前　言

随着微型计算机的迅速普及，各种软件也在不断推陈出新，令人眼花缭乱。但是，DBASE Ⅲ作为一种最基本、最实用的编程语言，仍然显示着强大的生命力。在大多数机关、团体、企事业单位，计算机所承担的任务主要是数据管理，例如：库存、销售、财务管理；资料、档案、人事管理；生产、计划、合同管理……。用 DBASE Ⅲ开发这些管理系统，简单容易，效率很高，使广大微机用户一接触便爱不释手，这也正是 DBASE Ⅲ持久不衰并将继续发挥作用的原因。

勿庸置疑，各种软件都有各自的优点。对于一般的用户，要详尽了解每种语言的细节，几乎是不可能也是不必要的。本书针对一般数据管理系统，以 DBASE Ⅲ为例，介绍各种常用程序的编制方法，在编程中可能遇到的问题、解决方法及各种编程技巧，使读者迅速学会用计算机解决实际问题，并很快从“会用”跃变为“用好”。这些方法和技巧对于其它语言也是适用的。为了找到一条捷径，把命令从死板的格式组合成形形色色的专用功能块，本书汇集各家之长，荟萃各种实用小程序，使读者体会其中的技巧，大多数都可以在实际应用中作为子程序调用。

值得注意的是，最近推出的一些新软件不少就是针对 DBASE Ⅲ扩充、升级的，如 DBASE Ⅳ，DBASE PLUS，FOXBASE 等，它们保留 DBASE Ⅲ的优点，增强了一些功能，且与 DBASE Ⅲ完全兼容。因此，建议您编程序时不妨先从 DBASE Ⅲ开始。

本书内容精炼准确，文字简明通俗。对于初学者，它是一本很实用的入门教材；对于行家，它是一本数据库编程经验、技巧集大成之作，是一本知识蕴含丰富的实用工具书。

谨对提供资料的同仁致以诚挚的谢意。错误之处，欢迎指正。

目　录

第一章　概　述

1.1　基本概念

dBASE Ⅲ是适合微型计算机使用的关系型数据库管理系统，所以有必要对涉及到的一些基本概念加以回顾。

当今世界被称为信息时代，而数据正是对信息描述的物理体现。计算机以其存储容量大、处理速度快、运算及逻辑分析能力强的优势，使数据处理从传统的手工方式中解脱出来，使急剧膨胀的信息量可以迅速反馈。计算机处理的对象是数据，这里指的数据不同于数学中的数据，而是指人们对客观世界中事物属性的具体描述，如对一个学生的档案，我们根据需要选择其属性有：姓名、性别、出生日期、入学日期、专业、班级……，这些属性的描述可利用一些汉字、字母、符号、数字来表示。所以说对计算机而言，数据包括具体数字、字母、符号、文字、图形、表格、图象、声音。

杂乱无章的数据是没有什么意义的。在现实世界中，任何事物之间都有一定的联系，因而反映到数据之间也必然有一定的联系。对数据的处理也绝对不是对孤立的、个别的数据进行处理，例如，尽管每个学生的具体数据不同，但它们的属性则可以共同描述。数据之间的逻辑关系通常有四种表示方法：

1. 层次模型
2. 网络模型
3. 关系模型
4. 数据独立存取模型

这种按照一定的数据模型存储于计算机内部的数据的集合称为数据库，现在微型计算机使用的大部分数据库管理软件，包括本书介绍的 dBASE Ⅲ都采用关系型数据库，它的特点是把数据用二维表的形式描述：每行为一个记录，每栏为一个字段。关系型模型比较灵活，数据组织整齐、规范，特别适合一般的运算及查询。

数据库仅仅是将数据放入“仓库”，建立数据库的目的是为了应用，数据库管理系统就是为了实现这一需要的一套软件，它实现对数据的处理，其内容一般包括对数据的收集、存储、传递、分类、排序、计算等。

计算机对外设(主要是磁盘)的管理是以文件的形式进行的。这里所说的文件也与日常生活中的文件不同。它是指一组已命名的信息的总和，通常是指一组指令的集合(即程序)或若干程序的集合，或一批有标识的数据。文件的标识为文件名。dBASE Ⅲ的用户文件以专门指定的扩展名标识。

