所有剩余的字符。

F4：从样板行中删除到指定字符。先按一下 F4键，再按某个字符键，样板行被删除到这个指定的字符为止。这正好与 F2键形成对照。

F5：改变屏幕当前行的命令定义，接受该行为样板行，以备进一步编辑。

以上论述是以 DOS 屏幕上的当前行和当前字符而言的，配合功能键的编辑，还有 INS 和 DEL 键。

(11) INS：插入键，将按此键后到启动功能键前输入的字符插入到编辑行中。

(12) DEL：删除键，每按此键一次，删除样板行当前字符后的一个字符。

注意：DOS 的功能键在不同的实用程序中可赋予不同的功能定义。如果其它软件改变了功能键的定义，具体使用方法可查阅相应软件的操作说明。

2—2 复合控制键

复合键是指按控制键(CTRL 或 ALT) 不放开，再按指定键。

(1)CTRL＋BREAK 可终止当前运行的命令或程序。

(2)CTRL+ENTER 启动当前行的继续行，结束本书写行但不结束命令行的输入，一般在命令行可能超过屏幕范围时才使用，回车键将会自动连接所有继续行。

(3)CTRL+NUMLOCK 使系统暂停到按下某个键再恢复运行。多用于控制屏幕信息的输出速度，以便能够阅览。

(4)CTRL+P 打印机通断切换键，启动一次连接屏幕到打印机，再次启动中断。联机设置成功后的所有屏幕信息被拷贝到打印机印刷。这对记录系统操作过程和系统出错信息非常有用。此时系统会因同步打印机而降低运行速度。

(5)CTRL+S 屏幕显示暂停往复控制键，第一次按下屏幕显示暂停，再按继续。

(6)CTRL+ALT+DEL(三键同时按下)，在系统已加电的情况下重新启动 DOS 系统，即热启动。

(7)ALT+ 功能键，一般用于汉字状态的选择和切换。随汉化版本不同各有所异，请参看汉字系统的有关说明。

在不同的编辑软件中，也经常给 ALT+键安排一些控制指令，一般是用于对 CTRL 键做些补充。

第三章　DOS 操作系统的基本概念

DOS 系统是微机自身的管理系统。它起着指挥、控制、协调内存与外设的工作并合理调度资源的作用，是其它软件运行所必须依赖的基础。下面介绍一些 DOS 初步知识。

3—1　DOS 的版本

DOS 系统有1.0 、2.0 、3.0 及4.0 等版本，其版本以适应硬件发展为前提，随着版本号的升级，其功能越来越强。在各级版本中又有不同的子版本，如 2.1、3.2 、3.30等小的级别。

目前用得较多的版本是2.1 和3.30。前者多用于 PC/XT，后者多用于 286 、386 等机型。各级版本和其子版本间是有差别的，尽管在命令形式上相同，但外部文件一般是很难通用的，越高级的版本通用性问题就越突出。DOS 版本向下兼容的习惯认识在内部命令上是对的，在外部命令的使用上就不一定可行。即使同一级别的子版本间，不能向下兼容的情况也是有的。为了适应外设的高速发展，兼容多种低级版本的趋势在减弱。人们习惯认为 DOS 版本全部向下兼容，其实不尽然。

由于 DOS 版本不同，造成版本冲突的情况时有发生。初学人员在上机时注意到操作系统的冲突是必要的。例如：用 BACKUP 备份的文件，很难用不同版本的 RESTORE 进行恢复。当然还有其它一些外部命令等等。

由低版本向高版本的操作错误也不容忽视，经常有人把高版本格式化的软盘放到低版本环境上运行，这样的读写得不到系统

的保证是理所当然的。尽管同是低密软盘，也可能会出现因版本级别不相容而不能正确读写的现象。

3—2 DOS 的进入

用装有 DOS 系统基础文件的磁盘，对计算机引导，就可进入到 DOS 系统。在配置了硬盘的计算机上，一般都在硬盘上装有 DOS 系统的引导文件。在硬盘启动的 DOS 系统，屏幕将出现 C> 的提示。在无硬盘或有特殊需要的情况下，也可用带有 DOS 引导文件的软盘在 A 驱动器上启动，屏幕提示是 A>。

用于启动机器的系统盘上必须具备计算机自举用的两个隐含文件（文件名称不尽相同）和一个内部命令管理文件 COMMAND.COM 。这些文件缺少或版本冲突都会造成计算机不能启动的后果。

在系统启动的过程中，系统配置的设置文件 CONFIG.SYS 和屏幕管理扩充文件 ANSI.SYS 是起作用的，它们是控制硬件环境的重要组成部分。有些软件对受这些文件控制的硬件环境提出了具体的要求，则必须在这些文件中得以相应的满足，否则将使它们的运行受到影响。如果原来运行正常的软件突然出现不能启动的情况，应检查这两个文件是否存在或被变更。

如果运行 DBASE 软件时，出现“打开的文件太多”的错误提示，应检查 CONFIG.SYS 中的内容，该文件中至少应有如下命令：

FILES=20

BUFFERS=15

FILES 规定的 DOS 允许打开文件的限制数，取值过小，就不能满足软件的运行要求。BUFFERS 是设置 DOS 的缓冲区，数值的大小不会起太关键的作用。

可用任何一种文本编辑或用屏幕拷贝，对 CONFIG.SYS 文

件修改或创建。具体的方法请参考有关编辑功能的使用说明。

大写英文字母加“>”号是 DOS 系统默认的系统提示，如 A>、B>、C>等。它的出现表示 DOS 系统对计算机所有硬件的检测已经完成，如果没有出现错误提示，系统处于监控的等待状态，可随时接受键盘输入的指令。

3—3 DOS 系统的设备管理

人们可以运用指向 DOS 设备的专用符号控制外部设备完成指定任务，这些设备指令在所有 DOS 版本中是统一的并用‘:’标识。下面对常用的设备命名介绍如下：

A：把当前操作指向 A 驱动器

B：把当前操作指向 B 驱动器

C：把当前操作指向 C 驱动器

D：把当前操作指向 D 驱动器

PRN：把当前操作指向打印机

CON：把输入操作指向键盘，输出指向屏幕

当然还有其它的设备名，这些只是经常用到的。通过调度外设，可构成键盘、屏幕及各磁盘间的相互联系。这种联系是构成外设之间信息交换的通讯渠道。经常涉及的文件操作就是信息交换的重要组成部分。这些专用的设备符号经常被用在 DOS 命令中，它们的作用是指出参于命令操作的有关设备。

3—4 DOS 系统的文件概念

凡能用于与计算机交流的所有信息流，统称计算机文件。它比日常习惯的文件概念内涵更丰富。文件不仅是我们自己使用的程序和数据，所谓的 DOS 及各种计算机语言等系统也都属于计算机文件的范畴。要全面理解计算机文件的正确含义。

计算机文件在构成上可分为ASCII码文件和二进制文件；在用途上可分为程序文件、数据文件、系统文件等等。计算机文件不仅在分类上很复杂，属性亦有差别，此处不做过多的介绍。

很多计算机文件是在内存中操作或生成的，这种内存文件在关机或断电后会自然消失。由于存盘的疏忽，大量工作成果前功尽弃的事情是屡见不鲜的。因此，有效文件的存盘是操作员必须随时注意的问题。文件只有记入磁盘方可长期保存或复制。

计算机使用的所有文件，仅以其名称相互区别。如果出现文件名的重复，后续文件将对同名的原有文件进行覆盖。在文件操作的过程中，防止对有效文件的错误覆盖是要时刻警惕的事情。

在DOS系统中的文件名包括两部分，前部称主文件名，是必须存在的；后部称文件扩展名，可用可不用。扩展名多用于表示文件的分类特征，很多计算机语言对文件扩展名都有自己默认的统一规定，以便区别其它语言系统。

在DOS系统中有关文件名的规定如下：

1. 主文件名 可由1—8个字符组成，包括字母、数字和其他可打印的标准字符，但不许有空格、‘.’、‘*’、‘?’。

2. 扩展名 又称文件的后缀名或后缀，可有0—3个字符组成，字符集要求同上。文件名和扩展名间用‘.’隔开。

由于磁盘可存放的文件数以百计以至更多，为了文件管理方便，DOS系统规定了操作文件的通配符，它是简化文件管理的有效方法之一。

通配符有‘?’和‘*’两个，在使用时，以‘?’表示对应本位的任意字符，可连用，如：???等；‘*’可代表任意位数的任意字符。通配符可在主文件名和扩展名中分别使用。使用通配符，可统一操作文件名具备某一共同特征的文件。

以下举例将有助于对文件通配符的理解：

AB??.PRG 指定前两位是AB字符，第三、四位是任意字符，扩展名是PRG的文件。所谓的任意字符包括字符不存在的情

况，即使文件仅以字符 AB 命名，也将视为满足通配符的指定。这一点对以下的举例中同样适用，不再一一解释。

??B. *　指定前两位是任意字符，第三位是 B，扩展名任意的文件。

M *. DBF　指定以字符 M 开头，其余七位任意，扩展名为 DBF 的文件。

*. *　指定任意文件名，任意扩展名的文件，即全部文件。

以上举例是对当前目录中的文件而言的。有关目录概念将在下面具体介绍。

3—5　DOS 目录

尽管使用了文件名通配符，但文件操作仍然不尽人意。在成百上千的文件表中寻找需要的文件，仍是耗费时间和精力的事情。如果忘记了文件名，就会更加令人烦恼。如果能够把众多的文件分门别类的存放，无疑是简化操作的有效办法。

对于众多的磁盘文件，可根据其用途或某一特征进行多分支多层次分类，把文件存放在这种按照分类建立的分支和层次结构中，这就是目录的概念。这个概念和图书馆的目录分类在客观反映上是很近似的。目录的实际效果如何，决定文件的分类和组织是否合理。

如果目录使用得科学，文件就会被管理得井井有条，对文件操作的效率自然会提高。如果盲目地使用目录，反而会使文件的搜索变得更加困难。DOS 系统提供了目录管理的能力，为文件的分类操作提供了可能的条件。要使目录真正发挥作用，更重要的是靠操作员自己的主观努力。

目录的命名规范与文件名相同，只是为了便于操作，往往省略扩展名部分。最原始的目录被称为根目录，以下可以是多分支和多层次的组合。各级的分支被称为隶属于上级的子目录。其整体形象

类似一棵倒置的树形。

下面是一个目录结构的示例(见图1)：

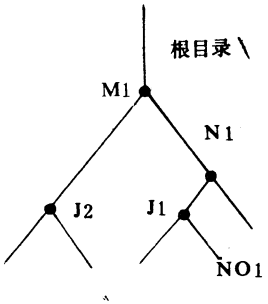


图 1

各级目录既可横向继续扩展又可纵向继续延伸，其中J2 ，N1为一级子目录；J1为二级子目录； NO1为三级子目录。

这种目录结构又叫路径。它是文件应用中不可缺少的实用手段，在多人合用或多任务合用的计算机上，这点就更为突出。目录的使用可使各目录中的文件被分别存放于相对独立的磁盘空间。由于目录间的隔离作用，不但有效的加强了文件的安全性，还会大大减小由于文件太多而造成的干扰和麻烦。因此，操作员必须熟练掌握目录的实用操作，这对保证自己或他人的数据不被破坏有着重要意义，也会给文件使用带来很多便利。

对于初接触目录概念的操作员来说，往往感觉目录的使用很抽象，一时难以建立明确的目录概念。产生畏难情绪的一个重要原因是目录应用的方式富于过多变化，这也正是能够方便地操作文件所必须依赖的特点。

变化多端绝不是无章可循，DOS 的目录管理有两个非常清晰的起点，把握住这两个起点是学习目录使用的关键。这两个起点一个是根 '\'，另一个就是当前的目录。凡是涉及跨越根目录的操作，

均用'\'号作为路径计算的开始，否则系统总是默认当前目录为路径计算的参考点。在明确了这两个起点之后，目录就会变得明朗，使用起来自然会做到得心应手。

在实际工作中，多级目录被广泛应用。多级目录的结构意味着在到达一个目的目录时有可能要通过若干个中间目录才能实现。这种由一个 DOS 目录起点到达另一目录所必须顺序经过的各级中间目录被称为路经表。路经表中明确的反映了各级目录的从属关系。

作为一种通俗的理解，可以认为 DOS 文件名不仅指主文件名和扩展名，在主文件名前还可包括驱动器符号和路径表。用这种统一的概念理解文件名的涵义虽然不尽科学，但会给掌握文件操作减少很多麻烦。

大部分文件名和目录名是由操作员定义的，虽然符合规范的命名均可被 DOS 系统承认，但在实际使用中切不可对命名过于随意，应既要有规律，又要意义明确。文件名要和内容密切相关，同一类文件要有共同的字符特征并要符合人们的日常习惯，力求做到记忆容易、查询方便、操作简单。对每个人来说，都应当有意识地培养自己的命名习惯，这会给今后带来很多效益。对任何一个操作员，忘记文件名的事情是必定会发生的，你会发现，符合一定规范的文件名是容易被确认的。如果必须在文件堆中漫无边际地翻阅查找自己随意定义的文件名，无疑是件既令人头疼又令人遗憾的事情。

第四章　常用的DOS内部命令

在DOS监控状态下可运行DOS命令，并得到运行结果。所谓DOS系统的监控状态，是由DOS系统解释输入的指令，对能够执行的给予实施，对非法的输入给出有关错误类型的提示。

计算机的指令必须装入内存后才能被执行。有些DOS命令是常驻内存的，随时都可调用，这部分被称为DOS的内部命令。还有一部分不常驻内存的命令，它们用磁盘文件的形式记录在磁盘上，一旦DOS系统检测到这些命令，则要先对命令进行装载，然后再执行。这部分就是DOS的外部命令。两种命令由于装载方式的差别，在调用格式上也不尽相同。

内部命令多用于功能较简单、使用频繁的命令。下面先介绍几个常用的内部命令

4—1　环境控制

1. CLS 初始化屏幕命令

C>CLS ↵

清除当前屏幕信息。

2. TIME 显示或变更系统时间

C>TIME ↵

屏幕出现

Current time is 9:23:19.55 Enter new time:

最后两位(55)是秒的小数值。仅想查询时间，回车即可。如进行时间校对可输入正确的时间后回车。

3. DATE 显示或变更系统日期 系统日期以系统时间为依据，在作用上比系统时间更重要。它可为许多有日期要求的场合提供日期数据。

文件的生成日期就是系统日期的生动体现。以后会看到，使用文件日期特征将有助于文件的替换和复制。

C>DATE ↵

屏幕出现

Current date is Sun 7—07—1991

Enter new date (mm—dd—yy):

这种月—日—年的格式是美国的表示方法，不符合我们的习惯，经常使用就会适应。如仅想查询日期，回车即可；如想变更，可按格式要求输入正确日期并回车。如果输入了非法的日期数据，系统不予承认。

4. VER 显示 DOS 版本号 目前 DOS 版本繁多，给软件的交流带来一些不便。当屏幕显示 DOS 版本错误时，核对 DOS 版本是必要的。只要选择了合适的版本，这种问题自然会解决。

C>VER ↵

屏幕显示

IBM Personal Computer DOS Version 3.30

引导当前操作系统的 DOS 版本号被显示在屏幕上。该例的 DOS 版本号为3.30。

4—2 有关目录的约定

目录的叫法符合文件管理的日常习惯，但在计算机中更普遍的是称目录为路径。在有关目录的创建、转移或撤消时，路径的叫法就显得更为贴切。如果你要对某一目录操作，就必须列出到达该目录所经过的所有路径。

为了介绍目录管理命令的方便，先说明关于目录使用的一些

共同的约定。

如果目录前有驱动器符号，表示该驱动器被指定为当前驱动器，后续的目录操作在该驱动器上进行。

如果目录用‘\’开头，表示路径从根出发，按路径表延伸到指定目录，此目录被定义为当前目录或操作对象；目录用字符开头，表示以当前目录为起点，按路径表向下延伸，路径终点被定义为当前目录或操作对象；如果当前目录是根目录，用字符和用‘\’开头是等价的。

路径表的延伸用‘\’分割，每经过一个‘\’延伸一级路径。两个‘\’号之间是中间路径的名字。在路径表中从起点到终点经由的所有路径名称都要列出，不能跨越。

4—3 目录管理命令

1. MD ＜目录名＞ 目录创建命令 在当前路径为 RR 目录时要求创建 RS 和 LZ 两个目录(见图2)，可发出如下命令

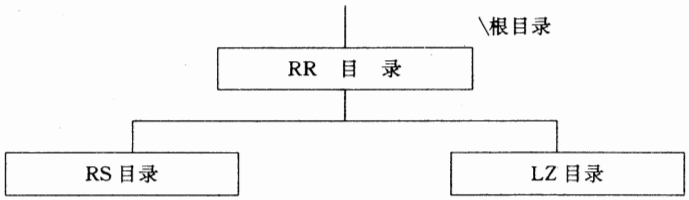


图2

C＞MD RS ↓

C＞MD LZ ↓

如图2所示：RR 为当前目录；RS、LZ 为新创建的目录；RR 的上级目录为根目录。

C＞MD RS\R1 ↓

此命令仍在 RR 目录发出，在 RS 目录下有 R1目录被创建(见图

3)。

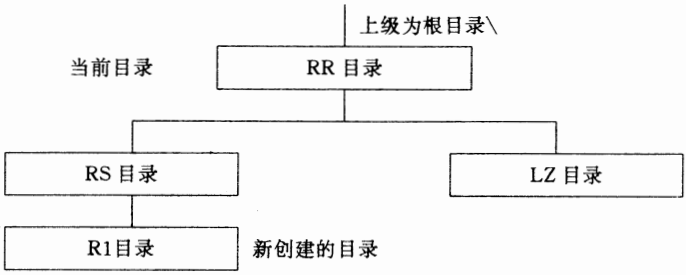


图3

注意,命令 MD RS\R1 是用字母开头,表示路径从当前向下延伸,跨越的路径用\隔开。

如果当前在 LZ 路径中,同样可创建 RS 所属的 R1目录,可用如下命令:

C>MD \RR\RS\R1 ↵

用\开头,这是以根为起点的操作,跨越的两级目录均用\隔开,最后是路径的终点,是我们要到达的目的(见图4)。

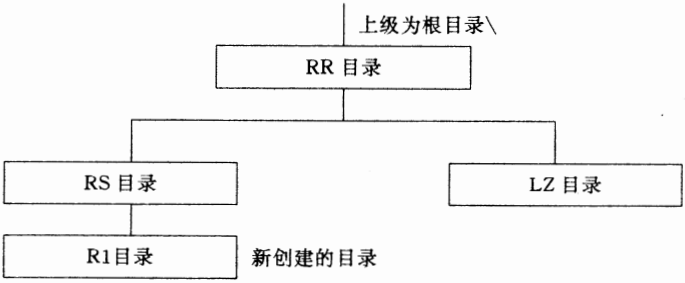


图4

此时要求在根目录下的 RR、RS 已经存在,RS 下有 R1目录被创建。可见,只要是这个目录结构,在其它任意路径下创建 R1的命令在形式上都是相同的。无论当前路径在哪里,符号"\"指出操作起点由根目录开始计算,这是 DOS 路径清晰起点的具体实现之一。

以上两种创建 R1目录的命令在方式上主要区别于一个\号,

这就明显地区分了路径起点的差别。究竟是以根为起点还是以当前目录为起点，取决于路径表是否用符号“\”开头。

2. CD ＜目录名＞ 目录转移命令 进入到指定的目录，并限定该目录成为当前工作的对象，在不进行新的指定之前，文件操作仅在本目录进行。这个目录被称为当前目录或当前路径。缺席目录或缺席路径与当前目录有严格的区别，注意不要混淆。MD 命令只建立目录结构并不转移当前目录，改变当前目录的操作必须用 CD 命令。

仍以图3为例，在当前 RR 目录时发出：

C＞CD RS\R1 ↓

即可由 RR 路径进入 R1，同时 R1被指定为当前目录。这是当前路径的向下延伸。再发出：

C＞CD \RR\LZ ↓

当前操作跨越了 RS，被转移到 RR 路径下的 LZ 目录。这是跨越根目录的转移，用了\开头。

C＞CD \ ↓

根目录成为当前目录。

更多的路径分支或更多的路径层次在操作方法上是完全一致的。

如果想进入的目录在当前目录以下，路径表从下面的第一级开始并逐级延伸。

如果想进入的目录在当前目录以上或不在同一目录链中，路径表一律从根开始，根以下的路径逐级延伸。应当明确认识到的是当前目录与根目录之间的路径总是被忽略不计的。

到此，关于 DOS 路径的两个起点应当很清楚了。如果明确了两个起点的概念，就已经掌握了 DOS 路径最主要的概念。

根据以上情况会发现，如果进入到多层的子目录后，想返回到上一级的目录，就要输入一个多层的路径表，使用并不方便。DOS 提供一个特殊的操作：

C>CD . . ↓

可返回到前一级目录。

3. CD 目录查询命令 无路径表的 CD 命令被解释为查询路径命令。当前路径表被显示到屏幕。在不清楚当前路径时，可用 CD 命令问讯：

C>CD ↓

C:\RR

当前在 C 盘根下的 RR 目录。

C>CD \RR\RS ↓

目录转移。

C>CD ↓

C:\RR\RS

查询结果是当前在 C 盘根下 RR 目录所属的 RS 目录。

4. RD <目录名> 目录删除命令 其路径表的应用与上相同，其作用是把所指向的目录删除。

删除指令必须从被删除目录的上级发出，删除当前所在目录或该路径表中的上级目录是不能被执行的。被删除的目录必须是空目录，若该目录中存有文件或有低级子目录存在，删除命令因属非法而被拒绝执行，只有为空目录时方可删除。

以图5为例，发出：

C>RD RS\R1 ↓

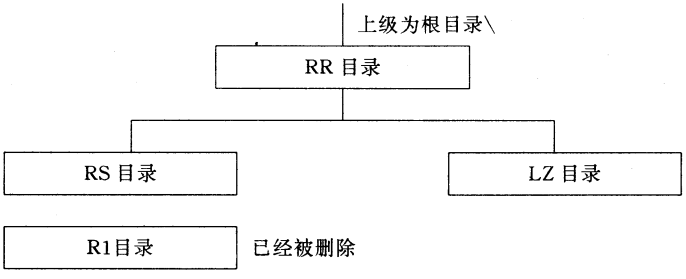


图5

在 RS 目录下的 R1目录被删除，此时的 R1必须是既无文件又无低级子目录的空目录，否则删除不能被实际执行。

C>RD LZ ↵

可删除 LZ 目录，要求同上。

如果在其它路径删除 LZ 目录，命令可按如下形式：

C>RD \RR\LZ

经过 RR 路径删除 LZ 目录(见图6)。

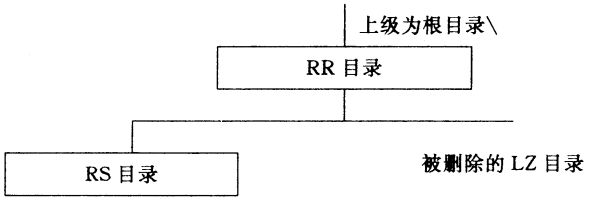


图6

两种方法的结果是等效的，区别在于发出命令的出发点不同。即起点的两种方式。

5. PATH <路径表列> 定义缺席路径命令 DOS 的当前目录限定了文件操作的有限区域，给操作员带来了文件使用的便利，同时也带来了使用系统命令的不便。我们知道 DOS 的外部命令必须有系统文件的支持才能工作，在任意路径下使用 DOS 的外部命令是随时需要的，如果在任意目录中都要求存放 DOS 的外部文件，无疑是一种资源的浪费。

为了解决系统文件的调用，DOS 规定了缺席路径命令。缺席路径是当前路径的后援，当文件不能在本目录中被发现时，DOS 自动到指定的缺席路径中去搜寻。

指示缺席路径的命令是 PATH。一般情况下，缺席路径仅限于可执行文件的调用，其它的文件操作仍是以当前目录为前提。DOS 的系统文件设置在缺席路径上是非常需要的。

定义的缺席路径宜少不宜多，过多会明显的影响文件检索的

速度。定义命令可由屏幕发出，但更多是在批处理文件中定义。

以下路径情况（见图7）可说明缺席路径的使用：

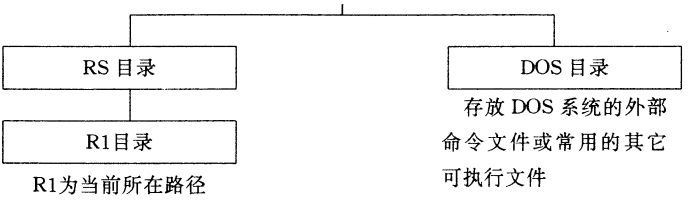


图7

磁盘的格式化命令 FORMAT 是外部命令，FORMAT.COM 文件存放在根目录下的 DOS 路径，当前使用的目录是 R1。如果想对 B 驱动器中的磁盘进行格式化处理可有多种选择，一是通过文件名指示路径，这是外部命令的调用形式，即

C>\DOS\FORMAT B: ↵

或进入到 DOS 目录后再执行外部命令，即

C>CD \DOS ↵

C>FORMAT B: ↵

以上两种情况都要涉及到由于文件不在当前目录而增加的附加操作（请参看第五章），在经常交叉使用本目录文件和 DOS 外部命令时这两种方式都是极不方便的。如果在使用 DOS 外部命令之前发出

C>PATH C:\DOS ↵

将 C:\DOS 指定为缺席路径，就意味着在任何路径情况下，一旦检索可执行文件失败，C:\DOS 目录将会被询问，询问成功，该文件就会被装载并执行。

缺席路径指定后，在任何驱动器的任何路径上发出 FORMAT 命令，格式化磁盘的操作都会被执行，不再需任何附加的路径操作。这就是使用缺席路径带来的便利。

4—5　文件操作

1. DIR <文件名或通配符> [/W] [/P]　文件名查询命令。按要求显示指定文件的信息表列。在文件名或通配符前可指示驱动器和路径表。

首先显示文件所在的路径，然后把符合指定特征的文件信息列表到屏幕，包括主文件名、扩展名、文件的 ASCII 码字节数、生成日期及时间五项。每个文件名的信息占一行屏幕。

采用/P 参数时，每屏显示24行信息自动暂停，按任意键继续显示，直到完成。此为 DIR 命令的标准输出。

采用/W 参数时，只显示主文件名和扩展名。每五个文件横排显示，占用一行屏幕。这样可使有限的屏幕显示更多的文件名，但每个文件的信息相对减少。

如欲显示当前路径中某一特征的文件可参考如下举例：

C>DIR AB??. PRG ↵

显示前两位是 AB 字符，第三、四位是任意字符，扩展名是 PRG 的文件信息。采用了标准的显示形式。

C>DIR ??B. * /W ↵

显示前两位是任意字符，第三位是 B，扩展名任意的文件信息。/W 采取了横排显示方式。

C>DIR M * . DBF /P ↵

显示以字符 M 开头，其余七位任意，扩展名为 DBF 的文件信息。/P采用了标准显示并分页暂停。

C>DIR * . *

显示任意文件名，任意扩展名的文件信息。即全部文件。

C>DIR ↵

等价于 DIR * . * 。

请注意以下两个命令间的区别：

C>DIR \RR\LZ ↵

或　C>DIR\RR\LZ*.* ↵

在当前目录显示 LZ 目录的文件信息。两种命令方式完全等价。显然前种方式更为简便。

C>DIR R1 ↵

或　C>DIR R1*.* ↵

在当前目录显示 R1目录的文件信息。通过以上两例说明，在检索全部文件时，可以使用单独的路径名称。如果不是全部文件，需要使用具体文件名或使用文件通配符时，最后的路径分界符'\'是不可省略的。请看下例：

C>DIR R1\AB??.PRG ↵

显示 R1子目录中，前两位是 AB 字符，第三、四位是任意字符，扩展名是 PRG 的文件信息。此时的起点是当前目录。

C>DIR \RR\LZ\??B.*/W ↵

显示 RR 路径下 LZ 目录中，前两位是任意字符，第三位是 B ，扩展名任意的文件信息。采取横排显示方式。此时的起点转向根目录。在任意路径上发出此命令可对该目录进行查询。

如欲完成跨越路径的文件查询，可参考如下举例：

设定的工作环境如图8，在 B 驱动器内的磁盘上已经存在目录情况，指定 RS 为当前目录。

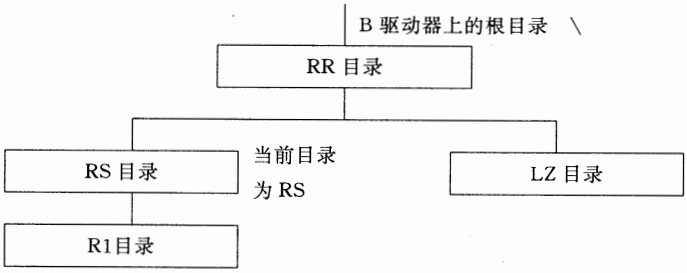


图8

C>DIR B:*.* ↵　　或用下面的等价命令

C>DIR B: ↵

显示当前目录中所有文件名，即 RS 目录中的全部文件。

C>B: ↵

转移当前操作到 B 驱动器。

B>DIR RS.PRG ↵

显示 B 驱动器当前目录中名叫 RS.PRG 的文件信息。

B>DIR \RR\LZ ↵

在 B 驱动器的当前目录显示 RR 目录下 LZ 子目录中的文件信息。

2. DEL <文件名或通配符>删除文件命令　计算机磁盘的容量虽然很大，但其必定是有限的资源。在计算机使用过程中，总会有大量的过期文件、临时文件，及自动生成的备份文件等等。这些没有使用价值的文件不但会占用大量的磁盘资源，还会给文件的检索、查询、拷贝带来许多不必要的麻烦。因此定期清理目录，删除不需保留的文件，是操作员经常要做的工作。DEL 就是 DOS 系统的文件删除命令。

DEL 命令支持驱动器符、路径表和文件通配符，其使用方法与 DIR 命令完全相同，只是执行的结果不同，DEL 是将指定的文件实施删除。

C>DEL RS.PRG ↵

删除当前目录中名叫 RS.PRG 的文件。

C>DEL R1 ↵

这条命令在 DOS 状态下有两种解释：

(1)如果在当前目录中有名叫 R1的文件(无后缀名)，则该文件被删除。DOS 认为完成了指定任务，不显示错误信息。

(2)如果在当前目录中没有名叫 R1的文件，有叫 R1的目录，DOS 解释认为操作员要删除 R1目录中的所有文件，但不包括 R1目录下的子目录。

DOS 对涉及到删除整个目录的操作是慎重的，在执行删除前发出问讯：

Are you sure (Y/N)?

如果确实想删除，回答 Y 并回车，删除被实施，回答 N 可使删除命令作废。

C>DEL R1 ↵

与下面的命令

C>DEL R1*.* ↵

在删除 R1子目录的操作上是等价的。但此时要注意到一个潜在的危险，前者会将当前目录中名为 R1的文件和路径删除，后者则只能是路径名。

C>DEL A:*.* ↵

删除 A 驱动器中磁盘上当前目录中的所有文件，但不包括磁盘上的其它目录及当前目录中的下级子目录。如果 A 驱动器上仅有根目录，实际上删除了 A 盘上的所有文件。如果 A 盘有多个目录，可在 A:后指示操作的路径。

操作员对文件的删除要十分小心，一个错误的删除操作可能会丢失大量有价值的文件。误删文件的事情几乎每个经常上机的操作员都遇到过。

值得特别强调的是，一旦发现误删了文件，切不可进行破坏性操作。所谓的破坏是指对已删除文件所在的磁盘区域进行了新的覆盖。向该目录拷贝文件或进行文件编辑等等都可能造成对删除文件的磁盘区形成覆盖，都是属于破坏性操作。实际上 DEL 命令只是改变了文件档案中与文件名有关的信息，原文件的内容仍然保留在磁盘上。虽然 DOS 系统不能再操作这些文件，通过工具软件尚可使文件全面恢复。这样的问题一旦发生，切不可急躁，更不要盲目处理。自己不懂时可请专业人员对文件进行恢复。对已被覆盖的删除文件是没有任何办法恢复的。

3. COPY <源文件名> <目的文件名> [/V] 这是复制文

件(拷贝)命令。重要或有保存价值的文件,要在软盘上可靠保存。软盘上的文件又随时需要复制到硬盘上使用。这些文件的复制和传送是靠拷贝命令完成的。

COPY 命令有两个支持通配符的文件名结构,第一个用于标识源文件,第二个用于标识目的文件。如果一项缺席,DOS 系统可在无冲突的情况下自动采用默认值,否则会出现错误提示。DOS 的默认值包括驱动器、路径表、文件名、扩展名或文件通配符。如果没有进行必要的指定,目的文件与源文件同名。

系统提示符 A>、B>、C>等已经显示了当前占用的驱动器,这个驱动器符号在 COPY 命令中是可缺席的参数。这个驱动器称缺席驱动器。在实际中,相当的拷贝命令是在不同的驱动器之间进行的,凡是涉及到缺席驱动器的符号总是可以省略的,无论它是在源侧还是在目的侧。换个角度说,当前系统提示所标明的驱动器符在 COPY 命令的参数表中总是可以被省略的,这是为了操作的简化。如果你愿意按命令规范把它写上,当然也不会出错,只是添了些麻烦而已。

在了解 COPY 命令的使用时,一定要明确缺席驱动器和当前路径的概念,否则会感到命令的参数表令人费解。

在下面的举例中涉及到的路径和指定的文件都假定是已经存在的。

举例: 单一文件的拷贝

C>COPY AA.P A:↵

拷贝当前路径中名为 AA.P 的源文件到目的驱动器 A 的当前目录且文件名不变。如果 A 盘没有任何目录,应认为 A 盘当前目录是根目录。由于系统的提示符是C>,当前路径必定是C 盘上的某一目录。在 AA.P 前面的 C:是被省略的,即

C>COPY C:AA.P A:↵

与上一个命令等价,只是未省略缺席驱动器符号。

类似命令无论在C 盘的哪个目录发出,该目录中被指定的文

件被拷贝到目的驱动器。

如果把 A：换成 B：文件按上述约定拷贝到 B 驱动器。

C>COPY AA.P B:AA.P ↵

AA.P 的相互对应，明确的表达了在拷贝过程中不改变文件名的意图。实际上这是 DOS 默认的文件拷贝形式。在没有特别指出的情况下，复制的文件总是与源文件同名的。

如果在同一驱动器复制文件，仍然要使用源文件的名字，就必须指出新的路径。若 C 盘根下的 BB 路径存在且不是当前路径，可用如下拷贝命令实现同盘同名的文件拷贝：

C>COPY AA.P \BB\AA.P ↵

除了具体的文件名外，用通配符匹配也是许可的。目的文件名用 *.*、AA.*、*.P、??.?等不同的形式都可得到与用 AA.P 相同的结果。这是因为在单一文件的同名传送中这些通配符都不会造成文件名的冲突，都是合法的。在多文件传送时通配符使用得是否合法，就要斟酌一番了。

对上面的课题，有人为了操作简便，干脆省略了文件名的匹配标识，把命令写成

C>COPY AA.P \BB ↵

作为 DOS 命令，这是完全合法的，是肯定会被执行的。但执行结果可能会偏离本来的意图，毛病出在该命令有二意性。

如果根下确实存在 BB 目录，其执行结果与上相同，可以达到预期目的。否则目的参数的解释就会发生变化，DOS 认为操作员的目的是把源文件拷贝到根目录并重新命名为 BB。因此，这种简化必须在环境明确的情况下准确的掌握其执行状态，不论你希望哪一种。

对拷贝的文件可重新命名，请看下例：

C>COPY AA.P BB.P ↵

目的文件名的改变，意味着目的文件要在传送的过程中重新命名。当目的文件名用 BB.P 与源文件匹配时，AA.P 的复制文件将以

BB.P 命名。这也叫换名拷贝。换名拷贝可在本目录进行，当然也可在不同的目录上进行。

以上举例是将C 盘文件拷贝到A 盘(或B 盘)，相反方向的拷贝只要按照命令规范把源驱动器符和目的驱动器符交换即可，即

C>COPY A:AA.P ↵

拷贝 A 盘当前路径中名为 AA.P 的源文件到目的驱动器 C 的当前目录中，文件名不改变。由于系统的提示符是 C>，目的驱动器符省略，文件到当前目录，路径表省略。文件同名，目的文件名省略。

假设当前路径为根下的 RR，完整的命令形式是：

C>COPY A:AA.P C:\RR\AA.P ↵

由此可以看出，若不想省略参数，就大可不必去管缺席的驱动器和路径，完整的参数表在任何情况下都可以准确无误地把源文件复制到目的文件，只是过于繁琐。对于尚不能理解参数表的含义的操作员，这个办法倒是解决实际问题的一个手段。

以上关于 COPY 命令的论述，概括了参数表常用的一些特征，实际问题要比这些更复杂。如果能够从这些简单的举例中领悟到定义参数表的基本规律，就会在循序渐进的实践中逐步做到运用自如。以上对命令状态的解释虽然较为严谨，但读起来却难免令人感到过于呆板。在以后的举例中，凡有与前重复的概念，叙述从简，但仍要在这些概念的基础上去理解。这样的做法会使问题的重点更突出。

试分析下例中的缺席、源、目的驱动器：

C>COPY A:AB.C B: ↵

把 A 盘的 AB.C 拷贝到 B 盘，当前驱动器是 C。

A>COPY C:BB.B ↵

把 C 盘的 BB.B 拷贝到 A 盘，当前驱动器是 A。

B>COPY A:91.01 C: ↵

把 A 盘的91.01 拷贝到 C 盘，当前驱动器是 B。

A>COPY B:B5.1-1 C:\BB ↵

把 B 盘的 B5.1-1拷贝到 C 盘根下的 BB 目录，当前驱动器是 A。

用通配符提取源文件的拷贝为

B>COPY *.PRG A: ↵

拷贝 B 盘当前路径中所有扩展名是 PRG 的源文件到目的驱动器 A 。

B>COPY *.PRG A:\PRG*.* ↵

此命令与上不同的是，指定文件传送到目的盘 A 上名叫 PRG 的子目录中。命令中的两个'\'号限定了 PRG 只能是路径名。通配符'*.*'限定了文件名的匹配，即文件名不变。

C>COPY A:*.PRG B:\PRG*.PPP ↵

这里用 *.PPP 与 *.PRG 匹配，表示要在拷贝过程中变更文件名称。这些在 A 盘是 PRG 后缀名的文件拷贝到 B 盘的 PRG 目录后全部改变成以 PPP 为后缀名的文件。* 和 * 的对应表示主文件名不改变。不用担心，文件的内容是不会改变的。

这种换名的拷贝只是在变更文件名后没有冲突的情况下才能被执行。仍以上例，在 PRG 文件不止一个的情况下，如果用 PPP.* 去与 *.PRG 匹配，其含义是把原来的主文件名全部改成 PPP，而原扩展名全部是 PRG 不变，结果形成所有的文件都是一个同样的名字。这当然是非法的匹配方式，象这样有文件名重复的参数，DOS 认为是不可执行的。

再强调一下不完整参数的二意性，如

A>COPY *.DBF C:\BB ↵

其一种可能是指定的 C 盘确实存在路径 BB，则 A 盘所有的 DBF 文件被拷贝到 C 盘根下的 BB 目录，且文件名完全对应。另一种可能是指定的 C 盘路径不存在，则 A 盘所有的 DBF 文件被连接拷贝到 C 盘根下名为 BB 的文件之中。

这点与单一文件是有区别的，多了一个连接过程。我们知道由 DOS 连接的 DBF 文件是不能使用的，也是很难分离的。如果误认

为有了备份而破坏了源盘，数据丢失就是不可避免的了。

当你确实要把一批 ASCII 文件连接在一起时，这倒是一种非常简便易行的好方法。关键要看你的目的到底是什么。如

C>COPY *.TXT SSS

把所有后缀名是 TXT 的文件连接拷贝到名为 SSS 的文件中。若有选择的连接文件，可用'+'号连接文件名来达到目的，即

C>COPY AB.C+BB.B+91.01+B5.1-1 SSS

执行结果是把文件按顺序连接到 SSS 文件中。

最后再补充说明两点：

COPY 命令有一可选的校验参数/V，在 COPY 命令的末尾加该参数时，DOS 对复制的文件重新读出并与源文件比较，以确认复制的文件是可使用的、正确无误的副本。

COPY 命令对同名文件进行覆盖，文件名的冲突隐含着丢失文件的危险。即使是同一文件，也要注意时效性，用过时的文件覆盖了新文件同样是令人遗憾的事。

在同路径下欲拷贝同名的文件，被 DOS 解释为自己拷贝自己，不能执行，系统提示是

File cannot be Coiped onto itself

0 File (s) Copied

在源文件不能发现时，DOS 提示为

File not foude

0 File (s) Copied

COPY 命令还有更多的变通方式，进一步的学习请参考有关书籍。

4. REN <原文件名> < 现文件名>文件更名命令 对文件的现有名称不满意，用 REN 命令可对文件重新命名。

C>REN AA.P BB.C ↵

将文件 AA.P 更名为 BB.C。

REN 命令承认通配符，与 COPY 命令类似的是只有在通配

符操作不会引起冲突的情况下才能正确执行。

C>REN *.PRG *.PPP ↵

把当前目录中用 PRG 为后缀的文件改成以 PPP 为扩展名。对其它驱动器或路径操作在文件名前注明即可。

REN 命令承认驱动器和路径表，可在任意路径下执行对其它目录的文件更名，即

C>REN \RR\LZ*.PRG *.PPP ↵

这是跨越目录的更名方式。

5. TYPE < 文件名> 显示指定文件内容命令

C>TYPE A:\BK\LL.TXT ↵

显示 A 驱动器上 BK 路径中名为 LL.TXT 的文件内容到屏幕。该命令不支持通配符，只能使用具体的文件名。在翻阅文件时，这条命令很有用。

第五章　常用的 DOS 外部命令

DOS 的外部命令实际是本系统可执行文件的文件名。这些文件一般具有.COM 或.EXE 的扩展名，这是标识可执行文件的统一方法。在使用外部命令时，必须保证计算机能够搜寻到外部命令文件所驻留的磁盘及路径，否则命令不能被装载执行。

为了保证文件被可靠装载，指示文件存在的方式可有以下三种选择，至少要有一种方式是可靠的。

(1)外部文件在当前目录。

(2)外部文件在已经用 PATH 指定的路径上。

(3)在外部命令前增加引导路径，DOS 将按先导的驱动器符和路径表去查找文件。这是最灵活的方式，但输入字符最多。

请看下面的格式

驱动器　路径表　外部命令 ……

C＞\PCDOS\外部命令……

在 C 盘的根目录下有 PCDOS 目录，DOS 外部命令驻存在该目录之中。在任意路径中可用\PCDOS\指定 DOS 执行外部命令的文件导向。如果不在当前驱动器，路径前可加驱动器符，如

A＞C:\PCDOS\外部命令……

这是外部命令前半部分的标准形式，这种格式对所有外部命令通用。为了突出重点，下面介绍的外部命令将不再对前导驱动器和路径一一陈述，实际中可参照这种格式使用。

5—1 磁盘格式化命令

FORMAT <驱动器符> /S /4 磁盘格式化

出厂的软盘一般是不能直接进行数据读写的，或者说磁盘是无格式的。只有在格式化处理后的磁盘才能实际使用。顾名思义，FORMAT 命令是用于对软盘进行格式化的。虽然该命令也可对硬盘格式化，但不能进行初始化。硬盘的初始分区尚需其他外部命令的支持。

常用的驱动器有两种规格，一种是普通软驱，可格式化360K的低密软盘，还有一种是高密度软驱，可格式化1.2M 的高密软盘。与驱动器对应，磁盘也有两种。规格标有“48TPI ”的是360K 低密盘，规格标有“96TPI ”的是1.2M 高密盘。低密度磁盘进行高密度格式化一般是不允许的，即便能够成功，数据的可靠性也很难得到保证。

在格式化前要清楚驱动器的类型和磁盘的密度。磁盘与驱动器对应使用。把软盘放入相应的驱动器后，视所在驱动器键入下列命令之一，即

C>FORMAT A：↵

或 C>FORMAT B：↵

确认磁盘插好后，再回车一次，格式化开始进行。3.0 以上的版本将在屏幕报告进度。软盘格式化后，屏幕报告磁盘信息，包括磁盘共有多少字节，其中有多少可用字节，有多少不可用字节。如果传送了系统文件，还报告文件所用空间。

一张磁盘的格式化完成后，若继续进行，可更换新的磁盘，回答 Y 并按两次回车。中止操作可回答 N 并回车。

如果必须在高密驱动器上格式化低密软盘，可在命令后加/4参数。假设 A 驱动器是高密驱动器，下面的命令可格式化低密磁盘：

A>C:\PCDOS\FORMAT A:/4 ↵

在格式化时选择/S 参数，指定操作系统在格式化后传送系统的隐含文件和 COMMAND. COM 文件到软盘。用这样的软盘可引导计算机的启动。系统的隐含文件是不能用 COPY 命令复制的。

在格式化磁盘时要特别注意：

系统对坏字节做出标识，但不会影响正常使用，只是磁盘容量受些影响。反复的格式化可能使坏字节减少甚至消失，但这并不是可取的办法，尤其是长期保存的数据，往往造成因小失大的后果。为了几 K 的磁盘空间冒着因磁盘读写不稳定而造成数据损失的风险，实在是得不偿失的事情。由此看来，还是保持坏字节原来的属性好一些。

对存有数据的磁盘进行格式化要小心，格式化会清理掉所有的数据，这种清理是不可恢复的。尤其要注意，如果把格式化指向硬盘，将会使所有硬盘数据丢失，其后果将是非常严重的。

5—2 目录结构查询命令

TREE <驱动器符> /F

查询指定驱动器的路径分配情况，在屏幕报告指定磁盘上所有的路径名称。无驱动器符显示默认驱动器。

C>TREE ↵

屏幕将按 C 盘的目录结构显示各个目录名及目录的从属关系。

/F 选用，在显示路径名称同时，显示路径中的文件名称。即

C>TREE/F ↵

当确知文件存在于磁盘，一时难以找到时，本命令可在整个磁盘上全面搜寻。

工具软件可代替这个功能，有时会更方便一些。

5—3 磁盘复制命令

DISKCOPY ＜源驱动器符＞ ＜目的驱动器符＞

复制整个的磁盘。

它不仅在外部命令上区别COPY命令，执行状态也有区别。

COPY只能按文件名拷贝文件，而该命令不计较磁盘上有多少以至有无文件，它只是把整个源盘按磁盘的磁道和扇区逐格复制到目的盘。因此，它不需文件名参数，只要有驱动器符就行了。

COPY只能在格式化好的磁盘上复制文件，而该命令可对未格式化的目的盘操作，格式化和复制文件可一次完成。

COPY不能复制系统的隐含文件，它能。

COPY能在软、硬盘间传递文件，该命令只能软驱之间或软驱本身复制文件。

举例：

C＞DISKCOPY A：B：↵

将A驱动器中的磁盘整版复制到B驱动器中的磁盘。

注意，高密驱动器可兼容低密驱动器，相反则不能。低密磁盘可在两驱动器间任意交换，均属合法。如果只有一个高密驱动器，在复制高密磁盘时必须在本驱动器上完成。命令如下：

C＞DISKCOPY A：A：↵

或　C＞DISKCOPY B：B：↵

源驱动器和目的驱动器相同，在该命令中被认为合法并可执行。在本驱动器复制磁盘时，DOS先将磁盘内容向内存读入直到占满全部可用空间，即使尚未读完，写操作也将开始，屏幕提示将源盘更换成目的盘，用回车键激活写操作，内存写完后，屏幕再次提示更换源盘，重复上述过程直到整个磁盘拷完为止。

对于360K磁盘可换盘一至两次，1.2M磁盘要换四到六次。如果内存的空间小，次数还要多。在大多数机器上，高密盘只能用

这种方法复制。

5—4 磁盘备份命令

BACKUP ＜源文件＞ ＜目的驱动器＞ /A /D:/M /S

磁盘备份命令，虽然支持软盘，但多数用于硬盘备份。

源文件可包括驱动器符和路径，缺席取当前值，支持文件通配符。目的驱动器仅指出驱动器符即可，无需标识文件。可选参数的意义如下：

/A 对原有的备份磁盘进行文件追加，不破坏已经存在的备份文件。

/D:月-日-年仅备份指定日期以后建立或修改过的文件。

/M 仅备份目的盘前一次备份以后建立或修改过的文件。

/S 指定备份包括源路径下的子目录。

使用 BACKUP 命令要注意几个以下问题。

(1)参数可多个装配，但/D、/M 只能选择一个。

(2)要求目的磁盘必须已经格式化。

(3)除非用/A 指定，否则将破坏目的盘上的所有文件。

(4)准备足够的磁盘，大量的备份文件是缺乏效率的。

(5)用 BACKUP 命令备份的文件，必须使用相同版本的 RESTORE 命令恢复，且路径必须相同，这是恢复备份文件的前提。如出现目录不能对应的情况，必须在恢复文件之前首先恢复目录结构。为了确保文件的恢复，必要时可复制 DOS 的基本文件和 RESTORE 外部命令，在软件交流时这种做法更有必要。

