

微机快速入门丛书

FOXBASE+ 三周通

文忠 等 编著

机械工业出版社

微机快速入门丛书

FoxBASE+ 三周通

文忠 等 编著



机械工业出版社

(京)新登字 054 号

本书详细、全面地介绍了 FoxBASE+ 数据库管理系统。作者从初学者的角度出发,合理地安排了全部内容,使读者在三周内掌握有关知识。全书内容包括:数据库的基本概念和 FoxBASE+ 的概况;数据库的基本操作;获取信息的方法;建立数据库的高级方法;系统参数的设置;内存变量的建立和使用;常用命令和函数;学习写程序;报表格式文件、标签格式文件和屏幕格式文件;命令和函数总论;菜单设计;实用程序介绍;常见错误的处理;系统保护;数据库管理系统的实例;高级程序设计技巧等。书中还附有小结和练习,以便于读者巩固所学的知识。

本书语言精炼,结构严谨,适合广大初学者自学,又能为熟悉系统的用户提供很好的帮助。

图书在版编目 (CIP) 数据

FoxBASE+ 三周通/文忠等编著. —北京:机械工业出版社, 1995
(微机快速入门丛书)

ISBN 7-111-04476-2

I. F... II. 文... III. 数据库管理系统, FoxBASE+ IV. TP311. 13

中国版本图书馆 CIP 数据核字 (94) 第 12412 号

出版人 马九荣 (北京市百万庄南街 1 号 邮政编码 100037)

责任编辑:王中玉 王福俭 版式设计:霍永明 责任校对:肖新民

封面设计:肖 晴 责任印制:卢子祥

三河永和印刷有限公司印刷·新华书店北京发行所发行

1995 年 2 月第 1 版·1995 年 2 月第 1 次印刷

787mm×1092mm¹/₁₆·24.75 印张·601 千字

0 001—3 000 册

定价:29.00 元

出版者的话

微型计算机（简称微机）的应用已广泛地深入人们的物质生活和文化生活，极大地促进了信息社会的发展。随着微机应用的普及，它已不仅仅是人们从事生产活动的现代化工具，而且也是人们生活中不可缺少的好助手。它不仅可以从事工程计算、数据处理、信息管理、数据通信，还可以帮助学生学习和娱乐，帮助作家撰稿、作曲、画图，为人们理财、档案管理乃至治病……。由于它的用途广泛，可以大大提高人们工作、学习的效率，因此越来越多的人开始学习、使用它。为了帮助广大初学者学习微机的基础知识，掌握微机的操作方法，学会使用日常工作和生活中经常用到的软件，我社特组织出版一套系列丛书——微机快速入门丛书。

我们编写这套丛书的目的是为了提高初学者的学习效率，使读者用较短的时间学会操作微机的方法，掌握微机软件的使用。因此，我们邀请一些有教学经验的老师和长期从事软件开发的程序设计人员，在总结经验的基础上撰稿。

由于本丛书是奉献给广大初学者的，因此在写作上注意了由浅入深、通俗易懂，以比较生动的语言来讲解枯燥的技术内容，或者是用实例帮助读者理解，供读者练习，巩固和消化所学知识。

本丛书即将出版的有《外行跟我学电脑》、《最新汉字录入方法与文字处理技术》、《FoxBASE+三周通》等，我社将根据微机及各种软件的升级和更新，不断推陈出新，根据读者的实际需求，陆续出版新书。但愿本丛书能成为广大读者工作和生活的好帮手。

机械工业出版社

前 言

FoxBASE+ 是 1986 年 6 月由美国 Fox 软件公司推出的数据库语言,已在数据库管理系统中显示出了强大的生命力。

由于 dBASE III 有很多固有的弱点,如计算能力较弱、处理速度较慢、无数组、无图形功能、无多用户功能等等。它们限制了 dBASE III 在更多应用领域的发展,难以满足综合信息管理等较高要求。近年来,不断有新的与 dBASE III 兼容的数据库软件推出,如 dBASE III plus、Clipper、Paradox、FoxBASE 等等,其中最为成功、性能最优良的首推 FoxBASE+。