1.2 DBASE Ⅲ 简介

dBASE Ⅲ是美国 Ashton—Tate 公司于 1984 年 6 月推出的。它对 dBASE Ⅱ的一些不足和缺点做了改进，是针对 16 位微机而开发的一个关系型数据库管理系统。dBASE Ⅲ的主要部分是用 C 语言编写的，在 DOS2.0 或更新的 DOS 版本支持下，可以在各种档次的 IBM 系列微机及其兼容机上运行。

dBASE Ⅲ系统比较大，仅常驻内存部分(dBASE·EXE)就有 113KB，再加上可覆盖模块及 DOS，因此至少需要 256KB 以上内存才能运行。除此之外，dBASE Ⅲ还要求系统至少配置两个 360K 软盘驱动器(或者一个软盘驱动器加一个硬盘)，一台 CRT 设备供显示，一台行宽 80 列以上的打印机供打印。

在汉字操作系统下，dBASE Ⅲ可以使用汉字。一般说来，凡是允许使用西文字符的地方都可以使用汉字。用户可以把汉字看作一种特殊字符——一个汉字相当于两个西文字符。

dBASE Ⅲ装在两张软盘上，一张是系统盘，另一张是实用程序盘。系统盘上主要有以下文件：

DBASE·EXE	dBASE Ⅲ的总控模块及常驻内存模块
DBASE·OVL	dBASE Ⅲ的可覆盖模块
HELP·DBS	dBASE Ⅲ的说明书，供 HELP 命令使用
ASSIST·HLP	供 ASSIST 命令使用的文件
READ·ME	dBASE Ⅲ的使用说明
CONFIG·SYS	DOS 的系统配置文件

实用程序盘上的文件较多，主要有：

DFORMAT·EXE	格式文件生成程序
DFM·MSG	DFORMAT 程序的使用说明
DCONVERT·EXE	把 dBASE Ⅱ文件转换成 dBASE Ⅲ文件的程序

dBASE Ⅲ的主要性能如下：

一个数据库最多可容纳的记录个数： 10^9

每个记录最多可包含的字符数(byte)： 4000

每个记录最多可包含的字段数： 128

每个字段最多可包含的字符数： 254

数值运算精度： 16 位有效数字

最多可使用的内存变量个数： 256

所有内存变量最多可占用的字节数： 6000

工作区个数： 10

与其它高级语言相比，dBASE Ⅲ有以下显著优点：

1. 简单易学。dBASE Ⅲ命令比一般高级语言更接近于自然语言，dBASE Ⅲ程序如同一篇自然语言文章，用户阅读理解程序及自己编写程序都比较容易。

2. 功能强。一条 dBASE Ⅲ命令抵得上一般高级语言的好多条指令。数据处理中的常见操作，如排序、索引、合计、求平均值等等，只需要一条命令就可完成，这就大大减少了

编制应用软件的难度，降低了对程序设计人员的要求。

3. 数据和程序互相独立。修改数据后不用更改程序，程序修改后原先的数据也勿需重新组织。多个用户或多个应用软件可以共用数据。

4. dBASE Ⅲ采用交互处理方式，这就使得使用计算机变得非常简单了。计算机与用户可以互相“对话”，通过“和计算机交谈”、“点菜单”、“填表格”来完成各种复杂的数据管理工作。一般人员只要经过短时间的学习，就有能力借助计算机处理日常数据业务，因此特别适合在不具备计算机知识的一般企事业干部中推广普及。

1.3 dBASE Ⅲ 的安装、运行及退出

为了使dBASE Ⅲ能够工作，在DOS目录下应该有一个结构设置文件，文件名为CONFIG·SYS。虽然它不是系统启动的必要文件，但系统启动后首先检索它，如果存在，则按该文件内容对系统进行初始化设置；如果不存在，系统按约定值进行初始化设置。由于一般应用软件（如BASIC等）对系统的约定值都能满足需要，所以对CONFIG·SYS文件的存在与否不大在意。但在dBASE Ⅲ中，由于运行需要同时打开多个各类文件，而且为了加快运行速度需要增加缓冲区，就需要按要求将配置设定到CONFIG·SYS文件中。如果没有CONFIG·SYS文件或者设置不正确，dBASE Ⅲ就无法运行。

CONFIG·SYS文件通常包含以下几条命令：

1. BREAK＝ON/OFF　（约定值OFF）

设置中断状态，对dBASE Ⅲ无效 。

2. device＝文件名全称　（无约定值）

目的是在DOS系统下将一些特殊功能软件装入内存，作为对操作系统功能的扩充，这些文件的扩展名为“·SYS”。例如：DEVICE＝GRD·SYS就是将图形软件包装入内存，供应用软件做图用。

3. FILES＝n　（n＝3～99　约定值＝8）

在dBASE Ⅲ中，一般设置成FILES＝20。在这20个文件中，DOS占用5个作为标准输入、标准输出、标准错误输出、辅助设备和标准打印机，另外dBASE Ⅲ本身占用两个供主程序和覆盖程序使用，留给dBASE Ⅲ应用程序的文件还有13个，一般情况下是够用了。