(6)备份磁盘的标签必须记录源文件的驱动器、路径、备份日期和 DOS 版本号。缺乏这些基本的信息，可能造成备份文件因为无法确定恢复现场而失去价值的后果。

C＞\DOS\BACKUP ＊.DBF B:/S↵

将当前目录及当前子目录中的所有 DBF 文件连同目录结构备份

到 B 驱动器的磁盘，不保留 B 驱动器中的原有文件。前导路径用于指示 BACKUP 文件所在的路径。此时的 BACKUP 驻存在\DOS\目录，要求备份的是当前目录。

C>\DOS\BACKUP *.PRG B:/S/A ↵

将当前目录及当前子目录中的所有 PRG 文件用追加的方式备份到 B 驱动器的磁盘，保留 B 驱动器中的原有文件。这是利用磁盘剩余空间的方法。

C>BACKUP *.* A: /D:06-20-91

将当前目录中的91年06月20日以后建立或修改过的所有文件备份到 A 驱动器的磁盘。此时的 BACKUP 必须在 PATH 指示的路径或存在于当前路径。

5—5 恢复磁盘备份命令

RESTORE <源驱动器> <目的驱动器符>
<指定恢复的文件> /P /S

恢复备份文件到原路径。文件名支持通配符。

此时的源驱动器指备份磁盘所在驱动器，目的驱动器指文件恢复到达的驱动器，这是不可缺省的参数。文件名参数略有变化，在格式上必须用空格与驱动器符分开，表示提取备份中的指定文件。恢复文件可覆盖设置成只读属性文件。

/P 限定恢复备份时间以后修改过的文件，即恢复备份文件时的状态。

/S 指定恢复包括子目录。

这是与 BACKUP 对应的恢复命令，如果在备份时的注意事项都做得完美，此时恢复就不成问题。如果缺乏必要的信息或环境则很难找到处理办法。

C>RESTORE A: C: SUB?.* /P ↵

恢复 A 驱动器中的备份文件中符合通配符特征的文件到 C 盘的

原有目录，/P 限定了这些文件必须是备份时间以后进行过修改的。如果装配/S 参数，则恢复范围将包括子目录。

注意这里的文件名格式与其它命令的区别。驱动器符不可再定义路径，其自动采用备份生成时的路径。

5—6 查询和设置文件属性命令

ATTRIB <通配符> [±A] [±R]

改变文件的档案位可选±A 参数。改变文件的只读属性可选±R 参数。

举例：用 ATTRIB 命令查询文件的档案位

C>ATTRIB *.* ↵

屏幕显示

```
A        C:\JJ\ZRED.EXE
A        C:\JJ\Z.BAT
A        C:\JJ\YU02.TXT
A        C:\JJ\JC001
```

A 是文件的档案位，依次是文件的驱动器、目录、文件名。系统在文件生成时自动定义了档案位的值。

通过+R 参数设置文件的只读属性，即

C>ATTRIB *.EXE +R ↵

C>ATTRIB ?.* +R ↵

再次查询，屏幕显示

```
A    R   C:\JJ\ZRED.EXE
A    R   C:\JJ\Z.BAT
A        C:\JJ\YU02.TXT
A        C:\JJ\JC001
```

出现的 R 是文件的只读标识。可见设置只读属性成功。

如果在 XCOPY 命令中使用了/M 参数(见下节)，拷贝后对

文件属性进行观察，可以看出文件的档案位被设置成空位。

C>ATTRIB *.* ↵

屏幕显示

C:\JJ\ZRED.EXE

C:\JJ\Z.BAT

C:\JJ\YU02.TXT

C:\JJ\JC001

恢复档案位的方法是使用+A 参数，即

C>ATTRIB *.* +A ↵

此时的文件档案位被重新标识，可观察

C>ATTRIB *.* ↵

屏幕显示

A C:\JJ\ZRED.EXE

A C:\JJ\Z.BAT

A C:\JJ\YU02.TXT

A C:\JJ\JC001

5—7 拷贝命令

XCOPY ＜源通配符＞ ＜目的通配符＞/A /M /D: /P /E /S /V /W

这是外部拷贝命令，适用于3.2 以上的 DOS 版本。

有关文件参数表的使用同上。各可选参数意义如下：

/A——只拷贝设置了档案位的文件，拷贝后档案位不变。

/M——只拷贝设置了档案位的文件，拷贝后档案位撤销。

/D:—月-日-年，只拷贝指定日期及以后的文件。

/P——在文件的拷贝过程中，先显示文件名，操作员可用 Y 和 N 键确定该文件是否进行拷贝。

/E——在目的盘建立与源盘子目录对应的目录结构。此参数

不理睬当前目录中的文件。

/S——在拷贝当前目录中源文件的同时，连同下级子目录一起拷贝。无此参数时不理睬子目录。

/V——在拷贝过程中对目的文件进行校验。

/W——在拷贝前等待按任意键后开始。

从功能上看，该命令大约介乎于 COPY 和 DISKCOPY 之间。从可装配的参数来看，它可提供更多的文件信息作为后备数据使用，多参数的选择装配可使文件分类更灵活，拷贝方式更丰富，操作员的自由度更充分。前两种拷贝命令所不能支持的状态，在这里得到补充。

XCOPY 命令在执行时，先将源文件尽可能多的读入内存，然后再集中写到目的驱动器。对于复制文件字节较少的多个文件，具备较高的效率；如果文件数目少，且每个文件的字节数又很大，这种拷贝就失去了速度上的优势。

档案位是DOS 系统在文件生成时自动记入的标志位，它包括在文件名信息之中。参数/M 与 ATTRIB 命令的配合可用档案位作为拷贝控制的条件。

关于文件匹配与前面基本相同，这里着重讨论目录的转移和复制。现 C 盘有目录结构如图9：

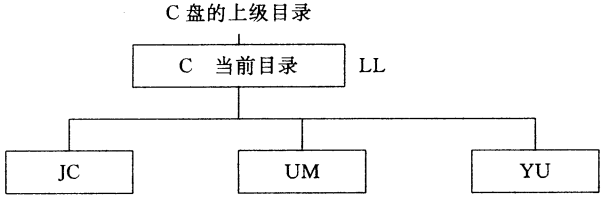


图 9

C>XCOPY *.* B:/S

屏幕显示如下：

Reading source file(s)…

LL
L1
JC\JC.D3
JC\JC.D1
JC\JC.D4
JC\JC.D2
UM\GYUM
UM\UJUM
UM\JJUM
UM\JGUM
YU\YU01
YU\YU02

12 File(s) copied

可见,XCOPY 是在执行时拷贝了子目录中的文件,其实是连同目录结构一起拷贝的。可用命令查询 B 盘(见图10)

C>DIR B:*.*

屏幕显示

Volume in drive B has no label

Directory of B:\

LL	167	7—06—91	11:47a
L1	521	7—06—91	2:49p
JC	<DIR>	7—06—91	3:03p
UM	<DIR>	7—06—91	3:03p
YU	<DIR>	7—06—91	3:03p

5 File(s) 258048 bytes free

通过磁盘的观察，证实了目录结构的拷贝是成功的。如果B盘对应的目录存在，文件被拷贝到相应的目录之下，否则将在B盘创建对应的目录结构。还有一点，要领会的是C盘的LL目录与B盘的根目录形成了对应关系。这一点在查询结果上很清楚（见图10）。

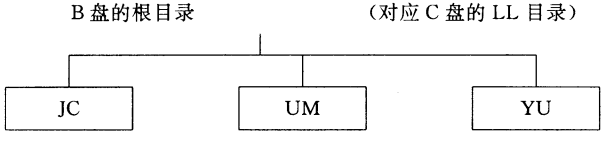


图 10

认为只能在根目录对应的情况下才能选择/S参数的认识是错误的。这种仅以当前目录相互对应的拷贝会给目录结构的剪接提供方便。同理，从B驱动器向C驱动器拷贝仍然如此。

当XCOPY存在于软盘，驱动器又必须用来执行XCOPY命令，选择/W参数可以在XCOPY装载后，提供一个更换磁盘的机会。

A>XCOPY A:*.* B:/W

命令装载完成并不立即执行，屏幕提示

Press any key to begin copying file(s) Y

在更换磁盘后，按任意键，才开始执行拷贝。

/P参数可对匹配于通配符的文件提供一个再次确认的机会，在每个文件的拷贝之前，屏幕显示文件名。回答Y，该文件被拷贝；回答N，该文件被放弃。

LL (Y/N)? Y

L1 (Y/N)? N

1 File(s) copied

通过查询可证实这种拷贝的结果。

关于/M 参数请参看 ATTRIB 命令，体会文件档案位是怎样被撤消和恢复的。实际上，重复的使用带有/M 参数 XCOPY 可不重复的拷贝文件。这种拷贝比 BACKUP 命令更方便，它不用专门的恢复命令。但如果单个文件的长度超过了一张磁盘的容量，还是用 BACKUP 方便一些。

其他参数参考 REPLACE 命令。

5—8 文件更新命令

REPLACE ＜源文件＞ ＜目的文件＞ /A /P /R /S /W

该命令的作用和使用方法和 XCOPY 很类似。不再过多解释。

/A 指定替换目的盘上没有的文件。同名文件被忽视。

/P 指定在替换前给出一个可供确认操作的选择。

/R 指定替换包括只读文件，否则将忽视。

/S 指定在所有子目录中搜寻同名文件再进行替换。

/W 指定开始替换前给出一个等待的提示，按任意键开始。

A＞REPLACE ＊.90? C：/S /P /R

用 A 盘中满足通配符指定的文件替换 C 盘的同名文件。

/S——指定替换包括与 A 盘对应的子目录。

/P——在每个替换前给出确认提示，仅对回答 Y 的文件替换。

/R——如果文件中用只读文件，将被重新改写。

XCOPY 、ATTRIB、REPLACE 命令的配合使用是十分生动的，掌握了这些不同的文件交换方法，拷贝就会得心应手。

备份和拷贝、替换在对文件的管理上有差别。局部的文件操作用拷贝和替换比较方便，对整版的文件复制，备份的方式好一些。

5—9 磁盘检测命令

CHKDSK /F /V

/F——追踪文件中丢失的文件簇并记录成新的文件，命名为FILEnnnn.CHK。nnnn是从0001开始的序列号。

/V——指定显示磁盘上所有文件的名称和路径。

这是一个功能性非常强的命令，详尽的解释还要说明很多概念，这里仅对该命令的查盘报告做简单的介绍。

C>CHKDSK /V ↵

从根目录显示所有路径及路径中的文件名。

C>CHKDSK ↵

屏幕显示报告如下：

21344256 bytes total disk space	磁盘总容量字节数
57344 bytes in 4 hidden files	隐含文件占用字节
51200 bytes in 19 directories	目录管理占用字节
10887168 bytes in 454 user files	用户文件占用字节
399360 bytes would be in	丢失的文件字节数
10 recovered files	丢失的文件数

20480 bytes in bad sectors	坏的磁盘字节
9928704 bytes available on disk	磁盘可用空间字节
655360 bytes total memory	实际内存字节数
426400 bytes free	可用内存字节数

这是一个 CHKDSK 的实际报告，有关文件跟踪请参看 DOS 的说明。

第六章　DOS 系统的管道功能

6—1　管道概念

在常用的 DOS 设备中，键盘仅属于输入设备，打印机仅属于输出设备，显示器和驱动器（包括软盘和硬盘）属于输入和输出的共用设备。具体属于哪一种要视命令的要求而定。

在 DOS 命令的执行过程中，经常涉及到标准设备的概念，对于不同的 DOS 命令，使用的标准设备有所不同。

例如：

DIR 命令默认的标准输出设备是显示器。

COPY 默认的标准设备是驱动器磁盘。

还有一些命令要求同时使用输入、输出设备。例如：

MORE 标准的输入设备是键盘，标准的输出设备是显示器。

TYPE 标准的输入设备是驱动器中的磁盘文件，标准的输出设备是显示器。

如此强调标准设备是不无道理的，标准设备是系统默认的使用设备，并不是仅可使用的设备。很多 DOS 命令允许用户在使用时对标准设备进行重新指定。如果用户提出的变更请求被当前的 DOS 命令确认为合法，那么 DOS 会将有关的操作由标准设备转移到用户指定的设备。这种转移标准设备的操作是通过 DOS 系统的管道功能完成的。管道可作用于 DOS 命令、磁盘文件或设备。

可用于 DOS 设备的转向管道有以下几种：

＜　转入管道，可接通对 DOS 命令的输入。可将一个 DOS 命令的标准输入设备（或磁盘文件）转移到一个指定设备（或磁盘文件）。

> 转出管道，可接通DOS命令的输出。可将一个DOS命令的标准输出设备转移到另一个DOS命令的输入或重新指定的DOS输出设备或写成磁盘文件。如果文件存在将被覆盖，否则将被创建。

>> 管道作用和> 相同，区别在于对目的文件的处理方式。该管道对存在的文件不覆盖，将DOS的输出内容连接到指定文件的尾部，只有当文件不存在时，才对文件进行创建。

| 连接DOS命令的输入和输出，或说将一个DOS命令的输出转向另一个DOS命令的输入。

PRN：和CON：都是DOS的设备名，它们可直接作为指定DOS设备的命令参数使用，也可作为管道的目的设备，为了简化章节，在这里一并介绍。

6—2 必须由管道支持的命令

1. DOS的字符串检索

FIND [/N] [/C][/V] "字符串"

DOS的字符串查询，从标准的输入设备上读取文件并在文件中查询指定的字符串，将查询结果写到标准的输出设备上。

/N 显示所有包括字符串的行号；

/C 显示包括字符串行的个数；

/V 显示所有不包括字符串的行。

文件名支持驱动器符和路径，但不支持通配符。

特征字符使用必须严格，大小写具备不同的定义。

2. DOS的分页

MORE

从指定的输入设备上读取信息并进行分页处理，再写到指定的输出设备上。

3. DOS的排序

SORT [/R] [/+N]

DOS 排序，从标准的输入设备上读取文件，排序后再写到标准的输出设备上。

/R　　指定排序的降序方式，否则为升序方式；

/+N　　指定排序从第 N 列开始，否则从第一列开始。

6—3　输出管道和设备符的实际应用

DOS 管道在处理实际问题中有一定的作用，尤其是在一些不能调用 DOS 中断的高级语言中，可利用 DOS 管道提取一些有关的 DOS 信息到程序中去。

下面对 DOS 管道的使用举些实际的例子。

1. 将屏幕输出转移到磁盘文件

先查询 B 驱动器中的磁盘

C>DIR B:

屏幕显示：

Volume in drive B has no label

Directory of B:\

GYL	LX	2356	6—08—89	2:47p
TMC	LL	5522	6—08—89	3:30p
AML	LL	10431	7—08—89	8:56a
ZRM	LX	16375	7—09—89	1:13a
CCED	BLK	1152	7—29—91	11:17p
CCED	EXE	86962	7—30—91	12:16a
CCED	OVL	34944	1—13—89	9:55p
CCED	HLP	18560	6—13—91	2:58p
KMX		128	7—11—90	4:16a

9 File(s) 180224 bytes free

为了叙述问题的方便，将此信息作为下面演示的基础。这是B盘文件的屏幕报告，此时的标准输出设备是显示器。如果在处理实际问题时要求输出设备不再是显示器，可用设备名或管道转移输出到由用户指定的目的设备。

C>DIR B: >WJMB. TOD

由于此处使用了输出转向管道，输出结果将由屏幕改成磁盘文件。指定文件名是表面的现象，更重要的是在指定文件名的同时转移了DOS的标准输出设备。

此时，屏幕上不再出现文件目录清单，屏幕应当显示的所有信息将被抄录到指定的磁盘文件中。在指定文件名时，同样可指示驱动器和路径表。在上面的命令中，文件被创建在当前的目录中。

为了检查执行结果，可显示WJMB. TOD的内容，即

C>TYPE WJMB. TOD

屏幕显示：

Volume in drive B has no label

Directory of B:\

GYL	LX	2356	6—08—89	2:47p
TMC	LL	5522	6—08—89	3:30p
AML	LL	10431	7—08—89	8:56a
ZRM	LX	16375	7—09—89	1:13a
CCED	BLK	1152	7—29—91	11:17p
CCED	EXE	86962	7—30—91	12:16a
CCED	OVL	34944	1—13—89	9:55p
CCED	HLP	18560	6—13—91	2:58p
KMX		128	7—11—90	4:16a

9 File(s) 180224 bytes free

WJMB. TOD的内容与屏幕显示的结果完全一致。

由此产生的文件是 ASCII 文件。在很多高级语言中虽然不能调用 DOS 中断，但支持对 ASCII 码文件的读写。因此，这种记录 DOS 输出的 ASCII 码文件就成了连接其它高级语言的中间桥梁。

与此相同，凡是可产生屏幕输出报告的 DOS 命令，都可用此方法将屏幕报告转移到磁盘文件。作为信息的保存，文件要比单纯的屏幕显示有更大的价值。

再举一个很有实用价值的实例

C>CHKDSK B: >CPJIBG

检查 B 磁盘并将检查报告写到文件 CPJIBG 中去。显示文件内容如下：

```
C>TYPE CPJIBG
      362496 bytes total disk space
      182272 bytes in 9 user files
      180224 bytes available on disk

      655360 bytes total memory
      425104 bytes free
```

这是一个用 ASCII 码形式记录的磁盘检查报告，磁盘一共有362496个字节，有9个用户文件占用182272字节，还有180224字节可以使用，机器的内存一共有655360字节，目前没有被占用的有425104字节。

很多使用 DBASE 语言的用户，经常会为如何获取软盘上的自由空间字节数大费苦心而不得结果，其实通过 RUN 调用 DOS 命令并产生文件形式的报告后，用数据库处理这个报告将容易地获得需要的信息。

再介绍一个实用的例子：若硬盘上有复杂的目录结构并有很多的文件，如果已经知道一个文件名，不知道该文件是否存在于硬盘，此时要迅速的得到肯定的结果，用什么办法呢？

通常的办法是用 DIR 命令查看目录中的文件，这种方法虽然

简单，但缺乏效率。准确快速的方法是用 TREE/F 或 CHKDSK/V 将屏幕结果转移到指定的文件，再用 EDLIN 中的 S 命令查询，即可知道该文件是否存在，以及文件所在的路径。

例如从硬盘上查询一个名为 FTWW.DBF 的文件

C>TREE /F >WJQD

送 DOS 查询结果到磁盘文件 WJQD。

C>EDLIN WJQD　　用行编辑来查询 WJQD 文件的内容。

屏幕显示

End of input file

```
*SFTWW.DBF
     308                                FTWW.DBF
*S
     504                                FTWW.DBF
```

上面的操作结果表明，在 WJQD 文件中的388行和504行有 FTWW.DBF 文件。

由此说明，在磁盘的两个目录中存在 FTWW.DBF 文件。那么在哪两个目录中呢?继续执行

```
*S
Not find

*lrPath^ ZPath
      15 Path:\CCDOS
      87 Path:\PCDOS
     121 Path:\ZRM
     154 Path:\FOX
     243 Path:\DBASE3
     309 Path:\DBF
```

488 Path:\DBL

在 EDLIN 中两次使用 S 命令查询文件存在，用 R 命令确定了文件所在的路径。在309行到488行之间的数是 DBF 子目录下的文件，在488行以后的数是 DBL 子目录下的文件，而388行在309到488之内所以在 DBF 子目录中有一个 FTWW.DBF 文件。以此推断出 DBL 子目录中有另一个 FTWW.DBF 文件。现在可以肯定的说该文件在硬盘上有两份，它们分别存在于\DBF 和\DBL 路径中。

COPY 命令标准的读写设备是驱动器，标准的显示设备是显示器，如果使用管道和设备名，这些标准设备都可重新指定。如

C>COPY A:*.* B: >WJQD

这里改变了标准的显示设备，拷贝的提示信息被记录到磁盘文件 WJQD。

如果从 A 到 B 拷贝多张磁盘并要求记录所有的文件名到 WJQD 中，可通过接续管道达到目的，在后续命令的形式如下：

C>COPY A:*.* B: >>WJQD

C>COPY A:*.* B: >>WJQD

由于使用了接续管道，记录屏幕的工作被连续进行，显示文件内容可得到三次拷贝的完整报告。

C>TYPE WJQD

```
GYL        LX
TMC        LL
AML        LL
ZRM        LX
        4 File(s) copied
CCED       BLK
CCED       EXE
CCED       OVL
```

```
CCED        HLP
KMX
            5 File(s) copied
JGUM        LLL
KVUM        LLL
MLUM        LLL
05UM        TXT
06UM        TXT
07UM        TXT
JC          D3
            7 File(s) copied
```

这个命题的本身不一定有很大的价值，但这种使用 DOS 管道的方法是有价值的，尤其是对大批文件的统一操作时，它的意义就会更明显。

2. 将 DOS 输出引导到打印机 用管道可改变输出设备到文件，是否还可到其它设备呢，回答是肯定的。

例如：如果需要打印 DOS 命令的输出结果，启动 DOS 的键盘控制指令 ^ P 固然可以，打印完成势必要再进行撤消的操作，如果出现打印机空纸，而打印状态尚未撤消，键盘操作将可能会出现锁定，给操作带来很多不必要的麻烦。如果使用设备名指定的方式启动打印机，可能会使操作更简单、更可靠。

```
C>DIR B: PRN:
```

该命令可将 B 盘的文件信息直接送到打印机。这里虽然没有使用管道，但指定了使用的 DOS 设备是打印机。

同理可用显示或拷贝命令启动打印机，即

```
C>COPY  WJQD >PRN:
C>COPY  B:CCED.HLP >PRN:
C>COPY  B:*.TXT >PRN:
```

这里的打印不在屏幕上显示，由于 COPY 命令支持通配符，这就

意味着该命令一次可送一批文件到打印机，此时文件的连接可能不尽人意，但对于格式要求不高的文件，的确提供了一个非常简单的操作方式。当用打印设备符 PRN 代替了命令的目的文件名时，管道符号“>”可以用空格代替。应当指出的是，如果用指定设备名的方式打印文件，该文件仅限于 ASCII 码文件，其它二进制或可执行文件将会引起打印机失控或计算机死机。可用 ^ Break 键随时终止打印的进行。

COPY 文件到打印机时，文件内容不在屏幕上显示，如果要求显示可改用 TYPE 命令，即

C>TYPE　WJQD >PRN:

C>TYPE　B:CCED.HLP >PRN:

由于 TYPE 命令不支持通配符，打印只能单个进行。再有，这里采用了两个输出设备，一个是显示器，一个是打印机，这时的管道符号“>”是必须要使用的，否则不能达到同时操作两个设备的目的。

3. 对输入设备的改变　不但输出设备允许改变，输入设备同样可以重新指定。可指定的专用输入设备只有键盘，但它必须和显示器共同完成信息的接收，前面已经介绍了可指向键盘的设备符 CON:，请看实例

C>COPY　CON: CONFIG.SYS

此命令的意图是将拷贝命令的标准输入设备由驱动器改成键盘，系统应答后，可在屏幕上接受键盘输入的字符数据，直到发现用户输入 ^ Z 键并回车时认为输入结束。屏幕输入的数据将被拷贝到指定的输出设备，这里指定的是磁盘文件 CONFIG.SYS。

CONFIG.SYS 是系统配置文件，一般命令行不多，用拷贝键盘屏幕的做法是建立该文件的一种简单方便的形式。如果键盘输入的命令是

DEVICE=ANSI.SYS

DEVICE=VDISK.SYS 384 512 112 /E

BUFFERS=25

FILES=36

^Z↓

这些命令将被写入创建(或覆盖)的 CONFIG.SYS 文件中。当然使用其它的编辑功能同样可达到目的。

C>COPY CON: CONFIG.SYS >PRN:

这条命令将屏幕数据写入文件并同时在打印机上输出。

6—4 DOS 的查询、排序及分页命令

1. DOS 的检索查询 检索命令 FIND 的输入设备必须是指定的,否则 FIND 命令将会因无输入数据而不能工作。

在上面的文件搜寻的命题中,使用了 EDLIN 辅助查询,这是个费事的作法。如果用 DOS 的检索命令处理,会使操作简捷一些。命题是在硬盘上搜寻所有在文件名中带有字符"CS"的文件。

C>TREE /F >WJQD

将文件名抄录到磁盘文件 WFQD,调用检索命令查询指定特征的文件。

C>FIND <WJQD "CS"

GRAPHICS.COM

CS .COM

CSUB0 .DBF

CSUB0 .BAK

这是在屏幕上产生的检索报告,该命令没有指定输出设备,检索命令使用默认的输出设备显示器。这种查询只能指示文件是否存在,但不能确定文件的路径,要想得到更确切的信息,可用参数的限定和输出管道。

C>FIND /N <WJQD "CS" >LL

C>TYPE LL

```
[46]          GRAPHICS.COM
[125]         CS          .COM
[220]         CSUB0       .DBF
[232]         CSUB0       .BAK
```

检索命令使用了参数/N，意图是在搜寻结果中反映出字符串所在行的行号。为了保存屏幕信息，将输出报告转移到文件 LL，用 TYPE 显示文件内容，可证明达到了目的。为了检索到路径，再执行一次检索。

```
C>FIND /N <WJQD "Path" >>LL
```

总之这是旨在检索路径的查询，为了比较方便，使用了追加的接续管道，目录的查询结果将被连接到 LL 文件的后面。

再次显示 LL 文件可得到完整的查询结果。

```
C>TYPE  LL

[46]          GRAPHICS.COM
[125]         CS          .COM
[220]         CSUB0       .DBF
[232]         CSUB0       .BAK

[15]Path:\PCDOS
[74]Path:\213
[139]Path:\FOX11
[153]Path:\FOX
[167]Path:\RS
[335]Path:\ZR
[367]Path:\JC
```

分析两次的屏幕报告可明确知道，硬盘中有四个带有 CS 的文件，存在的路径也很清楚。

```
GRAPHICS.COM    存在于\PCDOS
CS        .COM    存在于\213
CSUB0     .DBF    存在于\RS
CSUB0     .BAK    存在于\RS
```

2. DOS 的分页处理命令 MORE 的输入输出管道与 FIND 相同。

使用 TYPE 命令显示文件的内容往往会感到不方便，因为屏幕连续显示的过程很快，为了看清屏幕内容，经常要用 ^S 键控制暂停，稍有失误就会使显示失控。如果使用 MORE 命令将会改变这种状态。下面是一个观察文件的实例：

```
C>MORE <WJQD

DIRECTORY PATH LISTING

Files:          COMMAND   .COM
                AUTOEXEC  .BAT
                AU213-11  .BAT
                ANSI      .SYS
                CONFIG    .SYS
                VDISK     .SYS
                ZR-V      .COM
                SCAN      .EXE
                213       .BAT

Path: \PCDOS

Sub-directories:None

Files:          COMMAND .COM
```

FORMAT .COM
AUTO .BAT
README .DOC
4201 .CPI
5202 .CPI
⋮
MORE

此后每显示24行屏幕自动暂停，并显示 MORE，这时可有充裕的时间观察屏幕信息，等到按任意键再继续显示下一屏，也可以说是下一页，直到整个文件显示完成。同样 ^ Break 键可中断显示。

当有文件输入到 MORE 后，它采用的标准输出设备是显示器。MORE 支持输出管道，如果仅使用输出管道时，它标准的输入设备是 CON:，但一般很少使用。它可将键盘输入的信息到文件。

C>MORE >LXWJ

可在屏幕接收信息数据到文件 LXWJ，此时的状态与 COPY CON : 相同，区别仅在于每24行有一个换页符。

3. DOS 的排序命令 排序命令 SORT 可对输入的文件行序列做排序处理，当有输入设备指定时，标准的输出设备是显示器，也可指定打印机或文件名。下面用一个屏幕信息文件做些演示，以说明 SORT 的用途。

建立一个用于演示的文件：

C>DIR B: >WJMB. TOD

显示该文件中的数据：

C>TYPE WJMB. TOD

Volume in drive B has no label
Directory of B:\

GYL	LX	2356	6—08—89	2:47p
TMC	LL	5522	6—08—89	3:30p
AML	LL	10431	7—08—89	8:56a
ZRM	LX	16375	7—09—89	1:13a
CCED	BLK	1152	7—29—91	11:17p
CCED	EXE	86962	7—30—91	12:16a
CCED	OVL	34944	1—13—89	9:55p
CCED	HLP	18560	6—13—91	2:58p
KMX		128	7—11—90	4:16a

9 File(s) 180224 bytes free

这是前面用过的例子，现在要求对该文件的内容进行排序处理，可通过屏幕观察到 SORT 的工作结果：

C>SORT <WJMB. TOD

9 File(s) 180224 bytes free

Directory of B:\

Volume in drive B has no label

AML	LL	10431	7—08—89	8:56a
CCED	BLK	1152	7—29—91	11:17p
CCED	EXE	86962	7—30—91	12:16a
CCED	HLP	18560	6—13—91	2:58p
CCED	OVL	34944	1—13—89	9:55p
GYL	LX	2356	6—08—89	2:47p
KMX		128	7—11—90	4:16a
TMC	LL	5522	6—08—89	3:30p
ZRM	LX	16375	7—09—89	1:13a

用输入管道将文件内容送入 SORT，结果被输出到在默认设备显示器上。由于没有使用参数，排序从每行的开始进行比较。这是文件名的有序排列。

参数/+N 限定排序从行的第 N 列开始，前导字节将忽视，这就为排序提供了不同的起点。

```
C>SORT /+9 <WJMB.TOD
```

```
KMX          128     7-11-90     4:16a
CCED  BLK    1152    7-29-91    11:17p
GYL   LX     2356    6-08-89     2:47p
TMC   LL     5522    6-08-89     3:30p
AML   LL    10431    7-08-89     8:56a
ZRM   LX    16375    7-09-89     1:13a
CCED  HLP   18560    6-13-91     2:58p
CCED  OVL   34944    1-13-89     9:55p
CCED  EXE   86962    7-30-91    12:16a

Directory of B:\
        9 File(s) 180224 bytes free
Volume in drive B has no label
```

限定排序从第九位开始，即以后缀名排序。

```
C>SORT /+14 <WJMB.TOD >WJZJPX
```

限定排序从第十四位开始，即以字节数排序。

这里增加了输出管道，对于产生输出的 DOS 命令，输出管道总是可以使用的，排序结果将避开屏幕直接写到文件 WJZJPX。显示文件内容：

```
C>TYPE  WJZJPX
KMX          128     7-11-90     4:16a
```

```
CCED    BLK      1152     7—29—91      11:17p
GYL     LX       2356     6—08—89       2:47p
TMC     LL       5522     6—08—89       3:30p
AML     LL      10431     7—08—89       8:56a
ZRM     LX      16375     7—09—89       1:13a
CCED    HLP     18560     6—13—91       2:58p
CCED    OVL     34944     1—13—89       9:55p
CCED    EXE     86962     7—30—91      12:16a
Directory of B:\

                9 File(s) 180224 bytes free

Volume in drive B has no label
```

这个新文件是 WJMB. TOD 经指定位排序后产生的。

```
C>SORT /+24 <WJMB. TOD

Directory of B:\

CCED    OVL     34944     1—13—89       9:55p
GYL     LX       2356     6—08—89       2:47p
TMC     LL       5522     6—08—89       3:30p
CCED    HLP     18560     6—13—91       2:58p
AML     LL      10431     7—08—89       8:56a
ZRM     LX      16375     7—09—89       1:13a
KMX       .       128     7—11—90       4:16a
CCED    BLK      1152     7—29—91      11:17p
CCED    EXE     86962     7—30—91      12:16a
                9 File(s) 180224 bytes free
```

Volume in drive B has no label

限定排序从第二十四位开始，即以日期排序。

4. 可用于连接 DOS 命令的管道 前面分别介绍了 DOS 的输入输出管道，在不同的 DOS 命令之间的联系是通过文件桥梁达到连接不同的 DOS 命令的目的。实际上 DOS 系统本身就具备连接命令的转向管道，即便不使用文件过渡，同样可达到 DOS 命令的有效组合。

C>DIR B: |SORT /+9

在输出结果上完全等价于

C>DIR B: >WJMB.TOD

C>SORT /+9 <WJMB.TOD

此处要着重领会什么叫转向。输出、输入侧重于对使用设备的指定，转向管道则侧重于将一个 DOS 命令的输出指向另一个 DOS 命令的输入，或是说侧重于构成两个 DOS 命令之间的联系。

分析上面的例子，WJMB 是 DIR 的输出，也是 SORT 输入，文件只是起到了过渡两个 DOS 命令的桥梁作用，如果对文件的存在不是特别关心，就大可不必使用文件过渡，直接使用 DOS 的转向管道将会更简单。

在实际的应用中，对 DOS 的输入、输出管道一般感到容易理解，很少在使用中出现错误。但对于转向管道就不同了，经常要实验几次才能达到目的，究其原因，仍然是对 DOS 标准设备的认识上概念含糊。如果能够清楚地了解每个 DOS 命令的输入、输出设备，就可准确无误的使用转向管道。

为了加强概念，下面对一些命令的组合给出设备使用上的解释，不再一一录出演示结果。

C>DIR |FIND "<DIR>"

将 DIR 的输出转向 FIND 的输入，并在 FIND 的标准设备上输出

结果，即在显示器上输出。显示的内容将是当前路径中的所有子目录信息。

C>TYPE WJQD.DOT |MORE

将显示文件的内容转移到分页处理后再输出。虽然 TYPE 和 MORE 的标准输出设备都是显示器，转向的目的在于分页处理。

在上述命令后继续连接输出管道仍然是合法的，形式如下：

C>DIR |FIND "<DIR>" >LLPT

C>TYPE WJQD.DOT |MORE >>LLPT

两个命令的输出均由屏幕转移到文件 LLPT。

再重新分析已经演示过的实例：

C>TREE /F >WJQD

C>FIND /N <WJQD "CS" >LL

C>FIND /N <WJQD "Path" >>LL

这是在中间文件的支持下完成的文件检索，请看在使用转向管道的情况下的命令形式：

C>TREE /F |FIND /N "CS" >LL

C>TREE |FIND /N "Path" >>LL

TREE 的输出直接送到 FIND 的输入，FIND 的输出也在管道的作用下由屏幕转向了磁盘文件，这是为了最后查询的方便。

5. FIND 命令的进一步应用 命题是这样开始的，如果需要在一批文件中同时检索一个字符串，FIND 会被多次执行，重复的键盘输入或编辑即费事又容易造成遗漏或错误，此时使用 DOS 的过程迭代，达到目的会容易的多。

迭代命令的基本格式是

FOR < 参变量> IN（通配符）DO <命令> < 参变量>

对它的意义可粗糙地理解为对满足（通配符）指定的文件逐个执行指定的 <命令>，参变量是实现逐个替代文件名的表达形式，它必须用“ %字母”定义，如果用于批文件，则必须用双百分号“%%字母”的形式。

下面是一个有实际意义的具体命题的实现形式：

```
C>FOR %F IN (*.PRG) DO FIND /N "PROC" %F
    >>IXJG
```

该命题旨在所有的.PRG 文件中搜寻带有 PROC 字符串的命令行。对于熟悉 DBASE 的用户会理解,.PRG 是程序文件,PROC 是程序文件中的子过程。当实际工作要求确切的知道子过程的数量和命名时,该命令可使你迅速达到目的。

再有,经常使用 C 或 FORTRAN 等语言的人,大都要有一个属于自己的程序库,时间一长,对子程序的作用、参数及程序完成的任务会遗忘,再逐个的审查源程序将会感到得不偿失。为此很多人将程序的任务、参数、调用规范写到程序内,以备查询。即便如此,逐个的打开文件检查仍然是个效率极低的办法。如果使用上述的迭代过程查询所有程序中的注释行,整个函数库的所有说明将会一目了然。

```
C>FOR %F IN (*.FOR) DO FIND /N "C      " %F
    >>IXJG
```

其中的字符串"C　　"是*.FOR 程序中注释行共同具备的一个字符特征。该命令可翻阅出所有的.FOR 程序中含有字符串"C　　"的物理行,如果这一行中有关于程序的注释,就相当于用一条命令翻阅了所有的程序。这种用法还可延伸到其它类似的应用中去。

这是基于实用的粗浅介绍。DOS 迭代的更多用途不再详叙。

为了学习方便和使教材具备一定的系统性,第九章准备了完整的 DOS 3.30 版本命令集,以供查阅。

第七章　DOS 的批处理文件

7—1　批处理文件的概念

前面介绍的 DOS 命令都是在交互状态下使用的，所谓的交互状态是用户从键盘发出一条命令，计算机就立即去执行并报告执行的结果。这种逐条执行的状态被称为交互状态。

交互状态的最大不足是每次执行的命令都必须是当时从键盘上输入的。如果有一条或几条命令经常要求计算机执行，在交互状态下，势必每次执行都要进行重复的输入。

是否可以找到更简便的方法呢？

这就是批处理的概念。DOS 的批处理工作方式可以避免每次都从键盘读取指令，改成从磁盘文件中读取，这个磁盘文件被称为批处理文件。不难想象，批处理文件是一组 DOS 命令的组合序列。一旦启动了批处理文件，DOS 将逐个地执行批处理文件中的命令。这是批处理的一个含义。

还有一个含义是在批处理文件中，可使用 DOS 的判断、转移和过程的迭代，因此一些在交互状态下难以解决的问题在批处理的过程中可以解决。或者说，批处理形式比交互状态在处理问题的方式上具备更多的手段和更强的能力。

7—2　批处理文件的建立和执行

如何将 DOS 命令组合成文件的形式，这个过程叫 DOS 编程，这种文件叫批处理文件，它的后缀名必须是. BAT。前面涉及到的所有 DOS 命令都可用于批处理文件。

请看一个批处理文件建立方法和工作过程的示意，文件名是BBUJ.BAT，目的是显示DOS的版本号及变更系统的日期和时间。

```
C>COPY CON: BBUJ.BAT
ECHO OFF
ECHO          请检查DOS的版本并输入日期、时间
VER
DATE
TIME
^Z
```

这里的批处理文件是用屏幕拷贝的方法创建的，当然用任何一种编辑软件都可达到这一目的。屏幕上输入的命令序列被记录到磁盘中一个名为BBUJ.BAT的文件中。每当需要执行这组命令时，只要在本路径中输入该文件名并回车即可。

如果批处理文件不在当前路径，DOS外部命令的调用规范同样适用于批处理文件的启动。

```
C>BBUJ

     请检查DOS的版本并输入日期、时间
MS-DOS Version 3.30
Current date is Sat 8-03-1991
Enter new date (mm-dd-yy):
Current time is 7:26:45.01
Enter new time:
```

这个屏幕结果应当是很熟悉的，它基本相当DOS启动时的情况。在文件中ECHO是尚未介绍的命令，它的标准形式是

ECHO ON/OFF/字符串

它定义在执行批处理文件时，是否显示将要执行的DOS命令。如

果后续是一个字符串，无论显示是ON/OFF，该字符串被显示到屏幕，它可用于批处理中的提示。

将文件中的ECHO改成ON后，再执行BBUJ，观察其区别。

```
C>BBUJ
C>      ECHO ON
C>      ECHO      请检查DOS的版本并输入日期、时间
        请检查DOS的版本并输入日期、时间
C>      VER
MS-DOS Version 3.30
C>      DATE
Current date is Sat    8-03-1991
Enter new date (mm-dd-yy):
C>      TIME
Current time is  7:26:45.01
Enter new time:
```

7-3 可用于批处理文件的DOS命令

1. REM [字符串]

显示批处理期间的说明，最长123字节，只有在ECHO ON时显示才被放到屏幕上。

2. PAUSE [字符串]

对字符串的显示同REM，它主要目的是暂停批处理文件的执行，在屏幕上显示

Strike a key when ready ?

按任意键继续执行。

3. ECHO ON/OFF/字符串

设置显示开关或显示字符串，字符串长度117字节。

4. GOTO <标号>

转移到批处理文件中指定的标号处，继续执行标号后的命令。标号必须是":" 开头，后续任意字符，仅前 八位有效。

5. IF [NOT] <条件> <DOS 命令>

根据条件成立与否决定是否执行指定的 DOS 命令。可使用的条件有

①ERRORLEVEL n

其中 n 为预先指定的限定值。在每个 DOS 命令执行后，有一个指示执行状态的出口值返回给用户，这就是 DOS 命令的状态编号，当此编号大于等于指定值 n 时，IF 认为条件成立。DOS 命令返回状态编号可参看有关的 DOS 手册。

②EXIST < 文件名>

当文件存在时，IF 认为条件成立。

③字符串1==字符串2

当两个字符串完全相等时，IF 认为条件成立。

NOT 可对使用的条件取反。

6. FOR <%%参变量> IN (文件名序列) DO

<DOS 命令><%%参变量>

文件名序列可以是一个实际的文件名表，每个文件名用一个以上的空格分开，也可以使用文件通配符对文件进行指定。这些指定的文件是 DOS 命令处理的对象。

用<%% 字母> 可指定一个参变量，文件名序列中的每个文件逐个被送到这个参变量中。

如果用于交互命令，参变量将用一个% 指定。

7. SHIFT

左移批处理参数。

在启动使用批处理文件时，经常需要临时指定一些参数，用于加强批处理文件的适应性。通过哑参数 %0 — %9 的运用，可将这些信息带到批处理文件内部。其中%0指批处理文件本身，%1 — %9 可由用户随机定义。

当使用10个以上的参数时，SHIFT 可将参数序列逐个左移。

7—4 批处理文件的应用

对于无控制语句的批处理文件形式，7—2节中已作介绍，只要区别不同的命题写上相应的命令即可。如果必须进行条件控制，请看几个条件转移的范例

例1：

```
IF EXIST \213\CS.COM \213\CS
```

如果213 路径中存在 CS.COM 文件，执行该文件。

例2：

```
IF EXIST \213\CS.COM GOTO :CNC
CHKDSK
GOTO :END
:CNC
\213\CS
:END
```

较上例的变化是在 CS 和 CHKDSK 中选择一个文件执行。这里加入了段的概念，如果 CS 文件存在，转向段:CNC 后执行 CS；否则执行 CHKDSK 后转向段:END，实际上:END 是为越过 CS 的执行而设置的标号。

例3：

这个命题的目的是对.PRG 文件做统一处理，其中可带一个控制执行状态的参数。假定文件名是 P.BAT ，程序清单如下：

```
ECHO OFF
IF %1==A GOTO :AA
IF %1==B GOTO :BB
IF %1==D GOTO :DD
ECHO                                参数错误
```

```
ECHO                    A. 备份.PRG 文件到 A 驱动器
ECHO                    B. 备份.PRG 文件到\PRG 目录
ECHO                    D. 删除.BAK 文件
GOTO TVIU
:AA
COPY *.PRG A:
GOTO TVIU
:BB
COPY *.PRG \PRG
GOTO TVIU
:DD
DEL *.BAK
:TVIU
```

调用方式是

```
C>P A
```

通过%1传递执行状态的指定参数 A ，执行结果是将.PRG 文件拷贝到 A 驱动器。

```
C>P B
```

拷贝.PRG 文件到路径\PRG。

```
C>P D
```

删除所有.BAK 文件。

除以上三种参数外，屏幕显示功能菜单。目的是告诉用户可使用的三个参数，以便重新操作。

这里用到的参数传递，是 DOS 批文件编程的一个重要方法，它是加强程序通用性和功能性的有效手段。在 DOS 的批文件中，最多可使用10个预先设置的参数，这些参数分别用%0～%9表示。它们在编程时并没有实际的值，只是一种形式上的描述，实际值由用户调用该程序时的参数表给出，第一个参数对应%1的值，类推到第九个参数对应%9的值。%0则表示批文件名本身。

结合上面给出的实例，深入理解参数和段标识的使用对提高DOS编程水平有着重要意义。

第八章　DOS 的文本编辑功能

EDLIN 是 DOS 系统中的一个外部命令。可用来编写源程序或其它 ASCII 码的文本文件。这是经常要用到的。它靠 EDLIN.COM 文件支持，启动方法如下：

C>EDLIN < 文件名> ↓

若 EDLIN.COM 文件不在默认或缺席的驱动器或路径上，可在命令前面指定。如

C>A:EDLIN < 文件名> ↓

或　C>\DOS\EDLIN <文件名> ↓

视现场情况而定。

文件名指被编辑的文件，如果该文件存在，即被调入内存，否则创建该文件。

EDLIN 的专用提示符是'*'。在该提示出现后，可使用其自己的编辑命令。

所有命令只用一个字符定义，有 T、W、L、P、S、M、C、R、I、A、E、Q 共十二个，功能如下：

nT 文件名	把文件传入当前第 n 行。
nW	把 1—n 行写出到磁盘。
A	将当前文件的后续部分继续装入（如果有）
n1,n2L	由 n1—n2列表，当前行不变。
n1,n2P	由 n1—n2列表，最后行为当前行。
n1,n2S 字符串	在 n1—n2行查找字符串。
n1,n2,n3M	把 n1—n2行转移到 n3位置。

n1,n2,n3,n4C	把 n1—n2行复制到 n3共复制 n4份
? n1,n2R 串一^ Z 串二	在 n1—n2间把串一换成串二。
nI	从第 n 行开始插入。
E	保存编辑结果,记盘退出。
Q	编辑作废,不记盘退出。

虽然命令少且很简单,但却很实用。它把文件行自动编号,作为操作的标识。它可方便地显示、查询、搬移行或替换字符串,可准确简捷的完成各种文字处理。

了解命令的基本状态固然重要,但很不够。还有许多可简化操作的实用技巧。它可使本来就很简便的操作更迅速。

S、R 命令在一次执行后,可用单字命令再次重复。它们给出的字符串特征一经给定,在下一次重新定义之前总是保持有效。

文件中插入特殊的字符行,可使定位查询变得简单。

多次重复的数行可单独作成小文件,在需要使用的地方用 T 命令传入,这比用搬移或复制更方便。

把需多次重复的单行写在开头,用 C 复制更容易。

出现频繁的复杂字符串可用简单的符号代替,再用 R 命令将简单符号替换成需要的字符串,这可简化输入过程,又可提高输入的准确度。

W 可把指定行号以前的行写到文件,并重新初始化行号。在需要的时候初始化行号可简化行操作。

一次读入内存的数据约40K ,写出后可由 A 命令继续装载文件的后继部分。大文件可通过 W、A 的交互使用完成编辑。

用 R 命令的等值替换,可在查询程序结构时起作用。

在 DOS 系统的书籍中,对 EDLIN 的命令介绍得更科学、更完整。怎样用于处理实际问题,要靠个人的领会和发挥。

进入 EDLIN 后,功能键的定义如下:

F1　一个一个字符的复制样板行。

F2　复制样板行到指定的字符。

F3　复制整个样板行。

F4　删除到指定的字符。

F5　把当前行做为样板，重新进行编辑。

F6　相当于^ Z，在 R 命令中起分割作用。

第九章　DOS 命令集

上面介绍的 DOS 命令旨在说明 DOS 命令的基本功能和使用方法，这些只是 DOS 命令中的一小部分。为了全面的了解 DOS 系统的功能，下面介绍一个较完整的 DOS 命令集，以供上机参考。这个命令集属于 DOS3. 30版本。

1. APPEND［驱动器：］［路径］［；驱动器：［路径］…］
或 APPEND；［路径］

该命令与 PATH 命令相似，APPEND 指定 DOS 每次打开数据文件时应查找的子目录路径串。

如果 APPEND 不带任何参数，将解除原定义的数据文件路径。如果 DOS 在当前目录未找到数据文件，不再继续查找。

2. ASSIGN［驱动器1［＝］驱动器2［…］］

该命令可将一个驱动器符分配给另一个驱动器。如果有一个应用程序只使用驱动器 A 或 B，那么该命令能把对驱动器 A 或 B 的访问转向驱动器 C ，如：

A＞ASSIGN A＝C

该命令将把所有对驱动器 A 的访问转向驱动器 C ，对驱动器 C 的访问保持不变。

A＞ASSIGN

不带任何参数，将取消对所有驱动器转向的指定。

注意：此命令不要和 BACKUP 及 PRINT 命令一起使用，否则可能会产生意想不到的后果。

3. ATTRIB［＋R 或 －R］［＋A 或 －A］〈文件名或通配符〉

该命令设置文件的只读或档案属性。

+R　设置文件为只读属性，被设置的文件不可删除或修改。

−R　设置文件为读写属性，被设置的文件可以删除或修改。

+A　设置文件档案属性位。

−A　撤销文件档案属性位。

4. BACKUP [源盘符:[路径][文件名]] 目标盘符

[/S] [/M] [/A] [/D:mm−dd−yy]

该命令可备份一个或多个文件至新盘。

5. BREAK [=ON 或 =OFF]（中止）（默认值:BREAK=OFF）

用组合键 ^ C 随时中止应用程序运行非常方便，该命令能够增加 DOS 在运行中检查 ^ C 的次数。

如果在文件 CONFIG. SYS 中键入：

BREAK =ON

DOS 将在完成磁盘读写时也检查 ^ C，设置了该状态，多占了系统处理问题的时间，整个处理将变慢。

BREAK =OFF

只在屏幕显示、送打印机及读键盘时检查 ^ C。

6. CD [盘符:[路径]]

该命令可显示或改变指定驱动器上的当前目录。

7. CHKDSK [盘符:][路径] [文件名] [/F] [/V]

该命令可分析或维护目录、文件及文件分配表 FAT 。此命令可以显示出指定盘的现有文件数、占用的空间、目录管理占用的空间、尚未使用的空间以及内存空间的使用情况。

8. CLS

该命令用于清除屏幕的所有信息。

9. COMMAND [驱动器:][路径][/P][/C 辅助命令]

[/E:nnnnn]