FoxBASE+ 在运算速度上平均比 dBASE III 快 6~7 倍,比编译 dBASE III 快 2 倍左右,且与 dBASE III 和 dBASE III plus 完全兼容,用后两者语言编写的程序无需改动就可运行。不仅如此,新增加了数十个命令和函数,在性能和功能上作了许多扩充和发展。例如:内存变量数组、系统状态和参数的测试控制能力、键盘输入的管理、内存的动态管理、与汇编语言的接口、错误处理功能、过程文件组合、命令文件编译、多用户功能、数据编辑中的检验功能、直接访问 dBASE III 的数据库、数据库间的“一父多子”关系、多个数据工作区的同时操作等等,且具有良好的移植性;可在 DOS、XENIX 等多种操作系统支持下工作。

中西文 FoxBASE+ 是根据 FoxBASE+ 的最新多用户版本汉化开发而成的,保留了原版 FoxBASE+ 的全部性能和功能,适用于 IBM-PC/XT、AT、286、386 及其兼容机,既是一个多用户的关系数据库管理系统,可在网络软件支持下工作,又可以在单用户环境下运行。它不仅为人们提供了新的数据库应用环境和开发工具,又为原来用 dBASE III 或 dBASE III plus 开发的应用项目带来了数倍效率和面貌一新的前景,因此,一经问世,便引起国内各行各业用户的极大兴趣,流行之快,范围之广,无与伦比。

为了在我国更好地推广和普及 FoxBASE+,同时也为了读者使用方便,根据大量 FoxBASE+ 的资料,编著了本书,奉献给广大读者。

本书从实用的角度出发,做了新的尝试,首次采用三周通的形式,将 FoxBASE+ 整个系统化整为零。FoxBASE+ 系统的内容非常多,要想在很短的时间内全部掌握是非常困难的,作者为了方便读者,花费了相当大的精力和时间,精心编排,力求内容丰富,介绍详细,分隔合理,语言文字准确,并配有大量实例,使得没有计算机程序设计基础的读者也能很容易地掌握本书的内容,成为一个熟悉的 FoxBASE+ 系统的专家。

参加本书编写的人员有:文忠、刘明、葛世达、赖巧玲、王伟国、李山、山昌祝、李凤云、杨荣、郭世民、章理。

本书不但可作为广大初学者的学习教材,也可作为计算机用户、计算机工作者的参考资料。

由于时间仓促,书中难免有疏漏和不当之处,敬请读者批评指正。

目 录

第一天 数据库的基本概念和

FoxBASE+ 概况	1
一、基本概念	1
1. 数据库	1
2. 数据工作区	2
3. 数据库别名	2
4. 记录指针	3
5. 数据类型	3
6. 数据种类	4
7. 常量	4
8. 变量	5
9. 运算符	6
10. 表达式	8
11. 函数	8
12. 命令	8
13. 程序	9
14. 文件	10
15. 交互命令状态	11
16. 编辑和全屏编辑	11
17. 菜单和用户定义菜单	12
18. 系统配置文件	12
二、运行环境	13
三、主要性能指标	13
四、启动和退出	13
五、FoxBASE+ 语言成分	14
1. 命令	14
2. 文件	15
3. 变量	16
4. 表达式	17
六、直接建立数据库	19
七、录入数据	23
第二天 数据库的基本操作	26
一、修改数据库结构	26
二、修改数据库记录的内容	30
1. 按序号进行屏幕修改 EDIT	30