4. BUFFERS＝n　（n＝3～99　约定值＝2）

用于设置系统的文件读写缓冲区数量，dBASE Ⅲ一般设置成BUFFERS＝24。系统运行时，内存与外存间的读写是借助读写缓冲区作为中转环节来进行的，磁盘内容读入内存，实际上是首先将所在扇区的全部代码读入缓冲区，然后有选择地把所需内容从缓冲区送到文件的字段或者变量中。由于缓冲区存储的内容可以反复多次读，这样就减少了访问磁盘次数，提高了运行速度。缓冲区多，程序运行速度就快。但是缓冲区也不能太多，因为每增加一个缓冲区，就要占用512字节内存，并增加判断和查找时间。所以内存不够时可以减少缓冲区数量，但不能少于12个。

5. SHELL＝文件名全称　（约定文件名为COMMAND·COM）

用该语句可以把 DOS 命令处理程序改为自己开发的命令处理程序(文件名不同)。

如果 CONFIG·SYS 文件不存在，可在 DOS 状态下按以下步骤建立之：

```
C>COPY  CON: CONFIG·SYS
FILES=20
BUFFERS=24
按 CTRL+Z 键
```

这样 DOS 盘上就建立了一个 CONFIG·SYS 文件。当然也可以使用其他软件如 EDLIN 等来生成该文件。

另外，在 dBASE Ⅲ 系统盘上最好再建立一个 CONFIG·DB 文件，以便简化 dBASE Ⅲ 的某些操作。例如：欲在 A 盘上运行 dBASE Ⅲ，可在该文件中加入这样一条命令：

```
DEFAULT=A:
```

它的意义是将当前驱动器置为 A:，这样设定以后，在 dBASE Ⅲ 的所有操作中，如果文件名前不加盘符，系统就默认为是 A 盘。欲在 dBASE Ⅲ 中使用其他编辑软件，例如 EDLIN·COM，可在该文件中加入以下命令：

```
TEDIT=EDLIN·COM
```

这样设定以后，在 dBASE Ⅲ 下键入 MODIFY COMMAND 命令后，将会自动进入 EDLIN 编辑状态。利用这一方法，用户可选择自己熟悉的软件来编辑程序，如 HW、WPS 等。CONFIG·DB 文件还可以用来设置 dBASE Ⅲ 的初始状态，包含屏幕颜色设置、功能键定义等命令。文件中的每一行均呈以下形式：

```
状态参数名=参数值
```

如果要在硬盘上运行 dBASE Ⅲ，应先把 dBASE Ⅲ 系统盘插入 A 驱动器，用 COPY 命令把 dBASE Ⅲ 系统盘上的全部文件都拷贝到硬盘上：

```
A>COPY  A:*·*  C:
```

之后，就可以使用硬盘运行 dBASE Ⅲ 了。

dBASE Ⅲ 的运行环境设置好后，可按以下步骤进入 dBASE Ⅲ：

把 dBASE Ⅲ 系统盘插入某个驱动器，并置该驱动器为当前驱动器(设为 A:)。在操作系统提示符后键入：

```
A>DBASE
```

屏幕上显示出一段有关版权的说明，在这段说明文字下面，显示出一个圆点“·”。圆点是 dBASE Ⅲ 的提示符，它表示 dBASE Ⅲ 正等待用户敲入命令。

如果 dBASE Ⅲ 已拷入硬盘，则直接键入：

```
C>DBASE
```

就可进入 dBASE Ⅲ 运行了。

退出 dBASE Ⅲ 的命令是，在圆点提示符后键入：

```
·QUIT
```

则关闭所有被打开的文件，终止 dBASE Ⅲ 运行，返回操作系统，屏幕显示操作系统提示符。每次结束运行 dBASE Ⅲ，都要通过执行 QUIT 命令正常退出，否则有可能破坏数据库或者造成数据丢失。

1.4 dBASE Ⅲ 的文件类别

为了对数据进行管理，dBASE Ⅲ用 9 类文件存放数据及数据的有关信息。这些文件存放在磁盘上，当需要使用时被读入内存。dBASE Ⅲ借助它们对数据进行各种各样的处理。