该命令用于启动辅助命令处理器。

[/C 辅助命令]　指定由辅助命令处理器执行的命令。如果省

略限定，COMMAND 将显示带有标准 DOS 提示的命令。

[/E:nnnnn] 用字节数指定辅助命令处理器环境的大小，该数的范围是160～32768 。

[/P] 指示 DOS 安装辅助命令处理器并长驻内存，如果同时指定[/C]，DOS 将忽略[/P]。

如：COMMAND /C CHKDSK

表示装入辅助命令处理器，并将执行 CHKDSK 命令。

10. COMP [文件名] [文件名]

该命令比较两个文件，同时显示文件中前十个不同之处。

11. COPY [/A][/B][源盘符:][路径][文件名]

[/A][/B][+[,,]

[盘符:][路径][文件名][/A][/B]…]

[目标盘符:]

[路径][文件名][/A][/B][/V]

该命令用于拷贝源文件内容至目标文件并可对文件进行连接拷贝和换名拷贝。

/V 可指定对拷贝后的文件进行校验。

/B 指定拷贝以文件在目录表中的长度为依据。

/A 指示拷贝将文件作为 ASCII 码对待，直到 CTRL+Z 结束。

12. CTTY 设备名

该命令可修改 DOS 接收命令的设备，即改变控制台。

13. DATE [月－ 日－ 年]

该命令可设置或显示系统当前日期，如：将系统时间改为1991年7 月1 日，键入命令

C>DATE 7－1－1991

14. DEL [盘符:][路径][文件名]

该命令用于删除盘上的指定文件。支持通配符。

15. DIR [盘符:][路径][文件名] [/P][/W]

该命令用于列出指定的文件名信息。/P 为分页显示，/W 为横排显示。支持通配符。

16. DISKCOMP 驱动器1 驱动器2 [/1] [/8]

该命令用于比较两驱动器中的磁盘并显示两者间不同的面和磁道数量。

/1 指定比较软盘的1 面。/8 指定比较每个磁道上的8 个扇区。

如：DISKCOMP 源驱动器 目的驱动器 [/1]

表示将源驱动器中的磁盘与目的驱动器中的磁盘进行比较。

/1 指定比较软盘的1 面。

17. ECHO [ON 或 OFF]

该命令可在执行批处理文件时，打开或关闭显示设备。

18. ERASE [驱动器] [路径] [文件名]

该命令可删除指定文件，作用同 DEL 。支持通配符。

19. EXE2BIN [源驱动器：\路径\文件. EXE]
[目的驱动器：\路径\文件. COM]

该命令可将 EXE 文件转换成 COM 文件。

20. FDISK

该命令用来确定硬盘分区，也称硬盘低级格式化。

21. FIND [/V] [/C] [N] "字符串"

该命令可在文件中查找指定的字符串，并在屏幕上显示。

/N　在显示行号前自动加上行号。

/C　只显示包括字符串行的个数。

/V　显示所有不包括字符串的行，即对原查询取反。

22. FOR %%变量 IN（文件名）DO DOS 命令 %%变量

该命令用于重复处理，对指定的所有文件执行一个相同的 DOS 功能。文件名支持通配符。

23. FORMAT [盘符：] [/S] [/1] [/V] [/B] [/4] [/8]

该命令用于对指定驱动器上的磁盘格式化。

/S　传送系统的隐含文件和 COMMAND.COM 文件。

/4　在高密驱动器上完成低密软盘的格式化。

/V　进行读写校验。

/1　指定格式化按单面进行。

/8　指定每磁道8 个扇区，默认每磁道9 扇区。

/B　指定在磁盘上预留系统引导文件的空间。

24. IF［NOT］条件 DOS 命令

该命令表示条件成立，指定的 DOS 命令被执行。

条件必须是下列三种之一：

ERRORLEVEL n　n 值小于 DOS 出口返回值时取假。

EXIST　文件标识符，文件存在取真。

字符串1＝字符串2　两字符串相等取真。

25. JOIN［盘符：］［路径］［/D］

该命令可将驱动器连接到路径名。此后可用路径的方式操作驱动器。

/D　将取消这种设置状态。

26. LABEL 驱动器：卷标字符串

该命令可为指定的磁盘设置卷标。

27. MD［盘符：］［路径］

该命令用于在目的盘上建立指定的子目录。

28. MODE［设备参数］［设备＝参数］

该命令可为设备定义工作参数。

29. MORE

该命令用来分页从标准设备上读取数据。

30. PATH［盘符：］［路径］；［［盘符：］［路径］；…］

该命令可定义缺席的 DOS 查找路径。

31. PRINT［/C］［/B：］［/D：］［/M：］［/P］［/Q］［/S］
［/T］［/U］

该命令用于排队打印文件。

/C　清除打印队列中的指定文件。

/B：设置打印缓冲区长度 512—512 * n 字节。

/D：指定打印设备。

/M：指定打印执行时 CPU 的时钟数滴嗒数 1—255。

/P　向打印队列增加文件。

/Q　限定打印队列的文件总数。

/S　指定打印的时间片 1—255。

/T　清除打印队列中的所有文件。

/U　指定等待打印设备有效的 CPU 滴嗒数 1—255。

32. PROMPT 系统提示符

用下列指定的方式设置系统提示符：

b　字符

d　当前系统日期

e　字符串 ESC

h　空字符串(删除前面的字符)

g　字符>(默认值)

i　字符<

n　当前驱动器

p　当前目录

q　字符=

t　当前系统时间

v　DOS　　版本号

—　回行首(回车换行)

$　字符 $

例：系统提示符为 C>，设置系统提示符同时显示当前目录。

C>prompt **Pg** ↓

C:\>

提示符前显示目录，该例在根目录下。

C:\>CD\DOS ↓

C:\DOS>

表示当前工作在 DOS 目录。

33. RECOVER 文件名或驱动器符

当目录已经损坏，该命令可从磁盘上恢复文件。

34. REM 字符串

该命令可在标准设备上显示字符串，被用于在批文件中显示需要说明的信息。

35. REN 源文件名 目的文件名

该命令用来改文件名。

36. REPLACE [源文件][目的文件]/A /P /R /S /W

该命令可有选择的替代目的盘上的文件。

/A　指定替换目的盘上没有的文件。同名文件被忽视。

/P　指定在替换前给出一个可供确认操作的选择。

/R　指定替换包括只读文件，否则将忽视。

/S　指定在所有子目录中搜寻同名文件再进行替换。

/W　指定开始替换前给出一个等待的提示，按任意键开始。

37. RESTORE [源盘符:][目的盘符:][路径] 文件名
[/S][/P]

该命令可恢复由 BACKUP 备份的指定文件。

38. RD [盘符:][路径]

该命令可在指定的驱动器上删除一个子目录，但必须保证该目录是空目录，否则不能被执行。

39. SELECT 盘符:\路径\参数

该命令用于指定不同国家或地区的键盘配置及 DOS 所用的日期格式。

40. SET [名[=值]]

该命令可插入指定的字符串到命令处理器环境。

41. SHARE [/F：文件区域][/L：锁定值]

该命令支持 DOS 文件共享成为可能。

/F： 以字节为单位指定内存分配的区域。

/L： 指定为期望的锁数分配内存，默认20。

42. SHIFT

该命令用于左移批处理参数一个位置。

43. SORT [/R] [/+N]

该命令用于 DOS 排序，从标准的输入设备上读取文件，排序后再写到标准的输出设备上。

/R 指定排序的降序方式，否则为升序方式。

/+N 指定排序从第 N 列开始，否则从第一列开始。

44. SUBST 盘符：[路径][/D]

该命令可用驱动器符代替 DOS 路径。

/D 取消这种代替。

45. SYS 盘符

该命令可传送操作系统文件到指定的磁盘。

46. TIME [时：分：[秒[.小数]]]

该命令用于显示或改变系统时间。

47. TREE [盘符：] [/F]

该命令可显示指定磁盘上的目录结构。

/F 显示包括目录中的文件名，否则仅显示目录名。

48. TYPE [文件名]

该命令可显示指定文件的内容到标准的输出设备。文件名不支持通配符。

49. VER

该命令可在标准设备上输出当前执行的 DOS 版本号。

50. VERIFY [ON 或 OFF]

该命令用于打开或关闭磁盘 I/O 检验。

51. VOL 盘符

该命令可显示指定驱动器中磁盘上的卷标。

52. XCOPY [盘符：][路径] 文件名 [盘符：][路径]

文件名[/A][/D:MM－DD－YY][/E][/M]
[/P][/S][/V][/W]

该命令提供有选择地拷贝指定文件至目的盘(DOS 3.2以上版本)

/A　只拷贝设置了档案位的文件,拷贝后档案位不变。

/M　只拷贝设置了档案位的文件,拷贝后档案位撤销。

/D:月—日—年 只拷贝指定日期及以后的文件。

/P　在文件的拷贝过程中,先显示文件名,操作员可用 Y 和 N 键确定该文件是否进行拷贝。

/E　在目的盘建立与源盘子目录对应、内容相同的子目录。此参数不理睬当前目录中的文件。

/S　在拷贝当前目录中源文件的同时,连同下级子目录一起拷贝。无此参数时不理睬子目录。

/V　在拷贝过程中对目的文件进行校验。

/W　在拷贝前等待,按任意键后开始。

第十章　DOS 常用的提示信息及错误信息

当 DOS 系统发现不能理解用户的命令或不能全程完成用户指定的任务时，将在屏幕上发出相应的错误提示信息，以便用户知道发生问题的性质。这里只摘录了一些最常见的错误提示信息，并指出了一般性的错误原因，现象及处理方法，以便上机参考。

1. Abort, Retry, Ignore

来源：DOS 命令的 I/O 接口。

原因：DOS 不能读/ 写指定设备。

处理：在以下三种状态中选一：按下 A 中断命令的执行；按下 R 不放弃错误，继续执行；按 I 放弃当前错误，继续执行。

2. Are you sure (Y/N)?

来源：DEL，ERASE。

原因：启动 DEL 或 ERASE 命令删除所有文件时，屏幕要求对操作进行确认。由于命令的执行将产生成批文件的毁灭，在此给用户提供一个再次确认的机会以避免灾难性的后果。

处理：回答 Y 并回车文件删除被实施；回答 N 并回车则放弃执行删除命令。

3. Attempted write－protect violation

来源：FORMAT。

原因：企图格式化已贴了写保护标签的软盘。

处理：换一张新软盘或拆掉写保护标签。

4. Backup file sequence error

来源：RESTORE。

原因：在多张盘上的备份未从第一张开始恢复。

处理：按照正确磁盘顺序启动 RESTORE。

5. Bad command or file name

来源：解释 DOS 命令行。

原因：键入了非法的 DOS 命令，或在可到达的路径上找不到支持命令的可执行文件。

处理：检查命令的书写是否有误，文件是否存在或路径引导是否正确。

6. Cannot start COMMAND , exiting

来源：DOS。

原因：CONFIG. SYS 中 FILES 设定的值太小，DOS 的文件通道不够分配。

处理：增加 CONFIG. SYS 中的 FILES= 参数值(8－99)，具体设定可按软件的运行要求确定，一般情况下 DBASE 要求大于20，FOXBASE 的满载环境要求大于50，FORTRAN 可控制99个 DOS 句柄。新的设置必须在重新启动系统后才生效。

7. Compare error(S) on Track NN side, NN

来源：DISKCOMP 磁盘比较命令。

原因：比较两个软盘时，发现某磁盘某个磁道内容不符。

处理：用 DISKCOPY 重新复制或更换磁盘，要判断引发不符的原因在哪一方。

8. Copy another (Y/N)?

来源：DISKCOPY。

原因：当前磁盘的拷贝已经完成，问是否继续进行其它磁盘拷贝。

处理：继续磁盘拷贝，则回答 Y 并回车，否则回答 N 并回车。

9. Disk boot failure

来源：DOS。

原因：在当前盘的 DOS 基础文件缺少，引导失败。

处理：检查启动盘是否有系统文件，确认后重新引导，再不成

功，可更换系统盘。

10. Duplicate file name or file not found

来源：RENAME。

原因：RENAME 未找到源文件名或目的文件名有冲突。

处理：对文件名给出合法的指定，必要时可列出目录清单查找有关的文件信息。

11. File not found

来源：DOS 命令。

原因：在命令中或作为命令参数指定的文件名不存在。

处理：重新启动带有正确文件名的命令。

12. Format failure

来源：FORMAT。

原因：FORMAT 在完成时发现盘错。

处理：磁盘的目录分配表所驻存的扇区已坏，磁盘不能使用，可更换新盘。

13. Incorrect DOS version

来源：DOS 命令行中断。

原因：所键入的命令不能在当前 DOS 配置下运行。

处理：用合适的 DOS 版本引导系统再重新启动命令。

14. Insert first diskette in drive A:(B:)

Insert second diskette in drive B:(A:)

来源：DISKCOMP 磁盘比较命令。

原因：要求你插入需比较的磁盘。

处理：准备工作完成后，按任一键磁盘开始比较。

15. Insert source diskette in drive A:(B:)

Insert target diskette in drive B:(A:)

来源：DISKCOPY 磁盘拷贝命令。

原因：在软驱中准备好要复制的磁盘和空盘。

处理：准备完毕后按任一键开始磁盘复制。

16. Insufficient disk space

来源：DOS 命令的写操作。

原因：磁盘空间不够，写文件的命令被迫停止。

处理：清理或更换磁盘，保证文件的空间后再执行。

17. Invalid disk change

来源：DOS 命令。

原因：指定的路径中有一个目录不存在。

处理：检查指定的路径表，重新启动带有正确路径表的命令或重新设置缺席路径。

18. Invalid drive specification

来源：DOS 命令。

原因：命令行中指定的驱动器非法。

处理：修正驱动器标识符，再次启动命令。

19. Memory allocation error

Cannot load COMMAND. COM, system halted

来源：DOS。

原因：DOS 的内存分配表被应用软件破坏。

处理：这是不可恢复的状态错误，只能重新启动系统。

20. Program too big to fit in memory

来源：DOS。

原因：由于没有足够的内存空间，常驻内存的 COMMAND. COM 文件不能被装载。一般是因环境要求超过了内存的实际支付能力所致。

处理：用参数对可控制的内存装配进行适当剪裁，如从 CONFIG. SYS 文件中减少通道或缓冲区的数量、减少常驻内存的文件等等，再重新启动系统。

频繁出现这个提示则应考虑是否需要增加内存的容量。

21. Syntax error

来源：DOS 命令。

原因：命令行非法，不能被 DOS 解释。

处理：修改命令行中的错误。

22. Unable to create directory

来源：DOS 命令。

原因：建立目录的命令被下述原因之一阻塞：

(1) 目的目录名已经存在；

(2) 路径表中有找不到目录；

(3) 目录数已达到 DOS 的最大限定，不能再增加；

(4) 目录名与文件名冲突；

(5) 目录名非法。

处理：分析输入命令属于上述哪一种，纠正后再启动命令。

下面是经常出现在 DOS 错误提示中的一些短句：

1. Abort, Retry ,Ignore Drive not ready error

原因：因驱动器没有准备好而不能执行读写。

处理：检查磁盘的到位和门柄的状态，确认可靠后再启动磁盘设备。

2. No paper

原因：打印机纸空或不能联机。

处理：检查打印机纸或状态的设置情况。

3. Not read

原因：不能读。

处理：检查磁盘及驱动器。

4. Not ready

原因：涉及的设备没有准备好。

处理：磁盘操作时检查驱动器，打印操作检查打印机。在确认设备准备好后，再次执行。

5. Read fault

原因：DOS 不能读，多见于磁盘不能被可靠读出。

处理：检查驱动器和磁盘，如果确认是磁盘数据不能被读出，

可更换不同的驱动器进行实验，或许还有读出的可能。

6. write fault

原因：DOS 不能写，多见于磁盘有损坏的扇区但未被 DOS 标识的情况。

处理：如确认磁盘安装无误，可用 CHKDSK 命令检查磁盘的文件分配表是否存在错误及可利用的空间是否充足。如果排除了这些可能情况后错误仍然存在，则要考虑重新对磁盘进行格式化。

7. write protect

原因：DOS 不能向贴有保护签的盘进行写操作。

处理：更换磁盘或拆下写保护标签。

第十一章　DBASE Ⅲ基础知识

11—1　概　述

DBASE Ⅲ属工具型语言，是数据库应用技术的杰出产品。在众多的工具软件中它不仅能够独占鳌头，长久不衰，且用户还在日益增加。尤其在办公和管理领域，它已成为当前计算机用户自行开发应用软件的主体语言。

1. 优点　它以对二维表格的管理为中心，侧重表格数据的关系处理，辅以必要函数和常规的计算，开放式的交互操作，内涵丰富、变化灵活的基础命令，完整独特的状态控制，独树一帜、自成系统。它命令直观，易懂易学，适应办公人员的文化层次。其表达方式接近思维规律，二次开发周期简短，编程简单，见效迅速。

2. 缺点　作为一个系统，它也并非十全十美，最突出的问题是用户对其速度经常是深表遗憾。在数据不多的情况下，尚可对其容忍，但管理较大规模的数据，就会让人忍无可忍。这与近来开始流行的 FOXBASE 相比，不得不承认是稍逊一筹。

11—2　数据库和变量

1. 数据库　数据库是形象的二维表格，分为横、纵两向。

横向称记录，每个记录都有相同数据格式。记录是总体数据中一个基本的数据单位。限定记录长度不超过4000字节。

纵向称字段，又称栏目或栏变量。每列数据都有相同的数据类型和固定的长度。在满足记录长度的情况下，字段数被限定在128

个以内。

具有这种结构的数据集合称之为数据库，每个数据库的记录数被限制在10亿个以下，数据库只能以磁盘文件的形式存在，也被称为数据文件。

DBASE Ⅲ的数据库属于关系型，数据库可扩充、筛选、投影、剪裁、连接，具备初步的数据库管理概念。作为小规模的数据管理，关系型数据库简单实用，与日常习惯最接近。

数据库的主文件名一般可由用户自己定义。后缀名被系统默认为.DBF，也可自己定义，但不如使用默认名简便。

本系统的文件名可由 1—8 个字符组成，基本符合 DOS 文件名规范，以 A—J 的单个字为文件名虽合法，但最好不要使用。它与系统默认的数据库别名相矛盾，可能引起数据库操作的失控。

2. 字段类型 一般数据库是由多个字段组成的。根据处理问题的实际需要，对不同的字段在操作、计算、使用上的具体要求不尽相同。为了区分数据在内涵方面的差别，就要对字段的属性做出具体的规定，这就是字段的类型。

在本系统可以使用五种约定的字段类型。

(1)字符型(Character)又称 C 型。它可接受任何字符(包括不可打印字符)，最长254 字节。C 型字段是使用最广泛的类型，它表现能力强，处理灵活，可进行连接和比较，在可能的情况下应首选此种方式。

(2)数值型(Numerie)又称 N 型。它仅接受正负数值(包括负号和小数点)，最长19字节，数值精度为15.9位。它可进行任何算术运算。

(3)日期型(Date)又称 D 型。它仅接受合法的日期值，定长度 8 字节，一般以美国的日期格式表示，月/日/年各占两字节。它可做比较运算和一些限定的算术运算。

D 型字段有些先天不足，在检索查询时对空值的记录不能作出令人满意的反应，给数据的使用带来不便。在可能的情况下，最

好用C 型字段代替。

(4)逻辑型(Logic)又称L 型。它定长1 字节,仅有两个值,用t 、T 、y 、Y 表示真值,用n 、N 、f 、F 表示假,其它值非法。

同样用1 字节,L 型字段的表现力比C 型要差的远。除非有特殊需要,一般很少采用。

(5)摘要型(Momery)又称M 型。每当摘要字段被定义,系统自动生成一个同名的.DBT 文件,用于记录字段内容,每个字段可容4096字节,但在主数据库中仅占10字节。

由于系统对摘要字段缺乏支持,记入的内容不能有效的发挥作用。在打印格式上也存在排版缺陷,磁盘资源浪费很大。一般很少使用。有时宁可多使用C 型字段,也不愿使用M 字段。

C 、N 、D 型字段可用于数据库的排序和索引,L 、M 型字段则不行。

字段的命名由 1—10个字符组成,字符集与文件名相同,但必须用字母开头。用数字开头的字段名在计算表达式时易与实际数值相混淆,故定义数字开头的字段名被系统鉴定为非法命名。

文件名与字段名的区别可用字符串'91M '说明,它可以是文件名,不能是字段名。

3. 内存变量 数据库字段是特殊的变量,仅有字段变量是不能满足需要的,在程序设计时不可避免的要使用内存变量。为了叙述方便,一般把字段变量叫做字段,把内存变量简称变量。

变量的概念是建立一个唯一的命名与对应数据的关联,并可通过命名调用它所对应的数据。这个命名叫变量名,对应的数据叫变量的值。若更形象地描述,可认为变量是在计算机内存中用一个唯一的代码(变量名)指定一个内存空间并存入对应的数据。当变量名出现时,计算机用它对应的数据去替换掉这个名字。

例如用A 作为变量名,令其值等于6 ,算式可用A 代替6 书写,当计算机发现A 时,就自动用6 去完成运算。为什么必须用A 而不能直接用6 ,这是因为6 可能是在计算到一定程度才能确定

的值，并不是能够事先知道的。初接触变量概念时，会感到有些抽象，到实际使用时就会知道，变量的一般应用并不难。

本系统可承认256 个变量，总的字节数限制在6000字节之内。变量的命名规范与字段的命名规范完全一致。变量类型有 C、N、D、L 四种。

4. 字段与变量的同异 变量和字段有相同的类型概念，在使用方法上也基本一致。在本系统中，字段和变量基本属于同一范畴，尽管在表现形式上十分类似，但在实质上仍是有明显区别的：

变量是计算机暂存数据的手段。它在辅助计算或控制转移等很多方面有着至关重要的作用。在使用变量前，必须首先建立变量，使用没有定义的变量会导致错误的出现。在断电或退出系统时，变量自行消失。

字段是指定格式的磁盘文件在计算机中的反映形式，它的定义来源于数据库结构，值来源于记录。它的主要作用是反映记录的某个特征。在正常退出系统后仍以数据库的形式存在。

对得到数据后不许再任意改变的量称常量。它是变量的一个子集。在本系统的变量中，没有严格的常量概念。只有在算式中使用的实际值才与常量的概念相符。

若出现字段名与变量名冲突时，系统优先使用字段名。

11—3 运 算

1. 有关运算的规定

(1)算术运算

符号：±号、* *、*、/、+、—。

运算：按±号、乘方、乘除、加减顺序优先。

(2)关系运算

符号：=、<、<=、>、>=、<>。

运算：等于、小于、小于等于、大于、大于等于、不等于，无优

先顺序。

(3)逻辑运算

符号：.NOT.、.AND.、.OR.。

运算：按非、与、或顺序优先。

(4)字符运算

符号：+、-、$ 字符串运算无优先顺序。

运算：+、-用于字符串连接，$ 用于字符串检索。

在混合运算中，按算术运算和字符运算→关系运算→逻辑运算的优先顺序进行。用括号可改变任何运算的优先级。

要注意到，关系运算符的使用，将使整个表达式返回逻辑结果，如果定义的关系成立，将返回逻辑(.T.)真，否则将返回逻辑(.F.)假。

2. 表达式 用运算符连接字段、变量、常量和函数组成的合法算式，或这些基本量的本身统称为表达式。表达式比变量有更强的功能和更广泛的实用性。在运用时要注意以下几点。

(1)这些基本量本身就是表达式，它是大概念中的一个子集。不要认为必须有运算符才能称为表达式。凡是表达式可以出现的地方，都可使用这些基本量。

(2)运算不同类型的变量要通过系统函数转换成合法的形式。本系统对合法性要求很粗糙，只要指定的运算能被继续并完成，就认为是合法的。

(3)函数的输入、输出值都有严格类型的规定。

(4)相同类型表达式通过运算符或通过使用函数可能会使类型发生变化。

关系型数据库语言以关系处理为中心课题。表达式是构成关系所必须依赖的基础。要达到关系处理的目的，构造条件表达式是关键。表达式的合法性和准确性是发挥数据库作用的基本前提。在实践中注意总结表达式应用的规律是提高数据库表达能力唯一有效的途径。

3. 字段表 把多个字段名用“,”分割后连成的表列，用以限定数据库的有效字段并重新安排它们的顺序。一般情况下，缺席默认为所有字段，自然顺序。严格的说它并不属于表达式。但在命令参数中有时可代替表达式使用，在有限的范围，可认为字段表列是一种特殊的表达式。当两者概念不可混淆时，还要严格区分。

11—4 命令格式

1. 命令分类 DBASE 系统有177 条基本指令，大略可分为三类：

命令：执行一个规定的操作和完成一个指定的功能。

函数：完成转换数据、系统检测、查询等功能。

状态：设置数据格式、环境，控制外设状态等。

在大的概念上说，这些指令统属命令范畴，但就系统内部的很多情况而言，命令的概念专指第一类操作指令。每类指令都有共同的特点，是以其作用划分的。

命令是完成任务的基本定义，函数和状态在完成的过程中起到一定的辅助作用。函数不仅是命令的必要的后援，也是实施控制的重要支柱。命令和函数都是以即时反应区别于状态，状态可在时间上延续，命令和函数则不能。状态可为后续命令管理环境和设备，为命令的执行结果提供必要的后援。从反映的形式上看，不同的状态会影响到同一命令的执行结果。

2. 命令特点 本系统的命令，与其它语言相比，具备几个鲜明的特点：

(1)单体命令的功能性极强，简单的一条命令足可与其它系统的大块程序相比美，甚至更强。

(2)与英语表达方式很接近，容易理解，便于记忆。

(3)数据库命令具备可装配的参数，通过参数的选择可演化出多种多样的执行状态。自由，灵活，实用。

(4)支持交互操作,有自己的全屏幕控制,即时反映,迅速简捷。

(5)用系统的状态控制命令的执行结果,可大大简化编程并有效提高速度。

3. 系统命令的基本格式和通用约定

(1)格式的原则形式

动词 ＜参数＞ [可装配的短语] [选择1/选择2]

由动词指定执行的基本指令,如有参数,它是执行指令所必须具备的信息,此项不能缺席;短语的作用是对基本指令的执行结果提供状态控制的某种具体的要求,起到限制作用。限定短语一般有以下几种:

[范围] [表达式] [条件] [字段表列] [设备]

按命令集的规定,在表述可装配的短语词时有如下通用的约定,很好的领会这些约定的含义,会大大有助于命令的理解。

(2)命令行中对后续部分的表述方法

＜ ＞用户必须提供的参数表,缺省属非法。

[ ] 可用可不用的参数。

/ 在可能选择的项目中只能使用一个。

…… 可重复使用前面定义的格式。

这些都是为表述书写命令的规范而人为加入的说明符,只起表达命令规范的作用。它可帮助人们理解命令结构中,哪些部分是必须写的,哪些部分是可写可不写的,哪些部分的格式是可以重复使用的。在实际书写命令时这些符号并不出现。如果认为这些符号是属于命令本身的符号,那就错了。

(3)常用短语的含义

空格 用于分割命令动词、短语、参数。它是命令集中的重要字符。

ON 开状态。或导向索引排序的关键字。

OFF 关状态。

TO　　导向一个文件、设备或状态。

PRINT　输出到打印机。

FIELDS　必须后续标准的字段表列。

FOR　　必须后续条件表达式，对满足条件的记录执行。

WHILE　必须后续条件表达式，由当前开始到不满足条件为止。

WITH　必须后续表达式和参数表列，可以粗糙地理解为用或携带什么。

ALL　　指所有处于开放状态的记录(不包括被系统隐蔽的记录)。

NEXT/RECORD[N 型表达式]　若表达式的值为 N，NEXT 认为是从当前开始的第 N 个记录；而 RECORD 则认为只对记录号是 N 的记录执行。

NEXT/RECORD 和 ALL 同属关于范围的约定。在需指示范围时，可视情况选择其中一个。默认的记录范围仅有两种可能，一是等价 ALL，另一种是当前记录。不同的命令规定有自己的默认状态，涉及数据库整体操作的命令多为前者；涉及记录单体操作的命令，多为后者。

这些短语可以是命令的组成部分，起限制命令状态的辅助作用。如果使用，必须是一个完整的小结构，每个完整的小结构被称为该命令的一个子句。对有后续参数要求的短语，必须给出参数的定义，短语结构缺席，表示系统的默认状态被选用。短语不止一个时通常与顺序无关。

4. 系统命令的书写规范　无论是交互操作还是设计程序，都要掌握命令的书写。命令书写的统一要求如下：

(1)必须用动词开头，如果有后续参数，用户必须提供。如可装配短语，可视情况决定是否装配。

(2)动词、短语、参数必须用一个以上的空格分开。

(3)所有使用的字母不计较大小写。

(4)动词或短语可仅使用前四位，其余部分如写则必须正确。

(5)命令行限制254 字节长，空格计算长度。

(6)在命令行任意位置，可用“;”启动继续行。

第十二章　DBASE Ⅲ 系统及技术指标

12—1　技术指标

作为系统，对自身资源占用、可利用资源的分配以及对用户支持的程度都要有一定的限度，这些具体的规定，就是系统的技术指标。在指标限定的范围内，系统能可靠工作。如果设计较大的程序，对技术指标更要充分重视。

(1)最多可使用 15 个各种类型的文件。

DOS 句柄大于 20；

最多可同时打开 10 个数据库文件；

一个 DBT 算两个文件；

每个数据库最多可有 7 个活跃的索引文件。

(2)每个过程文件最多可装子过程 32 个

(3)每个数据库最多记录数为 10 亿个

(4)每个数据库最多字段数为 128 个

(5)每个记录最大长度为 4000 字节

(6)字符最大长度≤254 字节

(7)字段最大长度

字符型≤254 字节

数值型(小数点和负号占位)≤19 字节

逻辑型定长 1 字节

日期型定长 8 字节

备注型(需 DBT 文件的支持)定长 10 字节

(8)命令行最大长度≤254 字节

(9)运算精度不小于 15 位

(10)数值范围为 10 的±307 次幂
(11)内存变量数量≤256 个
(12)内存变量最大空间为 6000 字节

12—2 系统默认的文件名

(1) .DBF 系统的数据库文件。
(2) .NDX 数据库索引文件(逻辑顺序对照表)。
(3) .DBT 数据库中备注字段支持文件。
(4) .LBL 由记录生成标签(名片)的格式文件。
(5) .FMT 记录输入或查询的格式文件。
(6) .MEM 内存变量的映像文件。
(7) .FRM 统计报表生成条件及格式文件。
(8) .TXT ASCII 文件,用于数据通讯。
(9) .PRG 用本系统语言编制的命令及过程文件。
(10) .BAK 由系统自动产生的各种备份文件。

12—3 全屏幕操作

在全屏幕编辑方式下各控制键的功能如下表所示。

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	将光标左移一步,在菜单选择方式下,向左移一个选择项
→	^D	将光标右移一步,在菜单选择方式下,向右移一个选择项

续表

控制键	等价键	功　能
^ →	^ 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，但方向相反，在 CREATE 命令中，将光标移回到本字段开始处
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		进入或退出菜单选择方式的开关
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，键盘上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
^ N		插入一个新的行或新的字段
^ KR		在 MODIFY COMMAND 命令中，把另外一个文件读进来
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)

续表

控制键	等价键	功　能
RETU		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，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
PgDn		当光标停在备注型字段时按下此键，便进入字处理方式
^T		自光标处开始向右抹去一个字
^Y		自光标处起抹到字段末或该行末
^U		在 BROWSE 或 EDIT 命令下给记录做删除标记，在 MODIFY 和 REPORT 及 MODIFY STRUCTURE 命令中，删除一个字段定义

12—4　DBASE Ⅲ命令表

1. ? /?? ＜**表达式**＞[,＜**表达式**＞.....]

在屏幕或打印机显示一个或多个表达式的值。一般是在屏幕显示，通过状态设置连接打印机可同时输出或关闭屏幕仅向打印机输出。“??”显示在设备的当前坐标，“?”显示在设备的下一行。

2. @＜**行，列**＞[**SAY** ＜ **表达式**＞ [**PICTURE** ＜**格式**＞]]
[**GET** ＜**变量**＞ [**PICTURE** ＜**格式**＞]]
[**RANGE** ＜**下限表达式，上限表达式**＞]
[**CLEAR**]

SAY 用指定的格式在显示器或打印机上显示表达式的值；

GET 用指定的格式在显示器上显示或编辑已经存在的变量或字段。详见 14—7 节数据的格式化处理。

3. ACCEPT [< 提示字符表达式>] TO < 变量>

在屏幕显示提示字符并将屏幕输入信息送入一个 C 型变量。如果变量存在将被覆盖，否则将被创建。

4. APPEND [BLANK]

用全屏幕的状态在数据库文件末端追加一个记录。选择 BLANK 则制止进入全屏幕状态，一般用于程序。

5. APPEND FROM < 源文件名> [FOR <条件>]
[SDF / DELIMITED]

从其它文件向当前打开的数据库文件追加记录。默认的源文件是. DBF 文件，如果 SDF/DELIMITED 被选定，默认的文件是 .TXT文件。非默认的文件要在文件名中具体指定。

6. ASSIST

帮助执行 DBASE Ⅲ 命令，由菜单驱动。只是对初学者略起些作用，一般无大用途。

7. AVERAGE <表达式表列> [<范围>]
[FOR/WHILE<条件>][TO<变量表列>]

按数据库记录分别计算数值表达式表列的平均值，这些值被放到自行创建的变量表列中，表达式表和变量表必须逐个对应。

8. BROWSE [FIELDS <字段表列>]

使用全屏幕窗口显示进行编辑，每屏可达 17 个记录；如果 FIELDS 被选用，用户可改编字段的顺序和限制字段的多少。

9. CANCEL

终止程序执行，返回圆点提示符。在嵌套层次很深的程序中这是退出的一个简单可靠的办法。

10. CHANGE [<范围>] [FIELDS <字段表列>]
[FOR/WHILE < 条件>]

在数据库中编辑指定的字段和记录；除非使用了范围或条件的限定，否则将编辑全部记录；除非使用了字段表列，否则按数据库的字段顺序编辑所有字段。该命令申请全屏幕的工作方式，如果

有活跃的屏幕格式文件，将被该命令采用。

11. CLEAR

清屏。

12. CLEAR ALL

关闭除过程文件外的所有文件，释放所有内存变量，置当前工作区为一区。在任何情况下，该命令可给出一个已知的环境。

13. CLEAR GETS

释放 @....GET 的键盘操作，使 READ 不理睬在此命令之前发出的所有 @....GET 命令；或说使先前定义的变量编辑作废。

14. CLEAR MEMORY

清除所有的内存变量，或说对内存变量的环境进行初始化。

15. CLOSE [ALTERNATE/DATABASES/FORMAT/INDEX/PROCEDURE]

关闭指定类型的文件。CLOSE 与各子句组合可分别完成下列文件的关闭：

ALTERNATE 关闭屏幕记录员文件。

DATABASES 关闭所有打开的数据库文件和所有支持数据库的其它文件，如摘要文件、索引文件、格式文件等。

FORMAT 关闭当前区的格式文件。

INDEX 关闭当前区的索引文件。

PROCEDURE 关闭过程文件。

除 CLOSE DATABASES 外，其它文件也可通过等价的 SET 命令关闭。

16. CONTINUE

将记录指针从当前开始移动，定位到下一个记录，这个记录满足由前一个 LOCATE 命令的所有限定和约定。它是 LOCATE 的后续命令，不能独立使用。

17. COPY FILE ＜源文件＞ TO ＜目的文件＞

复制任意类型的文件。

18. COPY TO

＜文件名＞[＜范围＞][FIELDS＜字段表列＞]
[FOR/WHILE＜条件＞][SDF/DELIMITED]
[WITH＜定界字符＞]

将打开的数据库复制到另一个数据库或文本文件。

默认的目的文件是. DBF，如果 SDF/DELIMITED 被选定，默认的文件是. TXT。

如果使用了范围或条件，仅满足限定的记录被复制，否则复制全部记录。

字段表列的使用可选择字段和改变字段顺序，否则按原数据库结构复制。

除非使用了 SET SAFE OFF 状态，否则覆盖文件将给出要求确认的屏幕提示。

19. COPY STRUCTRURE TO ＜文件名＞
[FIELDS ＜ 字段表列＞]

将打开的文件结构复制到新的数据库。

除非字段表列被使用，否则按原数据库结构复制。

20. COUNT [＜ 范围＞][FOR/WHILE ＜ 条件＞]
[TO ＜变量＞]

对指定限定的记录计数。若没有限定则对数据库的所有开放记录计数。统计结果可被存放在指定的变量中。

21. CREATE ＜文件名＞

创建一个新的数据库结构并将该文件加入目录。

除非特别指定，否则文件名为. DBF。该命令以全屏幕方式工作，操作请参看全屏幕的控制键。

注意：字段名必须用字母开头；字段类型可用 C、N、L、D、M 单字母指定或用空格键选择并用回车键确认。

22. CREATE LABEL ＜表格文件名＞

由菜单驱动生成标签文件。等价于 MODIFY LABEL 。

23. CREATE REPOTE ＜ 标签文件名＞

由菜单驱动生成报表格式文件。等价于MODIFY REPOTE。

24. DELETE [＜ 范围＞] [FOR/WHILE ＜ 条件＞]

对指定的记录作删除标记，标记符号是记录前的'＊'。

除非使用限定，否则仅对当前记录做删除标记。

在全屏幕方式下＾U 是删除和恢复的切换开关。

做删除标记的记录可被 PACK 命令实施删除，可被 SET DELETE ON 状态隐蔽。

25. DIR [＜驱动器：路径][＜文件名或通配符＞]

显示指定磁盘驱动器上的文件名。

除非文件名或通配符被采用，否则仅显示.DBF 文件。其余与 DOS 的 DIR 命令基本相同。

26. DISPLAY [OFF][＜范围＞]

[FIELDS ＜字段表达式列＞]

[FOR/WHILE ＜ 条件＞][TO PRINT]

显示当前数据库中的记录信息。

除非使用限定，否则仅显示当前记录。

除非使用字段表列，否则所有字段按数据库结构顺序显示。

范围子句不仅限定记录的个数，同时还限定显示从当前记录开始。范围的限定优先于条件限定。

条件子句不仅限定记录必须满足的逻辑表达式，同时限定检查条件是从头开始。

选择 TO PRINT 可将显示内容送打印机，否则仅在屏幕显示。

除非使用 OFF 限定，否则在记录前显示记录号。

显示记录超过20个，屏幕出现

Press any key continue...

等待按任意键继续显示。

要求显示非当前区的字段可用"工作区号—>字段名"的显示形式。

27. DISPLAY MEMORY [TO PRINT]

显示当前内存变量环境的全部信息。TO PRINT 将打印这些信息。

28. DISPLAY STRUCTURE [TO PTINT]

显示打开数据库文件的结构，包括字段名、字段类型、长度及小数位，同时显示总记录数、每个记录的总字节数。内容超过20行屏幕等待，出现提示

Press any key continue...

29. DISPLAY STATUS

显示描述工作环境的所有信息。有关现用数据库、索引文件、交替文件及系统参数方面的信息。请看演示结果(括号内是后加入的注释)：

LIST STAT

当前所选择的数据库的名称是 ：

选择数据区－1，使用的数据库是：C:SUBO. dbf 别名－SUBO。

(一区的数据库文件名和别名)

Index file: C:CODE. ndx key － CODE

(一区的索引文件名和索引键表达式)

选择数据区－3，使用的数据库是：C:CSUBO. dbf 别名－CSUBO。

(三区的数据库文件名和别名)

Index file: C:NAME. ndx key － NAME

(三区的索引文件名和索引键表达式)

Alternate file－ C:LL. TXT　　(屏幕记录员文件在使用)

文件检索的路径：　\HU\　　(缺席路径)

默认的磁盘驱动器：　C:　　(当前驱动器)

(21种状态及功能键的设置情况)

ALTERNATE	—ON	DEBUG	—OFF
BELL	—OFF	DELETED	—OFF
CARRY	—OFF	DELIMIT	—OFF
CONFIRM	—OFF	DEVICE	—SCRN
CONSOLE	—ON	ECHO	—OFF
ESCAPE	—ON	MENU	—OFF
EXACT	—OFF	PRINT	—OFF
HEADING	—ON	SAFETY	—ON
HELP	—OFF	STEP	—OFF
INTENSI	—ON	TALK	—OFF
UNIQUE	—OFF		

边 界 = O

功能键 F1 — HELP
功能键 F2 — MODI COMM
功能键 F3 — RUN EDLIN
功能键 F4 — MODI STRU
功能键 F5 — WITH
功能键 F6 — NEXT
功能键 F7 —TO PRIN
功能键 F8 — BROW
功能键 F9 — RUN ZRED
功能键 F10 — .AND.

功能键值是人为定义的,与系统默认值有所不同。

30. DO <程序文件名>/< 过程子程序名>
[WITH < 参数表>]

执行一个程序或过程,并可与该程序或过程交换数据。

除非特别指定,否则默认文件名是.PRG。

参数表被使用，该程序的第一条有效命令必须是 PARAMETERS，其后的参数表必须与 WITH 在数量上一一对应。数据交换仅按参数表中的变量位置进行，与变量名无关。在数据交换中要充分注意到数据的类型不能与程序中定义的类型发生冲突。

31. DO CASE

CASE <条件>
[命令组]
CASE <条件>
[命令组]
↓
[OTHERWISE]
命令组
ENDCASE

用于程序的多分支控制命令，它必须用 DO CASE 开头，每个 CASE 语句可开辟一个命令组分支，OTHERWISE 可包含剩余的所有情况，结构必须用 ENDCASE 结束。在多个可能的分支中最多只能有一个分支的命令组被执行，然后执行 ENDCASE 后面的语句。

32. DO WHILE 条件表达式

命令组
[LOOP]
[EXIT]
ENDDO

用于程序循环控制。条件表达式成立时命令组被重复执行，条件表达式不成立时则退出循环，执行 ENDDO 后面的命令语句。LOOP 命令可在到达 ENDDO 之前直接返回到循环的开始，EXIT 命令可在条件表达式成立时跳出整个循环。

33. EDIT [[RECORD]<数值表达式>]

对记录数据进行屏幕编辑，可使用全屏幕控制键操作。如果没

有子句限定，编辑从当前记录开始；否则编辑将从数值表达式的值所对应的记录开始。编辑在全屏幕退出命令的出现或到达文件尾时终止。

34. EJECT

在打印机上执行一个换页。

35. ERASE ＜ 文件名＞

从目录中删除一个指定的文件。

36. EXIT

提前退出 DO WHILE 循环，参看 DO WHILE 语句。

37. FIND ＜ 字符串＞

将记录指针定位于其索引关键字与所指定字符串相匹配的第一个记录。

38. GO/GOTO [BOTTOM/TOP]/＜数值表达式＞

直接将记录指针定位于由数值表达式的值所指定的记录。

GO 和 GOTO 是一个命令的两种形式，可任选一个。

数值表达式的值与定位的记录号对应。小数自动取整。超过数据库记录范围出现错误提示。这是对记录号的绝对定位方式。对于被状态隐蔽的记录，这是到达记录的唯一方式。

BOTTOM 将指针指向文件的最后一个记录，TOP 将指向第一个记录。如果索引文件在使用，它们服从逻辑顺序；如果使用了过滤器，它们仅对活跃的记录起作用。

39. HELP [＜ 命令字＞]

解释 DBASE Ⅲ 命令并提供其他信息，由菜单驱动。仅在学习时起作用，一般无大用处。

40. IF ＜条件＞

命令组1

[ELSE]

命令组2

ENDIF

程序中的双分支结构命令。条件成立执行命令组1，否则执行命令组2。IF 必须用 ENDIF 结束。

41. INDEX ON <索引键表达式> TO <文件名>

创建当前数据库的索引文件。索引键必须是与数据库字段相关的表达式，类型限定是 C、N、D 中的一种，不同类型的数据要通过函数转换。每个数据库最多可同时保持7 个活跃的索引文件。

除非特别指定，文件名为. NDX。

42. INPUT [<提示表达式>] TO < 数值型变量>

显示提示表达式的值并接收键盘的输入到数值型内存变量。变量存在将被覆盖，否则被创建。

43. INSERT [BLANK][BEFORE]

将一个记录插在数据库文件中指定的位置。

BLANK　　插入一条空白记录在当前记录之后。这是默认状态。

BEFORE　　在当前记录之前插入一条记录。

这种插入记录的方法将会引发对全部后续记录在磁盘上重新抄写，这是很缺乏效率的操作。除非特别需要，一般不采取这种操作方式。若能在 APPED 追加时用索引控制文件的顺序，在效率上要高的多。

44. JOIN WITH<别名>TO <目的文件名>FOR<条件>
[FIELDS<字段表列>]

将当前数据库和用别名指定的数据库中的记录和字段连接到新创建的数据库中。默认文件名. DBF。

没有字段表限定，默认两个数据库的全部字段，但字段总和不可超过128个的限制数。

别名数据库必须在其它工作区打开，连接是用当前区记录对别名数据库的全部记录进行条件判定，以决定是否连接。在没有条件限定时两个由100条记录作成的数据库可生成有10000条记录的新数据库。

这条命令占用大量的磁盘空间和时间。简单的连接可通过多区关联的拷贝完成，在资源占用上要少得多。该命令虽然功能性很强，除特殊需要外一般很少使用。

45. LABEL FORM ＜标签文件名＞[SAMPLE][＜范围＞]
[FOR/WHILE＜条件＞][TO PRINT]
[TO FILE＜文件名＞]

按标签文件指定的格式，输出数据库内容到打印机或文件。SAMPLE 选用时，输出标签之前先显示其样板。

46. LIST [＜范围＞][FIELDS＜字段表列＞][OFF]
[FOR/WHILE＜条件＞][TO PRINT]

显示当前数据库中的记录信息。除非使用限定，否则从头开始显示所有记录。

范围子句不仅限定记录的个数，同时限定显示从当前记录开始。范围限定比条件限定优先。

除非使用字段表列，所有字段按数据库结构顺序显示。

选择 TO PRINT 可将显示内容送打印机，否则仅在屏幕显示。

除非使用 OFF 限定，否则在记录前显示记录号。

47. LOCATE[＜范围＞]FOR＜条件＞

将记录指针定位于满足指定条件的第一个记录。

范围子句不仅限定记录的个数，同时限定查询从当前记录开始。

查询失败，指针被定位到范围的最后一个记录。

如果到达文件尾，则 EOF()返回真。

48. LOOP

跳过该语句与 ENDDO 之间的所有命令，返回到 DO WHILE 的开始。参看 DO WHILE 命令。

49. MODIFY COMMAND ＜文件名＞

建立或编辑命令文件(.PRG)或 ASCII 码的文本文件。

除非具体指定，文件扩展名为.PRG。

50. MODIFY　LABEL＜文件名＞

启动全屏幕状态，生成或编辑标签格式文件(.LBL)。

51. MODIFY REPORT ＜文件名＞

启动全屏幕状态，生成或编辑报表格式文件(.FRM)

52. MODIFY STRUCTURE ＜文件名＞

启动全屏幕状态，修改数据库结构。省略文件名时，修改当前数据库结构。

53. NOTE/＊ ＜注释字符串＞

在命令文件中插入注释行，如果使用继续行，仍被认为是注释。

54. PACK

删除数据库中带有删除标记的记录。重新整理数据库，并释放这些记录占用的磁盘空间。

55. PARAMETERS ＜参数表列＞

接收 DO〈程序名〉WITH …命令传递的参数值。

56. PRIVATE [ALL[LIKE/EXCEPT ＜ 框架＞]]

[＜ 变量表列＞]

当 PRIVAT 说明专用变量时，把与之同名公用变量或高层模块中的专用变量隐蔽起来，待从低级程序返回后再对变量的值进行恢复。

57. PUBLIC [＜ 变量表列＞]

使内存变量全局化，由该命令定义的变量在任意子程序中变值后全程有效。

58. QUIT

关闭所有文件并退出 DBASE Ⅲ 系统 。

59. READ

接收键盘输入的数据到 GET 定义的变量。

60. RECALL[＜ 范围＞][FOR/WHILE＜条件＞]

恢复带有删除标记的记录。

61. REINDEX

对已存在的现用索引文件重新定义。这是更新过时的索引文件的简单方法。它不需要知道索引键的表达式。

62. RELEASE [<**变量表列**>]
[**ALL**[**LIKE/EXCEPT** <**框架**>]]

释放当前内存变量。框架可使用通配符。

63. RENAME <**旧文件名**> **TO** <**新文件名**>

文件换名。

64. REPLACE [<**范围**>] <**字段1**> **WITH** < **表达式1**>
[,< **字段2**> **WITH** <**表达式2**>...]
[**FOR/WHILE** < **条件**>]

用表达式的值替换字段内容。范围和条件可参看 LIST 命令。

表达式的类型必须与字段匹配。跨越工作区的字段必须用"别名—>字段名"的形式指出。

没有条件限定,仅对当前记录替换。

范围不仅限定记录的个数,同时限定替换从当前记录开始。

范围限定优先条件限定。

如果指针到达文件尾,EOF()置真。

对当前文件的控制索引不能执行成批的替换,这种替换的执行结果一般不会全面达到命令本身表达的目的。

65. REPORT FORM < **文件名**>
[<**范围**>][**FOR**<**条件**>]
[**PLAIN**][**HEADING**< **字符串**>][**NOEJECT**]
[**TO PRINT**][**TO FILE** < **文件名**>]