2. 修改指定记录 GO/GOTO, EDIT	31
3. 按条件进行修改 CHANGE	33
4. 多记录修改 BROWSE	33
5. 批替换 REPLACE	35
6. 记录插入 INSERT	35
7. 记录删除 DELETE, RECALL, PACK, ZAP	37
第三天 如何获得所需要的信息	41
一、分类	41
二、建立索引文件	43
三、快速检索	49
四、顺序查找	52
第四天 建立数据库的高级方法	56
一、复制数据库结构和内容	56
二、数据库文件间的关联	61
三、数据库文件间的更新	64
四、数据库文件间的连接	66
五、数据库文件间的记录追加	68
第五天 SET 命令的使用	69
一、SET	69
二、SET <参数> ON/OFF	72
三、SET <参数> TO <参数值>	73
四、SET <参数> <参数值>	74
第六天 内存变量的建立和使用	75
一、内存变量的建立	75
1. 赋值命令	75
2. 输入数据的命令(一)	76
3. 输入数据的命令(二)	76
4. 等待命令	77
二、数据类型和运算符	78
三、内存变量的作用域及清除	79
1. 内存变量的类型	79
2. 内存变量的显示和清除	80
3. 内存变量的保存和恢复	80
四、数组	81

1. 数组的定义	81	三、标签格式文件	138
2. 数组的使用	82	四、与打印有关的命令和函数	142
3. 文件数据复制成数组数据	83	五、自定义格式文件	144
4. 数组数据复制成文件数据	83	第十一天 FoxBASE+命令和函数	
第七天 小结与练习	86	总论	151
一、小结	86	一、FoxBASE+ 命令总论	151
二、练习	86	二、FoxBASE+ 函数总论	205
第八天 常用命令和函数	88	第十二天 学习写程序(二)	219
一、常用命令	88	一、程序的分支结构	219
1. 磁盘文件操作命令	88	1. 简单选择语句 IF...ELSE	
2. 文件复制命令	89	...ENDIF	219
3. 文件删除命令	89	2. 多重选择语句 DO...CASE	
4. 文件更名命令	89	...ENDCASE	222
5. 文本文件显示	90	二、循环结构	225
6. 清除命令	90	三、过程及参数传递	229
7. 状态显示命令	91	1. 过程	229
8. 键盘填充命令	91	2. 过程文件	231
9. 存储命令	91	3. 参数传递	233
10. 退出命令	91	四、调用外部程序或命令	234
二、函数	91	1. 执行外部程序	234
1. 数学运算函数	92	2. 调用二进制程序文件	234
2. 字符函数	95	第十三天 菜单设计	238
3. 日期函数	100	一、设计显示菜单	238
4. 转换函数	103	二、系统管理	238
5. 返回逻辑型值的测试函数	106	1. SET 命令详解	238
6. 返回数字型值的测试函数	109	2. 系统配置文件	254
7. 返回字符型值的测试函数	112	三、色彩变化	259
8. 输入函数	115	第十四天 小结与练习	262
第九天 学习写程序(一)	118	一、小结	262
一、程序文件的建立、修改、执行	118	二、练习	262
1. 建立程序文件	119	第十五天 实用程序介绍	264
2. 修改程序文件	119	一、过程文件生成器 FOXBIND	264
3. 程序执行	120	二、程序文件的编译	269
二、编写程序实现简单的数据库操作	120	第十六天 常见错误的处理	272
三、程序注释	123	一、错误捕获和处理	272
第十天 报表格式文件、标签格式		二、程序调试	274
文件和屏幕格式文件	125	三、常见错误举例	277
一、@命令	125	四、出错报告和出错信息	285
1. 用于输入输出	125	五、出错信息和出错号码	287
2. 用于清屏	130	第十七天 设置口令字的方法	300
3. 用于画框	131	第十八天 人员档案管理系统的	
4. 用于菜单设计	131	实例	311
二、报表格式文件	133		

第十九天 程序设计技术精益求精	374	三、程序文档	380
第二十天 高级程序设计技巧	378	四、程序维护	381
一、程序设计	378	第二十一天 结束语	383
二、程序检验	379	参考文献	384