要给每个文件起一个名，它由文件名和类型名两部分组成。文件名由字母或汉字打头的 1～8 个字符组成。组成文件名的字符可以是：大小写英文字母、汉字、数字及除了空格、逗号、尖括号、竖线“|”、下斜线“\”之外的符号。类型名(亦称扩展名)由系统自动给出，它们指明了文件的类型，一般毋需用户定义或给出。

dBASE Ⅲ的九类文件如下(括号中是该类文件的类型名)：

1. 数据库文件（·DBF）

数据库文件是 dBASE Ⅲ的基本文件。它存放数据库的结构(该数据库有多少个字段，各字段的名称、类型和宽度)和一系列记录。通常简称数据库。

2. 备注文件（·DBT）

若数据库中有备注型字段，则在建立数据库的同时，也建立了一个文件名相同但类型名为“·DBT”的备注文件，用来存放该数据库的所有备注型字段。若数据库中无备注型字段，则不产生备注文件。

每个含有备注型字段的数据库打开时，与其同名的备注文件也同时被打开。如果把备注文件删掉，则它对应的数据库就打不开了，要想打开这个数据库，必须再建立同名的备注文件。

3. 索引文件（·NDX）

索引文件是由索引命令生成的，只包含排了序的“关键内容”及其对应的记录号。

4. 内存变量文件（·MEM）

用来存储内存变量的名称、类型和值。

5. 命令文件（·PRG）

它是一系列 dBASE Ⅲ命令组成的程序，简称程序。命令文件是 ASCII 码文件，它可以用 dBASE Ⅲ的 MODIFY COMMAND 命令或其它字处理软件如 EDLIN、WORDSTAR 等来建立。

6. 屏幕格式文件（·FMT）

屏幕格式文件由@—SAY，@—GET，NOTE 三种命令组成。它规定了一种屏幕显示格式，使得用户可以用自己安排的屏幕格式来输入或修改数据。

7. 报表格式文件（·FRM）

报表格式文件是一种用来描述输出报表格式的文件，它可以用 MODIFY REPORT 命令建立或修改。

8. 标签格式文件（·LBL）

标签格式文件由 MODIFY LABEL 命令建立，用来把数据库中的某些栏目按照一定的格式打印成标签，如名片、某种物品的品名、价格及产地等。

9. 文本输出文件（·TXT）

文本输出文件是以 ASCII 码形式存放的数据文件，它可以用 DOS 的 TYPE 命令显示或打印出来。

DOS 以及 DOS 支持下的各种高级语言都允许建立和使用 ASCII 码数据文件，其扩展名均为“·TXT”。在 dBASE Ⅲ 下，利用 COPY 命令可以产生“·TXT”型文件；利用 APPEND 命令可以从任何“·TXT”型文件中读取数据。因此，借助“·TXT”型文件，dBASE Ⅲ 可以和 DOS 及其支持下的各种语言共享数据资源。

除此之外，每当用 MODIFY 命令修改某个文件或数据库的结构时，系统还自动生成一个该文件的副本，称为后备文件，其类型名为“·BAK”，保存该文件未被修改时的数据。万一在操作过程中文件的数据丢失了(如发生断电等意外事故)，用户可以从后备文件中恢复数据。

dBASE Ⅲ 同时最多允许打开 15 个各种类型的文件，同时最多可打开 10 个数据库文件，如果某个数据库包含备注型字段，则打开该数据库意味着打开了两个文件，一个是该数据库本身，另一个是它的备注文件。一个文件打开后，在没有关闭之前不能再次被打开。

1.5 dBASE Ⅲ 的数据组织方式

一个关系型数据库从逻辑上看相当于一张二元表格，例如工资表、人事档案表、订货明细帐等等。请看下表。

表 1—1　人事档案数据库(RSDA.DBF)的数据结构示意

	姓名	性别	出生日期	籍贯	工资	职务	婚否	简历
1	耿太白	男	02/12/25	四川	130.00	工人	.T.	memo
2	司马威	女	11/23/55	山东	50.00	技术员	.F.	memo
	⋮							

每一张“表格”有固定的若干个纵栏和许多横行(横行的个数是任意的)。表中的一拦称为一个字段(field)，每个字段有一个名称——称为字段名，它相当于表格中的栏目；表中的一行称为一个记录(record)，每个记录有一个序号——称为记录号，它相当于该记录号所对应的“行”在表中的行号。

以上面“人事档案数据库”为例，它有姓名、性别、出生日期、籍贯、工资、职务、婚否、简历 8 个字段和若干个记录(记录的数目可以随时增减)，每个记录是某个职工的具体姓名、性别、出生日期……

在 dBASE Ⅲ 中，一个数据库最多可有 128 个字段，这些字段所占用的字节总数不得超过 4000；一个数据库最多可有 10^9 个记录，最多可占用的字节数为 2×10^9。