从数据库记录中用文件名(.FRM)定义的格式输出数据报告到屏幕或打印机。

PLAIN 被使用日期在每页上出现,否则仅在第一页出现。

HEADING 将定义的字符串输出到页号的上一行。

NOEJECT 限定从当前页打印，否则先执行一个换页再打印。

66. RESTORE FORM <文件名> [ADDITIVE]

将存入磁盘文件(.MEM)的变量恢复到内存。

ADDITIVE 将保留内存中的现有变量，否则现有内存变量被全部清除。

注意内存变量的总数不可超过256 个，空间不可超过6千字节。

67. RETURN [TO MASTER]

程序结束，返回到本程序的上一级；如果程序只有一级则返回到系统。

系统在确认程序完成后可自动执行一个 **RETURN** 命令。对独立的程序，**RETURN** 可省略，但过程文件中的子程序一般要求用 **RETURN** 作为结束语句。

68. RUN <DOS 系统的命令或可执行文件>

执行 DBASE Ⅲ 以外的程序命令或功能，然后返回 DBASE Ⅲ 系统。这条命令必须在有足够的内存空间的情况下才能被执行。

69. SAVE TO < 文件名 > [ALL [LIKE/EXCEPT < 框架>]]

将当前内存变量保存到磁盘文件(.MEM)。由此保留的变量可用 RESTORE 命令再次恢复到内存。

70. SEEK <索引键表达式>

将记录指针定位于其索引关键字与索引键表达式值相匹配的第一个记录。

71. SELECT <别名>

选择十个工作区之一为当前区。

别名可用数据库名、数据库别名、数字1 —9 、字母 A —J 中任选一种形式。如果使用宏代换，参数必须是 C 型。

72. SET

驱动全屏幕状态，可设置或变更 DBASE Ⅲ 的状态。

73. SKIP ＜数值表达式＞

使记录指针相对当前记录移动，其移动的方向和记录个数满足表达式值的限定。小数自动取整。超过范围将出现错误提示。

74. SORT TO ＜ 新文件名＞ ON ＜字段名1 ＞［/A］［/D］
［,＜字段名2 ＞［/A］［/D］,……］
［＜范围＞］［FOR ＜条件＞］

根据指定的排序要求，生成一个新的排序数据库(. DBF)

范围和条件可参看 LIST 命令。

/A 限定字段按升序排列；这是默认的排序方式。

/D 限定字段按降序排列；这种要求必须明确指定。

75. STORE ＜ 表达式＞ TO ＜变量＞［,＜ 变量表列＞］

将一个表达式的值存入一个或多个内存变量；如果变量存在则将被覆盖，否则被创建。

76. SUM［＜范围＞］［＜表达式表列＞TO＜内存变量表＞］
［FOR/WHILE ＜条件＞］

计算数据库中 N 型字段表达式的总和送内存变量并在屏幕显示。范围和条件参看 LIST 命令。

77. TEXT［字符文件块］ENDTEXT

显示字符文件块到屏幕。

这是一个结构命令，TEXT 必须用 ENDTEXT 结束。

78. TOTAL TO ＜文件名＞ ON ＜索引键＞［＜ 范围＞］
［FIELDS＜字段表列＞］［FOR/WHILE＜条件＞］

根据当前数据库记录生成一个合计数据库。其中同一索引键的数值字段被求和。当前数据库的索引键必须是活跃的，或是当前数据库已经按索引键排序。

79. UPDATE［RANDOM］ON ＜索引键＞FROM＜别名＞
REPLACE ＜字段1＞ WHITH ＜表达式1＞
［,＜字段2＞ WHITH ＜表达式2＞...］

用别名指定的数据库对当前数据库进行修改。别名指定的数

据库必须在其它区打开。两个区的索引字段必须相同。相同索引键值的记录仅有第一个记录被修改。

80. USE [<文件名>][INDEX < 索引文件名>]
[ALIAS < 别名>]

打开数据库文件及索引文件。

81. WAIT [< 提示字符串> [TO < 变量>]

暂停程序执行，显示字符串，等待输入一个字符到变量，然后继续执行程序。该变量可接受不可打印字符，这对实现特殊的控制很有用。

82. ZAP

从当前数据库删除全部记录。这种删除是毁灭性的，不可恢复。除非使用了 SET SAFE OFF，否则屏幕将要求确认。

12—5 DBASE Ⅲ 的状态设置

1. SET ALTERNATE TO <文件名>

创建一个用以保存屏幕输出信息的文件，除非进行指定，文件名为. TXT。可以理解为该文件是屏幕的书记员。

没有文件名可关闭当前的屏幕记录文件。

2. SET ALTERNATE ON/OFF

确定当前的屏幕输出发送/不发送到屏幕记录文件。该命令只可对屏幕文件追加内容，不能创建。

3. SET BELL ON/OFF

屏幕在接受键盘数据满界时响铃/不响铃。

4. SET CARRY ON/OFF

将上一个记录的内容写入/不写入到追加的记录中。

5. SET COLOR TO <前景/背景>
[,<前景/背景>][,<边框>]

设置屏幕显示的颜色。第一组设置标准输出的颜色，第二组设

置反相输出的颜色。

6. SET CONFIRMET ON/OFF

全屏幕方式下自动/回车到下一个字段或变量。

7. SET CONSOLE ON/OFF

将输出信息送/不送至屏幕。在用?或文本格式向打印机输出数据时经常用该命令隐蔽屏幕信息,这可提高输出效率或保持屏幕环境。

8. SET DEBUG ON/OFF

将 ECHO 的输出送/不送至打印机。

9. SET DECIMALS TO <n>

设置某些运算和函数返回结果的小数位数。默认两位。

10. SET DEFAULT TO <驱动器>

为文件搜索指定当前驱动器。

11. SET DELETED ON/OFF

隐蔽/不隐蔽有删除标记的记录。

12. SET DELIMITER TO [< 定界符>] [DEFAULT]

为字段和变量的全屏幕显示指定分隔符。

13. SET DELIMITER ON/OFF

全屏幕显示字段和变量时,以正常/反相显示定界。

14. SET DEVICE TO SCREEN/PRINT

将@...SAY 命令的结果送至屏幕/打印机。

15. SET ECHO ON/OFF

对执行的程序命令在/不在屏幕或打印机上输出。该命令可跟踪程序的执行情况,对程序调试很有作用;但它会大幅度的降低程序的运行速度。

16. SET ESCAPE ON/OFF

定义 ESC 键有效/无效。

17. SET EXACT ON/OFF

字符比较时要求精确/不精确比较。不精确比较以左侧字符串

的结束为终止，这种约定使大多数查询得以简化。精确比较用于索引很不可靠。

18. SET FILTER TO ＜条件＞

数据库过滤器开关。使用条件时数据库将滤除所有不满足条件的记录，使文件外观看来就象仅由满足条件记录组成。

没有条件可使过滤器撤消，使被过滤器隐蔽的记录恢复。

19. SET FIXED ON/OFF

固定/不固定数值显示的小数位数。

20. SET FORMAT TO ＜文件名＞

为数据输入打开一个格式文件. FMT。

21. SET FUNCTION ＜功能键号＞ TO ＜字符串＞

设置功能键的值，“;”可在字符串中代替回车。

22. SET HEADING ON/OFF

在执行 LIST 或 DISPLAY 命令时，显示/不显示字段名。

23. SET HELP ON/OFF

设置功能键 F1 有效/无效。

24. SET INDEX TO ＜索引文件1＞[,＜索引文件2＞ ,....]

打开被指定的索引文件。

25. SET INTENSITY ON/OFF

全屏幕操作时采用/不采用反相显示。

26. SET MARGIN TO ＜n＞

设置打印机的左边界。n 为 ASCII 码字符数。

27. SET MENUS ON/OFF

全屏幕操作时显示（或不显示）菜单；

全屏幕状态时^ Home 可开/关菜单。

28. SET PATH TO [＜路径＞]

为文件搜索指定或撤销一条缺席路径；无路径参数时，为原设定的路径。

29. SET PRINT ON/OFF

将输出送/不送到打印机。

30. SET PROCEDURE TO ＜过程文件名＞

打开/关闭指定的过程文件。无过程文件名时为关闭命令。

31. SET RELATION TO ＜关联键表达式＞INTO＜别名＞

按照关联键表达式连接两个数据库。无表达式时，撤消这种连接。

32. SET SAFETY ON/OFF

设置/不设置文件覆盖保护。当状态为 ON 时，文件覆盖将发出要求用户确认的屏幕提示：

文件已经存在，重写它吗？　　(Y/N)

回答 Y 则文件被覆盖，回答 N 则中断当前的操作。

33. SET STEP ON/OFF

每条命令处理完后暂停/不暂停程序的执行。这是程序的调试命令。

34. SET TALK ON/OFF

将命令执行的结果送/不送到输出设备。

35. SET UNIQUE ON/OFF

使索引文件中保留第一个/全部带有相同关键字的记录。

12－6　DBASE Ⅲ 函数

1. 函数索引表

函数名称	说　明
&＜变量名＞	宏替换，延伸变量或表达式
ASC(C 型表达式)	返回该字符的 ASCII 码值
AT(＜C 型表达式1＞	返回子字符串1在字符串2
＜C 型表达式2＞)	中起始位置的字节数
BOF()	返回指针是否到达文件头的逻

	辑值
CDOW(<D 型表达式>)	返回 D 型表达式的星期几
CHR(<N 型表达式>)	将 ASCII 码值变换为对应的字符,这是获得不可打印字符的唯一方式
CMONTH(<D 型表达式>)	给出 D 型表达式的月份字符串
COL()	返回当前光标所在的列
CTOD(<C 型表达式>)	将合法的字符串变换为日期变量
DATE()	返回 D 型的系统日期
DAY(<D 型表达式>)	给出日期变量的日历值(月内)
DELETED()	返回记录删除标志是否存在的逻辑值
DOW(<D 型表达式>)	给出 D 型变量的星期值
DTOC(<D 型表达式>)	以字符串的形式返回日期变量的值
EOF()	返回指针是否到达文件尾的逻辑值
EXP(<N 型表达式>)	求指数函数
FILE("<文件名>")	返回测试该文件存在与否的逻辑值
INT(< N 型表达式>)	返回表达式的整数部分(不舍入)
LEN(<C 型表达式>)	返回字符串的字节长度
LOG(<N 型表达式>)	求自然对数函数
LOWER(<C 型表达式>)	将表达式中所有大写字符换成小写
MONTH(<D 型表达式>)	给出日期变量的月份值
PCOL()	返回打印机当前的列坐标
PROW()	返回打印机当前的行坐标

RECNO()	给出当前记录号
ROUND (<N 型表达式>, <小数>)	按指定的位数进行四舍五入
ROW()	返回光标当前的行坐标
SPACE(<N 型表达式>)	按 N 型表达式的值产生空白字符串
SQRT(<N 型表达式>)	求平方根函数
STR(<N 型表达式> [<长度>][,<小数位>])	将数字表达式变换为字符串
SUBSTR(<C 型表达式>, <起始> [,<长度>])	字符串的子串截取
TIME()	给出系统时间
TRIM(<C 型表达式>)	压缩字符串尾部空格
TYPE("<C 型表达式>")	求表达式数据类型
UPPER(<C 型表达式>)	将小写字母变换为大写
VAL(<C 型表达式>)	字符变换为数值
YEAR(<D 型表达式>)	给出日期变量的年历值

2. 函数演示

(1)& <变量名> 宏替换

对变量名进行延伸,用变量的值作为变量名。

```
ABC=12
. ?ABC          显示 ABC 的内容
12
ABC='BBB'

BBB=12
. ?BBB          显示 BBB 的内容
12
```

. ?&ABC　　显示宏的内容

12

. ?ABC　　显示 ABC 的内容

BBB

(2) ASC()　返回该字符的 ASCII 码值

. ? ASC('ABC')

65

字符 A 的 ASCII 码值是65。

(3) AT(<C 型表达式1> <C 型表达式2>)　字符搜索函数

返回子字符1在字符串2中起始位置的字节数字。

. ? AT('HG','ABDHG')

4

字符串 HG 在 ABDHG 中的第四字节开始。

(4) BOF()　文件头测试

返回指针是否到达文件头的逻辑值。

. GO TOP

文件指针指向文件的第一个记录。

. ? BOF()

.F.

文件头函数返回假。

. SKIP −1

指针继续向前移动。

. ? BOF()

.T.

文件头函数返回真。

注意,只有指针超越了文件的第一记录之后,BOF() 才返回真值,无论文件是物理顺序还是逻辑顺序。

(5) CDOW(<D 型表达式>)　　日期函数

返回对应 D 型表达式的日期是星期几。

.MC=DATE()

07/15/91

当前日期是1991年 7月15日。

. ? CDOW(MC)

Monday

这是星期一。

(6) CHR(<N 型表达式>)　转换函数

将 ASCII 码值变换为对应的字符,这是获得不可打印字符的唯一方式。经常用于对计算机发出可由 ASCII 码定义的命令。

.MN=62

. ? CHR(MN+3)

A

A 对应的 ASCII 码是65。

. ?? CHR(7)

将使计算机鸣铃一声。

. ? CHR(27)+CHR(12)

将引起打印机的换页或行。CHR(27) 是逃脱码 ESC 的 ASCII 表示,可用于控制打印机的字形变换和状态控制。

(7) CMONTH(<D 型表达式>)

给出 D 型表达式的月份字符串。

. MD=CTOD('06/21/91')

06/21/91

. ? CMONTH(MD)

June

这是六月份的英文表示方式。

(8) COL()　光标测试

返回当前光标所在的列坐标。

. ? SPAC(20),COL()

21

在显示20个空格后有一个字节的间隙，当系统发现 COL() 函数时，光标正好是在屏幕的21列。

(9) CTOD(<C 型表达式>) 转换函数

将字符串变换为日期型变量。要求字符串必须是一个合法的日期形式，否则只转换成日期类型。

. MD1=CTOD('07.06.91')

07/06/91

. ? TYPE('MD1')

D

将字符串'07.06.91'转换成 D 型变量。

另一种转换格式

. MD2=CTOD('07/06/91')

07/06/91

. ? TYPE('MD2')

D

两者虽然使用的分界符不同，但都是合法的，可等价。

. MD3=CTOD('070691')

/ /

. MD4=CTOD('13/21/91')

/ /

. ? TYPE('MD3'),LEN('MD4')

D D

MD3 无定界符非法，MD4 月份非法，转换后均只生成格式，忽视要求转换的非法值。

(10) DATE() 系统测试

返回 D 型的系统日期。

. ? DATE()

06/21/91

计算机中的系统日期是1991年6月21日。

. MD=DATE()

. ? MD,TYPE('MD')

06/21/91 D

将系统日期送入变量MD,测试MD类型是D型。

(11) DAY(<D型表达式>) 测试日期

给出日期变量的日历值。

. MD=CTOD('12/28/90')

12/28/90

. ? DAY(MD)

28

指定的日期变量是某月份的28天。这里是12月。

(12) DELETED() 记录测试函数

返回记录删除标志是否存在的逻辑值。此演示有一打开的数据库。

设定原数据库没有做删除标记记录。

. 5

. ? DELETED()

.F.

对没有删除标记的记录,返回逻辑假。

. DELETED

. ? DELETED()

.T.

对有删除标记的记录,返回逻辑真。

. DELETE RECO 12

指定对12号记录做删除标记。在命令中第四位以后的字符可以省略。

. LIST FOR DELETED()

5 号、12号记录将被列表显示。

DELETE 是命令，DELETED() 是函数，注意区别。

在未执行 PACK 命令之前，可用 RECALL 使删除标记得到恢复。

(13) DOW(<D 型表达式>)　　星期测试函数

返回日历值所对应的星期几。

. ? DOW(DATE()),DATE()

2 07/15/91

这是星期一。星期日作为每个星期的第一天。

(14) DTOC(<D 型表达式>)　　转换函数

以字符串的形式返回日期变量的值。

. MC=DTOC(DATE())

. ? MC,TYPE('MC')

07/15/91 C

将系统日期转换成字符串。测试为 C 型。

(15) EOF()　　文件测试函数

返回指针是否到达文件尾的逻辑值。

. GO BOTT

调文件指针到最后一个记录。

. ? EOF()

.F.

指针不是文件尾。

. SKIP

继续向后移动指针。

. ? EOF()

.T.

指针到达文件尾。

注意，只有指针超越了文件的最后一个记录之后，EOF() 才返回真，无论文件是物理顺序还是逻辑顺序。

(16) EXP(<N 型表达式>) 数学运算函数

求出自然指数的真数值。

. ? EXP(4)

54.60

54.60 是 e 的四次方的值。

(17) FILE("< 文件名>") 文件测试函数

返回测试该文件存在与否的逻辑值。如果文件有扩展名，必须被包括。如果文件不在当前目录，文件名前可用路径表；如果文件不在默认的驱动器，必须用设备符指定。

. ? FILE('RS.DBF')

.T.

测试文件 RS.DBF 是否存在，回答.T. 表示文件存在。测试在默认驱动器的当前目录进行。

. ? FILE('A:\RS\LX.PRG')

.F.

测试 A 驱动器上 RS 目录中的 LX.PRG 是否存在，回答.F. 表示文件不存在。测试在指定驱动器的路径终端进行。

(18) INT(<N 型表达式>) 取整函数

返回表达式的整数部分(不舍入)。

. MN=17/3

5.67

这里的小数舍入是按系统规定的两位小数自动进行的。

. ? INT(MN)

5

通过 INT() 函数取出了变量的整数部分。INT() 忽略小数，不对小数部分进行舍入处理。

. ? INT(MN+0.5)

6

这是对第一位小数四舍五入的人为控制方法。对第二位小数

可加0.05,类推。

(19) LEN(<C 型表达式>) 字符串测试函数

返回字符串的字节长度。

```
. MC='DFGHJK '
DFGHJK
. MN=LEN(MC)
      9
. ? MN+MN
      18
```

字符串 MC 的字节长度是 9,通过运算验证 LEN() 将返回 N 型值。

字符串的最小长度是 0,它是仅有变量名的字符串。

```
. MC1=''
. ? LEN(MC1+MC1+MC1)
      0
```

(20) LOG(<N 型表达式>) 计算函数

返回的是自然对数,它是 EXP() 的逆运算。

```
. ? LOG(32)
      3.47
```

32是 e 的3.47次方。

(21) LOWER(<C 型表达式>) 字符串转换函数

将表达式中所有大写字符换成小写,其它字符不变。

```
. MC='DJFf12[ AZ'
. ? LOWER(MC)
djff12[ az
```

(22) MONTH(<D 型表达式>) 日期转换函数

给出日期变量的月历值(N 型)。

```
. MN=MONTH(CTOD('06/21/91'))
      6
```

. ? MN * MN

36

指定的 D 型变量是六月份。验证结果是 N 型。

(23) PCOL() 打印机测试函数

返回打印机当前的列坐标。

. SET PRINT ON

设置打印机开态。

. ? PCOL()

0

打印机头定位在0 列。上述命令的结果将被打印并换行,再问

. ? PCOL(),PCOL()

0 4

因 PCOL( )输出占0—2列,空一列,再次测试时正好输出在第四列。

. SET PRINT OFF

设置打印机关态。

(24) PROW() 打印机测试函数

返回打印机当前的行坐标。

. SET PRINT ON

. ? PROW()

40

打印机头定位在40行。上述命令将被打印,再问

. ? PROW()

41

.SET PRINT OFF

打印机头定位在41行。

(25) RECNO() 记录测试函数

给出当前记录的物理序号。

. USE RS

打开 RS 数据库。

. ? RECN()

1

当前的记录指针定位在第一号记录。

. LOCA FOR DELE()

搜寻做了删除标记的第一个记录。

. MN=RECN()

12

当前记录号是12。

. ? MN+MN

24

函数值是 N 型。

(26) ROUND(<N 型表达式>,< 小数位数>) 舍入函数

按指定的位数进行四舍五入。

. MN=82.1872

. ? ROUND(MN,2),ROUND(MN,3)

82.1900 82.1870

(27) ROW()

返回光标当前的行坐标。

. CLEA

清屏。

. ? ROW()

0

当前光标在0 行。

. ? ROW()

3

再次查询,光标当前位置是第3 行。

(28) SPACE(<N 型表达式>)

按 N 型表达式的值产生空白字符串。

. MC1='N'+SPAC(5)+'N'

N N

两个 N 中间是5 个空格。

. MC2=SPAC(LEN(MC1) * 3-1)

.?LEN(MC1)

7

.?LEN(MC2)

20

用 LEN() 函数测 MC1 的长度,返回值是7 ,括号内的表达式值是20,MC2 将是由20个空格组成的字符串。

(29) SQRT(<N 型表达式>) 计算函数

求平方根,返回表达式的开平方根。

. ? SQRT(15+1)

4

16 的平方根是4 。

(30) STR (<N 型表达式>[< 转换长度表达式>]

[,< 小数位表达式>])

数值转换函数,将数值表达式变换为字符串。

. MN=128.69

128.69

.MC1=STR(MN)

128

长度参数缺席,机器默认10字节长;

. MC2=STR(MN,4)

128

小数位数缺席将忽视小数部分;

. MC3=STR(MN,6,2)

128.69

小数占一字节。

. ? LEN(MC1),LEN(MC2),LEN(MC3)

10　　4　　6

LEN() 函数不仅测试了三个字符串的长度，也间接的检查了三个字符串的类型是 C 型。

(31)　SUBSTR(<C 型表达式>,< 起始>[,<长度>])

子串截取从指定位截取字符串到指定长度。

. MC='1991年07月18日'

. MC1=SUBSTR(MC,3,4)

91年

. MC2=SUBSTR(MC,11,2)

18

(32)　TIME()系统测试函数

用八位定长的字符串给出系统时间。

. MC=TIME()

13:26:32

. ? LEN(MC),MC+MC

8 13:26:3213:26:32

可见返回的时间是字符串。

(33)　TRIM(<C 型表达式>)　字符串调整函数

删除字符串尾部空格。

. JTVV='北京市海淀区夏关村128号　　'

北京市海淀区夏关村128 号

. JTVV1=TRIM(JTVV)

北京市海淀区夏关村128 号

. ? LEN(JTVV),LEN(JTVV1)

30　　24

调整函数删除了字符串尾部的空格。

(34)　TYPE(<C 型表达式>)　表达式测试函数

用单字节的大写字母返回表达式数据类型的定义字，

含义如下：

C　字符型

N　数值型

D　日期型

L　逻辑型

U　未定义

注意，表达式必须用字符串的形式输入。

在上面的举例中已经多次涉及变量类型的测试，实际应用可在表达式的含义上推广。

```
. ? TYPE('姓名="张"'),TYPE('38/4=3')
      L     L
```

这是两个逻辑表达式，算式的值和算式本身是完全不同的概念，作为表达式的值，认为第一个表达式是C型、第二个表达式是N型是对的；作为表达式本身，它们只能是成立或不成立两种可能，返回的是逻辑结果。

(35)　UPPER(<C型表达式>)　字符转换函数

将字符串中的所有小写字母变换成大写，其它字符不变。

```
. MC='dhjk12 [XY'
dhjk12 [XY
. ? UPPER(MC)
DHJK12[ XY
```

(36)　VAL(<C型表达式>)　转换函数

将字符串变换为数值。

```
. MC1='137AB'
    137AB
. ? VAL(MC1)
    137.00
. MC2='SBFG'
```

. ? VAL(MC2)

0.00

. MC3='1234'

. ? VAL(MC3),VAL(MC3)+VAL(MC3)

1234 2468

除非改变小数位的状态设置,默认的小数位是两位,超过两位的小数四舍五入。

. MC4='432.768'

. ? VAL(MC4)

432.77

(37) YEAR(<D 型表达式>)

给出日期变量的年历值。

. ? YEAR(DATE())

1991

系统日期的公元年是1991年。

第十三章　DBASE Ⅲ的交互应用

13—1　系统的进入和退出

保持系统文件在当前目录，键入

C>DBASE ↙

即可启动本系统。

系统也可在软盘启动，只要将系统所在驱动器指定成当前驱动器，命令相同。一旦系统启动成功，本系统提示符出现，如果没有指定，默认的提示符是“.”。

此时处于 DBASE 系统的监控状态，可直接使用系统命令交互操作，也可执行预先编制好的程序。

启动系统需占用内存一般为 256K 字节，在各项技术指标同时达到系统规定的极限情况下，占用内存也不会超过 320K 字节。（系统指标参看有关章节）

如果在启动系统后，需要改变当前驱动器，可通过系统的状态设置命令实现。

. SET DEFAULT TO 驱动器符：↙

如果内存环境允许，也可通过本系统的外部功能调用命令指挥 DOS 系统去完成

. RUN 驱动器符：↙

两者都可达到目的，此时的默认驱动器已经转移。

注意，如果系统以软盘启动，该软盘不能取出，系统随时可能再次访问软盘。

RUN 既然可以调用 DOS 命令，是否还能做更多的事情，回

答是肯定的。它的命令形式是

. RUN <DOS 命令>/ <其它可执行命令> ↙

不妨用它做一个路径的转移，形式如下：

. RUN CD 路径表↙

这样的系统启动方式不易被初学者接受（如果感到困难，暂时可不深究），但在硬盘空间紧张的情况下，这是节约资源的有效方法之一。无论系统在何处启动，都能在启动后顺利地进入到自己希望的工作目录。

当工作完成后，必须键入退出命令

. QUIT↙

系统在接受退出命令后立即对内存数据作出反应，把需要保存的数据写到磁盘文件，并设置好磁盘文件的各项标识。非正常退出，内存数据丢失，如果文件标识出现错误，将会使文件的安全受到威胁，严重时会使整个文件被破坏。养成合法退出系统的习惯是绝对必要的。在未执行退出命令就关机的作法是万万使不得的。

13—2 数据库的创建

在涉及文件生成之前，先介绍本系统的文件生成过程。如果要求系统产生一个文件，首先要指定一个文件名，系统可根据命令要求自动定义一个默认扩展名，如果用户已经给出扩展名，将被优先采纳。在指定文件名时使用驱动器和路径，文件将在该路径下被创建，否则文件在默认驱动器的当前路径下创建。系统在接到文件名后，先按路径要求查询这个文件名，在没有发现同名文件时，该文件被立即创建；如果该文件已经存在，就先发出一条告警性的提示：

文件已经存在　是否覆盖（Y/N）

可根据实际情况确认后回答。回答 Y 将使同名文件被覆盖，回答

N 可使生成文件的操作中止。

如果将要覆盖的文件不但存在，而且正在被系统打开，这时不再要求用户确认，直接显示：

文件正在使用

然后自动中断生成文件的操作。

凡是可有新文件生成的命令，都会涉及这个统一的约定。此后，介绍有关命令时不再重复。

1. 定义数据库的结构 本系统以数据库管理为核心，围绕数据库的操作和应用，设计了很多实用完美的交互命令。

下面就从数据库的创建入手，开始对交互命令的学习。

创建数据库的操作是很简单的。在建库命令发出后，通过屏幕提示定义数据库结构，完成后记盘即可退出。

. CREAT ＜库名＞ ↙

系统在接受命令后开始该数据库的创建。该数据库是以输入的＜库名＞ 为文件名称。默认的扩展名为 .DBF。

系统首先进入全屏幕状态，屏幕给出与字段相关的提示信息，并用反相亮框标识字段名、类型、长度和小数位的输入位置。首先在字段名处按规范定义一个名称，用回车转入定义类型的操作，通过字母 C 、N 、D 、L 、M 单键可指定字段类型，也可用按空格使菜单选择项变动，在所需类型出现时用回车键确认。如果是 D 、L 或 M 型字段，系统自动采用自定义长度，该字段定义完成。如果是 C 型，随后再输入字段的长度。如果是 N 型还要确定小数位（如果有的话）。这是一组完整的信息，它可定义一个字段。依次重复进行，直到所有字段定义完毕。如有字段名非法、重复或长度超出等不符合规范的信息输入，系统鸣铃告警并封锁当前状态，直到输入合法的信息为止。

以定义一个名为‘姓名’长度 6 字节的字符型（C 型）字段为例：

先在字段名的亮框内输入‘姓名’并回车，光标进入到类型

定义的亮框，由于C型是默认的类型，直接回车即可，在长度框中再输入6回车，该字段的定义完成。

在最后字段的提示处空值回车，隐含记盘退出命令。修改已经定义的字段，可用箭头调动光标到位后修改，光标在任意位置可用Ctrl＋W键指定记盘退出。

系统在接到退出命令后，屏幕显示：

记盘则按回车键，否则按任一键继续 ……

不按回车可继续进行定义或修改，否则屏幕提示：

现在输入数据吗？(Y/N)

回答N可返回到系统提示符，整个数据库的定义操作全部完成。回答Y可立即进入建立数据记录的输入操作，此时录入数据是最原始的方法，在屏幕利用度和数据规范等方面都存在一定的问题。在正规的操作中，数据的录入一般要靠专用支持程序。

2. 在定义结构时的注意事项 结构的定义本身并不难，但定义一个合理的数据库结构也并非简单。数据库的结构一定与某种具体的业务密切相关，如果对具体业务了解不深刻，定义出一个好用、合理的数据库结构确是很困难的。

评价数据库结构好坏是个复杂的问题，一般依据是用较少的资源，做到尽可能丰富的数据表达能力，同时要求便于操作，并能适应编程需要。

在定义数据库时要注意：

(1) 字段名字要有意义或有规律性，要考虑使用的直观与方便或有利于程序设计。

(2) 一个字段要表达一个完整的定义，并仅表达一个定义。一个字段表达的内容越多，使用就会越费力；如果表达内容不唯一，在建立数据时就会令人困惑，无从下手。

(3) 字段的多少、长短要做到合理。这一点对资源的利用率和应用的方便性都很重要。资源利用率高时使用不一定方便，要在实际工作中体会字段的应用情况，结合实际工作找出比较恰当

的结构方案，并应不断的优化提高。

（4）在较大的数据库中要留有必要的冗余字段，它在处理数据库计算、数据类型转换等操作中会起到简化操作和编程的重要作用。

为了阐述数据库的操作，在这里先明确一个概念，就是记录的指针。数据库可以有很多记录，修改、显示、替换等操作均可仅对一个记录进行，究竟是哪个记录，要靠系统的指针控制。可以想象出一个可在整个数据库中任意移动的指针，用全屏幕的控制键或命令可把指针移向或移出某个记录。指针指向的记录是可操作的记录，称当前记录。凡是单记录的所有操作仅对当前记录起作用。

13—3　数据的录入

1. 操作　数据库一经建立，就要输入数据。在录入数据前应保证数据库是打开的。如果没有打开，可用 USE 命令打开数据库。

. USE ＜ 数据库名＞ ↙

如果数据库没有使用系统默认的扩展名（. DBF），必须在命令中指定，否则找不到文件。如果文件不在当前驱动器或路径，可在文件名前指出驱动器或路径。

在交互状态下，用 APPEND 命令可向打开的数据库追加记录。

. APPEND↙

追加操作是以全屏幕方式进行的。它依照数据库结构逐个字段进行，字段名在前，输入数据的地方用反相亮框标识出字段的宽度，数据填满，鸣铃并进入下一字段。一个记录完成自动进入下一个记录，一次可追加多个记录。在新记录的第一个字段处空值回车或用箭头向后翻阅超过当前记录，都隐含追加操作的结束命令。如果由向前翻阅退出追加，以后的状态等同于系统的修改编辑，可

用 ^ W 键记盘退出，或用 ^ Q 为不记盘退出。

如果需要对已经存在的记录进行修改，可用 EDIT 激活记录的编辑状态。命令是

. EDIT↙

除不能增加新的记录之外，其余均与 APPEND 相同。跨越文件尾的操作隐含退出命令。

如果要对当前记录进行指定，可在命令后增加限定参数，即

. EDIT RECORD n ↙

RECORD 是引导记录号的限定短语，n 是指定的记录号（数值型）。对 n 非法取值，系统提示：

记录号超出了范围

定位成功，进入编辑。

对于当前记录，还有一种便利的指定方式，就是直接输入该记录号并回车。

12↙

当前指针被指定在 12 号记录上。如果数值超过最后的记录号，提示与上面相同。更多的记录定位方法，后面专题介绍。

追加记录的另一种形式，可用下面的命令

. APPEND BLANK↙

这时确实在文件尾增加一条记录，但不进入全屏幕的编辑状态，虽然可用 EDIT 调动编辑，但在交互状态下很少使用。这是在程序中扩充数据库的方法。

APPEND 命令还具备从磁盘文件中成批的向数据库追加记录的功能，这将在拷贝命令中再做介绍。

有了对记录的增加、修改，还必定要有对记录的删除，用

. DELETE↙

可对当前记录做删除标记。在需要指定记录号时可用

. DELETE RECORD n ↙

当然也可用记录号定位，其操作与上面相同。

这里所说的删除标记是一种状态，它用系统规定的符号标识指定删除的记录，但并不立即就从数据库中将其真正删去。对于这样的记录仍可用命令恢复。

. RECALL↙

可恢复当前记录。

. RECALL RECORD n ↙

可恢复指定记录。

如果尚未实施删除的记录干扰了数据的使用，可通过系统的状态命令将这样的记录隐藏起来。

. SET DELETE ON ↙

系统在接受了这一命令之后，所有做了删除标记的记录就像不存在一样。这时即使发出恢复命令也不能被有效执行，除非解除这种封锁。通过状态的重新设置

. SET DELETE OFF↙

可达到解除隐蔽删除记录目的。若想彻底清除这些记录，可发出

. PACK↙

这时系统将数据库中的记录重新抄写，所有做了删除标记的记录被清除。采取两步清除的方式，既是为了数据的安全，更重要的是保证处理数据的速度，因为重写大数据库是很浪费时间的。如果每删除一个记录就进行一次重写，在时间的花费上是不能被人接受的。

DELETE 和 RECALL 命令还可以操作更大的记录范围[ALL/NEXT]或使用条件限制[FOR/WHILE]达到控制更多记录的目的，这将在后面的学习中领会到它的用途和用法。

在完成数据的录入或修改后，必须关闭数据库，直接关机会造成数据丢失。关闭数据库的命令是

. USE ↙

与打开数据库的命令相比，只是没有了文件的名字。实际上在打开数据库的同时，隐含关闭前一个正在使用的数据库的命令。在

开库命令前如果已经有数据库工作，系统首先执行原数据库的关闭，然后把新指定的数据库打开，使其进入工作状态。这是对开库命令的完整理解。

2. 对数据的基本要求 数据库的存在决定了数据录入的必要性，认识数据录入的规范要有一个过程。初接触数据库的人，往往忽视数据规范的重要意义。如果开发自己的程序系统，数据规范就显得更为重要。对数据的研究不仅在于表达内容的需要，还将会对程序的设计产生巨大的影响。

数据规范是个科学的命题，有着严格的专业理论。这里仅对办公涉及的简单数据，谈些粗浅的看法。

（1）输入的数据要有严格的规范，表示同一概念的数据必须具备完全相同的表现形式。单体数据力求确切。

（2）数据的分类要科学，每个数据的内涵必须唯一。对于一个既定的情况，用这个数据行，用那个数据也行的情况是绝对不许发生的。分类数据要完整，力求做到既不含糊，又无遗漏。

（3）从增加数据的整体的表现能力出发，统筹考虑数据的科学分配。不要过于追求冗余度，数据冗余度最小，使用不一定最方便；数据组合要适度，组合内容过多，亦会造成应用的困难。

事实上，数据组织的好坏表现了程序员的水平，好的数据不但使用方便，表达力强，还会使程序设计大大简化。对于一个复杂的应用系统，处理好数据规范是编程的重要基础。可以说没有好的数据规范就编不出好的程序。

对数据规范的要求看来不多，但要实际做到是非常困难的。在一般情况下，只要满足处理实际问题基本需要就算可以了。

3. 录入数据的正确性 上面谈的数据规范是从组织数据的角度考虑的。实际中还有一个非常重要的因素，就是数据输入时的失误。这种失误会使数据规范的价值大为降低。在数据库建立的过程中，对数据的校核、修改一般要经多次的反复才能达到使用的要求。

统计查询是数据库应用的重要组成部分，如果没有规范化的数据，数据库就会失去统计意义。这是我们要解决的一个重要问题。

数据的常见错误主要有两个方面，一是定义正确，输入错误，如把“工程师”输成“工程　师”；再有是定义本身就不能正确的表达对象特征，如把“工程师”输成“助理工程师”。前者会造成统计的遗漏，后者会造成分类错误。

13—4　排序与索引

数据库的使用基本可分为文件输出和统计分析。简单的计数容易办到，综合性的数据分析是比较复杂的过程，在具备一定的基础后再做研究。这里仅介绍文件的输出和计数操作。

1. 排序

(1) 排序的必要性。数据库的形成一般是在时间延续的过程中逐步积累的结果。输入的数据是按先后顺序存储在文件中。这样的顺序被称为自然顺序或物理顺序。如果用数据库生成办公文件或管理文件，这样的顺序在大多数情况下不能满足实际需要。如何能够把数据库的物理顺序转变成满足实际要求的顺序，这就是文件序的概念。

(2) 排序命令。把文件按照指定的排序特征重新进行整理，这对计算机说来是轻而易举的。完成这个任务的命令是

. SORT ON ＜字段名＞ TO ＜数据库名＞ ↙

SORT 是排序指令，ON 是关键字段的引导指令，此处用户必须提供一个字段名（或字段表列），这个（或多个）字段是排序的依据，被称为关键字段，TO 是目的导向指令，把输出引导到指定数据库的创建。在这个新的数据库中的记录是以关键字段的升序为排列依据。如果不指定文件的扩展名，系统采用默认值（. DBF）。

对本命令的执行结果还可进行更多的限定，这条命令的标准

表达形式如下：

SORT TO ＜新文件名＞ ON ＜字段 1＞ [/A] / [/D],
[[＜字段 2＞ [/A] / [/D], ...]
[NEXT ＜数值＞] [FOR ＜条件表达式＞]

在命令中各短语结构的作用如下：

可用字段表列指示多个字段为关键字段。

可规定每个字段的排序方式，/A 升序排列，/D 降序排列，默认为/A 状态。

用 NEXT ＜数值＞ 可限制对数据库记录的使用范围。

用 FOR 及条件表达式构成限制条件，可对数据库中的记录进行筛选使用。

用户可按规范任意装配后续的限制短语，用以达到控制输出结果的目的。对该命令还应了解如下内容：

每个短语的小结构必须完整，可以不计前后顺序。

逻辑和备注字段不能被指定成关键字。

对记录不加限制，所有记录被排序。

限定了范围，仅有满足范围的记录被排序。

使用了条件，仅有满足条件的记录被排序。

可见，新数据库可以是当前库的全集或子集。这个新数据库在记录顺序上满足所有关键字的指定，记录数目上满足范围和条件的限定。只要在命令发出时对短语装配合理，生成理想的实用数据库确实不是难事。

如果前面对可装配参数感到不易理解，通过这个具体的实例就容易形成明确的印象。随着使用命令的增多，掌握住命令的规律，不用很长的时间，就可对系统命令形成全面的认识。

2. 索引

(1) 排序存在的问题。排序的做法虽然可以达到文件的顺序要求，但不是最有效的方法。当数据库较大时，就暴露了它的致命缺陷。

一是在执行的过程中，要大量占用磁盘的空间资源（大约是原数据库的三倍），这个花费在很多情况下是不容忽视的。

二是其时间的花费不能令人满意，因为在有限的内存空间中，扫描众多的数据并进行计算处理，势必要靠时间的支持。

是否能够用较少的时空资源，达到文件有序使用的目的，这是我们现在要讨论的问题。结论是肯定的，手段是启用系统提供的索引文件。

（2）索引的概念。在不改变数据库顺序的前提下，把一个关键字特征做出有序排列的清单，并与原数据库记录的自然序号建立关联，将这个关联关系的对照表写成磁盘文件，这就是所谓的索引文件。

由于索引文件中的关键字是有序的，可采用对分或链表等先进方式进行记录特征的定位和查询，在执行速度上较顺序查询具备绝对优势。它排除了附加数据的干扰，仅与记录的数量和关键字的长度有关，大大节省了时空资源。

记录在索引文件中的位置称逻辑顺序，数据库可在索引文件的控制下以逻辑顺序输出数据。系统在输出数据前，先按索引的逻辑顺序找到一个记录，立即查出它的自然序号并把指针指向这一记录，如此重复，达到逻辑顺序的输出方式。此时，记录的自然序号变成了与关键字段值关联的随机排列。

控制数据库顺序的索引文件叫主用索引或当前索引。如果数据库同时和多个索引文件保持关联关系，则认为这些索引文件都是活跃的。数据库中关键字的变化，可被活跃的索引文件自动跟踪，以保持和数据库的严格对应。

在使用索引文件时要充分注意，没有激活的索引文件是不能自动更新的。使用过期的索引文件会造成记录顺序的混乱，也会造成新增数据的丢失。如果需要更新已经过期的索引文件，可在激活该文件后使用专用的索引更新命令 REIN。该命令将按原索引文件指定的关键字重新索引数据库。

（3）建立索引的方法。建立索引文件的命令格式是

. INDEX ON ＜表达式＞ TO ＜索引文件名＞

INDEX 是创建索引文件指令，ON 是关键字段的引导指令，此处用户必须提供一个合法的表达式，这个表达式的值是索引文件所依赖的关键字，TO 是目的导向指令，把输出引导到指定的索引文件。如果这个索引文件不存在，将被创建，如果不指定文件的扩展名，则系统采用默认值（. NDX）。

在以后的论述中，凡是前面出现过并已经做了解释的短语结构就不再重复。

（4）索引与排序的区别。索引文件的关键字与排序中的关键字段要严格区分，它们不仅格式不同，在使用的方法和概念上都有很大差别。

排序可在多个字段进行并仅可在字段上进行，字段的类型可以不同，每个字段的排序方式可以指定。一旦排序完成，关键字段失去作用。新数据库可以是源库记录的子集。

索引是使用关键字，它必须对应一个特定类型的表达式的值。可以用单一字段，也可以由多字段的信息拼接，还可以有内存变量和系统函数参与，但必须组成一个（只能是一个）合法的表达式。

如果不是单一字段，其类型必定是C 型或N 型。D 型数据仅可独立使用。L 、M 型字段不可用于索引键。

不同类型的字段要按照组成表达式的要求，通过函数转换成合法形式。通过转换的D 型字段可与C 型字段组成C 型表达式。

索引的顺序只能是升序的，不能再作指定。

索引文件必须是源文件关键字的全集。

索引文件可以长期有效。

排序可直接生成数据库文件，便于长期应用或转储保存。索引在处理随机变化的数据和快速检索方面有优势。针对实际情况选择其一或两者综合应用均可。使用时一般以对时空资源的要求

作为选择依据。

(5) 索引文件的使用。

在明确了索引文件的作用后,还要掌握如何正确有效的使用,使用方法不当,将影响其作用的发挥。

在打开数据库的同时可打开一个或多个索引,命令如下:

USE ＜ 数据库名＞ INDEX ＜ 索引 1 ＞ [, 索引 2, ……]

(索引 1 、索引 2 …… , 指索引文件名)

一个数据库最多可同时打开 7 个索引文件 (受使用文件总数的制约),第一个对数据库起控制作用,称主用索引或当前索引,其它保持活跃状态。短语 INDEX 限定了文件名的默认后缀名是 . NDX,否则要在文件名中指出。

如要改变控制主用索引,可用状态设置命令进行转换:

SET INDEX TO ＜索引 1 ＞ [, 索引 2, ……]

此时把主用索引放到第一位。SET 是设置状态的命令动词。这种转换是在不关闭数据库的情况下进行的,较前一种方式的速度要快些。INDEX 仍是导向短语,由此限制了 TO 后面的文件默认值取为. NDX。

对于索引文件来说,对数据库进行跟踪就是关键所在,这也是同时打开多个索引文件的唯一目的。要保持索引文件有效,就要使这些索引文件在与之相关的关键字变化时必须作出相应的变更。很多使用索引文件失败的原因,就是因为使用了没有跟踪数据库变化的索引文件,或说使用了过时的索引文件。

可以发现,INDEX 即可作为建立索引文件的动词命令,又可作为限制短语,同是一个词汇,因出现的位置不同其作用会有很大差别。还有类似的情况,如前面讲的 DELETE,它以命令、函数、状态三种形式出现。这些是容易混淆的概念,要注意区分。

(6) 无重复键值的索引。我们在数据库的操作过程中,经常会产生一种需要,这就是想知道在某个数据库的某个字段中,究竟包含着多少个数据的分类。系统对索引提供了一个状态,对于

解决这类问题是很有效率的。这要求在索引文件建立前发出指令

SET UNIQUE ON

然后再建立索引文件。

在索引文件中每个相同的键值只保留一个，其余的被隐蔽。或是说，索引文件是无重复键值的。

例如：对工程系列的技术职务建立这样的索引后，索引文件最多不会超过四项记录，即

高级工程师

工程师

助理工程师

技术员

在保持索引文件打开时，仅有数据库中第一次出现这四个值的四条记录，其余的记录被索引文件封锁。

这种索引文件有什么用处呢？

①用于检查数据规范。可在字段中的所有分类中发现是否有非法数据存在。

②调整数据规范。把该组数据拷贝到一个过渡数据库并保留数据不变，在过渡数据库中填写与原数据对应的新数据，通过关联替换的方式，对字段的部分和全部内容进行调整、变更或规范处理等（参看 15—4 节）。

③生成统计规范。通过不重索引，提取统计的全部特征作为依据，就会使随机统计做到无遗漏。因为在随机统计开始之前是很难知道数据分类的具体情况。

这种方式还能解决很多其它问题，在此就不一一举出。

释放索引状态用

SET UNIQUE OFF

索引状态的改变并不影响已经建成的索引文件的性质，因为它只与建立时的状态设置有关。

第十四章　基本的库操作

数据库的应用方式是千变万化的，可有多种的操作和使用形式，这里仅介绍交互情况下的一些基本的操作。

14—1　显　示

1. 数据库显示　当要显示数据库的全部内容时，可用

. LIST↙　　(1)

屏幕从横向显示 Record（记录号提示）后，依次显示数据库的字段名。行满自动换行。这是对记录内容的说明信息。

下面从数据库头开始显示记录号（自然顺序号）并对应上面的提示，逐项显示记录数据。每个记录开始一个新行，直到文件尾为止。

如果打印这些数据，可在回车前先用 Ctrl＋P 启动打印机，这是调用 DOS 的支持。本系统也有可用于打印目的引导的方法，能使文件输出到打印机，即

. LIST TO PRINT↙　　(2)

PRINT 是打印机指示短语，起到启动打印机的作用。

如果数据库在使用索引文件，输出符合该索引的逻辑顺序，此时记录号的排列是随机的。装配 OFF 短语可关闭记录号的输出

. LIST OFF TO PRINT ↙　　(3)

这样的装配对单纯的显示也是可以的，如

. LIST OFF↙　　(4)

(4) 较 (1) 增加了关闭记录号的限定，(4) 较 (3) 剪掉了通向

打印机的引导，短语字句的装配再次形象体现。之所以对此强调，是因为有大量的命令使用这些短语。这也是初学者容易感到困惑的地方。LIST 命令包括了短语结构的绝大部分，如果对此能够有效使用，其它命令的短语装配就容易了。

2. 该命令的完整形式

. LIST［显示表达式（或表达式列）］

［NEXT ＜范围表达式＞］［OFF］［FOR/WHILE

＜条件表达式＞］［TO PRINT］

执行状态的完整解释如下：（各种状态均为连续显示）

显示表达式　限定显示的内容及内容的规范和顺序。

NEXT　限定从当前记录开始，共显示 N 个（范围表达式的计算结果）记录结束，默认是从头开始，到文件尾结束。

OFF　限定关闭记录号显示，默认显示记录号状态。

FOR　限定显示满足条件的记录，默认所有记录。

WHILE　限定从当前记录开始，到不满足条件时停止。

PRINT　定义输出送到打印机，由于涉及了设备，要用 TO 进行设备的引导。默认状态是输出到显示器。

条件表达式　提供对显示记录的过滤器，滤除不满足指定条件的记录。

注意：FOR 和 WHILE　只能选择其一。

为了能对这些状态有更全面更细致的理解，再对细节问题做如下讨论。

显示表达式可以是前面介绍的字段表列。

没有字段表列，默认所有字段，同时默认数据库结构的字段顺序。如有，仅显示在字段表列中规定的字段并遵循字段表中规定的顺序。

如果表达式中包含运算，所有参与表达式的字段和变量在类型和运算关系上必须合法，类型不同要通过函数转换。如果显示表达式中有内存变量参与，变量必须已经被定义。

如果使用表达式列，每个表达式必须用‘,’分割。

对表达式的支持较单纯的支持字段大大加强了该命令的功能性，但该命令是基于数据库的显示命令，必须以活跃的数据库为命令依据，在没有数据库支持的情况下，即使表达式中没有字段(仅有内存变量)，同样被系统认为是非法的操作方式。

3. FOR 和 WHILE 之间的差别 仅用 FOR 可在任何情况下完整的筛选出全部满足条件的记录。设想，若有一个很大的数据库并在使用索引文件，计算机将在自然顺序的数据库中以逻辑的顺序大幅度跳跃式的提取记录，然后与条件比较，决定是否输出。这无疑是很缺乏效率的。在许可的情况下关闭索引文件会大大提高命令的执行速度，只是牺牲了对顺序的要求。

如果符合条件的记录在某个索引文件的关键字中是连续的，WHILE 是可以体现速度优势的。先把指针定位到这组记录的开始（用后面讲到的快速定位），然后用 WHILE 限定条件，这时显示从当前记录开始，到不满足条件时就会立即停止。可见，这种方法借助了索引文件的支持，在满足了顺序要求的同时，缩小了搜寻范围，大大加快了查询速度。

关于 FOR 和 WHILE 的使用技巧在后面的拷贝（COPY）和替换（REPLACE）命令中也是经常碰到的。在不同的情况下，善于把相同的概念变通应用，这是很重要的学习方法。

4. LIST 与 DISPLAY 命令的区别 还有一条与 LIST 命令非常近似的 DISPLAY 命令。DISPLAY 的默认范围是单一记录，显示方式是分屏的，每显示 23 个记录暂停，按任意键继续，其余状态与 LIST 完全相同，可视情况选择。

显示命令的其它应用：

上面的介绍，仅限于 LIST/DISPLAY 命令在数据库显示方面的应用，实际上这两个命令还可以通过装其它导向短语，指向系统环境和设备。此时的 LIST/DISPLAY 仅在是否分页上有区别。

(1) LIST/DISPLAY STRUCTURE [TO PRINT]

指向数据库结构，显示当前数据库的名称、记录总数、最后一次的修改日期及所有字段的字段名、类型、长度和小数位数。

(2) LIST/DISPLAY MEMORY [TO PRINT]

指向内存变量的管理环境，显示当前定义的内存变量的名称、类型和它们的值，以及这些变量的个数和它们所占用内存的字节数，并显示尚可使用的变量个数、字节数及它们的总数。

(3) LIST/DISPLAY FILE [LIKE <框架>]
[TO PRINT]

指向 DOS 的目录管理，显示文件名信息。无 LIKE 限制，默认显示. DBF 文件，否则显示与框架匹配的文件。框架包括驱动器符、路径表和文件通配符。这是为兼容 DBASE Ⅱ 的命令，在 DBASE 环境中，使用与 DOS 兼容的 DIR 命令更简单些。

(4) LIST/DISPLAY STATUS [TO PRINT]

指向本系统设置的环境，显示系统可变状态的当前值。用户可用 21 个状态命令(SET)改变这些设置(详见状态命令组)。同时还显示所有打开的数据库名、别名、各索引文件名、各索引文件的索引表达式及数据库间的关联关系。这里提到的新概念在后面介绍。

14—2 拷 贝

顾名思义，这是文件复制的指令。

在该系统中的 COPY 命令与 DOS 系统中的 COPY 命令不同，区别在于它默认的操作对象是当前开放的数据库。执行的结果是复制当前数据库的全部或部分记录到一个新创建的数据库文件中去，通过输出格式的限定，它可完成与其它语言的数据通信。

1. 命令形式

COPY TO < 文件名> [NEXT <范围值>]

[FIELDS ＜字段表列＞]
[FOR/WHILE ＜条件＞] [SDF] /
[DELIMITED [WITH ＜定界符＞]]

将当前数据库文件的全部或部分内容复制到目标文件。

范围、条件的作用和默认状态与 LIST 命令基本相同，只是把输出由屏幕改向数据库文件。

(1) 先讨论默认的文件格式。如果装配 FIELDS 必须跟字段表列，不能是表达式。这是因为在数据库的复制时，只能在原结构的基础上剪裁或重组，并不能进行新的结构创建。如果缺席，默认原顺序和全部字段，文件类型仍是数据库文件。

由于以 FIELDS 引导，就要求使用标准的字段表，在其它命令中亦是如此。在这里字段表列和表达式有严格的区别，这也是与 LIST 命令在参数要求上的差异。

TO 可把输出引向到 DOS 设备的写操作。数据复制到由文件名指定的数据库，后缀为.DBF。

可以想象，如果在索引文件的控制下进行文件拷贝，其结果必是一个符合索引顺序的库文件。由此看来，该命令在大多数情况下可代替排序操作，其在资源占有和速度上都占有优势。

(2) 其它格式的选择（统称文本文件，默认后缀. TXT）

如果不想使用数据库的格式，可用短语对文件格式的限定。在需要与其它语言交流数据时，经常需要这样做。

①无格式文件。SDF 限定输出数据采用无格式方式，所需字段在无格式符限定、无类型区别的情况下用标准 ASCII 码顺序写出，记录定长并用回车结束。这种文件被称为无格式文本文件。它符合大多数高级语言的数据规范，是在不同的语言中进行数据通信的中间方式之一。

② 标准格式文件。DELIMITED 限定输出采用标准格式。在输出的数据文件中，采用系统规定的标准格式：字符字段用双引号标记，每个字段用逗号分开，每个记录用回车结束。这种格式

对数据进行压缩处理，C 型字段的尾部空格被调整，数字仅以有效位表示。数据这样的书写格式可构成与 BASIC 语言的数据通信。

③ 自定格式文件。WITH 必须在 DELIMITED 的引导才能使用，作用是把默认的字符字段的定界标记符（双引号）换成指定的符号，其余格式完全相同。仅在非常特殊的情况下，才会使用这种格式。这属于非标准的自定格式文件。

在选择文本格式文件时，范围、条件、字段表短语同样可被装配。

2. 拷贝数据的回收 对于用拷贝命令生成的标准文件，可用同样的限定短语收回到数据库中。作为数据通信，这是接受信息的形式。此处对前面的 APPEND 命令进行补充的说明。

追加命令

APPEND FROM ＜文件名＞［FOR /WHILE ＜条件＞］
［SDF］/［DELIMITED［WITH ＜定界符＞］］

此命令是拷贝命令的反向操作，把指定的数据库或文本文件以当前库的格式追加到末尾，当前库是目的文件。

SDF /DELIMITED 限定源文件是文本文件（. TXT），否则为数据库文件（. DBF）。

FOR /WHILE 限定录取满足条件的记录，否则所有记录将被追加。

（1）由数据库追加记录。数据只在字段名和类型都相同的字段间传递，其它则被忽略；源字段超长时 C 型被截取，N 型用“＊”代替。

条件中使用的字段，必须是两个数据库中同时存在的字段。

（2）由无格式文件追加记录。SDF 指定源文件是无格式文件，数据按字节逐个录入，不考虑类型，用回车分割记录。由于无结构，也就无字段可言。这是一种比较自由的数据通信方式，无论文本文件是如何产生，只要设计的数据库格式恰当，数据都能被有效录入。如果数据格式不好，会造成数据类型的混乱。

(3) 由标准及自定格式文件追加记录。DELIMITED 指定源文件是标准格式文件。如果继续使用 WITH 则指定源文件是自定格式文件。原来压缩的数据，可根据当前数据库的结构释放。由于有格式限定，系统检测录入数据的类型和长度，类型不对的数据被忽略，源文件数据超过对应字段长度，字符字段被截取，数值字段用 * 标识。

COPY 和 APPEND FROM 的格式在同一数据库结构间传递数据可以是无顾虑的，但在不同的数据库结构间传递数据就要注意，同样的格式限定短语在拷出和录入时是有些差别的。如果对文本数据进行了变通，这一问题就会更突出。

3. 拷贝的其它形式 除复制数据库记录外，COPY 命令还可以完成以下功能：

(1) 拷贝单一文件

COPY FILE ＜源文件名＞ TO ＜目的文件名＞

由于指定了 FILE 源文件可以是任意的一个文件，这是提供了一种通用的手段，文件名中可以有驱动器符和路径表，但不承认文件的通配符。

(2) 拷贝数据库结构到新的数据库

COPY STRUCTURE TO ＜数据库名＞

[FIELDS ＜字段表列＞]

复制当前数据库结构（无记录）到目标文件（. DBF)。除非有 FIELDS 限定，否则所有字段将被复制。

(3) 拷贝数据库结构到标准的结构数据库

COPY TO ＜结构数据库名＞ STRUCTURE EXTENTED

所谓的结构数据库是这样的一个数据库，它具有四个规定的字段，这四个字段的定义如下：

字　段	类　型	长　度
FIELD—NAME	字符型	10

FIELD—TYPE	字符型	1
FIELD—LEN	数值型	3
FIELD—DEC	数值型	3

如果当前数据库有 12 个字段，在命令执行后，该结构数据库就有与 12 个字段对应的 12 个记录，原数据库的 12 个字段的名字、类型、长度及小数位数分别被存放在 12 条记录中的 FIELD—NAME、FIELD—TYPE 、FIELD—LEN 和 FIELD—DEC 字段中，且顺序与原字段顺序相同。可见这是把数据库结构转换成一种标准形式记录的操作，这些记录存放在所谓的结构数据库中。这种操作究竟有什么意义呢？我们感兴趣的是它的逆向过程。让我们来研究创建数据库的另一种方式。

不管用什么方式生成，只要有这样一个标准的结构数据库存在且各字段中的数据无一例外的符合定义数据库结构的规范，我们可以用它的记录作为创建一个新数据库结构的依据。

CREAT ＜ 数据库名＞ FROM ＜ 结构数据库名＞

在目的数据库中，关于字段的定义来源于结构数据库的对应记录。

在交互状态下，这种操作没有明显的意义。如果在程序中要随机的生成或改变数据库结构，这个命令就成了达到目的的唯一手段。具体操作参看 17－6 节在程序中变更数据库结构的演示 。

14－3　替　换

对数据库中的数据进行有条件的成批替换是本系统非常有特色的功能，也是最常用的操作之一。

REPL [ALL/NEXT ＜范围值＞] ＜字段 1＞ WITH
＜表达式 1＞ [，＜字段 2＞ WITH ＜表达式 2＞
……] [FOR / WHILE ＜条件＞]

鉴于前面的叙述，仅对该命令做以下解释：

WITH　用表达式的值替换字段中的内容，类型要一致。

若字符表达式超长，字符被作截取处理。

若数值表达式超长，系统便显示溢出并中断或用 * 标识。

没有范围和条件的限定，仅对当前记录执行。

如果在活跃的索引键上执行替换，则相应的索引文件被更新。

特别需要指出的是，在当前的控制索引上进行成批的替换将会由于索引的即时更新造成状态的跳跃，使替换不能完整执行，造成替换遗漏的结果。

14—4　统　计

COUNT [ALL/NEXT ＜范围值＞]
　　[FOR / WHILE ＜条件＞]
　　[TO ＜内存变量＞]

对指定范围中满足条件的数据库记录计数。

如果屏幕报告处于显示状态，屏幕得到统计结果。

如果 TO 被选择，结果被存到指定的内存变量。如果变量已经存在，将被覆盖，否则被创建。

14—5　窗　口

显示命令在速度上的可控性太差，编辑命令在信息上显示得太少。如何有效利用屏幕资源，如何增加编辑的信息量，这就要使用窗口。对于数据库来说，屏幕的空间太小了点，但通过全屏幕命令可以得到弥补。

1. 窗口命令

BROWSE [FIELDS ＜字段表列＞]

这也是一个管理功能很强的命令。它允许用户使用一个能够在数

据库平面上任意移动的全屏幕的窗口，纵向每屏最多 17 条记录，横向以屏幕装满为界限。通过全屏幕操作键，可控制窗口在数据库上的相对位置，光标可指向任意记录的任意字段。对光标所在的字段，可进行编辑操作。如果光标超过文件尾，还可对数据库增加新记录。

用限定的字段表列可剪裁字段和指定字段顺序，缺席默认全部字段的自然顺序。

2. 用于基本控制的键盘指令

↑ ↓　前后调动记录指针，或说用光标指向记录。

→← 左右调动光标指向字段，到屏幕边缘自动移动指针。

PgUp 和 PgDn　前后翻页。

Ctrl＋ ←或→　窗口左右移动一个字段。

Ctrl＋Y　删除一个字段。

Ctrl＋U　记录删除标记开关，设置和恢复记录的删除标记。

Ctrl＋E（End）　存盘退出。

Ctrl＋Q 和 ESC　不存盘退出（只对未记盘的数据）。

Ctrl＋H　进入一个功能选择菜单，它被显示在零行，可用左右箭头调动光标，指向要求的功能项后回车。这些功能包括

Bottom　指针指向文件尾。

TOP　指针指向文件头。

LOCA　限定屏幕左侧固定的字段的个数，它不随 Ctrl＋ ← 或→移动，一经设置，在退出前有效。

Menu　改变 BROWS 可控制的 Menu 的开或关。

Record　指针指向由屏幕接收的一个记录号。

Freeye　指定编辑仅一个字段。光标仅在该字段上移动。

3. 使用的变通　如果要对满足一定条件的记录进行全屏幕编辑，不妨使用状态命令设置过滤器。

SET FILTER TO ＜ 条件表达式＞

此时，数据库在外观上看，就像只有满足条件的记录组成，其余

记录被当前状态所隐蔽。发出不带条件的

SET FILTER TO

命令可使过滤状态解除。这种设置过滤器的做法，在关系到数据库整体操作的任何情况下有效，如拷贝、替换、列表等。

全屏幕窗口是一个强有力的编辑功能，但如果操作对象是一个有严格的数据规范的数据库，这种操作常常带有破坏数据规范的危险性。在需要保证数据规范时，应先拷贝数据到临时库，然后再进入全屏幕编辑。

14—6 指 针

指针操作是数据库记录定位所不可缺少的手段。前面仅作了简单介绍，现作进一步的说明。为了说明指针，先介绍两个函数。

1. 文件的头、尾函数

BOF () 文件头函数。

当文件指针超越了文件的第一个记录，该函数返回逻辑值为真（. T.），否则返回假（. F.）。

EOF () 文件尾函数。

当文件指针超越了文件的最后一个记录，该函数返回逻辑值为真（. T.），否则返回假（. F.）。

注意，此处的文件头、尾不一定是文件的第一和最后的记录，只有在不使用索引文件时，这种状态才成立，否则头、尾状态以记录的逻辑顺序为准。再有，这里的文件头是在开始记录之前，尾在最后一个记录之后。只有超出了文件记录的范围以后，这两个函数才返回真值。其返回的逻辑值可作为控制参数或用于逻辑表达式。

2. 指针的绝对移动

GO/GOTO ＜ 数值型表达式＞

指针被定位在由表达式值所确定的记录号。其值必须在合法

记录号的范围之内，小数自动取整。

GO 5+3　　指针被指向记录号是 8 的记录。

GOTO 8　　与上等价

指针的绝对移动可适应各种情况，这是到达已经被系统状态所隐蔽的记录的唯一定位方法。

为了辅助指针的绝对移动，系统还规定了两个很有用的参数

GO TOP/BOTT　指针指向数据库开始/最后的记录。

这是限定短语，它确实是把指针定位到文件开始的记录或最后的记录。如果使用索引文件，它们服从逻辑顺序。区别于函数 BOF () 和 EOF ()，它们并未到达超越记录的实际范围，也没有值的概念，也不能参与逻辑运算。这是初学者经常含糊的两个概念，要用心体会它们之间的区别。

3. 指针的相对移动

SKIP ＜数值型表达式＞

相对当前记录，按表达式的值移动指针。正值向后，负值向前。其值必须在合法的范围之内，小数自动取整。

SKIP 5　　从当前向后移动 5 个记录。

SKIP 8－12　从当前向前移动 4 个记录。

如果有索引文件在使用，其相对位置由逻辑顺序决定。

如果设置了过滤器，滤除的记录不被计算在内。

如果隐蔽了标记删除的记录，这种记录不被计算在内。

4. 记录的查找定位

LOCA [NEXT ＜ 数值表达式＞] FOR ＜条件表达式＞

顺序查找数据库中满足条件的记录。

从头开始查找，发现第一个满足条件的记录便停止，指针定位到该记录；如果没有找到满足条件的记录，则指针定位到文件尾或限定范围的最后一个记录。如果到达文件尾 EOF ()，将返回真。

NEXT 不仅限定顺序查找记录的个数，还同时限定从当前开

始查找。在 限定范围内没有发现满足条件的记录时，指针被定义到限定范围的最后一个记录。范围超过文件尾时，将 EOF（）置真返回。

只要定位成功，屏幕显示当前记录号

Record＝ 记录号值

如果这种显示干扰操作环境，可用状态设置命令封锁

SET TALK ON/OFF

这是屏幕环境控制开关，ON 将命令的执行结果的报告送屏幕显示，OFF 将隐蔽这些报告（不是不执行，也不是没有结果，只是不送屏幕）。开关一经设置，对屏幕显示执行报告的任何命令同时生效。如拷贝、替换、索引、变量赋值、定位等等。使用专用的显示命令产生的屏幕输出不受此限制。

LOCA 还有一条后续的命令

CONT

其作用是从当前开始，恢复对最后一条 LOCA 命令定义条件的查找。只有在 LOCA 命令发出，指针又未到达指定范围时，此命令才能使用。其余状态与 LOCA 完全相同。

5. 在索引中的快速查询 数据库在索引文件的控制下，可利用索引特征对记录进行极快速的定位操作。对于较大的数据库，这是最有效率的定位方式。定位操作可在瞬息间执行完毕。

SEEK ＜表达式＞

把指针定位到与表达式值匹配的第一个记录。其表达式的类型受索引类型的制约。如没有与表达式值匹配的记录，指针定位在文件尾，EOF（）置真。SEEK 命令不能定位到有重复索引值的后续记录。对后续记录的操作一般先定位到第一个记录后，再用 SKIP 到达所需要指定的记录。

6. 不同定位方式的归纳 快速定位（SEEK）效率最高，但必须有索引文件支持，对有大量的定位要求时是首选方式。相对定位适应各种要求的处理，尤其是在承认记录隐蔽的情况下。绝

对定位同样具备速度优势，在定位要求明确时经常使用，其指向隐蔽记录的功能在特殊的情况下有一定的价值。顺序查找的速度太慢，只有不得不在索引之外寻求某个特征或数据库很小的情况下才考虑使用；如果在不追求速度（或万不得以）的情况下与恢复查找（CONT）配合，也能处理一些实际的问题。

14—7 数据的格式化处理

@命令用于为输入输出建立一定的格式，在给定的坐标下按指定格式输出信息。这是DBASE系统唯一的格式处理命令。

1. 句法

@ ＜行，列＞[＜SAY 表达式＞ [PICTURE ＜子句＞]]
[GET ＜变量＞ [PICTURE ＜子句＞]]
[RANGE ＜表达式，表达式＞] [CLEAR]

2. 坐标的行和列是数值表达式

3. 对屏幕格式的支持 对不同的显示器，列坐标统一从0到79排列。行坐标对25行屏幕从0到24排列，第25行相当于10行屏幕的DOS提示行；对10行屏幕从0到9排列。

第0行是系统的状态提示行，因此，用户使用第0行应考虑到系统提示的干扰。

4. 对打印机格式的支持 对大多数打印机，行坐标也有限制，在连续的@命令中减小行数将会引起换页，减小列数也会有类似结果。

5. 数据的显示和编辑

SAY子句仅可对任何表达式的值进行格式化显示。

GET子句在格式化显示变量或字段信息的同时可支持用户键盘对数据的编辑操作。用于编辑的变量或字段必须存在，否则出现系统的错误提示。该子句是屏幕数据与键盘联系的桥梁。

GET仅以类似全屏幕的方式显示，并不是全屏幕状态，但它

必须在全屏幕命令 READ 的激发下，才能实施接收屏幕数据的操作。如果用 APPEND 激活由 GET 控制的数据库记录，用户将能够对摘要型字段进行编辑或初始化，READ 则不行。GET 如果没有激活命令的支持，该子句将被忽略。

6. 数据合法性的鉴定 RANGE 子句将引发系统对屏幕输入的数据即时进行合法性验证。在子句中可用表达式对数值变量或日期变量的上下限进行指定，系统一旦发现输入数据超出范围，在第 0 行给出非法数据的提示并指示按空格键返回编辑状态，直到数据合法为止。

7. 环境控制

@行，列 CLEAR　清除该坐标以下的矩形区域。

@行，列　清除本行指定列后的屏幕信息。

8. 格式字的选择

(1) 子句结构。PICTURE 是用户格式化数据的实施子句，真正的格式化能力要靠该子句采用的格式限定字或格式限定字符串。系统将强制输出满足限定的格式。

描述格式限定的字必须用 @开始，格式字紧跟其后，共同完成格式的限定。格式字对描述对象的数据类型有一定的限制。如果出现格式字和数据类型的冲突，系统将忽视格式限定。如果格式字和格式字符串同时使用，它们之间必须用空格隔开。

PICTURE 子句支持格式字或格式字符串的变量形式，可用 C 型变量的值表达格式限定的要求，这将会使格式处理的手段更加方便和灵活。

(2) 格式字。用于 N 型数据。

C—— 显示 CR（贷方）在一个正数之后。

X—— 显示 DB（借方）在一个负数之后并加圆括号。

（C 、X 仅限定 SAY 使用）

B—— 把数值数据向左对齐。

Z—— 把数值数据 0 作为一个空白字符串显示，用于 D 型数

据（可按 D 型格式提供 C 型数据）。

D—— 美国日期格式。

E—— 欧洲日期格式用于 C 型数据。

A—— 仅仅是字母表的字符。

! —— 仅仅是大写字母。

R—— 描述中的文字可被作为数据格式的一部分混合在显示的数据之中，但它们不被包括在变量或字段的实际值中。

在不发生冲突的情况下，功能字可进行多于一个的装配，如用 '@XC' 共同作用于一个数值字段，将会在负数后显示 DB 而在正数后显示 CR。

(3) 格式字符串。它侧重于建立数据与格式字符在位数上的一一对应关系，格式字则侧重于对数据的整体进行限定，这是两者在作用上的区别。如果对这种区别感到抽象，可参看 14－8 节中的应用实例。在命令格式上，格式字可用一个字符完成限定，格式字符串则必须通过使用表达意图的全部字符位来完成。格式字符串在对数据类型的约束上要少一些，对不同类型的数据有可能使用同一种格式符。

可参与格式字符串的字符集如下：

9—— 仅允许 0～9 的 10 个数字，用于字符数据。

—— 仅允许数字、空格和正负号。

A—— 仅允许字母。

L—— 仅允许逻辑数据。

N—— 仅允许字母和数字。

! —— 将字符变换成大写形式且对其它字符无影响。

$ —— 在前置零的位置上显示 $ 符号。

* —— 在前置零的位置上显示 * 符号。

. —— 指定十进制数小数点的位置。

, —— 在数字中确定“,”的位置，仅当数字在左侧时才显示。

唯一能有效地用在 SAY 和 GET 中的符号是 !，它不关心大

小写键的设置和变量值是否是大写的，都将以大写字母的形式显示信息或接收数据。$ 和 * 仅被用于 SAY，其余的符号仅用于 GET ，因为它们不能对 SAY 产生影响。

假如一个 PICTURE 描述被用于 GET 的一个非整数，则必须在描述中明确小数点的位置。

在举例中的 ROW () 是屏幕的当前行函数，COL () 是屏幕的当前列函数。它们可用于被控制坐标的运算。

14—8 格式化应用举例

1. 环境控制

. @ 5，30 CLEA

清除 5 行 30 列至屏幕右下角的矩形区域。

. @ 4，20

清除第 4 行的 20 列以后到本行结束。

2. 格式显示

. 工资合计＝18294.35

. @ 5，12 SAY 工资合计 PICTURE '$99，999，999. 99'

显示结果是

$18，294.35　　这是位的不严格限定

. @ 5，12 SAY 工资合计 PICTURE '$$$，$$$，$$$. $$'

显示结果是

$$$，$18，294.35　　这是位的严格限定

. YUBL＝－2.56

. @ ROW ()，0 SAY YUBL PICTURE '@X ****，****，****. **'

******2. 56DB　　显示贷方指示符 DB

注意：$ 和 * 格式符可以相互补充并可与格式字 X 结合使

用。

. 当前日期＝'19910804'

. @ 9,20 SAY 当前日期 PICT '@R 当前日期 XXXX 年 XX 月 XX 日'

显示结果是

当前日期 1991 年 08 月 04 日

@R 格式将指定的格式字符嵌入了变量之中，GET 子句也可使用这种@R 格式字。

这些输出格式可用于屏幕显示，也可用于数据的打印输出。

3. 变量编辑

. 工资合计＝18294.35

. @ 5，COL（）＋2 SAY '工资合计' GET 工资合计 ；

PICTURE '999，999，999.99'

. READ

显示结果是

工资合计　　18，294.35

变量的值将用反相方式显示，此时光标停在变量的开始位置，可用键盘改变变量的值，数据位填满自动退出，可用回车键提前结束编辑。

可用命令查询编辑后的结果，如

. ? 工资合计

20345. 98

. 年＝1991

. 月＝03

. 日＝21

. @ 5，COL＋1 GET 年 RANGE 年，年－42

. @ 5，COL＋1 SAY '年' GET 年 RANGE 1，12

. @ 5，COL＋1 SAY '月' GET 月 RANGE 1，31

. @ 5，COL＋1 SAY '日'

. READ

屏幕显示

1991 年 3 月 21 日

可用键盘对三个变量进行编辑，无论哪一个值超过了限定的范围，屏幕立即提示错误信息，用空格键返回编辑状态，重新输入到数值合法为止。

这是基于参加工作时间的示范，对于日期概念，这种限定的方式是不严谨的，目的是说明数据上下限是如何实现的。

关于变量的编辑虽然支持交互状态，但主要是用于程序设计。程序中的屏幕环境和格式用@ 命令实现和规范是最简单，当然还可有其它的办法，但实现就会困难得多。

第十五章　多工作区操作

15—1　为什么要用多工作区

单一的数据库在应用中局限性很大，尽管可以使用的字段数很多，但在处理实际问题时不一定能满足需要。很多情况下要求必须同时打开多个文件。使用字典支持主用数据库就是经常见到的实例，它就要同时使用两个数据库。同时开十个数据库看来已经不少了，但在设计复杂的系统时仍会感到受拘束。如果没有多区的支持，就会使很多任务的完成处于困境。这是多区工作的必要性。

我们知道，打开一个数据库，要对系统申请环境，系统响应申请时要做好资源和设备的全面准备，这是一个十分复杂的过程。频繁的开闭文件不仅在时间的花费上令人难以忍受，对设备也是有损耗的。避免这种缺乏效率的操作的有效手段是保持频繁使用的文件随时处于打开的状态。如果同时使用多个文件，就必须同时开辟多个工作环境。在不开闭文件的前提下，通过选择工作区的方法达到操作不同的文件目的，无疑是个高效方便的好办法。

15—2　工作区

一个工作环境，只能有一个数据库处于工作状态，要想同时打开不止一个数据库，就要设置相应的工作环境，这个环境称为工作区。DBASE 可承认 10 个分别独立的工作区，就是说可有 10

个数据库同时相互独立地被使用。但是对于一个指定的时刻，只能有一个工作区是属于当前的，所有的库操作命令仅以当前区数据库为对象的。对于该区的数据库可进行全方位的操作。

相对当前工作区，其它各区的数据库（如果有的话）虽然是打开的，但在使用上要受到相当的局限。它们处于次要的活跃状态，虽然数据可以被使用，指针也可通过间接方式移动，但不能进行任何与变更数据有关的操作。在实际运用时，往往是在需要的时候，把当前区在各工作区之间进行切换，并在频繁的切换中达到分别操作各个数据库的目的。

没有相互独立的环境，数据库间势必形成干扰。在一定的情况下，仅是相互独立的工作环境又会感到呆板。如果能够构成数据库的联动或接通各数据库间的通信渠道，无疑会加强数据库的应变能力和表现能力。

15—3　文件别名

系统在打开数据库时，承认文件的全名；在打开之后，就只承认主文件名。如果直接使用主文件名相同的数据库，将不能被系统接受。同时打开数据库和它的备份文件时，就属于这样的情况。要达到同时打开有相同的主用文件名的文件，系统提供了别名的支持方式。

1. 重申 USE 命令的完整形式

USE ＜数据库名＞［ALIAS ＜别名＞］

［INDEX ＜索引文件表列＞］

其中 ALISA 是定义当前数据库的代替名称，或者说是数据库的别名。相同主名的数据库可通过指定不同的文件别名达到同时打开的目的。

假定某区已经有数据库 RS. DBF 被使用，在其它区试图打开数据库 RS. BAK，由于主文件名冲突，仅用 USE RS. BAK 的命

令形式不能使 RS. BAK 进入活跃状态，而用

USE RS. BAK ALIAS BB

则可以达到这一目的。在以后使用有关数据库名或工作区名时，“BB”可以代替 RS. BAK 中的 RS，从而避免了与 RS. DBF 中的 RS 相互冲突。

在没有冲突的情况下，别名可不特别指定，系统默认的文件别名是数据库名的本身。

文件别名的概念，不仅可用于代替文件参数，还可作为工作区参数使用。

2. 工作区的转移

SELE ＜ 工作区指定符＞

系统在接受该命令后，将当前的操作环境转移到由标识符确定的工作区。

标识符可采用以下几种方式：

（1）数值。可用 1—10 指定十个工作区中任意一个。

（2）字符。系统默认字符 A—J，顺序对应工作区 1—10。

（3）数据库名。用数据库名可指向所在的工作区。

（4）别名。别名可代替文件名指向其数据库所在工作区。

在开创一个新工作区时，必须使用数字或字母的形式进行指定，因为此时该区尚未使用数据库。一旦数据库被打开，它的文件名和别名（如果有的话）即可当做新的工作区参数使用。

请看举例：

SELE E	置 5 区为当前工作区，E 的顺序号与 5 相对应
USE GBUJ. BAK	打开 GBUJ. BAK（必须是数据库格式）
SELE 2	将当前区指向二区
USE GBUJ ALIAS GB	用别名 GB 打开 GBUJ. DBF 库

SELE 3　　将当前区指向三区

USE GBZD　　打开 GBZD 库此时当前工作区是三区

SELE B

SELE GB

SELE 2

最后三条命令等价，参数 B、GB、2 对上述环境都是指向第二工作区的合法参数，可选用任意一种形式完成转向二区的操作。

3. 跨区操作　每个使用的工作区都有一个当前记录，其数据可被其它工作区提取。各个工作区中的字段，在跨区使用数据时应在字段名前做出标识。标识方法是工作区符加 —>，如

E—> 姓名　　五区的姓名字段

B—> 工资　　二区的工资字段

C—> 汉字代码　三区的汉字代码字段

与之对应的标识方式还可以使用库名或别名

GNUJ-->姓名

GB—>工资

GBZD—>汉字代码

因为二区与五区的数据库同名，采用别名可区分开两区字段。

与此同理，特意指定内存变量可用 M—> 符号标识。当内存变量和字段名发生冲突时，系统优先使用字段；如果限定使用内存变量就要对内存变量做出标识。例如，“M—>工资”特指内存变量“工资”。

4. 建立多区之间的相互关系　通过状态的设置，可建立多工作区之间的相互关联状态。

SET RELATION TO <关联键值> INTO [< 别名>/
< 表达式>]

关联键值是本工作区对别名指定的工作区建立匹配关系的特

征依据。当前工作区的数据库为主动数据库，用别名指定的为被动数据库。当主动数据库的指针移动时，被动数据库的指针自动跟踪主动数据库的关联键值特征。

对关联键值作几点说明：

(1) 关联键值不同于关键字，它不是索引的概念。

(2) 关联键是一个表达式，值是表达式的计算结果。

(3) L 、M 型表达式不可用于关联键。

对关联关系作如下说明：

(1) 关联键值是 N 型，有两种可能。

①被动区没有打开索引，自动用记录号与主动区匹配。

②被动区打开 N 型索引，用索引键值与主动区匹配。

(2) 主动区用 C 、D 型关联键，被动区有三种对应情况。

① 没有索引，用当前记录与所有记录匹配。

②有类型相同的索引键，用索引键值匹配。

③关联键与索引键类型冲突是非法的匹配形式。

(3) 如果被动键用表达式指定，它必须是被动区的索引键。

(4) 被动区有多个相同索引键的记录，仅第一个被用于匹配。

(5) 被动区没有与关联键对应的记录，用文件头或尾去匹配。

(6) 主动区的关联键可以是索引的，也可以是不索引的。

(7) 在发出关联命令的当时，被动区的指针并不即时响应，但这种状态被系统监测，此后只要主动区的指针变化，被动区就会立即进行跟踪。

为了进一步说明关联的状态和如何实际应用，请看举例：设定当前为二区

SET RELATION TO RECNO () INTO E

或用 SET RELATION TO RECNO () INTO E—>RECNO () 通过记录号函数建立与五区的关联，记录间用记录号一一对应。形象的动作是二区移动指针到一个新值时，如 12，五区的记录指针同样被移动到与二区记录号相同的记录，即 12 号记录。如果主动

关联键是数值结果，被动区的指针将移向与这一结果匹配的记录号，小数自动取整。这仅限于被动区没有索引的情况。又如

. SET RELATION TO RECNO () +100 INTO E->
RECNO ()

这里指定二区用记录号加 100 后与五区记录号对应。用形象说明这一状态，当二区记录号移向 103 时，五区记录号将移向 203 号记录，依次 104 对 204 ，105 对 205 ，顺序对应。

在文件的数据替换、格式转换、纵向数据的横向排版等情况下，这种无索引的记录关联有重要的作用。下面是通过索引建立关联的例子。

. SET RELATION TO 技术职务 INTO GBZD

这里用数据库名指定工作区是合法的。被动工作区中的索引一般应是与主动区的数据有对应关系的，如果这种关系不存在，虽然命令本身合法，但实际的关联是主动区所有记录与被动区的一个空值记录对应，这显然是没有意义的。

这个关联的概念比较抽象，单纯靠文字叙述理解起来有些困难，现举出一个解决实际问题的例子来说明建立数据库的关联概念。实际问题是这样的：

在二区主数据库（RSUJ）中有“技术职务”字段，由于汉字的排序与定义无关，我们要给“技术职务”赋予一种有序的编码，用于解决汉字的顺序问题，数据库中已经定义了“代码”字段，准备存放编码数据。

数据很多，技术职务的分类也很多，无论是手工填写还是分类替换，操作都会很繁琐。此时可考虑采用数据库的关联处理方式。因为处理类似的问题，关联操作最容易达到目的。

在三区字典库（RSZD）中有两个字段，“代码”和“汉字”分别有如下内容：

代码	汉字
11	高级工程师
12	工程师
13	助理工程师
14	技术员
21	高级会计师
22	会计师
23	助理会计师
24	会计员
31	高级经济师
32	经济师
33	助理经济师
34	经济员
↓	↓
等等	等等

这是我们编码依据的版本，我们的目的是用这个编码规范对主数据库中的“技术职务”项填写所有代码。完成的命令清单为

SELE 3	指定到三区
USE RSZD	打开 RSZD 人事字典库
INDE ON 汉字 TO RSZD	用“汉字”字段建索引，该命令是为与主数据库高速匹配准备条件
SELE 2	转向二区
USE RSUJ	打开 RSUJ 人事数据库
SET RELA TO 技术职务 INTO C	用数据库中的技术职务作为与字典库建立关联关系
REPL 代码 WITH C—> 代码 ALL	完成代码写入

主数据库的指针从头开始逐个移动，每指向一个新的记录，系统首先取出二区“技术职务”字段的值，然后立即转向三区搜索与该值对应的记录，并将记录与二区记录建立关联后，再执行指定的替换。假如二区某一记录中的“技术职务”栏中存有“会计师”，系统扫描到该记录后立即转向三区找到“汉字”字段内容为“会计师”的记录，并将该记录与二区记录相联系，在执行 REPL 命令时，三区记录中的“代码”字段内容“22”被写入了二区当前记录的“代码”字段。当二区指针顺序移动时，这一过程被重复执行。本例的执行结果是把所有与“技术职务”对应的代码依次送到二区的“代码”字段。

无论二区数据库有多少记录，也无论技术职务有多少种分类，所有代码均按字典提供的规范准确无误地一次填写完毕。二区数据库的记录越多，这种方法的优越性就显示得越充分，它不但速度快，操作简单，对保证数据的规范性亦有重要的作用。只要数据有可比照的特征，这种方式总是具备简捷、可靠的特点。

同理，这种关联关系还可在数据交换、计算、合法性检查等很多实际问题上得以应用，它还可在很大程度上简化编程。

第十六章　编　程

16—1　程序的概念

把为一定目的而对计算机所下的命令和提供的数据记录到文件，这个文件就叫做计算机程序。在程序中命令的多少，要看达到目的的繁简程度，还要受系统及环境的限制。程序默认的后缀名是. PRG。

为了达到既定的目的，在很多情况下，要求计算机具备对原始和中间数据进行判断的能力，并能根据判断的结果决定后续执行的命令。这就是程序中的控制概念。直线式交互命令固然需要，但仅此是不够的，必须提供一定的控制命令，才能使交互操作过渡到程序。

程序一经启动，大批的命令会被有条不紊地执行，可完成复杂的任务要求。它以文件的形式存在，可重复运行。程序比交互操作有更强的处理功能和更简便的操作方式。

16—2　程序的调用和返回

DO ＜程序名＞

无论是在交互或程序中，DO 命令可进入一个已经定义的程序文件。在程序中可用 RETU 命令返回到系统或调用处。如果是独立的程序，最后的 RETU 命令可省略，但在程序中间的某种条件下返回，RETU 命令就不能再省略了。

程序又叫过程，两者没有严格的界限。但过程文件一般又指多个过程汇合成的文件，不指一个单一的独立程序。

在程序的设计中要使用一些专用的结构命令，这部分命令在交互状态下是不被支持的。

16—3 可接受屏幕数据的命令

在程序中由键盘和屏幕接受数据是不可避免的，系统提供了处理这种问题的命令，它们虽被交互状态支持，但主要是用于程序设计。

ACCEPT［提示字符表达式］TO ＜内存变量名＞

INPUT［提示字符表达式］TO ＜内存变量名＞

把提示放到屏幕的当前行，等待接受键盘数据到指定的变量中。如果变量存在则被覆盖，否则被创建。用回车键结束输入。这是相同之处。两者仅在创建变量的类型上有区别。

ACCEPT 指定变量是 C 型，不要求输入字符串定界符。

INPUT 指定变量是 N 型，只接收数值、小数点和±号。

还有一条与 ACCEPT 相似的系统等待命令：

WAIT［提示字符表达式］［TO ＜ 内存变量名＞］

WAIT 不一定创建变量，TO 是可选的，不是必须的。如果创建变量，指定长度为一字节且不要求回车。WAIT 可接受不可打印字符，这对于控制程序的转移有特别重要和特殊的意义。如果不指定提示字符，那么它自定义提示“按任意键继续”被采用。

@ 是本系统功能最强、状态最复杂的命令，也是唯一能够对内存变量和字段进行格式化处理的命令。它只能对已经存在的变量进行显示和编辑，不能创建。详细的实用情况可参看 14—7、14—8 节。

16—4 结构命令

1. 简单的分支命令

IF <条件表达式>
　　[命令组 1]
　ELSE
　　[命令组 2]
　ENDIF

(1) 执行状态。表达式成立或者说表达式的逻辑值为真，第一组命令被执行，否则第二组命令被执行。

(2) 说明。结构以 IF ENDIF 为基本形式，必须成对出现。ELSE 语句标识第一组命令结束，可用可不用。

本结构语句用于一或两个分支的判断，虽然通过嵌套可组成多分支的逻辑结构，但由于格式繁琐一般不用。

2. 多分支命令

DO CASE
　CASE <条件表达式 1>
　[命令组 1]
　CASE <条件表达式 2>
　[命令组 2]
　　↓
　可重复使用
　[OTHER]
　[命令组 n]
　ENDCASE

(1) 执行状态。允许执行 n 组命令中的一组命令，这组命令被该条件表达式成立所限定。如第二个表达式成立，仅第二组命令被执行。当没有被满足的表达式时，也就没有任何一组命令被

执行。

（2）说明。结构以 DO CASE 和 ENDCASE 为基本形式，必须成对出现，可以嵌套。

每个 CASE 语句结束前一个命令组，数量不限。

OTHER 语句包括前面表达式限定以外的所有条件，可用可不用。在所有条件不被满足又不使用这个出口时，本结构块空过。

无论哪组命令被选中，本组命令执行完毕后，继续执行 ENDCASE 后面的语句。

本结构语句用于多分支的判断，各分支条件可以是同一数据的不同值，也可以是完全互不相关的多种表达式，因此该结构处理问题的随意性较大，能使程序明朗清晰。

对于有规律值的条件表达式，这种方式显得有些呆板，程序行过多，随着值域的扩大，程序篇幅拉长。对于有规律的分支控制，用变量合成或截取配合宏代换往往更简捷。

3. 循环命令

DO WHILE ＜条件表达式＞

［命令组 1］

［ EXIT ］

［ LOOP ］

ENDDO

（1）执行状态。只要表达式是逻辑真值，命令组就被重复执行，直到表达式出现逻辑假值。这种可重复的结构称为循环。

（2）说明。结构以 DO WHILE 和 ENDDO 为基本形式，必须成对出现，可以嵌套。

EXIT 是循环中断语句，是在满足循环条件时跳出循环的出口。它迈过在它和 ENDDO 之间的所有语句，执行 ENDDO 后面的语句。它是跳过循环尾的语句。

LOOP 是循环返回语句，它迈过在它和 ENDDO 之间的所有语句，重新由 DO WHILE 开始执行。它是跳到循环头的语句。

16—5 结构命令的嵌套

结构命令仅此三种，看起来不难，用起来也容易，但要用得好、用得巧就并非简单了，要好好动些脑筋才行。

结构命令本身和它们之间可以任意嵌套，但嵌套必须是完整结构的嵌套。交叉嵌套会破坏控制层次，造成不可预料的控制结果。

嵌套的层数有限制，一般不超过 24 层。

合法的嵌套和控制形式

```
DO WHILE ....
.....
IF ....
.....
LOOP（返到头）
ELSE
.....
EXIT（跳过尾）
ENDI
.....
ENDDO
```

观察 LOOP 和 EXIT 的应用

不合法的嵌套

```
DO CASE
CASE ....
DO WHILE ....
.....
CASE ....
.....
ENDCASE
.....
ENDDO
```

注：ENDCASE 和 ENDDO 在结构上出现了交叉

16—6 过程文件的使用

1. 为什么使用过程文件 系统规定最多使用 15 个文件的限制是很苛刻的，在很多情况下是不满足需要的。如果子程序设计得多，就会占用多个文件名，影响资源效率的发挥。

作为缓解矛盾的一种方法，系统提供了过程文件的手段。

过程文件是把最多32个程序组装成一个文件，使用时只占用一个文件名；过程文件一旦打开，其中的所有程序活跃，程序调用的反应速度会大大加快。在完成一个程序设计时，最后工作是把分散的子程序连接成过程文件。它的默认的后缀名仍是. PRG。

2. 过程文件的使用方法 在过程文件中，每个子程序必须用PROCEDURE 开头，后续该程序的名字。这必须是子程序的第一条有效命令。最后一条命令必须是返回命令 RETU，它标志着一个过程的结束。

例如在一个名为 GIWJ. PRG 的过程文件中有下列过程：

```
PROCEDURE GI1              定义过程一 GI1
  ......
RETU
PROCEDURE GIXY             定义过程二 GIXY
  ......
RETU
PROCEDURE GIQT             定义过程三 GIQT
  ......
RETU
    ↓                      (可用任何合法的程序名)
  等等                     定义过程不多于 32 个
```

在主程序调用过程文件中的子程序之前，先要激活过程文件，下面是主程序调用过程的实例：

```
SET PROCEDURE TO GIWJ              激活过程文件
  ......
DO GIQT                            调用过程 GIQT
  .....
DO GI1                             调用过程 GI1
```

.....

DO GIXY　　调用过程 GIXY

.....

SET PROCEDURE TO PROC　　关闭过程文件

CLOS 是关闭过程文件的另一种方式，两者选一。

16—7　程序的设计

程序设计是对语言掌握程度的检验，也是对归纳问题水平的检验，这两者之间是相辅相成的。单纯的注重语言技巧，容易华而不实；过于追求务实会使程序冗长，降低编程和运行效率。一般情况下，初学时侧重于处理实际问题的能力，待语言的表达方法基本掌握之后，开始逐步地向语言技巧发展。如果语言技巧达到运用纯熟的程度，归纳和抽象实际问题的程序模型就成了编程的主要方面。程序的综合水平是两者的全面发挥。

在学习编程的过程中，要注意几个具体问题。

1. 基本功要扎实　基本功是指对基础命令的掌握和理解。每涉及一条新命令，力求对其准确全面了解。一不凭想当然，二不投机取巧；对基础命令不求甚解，满足于似是而非，这是事倍功半的做法。实际上掌握命令的写法并不重要，关键是理解命令的执行状态和结果。没有对命令的深刻了解，就很难准确的使用。

2. 向计算机求教　每当使用一个新命令前，要对命令进行尽可能全面的考察，不迷信教材和书本。最好的教师是计算机本身，只要我们有心求教，它永远是最诚实和诲人不倦的老师。对于尚不理解的命令，上机考察是最好的办法。未经实际确认的命令最好不要写进程序。程序中盲目使用尚不了解的命令，即使通过，收益有限。等到程序运行出了问题，再去分析和考察命令的状态和功能无疑是一种非常缺乏效率的坏习惯。

3. 编程思路　每当开始处理一个问题，把程序写入计算机不

应当是开始。完全相反，更重要的是应当知道最后的结果是怎样的，达到这一结果可有多少手段，如果必要，还要对各种可选方案中的关键部分进行实际比较。开始书写程序的本身是程序总体设计的基本完成。尤其对于比较复杂的问题，这种设计思想是到达彼岸的捷径。

开始就写，写到哪算到哪，待大篇程序写了之后才发现问题，这并不是达到目的的可行途径，只好推倒重来。这种欲速则不达的情况即使对熟悉语言的程序员也是屡见不鲜的。

4. 勤于思考，善于总结 计算机面对杂乱无章的具体问题，总能够做出令人满意的回答，但其内部却只能识别两个字。我们不要求每个人都具备像计算机那样的归纳能力，但可从中得到有教益的启发。

如何把千变万化的具体问题，抽象成可以被计算机语言描述的模型，再进一步的将模型归纳成通用的模式，这是我们要说明的问题。

仔细研究会发现，很多程序的结构组合及命令组合出奇的相似，之所以做成不同的程序段，就是因为它们多少还有些差别。如果我们能够通过减少一些雷同的段落，用统一的模块去处理类似的问题，就会使程序精练，编程的效率就会大幅度的提高。

达到这一境界的前提是要勤于思考，善于总结。从平铺直叙到小范围的通用优化，再到更大的通用范围。这个过程是在思考和总结的基础上循序渐进的，不经过一个勤奋的努力过程，将永远停留在一般化的水平上。

5. 程序的组织 一般情况下，一个任务很难用一个程序段达到目的，尤其是较大的任务，要有很多模块组成。在多数情况下，子程序的应用不是简化编程的需要，而是解决问题的必由之路。

把一个复杂的课题，分解成一个个单体模块，再通过辅助命令对小模块进行装配，这就是所谓的模块化设计。对基础的模块，功能越单一越好，不求功能多，但求功能完美。程序单体的功能

越强，其可组合性就越差，程序中雷同的模块就会越多。究竟如何在实际问题中抽象出这样的子程序，就要看程序员的水平如何了。

构造具备一定通用度的程序要比直叙式程序难度大，设计的关键是如何传递变量和控制参数。通过使用可传递的参数，同一子程序可对不同的调用者呈现出不同的功能或状态。

一个程序段不要过大，这会使结构分析变得困难。把能够独立的功能块定义成子程序，将会使程序结构明朗化。

程序名称最好要有规范，这既可方便程序的引导，又会为今后的维护打好基础。

6. 程序与数据的相互作用 程序和数据是不可分割的，但不能混为一谈。

对于受控的数据，程序应尽可能的远离，数据的变化一般不应牵扯到程序。这会为数据的变更提供良好的环境。因为数据的变更而修改程序是经常发生的情况。这说明了程序的适应性不够强，其寿命也不会长久。

与此相反，对程序中可能会发生变化的控制数据，最好设计在程序之外，在不进入程序的情况下，就可改变程序的状态和功能。这将大大增加程序的可塑性。

以上论述，在不同的编程水平上会有不同程度的理解。这只是纸上的东西，程序设计的巧妙之处往往是只能神会而不能言传的。真正的编程能力是靠在实践锻炼中提高。

第十七章　上机学习的示范

这一章对 DBASE 中的变量、命令、函数、状态进行了带有综合性的演示，旨在帮助读者尽快的掌握这一语言。由于排版原因，个别结果在格式上略有出入，但结论相同。凡是用".."开头的行是输入的命令，其它行则是对演示所做的解释，上机不要输入。

这里选择的演示顺序、方法、内容都是经过斟酌的，要能够对照教材做一遍演示，并能达到理解演示目的的程度，就基本上掌握了该系统的主要部分。这种面对机器的学习过程比单纯的看书更有效率，更有实际意义。

17—1　关于变量和函数的演示

1. 数值型变量及有关函数　可进行各种算术运算的变量是数值型变量。

. N1＝12

将 12 赋给变量 N1，N1 为数值型

. N2＝3

将 3 赋给变量 N2，N2 为数值型

请观察有关数值的运算及结果

. ? N_1, N_2, N_1+N_2, N_1-N_2, N_1*N_2, N_1/N_2

12　3　15　9　36　4.00

以上对 N_1、N_2 的加减乘除运算说明 N_1、N_2 是数值型变量。

几种数值型函数的例子：

(1)开平方函数

SQRT(数值表达式)

对输入数值表达式的值开平方并返回结果。

```
. NN2=SQRT(N1/3)
. ? NN2
 2.00
```

数值 3 是常量,2.00 是开平方的结果。

(2)取整函数

INT(数值表达式)

对输入数值表达式的值取整处理并返回结果。

```
NN=24
NN1=5
. NN2=INT(NN/NN1)
. ? NN2
    4
```

数值 4 是取整结果。这里没有四舍五入,只返回最大整数。

(3)四舍五入函数

ROUND(数值表达式 1,数值表达式 2,)

数值表达式 1 是输入的数值表达式,数值表达式 2 是要求舍取的位数。

函数对输入的数值表达式的值返回舍入后的结果。

```
NN=26
NN1=3
. NN2=ROUND(NN/NN1,4)
. ? NN2 ,ROUND(8.7869032,3)
    8.6667      8.7870000
```

(4)其它数值型函数

EXP() 指数函数

LOG() 自然对数

2. 字符型变量

(1)定义字符变量的规则：

①字符串的赋值，常量字符串必须用定界符限定长度。

②定界符有三对 '',"",[]。分别限定字符串的两端。

③定界符可以嵌套，以最外端的为准。

(2)赋值及运算

. C1='　12'　将　12 赋值给 C1，C1 的长度为 4 字节

. C2='AB'　将 AB 赋值给 C2，长度为 2 字节

. ? C1+C1,C1+C2,C2−C1,C2+C1,LEN(C2+C1)

　12　12　12AB　AB　12 AB　12　6

LEN()是字符变量的测长函数，被测表达式长度为 6 字节。

加减运算在处理尾部空格上有区别，对先导空格无效。

(3)变通应用

. C3=["C2"+"C2"]　将"C2"+"C2" 赋值给 C3，用[]定界符

? C3,C3+'+'+C3　查询 C3 的内容，用"+"连接两个 C3

"C2"+"C2"　"C2"+"C2"+"C2"+"C2"

注：这种合成字符串的运算在程序中可随机生成命令，是处理随机问题的重要手段。例如：

. ? &C3　将变量 C2 进行由 C3 规定的运算

ABAB

把字符串转换成了算式。这是 & 的用法之一。

(4)字符型变量的比较

. C1='　12'　将　12 赋值给 C1

. C2='AB'　将 AB 赋值给 C2

? C1=C2,'　A'=C2,C2='　A'

.F. .F. .T.