字段是数据库中最基本的数据单位，对每个字段都需要从以下三个方面来定义：

1. 字段名(field name)：字段名由字母打头的1～10个字母、数字或下划线“—”组成。定义字段名时使用大写字母或小写字母均可，但在显示时一律以大写字母出现。字段名也可以是汉字，一个汉字相当于两个西文字符，因此字段名最多可以是5个汉字。

例如：姓名、籍贯、No—3、LAST—NAME等都是合法的字段名；而以下几个字段名则是非法的：A＊B、5X、署期高温补助、A：B……

2. 宽度(width)：该字段最多允许包含的字符数。

3. 类型(type)：该字段中所存储数据的类型，有以下五种：

①字符型(C型)——字符型字段的值是由字母、数字、符号及空格组成的字符串(以ASCII码存储)，如人事档案中的“姓名”、“籍贯”、“职务”字段，最大允许宽度为254。当字段值实际宽度少于规定宽度时，在右侧用空格补齐，因此在列表输出时，C型字段均为左对齐。

②数值型(N型)——数值型字段的值是一个可进行算术运算的数，如人事档案库中的“工资”字段，最大允许宽度为19(包括小数点)。数值型字段又分整数型和小数型两类：对于整数型字段，数据最多可以是19位正整数或负整数；对于小数型字段，数据的整数部分不得超过16位，小数部分(不包括小数点)不得超过总宽度减2且最多只能取15。例如：当定义宽度等于10时，小数位数不得超过8；当定义宽度等于19时，小数位数最多只能取15。

当字段值实际宽度少于规定宽度时，在左侧用空格补齐，因此在列表输出时，N型字段均为右对齐。

③逻辑型(L型)——逻辑型字段的值是逻辑值“真”或“假”，如人事档案中的“婚否”字段，宽度固定为1。 输入时，“真”可输入.T.或.t.或.Y.或.y.；“假”可输入.F.或.f.或.N.或.n.。 输出时，“真”输出.T.；“假”输出.F.。

④日期型(D型)——日期型字段的值是日期，如人事档案库中的“出生日期”字段，宽度固定为8。其一般格式是：月/日/年，其中月、日、年均为两位数。例如：07/01/48表示1948年7月1日。

⑤备注型(M型)——备注型字段的值一般是一段文字，如人事档案中的“简历”字段。这段文字按512字节为一个数据块来存放，每个备注型字段最多可有8个数据块，占4K(4096)字节。M型字段的值存放在另一个专门的文件中(称为备注文件)，在数据库中仅存放它的一个指针——指示该字段的值在其备注文件中的位置，其宽度固定为10。显示数据库结构时，对这种字段仅标示“memo”以表示它是M型字段。dBASEⅢ提供了专门的手段，对这种字段进行输入、编辑及列表。每个数据库最多可有128个备注型字段。

只有当数据库中有超过254个字符的字段时，才有必要定义M型字段。否则最好不要定义M型字段，以免降低工作效率。

1.6 dBASEⅢ与dBASEⅡ的区别

由于dBASEⅢ是在dBASEⅡ的基础上做了改进和扩充，所以它比dBASEⅡ有显著优点是肯定的。凡是使用dBASEⅡ的用户明显感觉到的不足之处，在dBASEⅢ中都得到了满

意的解决。一般说来，dBASE Ⅲ几乎包含了dBASE Ⅱ的全部命令、函数和功能，弃dBASE Ⅱ转而改用dBASE Ⅲ没有什么困难，只会更加得心应手。但用dBASE Ⅱ编制的程序原原本本移到dBASE Ⅲ下是不能运行的，因为个别命令的格式做了改动；dBASE Ⅱ下建立的数据文件(·DBF)在dBASE Ⅲ下也不能被调用，因为二者的内部结构不同。由于dBASE Ⅱ是较早推出的数据库管理系统，普及范围较广。为了提醒对dBASE Ⅱ已颇熟悉的用户的注意，摘其要点列在下面：

dBASE Ⅲ与dBASE Ⅱ主要性能对比

性　　能	dBASE Ⅲ	dBASE Ⅱ
一个数据库最多可容纳的记录数	10^9	65535
每个记录最多可包含的字符数	4000	1000
每个记录最多可包含的字段数	128	32
每个字段最多可包含的字符数	254	254
字段类型	C,N,L,D,M	C,N,L
数值运算精度	16位有效数字	10位有效数字
最多可使用的内存变量个数	256	64
所有内存变量最多可占用的字节数	6000	1536
可同时使用的数据库个数	10	2
排序命令允许使用的关键字段个数	多个	一个