此处的？用于对表达式的返回结果进行检验。

这里的表达式是逻辑表达式，不是完成赋值的命令状态。

①关系表达式在任意情况下只能返回逻辑结果。

②字符可比较大小，其顺序符合 ASCII 表的排列。

③字符串比较以等于号后的变量结束位为比较的终点。

(5)子串的检索。如果字符串被另一个长字符串包含，被包含的字符串相对长字符串称子串。

'AB'被'XABC'包含，'AB'称为'XABC'的子串，运算符为$。

. ? 'AB' $ 'XABC','XABC' $ 'AB'

.T. .F.

. CB=' ON'

. CD='INDE ON LL'

. ? CB $ CD,CD $ CB

.T. .F.

字符的 $ 运算逻辑结果检索成功则返回.T.，否则返回.F.。

这种运算在数据库查询时可有效的代替“或”运算，使表达式得以简化。有些类似模糊查询的意味。

(6)子串检索函数

AT(字符串 1 ,字符串 2)

如果字符串 1 被字符串 2 包含，则函数返回字符串 1 在字符串 2 中的起始位置的字节数，否则返回 0 值。

. CB=' ON'

. CD='INDE ON LL'

. ? TA(CB,CD) ,AT(CD,CB)

6 0

数值 6 是字符串CB 在CD 中的起始位的值。CB 不包含 CD，所以返回数值 0。

这种检索在数据的合法性鉴定时，有特别简便的优点。

(7)字符串截取函数

SUBSTR(字符串，数值表达式 1，数值表达式 2)

字符串是函数的输入值，即作为被截取的对象。

数值表达式 1 指定截取的起始位置，超长时返回空串。

数值表达式 2 指定截取多少位，缺席或超长时取全部。

```
. MC='1989 1990 1991'
. ? SUBSTR('FDJKHGLK',4,2)
KH
.? SUBSTR(MC,6)
1990 1991
```

这是处理字符运算的辅助函数，用于字符串的剪接或重组。

(8)字符串转换函数

UPPER(字符串)

将输入字符串中的小写字母转换成大写。

LOWER(字符串)

将输入字符串中的大写字母转换成小写。

```
. MD='ghkj12kgf'
. MC='WIUD12HEU'
. ? UPPER(MD),LOWER(MC)
GHKJ12KGF wiud12heu
```

只转换字母，其它字符不变。在处理字符规范上很有用。

(9)字符串调整函数和测长函数

LEN(字符串表达式)　　输入 C 型，输出 N 型

将返回被测试的字符串长度的字节数。

TRIM(字符串)

将剪掉输入字符串的尾部空格。

```
. MC='工程师          '
.? LEN(MC),LEN(TRIM (MC))
```

12　　6

由于MC变量由3个汉字(6字节)和6个空格组成,测MC的字节长度为12,经过调整函数TRIM()处理后,尾部空格被去掉,其长度变为6字节。这里的测长函数嵌套了调整函数,类似的方法在处理实际问题中会经常用到。再看一个MC的运算:

. ? MC+MC,LEN(MC+MC)

工程师　　工程师　　24

3. 日期型变量

(1)日期型变量的建立。日期型变量不能用"="直接赋值,需经函数转换。

CTOD(字符串)

将字符串转换成日期型。

字符串必须是合法日期的字符形式,否则转换成空值。

. C1='05/21/91'　　将字符串送C1变量

. C2='21/05/91'　　将字符串送C2变量

. D1=CTOD(C1)　　将字符串转换成日期型并送D1变量

. D2=CTOD(C2)　　将字符串转换成日期型并送D2变量

. ? C1,C2,D1,D2　　查询变量C1,C2,D1,D2的值

05/21/91 21/05/91 05/21/91　/　/

由于D2不能表达合法的日期,转换成日期型空值。

下面检测C1,C2,D1,D2的类型:

. ? TYPE('C1'),TYPE('C2'),TYPE('D1'),TYPE('D2')

C C D D

测试变量C1、C2、D1、D2的类型分别为C、C、D、D型。

. DX=DATE()

将计算机的系统日期送变量 DX，检测其值和类型：

```
. ? DX,TYPE('DX')
07/02/91 D
```

(2)求年函数 YEAR()。输入 D 型，输出 N 型，返回日期变量中的公元年数。

```
. ? YEAR(DX)
1991
```

(3)求月份函数 MONTH()。输入 D 型，输出 N 型，返回日期变量中的月份数。

```
. ? MONTH(DATE())
7
```

使用了函数的嵌套，与 DX 同值。

(4)求日函数 DAY()。输入 D 型，输出 N 型，返回日期变量中的本月的号数。

```
. ? DAY(DX)
2
. ? TYPE('DAY(DX)')
N
. ? DAY(DX) * DAY(DX) * DAY(DX),MONTH(DX)
+MONTH(DX)
        8      14
```

对数值型输出的验证。日期连乘，月份相加。

(5)求星期函数 DOW()。输入 D 型，输出 N 型，返回日期变量所指定的日期是星期几。

```
. ? DOW(DX)
3
```

以星期日为第一天计算，返回的 3 实际是星期二。

(6)求月份函数 CMONTH()。输入 D 型，输出 C 型，返回日期变量中的月份数。

. ? CMONTH(DATE())

July

7 月份的英语表示方法。

(7)查系统时间函数 TIME()。无自变量,返回系统时间。

. ? TIME(),TYPE('TIME()')

14:12:22 C

定长 8 字节,C 型。

4. 数值型变量与字符型变量的相互转换 举例:

. N1=12

. N2=3

. ? STR(N1)+STR(N1) 以字符串的形式相加。

12 12

STR()函数默认的转换长度是 10 个字节

. CN1=STR(N1)

. CN2=STR(N2)

. ? N1+N2, CN1+CN2

15 12 3

可以看出转换前后的区别,N_1、N_2 是数值型变量,运算结果为 15。CN_1、CN_2 是字符型变量,运算结果仍是字符串。

VAL(字符表达式) 将字符转换成数值

说明:

(1)返回数值型结果。

(2)默认转换长度取两位小数位。

(3)非数值字符则返回 0 值。

. CN1='AB20'

. CN2='20AB'

. ? CN1+CN2,VAL(CN1),VAL(CN2)

AB2020AB 0.00 20.00

. CN3='123.256'

. ? VAL(CN3)

123.26

5. 变量的嵌套

符号"&"是一种特别的函数，它可把变量的名称进行延伸，用变量的内容作为延伸后的变量名参加运算。这种延伸被称为宏代换。这是在程序中处理带有随机性问题的唯一途径。请看：

. C1='MM'

. MM=12

. ? C1,MM,&C1+&C1

MM 12 24

数值 12 被存贮在 MM 变量中，C1 中存有字符 MM，宏代换将变量 C1 的内容 MM 作为变量名，取出 MM 的值参与运算。变量名在变量的值上进行了延伸。如果变量是字段名，情况仍然相同。

. MM='12'

. ? &MM+&MM

24

如果变量内容是字符型的数值，宏代换可将这种字符变成数值。这是宏代换的应用方式之一。

. MC=&MM+&MM

. ? MC,TYPE('MC')

24 N

从运算关系和类型测试都证明了用 & 将字符转换成数值是可行的。

更进一步的使用是宏代换的嵌套，请看：

. MD='&'+'MM'

. MM='JJKK'

. JJKK=33

. ? MD, &MD, &MD+&MD

```
&MM      33      66
```

这是终点为数值的宏代换嵌套。更换终点变量为C型后：

```
. JJKK='33'
. ? MD,&MD,&MD+&MD
  &MM    33    3333
```

由于宏代换的层次不够，终点变量仍然保持字符型。

宏代换的变通使用可使很多随机问题的处理得以简化，但使用上有一定的难度。关键是要明确的想象出宏代换最终的值所表示的内容是否符合原始的意图。空洞的讲解很难理解宏代换的用途和作用，只有到非此手段不能解决实际问题时，这种抽象才会被真正的认识和理解。

17—2 数据库排序和索引的演示

```
. USE YUK              打开数据库 YUK
. LIST                 列表显示内容
```

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
1	0311	三处	张林林	女	6902	22	财务科	助会
2	0332	三处	张国琴	女	5906	32	人事科	干事
3	0133	一处	韩中民	男	4802	43	人事科	科长
4	0323	三处	刘有才	男	5510	36	劳资科	科长
5	0231	二处	李一同	男	6002	31	人事科	副科长
6	0223	二处	刘华珍	女	3510	56	劳资科	经济师
7	0312	三处	李国珍	女	6902	22	财务科	助会
8	0121	一处	张　望	男	5906	32	劳资科	副科长
9	0134	一处	李志利	女	7002	21	人事科	干事
10	0212	二处	刘　伟	男	5410	37	财务科	副科长

1. 排序命令演示

. SORT TO BHK ON 编号

用编号做关键字段排序，指定数据库为 BHK。

. SELE 2　　转向第二工作区

. USE BHK　　打开 BHK

. LIST　　显示 BHK 内容

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
1	0121	一处	张　望	男	5906	32	劳资科	副科长
2	0133	一处	韩中民	男	4802	43	人事科	科长
3	0134	一处	李志利	女	7002	21	人事科	干事
4	0212	二处	刘　伟	男	5410	37	财务科	副科长
5	0223	二处	刘华珍	女	3510	56	劳资科	经济师
6	0231	二处	李一同	男	6002	31	人事科	副科长
7	0311	三处	张林林	女	6902	22	财务科	助会
8	0312	三处	李国珍	女	6902	22	财务科	助会
9	0323	三处	刘有才	男	5510	36	劳资科	科长
10	0332	三处	张国琴	女	5906	32	人事科	干事

排序后的数据库符合编号的顺序。

2. 索引命令演示

. SELE 1　　转向第一工作区

. INDE ON 编号 TO BH　　用编号做关键字，指定索引文件为 BH

. LIST　　显示索引后的数据库

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
8	0121	一处	张　望	男	5906	32	劳资科	副科长
3	0133	一处	韩中民	男	4802	43	人事科	科长
9	0134	一处	李志利	女	7002	21	人事科	干事
10	0212	二处	刘　伟	男	5410	37	财务科	副科长

6	0223	二处	刘华珍	女	3510	56	劳资科	经济师
5	0231	二处	李一同	男	6002	31	人事科	副科长
1	0311	三处	张林林	女	6902	22	财务科	助会
7	0312	三处	李国珍	女	6902	22	财务科	助会
4	0323	三处	刘有才	男	5510	36	劳资科	科长
2	0332	三处	张国琴	女	5906	32	人事科	干事

其输出顺序与排序数据库完全相同，仅区别于记录号仍未改变。如果不对记录号特别要求，索引不仅可以代替排序，而且更方便。

. INDE ON 性 TO XB　用性别做关键字，索引文件名 XB

. LIST

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
8	0121	一处	张　望	男	5906	32	劳资科	副科长
3	0133	一处	韩中民	男	4802	43	人事科	科长
10	0212	二处	刘　伟	男	5410	37	财务科	副科长
5	0231	二处	李一同	男	6002	31	人事科	副科长
4	0323	三处	刘有才	男	5510	36	劳资科	科长
9	0134	一处	李志利	女	7002	21	人事科	干事
6	0223	二处	刘华珍	女	3510	56	劳资科	经济师
1	0311	三处	张林林	女	6902	22	财务科	助会
7	0312	三处	李国珍	女	6902	22	财务科	助会
2	0332	三处	张国琴	女	5906	32	人事科	干事

. INDE ON 龄 TO NL　用年龄做关键字，索引文件名 NL

. LIST

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
9	0134	一处	李志利	女	7002	21	人事科	干事

1	0311	三处	张林林	女	6902	22	财务科	助会
7	0312	三处	李国珍	女	6902	22	财务科	助会
5	0231	二处	李一同	男	6002	31	人事科	副科长
8	0121	一处	张　望	男	5906	32	劳资科	副科长
2	0332	三处	张国琴	女	5906	32	人事科	干事
4	0323	三处	刘有才	男	5510	36	劳资科	科长
10	0212	二处	刘　伟	男	5410	37	财务科	副科长
3	0133	一处	韩中民	男	4802	43	人事科	科长
6	0223	二处	刘华珍	女	3510	56	劳资科	经济师

. INDE ON 性+STR(龄,2) TO XBNL 指定性别和字符型年龄联合做关键字表达式,索引文件名为 XBNL。

. LIST

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
5	0231	二处	李一同	男	6002	31	人事科	副科长
8	0121	一处	张　望	男	5906	32	劳资科	副科长
4	0323	三处	刘有才	男	5510	36	劳资科	科长
10	0212	二处	刘　伟	男	5410	37	财务科	副科长
3	0133	一处	韩中民	男	4802	43	人事科	科长
9	0134	一处	李志利	女	7002	21	人事科	干事
1	0311	三处	张林林	女	6902	22	财务科	助会
7	0312	三处	李国珍	女	6902	22	财务科	助会
2	0332	三处	张国琴	女	5906	32	人事科	干事
6	0223	二处	刘华珍	女	3510	56	劳资科	经济师

. CLOS INDE　　关闭索引文件 。

17—3 拷贝命令的演示

```
. USE YUK
. COPY TO L1 FOR 单位='一'
```

取单位为“一”的记录到 L1 数据库。

```
C:L1. DBF already exists, overwrite it? (Y/N) Yes
```

当前目录中有同名文件存在,系统要求回答确认结果。回答 Y 则对同名文件覆盖,回答 N 则作废拷贝操作。

```
. SELE 2                    转向第二工作区
. USE L1                    打开 L1 数据库
. LIST                      显示 L1 记录
```

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
1	0121	一处	张 望	男	5906	32	劳资科	副科长
2	0133	一处	韩中民	男	4802	43	人事科	科长
3	0134	一处	李志利	女	7002	21	人事科	干事

L1 数据库中只有满足条件的数据。这是原数据库的子集。

```
. SELE 1                    转回第一工作区
. 3                         指针指向 3 号记录
. COPY TO LL NEXT 4         从当前开始拷贝 4 个记录到
                            LL 数据库
. SELE 2                    转向第二工作区
. USE LL                    打开 LL 数据库
. LIST                      显示 LL 数据库记录
```

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
1	0133	一处	韩中民	男	4802	43	人事科	科长
2	0323	三处	刘有才	男	5510	36	劳资科	科长
3	0231	二处	李一同	男	6002	31	人事科	副科长

4 0223 二处 刘华珍 女 3510 56 劳资科 经济师

这个子集数据库是用范围限定生成的。对应原数据库可发现,记录范围是从当前记录开始的。这一点很重要。

. SELE 1 回到第一工作区

. COPY TO L2 SDF FOR 单位='二'

提取单位是"二"的记录到 L2。这里使用了格式限定字句 SDF,数据被拷贝到无格式文件 L2. TXT。

. TYPE L2. TXT 显示这个无格式文件

可见与数据库相比它的确是无格式的。

0231 二处李一同男 600231 人事科副科长 111 大本

0223 二处刘华珍女 351056 劳资科经济师 154 初中

0212 二处刘 伟男 541037 财务科副科长 125 中专

. COPY TO L3 DELI FOR 单位='三'

提取单位是"三"的记录到 L3。这是采用了本系统的自定义格式,L3. TXT 是压缩格式的文本文件。

. TYPE L3. TXT 显示 L3. TXT

限定字符的符号可用 WITH 指定。

"0311","三处","张林林","女",6902,22,"财务科","助会"

"0332","三处","张国琴","女",5906,32,"人事科","干事"

"0323","三处","刘有才","男",5510,36,"劳资科","科长"

"0312","三处","李国珍","女",6902,22,"财务科","助会"

. ZAP 删除掉 YUK 中的所有记录

Zap C:YUK. DBF? (Y/N) Yes

因为这是危及数据安全的命令,系统要求回答确认信息。此处回答 Y ,记录被实施删除。

. APPE FROM L1 从 L1. DBF 数据库中追加记录

. APPE FROM L2 SDF 从 L2. TXT 无格式文件中追加

记录

. APPE FROM L3 DELI 从 L3. TXT 自定义格式文件中追加记录

再次显示数据库的内容

. LIST

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
1	0121	一处	张　望	男	5906	32	劳资科	副科长
2	0133	一处	韩中民	男	4802	43	人事科	科长
3	0134	一处	李志利	女	7002	21	人事科	干事
4	0231	二处	李一同	男	6002	31	人事科	副科长
5	0223	二处	刘华珍	女	3510	56	劳资科	经济师
6	0212	二处	刘　伟	男	5410	37	财务科	副科长
7	0311	三处	张林林	女	6902	22	财务科	助会
8	0332	三处	张国琴	女	5906	32	人事科	干事
9	0323	三处	刘有才	男	5510	36	劳资科	科长
10	0312	三处	李国珍	女	6902	22	财务科	助会

可见这三种追加记录的方法都达到了目的。

. USE　　关闭 YUK. DBF

. QUIT　　退出 DBASE 系统

17—4　指针调动和过滤器

. USE YUK　　打开数据库 YUK

. LIST　　显示数据库记录(以下演示以此为依据)

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
1	0311	三处	张林林	女	6902	22	财务科	助会

```
 2  0332 三处  张国琴  女 5906  32   人事科   干事
 3  0133 一处  韩中民  男 4802  43   人事科   科长
 4  0323 三处  刘有才  男 5510  36   劳资科   科长
 5  0231 二处  李一同  男 6002  31   人事科   副科长
 6  0223 二处  刘华珍  女 3510  56   劳资科   经济师
 7  0312 三处  李国珍  女 6902  22   财务科   助会
 8  0121 一处  张  望  男 5906  32   劳资科   副科长
 9  0134 一处  李志利  女 7002  21   人事科   干事
10  0212 二处  刘  伟  男 5410  37   财务科   副科长
```

```
. 3              指针移动到 3 号记录
. DISP           显示记录
Record#  编号 单位  姓名 性别 出生 龄   部门   职务
     3  0133 一处  韩中民  男 4802  43   人事科   科长
```

```
. 8              指针移动到 8 号记录
. DISP           显示记录
Record#  编号 单位  姓名 性别 出生 龄   部门   职务
     8  0121 一处  张  望 男  5906 32   劳资科   副科长
```

这是指针的绝对移动，在交互状态下可输入记录号后回车，定位指针到指定记录。

如果在程序中使用，数字前必须使用动词 GO 或 GOTO 方可。

```
. GO 3           这是程序的绝对定位方式
. GOTO 8         这是程序的绝对定位方式
. GO TOP         移动指针到文件的初始记录
. ? RECN()       查询记录号
```

```
                1        为1
    . SKIP -1            相对向前移动一个记录
    Record No. 1         记录号仍然显示1
    . ? BOF()            查询文件头函数
    .T.                  返回逻辑真值,此时指针已经超过了第
                         一个记录
    . DISP               查询记录内容
Record#  编号 单位  姓名 性别 出生 龄    部门    职务
      1  0311 三处  张林林 女   6902 22  财务科   助会
```

这里虽然显示了第一个记录,但这并不是当前的指针,不信再查文件头函数

```
    . ? BOF()
    .T.                  文件头仍然返回真值
    . GO 1               移动指针到1号记录
    . ? BOF()            再次查文件头
    .F.                  返回假,指针离开了文件头
```

这里有一个令人费解的问题,1号记录到底算不算文件头?回答是不算,函数的返回值明确说明了这一问题。为什么在文件头时要显示1号记录呢?应当想象出一个记录号是0的空记录,指针指向该记录时,BOF() 函数返回真,否则返回假。

系统在这个状态下存在了二意性,这是系统设计的一个小小缺陷。当必须使用文件头函数时,这个二意性是绝不容忽视的。

```
    . LOCA FOR 姓名='张国'
```

在数据库中搜寻姓名是"张国"的记录,在这里要很好的体会字符串的比较规则。

```
    Record = 2
```

2号记录满足条件,指针定位到该记录。这种记录的搜寻是从文

件头开始的，如果使用了范围的限定则从当前记录开始搜寻。

. LOCA FOR '张国'＝姓名

搜寻“张国”等于姓名字段内容。

End of Locate scope.

超过了搜寻的范围，搜寻失败，字符串的比较规定是以被动变量结束为终止的。如果使用完整的特征进行搜寻，两种表达式的结果是相同的。如果使用子串作为条件，两者会有很大差别。之所以采用这种带有模糊边界的比较，因为它可使检索的表达式简单，同时功能得以加强。实质上，这是对特征进行了简单的归类，这样的归类是符合人们的日常习惯的。

. GO 2　　调指针到 2 号记录

. LOCA FOR 姓名='刘' NEXT 6

在数据库中搜寻姓名是“刘”的记录，限定搜寻从当前开始的 6 个记录的范围之内。

Record ＝ 4　　应答 4 号记录满足条件

.CONT　　再次恢复上面的查询

Record ＝ 6　　应答 6 号记录满足条件

.CONT　　再次恢复上面的查询

End of Locate scope.　　超过了搜寻的范围

虽然后面还有 9 号记录满足条件，因超过了限定的搜寻范围，搜寻中止。此时指针被定义到指定范围的下边界。

. ? RECN()　　查询记录号

7　　当前记录是 7 号

. DISP　　显示确认

Record#	编号	单位	姓名	性别	出生龄	部门	职务
7	0312	三处	李国珍	女	6902	财务科	助会

LOCA 命令在使用索引文件时同样有效，只是在搜寻速度上要受影响。如果数据库很大，在执行 LOCA 命令前先关闭索

引文件是必要的。

请看过滤器的作用。

```
. SET FILT TO 部门='人'        限定数据库仅对满足条件的
                               记录开放
. LIST                         显示数据库记录
```

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
2	0332	三处	张国琴	女	5906	32	人事科	干事
3	0133	一处	韩中民	男	4802	43	人事科	科长
5	0231	二处	李一同	男	6002	31	人事科	副科长
9	0134	一处	李志利	女	7002	21	人事科	干事

满足条件的记录仅 4 个,其余记录被过滤器所隐蔽。

```
. COUN TO L
```

无条件统计,此时应包括数据库所有活跃记录。

```
      4 records
```

屏幕报告统计结果,仅承认数据库中有 4 个记录。

TO L 子句可创建变量,查询变量 L 的值

```
. ? L
4
```

L 的值是 4 ,这是统计的结果。

```
. SET FILT TO 部门='财'
. LIST
```

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
1	0311	三处	张林林	女	6902	22	财务科	助会
7	0312	三处	李国珍	女	6902	22	财务科	助会
10	0212	二处	刘　伟	男	5410	37	财务科	副科长

```
. COUN TO LL
```

屏幕报告统计结果，仅承认数据库中有 3 个记录，将统计结果送 LL 变量，如果 LL 已经存在，则将被覆盖。

```
                3 records
. ?             LL,         LL * LL
                3           9
```

验证变量是 N 型。

在过滤器条件下的指针移动：

. GO TOP	指针指向文件头，应当是 1 号记录
. SKIP	相对移动一个指针
Record No. 7	指针被定位到 7 号记录，不满足条件的记录被忽略，这是过滤器的作用
. SKIP	再次相对移动指针
Record No. 10	跳到 10 号记录
. SKIP	再次相对移动指针
Record No. 11	11 号是记录，此时已经到达文件尾
. ? BOF(),EOF()	查询文件头、尾函数
.F. .T.	文件头是假、文件尾是真
. SKIP −2	相对反向移动两个记录
Record No. 7	记录返回到 7 号
. SKIP −2	再次相对反向移动两个记录
Record No. 1	指针指向 1 号记录，此时到达文件头
. ? BOF()	查询文件头函数
.T.	返回真值

通过以上演示，在无索引文件下的指针控制和过滤器的作用应当很清楚了，文件的头、尾函数的状态如何也应当明确。

下面是在索引文件中的记录定位情况：

. SET INDE TO BH	打开编号索引文件
. LIST	显示数据库内容（这是下面演示的基础）

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
8	0121	一处	张　望	男	5906	32	劳资科	副科长
3	0133	一处	韩中民	男	4802	43	人事科	科长
9	0134	一处	李志利	女	7002	21	人事科	干事
10	0212	二处	刘　伟	男	5410	37	财务科	副科长
6	0223	二处	刘华珍	女	3510	56	劳资科	经济师
5	0231	二处	李一同	男	6002	31	人事科	副科长
1	0311	三处	张林林	女	6902	22	财务科	助会
7	0312	三处	李国珍	女	6902	22	财务科	助会
4	0323	三处	刘有才	男	5510	36	劳资科	科长
2	0332	三处	张国琴	女	5906	32	人事科	干事

. SEEK '02'

快速查询关键字为 02 的记录，这里仍然符合字符串的比较规范；虽有多个记录，指针仅定位到相同键值的第一个。

. DISP

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
10	0212	二处	刘　伟	男	5410	37	财务科	副科长

. SEEK '04'	在索引键中查询特征是 04 的字符串
No find	满足特征的记录没有被发现
. ? EOF()	此时文件尾函数置真值，查询该函数
.T.	返回真
. SEEK '01'	在索引键中查询 01，指针定位到相同键值的第一个记录
. DISP	显示结果

Record#	编号	单位	姓名	性别	出生	龄	部门	职务

8 0121 一处 张 望 男 5906 32 劳资科 副科长

. SKIP 相对移动一个指针

Record No. 3 指针定位到 3 号记录

可见这是逻辑位移，不仅数据库顺序受到索引文件的控制，指针同样如此。

. DISP 显示验证

Record# 编号 单位 姓名 性别 出生 龄 部门 职务

3 0133 一处 韩中民 男 4802 43 人事科 科长

. SKIP 再次相对移动指针

Record No. 9 指向 9 号记录

. DISP 显示验证

Record# 编号 单位 姓名 性别 出生 龄 部门 职务

9 0134 一处 李志利 女 7002 21 人事科 干事

请观察分析下面的指针移动的结果：

. SEEK '44'

No find.

. ? EOF()

. T.

. SEEK '0223'

. GO 6

. SKIP −1

Record No. 10

. SKIP 2

Record No. 5

. SKIP 3

Record No. 4

以上的条件特征都是实际值为常量，这仅是很有限的查询方式。

请看更灵活的变通应用：

```
. M='01'          将字符 01 赋给变量 M
01
. SEEK M+'33'     用变量 M 与字符常量 33 构成查询特
                  征，指针被定位到索引键值是 0133 的
                  记录
. DISP            显示验证
Record#  编号 单位   姓名 性别 出生 龄   部门   职务
    3   0133 一处  韩中民 男 4802  43  人事科  科长

. MM='0323'       检索特征赋值给 MM
. BB='MM'         BB 变量被赋值为 MM
. SEEK &BB        通过宏代换查询满足特征的记录
. DISP            显示验证
Record#  编号 单位   姓名 性别 出生 龄   部门   职务
    4   0323 三处  刘有才 男 5510  36  劳资科  科长
```

作为演示，这种宏代换看不出什么价值，但在程序中，查询特征往往是不可预知的或是随机确定的，此时宏代换就是必不可少的了。这仅是宏代换应用的最简单的形式，有时还要通过宏代换的嵌套才能解决问题。

. GO TOP

指向逻辑的文件头。这是不超越的定位，是指针字句，注意不要与文件头函数混淆。

```
. DISP            验证记录确实是逻辑的文件头
Record#  编号 单位   姓名 性别 出生 龄   部门   职务
```

8 0121 一处 张 望 男 5906 32 劳资科 副科长

. GO BOTT

指向逻辑的文件尾。这是不超越的定位，是指针字句。

. DISP 验证记录确实是逻辑的文件尾

Record # 编号 单位 姓名 性别 出生 龄 部门 职务

2 0332 三处 张国琴 女 5906 32 人事科 干事

17—5 记录删除、恢复及替换

. USE YUK

. LIST

Record # 编号 单位 姓名 性别 出生 龄 部门 职务

1 0311 三处 张林林 女 6902 23 财务科 助会

2 0332 三处 张国琴 女 5906 32 人事科 干事

3 0133 一处 韩中民 男 4802 43 人事科 科长

4 0323 三处 刘有才 男 5510 36 劳资科 科长

5 0231 二处 李一同 男 6002 31 人事科 副科长

6 0223 二处 刘华珍 女 3510 56 劳资科 经济师

7 0312 三处 李国珍 女 6902 22 财务科 助会

8 0121 一处 张 望 男 5906 32 劳资科 副科长

9 0134 一处 李志利 女 7002 21 人事科 干事

10 0212 二处 刘 伟 男 5410 37 财务科 副科长

. BH='编号' 将“编号”送变量 BH，这是为宏代换做基础

. DELE FOR &BH='03' 删除编号字段中特征字符

是 03 的记录

4 records deleted

屏幕报告命令执行结果，删除了 4 个记录。所谓的删除是做删除标识，并未实施。

. LIST FOR &BH='03'　　查询命令执行结果

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
1	*0311	三处	张林林	女	6902	23	财务科	助会
2	*0332	三处	张国琴	女	5906	32	人事科	干事
4	*0323	三处	刘有才	男	5510	36	劳资科	科长
7	*0312	三处	李国珍	女	6902	22	财务科	助会

记录前的"*"是删除的标记，记录仍然存在。

. GO 2　　调动指针到 2 号记录

. DELE NEXT 3　　限定删除 3 个记录

1 records deleted

报告删除了 1 个记录。因 2、4 号记录已经做了删除标识，报告的结果是本次命令实际标识的记录数。由于当前为 2 号记录，NEXT 3 限定了从 2 开始的 3 个记录，是 2、3、4 号，其中只有 3 号记录未做删除标记，因此这一操作只对 3 号记录做了一个标记，与屏幕报告相符。

. ? RECN()　　查询记录号，这是为了验证范围的终点指针

4　　记录是 4 号，指针定位到范围的最后一个记录

. DELE RECO 10　　指定删除 10 号记录

1 records deleted　　该记录被标识

. DISP　　验证执行结果

Record#	编号	单位	姓名	性别	出生	龄	部门	职务

10 * 0212二处 刘 伟 男 5410 37 财务科 副科长

指针被定位到10号记录，并对10号记录做了删除标志。

```
. 3                    指定3号记录为当前记录
. DISP                 显示记录，该记录是被标记删除的
Record#  编号 单位   姓名 性别 出生 龄   部门   职务
      3 * 0133一处  韩中民  男 4802  43  人事科  科长

. RECA                 恢复做了删除标记的记录
   1 records recalled
. DISP                 再次验证记录，删除标记被取消
Record#  编号 单位   姓名 性别 出生 龄   部门   职务
      3  0133 一处  韩中民  男 4802  43  人事科  科长

. SET DELE ON          设置系统状态，隐蔽做了删除标记
                       的记录
. LIST                 验证隐蔽结果，做了删除标记的记
                       录被隐蔽
Record#  编号 单位   姓名 性别 出生 龄   部门   职务
      3  0133 一处  韩中民  男 4802  43  人事科  科长
      5  0231 二处  李一同  男 6002  31  人事科  副科长
      6  0223 二处  刘华珍  女 3510  56  劳资科  经济师
      8  0121 一处  张  望  男 5906  32  劳资科  副科长
      9  0134 一处  李志利  女 7002  21  人事科  干事

. GO 2                 指定2号记录为当前记录
. REPL 龄 WITH 0 NEXT 2 将年龄替换为0值，限定两
                       个
```

```
        2 replacements          报告执行结果,替换了两个
                                记录
. 2                             恢复定位
. LIST NEXT 2                   查询替换的结果
Record#  编号 单位   姓名 性别 出生 龄   部门   职务
      2 *0332三处 张国琴 女   5906 0   人事科  干事
      3  0133 一处 韩中民 男   4802 0   人事科  科长
```

按道理2号记录不应参加替换,实际是参加的,因为它是用记录号定位的,其他方式不能定位指针到隐蔽的记录,这种状态有好处,也有含糊的方面。请看其它定位方法。

```
. GO TOP                        指针定位到文件头
. ? RECN(),BOF()                查询记录号和文件头函数
        3 .F.                   文件的第一个记录是3号,文件
                                头是假
. SKIP -1                       指针向文件头方向移动
Record No. 3                    报告记录号仍是3,这是假的
. ? RECN(),BOF()                查询记录号和文件头函数
        3 .T.                   文件头函数成真值
```

实际指针已经超越了文件的第一个记录。指针的二意性再次体现。

将3号记录定义为文件的第一个记录是因为前两个记录被系统的状态所隐蔽。被隐蔽的文件不被计算在有效指针的范围内。

与上同理,请自行分析文件尾的指针移动情况:

```
. GO BOTT
. ? RECN(),EOF()
        9 .F.
. SKIP
```

```
Record No. 11
. ? RECN(),EOF()
          11 .T.
. LOCA FOR 姓名="刘"        在数据库中搜寻姓名为"刘"的记录
Record = 6                  报告6号记录满足条件
. CONT                      继续恢复查询
End of Locate scope.        没有发现满足条件的记录
```

实际上该记录的前后都有满足条件的记录，只是被系统状态隐蔽掉了。

```
. REPL 工资 WITH 工资+7   FOR &BH='02'
        2 replacements
```

以上是替换命令及报告。指定编号的特征字符是02的记录，将原工资增加7元。有两个记录被替换。另一个满足条件的记录被隐蔽。

```
. REPL 龄 WITH 0 ALL        清除所有活跃记录的年龄字段
     5 replacements         有5个记录被替换
. LIST                      验证执行结果，替换被实施
```

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
3	0133	一处	韩中民	男	4802	0	人事科	科长
5	0231	二处	李一同	男	6002	0	人事科	副科长
6	0223	二处	刘华珍	女	3510	0	劳资科	经济师
8	0121	一处	张　望	男	5906	0	劳资科	副科长
9	0134	一处	李志利	女	7002	0	人事科	干事

```
. REPL  龄 WITH 91- 出生/100 ALL   重新计算年龄
   5 replacements                  报告结果
```

```
. LIST                                          验证
Record#  编号  单位   姓名  性别 出生 龄   部门    职务
      3  0133  一处  韩中民  男  4802 43   人事科   科长
      5  0231  二处  李一同  男  6002 31   人事科   副科长
      6  0223  二处  刘华珍  女  3510 56   劳资科   经济师
      8  0121  一处  张  望  男  5906 32   劳资科   副科长
      9  0134  一处  李志利  女  7002 21   人事科   干事

. 1                                             指定 1 号记录
. RECA                                          恢复删除标记
    1 records recalled                          1 个记录被恢复
. DISP                          验证恢复结果,删除标记被取消
Record#  编号  单位   姓名  性别 出生 龄   部门    职务
      1  0311  三处  张林林  女  6902 22   财务科   助会

. RECA NEXT 3       在当前记录的情况下,限定恢复 3 个
                    记录
    1 records recalled
```

只有 2 号记录被恢复(1 号和 3 号记录无标记)。对其它命令,记录是隐蔽的,但对恢复命令是开放的,除了 GO 命令,这是可以绕过隐蔽状态的唯一命令。

```
. 1                         再次定位到 1 号记录
. LIST NEXT 3       验证恢复结果,记录被有效恢复
Record#  编号  单位   姓名  性别 出生 龄   部门    职务
      1  0311  三处  张林林  女  6902  0   财务科   助会
      2  0332  三处  张国琴  女  5906  0   人事科   干事
      3  0133  一处  韩中民  男  4802  43  人事科   科长
```

```
. RECA ALL                恢复所有做了删除标记的记录
  3 records recalled      有三个记录被恢复
. LIST                    验证恢复结果
```

Record#	编号	单位	姓名	性别	出生	龄	部门	职务
1	0311	三处	张林林	女	6902	23	财务科	助会
2	0332	三处	张国琴	女	5906	43	人事科	干事
3	0133	一处	韩中民	男	4802	43	人事科	科长
4	0323	三处	刘有才	男	5510	36	劳资科	科长
5	0231	二处	李一同	男	6002	31	人事科	副科长
6	0223	二处	刘华珍	女	3510	56	劳资科	经济师
7	0312	三处	李国珍	女	6902	22	财务科	助会
8	0121	一处	张　望	男	5906	32	劳资科	副科长
9	0134	一处	李志利	女	7002	21	人事科	干事
10	0212	二处	刘　伟	男	5410	37	财务科	副科长

17—6　在程序中变更数据库结构的演示

如何在程序中改变数据库的结构是许多程序员感到为难的问题，实际上 DBASE 具备这样的功能。在程序中不但可以改变一个数据库的结构，创建一个新的数据库也是完全可能的。由于早期的一些书籍中对此介绍很少，有的书中干脆就没有涉及，因此一些用户对此缺乏了解。

下面是一个在程序中改变数据库结构的过程演示。

```
. USE YUK                 打开数据库 YUK. DBF
. LIST STRU               显示数据库结构
Structure for database: C:\DB\YUK. DBF
Number of data records:      10
Date of last update : 08/02/91
```

Field	Field Name	Type	Width	Dec
1	编号	Character	4	
2	单位	Character	4	
3	姓名	Character	6	
4	性别	Character	2	
5	出生	Numeric	4	
6	龄	Numeric	2	
7	部门	Character	6	
8	职务	Character	6	
9	工资	Numeric	3	
10	学历	Character	4	

** Total ** 42

这是 YUK 数据库结构设置情况，它有 10 个字段。42 为一个记录占用的字节总数，它比各字段长度之和多一个字节，这一字节的空间用来标识记录的删除状态。

.COPY TO YU6 STRU EXTE

将当前数据库的结构生成一个结构数据库 YU6，它可将 YUK 中的库结构转换成结构数据库 YU6 中的记录，这就为改变数据库结构提供了最基础的条件。

.USE YU6　　打开结构数据库

.LIST STRU　　显示结构数据库的结构

Structure for database: C:\DB\YU6.DBF

Number of data records: 10

Date of last update : 08/02/91

Field	Field Name	Type	Width	Dec
1	FIELD—NAME	Character	10	
2	FIELD—TYPE	Character	1	

3	FIELD—LEN	Numeric	3
4	FIELD—DEC	Numeric	3

* * Total * * 18

它仅有四个字段，这四个字段的结构定义是系统规定的标准形式。在任何情况下，符合该结构的数据库都可生成新数据库的版本，前提是记录的内容符合字段定义的规范。

对这四个字段内容的要求和交互时定义字段的要求是完全相同的，它们的对应关系是

1	FIELD—NAME	字段名字
2	FIELD—TYPE	字段类型
3	FIELD—LEN	字段长度
4	FIELD—DEC	字段中的小数位

只要按照定义字段的规范在这些字段中填写数据，就能按这些数据的要求生成数据库结构。

请观察结构数据库中的记录内容：

.LIST

Record #	FIELD—NAME	FIELD—TYPE	FIELD—LEN	FIELD—DEC
1	编号	C	004	000
2	单位	C	004	000
3	姓名	C	006	000
4	性别	C	002	000
5	出生	N	004	000
6	龄	N	002	000
7	部门	C	006	000
8	职务	C	006	000
9	工资	N	003	000
10	学历	C	004	000

它分别与 YUK 的字段相对应。每条记录严格对应 YUK 中的一个字段，记录的顺序对应 YUK 的字段顺序。

改变数据库结构的操作是通过改变结构数据库中的记录实现的。

```
. DELE RECO 5              删除 5 号记录
      1 records deleted
. DELE RECO 6              删除 6 号记录
      1 records deleted
. DELE RECO 10             删除 10 号记录
      1 records deleted
. APPE BLAN                增加一条新记录
. REPL FIELD—NAME WITH '政面'
      1 replacements
. REPL FIELD—TYPE WITH 'C'
      1 replacements
. REPL FIELD—LEN WITH 4
      1 replacements
```

对增加的新记录赋值，这是与交互状态下定义数据库字段的操作等价。

```
. APPE BLAN                再增加一条记录
. REPL FIELD—NAME WITH '入党'
      1 replacements
. REPL FIELD—TYPE WITH 'N'
      1 replacements
. REPL FIELD—LEN WITH 4
      1 replacements
```

这里相当又定义了一个字段。

重新显示结构数据库内容：

```
. LIST
```

Record #	FIELD—NAME	FIELD—TYPE	FIELD--LEN	FIELD—DEC
1	编号	C	004	000
2	单位	C	004	000
3	姓名	C	006	000
4	性别	C	002	000
5	* 出生	N	004	000
6	* 龄	N	002	000
7	部门	C	006	000
8	职务	C	006	000
9	工资	N	003	000
10	* 学历	C	004	000
11	政面	C	004	000
12	入党	N	004	000

对记录的操作有效。

下面处理删除的记录：

.PACK

9 records copied

再次显示记录。

.LIST

Record #	FIELD—NAME	FIELD—TYPE	FIELD—LEN	FIELD—DEC
1	编号	C	004	000
2	单位	C	004	000
3	姓名	C	006	000
4	性别	C	002	000
5	部门	C	006	000
6	职务	C	006	000
7	工资	N	003	000

8	政面	C	004	000
9	入党	N	004	000

如果认为这是一个满足要求的数据库结构，可用下面的命令完成创建新数据库的操作。

.CREA YU7 FROM YU6

创建新数据库 YU7 并指定创建的结构存在 YU6 的记录中。新创建的数据库是当前数据库。

.LIST STRU

Structure for database: C:\DB\YU7.DBF

Number of data records: 0

Date of last update : 08/02/91

Field	Field Name	Type	Width	Dec
1	编号	Character	4	
2	单位	Character	4	
3	姓名	Character	6	
4	性别	Character	2	
5	部门	Character	6	
6	职务	Character	6	
7	工资	Numeric	3	
8	政面	Character	4	
9	入党	Numeric	4	
** Total **			40	

显示结果可以证实新创建的数据库满足原来定义的要求。

用 APPE 命令可从原数据库中抄录数据：

.APPE FROM YUK

10 records added

.LIST OFF

编号	单位	姓名	性别	部门	职务	工资	政面	入党
0311	三处	张林林	女	财务科	助会	90		
0332	三处	张国琴	女	人事科	干事	125		
0133	一处	韩中民	男	人事科	科长	139		
0323	三处	刘有才	男	劳资科	科长	125		
0231	二处	李一同	男	人事科	副科长	111		
0223	二处	刘华珍	女	劳资科	经济师	154		
0312	三处	李国珍	女	财务科	助会	90		
0121	一处	张　望	男	劳资科	副科长	125		
0134	一处	李志利	女	人事科	干事	90		
0212	二处	刘　伟	男	财务科	副科长	125		

与 YUK 比较取消了出生、龄和学历三个字段，增加了政面和入党两个字段。

进一步的操作是向新增加的字段输入内容。

.GO 1

.CHANGE FIEL 姓名，政面 ，入党

这时可在屏幕的提示下逐个输入每个人的数据。新数据录入完成后可再次显示。

.LIST 编号，单位，姓名，政面，入党

Record#	编号	单位	姓名	政面	入党
1	0311	三处	张林林	党员	8203
2	0332	三处	张国琴		
3	0133	一处	韩中民	党员	8506
4	0323	三处	刘有才	党员	8807
5	0231	二处	李一同		
6	0223	二处	刘华珍	党员	8109
7	0312	三处	李国珍		

8 0121 一处 张 望
9 0134 一处 李志利 党员 9001
10 0212 二处 刘 伟

这是一个完整的演示，各环节的数据都是系统的工作记录，依照这个演示过程，在程序中处理类似的问题是不难办到的。

17—7 示范程序

从交互应用过渡到程序不是简单的将命令罗列在一起，程序设计要有一定的整体构思。尤其在程序的控制、引导等方面要讲究技巧性。

下面是一个示范程序，它虽然很简单，但其中体现了程序设计的一些基本的要求。如果单纯的为了实用，这并不一定是好的方案。之所以设计成这种形式，为的是要突出程序的模块化设计及组织。

对示范程序的理解不要从实现功能的角度出发，重要的是理解实现这些功能的各个子程序是怎样相互协调地完成任务的，这是学习程序设计的基本要领。

再有，要观察变量在程序中是怎样使用的，在处理随机问题和简化编程等方面，变量的应用技巧是很值得研究的。

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* * *                    示范程序                              * * *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

                  * * * * *  主  程  序  * * * * *
CLEA                                  && 清屏
CLEA ALL                              && 关闭所有文件，初始化
SET PROC TO JC.DB5                    && 打开过程文件
```

```
SET TALK OFF                    && 关闭执行结果显示报
                                   告
SET DELE ON                     && 隐蔽带删除标记的记
                                   录
SET BELL OFF                    && 关闭变量填满的鸣铃
SET SAFE OFF                    && 覆盖文件不提示
          * * 该组命令是初始环境设置
GNXZ='功能选择'                 && 建立公用变量
CW='错误!'                      && 建立公用变量
BBHH='  '                       && 建立公用变量
USE YUK INDE YUK                && 打开数据库和索引文
                                   件
UIKV='  '
DO WHIL .T.                     && 开始一个循环结构
IXKV='  '                       && 初始控制变量
DO PMXU                         && 调用 PMXU 子程序
CLEA                            && 清屏
DO CASE                         && 开始一个多分支结构
CASE IXKV='  '                  && IXKV 为字符空格
EXIT                            && 跳出循环结构
CASE IXKV $'12345678'           && 放行字符 1 —8
OTHE                            && IXKV 的其它所有值
DO CW WITH GNXZ+CW              && 显示功能选择错误
LOOP                            && 返回到循环结构的开
                                   始
ENDC                            && 结束多分支结构
VXIX='IX'+IXKV                  && 合成一个字符串
DO &VXIX                        && 执行字符串表示的程
                                   序
```

```
CLEA                                    && 清屏
ENDD                                    && 结束循环结构
RETU                                    && 返回到系统监测
```

本程序是显示菜单、接受屏幕输入的控制字符并鉴定字符的合法性，合法的选择必有一个可执行的程序与之对应，否则鸣铃告警并要求重新输入控制字符。

```
*****  CW  子程序  *****
PROCEDURE CW
* 错误提示
PARA TU                                 && 接收提示字符串
@ 24,0                                  && 清除屏幕的第 24 行
?? CHR(7)                               && 鸣铃
@ 24,40-LEN(TU)/2  GET TU               && 计算屏幕位置并显示
                                           提示,GET 是使显示反相
CLEA GETS                               && 释放 GET 的变量
RETU                                    && 返回调用程序
* 根据提示符串的长度自动将字符串显示在屏幕的中部。
```

```
*****  JLIX  子程序  *****
PROCEDURE JLIX
* 记录查询
DO WHIL .T.                             && 建立循环控制
@ 9,0                                   && 清除屏幕的第 09 行
@ 9,32 SAY "编号" GET BBHH              && 接收编号
READ
IF BBHH='  '                            && 如果变量 BBHH 为空
EXIT                                    && 退出循环
ENDI                                    && 结束 IF
SEEK BBHH                               && 在数据库中检索
```

```
IF EOF()                          && 如果到达文件尾
?? CHR(7)                         && 鸣铃
DO CW WITH '记录不存在'           && 显示提示
LOOP                              && 返回控制头
ENDI                              && 结束 IF 结构
EXIT                              && 定位成功,退出控制
ENDD                              && 结束循环
RETU                              && 返回调用程序
* 本程序对存在的记录定位返回。
* 编号输入空格可中断查询返回。
* 对不存在的记录显示错误提示并封锁。

        * * * * *  PMXU 子程序  * * * * *
PROCEDURE PMXU
PL=20
@ 1,PL SAY '                    演  示  程  序'
@ 2,PL SAY '                                                    '
@ 3,PL SAY '    1. 增加记录          5. 列表显示                '
@ 4,PL SAY '    2. 删除记录          6. 列表打印                '
@ 5,PL SAY '    3. 显示修改          7. 全屏编辑                '
@ 6,PL SAY '    4. 打印卡片          8. 扩充功能                '
@ 7,PL SAY '                                                    '
@ 9,32 SAY GNXZ GET IXKV PICT '!'
                                  && 对 IXKV 赋值并限定
                                     大写
READ                              && 激活 IXKV 变量
RETU
* 这是菜单显示程序,变量 PL 的定义可使屏幕调整方便。

        * * * * *  JLXU 子程序  * * * * *
```

```
PROCEDURE JLXU
PI=20                                    && 定义屏幕左边界
@ 0,PI+13 SAY " 个人记录显示"
@ 1,PI SAY ;
"┌──────────────────────────────┐"
@ 6,PI SAY ;
"└──────────────────────────────┘"
@ 2,PI SAY "|  编号" GET 编号
@ 3,PI SAY "|  单位" GET 单位
@ 4,PI SAY "|  姓名" GET 姓名
@ 5,PI SAY "|  性别" GET 性别
PI=36                                    && 调整屏幕坐标
@ 2,PI SAY "出生" GET 出生
@ 3,PI SAY "龄" GET 龄
@ 4,PI SAY "部门" GET 部门
@ 5,PI SAY "职务" GET 职务
PI=48                                    && 调整屏幕坐标
@ 2,PI SAY "工资" GET 工资
@ 3,PI SAY "学历" GET 学历
PI=58                                    && 调整屏幕坐标
@ 2,PI SAY"|"
@ 3,PI SAY"|"
@ 4,PI SAY"|"
@ 5,PI SAY"|"
RETU                                     && 返回调用程序

        * * * * *  IX1  子程序  * * * * *
PROCEDURE IX1
* 增加记录
COPY STRU TO GZK                         && 准备工作库结构
```

```
SELE 2                          && 转向二工作区
USE GZK                         && 打开工作库
APPE BLAN                       && 增加一条记录
DO WHIL .T.                     && 建立循环控制
SELE 2                          && 转向二工作区
DO JLXU                         && 调用记录显示程序
READ                            && 激活键盘操作
IF 编号=' '                     && 判断编号是否为空
USE                             && 成立,将关闭二区
EXIT                            && 退出循环
ENDI                            && 结束判断
COPY TO GZK SDF                 && 拷贝记录成文本数据
BBHH=编号
SELE 1                          && 转向一工作区
SEEK BBHH                       && 检查编号是否重复
IF .NOT.EOF()                   && 不重复
DO CW WITH '编号重复'           && 显示错误提示
ELSE                            && 重复
APPE FROM GZK SDF               && 将文本记录录入
ENDI                            && 结束判断
ENDD                            && 结束循环控制
SELE 1                          && 转向一工作区
RETU                            && 返回调用程序
```

* * * * * IX2 子程序 * * * * *

```
PROCEDURE IX2
* 删除记录
KK=' '                          && 将空格送到 KK 变量
DO WHIL .T.                     && 建立循环控制
DO JLIX                         && 调用记录查询程序
```

```
IF 编号=''.                          && 判断编号是否为空
EXIT                                 && 退出循环
ENDI                                 && 结束判断
@ 9,18 SAY 单位+KK+部门+KK+姓名+' 是否删
除 Y/N:' GET UIKV                    && 准备接收变量
READ                                 && 激活键盘操作
IF UIKV='Y'                          && 判断操作员意图
DELE                                 && 做删除标识
ENDI                                 && 结束判断
ENDD                                 && 结束循环
RETU                                 && 返回调用程序

        * * * * *  IX3 子程序  * * * * *
PROCEDURE IX3
* 显示修改
DO WHIL .T.                          && 建立循环控制
DO JLIX                              && 调用记录查询程序
IF BBHH='  '                         && 判断编号是否为空
EXIT                                 && 退出循环
ENDI                                 && 结束判断
DO JLXU                              && 调用记录显示子程序
CLEA GETS                            && 放弃键盘操作
@ 9,32 SAY '是否修改 Y/N:' GET UIKV
READ                                 && 激活键盘操作
IF UIKV='Y'                          && 判断操作员意图
DO JLXU                              && 再次调用记录显示
READ                                 && 激活键盘操作
ENDI                                 && 结束判断
ENDD                                 && 结束循环
RETU                                 && 返回调用程序
```

```
*   本程序两次调用记录显示,目的不同,第一次是显示,
    第二次是修改,在不要求修改时仅完成显示。

        * * * * *  IX4 子程序   * * * * *
PROCEDURE IX4
* 打印卡片
DO WHIL .T.                          && 建立循环控制
DO JLIX                              && 调用记录查询程序
IF BBHH=''                           && 判断编号是否为空
EXIT                                 && 退出循环
ENDI                                 && 结束判断
SET CONS OFF                         && 屏幕封锁,提高打印
                                        速度
SET PRIN ON                          && 设置打印机开态
?"                  个人记录卡片"
?"┌──────────────────────────────────┐"
?" | 编号",编号," 出生"
?? STR(出生,5)," 工资",STR(工资,4),"|"
?" | 单位",单位," 年龄",STR(龄,4)
??" 学历",学历,"|"
?" | 姓名",姓名,"部门",部门
??"           |"
?" | 性别",性,"职务",职务
??"           |"
?" └──────────────────────────────────┘"
?                                    && 以上送数据到打印机
SET CONS ON                          && 恢复屏幕显示
SET PRIN OFF                         && 关闭打印机
ENDD                                 && 结束循环
RETU                                 && 返回调用程序
```

```
*****  IX5 子程序  *****
PROCEDURE IX5
* 列表显示
DO WHIL .T.                          && 开始一个控制结构
LIST                                 && 列表到屏幕
?                                    && 显示空行隔离
WAIT SPAC(26)+' 回车 返回,其它重新显示' TO MM
IF LEN(MM)=0                         && 检测变量是否赋值
EXIT                                 && 退出控制
ENDI                                 && 结束判断
ENDD                                 && 结束控制结构
RETU                                 && 返回调用程序
```

* 暂停命令可创建一个一字节的字符变量并可接收不可打印字符。

```
*****  IX6 子程序  *****
PROCEDURE IX6
* 列表打印
SET CONS OFF                         && 关闭屏幕显示
LIST TO PRIN                         && 列表到打印机
SET CONS ON                          && 打开屏幕显示
RETU                                 && 返回调用程序
```

```
*****  IX7 子程序  *****
PROCEDURE IX7
*全屏编辑
BROW                                 && 进入全屏幕编辑
RETU                                 && 返回调用程序
```

```
*****  IX8 子程序  *****
PROCEDURE IX8
```

```
* 扩充功能
PL=20
@ 1,PL SAY '                    [演  示  程  序]'
@ 2,PL SAY '  ┌──────────────────────────────┐  '
@ 3,PL SAY '  │1. 扩充功能－1    5. 扩充功能－5│  '
@ 4,PL SAY '  │2. 扩充功能－2    6. 扩充功能－6│  '
@ 5,PL SAY '  │3. 扩充功能－3    7. 扩充功能－7│  '
@ 6,PL SAY '  │4. 扩充功能－4    8. 扩充功能－8│  '
@ 7,PL SAY '  └──────────────────────────────┘  '
@ 9,32 SAY GNXZ GET IXKV PICT '!'
                                  && 对 IXKV 赋值并限定
                                     大写
READ                              && 激活 IXKV 变量
@ 9,0
ZIFU=VAL(IXKV)                    && 转换变量类型
IF ZIFU>=1. AND. ZIFU<=8          && 判断输入的合法性
ZIFU= SUBS('一二三四五'+;'六七八',ZIFU * 2-1,2)
@ 9,14 SAY '你已选定扩展'+;' 功能第'+ZIFU+'项，因无
程'+;'序按任意键返回' GET IXKV
                                  && 屏幕等待
READ                              && 此处为使屏幕等待
ELSE                              && 非法输入
DO CW WITH GNXE+CW                && 显示错误提示
ENDI                              && 结束判断
RETU                              && 返回调用程序
```

* 本段侧重领会字符的截取及向数值的转换。

* 变量 PL 的变值使屏幕得到调整。

第十八章　自然码编辑软件使用说明

ZRED 是自然码系统中的一个全屏幕中英文文字编辑软件，具有命令简单、操作灵活、占用空间少、快速等优点，特别适合程序员和初学者使用。

ZRED 编辑软件可提供文件的装入、写盘、打印、排版等工作。在编辑状态下则可对正文进行输入、整理复制、插入、删除、查找、替换等操作，可以对块标志作插入、删除、移动、拷贝，还可以将块标志写入磁盘文件或将磁盘文件读入到当前光标处。

ZRED 编辑软件能够编辑的最大文件允许 64K，超过限制的文件将被截断。

18—1　编辑状态下的命令

1. 光标部分

(1)	光标左移一个字符	←
(2)	光标右移一个字符	→
(3)	光标左移一个词	Ctrl — ←
(4)	光标右移一个词	Ctrl — →
(5)	光标上移一行	↑
(6)	光标下移一行	↓
(7)	向上翻页	Pg Up
(8)	向下翻页	Pg Dn
(9)	光标至行首	Home
(10)	光标至行尾	End
(11)	光标至页首	Ctrl — Pg Up

(12) 光标至页尾　　Ctrl — Pg Dn
(13) 光标至文件顶　　Ctrl — Home
(14) 光标至文件尾　　Ctrl — End
(15) 光标至块起始　　Alt — Y
(16) 光标至块结尾　　Alt — E
(17) 回到上次的光标位置　　F10

2. 插入与删除部分

(1) 插入/替换开关　　Ins
(2) 插入一行　　a—s 或 [Retrn]
(3) 删除一行　　Ctrl—Y
(4) 删除至行末　　F6
(5) 删除当前词　　Ctrl—W
(6) 删除当前字符　　Del 或 Ctrl—G
(7) 删除当前左边的字符　　Back Space 或 Ctrl—H
(8) 恢复当前行　　F5

3. 标志部分

(1) 置标志起始位置　　F7 或 Alt—B 或 Ctrl—KB
(2) 置标志结尾位置　　F8 或 Alt—N 或 Ctrl—KK
(3) 当前词置为标志　　Alt — C （英文）
(4) 隐含/显示标志　　Alt — U 或 Ctrl — KH
(5) 拷贝或插入标志　　Alt — Z 或 Ctrl — KC
(6) 移动标志　　Alt — M 或 Ctrl — KV
(7) 删除标志　　Alt — D 或 Ctrl — KY
(8) 从磁盘中读文件　　Alt — L
(9) 写标志到磁盘中　　Alt — W

4. 查找与替代部分

(1) 查找字符　　Alt — F 或 Ctrl — QF
（方式：B，G，U，数字）
(2) 查找并替代字符　　Alt — R 或 Ctrl—QA

（方式：B，G，U，数字）

(3) 继续查找或替代　　Ctrl － L

(4) 输入特殊控制字符　　Ctrl － P ＋ 控制键或字符

其中查找和替代字符时的“方式”与 WordStar 一样：B——逆向，G——全文范围，数字——次数，U——大小写不分，N——自动替代。

5. 其它

(1) 文件存盘　　F1

(2) 结束编辑并退到命令态　　F2　或　Ctrl － Z

(3) 另起文件名存盘　　F3

(4) 改变右边界　　F4

(5) 从当前位置排版　　Ctrl － B

(6) 全中文输入方式　　F9（默认为中英文方式）

18—2　命令状态下的命令

(1) 进入编辑　　D

(2) 文件存盘　　S

(3) 打印文件　　P

(4) 装入新文件　　L

(5) 更改当前文件名　　N

(6) 显示磁盘文件名　　F

(7) 设置文件宽度（汉字）O　　（隐含命令）
（1 个汉字 ＝ 2 个英文字符）

(8) 全文排版　　B　　（隐含命令）

(9) 结束编辑返回 DOS　　X

使用打印文件命令时，其中屏幕将预先显示出后面将要打印的几行。操作者可用空格键设置“逐行打印”方式，以便于控制

分页位；用回车键可恢复连续打印方式，用<ESC> 键可中止打印。

18—3 编辑实例

A>ZRED ZRM. DOC

```
行 8      列 7      替换态          A：ZRM. DOC

计算机已经成为各个管理部门不可缺少的重要工具，越来越多的文件和资料要求使用计算机管理。

jsj yijyigwzgegegl'bumfb k qtukd vsykgsju，ylydd wfjmh zilcykqquiys；jsj gl'
zilcxuykys jsj ll iuli.
```

<F2>

? S 存盘 \ZR. DOC

? X

ZRED 常用操作键表：

恢复当前行	F5
删除至行末	F6
全中文方式	F9
回到上次操作的光标位置	F10
至行首	Home
至行尾	End
至文件顶	Ctrl — Home
至文件尾	Ctrl — End

删除一行	Ctrl — Y
串查找	Alt — F
串替代	Alt — R
在编辑态下存盘	F1
在编辑态下换名存盘	F3
结束编辑并退到命令态	F2
在命令态下存盘	S
在命令态下打印	P

第十九章　WS 汉字文字编辑软件

当前微机汉字编辑软件很多，WORDSTAR 编辑软件（简称 WS）就是目前常用的、较优秀的汉字编辑软件。现就 WS 的使用方法及应用技巧介绍如下。

19—1　WS 汉字编辑软件简介

使用 WS 汉字文字编辑软件必须有 WS. COM、WSMSGS. OVR、WSOVLY1. OVR 三个文件驻留在磁盘上。WSMSGS. OVR 文件提供提示信息及命令选择清单。

WS 有四种命令类型：单键命令、双键命令、三键命令、圆点命令。

19—2　启动汉字编辑软件

若汉字操作系统已经装入，你就可以将带有 WORDSTAR 软件的软盘插入 A 或 B 软盘驱动器中，在操作系统提示符 A＞或 B＞下键入：WS↙。若该软件在 C 盘中，则应在 C＞下键入：WS↙。按回车键后，屏幕即刻显示《起始命令》表，表示“汉字文字编辑”软件已经装入。此时屏幕显示如下：

《起　始　命　令》

D　进入编辑	E　更换文件名
P　打印文件	O　拷贝文件
R　运行程序	Y　删除文件
N　编辑非文书文件	X　退出

你可根据自己的需要选择 D～X 的任一操作，便可执行相应的功能。下面分别介绍各命令的功能和操作方法。

19—3 进入文字编辑

1. 进入编辑 当你要编辑一个文书文件时，如起草文件、书信等，按“D”后，可进入编辑状态，并且屏幕立即显示如下说明和提示信息：

使用本命令建立新文书文件或更新现存文件。文件名前是一个驱动器符及冒号，如省略则隐含当前驱动器。文件名是：

文件名字？—

这时你就可以输入被编辑文件的文件名。文件名由三部分组成，其格式为：[<驱动器：>] <文件名> [<扩展名>]。需要说明的是：“WS”的文件名前一般不能加“路径”。文件名可为“中文”，也可为“西文”；长度：1—8 个字符，扩展名为 3 个字符，超过规定长度，超过部分自动删除，不会出现文件名错。但是，如果文件名中带有扩展名，则必须输入扩展名，否则 WS 认为编辑新文件或其他文件。

例： 所要编辑的文件名叫“WS. HLP”在 A 驱动器中，当屏幕出现：

文件名字？—

输入：

A：WS. HLP↙

如果 A 盘中已经存在文件“WS. HLP”，则屏幕显示该文件内容，否则出现提示信息：“新文件”，然后屏幕显示下列信息：

A：WS. HLP 页号 1 行号 1 列号 01 INSERT ON

L ! ! ! ! ! !·····! ! ! ! R

显示的第一行叫做状态行，它告诉你：

(1) 当前编辑着的文件名字；

(2) 光标所在页的页号；

(3) 光标所在行的行号；

(4) 光标所在列的列数；

(5) 是在插入状态下 (INSERT ON)。

下面一行是尺度行，它告诉你空白区和标志，并控制了每行可输入的字符数。尺度行下面的空白，即为文字编辑显示区，就象一张“空白纸”，等待使用者尽情书写自己的作品。

2. 书写文章 当前进入编辑状态后，使用者就可以在这张“空白纸”上书写自己的文章，从左往右，每写一个字，光标就向右移动，在新的位置等待你写下一个字。WS 的换行是根据设置的行宽自动进行，而不需要人为干预，写满一行后，光标就会自动跳到下一行的开头。当一个自然段的内容输入完毕，按一次回车键，表示形成了一个自然段，光标换到新的一行，继续写文章的下一段。屏幕写满以后所有显示行自动上滚，出现一行新的空白行。

按回车键为手动换行，它会在手动换行的行尾显示手动换行符“<”，有人也称它为回车符号。而自动换行的显示符为空。在你进行编辑时，屏幕还可能显示自动换页符“·····P”(一

页的行数写满后自动跳到下一页）。

需要注意的是：有些人在实际应用中，不注意根据一行字的多少来设置行宽，往往一行还没写满，就按回车键，这样“WS”认为每一行即为一自然段。当一篇文章写完后，需要重新调整行宽时，就会感到非常麻烦，因为“WS”的排版，每排一次只能对光标所在的一个自然段，按照尺度行的宽度排版。

在书写和编辑中，常会发生一些意外（例如，编辑当中断电）造成文章内容丢失。为避免意外的发生，在书写中应注意随时存盘，写一点存一点。操作为：^ KS（用左手按下CTRL键，右手按K，当屏幕左上角出现^ K后，再按S）

19—4 编　　辑

1. 移动光标　光标是用来提示你当前操作的字符位置。你可以操纵光标，使它能够在任意方向移动，以实施输入、修改、删字、加字、复制和位移。

控制光标移动的方法有三种：

（1）使用键盘右侧的小键盘（或叫数字键）。注意这组键有双重功能，一个是用来控制光标移动，另一个是用来输入数字，使用时应注意其状态。

小键盘：

NUM
LOCK

7	8	9
4	5	6
1	2	3

在光标状态下：

按8（↑），向上移一行；

按 2（↓），向下移一行；

按 4（←），向左移一 ASCII 字符位置；

按 6（→），向右移一 ASCII 字符位置。

注：如按了这组其中的某键而光标未移动却出现了数字，说明小键盘处于数字键状态，按一下小键盘上端的 NUM-LOCK 键，置成光标状态。

如果你的键盘有专用光标键，最好使用光标键。

光标键：

(2) 控制键。操作：用左手按下 CTRL 键，右手按 W，E，…X，C。

按 ^ E（=↑），光标上移一行；

按 ^ X（=↓），光标下移一行；

按 ^ S（=←），光标左移一 ASCII 字符；

按 ^ D（=→），光标右移一 ASCII 字符；

按 ^ R（=PgUp），光标移到显示页的前面一页；

按 ^ C（=Pgan），光标移到显示页的后面一页；

按 ^ W，光标不动而整个屏幕向下移一行；

按 ^ Z，光标不动而整个屏幕向上移一行；

按 ^ A，光标左移一句（或一个英文单词）；

按 ^ F，光标右移一句（或一个英文单词）。

(3) 光标速移“Q”命令。在书写或修改文章时，常常需要将光标移至其他位置。例如，一篇文章很长，需要从文章的开头移到文章的结尾，又从结尾跳到开头。如果用简单的光标移动，很不方便。“Q”命令可将光标快速移到文本中某一指定位置。

操作：在执行“Q”命令表下的子命令之前要按 ^ Q 进入命令，当屏幕左上角出现“^ Q”后，再按如下键：

按 S，光标移到当前行的左端；

按 D，光标移到当前行的右端；

按 E，光标移到屏幕顶端；

按 X，光标移到屏幕底端；

按 R，光标移到文章的开始　（或功能键 F9＝^ QR）；

按 C，光标移到文章的结尾　（或功能键 F10＝^ QC）；

按 B，光标移到字块之首；

按 K，光标移到字块之尾；

按 Z，上滚一行；

按 W，下滚一行；

按 P，回到原光标位置；

按 V，光标复置于最后一次查询处；

按 0－9，将光标移到标记处（设置标记下面介绍）。

2. 插入与删除　“插入”是指在光标所在位置的前面插入一个字符或文字。“删除”是删除光标所在位置或左边的字符及文字。如果删除和插入的是汉字，一定要把光标移到汉字的前半部分。因为光标只能盖住半个汉字，所以应注意分清是前半部分还是后半部分，特别是一些两字节的符号。如果只删后半部分会造成整句的错误。例如，句号与下一个字之间，从屏幕上看总空一个字符的位置，如果将光标移到该位，不管是删除或插入都将造成句子错误。

（1）加字。加字是在字和字之间插入文字，这可以是一个字，也可以是一段话。插入字前要看一下状态行上是否有“INSERT ON”出现。如果有，说明现在处于插入状态。否则，新加入的字会把原有的字覆盖了。

如果“INSERT ON”没有出现，按 ^ V 或在光标状态下按小键盘的“INS”键，便可进入插入状态。

（2）插入空白行。按 ^ N 在当前光标所在行的前面增加一行空白行。

（3）删字。删字有两种方法。一种是将光标移到要删的字的前

半部分，按两次 ^ G，才能完整地删除一个汉字。^ G 是删当前光标下的字符。

另一种删字方法是在光标状态下按小键盘的“DEL”键。不同的是这种方法删去的是光标左边的字符。用这种方法删字时，同样要按两次“DEL”键才能完整地删除一个汉字。

(4) 删一行。当需要删除一整行时，先把光标移到该行的任意位，再按 ^ Y 删除光标所在的行。

(5) 删除一句。按 ^ T 删除光标右边的一句。

(6) 使用“Q”命令。按“^ Q”后：

按 Y 删除光标所在行的右边的全部字符。

按 DEL 键删除光标所在行的左边的全部字符。

(7) 插入换行符。将某一行分成两行，其操作是：移动光标至分行点，在插入状态按“回车”键即在光标处插了一个换行符“<”，这时原来的一行分成两行，光标跳至新行的开头。

(8) 删除换行符。假设有一段话共两行，每行后面都有换行符“<”，现在需要将第一行的换行符去掉，操作如下：

先将光标移到第二行的开头，然后按“DEL”键，使两行合为一行，最后按“^ B”重新排版（排版后面讲）。

3. 修改字符 修改字符时必须注意当前编辑是否处于“插入”状态，若屏幕右上角出现“INSERT ON”，则应按“^ V”或“INS”键，取消插入状态方可进行修改。

修改时将光标移到要修改的字符位置上，如果修改汉字，必须将光标移到汉字的前半部，然后输入新的字符或汉字。

修改完后，应恢复插入状态。

19—5 编辑技巧

1. 屏幕设定 书写文章时，每个使用者可能对字距、行距、行宽、页宽及输出字型的要求各不相同，因此，编辑或调整时，就

要对屏幕进行必要的设定。下面即为主要屏幕设定的操作：

(1) 设置行宽

^OL　设置左边空格数。这就像我们在一张白纸上写文章一样，左边总要空出几列。这个功能实际用的不多。

^OR　设置右边空格数。该功能实际就是限制屏幕显示或输出打印每行的字数，即行宽。当你要变换行宽时，按^OR 或按 F4，屏幕左上角会出现：

右边空格数#

这时输入你所要的行宽列数，然后按回车键，这里要记住：一个汉字占两个字符。例如，行宽 20 个汉字，应输入 40。

为使得编辑排版容易，我们建议使用偶数行宽，并尽量保持一行内的半角字符数为偶数。

设完行宽后，尺度行的“标尺”相应地变成你所设置的长度，之后你就可以输入或排版。

(2) 设置行距

^OS　设置行距。当按下^OS 后，屏幕左上方会出现：

输入空格或新的行距 (1～9)

若输入 1，则行与行之间没有空白行。即写满一行后接着写第二行。

若输入 2，则每行之间有一空白行，即写满一行后自动跳到第三行。

以此类推。行距一经设置后就不会改变，直到重新设置新的行距为止。

默任值为 1。

(3) 其它屏幕命令

^OC　不管原来字符在一行上的什么位置，按^OC 后使一行上的字符处于屏幕的中间位置。适用于设置文章的标题。

^OF　使尺度行与一行中字符的头尾对齐，即尺度行的长

度与一行的长度相同且位置对齐。

^ OT　清除/设置尺度行。如果屏幕上已有尺度行，按^ OT 则消除尺度行，再按一次^ OT 则又恢复尺度行。

^ OP　清除/设置分页标志的显示。分页标志前面已提到，即一行虚线。按^ OP 可将虚线消除，再按一次又可恢复。

^ OD　清除/设置打印控制字符的显示。文章输出打印时，往往需要在文章内设置打印控制字符，如：^ D；^ E 等。按^ OD 可将这些符号消除，再按一次又可恢复它们。

2. 字块操作

(1) 文本内字块操作。所谓“字块”即一组连续的字符串、句子、一个或数个连续的自然段落。在编辑一篇文章时，常常会遇到需要将某一字块前后移动的情况，或将某一字块复制到其它内容相同或相近的位置，或删除某一字块，这些操作即为字块操作。字块操作是非常有用的一种功能，它大大提高了编辑的灵活性，即减少了书写量，也提高了速度。

在进行字块操作时首先按^ K，当屏幕左上方出现“^ K”后再按其他字母。

① 定义字块。定义字块是指定需要进行操作的字块，标明字块的开头和结尾。

^ KB　(=F7) 设置字块的首标志。将光标移到字块的第一个字符位置按^ KB (或 F7)，则光标前出现“<B>”。

^ KK　(=F8) 设置字块的尾标志。将光标移到字块的最后一个字符后面按^ KK (或 F8)，则光标前出现“<K>”。

② 字块移动。在编辑中“WS”允许你将文章的某一区段由一

个位置搬到另一位置。

^KV　把首尾标所定义的字块移到当前光标处。首先，将光标移到所需的位置，然后按^KV，则字块（包括<B>和<K>）移到光标所在的位置，而原来位置上的字块被删除了。现在光标在移动后的字块的第一个字符位上。

③ 字块复制。如果你的文章中，某一段同时存在文章内多处，“WS”允许将其复制到需要的位置。

^KC　把首尾定义的字块拷贝到当前光标处。首先，将光标移到所需的位置，然后按^KC，则字块（包括<B>和<K>）拷贝到光标所在的位置，而原来位置上的字块保持不变，但首尾标志“<B><K>”被删除。现在光标位于被拷贝字块的第一个字符位上。

④ 取消字块标志。如果不需要再进行字块操作，可取消定义标志。注意，这里所说的“取消”并不是取消字块中的字符，而是取消定义字块的首尾标志。

^KH　只删除首尾标志<B>和<K>。不管是^KV还是^KC操作后，光标所在的字块首尾标志仍然存在，只有进行了^KH操作首尾标志才被删除。

⑤ 字块删除。编辑中，若删除一段文字，可进行字块删除操作。

^KY　删除首尾标志<B>和<K>及其所定义的字块。如果要将文章某段内容删除掉，应先对该段内容进行字块定义，然后按^KY即被删除。

这里应注意几点：

第一，定义首尾标志的时间顺序不受限制，先定首后定尾或先定尾后定首均可。但是，首尾标志的位置是有限制的，必须首标在前，尾标在后，否则将出错。

第二，移动、拷贝和删除字块之前必须先定义好字块。

第三，注意汉字及全角字符的完整性。定义字块时不要定义半个汉字，否则将出错。

(2) 文件之间的字块操作。有时写文章需要将另一文件的某段内容加到本文中的某一位置，或两篇文章合并，都需要文件间的字块操作。例如：有的文章中常出现一些表格，而“WS”的制表功能较弱，表格线又不受保护，制表或修改表中内容时就非常麻烦。如果先用制表功能较强的汉字软件 CCED 制表，然后用文件间的字块操作将表插到文件中，就比较方便了。当然你也可以将表格移出去，用 CCED 修改后，再移回来。也许有人会问，用 CCED 编辑文章不是更方便吗？不错。但是，如果你的文章较大，超过了 64K，超过部分就可能丢失，而“WS”就不受限制。这就是取长补短的“妙用”。

文件之间的字块操作是通过一个临时文件完成的。假设我们要将文件 1 中的某些内容写到文件 2 中的某一位置，其操作如下：

在“文件 1”的编辑状态下找到所需的字块，设制首尾标志<B>和<K>，定义好字块，然后按：

^ KW

屏幕提问：

文件名?

这时你可取一个临时文件名，例如 L，然后按“回车”键。这样文件 1 中被定义了的字块就被写到了临时文件 L 中。然后退出编辑。

回到《起始命令》表后，再按 D 进入你所要编辑的“文件 2”，并将光标移到文章中要写字块的位置上，按：

^ KR

屏幕出现：

文件名?

这时输入刚才存的临时文件的名字“L”，然后按“回车”键。

这样，临时文件“L”中的内容就被读到“文件2”中当前光标位置上了。

3. 查找字符串 “WS”允许你进行找字操作，搜寻文章中的某一字符串，这里的字符串是指一组连续的汉字或字符，也可是单个的汉字或字符。“WS”规定查找的字符串最长为30个连续字符（半角字符），如果是汉字，即为15个汉字。

在一篇长文章中，查找一个字符串往往很麻烦，用光标移动的方法一行一行地人工查找文章中多次出现的字符或某人的姓名，既费时又费事，还可能漏掉。如果你用“WS”的字符串查找功能就非常方便，有如“踏破铁鞋无觅处，得来全不费功夫”之感觉。

(1) 设置标记。为了查找方便，在编辑文章时，可以预先在重点段落设置标记，在一篇文章中最多可设置10个标记，即数字0～9。其操作如下：

① 设标操作

^K+（0～9）

先将光标移到准备设置标记的字符位上，按^K，再按0～9中的任一数字，这样就会在光标处显示出“<N>”，这里的N为0～9中的任一个数字，这10个标记可以任意顺序设置在文章中的任意位置。文章中出现的标记“<N>”，打印时不会被打印。

例如：按CTRL+K（^K），放开这两键后再按1，这样“<1>”就会出现在文章中。

② 找标操作

^Q+（0～9）

为了找到标记“<N>”(N=0～9)，按^Q，再按0～9中的任一数字，这样光标就会回到“<N>”处。如果预先没有设置标记，而进行上面的操作将会出错。

例如：按^Q，再按1，光标就会回到<1>处。

③ 删除标记操作

^ K+（0～9）

操作与设标操作相同，不同的是它必须是设标操作之后进行，否则就为设标操作。

例如：按^ K，再按 1，这样“<1>”就会从文章中删去。

(2) 查找字符串。这里的“查找”与前面的设置标记不同，这个查找是查找真正的字符串。

例如：查找文章中的字符串“技术干部”，操作如下：

按^ QR（或 F9）使光标回到文章的开始，以便从文章之首开始往文尾方向搜寻。当然，你也可以将光标移到任意位上。然后按：

^ QF（或 F6）

屏幕显示出：

找?

输入要查找的字符串：“技术干部”

再按：

ESC 键

这时就从文首开始向后找，当找到第一个“技术干部”，光标便停在该字符串的后面，即找到了。

如果你还需要往下找，只要按：

^ L

即找下一个“技术干部”，直到文章不再出现该字符串为止。如果文章中没有你所要找的字符串，则屏幕显示：

＊＊＊没有找到：“技术干部”＊＊＊ 请按 ESCAPE 键

这时按 ESC 键，屏幕回到原来的显示状态。但是，光标已处在文章的尾部。

注意：查找字符串时，是从光标所处的位置向下找，而不可能从光标的位置向上找。因此，当你按了^ QF（或 F6）后，你要找的字符串是处在光标之上，这个字符串自然无法找到。

(3) 查找并置换字符串。置换功能可帮助你自动地将文章中

的某一字或字串用另一字或字串取代。

例如：将文章中的“技术干部”置换为“管理干部”，操作如下：

按 ^ QR（或 F9）回到文章的开始；

按 ^ QA（或 F5）；

屏幕提问：

找？

输入要找的字符串：“技术干部”，再按“回车”键；

屏幕又提问：

换成？

输入置换字符串：“管理干部”，再按“ESC”；

这时光标停在第一次找到的字符串“技术干部”的位置上，并且屏幕右上角显示：

换成（Y/N）：

键入“Y”则“技术干部”被“管理干部”自动取代。

键入“N”则不置换光标处的字符串“技术干部”，屏幕回到以前的状态，光标停在字符串“技术干部”的位置上。

置换完第一次找到的字符串“技术干部”后，如果继续找下一个，则按：

^ L

屏幕又提问：

换成？　（Y/N）

然后又选 N 或 Y，操作同上。

这样光标每遇到要找的字符串，就停下来问你是否要置换；直到全部置换完。

从上面的操作看，一篇文章多次查找、置换，这样的操作仍然比较麻烦。这时你可采用一种快速简便的方法：

在你输入完置换字符串后，不按“ESC”键而按“回车”键，这时屏幕显示：

选择？　（提示？）

然后输入：

G　在整篇文章中查找（从文首往后找）。

B　从当前光标处向文章的开始方向寻找（即从后向前找）。输入“G”或“B”后，再按“回车”键开始找第一个相符的字符串，找到后，屏幕右上角提示：

换成？　（Y/N）

键入 Y 则置换字符串，同时光标自动移到第二个相符的字符串的位置。屏幕又提示：

换成？　（Y/N）

重复以上操作，直到没有相符的字符串为止，屏幕又回到查找前的状态。

N　不提问是否置换，自动用新字符串代替旧字符串。按“回车”键执行。

NG　（或 GN）从当前光标处向文章之尾自动置换所有相应的字符串。按“回车”键执行。

BNG　（或三个字母的任意排列）　从文章尾向开始方向找，并自动置换所有相符的字符串。按“回车”键执行。

4. 排版　当你写好了一篇文章后，你可能发现文章中有不少错误，或为使文章更漂亮，需要进行增、删、改等工作。每当你在一行进行了删字、加字后，被修改行的字数可能就与其他行不一样多了，必须重新排版。

排版操作：

把光标移到要排版的那一行上，按 ^ B，即完成排版工作。每按一次 ^ B 只能排版光标所在的那一段文章，同时光标自动跳到下一段文章的开始，如果需要继续排版，可再按 ^ B，直到一篇文章排完。

有时因打印的需要，必须改变文章的行宽，这时你就应先重新设定行宽，然后对全文进行排版。设定行宽前面已讲过，这里

就不再重复了。

19—6 页设计与打印字型的控制

1. 页设计 当一篇文章书写完后，总要输出打印，而打印的格式可根据需要来定，如页长、页号、打印时行头的空格数、页尾空行数、标题与正文之间的空行数等等。在编辑状态下，我们可以通过“点命令”完成以上功能。所谓点命令是以“.”为前缀的双字母命令，它与正文一样写到文章中。每个点命令都要占一行，从行的第一列开始。在写点命令之前最好按一次 ^ N，增加一空白行，然后在该空白行上写点命令。打印时点命令不会被打印出来。

操作命令：

(1) 页长

. PLn 设置每页行数，n=（行/页）。

n 的取值范围：13～66，既定值为 66。但实际打印时每页正文长度（行数）为：n－11，例如：设 n=66，实际打印行为：66－11=55 行。当设定页长后，便自动将文章按 n－11 的长度分页。

. MTn 设置页头空行数=n，既定值=3。

. MBn 设置页尾空行数=n，既定值=8。

如以上三个命令同时设置，则每页实际行数=PLn－MTn－MBn。

(2) 标题正文间空行

. HMn 设置标题与正文之间的空行数=n，既定值=2。

. FMn 设置下标题与正文之间的空行数=n，既定值=2。

(3) 行头

. POn 设置打印时行头空格数=n，缺省时既定值=8。

(4) 打印起始页和页号

. OP 从此页开始，打印时省去页号。

. PN　从此页开始，打印页号。

. PNn　设页号＝n，并打印页号。

. PCn　设置打印页号所在的列号＝n，既定值＝33。

. PA　开始新的一页。

在需要换页行的前列写入“. PA”命令，则在. PA 的下一行上出现一行虚线，即分页标志。如果要删除分页标志则按＾OP，恢复分页标志则再按一次＾OP。

(5) 其他点命令

.“内容”引号内的“内容”是文章的注释，打印时不被打出来。

. HF“内容”　引号内的“内容”是标题。

. FO“内容”　引号内的“内容”是下标题，用来代替页号。

. CPn　若本页剩下的行数小于 n，则换新的一页，这经常用在打印表格的情况。如果某页剩下的行数不够打印一张表格，就换成新的一页，表格从新页开始打印，这样表格既完整又美观。

2. 打印字型控制　在输出打印一篇编辑好的文章前，还需要对输出字型进行控制，其方法是：将光标移到需要控制字型的字符前面，按＾P 再按所需的控制字符。例：按“＾PD”，在光标位置就会出现“＾D”，打印时则按“A”字型打印。一旦某字符前选择了一种字型，就从该字符起按这种字型打下去，直到遇见另一种控制字符为止。

十六种字型及控制字符：

A 型字	＾D	I 型字	＾R
B 型字	＾E	J 型字	＾S
C 型字	＾F	K 型字	＾T

D 型字	^ G	L 型字	^ U
E 型字	^ N	M 型字	^ V
F 型字	^ O	N 型字	^ W
G 型字	^ P	O 型字	^ X
H 型字	^ Q	P 型字	^ Y

这里讲的只适用于 9 针打印机，对 24 针打印机，字号控制要按照汉字操作的使用说明去做。

19—7 退出编辑

在编辑状态下，退出编辑有以下几种操作：

1. 存盘后退出编辑

^ KD （或 F1）按下 CTRL 键，再按 KD。

作用：把当前编辑的文件存到磁盘上，退出编辑状态到《起始命令》表下。

2. 存盘后退到操作系统

^ KX 按下 CTRL 键，再按 KX

作用：存盘后，退出汉字编辑状态并退到操作系统下。

3. 不存盘退出编辑

^ KQ （或 F2）按下 CTRL 键，再按 KQ

作用：放弃当前编辑的文件，退出编辑状态返回到《起始命令》表。

如果文件已进行过编辑，按 ^ KQ（或 F2）后，屏幕将问你是否放弃当前编辑的文本：XXX ?(Y/N)，键入 Y 则放弃，键入 N 则返回编辑状态。

另外，还有前面已讲过的“^ KS”存盘后继续编辑。

19—8 打印文件/中断

1. 在《起始命令》菜单下，按 P

屏幕上会提问：

文件名?

2. [<盘符：>] <文件名> [<. 扩展名>] ↙（输入文件名后按回车）

如果盘中没有你输入的文件名，屏幕显示：

X（盘符）：X…X（文件名）NOT FOUND

表示文件没找到，按“回车”键或“ESC”键回到《起始命令》菜单下。

找到要打印的文件后，屏幕显示：

输出到磁盘（Y/N)?

按 N 或回车键表示不做磁盘操作，此后，屏幕继续提问：

开始页号（如从第一页开始，按回车)?

从头打印按回车键即可，从某页打起，则输入那一页的页号，再按回车键。下面又提问：

终了页号（如要打印到文件结束，按回车键)?

按回车键则打印到文件结束，打印到某一页，则输入那一页的页号后再按回车键。

屏幕继续提问：

采用自动分页?

输 Y 或按回车键，则根据页设计自动换页。输 N，打印一页后不换页而连续打印第二页（页号照样打印)。

屏幕继续提问?

消除分页符号：

输 Y，不分页不打页号，文章连续打印。输 N 或按回车键，则正常打印。

屏幕继续提问：

换页时是否停一下？

输 N 或按回车键，换页时不停顿，一直打到结束。输 Y，每打完一页后停止打印，屏幕回到《起始命令》表。然后按 P 再继续打印下一页，直到打完为止。用暂停功能打印蜡纸十分有用。

回答完以上几种问题后，查一下打印机是否准备好，再按回车键。

在打印期间，随时可以按 P 来暂停打印，按 P 后屏幕显示：

停止打印按 Y，继续打印按 N

输 Y，停止打印，屏幕回到《起始命令》表。输 N，继续打印。

如果你要略去上述一连串提问，直接进入打印，当打印机已准备好后，可按 ESC 键。

19—9　运行程序

这是运行带有扩展名为. COM 或. EXE 的程序。输入 R 后屏幕显示：

输入要执行的程序名

程序名？

输入程序名后，程序开始运行，运行完毕后回到《起始命令》表下。

19—10　编辑非文书文件

这个部分主要用于编辑源程序文件和合并打印时用的数据文件，一般文字编辑不用这个命令。例如：建立 DBASE Ⅲ 的命令文件，若用 DBASE Ⅲ 的编辑功能建立，当文件超过 32K 字节时，文件的尾部就要丢失。而用“WS”的 N 编辑命令文件，就不受限制。其操作如下：

1. 在《起始命令》菜单下按：N

屏幕提问：

这个命令用来建立或修改源程序文件

一般文字编辑最好用 D 命令

文件名?

2. [<**盘符**：>] <**文件名**> [<. **扩展名**>] ↙ (**输入文件名后按回车键**)

屏幕第一行显示：

X(盘符)：X…X(文件名)　FC=1　FL=1　列 01 INSERT ON

FC 代表文件中总的字符数，包括回车符(一个回车符占两个字节)。

FL 代表当前光标所在的行。

现在可以开始编辑你的文件了，其编辑（增、删、改、插入)、存盘、退出等操作与 D 命令相同，不再重复。

19—11　合并打印

在日常管理中，常做的一项工作是给许多干部下同样内容的命令（或通知)，如果要一份一份的写则太费事了，这时你如果掌握了“WS”的合并打印的操作方法，在处理该工作时，你便会感到得心应手，取得事半功倍的效果。

1. 对主文件的处理　合并打印的文件，在编辑时首先要进行如下处理：

(1) 要在文件前面写. DF 和. RV 命令或者写. AV 命令。

(2)“变量”要用“&”号括起来。在文件中有变化的部分叫做“变量”(如命令中的姓名和单位)，在编辑时“变量”必须用“&”号括起来，如：& 姓名 &；& 单位 &；& 职务 & 等，这样“变量”的具体内容将在合并打印时输入。输入的方式有两种：

① 把变量的内容放在数据文件中，打印时由数据文件输入(这时在文件前要写上 . DF 和 . RV 命令)。

② 由操作员输入（这时在文件前要写上 . AV 命令)。

2. 由数据文件输入变量 当一篇文章中变量名很多，而且变量值很多时，将这些变量值放在数据文件中，打印时由数据文件来输入变量值，这样对操作员来说就方便得多，而且打印速度也快。用数据文件输入变量时，主文件前一定要写．DF 和．RV 命令，还要建立数据文件。下面分别介绍：

(1)．DF 命令

形式：．DF＜数据文件名＞．DAT

．DF 命令用来定义存放变量值的数据文件。数据文件名由以字母打头的 1～40 个字母或数字字符组成。

数据文件名的扩展名为：．DAT。

(2)．RV 命令

形式：．RV＜变量 1，变量 2，……，变量 n＞

．RV 命令必须与．DF 命令一起使用。变量 1～变量 n 是数据文件中的变量。．RV 命令从数据文件中读出各个变量值，输入到文章中相应的变量中去。

注意：变量名不能用汉字。

(3) 建立数据文件。在《起始命令》表下输入 N 进入非文书文件编辑，当屏幕提问“文件名?”时，输入的文件名一定要与．DF 命令中的文件名一致。

数据文件是由许多记录组成，记录写在每行上，一个记录包含若干个变量值，每个变量值之间用“逗号”(,) 隔开，若变量本身包含“逗号”，则此变量必须用“引号”(“”) 括起来，每个记录用“回车”(↙) 结束。一条记录写完了，在下一行上紧接着写下一条记录。数据文件的大小只受磁盘容量的限制。

(4) 例子。下面举例说明．DF 和．RV 命令的使用。例如：我们给一批干部下技术职务聘任通知，其方法如下：

① 编辑主文件。在《起始命令》表下输入 D，文件名为：file1

．．file1

．OP

. DF file2. dat

. RV no，name，aw，un

石 家 庄 铁 路 分 局 （通知）

石铁分干（聘） 第&no&号

★

聘 任 &name& 技 术 职 务

经考核，&name&同志具备&aw&任职资格。聘任为分局&un&科&aw&技术职务。自一九九一年十二月一日生效。

分 局 长 XXX

1991年12月30日

主送：&un&科

发给：干部科、本人、存档

② 编辑数据文件。在《起始命令》表下输入N，文件名为：file2. dat，共有多条记录：

0100，王为民，王为民，工程师，运输科，工程师

0101，王为东，王为东，经济师，劳人科，经济师

0102，张 华，张 华，统计师，统计科，统计师

⋮ ⋮

⋮ ⋮

⋮ ⋮

⋮ ⋮

0140，李自强，李自强，主治医师，石铁医院，主治医师

③ 合并打印。在《起始命令》表下，输入M，进行合并打印。

合并打印的操作同一般打印一样，只是屏幕要提问“打印份数?”，如果打印一份，则按“回车”键，若打印多份（例：打印40人的命令或通知），则输入打印份数后再按“回车”键，机器一

边输入变量值一边打印，打完规定的份数后屏幕返回到《起始命令》表下。

现在打印出如下三张命令（通知）：

石 家 庄 铁 路 分 局 （通知）

石铁分干（聘） 第0100号

★

聘 任 王为民 技 术 职 务

经考核，王为民同志具备工程师任职资格。聘任为分局运输科工程师技术职务。自一九九一年十二月一日生效。

分 局 长 XXX

1991年12月30日

主送：运输科

发给：干部科、本人、存档

石 家 庄 铁 路 分 局 （通知）

石铁分干（聘） 第0101号

★

聘 任 王为东 技 术 职 务

经考核，王为东同志具备经济师任职资格。聘任为分局劳人科经济师技术职务。自一九九一年十二月一日生效。

分 局 长 XXX

1991年12月30日

主送：劳人科

发给：干部科、本人、存档

石 家 庄 铁 路 分 局　　（通知）

石铁分干（聘）　第0102号

★

聘 任 张 华 技 术 职 务

经考核，张　华同志具备经济师任职资格。聘任为分局计统科统计师技术职务。自一九九一年十二月一日生效。

分 局 长　　XXX

1991年12月30日

主送：计统科

发给：干部科、本人、存档

打完规定的份数后，屏幕返回到《起始命令》表下。

3. 由操作员输入变量值　当写相同内容的命令（通知）数量不是太多时，或命令（通知）中变量个数不是太多，就没有必要建立数据文件，而由操作员直接输入变量值将方便一些。这时在主文件中必须写．AV命令。

形式：．AV“提示信息”，变量

提示信息是为了机器与操作员对话用，可写可不写，提示信息要用“引号”括起来，变量名不能用汉字。每个．AV命令中只能有一个变量。

仍以上面命令（通知）为例，首先在《起始命令》表下，输入D，进入编辑，文件名为：file1