一. dBASE Ⅲ的几点改进

1. 处理速度大为提高。因为dBASE Ⅲ充分利用16位微机的硬件特性，并增加了过程文件的调用，减少了磁盘访问次数，所以运行速度比dBASE Ⅱ快得多。

2. 指标的限制大幅度放宽。从上面的性能对比可以看出，数据库和变量的使用都比dBASE Ⅱ方便得多，不用担心超出规定，且变量提供了全局和局部之分。

3. 数学运算能力增强。dBASE Ⅲ提供了乘方、开平方、指数、对数函数，运算精度由10位提高到16位。

4. 扩充了数据类型。由原来的数值型、字符型、逻辑型三种增加到五种，新提供了日期型和备注型。

5. 对库结构的修改更方便。利用MODIFY STRUCTURE命令对数据库结构进行修改后，原库中的数据将自动从后备文件中恢复，不用担心数据库结构的修改会造成记录的丢失。

6. 若内存允许，可直接在dBASE Ⅲ下执行DOS命令，执行完返回dBASE Ⅲ。

7. 增加了两条提示命令：HELP和ASSIST，便于用户学习和使用dBASE Ⅲ。

二. dBASE Ⅲ与dBASE Ⅱ的几点差别

1. 二者数据库的内部结构不同，不经过转换无法直接使用。

2. dBASE Ⅲ的字段名、内存变量名都不允许使用冒号“:”，所有需要使用冒号处可改用下划线“—”代替。

3. 使用SKIP命令使记录指针上跳超过库顶时，dBASE Ⅱ的记录指针指在0号记录，而dBASE Ⅲ的记录指针仍指在库顶，BOF() =.T.；记录指针下跳超过库底时，dBASE Ⅱ、dBASE Ⅲ均使指针指在最后一个记录之后的空记录上，EOF() =.T.。

〔例1—1〕在dBASE Ⅱ中

```
·USE  RSDA
```

```
·? #
   1
·SKIP  -1
·? #
   0
```

〔例 1—2〕在 dBASE Ⅲ 中

```
·USE  RSDA
·? RECNO ()
   1
·SKIP  -1
·? RECNO ()
   1
·? BOF ()
   .T.
```

4. 使用 FIND 命令找不到满足条件的记录时，dBASE Ⅱ 的记录指针指在 0 号记录上，EOF()=.F.，而 dBASE Ⅲ 的记录指针指在最后一个记录的空记录上，EOF()=.T.

〔例 1—3〕在 dBASE Ⅱ 中(设 RSDA 库共有 50 条记录)

```
·USE  RSDA
·INDEX  ON  姓名  TO  XM
·FIND  唐伯虎
   No  find
·? EOF ()
   .F.
·? #
   0
```

〔例 1—4〕在 dBASE Ⅲ 中(设 RSDA 库共有 50 条记录)

```
·USE  RSDA
·INDEX  ON  姓名  TO  XM
·FIND  唐伯虎
   No  find
·? EOF ()
   .T.
·? RECNO ()
   51
```

5. 以下 dBASE Ⅱ 命令在 dBASE Ⅲ 中不能使用：

```
QUIT  TO
READ  NOUPDATE
REMARK
RESET
```

TEST

UPDATE ADD

有些命令的功能在dBASE Ⅲ中做了变动或改进，使用时要注意其差别，如：

ERASE

CLEAR

CLEAR ALL

MODIFY

UPDATE

@—SAY

@—SAY—GET

6. 给逻辑变量赋值时，在dBASE Ⅱ中，逻辑值不能用点“·”括起来；而在dBASE Ⅲ中，为了避免逻辑值与变量名混淆，规定逻辑值必须用点“·”括起来。例如，给变量A赋以逻辑值“真”，在dBASE Ⅱ中，命令格式是：STORE T TO A； 而在dBASE Ⅲ中，命令格式是：STORE .T. TO A 或 A=.T.。

7. dBASE Ⅲ向用户提供了37种函数，比dBASE Ⅱ增加了20个，其中有10个函数的名称与功能和dBASE Ⅱ完全一样，他们是INT，TRIM，CHR，STR，VAL，EOF，FILE，LEN，DATE，&；TYPE函数名称未变，但功能加强了；有6个函数的功能与dBASE Ⅱ一样，但为了使用户更容易理解记忆，函数名做了修改，如下：