```
.. file1
. op
. AV“请输入令号：”，no
. AV“请输入姓名：”，name
```

. AV “请输入姓名：”，name
. AV “请输入技术职务：”，aw
. AV “请输入单位：”，un
. AV “请输入技术职务：”，aw
. AV “请输入单位：”，un

石 家 庄 铁 路 分 局　　（通知）

石铁分干（聘）　第 &no& 号

★

聘 任　&name&　技 术 职 务

经考核，&name& 同志具备 &aw& 任职资格。聘任为分局 &un& 科 &aw& 技术职务。自一九九一年十二月一日生效。

分 局 长　　XXX

1991 年 12 月 30 日

主送：&un& 科
发给：干部科、本人、存档

然后在《起始命令》表下，输入 M，进入合并打印，当屏幕提示“打印份数?”时，输入打印份数，再按“回车”键。此后屏幕显示：

请输入令号：（输入 0100↙）
请输入姓名：（输入王为民↙）
请输入姓名：（输入王为民↙）
请输入技术职务：（输入工程师↙）
请输入单位：（输入运输↙）
请输入技术职务：（输入工程师↙）
请输入单位：（输入运输↙）
打印机开始打印第一份令，第一份令打印完后，又显示第二

份令的：

请输入令号：

：

：

：

这时你输入第二份令的令号、姓名、技术职务、单位等。依此类推，当最后一份令打完后屏幕显示：

press space bas after reading screen

这时你按下空格键，屏幕返回到《起始命令》表下。

19—12 WS命令表

1. 起始命令

D 打开文书文件，进入编辑状态。

N 打开非文书文件，编辑信件、源程序及数据等文件。

P 文件的输出；可打印输出，也可输出到磁盘。

E 更换文件名。

O 拷贝文件

Y 删除文件

R 运行带有. COM 和. EXE 扩展名的程序。

X 退出 WORDSTAR 回到操作系统下。

2. 编辑命令

(1) 光标操作。为书写简便起见，用符号" ^ " 代替 CTRL 键。

^ E 光标上移一行（= ↑）

^ X 光标下移一行（= ↓）

^ S 光标左移一位（=←）

^ D 光标右移一位（=→）

^ A 光标左移若干个连续的中文信息（或一个英文单词）

^F　光标右移若干个连续的中文信息（或一个英文单词）
^N　在当前光标处增加一行
(2) 屏幕滚动
^Z　光标不动，屏幕上移一行
^W　光标不动，屏幕下移一行
^C　屏幕上翻一页
^R　屏幕下翻一页
(3) 删除
^G　　删除光标处的字符
<DEL>　删除光标左边的字符
^T　　删除光标右边的一句（或一个英文单词）
^Y　　删除一行
(4) 其它
^B　排版
^V　进入/退出插入状态
^U　中断命令的执行

3. 字块操作命令

(1) 文件操作
^KS　将当前编辑的文件存盘，然后退回到编辑状态
^KD　存盘后退出编辑状态
^KX　存盘后回到操作系统状态
^KQ　放弃当前编辑的文件
(2) 设置标记
^K（0—9）　在文件中设置 10 个快速定位标记
(3) 字块的操作
^KB　设置字块的首标记
^KK　设置字块的尾标记
^KH　清除字块首尾标记
^KC　拷贝字块

^ KY　删除字块
^ KV　移动字块
^ KW 将字块写入临时文件
(4) 文件操作命令
^ KR 将文件读到当前编辑的光标处
^ KP 打印文件
^ KO 拷贝文件
^ KE 更换文件名
^ KJ 删除文件

4. Q 命令

(1) 光标移动
^ QS　光标移到当前行的左端
^ QD　光标移到当前行的右端
^ QE　光标移到屏幕的顶端
^ QX　光标移到屏幕的底端
^ QR　光标移到文章的开始
^ QC　光标移到文章的末尾
^ QB　光标移到字块之首
^ QK　光标移到字块之尾
^ QZ　连续上滚一行（终止滚动按空格键）
^ QW 连续下滚一行（终止滚动按空格键）
^ QP　回到原光标位置
^ QV　光标回到最后一次查询处
^ Qn　（n＝0－9）将光标快速移到标记处
^ QQ　连续重复执行某一命令
(2) 删除
^ QY　　删除光标右边一整行
^ Q<del>　删除光标左边一整行
(3) 综合命令

5. 打印字型控制

注意：同一个控制字型码对于不同的打印驱动程序其对应字型的大小和种类也不一样。

^ PD，^ PE，^ PF，^ PG，^ PH，^ PI，^ PK，^ PQ

6. 点命令

. PLn 设置每页长度（行/页）=n
. MTn 设置页头空行数=n
. MBn 设置页尾空行数=n
. HMn 设置标题与正文之间的空行数
. FMn 设置下标题与正文之间的空行数
. OP 从此页开始，打印时省去页号
. PN 从此页开始，打印页号
. PNn 设页号
. PCn 设置打印页号所在列号
. PA 开始新的一页

7. 功能键

F1 存盘后退出编辑（等于^ KD）
F2 放弃当前编辑的文件（等于^ KQ）
F3 设置左边空格数
F4 设置右边空格数
F5 更换字符串（等于^ QA）
F6 查找字符串（等于^ QF）
F7 设置字块首标（等于^ KB）
F8 设置字块尾标（等于^ KK）
F9 将光标移到文章开始处（等于^ QR）
F10 将光标移到文章末尾处（等于^ QC）

第二十章　CCED 软件使用说明

20—1　CCED 软件的特点

(1) CCED 软件的主要特点是将字处理、画线制表与数据加工融为一体，并且吸收了有关软件中颇为实用的编辑功能，优点突出。

(2) CCED 软件比较实用，完全可以根据各种需要，满足日常工作中打字、制表、简单数据加工。如果对各种控制指令键功能熟练掌握，可以随心所欲地制出所需的表格、文件。

(3) CCED 软件易学，操作简便。

20—2　启动 CCED 文字编辑软件

CCED 文字编辑软件约占 149074 字节，共由五个文件组成：

CCED	OVL	34816	1—01—80	12：22a
CCED	HLP	18816	3—26—89	12：05p
CCED	OV1	256	1—01—80	12：01a
CCED	EXE	79058	3—26—88	8：56a
CCED	BLK	16128	1—01—80	12：29a

其中 CCED. HLP 为取得帮助的说明文件。

在已建立 CCED 子目录的前提下，开机后进入子目录，键入 CCED＜回车＞ 即可启动 CCED，屏幕提示输入文件名。具体操作

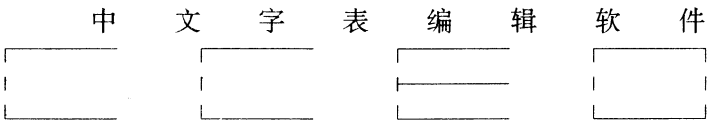
为：

在当前盘下

C>cd \cced　　<回车>

C>cced　　<回车>

屏幕显示：



键入文件名并回车，屏幕下方的状态行分别显示在编文件的文件名、光标所在的行数（Line n）（文件中的行数）和列数（Col n）、以及是否插入状态（Ins ON）、是否线保护状态（Lcp OFF）和是否画线状态（Draw OFF）。后文将详细说明。

20—3　介绍光标移动键（在 DRAW OFF 状态下）

→　光标向右移动一个字符或汉字

←　光标向左移动一个字符或汉字

↑　光标向上移动一行

↓　光标向下移动一行

Ctrl＋ →	屏幕窗口向右移动 40 列
Ctrl＋ ←	屏幕窗口向左移动 40 列
PgUp	屏幕窗口向上移动一页
PgDn	屏幕窗口向下移动一页
Ctrl＋PgUp	将光标移至文件头
Ctrl＋PgDn	将光标移至文件尾
Home	将光标移至行头
End	将光标移至行尾
Ctrl＋Home	将光标移至屏幕左上角
Ctrl＋End	将光标移至屏幕左下角
Tab	将光标右移一个制表位（有表线时，走一列表格）
Shift＋Tab	将光标左移一个制表位（有表线时，走一列表格）
Ctrl＋V（U）	可用来设置（或取消）制表位。
Ctrl＋G	当文件内容多于两个屏幕时，文件将在屏幕上自行滚动，此时，可用↑和↓改变滚动方向；用数字键可调整滚动速度；用空格键可使滚动停止或继续滚动；用字母键或回车键可退出滚动状态，回到编辑状态。

20—4 制 表

1. 方法一 用 Shift＋F8 在当前行下，生成一个规则表格，按此键后，屏幕提问各列宽度、每行表内可打字的行数以及表格的总行数。在输入各列宽度时，若表格只有 n 列，则在第 n＋1 列上按回车键即可。

例：进入 CCED 软件后

输入文件名：＜LX＞　　回车

光标出现在屏幕左上角，按 Shift+F8 键，屏幕下方提问：

“＊＊表的第一列为几个汉字宽度：？＊＊＊＊＊＊＊＊＊＊＊＊＊”

可根据事先设定的列宽，依次输入。

列宽输入完毕后，回车结束，屏幕继续提问：

“＊＊每行表线内要几行汉字（默认为 1）：？＊＊＊＊＊＊”

输入汉字行数，回车。屏幕继续提问：

“＊＊该表共需多少行（默认为 10 行）：？＊＊＊＊＊＊＊＊＊＊＊”

按照事先设定的行数输入，回车，至此一个规则的表格制成。

注意：

① 每个表列的宽度是按汉字的个数确定，每列的字符宽度＝字数 ＊ 2 ＋ 2；表宽＝ 各列宽之和。

② 制十六开纸表格，宽度约等于 78 字符，表格若每行内一行汉字，表格的总行数约等于 22 行。（其一行的设定包含表线占一行，汉字占一行，实际总行数为 46 行。）

2. 方法二 用 Ctrl+D 将状态设置为 Draw ON，然后用光标键画线，用 Ctrl+ → 和 Ctrl+←抹横线，用 PgUp 和 PgDn 抹竖线。

例：进入 CCED 软件后

输入文件名：LX1

这时在屏幕下方由左向右顺序出现如下字段：

LX1 刚输入的文件名；

Line 1 光标所在行的位置，数字‘1’即行数，是变量。

Col 1 光标所在列的位置，数字‘1’即列数，是变量。

Ins ON 插入方式与非插入方式之间的转换开关，开关由“Ins”键控制。此状态是插入方式。

Lcp OFF 表线非保护状态。其表线保护状态为 Lcp ON。它

们之间的转换开关为 Ctrl＋F 。在表线保护状态 Lcp ON 下，向表中填写数据时不会抹掉表线符；插入或删除字符时表线不会随着移动。

Draw OFF　非画线状态。画线状态为 Draw ON。它们之间的转换开关为 Ctrl＋D。在 Draw ON 状态下，用→←↑↓四个光标键可以画线；用 Ctrl＋→和 Ctrl＋←可以抹横线，用 PgUp 和 PgDn 可以抹竖线。

了解了上述几个字段的作用，就可以通过控制画线与非画线状态开关，并配合其它有关键，制出所需要的表格。

在制表过程中有几个问题需要注意：

① 每条横的表线，在表中要各占一行的位置，制成十六开纸表格约需 46 行；表的总宽度约 79 个字符。(在表内填写 24＊24 点阵汉字的情况下)

② 表格制毕后，在输入汉字前要关闭画线状态，避免光标继续画线破坏表格，并应通过 Ctrl＋F ，进入表线保护状态（Lcp ON)。

③ 表格制毕后，若想增加或减少某列（或某行）宽，仍然用画线或抹线操作将十分繁锁。这时应充分利用以下键的作用：

Ctrl＋N　在光标所在行上面插入一个空白表行，使此行加宽。

F10　删除光标所在的一行。（可由 Shift＋F10 恢复）可用 F10 删除以及 Shift＋F10 的多次恢复功能进行复制。

F6　扩展表格中当前表列的列宽（条件：光标需在此列表内)，每按一次 F6 键，表列加宽 2 个字符。

Shift＋F6　压缩表格中当前表列的列宽（条件同上)，每按一次 Shift＋F6 键，列宽减少 2 个字符。

3. 方法三　上述两种方法，各有利弊，在画表时可将两种方法结合起来使用。即：先用 Shift＋F8 键在当前行下，生成一个规则表格，然后用 Ctrl＋D 将状态设置为 Draw ON，并配合有关键

画线或抹线，满足表格局部的特殊需要。这样就能快速、方便地制出理想的表格。具体画法，不再赘述。

在向表内填写内容时，常用到以下几个键：

Ctrl＋X　　将表格中本行本列的内容对中。

Tab　　将光标右移一个表列。

Shift＋Tab　　将光标左移一个表列。

Ctrl＋V（U）　可用来设置（或取消）制表位。

另外，一般表格内的字规格无特殊要求时为 24＊24 点阵汉字，若选择 24＊24 规格以外的字型，表格的宽度或长度就会有变化，不能按前面谈到的字符宽度与行数设定表格。

20—5　文字输入

打开 CCED 并输入文件名后，一个新文件建立。此时回车，光标出现在屏幕的左上角，在当前行下，即可开始正式输入该文件内容。

1. 字型　根据使用实践，十六开纸文件的每行字符宽度约为 52 字符，文件长度大约为 26 行。这两个数据的前提是，文件内容的字型为 h 型，文件标题的字型为 l 型，规格都为 36＊36。由于各种字型的规格不同，虽然每个字在屏幕上都是占 2 个字符宽度，但实际上采用不同规格的字体后打印出来的文件在纸上的宽度与长度是不同的，对此应引起注意。这里将 2. 13H 汉字打印字型归纳为下表，供参考。

表中每个字母代表一种字型，可通过横纵两栏显示的数字，查出该种字体的规格。

例：表中 AEIM 表示 A 、E 、I 、M 四种字型，都为 24＊24 点阵汉字。

从表中归纳的字型看，大约有四种规格，即：24＊24；32＊32；36＊36；48＊48，共十三种字型是常用的。其中 24＊24 常用在

表格内；32 * 32 、36 * 36 常用在打印公文上；48 * 48 一般用在文件头上。

字 型 规 格 表

字型 横/纵	16	24	32	36	48
16	Q	—	R	—	—
24	aeimq	AEIM	r	bfjn	BFJN
32	S	—	T	—	—
36	—	cgko	—	dhlp	—
48	s	CGKO	t	—	DHLP

2. 修改 文件内容输入后，往往要根据不同需要做各种修改，如错别字、某一行文字位置的颠倒，部分内容的更动等。在修改时，常要进行文字搜索与替换，或进行文字块操作，以便进行删改或增加文件内容。

(1) 文件中字符串的搜索与替换。

F5 按此键后，即输入搜索命令。其功能是：可根据要求迅速找到文件中指定的位置(某行或某字符串)，光标将移到此处。并且可用事先设定的字符串去替换某字符串(或全部该字符串)。

输入搜索命令后：

只按回车键 放弃搜索

数字＋回车键 到文件中某行

字母＋回车键 光标移到文件末尾处

/ 字符串 搜索该字符串

C/串 1/串 2/ 找到一个串 1 后用串 2 替

C/串 1/串 2// 将后文所有的串 1 用串 2 替

/ 继续搜索

搜索找字时，不计较字母的大小写。

例 1：进入文件后，搜索某字符串。

① 按 F5 键，屏幕下方出现如下提示：

" * * * 现找 (Look for)? * "

② 若搜索文件中"专业"字符，可在问号后键入：

" * * * 现找 (Look for)? / 专业 * "

③ 回车后，光标将指示搜索到的"专业"二字。

例 2：找到一个字符串后用另一个字符串替换。

① 按 F5 键，屏幕下方出现如下提示：

" * * * 现找 (Look for)? * "

② 若将文件中"专业"字符，替换成"技术"，在问号后键入：

" * * * 现找 (Look for)? C/ 专业/ 技术/ * * * * * * * * * "

③ 回车后，搜索文件中遇到的第一个"专业"二字将被替换为"技术"二字。若在设定时，将"技术"二字后的斜杠置成双的，那么文件中所有的"专业"二字将被替换成"技术"二字。

④ 若要继续搜索，按下 F5 键，屏幕提示：

"* * * 上次找/ 专业 * * * * * * * * * 现找 (look for) * * * * * * "

键入/则 继续搜索。

(2) 文字块的操作（文字块包括矩形块和行块）。在文件修改时，文字块的操作可十分方便地将文件的某一行或某一段移动位置，达到文件修改的目的。

F8　　定义行块（矩形块）或撤销行块（矩形块）。将光标移到块的一端，按 F8 键，再将光标移到另一端，按 F8 键，块的定义就完成了。定义好的块既可做矩形块用，也可以做行块用。行块的标志是屏幕的最右列出现"＜"符，只要行块存在，这个标

志总是存在的；矩形块的标志是用反向显示，即字符和背景的颜色恰与原定义前文件字符和背景颜色相反，形成一有颜色的矩形块，任何一个操作都将引起这种显示的消失。为了重新显示矩形块可用 F4 键。矩形块是列固定的，它的内容将随着块左边内容的修改而变化。

F8 是一个反复键，块定义好后若再按一个 F8 键，则撤销原来的定义。

F4　将所定义的矩形块显示在屏幕上。

F7　将所定义的行块复制一份插入在当前行下面。

Ctrl＋L　将定义好的行块移到光标所在行下方。

Shift＋F5　删除所定义的行块。条件是在屏幕上能看见行块。行块被删除之后，在未定义新的文字块之前可用 F7 键将其恢复于当前行的下方。

Shift＋F2　改名存盘或编另一文件。此时可实现两个文件之间的块交换。

Ctrl＋O　将定义好的矩形块复制一份用来覆盖光标的右下区。

Ctrl＋Z　将定义好的矩形块复制一份插入在光标位置。

Ctrl＋K　将所定义的矩形块以插入方式移到光标处，使光标右下方的内容向右移动。

Ctrl＋B　找块首。

Ctrl＋E　找块尾。

例 1：改变某一行块在文件中的位置。

① 用 F8 键定义行块。即将光标移到块的一端，按 F8 键，再将光标移到另一端，按 F8 键，块的定义就完成了。

② 复制行块或删除行块。若将定义好的行块移到光标行的下方，按 Ctrl＋L；将定义好的行块复制一份插入在当前行的下面，按 F7 键。此后，如果要删除被定义的字块，在屏幕上能看见该行块的前提下，用 Shift＋F5 即可。

例 2：改名存盘或用某一文件编另一文件。

① 在先编的文件 1 中用 F8 键定义好要复制或转移到文件 2 中的行块，若改文件名，要对该文件进行全部定义。

② 如果文件 1 中不需要保留被转移的行块（或改该文件名）可将该行块或该文件全部，用 Shift＋F5 键删除，此后不可再按 F8 键。若无必要，也可不删。

③ 然后按 Shift＋F2 键，屏幕提问：

“若编另一文件，请输入文件名？＜LX 文件名＞回车”

调入该文件（或改名的文件）后，将光标移到需插入行块的地方，按 F7 键就可把文件 1 中的行块插入当前行的下面或生成改名后的文件，在此以前不能再用 F8 键定义新的文字块。

(3) 删除、恢复与插入新行。

文件修改时，删除字符或插入新行是难免的，在此介绍几个键的功能：

Del	删除光标上的一个字符或汉字。
Backspace	删除光标前的一个字符或汉字。
F10	删除光标所在的一行。
Shift＋F10	将最近一次由 F10 删除的一行恢复并插入在当前行上面。
F9	抹掉光标所在位置及其之后的半行。
Shift＋F9	抹掉光标所在位置之前的半行。
Ctrl＋N	在光标所在行上面插入一个空白行，或一空白表行。
Enter	当 Ins ON 以及 Lcp OFF 时，可在光标处插入一个换行符。当 Lcp ON 时，此键可用于表的列向输入。

另外，在 Lcp OFF 状态下，当光标处于行尾时，可用 Del 键将下行连接上来。当光标处于行首时，可用 Backspace 键将本行连到上行末尾。

3. 排版 文件汉字键入修改完毕后，由于对文件宽度、行间距离、列间距离、行间文字位置要求不同，因此要用 Ctrl+A 键并配合其他有关键进行排版。

例：对文件进行排版。

① 先用 F8 键将要排版的范围定义成行块。

② 然后按 Ctrl+A 键，则屏幕下方提问：

“＊＊现对行块内容排版，宽度为 76 若改之，请输新值：？＊＊＊＊（字母＋ 回车键，放弃排版）”

在提问处，输入排版宽度，回车。按照此宽度排版，排版后内容一律向右对齐，靠左放置。

③ 在具体排版过程中，要注意发挥下述键的功能：

Ctrl+X　将本行内容根据排版宽度对中。

F6　在光标位置处向所定义的行块插入一个汉字的空白列。

Shift+F6　删除由 F6 插入的一个汉字的空白列。

Ctrl+N　在光标所在行上面插入一个空白行。

F10　删除光标所在的一行。（可由 Shift+F10 恢复）

以上几个键可解决文件的行间与列间距离。Enter 键（在 Ins ON 以及 Lcp OFF 时）可进行换行，划分段落；空格键可解决汉字空位。此外，在编辑时：

F1　存盘并退出。

Shift+F1　同 F1。

F2　存盘但不退出，可继续编辑。

Ctrl+Q　退出但不存盘。

4. 打印 打印之前，先将汉字输入法设置在 ASCII 状态。

具体打印过程：

① 按 Ctrl+P 应当注意光标所在位置，即开始打印位置。若用 Ctrl+T 打印，只打印所定义的行块，光标位置不影响打印起始位。（用 F8 键定义打印范围）

② 置于打印状态后，屏幕将会提示并提问：

“从当前行开始打印…请接通打印机…”

这是按 ctrl＋P 提示，若按 Ctrl＋T 则出现：

“打印所定义的行块…请接通打印机…”

不管选择当前行打印还是块打印，屏幕都会让你选择：

“1…定页长打印

2…普通打印（按文中＾＾分页）

其它键…返回 ”

一般选择：＜1＞，定页长打印。屏幕会依次提问：

“左边空白＝1?”

此问指文件打印时左边预留的空白字符列数。

“页 长＝ 58 ?”

指每页长需设定的行数。可根据需要自定义。

“页间空白＝1（输入 1 表示页间暂停）?”

有页号时，页间空白指页号与下页之间的空白行数。

“起始页号：（ 回车键表示不要页号）”

“页号起始列＝36 ?”

指页号在其所在行的字符列位置。等号后的数字是可变量。

此时屏幕提示：

“Q…退出 F…打印到文件 其他键输出到打印机”

回车，文件打印。

另外，若选择“2… 普通打印”，需事先在文件中分页，写好页号。分页控制符是：在行尾加上＾＾ 符号。

20—6 计 算

在文件和表格中常用到一些简单的计算，可用如下两功能键完成：Ctrl＋S 对所定义的行块中从光标所在列开始的一列数字求和，结果写在光标处。

例：计算下表中每列数字之和。

数字 / 列 / 行	C1	C2	C3	C4	C5
1	20	2	6	7	8
2	10	4	4	3	2
3	30	6	3	4	5
4	25	8	2	3	4
5					

① 用 F8 键对求和的范围进行矩形块定义。

② 然后将光标置在放答数的第 5 行的 C1 列处。

③ 按 Ctrl+S，则 C1 列之和求出。然后依次将光标移到 C2～C5 列，分别再按 Ctrl+S，则 C2～C5 各列和求出。

注意：所有被求和的数据，必须在光标所在列之后或从光标所在列开始，且从光标所在列到被求和的数字之间不得有空格以外的任何字符。一个多位数，数码之间也不能留有空格。

按 Ctrl+C 键可计算光标位置左边的一个算式，此算式可包含：+ − * / ^ 及 ( ) 等六种运算以及 ROUND () 函数。其中 ROUND() 函数是按指定的位数进行四舍五入，但必须放在算式最外层。

例：求下表中每行混合运算之得数。

① 用 F8 键对混合运算的范围进行矩形块定义。

② 列算式：

C6=（C2−C4）+C3/2 * C5

并将算式置于光标位置的左侧。

③ 按 Ctrl+C 键，此时表中 C6 列各行混合运算结果求出。如

果此算式不是从文件的第一列开始，那么算式的左边至少应有一个空格。

数字 行 \ 列	C2	C3	C4	C5	C6
1	20	2	6	9	23
2	10	4	4	8	22
3	30	6	3	2	33
4	25	8	2	1	27
5	85	20	15	2	90

20—7 几点说明

(1) CCED 因受库存限制，文件不宜太长，超过 64K 后易丢失文件及内容。因无备份文件，又不易恢复。

(2) 本材料主要参考 CCED 说明书写成，有不清楚的地方，请以原说明书为准。

第二十一章　自然码汉字输入方法

目前流行的汉字输入方法很多，如全拼音、压缩拼音、五笔字形、区位码、大众码、自然码等等。一般计算机用户都要选择一种或几种汉字输入方法作为上机的基础。究竟使用哪种方法最好还要看个人的条件和工作实际的需要。

早期的汉字输入是以 CCDOS 定义的压缩拼音为基础，这种方法对熟悉汉语拼音的用户很容易掌握，但它重码问题相当突出，人为的选字严重影响输入的效率，丢漏的字多，经常需要区位码的辅助才行。在众多新的输入方式相继问世的今天，这个方式确实有些过时了。尽管如此，这种方式仍然以自己的优势占有相当的一部分较低级的微机用户，尤其是一些以使用程序为主的操作员更是如此。

压缩拼音和标准的汉语拼音很相似，只是为了提高效率对拼音中的复声母和长韵母进行了压缩，将其定义在一个键上，对应关系见下表：

压缩拼音键盘对照表

ang＝H	an＝J	zh＝A
eng＝G	ao＝K	ch＝I
ing＝Y	ai＝L	sh＝U
ong＝S	en＝F	v＝ü

区位码和电报码属于无重编码，虽然具备速度的优势但因其记忆量之大并不是一般计算机用户所能问及的，除非特别需要的

用户，很少有人问津，至多是作为一种补偿性的辅助手段。

五笔字形是流行较为广泛的输入方式，它重码少，编码规范，字词操作无切换，利于实现盲打，属于速度型的方式，在专业的汉字录入人员中有很大的市场。但作为非专业的计算机操作员，掌握这种方法往往感觉有些困难。尤其是使用汉字频度不高的用户，很难达到高效率应用。

自然码输入方法是近期流行的一种新的汉字输入法，它在CCDOS键盘的基础上对拼音进行了进一步的压缩，在单字输入时，将所有的拼音定义成两个键，第三、四码是偏旁拼音头码，其在减少重码方面有非常明显的作用。这种方案较压缩拼音简捷，效率确有提高。

21—1 自然码的基本使用方法

自然码输入方法是目前流行广泛的输入方法之一。它是一个以拼音为主的输入方式，对声母的定义基本延续了CCDOS的编排方案，只有zh采用了“v”键而不是“A”键。对CCDOS中的复韵母被压缩成了一个键，所有汉字的拼音仅用两个键就可完成定义。对于熟悉CCDOS键盘的用户，能够很快的掌握输入方法。

自然码可通过连续拼音的方式直接操作词组，大大加快了汉字的输入速度，具有记忆量小、方便易学、操作简便的优点。尤其是一些经常使用汉字但又不是专职的计算机使用人员，这种输入方法很有效率。

1. 在软盘上启动自然码系统 自然码是一个悬挂式的汉字输入系统，它本身没有汉字库，在启动自然码之前必须有一个汉字显示系统的支持。常用的汉字系统有CCDOS、长城DOS、联想DOS、2. 13等均可。

将装有《自然码》软件的磁盘插入A：驱动器内后，用批处理命令可直接启动自然码系统。

A>ZRM <↙>

屏幕提示：

```
          《自然码汉字输入系统》
中 国 软 件 技 术 公 司  1989年4月
        东城科技协作中心      周志农

0/a   — CCBIOS          1.0 — 4.0  11  行
1/b   — DESIGN          自定义系统  ??  行
2/c   — GWBIOS          长城高分辨  28  行
3/d   — LXBIOS          联想式汉卡  27  行
4/e   — CCBIOS          全兼容系统  25  行
        请 选 择  (0—4, a—e) —
```

用户可根据自己使用的监视器类型和汉字状态选择一个合适的字母或数字。

两种选择在系统的装入方式上有如下差别：

0—4　　将装入全部单字和固定词库，约占内存空间98K字节。一般常用的词组将装入内存，如果对内存环境没有过高的要求，一般选择0—4方式装入，这可使词组的输入大为方便。

a—e　　只装入单字，不装固定词库，约占内存空间52K字节。这对运行FOXPLUS等对内存要求较高的软件有用，但牺牲了自然码系统中的词组功能。用户自己定义的词组仍然可以使用。

对于CCDOS中分辨率彩显可选择第一行中的参数。长城机可选择与第三行对应的参数。

如果已经知道了应当选择的参数，也可在进入时直接带参数启动系统。

A>ZRM 0 <↙>

A>ZRM a <↙>

是适用于第一行选择的调用方法，其余类同。系统启动后在屏幕

上提示：

使用时，请在小写状态下，先按<Shift>键 放开后，再按<F1>键

如果已经定义了自造词组库，可用自造词组装入命令激活，执行命令的方式如下：

A>ZC L <↙>

实际上这条命令已经被写入批处理文件。

在B：驱动器启动自然码只需将B：指定为当前驱动器，其余相同。

2. 在硬盘上启动自然码系统 现在大多数计算机都装有硬盘，自然码系统也可在硬盘安装和启动。其方法是先按下面的方法将自然码系统装入硬盘：

C>MD ZRM <↙>　　；建立自然码子目录

C>CD ZRM <↙>　　；转入自然码子目录

C>COPY A：*. * <↙>　　；拷贝文件到硬盘

在系统装入硬盘之后，每次启动只需用如下命令：

C>CD \ZRM <↙>　　；转入自然码子目录

C>ZR <参数> <↙>　　；执行自然码引导命令

C>ZC L <↙>　　；调入自定义词组文件

便可在硬盘上启动自然码系统。这三条命令也可写入启动汉字系统的批处理文件后面，用批命令的方式直接进入自然码。

如果使用的自然码系统是加密的软盘，在硬盘启动过程中必须保证加密软盘在A：驱动器中插好，否则会造成死机。

自然码系统用<Shift>键和<F1>～<F10>键组合，定义了系统的十种功能，用于满足用户各种不同的需要。这些状态都有自己的初始值，对于熟悉该系统的人，可对通过系统中的ZRM.SYS定义和改变系统的初始值。这样可不通过键盘使系统自动进入需要的状态。对尚不熟悉系统的人，可通过键盘观察这组状态

的改变后的执行情况，摸索出适用的组合方式。

3. 自然码输入状态的进入 置键<Caps lock>为小写状态（对应该键的指示灯不亮的状态）后，依次按<Shift>键，再按<F1>键，可激活自然码输入状态。

屏幕提示行将显示：

自然：F2—音义词 F3—联想 F4—南方 F5—字集 F6—ascii
F7—检错 F8—提示 F9—表格 F10—标点

21—2 单字输入

初学自然码系统的人可打印出磁盘上的“双拼对照卡”，以便在学习中查询。方法是：

A>COPY ZRM. TAB >PRN：<↙>

1. 简码输入 自然码输入方式设置后，可用小写字母状态输入汉字。对应26个英文字母有26个使用频度较高的单字。它们可用空格键或数字1键选出。即：

q	w	e	r	t	y	u	i	o	p
七	我	二	人	他	一	是	出	欧	片
a	s	d	f	g	h	j	k	l	
按	三	的	发	个	和	及	可	了	
z	x	c	v	b	n	m			
在	小	次	着	不	你	拿			

如：“了”可用了的声″l″加空格键输入。

自然：l 1 了 2 力 3 利 4 用 5 例 6 子 7 离 8 开 9 历

此时屏幕上显示的字中会出现某些不是用″l″开头，如“用”字、“离”字。屏幕出现了一些拼音不是用“l”开头的字，如“用”字、“开”字，这些字可与前一个字组成一个使用频度较高的单词，选

择对应这些字的数字键，并不是输入这个字的本身，而是输入对应的单词。如选择数字键“4”可输入“利用”，“8”可输入“离开”。这种方式为初学者和不经常使用汉字的操作员提供了一种提高输入速度的方法。

2. 拼音输入 由于自然码在两个键上可拼出所有汉字，通过双拼对照表可拼出所有汉字。

万＝wj	之＝vi	试＝ui	虫＝is	浪＝lh
好＝hk	平＝py	安＝an	桥＝qc	就＝jq
类＝lz	窗＝id	进＝jn	盘＝pj	

如：“字”可用″ zi″ 加空格键输入。

自然：zi 1 字 b 2 自/ 3 资 b 4 子 a 5 姿 n .... 做出 ’

如果所要求的字不被包括在屏幕当前的提示行，可用“[”键向后翻页，翻过的页用“]”键向前翻。如果屏幕上已经有了所需要的汉字，可用数字键指定。自然码拼音中对各种多音字可采用不同的拼音输入，对部分常错字也可通过习惯的拼音方法找到。这会给用户带来一定的方便。

重＝is，vs　长＝ih，vh　率＝lv，uy　塑＝su，so

膝＝xi，qi　朴＝pu，pc

如：“波”可用″ bo″ 加空格键输入。

自然：bo 1 波；2 玻 w 3 播 f 4 拨 f 5 博 t 6 勃 l…

也可用″ po″ 加数字″ 7″ 输入。“波”字后面的分号“；”也属偏旁编码之一，类似的还有“。”、“/”、“，”；这样的安排有利于保持指法和减小重码率。

自然：po 1 迫 z 2 坡 t 3 婆 n 5 魄 b 6 泼；7 波；…

3. 拼音加形 很多使用拼音输入的人对汉字的重码非常恼火，有的汉字需要翻四五屏，才能找到，操作极为不方便。自然码较好的解决了这个问题。在双拼后面可输入一至两个字形码，大大降低了汉字的重码率。字形码中的大部分仍然采用偏旁的拼音头，个别的采用了形码的特点。且双拼后的形码可在屏幕的提示行中看到，在要求速度不是很高的情况下，可通过观察提示逐步熟悉。当要输入的汉字出现在提示行的第一位时用空格键选入。这种方案记忆量较小，效率很高。

如：我们需要找“亻”旁的"例"。

自然：l　1了 2力’3利 4用 5例 6子……
自然：li　1里 t 2力/ 3…
自然：lir　1例 d 2傈 x 3俐 h 4俪 a 5俚 t ……
自然:lird　2例

部首"亻"（人）用拼音的声母"r"表示，部件"歹"用拼音的声母"d"表示，采用了部首的拼音字头。自然码在输入四键后，自动认为是进入词组状态，提示编码有所改变，不再是连续数字编码，改为偶数编码，此时数字键的作用请参看词组输入部分。

又如：需要输入“心”旁的"想"，可参照如下方式输入。

自然：xdx 1想 m 2向下 w 3象形 y 4象限 m 5……

“d”是代表拼音“iang”，部首“心”用拼音的声母“x”表示。

4. 韵母键的屏幕查询 由于自然码压缩了复韵母，对于一时不够熟悉的初学者，系统设计了屏幕帮助功能。尤其是一些使用较少的复韵母，可在逐步使用中尽快掌握。

查询的方法是在输入了第一个声母后按“[”键，屏幕的提示行按字典的韵母顺序显示可与该声母拼音的压缩声母。

例：在输入“键”字时可先输入声母"j"，屏幕显示：

自然：j 1 及 2 即 ' 3 几 4 几 5 九 6 九 7 集 …

为了查询" ian" 定义在哪个键上，可按一下“ [” 键，此时屏幕显示所有用声母" j" 开头的所有拼音组合。

屏幕提示行显示：

---

j — 较 c 将 d 机 i 建 m 进 n 均 p 就 q 卷 r 窘 s 决 t 具 u 家 w 阶 x 经 y

“建” 字对应" m" 说明" ian" 被定义在键盘的" M" 键上，这时输入" m" ，可完成双拼音，用数字键" 4" 可完成“建” 字的输入。为了提高输入速度，一般再输入一个偏旁的特征字符，由于“建” 字是走针部首，可再输入“走” 字的拼音字头" z" 然后用空格键将“建” 字送上屏幕。

操作过程中的提示行：

自然：jm 1 件 r 2 间，3 简 c 4 建 t 5 见... 会晤 '
自然：hwz 1 建 r 2 践 a 3 建造 k 4 检字 i 5 尖子 i

输入三个字母后，如果有对应的词组将会在屏幕上显示，自然码输入时的词组和单字不需要切换操作，在这点上明显优于 CCDOS 。

21—3 使用联想方式

如果用户熟悉联想输入方法，在自然码系统中仍可使用。激活联想方式的操作键是依次按<Shift> 键和<F3>键。

屏幕提示：

自然：设置联想方式： （在联想状态下输入数字时，可先按一下<Shift> 键）

如：输入“计算机”时，先找到“计”字，用“0”选出；然后根据联想选择数字″ 2″，挑出“算”字；再根据下层联想选择数字″ 4″，即可输入完“计算机”这个词。

自然：j 1 及 2 即 '3 几 4 个 5 九 6 十 7 集 8 合 9 击 0 键
自然：ji 1 机 m 2 级 s 3 极 m 4 己 q 5 几… 0 计
自然：联想 1 划 2 算 3 量 4 时 5 件 6 较 7 策 8 分 9 谋
自然：联想 1 式 2 帐 3 法 4 机 5 盘 6 术 7 出 8 是 9 了

21—4　双字词输入

由双汉字组成的常用词汇可用双拼音方法一次完成输入。输入可分为两种形式。

1. 简码词输入　有些使用频度较高的双字词被定义为简码词，这些词汇可用两个单字的声母加″ '″ 键输入。当输入了两个拼音后，提示行的最后显示与键值对应的简码词。

下面是常用简码词的部分实例：

北京＝bj'	一个＝yg'	安排＝ap'	按照＝av'
完成＝wi'	日期＝rq'	方面＝fm'	总计＝zj'
有的＝yd'	需要＝xy'	我们＝wm'	中国＝vg'

如输入“可以”一词，可用″ ky″ ＋″ '″ 输入。

自然：ky 1 快 x 2 块 t 3 侩 r 4 筷 v 5 会 r .... 可以 '

又如：“使用”，可以用″ uy'″ 输入。

自然：uy 1 帅，2 率. 3 衰 p 4 摔 f 5 蟀 i 使用 '

如果在汉字操作过程中充分利用简码词输入可提高输入效

率。本系统中定义的简码词被记录在磁盘文件 JMC. LST 中，用户可将其打印后作为上机的参考。

2. 声韵双拼输入（声韵声韵、双拼双音） 实际中简码词输入的只是双字词中的一小部分，绝大多数双字词还要使用声韵声韵、双拼双音的输入方式。这种输入方式是连续输入两个单字的压缩拼音码，如果有对应的词汇将显示在屏幕的提示行上，对应的词汇不止一个时，有鸣铃声提示。

如："学习"，可用" xtxi" 输入。

自然：xtxi 2 学习

可以用空格送出，也可继续输入下文，"学习"一词将被顶出。

又如："实际"，可用" uiji" 加空格键输出，此时屏幕出现了重码，此时有鸣铃声提醒用户进行选择，同时显示：

自然：uiji 2 实际 4 事迹；6 试剂，8 世纪．0 时机/

数字键或指定的选择键可挑选需要输出的词。" 8" 可选"世纪"，" 4" 可选"事迹"。重码挑选键有规定的次序，依次是空格" "、分号";"、逗号","、句号"."、斜线" /"，操作这些键要比数字键方便，有利于保持键盘的操作指法。

词组的数字用偶数定义是为了能够从词组中选出单字，在上例屏幕中输入" 3" 可从词组"事迹"中选出"事"字，" 7" 键可选出"世"字。

又如：

安装＝anvd	字母＝zimu	编码＝bmma
需要＝xuyk	人民＝rfmn	能力＝ngli
积极＝jiji	应该＝yygl	教室＝jcui
文件＝wfjm	信息＝xnxi	影响＝yyxd

21—5 输入三个字的词组

三个字的词组可依次使用每个字的声母，如果屏幕提示的词组已经唯一，可用空格键完成输入，否则再加" ' " 键输入。

如三字词“计算机”，在击" jsj" 键后屏幕显示：

自然： jsj 1 计算机”

此时可用空格键送出，也可用“”键送出。如果三字词出现重码，选择方式与双字词组相同。

又如：

北京市＝bju'	科学家＝kxj'	进一步＝jyb'
大多数＝ddu'	自行车＝zxi'	大部分＝dbf'
办公室＝bgu'	比利时＝blu'	大使馆＝dug'
负责人＝fzr'	工程师＝giu'	近几天＝jjt'

输入过程中的三字词组经常和单字混在一起，如果不愿使用" ' " 挑选，可在三字词组的声母前加字母" e" 。

如：“锦标赛”可用" ejbs" 输入。屏幕显示：

自然：e 1 二 2 儿 ' 3 而 4 且 5 二 6 十 7 儿…
自然：ej 1 锦标赛 2 基督教 3 景德镇 4 解放后
自然：ejb 1 锦标赛
自然：ejbs 1 锦标赛

在输入“ej”后，用空格键或“1”键可输入“锦标赛”，用“4”键输入“景德镇”也是可以的。

21—6 多字词输入

四字词或多于四字的词最多用四个拼音字母输入，四字词可用四个声母直接输入，超过四个字的多字词用该词的前三个字的声母加最后一个字的声母输入。

如："千方百计"，可用" qfbj" 输入。

自然：qfbj　　2 千方百计

又如：

汉语拼音＝hypy　　澳大利亚＝adly

中国人民＝vgrm　　诺贝尔奖金＝nbej

高级工程师＝gjgu　　有关负责人＝ygfr

据不完全统计＝jbwj　　中华人民共和国＝vhrg

第二次世界大战＝decv　　大规模集成电路＝dgml

21—7 自造词及短语

1. 实用自定义词组 自然码系统能够在汉字输入的过程中定义词组和短语，词组和短语一经定义立即可以使用，不需要专门的定义程序，也不需要重新启动机器。

如"知识分子"，可用" vufz" 输入。

自然：vufz 2 知识分子

又如："七五期间"＝qwqj

按劳分配的原则＝alfz

自然码汉字输入系统＝zrmt

转换企业经营机制＝vhqv

自定义词组和短语在输入上与系统中词组没有区别，完全可以在相同的输入方式上共容。如果自定义词组和系统中词组重码现象严重，可在输入声母数量不足四个（一至三个）时按＜TAB＞键，此时屏幕只显示自定义词组。此方法除可避免重码外，还可加快自定义词组的输入速度。

假如已经定义了词组“领导干部”，可用″ l″ ＋＜TAB＞＋″ ″ 键完成输入，按″ l″ 键后提示：

自然：l 1 了 2 力 ' 3 利 4 用 5 例 6 子 7 离 8 开 9 历

此时按＜TAB＞ 键，提示变成：

自然：l　　1 领导干部

如：″ bj″ ＋＜TAB＞＝ 北京电视台

″ ux″ ＋＜TAB＞＝ 首先听取了关于

2. 在汉字输入方式下增加自定义词组　在汉字输入方式下增加自定义词组可先输入该词的一至四个声母，然后按住＜Shift＞ 键不放，再去按＜TAB＞ 键即可进入加词增加状态。

如定义“改革开放”一词，可先输入代码″ ggkf″ 中的一至四个，再按＜Shift＞＋＜TAB＞ 键。假设输入了″ gg″ 后启动增加状态。

屏幕显示：

自然：gg　　（按＜Shift＞＋＜TAB＞ 键）
请确定自定义字词代码 （1 至 4 个字母或符号，回车或空格键结束）：gg＿

此时允许修改和补充自定义词组的代码，不需要改动或改动完成按回车键确认。如果定义字光标的位置与屏幕对不上，也可

用空格键来确定。继续输入声母"kf"并回车确认，屏幕提示变成：

请输入：允许字母、数字或符号，按回车键结束， ＜Ctrl＞加回车键作废。

光标返回到原来的汉字输入位置，此后输入的汉字被定义为增加的词组，伴随增加自定义词组的过程有声音提示用户，词组长度用回车键结束。

实际的输入过程是先输入“改革”再输入“开放”并回车，自定义词组完成。

自然：glge 2 改革
自然：klfh 2 开放　　（然后按空格或回车键）
自造：ggkf 2 改革开放

此后可按系统中的词组使用方法操作该词组。在自定义词组时也可以采用自己认为合适的拼音方式。如“转换企业经营机制”用“zhjv”定义而不用常规方式“zhqv”。这样的方式在不同的操作员之间会产生影响，请斟酌采用。

3. 删除自造词　先将要删除的自定义词组调出到提示行，然后按下＜Ctrl＞键和回车键，该词即被删除。

如删除“改革开放”，则先输入"ggkf"，然后按＜Ctrl＞键，再按回车键。

自然：ggkf 2 改革开放　　（＜Ctrl＞＋回车键）
删词：ggkf 2 改革开放
自然：

如果对应该代码的自定义词组不止一个，可按选词的方法确

定删除的具体对象。

4. 保存自造词 在汉字输入过程中建立的自定义词组被放到计算机内存中，关机后自动消失。如果需要保留这些词组，就必须将内存中的自定义词组记录到磁盘文件。记录后的自定义词组可再次使用。

自然码系统中有一个自定义词组管理程序 ZC. EXE，可在 DOS 状态下执行。生成的磁盘文件名是 ZR. CZ。

在硬盘上调用自定义词组管理程序的方法：

C>ZC <↙>

```
自然码自定义字词库管理程序
L — 装入字词库          S—保存字词库 ？
```

" S" 选择将内存中的自定义词组写到磁盘文件；

" L" 选择将磁盘上的自定义词组文件读入内存。

选择的操作方式可用 DOS 的参数传递方式调用，此时不再显示功能菜单。

C>ZC S <↙>

```
自然码自定义字词库 ZR. CZ 已存盘!
```

C>ZC L <↙>

```
自然码自定义字词库已经装入 !
```

装入命令经常被写到启动自然码系统的批处理文件中。

21—8 输入中文标点

在键盘处于小写状态时，依次按<Shift><F10>键，便进入了中文标点方式。

屏幕提示：

自然：设置中文标点方式：
（在小写状态下可直接按出与其对应的中文标点符号）

此时输入中文标点的键盘值如下表：

英文	^	@	=	—	′	′	|	&	*
汉字	•	.	…	—	“	”	/	々	*

英文	,	.	/	;	{	}	[	]
汉字	，	。	、	；	“	”	《	》

再次操作该组键则退出中文标点状态。屏幕提示：

自然：退出中文标点方式：
（在自然码下，用“of”也可引出中文标点符号）

为了避免中文标点状态和一些常用符号冲突，使用特殊拼音码也可输入中文标点符号。

中文标点符号在拼音″ of″ 中，用“［”和“］”翻页，用数字键挑选。其中包含的符号有：

自然：of 1。2 、3，4. 5 “ 6 ” 7《 8》9… 0—
自然：of 1～ 2 • 3 〃 4々 5“6” 7〔 8〕9〈 0〉
自然：of 1「 2」3『 4』 5〖 6〗 7【 8】9→ 0←

另外，数字符号在″ oi″ 中，特殊符号在″ od″ 中，杂符在″ oz″ 中，罗马数字在″ ol″ 中，中文数字及字母在″ oq″ 中，日文在″ or″ 和″ op″ 中，表格符在″ ob″ 中。

经常使用的中文标点和特殊符号也可通过自定义词组的形式用拼音调出，可根据个人的情况自己决定。

21—9 输入制表符

在键盘处于小写状态时，依次按<Shift><F9>键，便进入了用大写字母（或小键盘数字）制表状态。

屏幕提示：

> 自然：设置直接画表方式：
> （用大写字母方阵和小键盘方阵可按出对应的表格符）

大写字母和制表符的对应关系如下表：

	W ┌	E ┬	R ┐	T ┌	Y ┬	U ┐		
A ─	S ├	D +	F ┤	G ├	H +	J ┤	K ─	L |
Z |	X └	C ┴	V ┘	B └	N ┴	M ┘		

小键盘数字和制表符的对应关系如下表：

7 ┌	8 ┬	9 ┐	─
4 ├	5 +	6 ┤	+ |
1 └	2 ┴	3 ┘	
0 ─		• |	

此时小键盘中的" 0" 可连续的划横线，"." 可连续的划竖线。

在表格输入状态时，屏幕提示行会显示：

> 自然：… 直接画表状态中…

输入完表格后，一定要再次操作<Shift><F9>键，用以退出制表格状态！否则系统将不接受大写字母键。

自然：退出直接画表方式：
（在自然码下，用″ ob″ 也可引出各种表格符）

20—10 输入中文数字及年月日

中文数字和日期经常用到，用压缩拼音的方式输入缺乏效率。自然码系统中提供了一种简捷的输入方式。在自然码的汉字输入状态时，按两次″ \″ 键，即进入中文数字输入状态。此时可用数字键直接输入中文数字，经常与数字对应的汉字可由一个字母键输入。中文数字输入完毕按一次″ \″ 取消中文数字状态。

按一次″ \″ 屏幕显示输入方法的控制键：

自然：\：形义 \\：数字 [,]：前后翻页
：短码词，三字词 =：双字词 TAB 键：自造

按两次″ \″ 屏幕提示行显示字母与汉字的对应关系：

数字：n 年 y 月 r 日 u 十 b 百 q 千 w 万 e 亿
o 元 j 角 f 分 /分之 %百分之 . 点

如：三亿二千一百三十八万美元 =\\3e2q1b3u8w$\
一九九一年十二月二十三日 = \\199lnu2y2u3r\
百分之七十八点三四 = \\%7u8. 34\
六时三十九分五十四秒七四 = \\6：3u9′5u4″74\

21—11　南方发音的汉字输入方式

自然码系统设置了“南方”功能，为″z，c，s″和″zh，ch，sh″发音混淆的用户提供了方便。

进入“南方”方式的方法是按<Shift>键放开后再按<F4>键。

自然：设置南方方式： （可以不必区分声母 z，c，s 与 zh，ch，sh，韵母 in 与 ing）

如：　sisi ＝四十　私事　时事　实时　实施　十四　事实　史诗　失事　逝世

sijm ＝时间　实践　事件

又如：长度＝chdu　　重量＝zsld　　重复＝csfu

设置＝sezi　　制造＝zizk

少数服从多数＝ssfs　中华人民共和国＝zhrg

除此之外＝cczy

南方发音的输入方式仅对声母起作用，部首代码和自定义词组代码不受南方音转换的影响。偏旁只许使用标准发音方式输入。如“车”旁，仍必须用″i″，不能用″c″。这是因为偏傍编码已在系统内做了唯一的规定，用户不能改变。自定义词组完全可由用户根据自己的需要确定。

如：“输”字的偏旁。

自然：sui　1 输 r　2 速成 g　3 素材 l　4 数词 i

其中的“i”、“r”仍是系统规定的“车”字、“人”字的标准发音，它不随“南方”方式的设置而改变。

21—12 摘除已经安装的自然码系统

当用户需要使用自然码占用的内存空间，又不想重新启动机器时，可摘除已经安装的自然码系统。

按<Shift>键放开再连续按<F6>三次（ascii）则屏幕会提示：

ascii：
退出自然码汉字输入系统吗（Y/N）？—　　（鸣铃）

按" Y" 键即可摘除自然码系统，此时内存中如果有自定义词组没有存盘，屏幕会自动提示：

自定义词组没存盘，退出系统吗（Y/N）？—

如果需要存盘按" N" 键，然后在DOS下执行ZC命令存盘，再重新操作退出命令。退出自然码系统后，屏幕将提示：

自然码汉字输入系统退出完毕！


广州
人民出版社