在dBASE Ⅱ中的函数名	在dBASE Ⅲ中改为
@	AT
$	SUBSTR
!	UPPER
RANK	ASC
*	DELETED
#	RECNO

8. dBASE Ⅲ删除了dBASE Ⅱ中的一些SET命令，这些命令在dBASE Ⅲ中不能用：

SET COLON ON|OFF

SET EJECT ON|OFF

SET LINKAGE ON|OFF

SET RAM ON|OFF

SET SCREEN ON|OFF

SET DATE ON|OFF

SET HEADING ON|OFF

9. 以下列出dBASE Ⅲ新增加的一些函数和命令：

dBASE Ⅲ新增函数表

函 数	内 容
BOF ()	数据库首部
CDOW ()	星期数（英文名）
CMOUNTH ()	月份名（英文名）
CTOD ()	字符型数据转换为日期型数据

DAY ()	月内日期数（1—31）
DOW ()	星期数（1—7）
DTOC ()	星期型数据转换为字符型数据
EXP ()	自然指数
LOG ()	自然对数
MONTH ()	以数字形式表示月份（1—12）
ROUND ()	对小数进行四舍五入
SQRT ()	平方根
YEAR ()	年份（1900—1999）

dBASE Ⅲ 新增命令表

命　令	功　能
ASSIST	用菜单方式帮助用户使用系统
AVERAGE	求平均值
CLOSE	关闭文件
COPY FILE	复制文件
HELP	用菜单方式对命令用法进行说明
LABLE FROM	输出标签
MODIFY LABLE （或 CREATE LABLE）	建立和编辑标签格式文件
PRIVATE	规定内存变量为局部型
PUBLIC	规定内存变量为全局型
RUN	执行 DOS 命令或目的程序
SEEK	按索引文件查找记录
TYPE	输出 ASCII 文件
ZAP	清除数据库中的所有记录

dBASE Ⅲ 新增 SET 命令

命　令	功　能
SET	用菜单方式设置系统各种参数
SET COLOR TO	设置屏幕颜色特性
SET DECIMALS TO	设置小数位数
SET DELIMITER	设置数据边界符
SET FILTER TO	列表时“过滤”掉某些记录
SET FIXED	决定是否所有数据显示时均有固定小数位数
SET FUNCTION	定义各功能键
SET HEADING	决定列表时是否列出字段名
SET HELP	决定输入错误时是否询问“DO YOU WANT SOME HELP?”
SET MENUS	决定某些命令在全屏幕编辑方式下是否附简要控制键功能菜单
SET PATH	决定文件查找路径
SET PROCEDURE	打开一个过程文件
SET SAFETY	决定文件重写时是否发出询问
SET UNIQUE	用以建立无重复项的顺序清单
SET RELATION TO	对两个数据库设置关联

第二章　dBASE Ⅲ的语法规定

2.1 常　量

dBASE Ⅲ有三种常量：数值型常量、字符型常量及逻辑型常量。

1. 数值型常量

数值型常量可以是整数或实数，如 15，－203，3.14159……，数字的最大位数为 19 位（包括小数点）。

注意！　在 dBASE Ⅲ中，数不能写成指数形式。例如 1.3×10^8 不准写成 1.3E＋8 或 1.3D＋8，而只能写成 130000000。

2. 字符型常量

字符型常量是用单引号、双引号或方括号括起来的字符串。如果字符串本身包含这三种定界符之一，则该字符串要用另一种定界符括起来。一个字符串最多可包含 254 个字符。一个汉字相当于两个西文字符。

例如："A12"、〔AB'CD'EF〕、〔北　　京〕，都是正确的字符串；"AB"CD"EF"、〔上海〔复兴路〕〕，则是错误的字符串。

3. 逻辑型常量

逻辑型常量只有"真"、"假"两个值。

输入时，"真"可输入.T.或.t.或.Y.或.y.；　"假"可输入.F.或.f.或.N.或.n.。　输出时，"真"输出.T.；　"假"输出.F.。

2.2 变　量

dBASE Ⅲ有两种变量：字段变量和内存变量。每种变量又分为数值型、字符型、逻辑型和日期型四种。（字段变量还有一种备注型，但备注型字段一般不会在表达式中出现）

字段变量是数据库系统要处理的最基本的变量。字段变量的名称就是数据库中的字段名，字段变量的值是记录指针此刻所指记录（称为当前记录）中该字段的数据。

内存变量是独立于数据库而存储在内存中的一种变量，用作程序中的临时性工作单元，用于存放常数、中间结果或最终结果。内存变量的命名规则与字段名相同。dBASE Ⅲ允许同时最多使用
256 个内存变量，所有内存变量最多可占用 6000 字节的内存。

dBASE Ⅲ没有专门定义变量类型的命令，因此也毋需象一般高级语言那样，变量要先定义类型后才能被使用。字段变量的类型在建立数据库时由用户指定，而内存变量的类型取决于输入数据的类型，把数据输入内存变量的同时，也就决定了该内存变量的类型。例如，对

内存变量A，若给它输入一个数值，则A的类型是数值型；若给它输入一个字符串，则A的类型是字符型。

内存变量名可以与数据库中的某个字段名相同。为防止混淆，在访问内存变量时，可在该内存变量名前面加一个前缀“M－>”，如下：

M－>内存变量名

内存变量又分为全局变量和局部变量两种。全局变量是在任何一级应用程序中均可使用的内存变量。在dBASE Ⅲ提示符号下直接定义并赋值的内存变量均为全局变量。除此之外，在任何一级应用程序中，都可以用PUBLIC命令把一些内存变量定义为全局变量：

PUBLIC　内存变量表

出现在应用程序中但未定义为全局变量的内存变量都是局部变量。局部变量仅在定义它的程序及低层程序中有效，当定义它的程序执行结束时被自动删除，当返回主程序时该内存变量的值是在子程序中被新赋予的值。

如果某个被调用的程序需要使用几个与上层程序的局部变量或全局变量同名的局部变量，但又希望不破坏这些内存变量的原值，这时可在该程序中用PRIVATE命令把这些“同名局部变量”定义为“隐蔽局部变量”如下：

PRIVATE　〔内存变量表〕|〔ALL〔LIKE/EXCEPT　含替代符的内存变量名〕〕

这样，这些内存变量的原值就被保护(隐蔽)起来，当该程序运行结束，又返回其上层程序时，这些内存变量又恢复原值。PRIVATE命令有以下几种形式：

PRIVATE　内存变量表

把内存变量表中列出的内存变量定义为隐蔽局部变量。

PRIVATE　ALL　LIKE　含替代符的内存变量名

把与“含替代符的内存变量名”相匹配的内存变量定义为隐蔽局部变量。

PRIVATE　ALL　EXCEPT　含替代符的内存变量名

把与“含替代符的内存变量名”不匹配的内存变量定义为隐蔽局部变量。

〔例2－1〕有A·PRG和B·PRG两个程序，B·PRG是A·PRG的子程序，其内容如下：

```
    A·PRG 程序
SET TALK OFF
RELEASE ALL
X=1
Y=2
Z=3
?“调用B程序前,A程序的内存变量:”
LIST  MEMORY
DO  B
?“调用B程序后,A程序的内存变量:”
LIST  MEMORY
RETURN
```

B·PRG 程序

```
PRIVATE  X
PUBLIC  A2
X=10
Y=20
A1=11
A2=22
?"B 程序的内存变量:"
LIST  MEMORY
RETURN
```

运行 A·PRG 程序,屏幕上将依次显示以下信息:

```
调用 B 程序前,A 程序的内存变量:
  X     Priv     N     1 (1.000000000)     B: A·PRG
  Y     Priv     N     2 (2.000000000)     B: A·PRG
  Z     Priv     N     3 (3.000000000)     B: A·PRG
  3  variables  defined, 27  bytes  used.
253  variables  available, 5973  bytes  available.
B 程序的内存变量:
  X     Priv (hidden)     N     1 (1.000000000)     B: A·PRG
  Y     Priv              N     20(20.000000000)    B: A·PRG
  Z     Priv              N     3 (3.000000000)     B: A·PRG
  A2    Pub               N     22(22.000000000)    B: B·PRG
  X     Priv              N     10(10.000000000)    B: B·PRG
  A1    Priv              N     11(11.000000000)    B: B·PRG
  6  variables  defined,54  bytes  available
250  variables  available, 5946  bytes available
调用 B 程序后, A 程序的内存变量:
  X     Priv              N     1 (1.000000000)     B: A·PRG
  Y     Priv              N     20(20.000000000)    B: A·PRG
  Z     Priv              N     3 (3.000000000)     B: A·PRG
  A2    Pub               N     22(22.000000000)    B: A·  PRG
  4  variables  defined, 36 bytes used.
252  variables  available, 5964 bytes available.
```

从屏幕显示可以看出:本例中只有一个全局变量 A2; X、Y、Z 是 A 程序的局部变量, X、A1 是 B 程序的局部变量。B 程序可使用 A 程序的局部变量,反之则不行。因为 B 程序要使用与 A 程序中的局部变量 X 同名的局部变量,所以在 B 程序中先把 X 定义为隐蔽局部变量,使得 A 程序中 X 的值得到保护。

在 dBASE Ⅲ 中,内存变量可以以文件的形式存储和调用。