第一天 数据库的基本概念和 FoxBASE+ 概况

概要:了解数据库的一些基本概念、运行环境和性能指标;掌握如何建立 FoxBASE+ 的数据库并录入数据。

一、基本概念

FoxBASE+ 是新颖的关系数据库管理系统,它的职能是接受并执行用户通过应用程序或键盘提出的命令,完成各类数据处理和数据库维护功能。FoxBASE+ 数据库系统的基本概念有:数据库、数据工作区、数据库别名、记录指针、数据类型、数据种类、常量、变量、运算符、表达式、函数、命令、程序文件、过程文件、交互命令状态、全屏幕编辑、用户定义菜单、系统配置文件,等等。

1. 数据库

数据库(Database)是结构化的数据集合。它是根据数据的性质、内在的联系,按照对所有相关数据实行统一的、集中的、独立的的管理的要求来设计和组织的,其数据的一致性、共享性都较好,冗余度小,并且独立于程序,易维护、易扩充。

目前流行的数据库有关系型、层次型、网络型等三种。关系型数据库与层次型、网络型的主要区别是它的数据描述的一致性,它把每个数据子集都分别按同一方法描述为一个关系,不像后二者要选规定子集之间的从属或层次或先后顺序,而是让子集之间彼此独立。在使用时,通过过滤、投影、联接等方法,使数据之间或子集之间按某种关系进行操作,因此关系数据库的数据表示能力较强,初学者好理解、易接受,使用上也较方便,并且容易实现通用的数据管理功能。

关系型数据库(Relational Database)中每一个关系(即数据子集)都是一个二维表,其形式好比我们日常生活中的各种报表。如:学习成绩表、产品数量表、供求关系表、工资表等。在表中的每一行放的是一条由若干项组成的数据,它对应数据库中的一条记录(Record);表中的每一列都是同一项目的数据,它对应数据库中的一个字段(Field);项目的名称称为字段名;该项数据所占格数(字节数)称为字段的宽度,根据它是数字、字符、日期等确定其类型,称为字段的类型;表的长度(即行数)对应数据库的记录个数;两个同样项目的表,数据可以有多有少,即记录的个数随数据多少而改变,但它的项目却是相同的,也就是字段是相同的且字段的个数也相同,即字段相对固定。决定了一个表,也就定了一个数据库。一个二维表用字段个数(项目数)、字段名(项目名称)、字段类型(项目类型)、字段宽度(项目所占格数)、记录个数(数据个数)等要素便完全决定了。在 FoxBASE+ 中,这些二维表以文件的形式存储在磁盘上,一个文件就是一个关系。这样的文件称为数据库文件(Database file),有时也简称为数据库(Database)。

比如,我们有如下的一张表:

姓 名	性别	出生年月	家庭住址	电话
张 明	男	1962 年 5 月	北京市海淀区	6543218
李 佳	女	1970 年 1 月	北京市朝阳区	2549231
刘志刚	男	1969 年 4 月	北京市崇文区	4231629
何 玲	女	1971 年 8 月	北京市宣武区	2596721

画一个表时要确定该表由多少项目组成,每个项目都要填何种数据,这些是表的框架,也可称为结构;再确定这个表有多少格,即数据的多少。构造一个数据库也是如此,要确定数据库中含有多少个字段(字段数)、字段名、字段类型和字段宽度,如果带有小数时,要确定小数有几位,这些都是数据库的结构部分;有多少数据构成数据库的数据部分,即数据库的记录。

因此,数据库由两部分组成,一是结构定义部分,二是数据部分。结构定义要说明数据的存放形式、存放顺序,这其中要详细说明的有数据库文件名、构成数据库的字段、字段名称、字段类型、字段宽度和小数位位数。数据部分是数据库文件要保存的内容,是按照结构定义部分的说明进行有序存放的。数据库由一个个的记录构成,记录由若干字段构成。

2. 数据工作区

数据库文件要先打开再使用。在 FoxBASE+ 中,可以同时打开和使用多个数据库文件,为使这些同时使用的文件不互相干扰,便于识别,FoxBASE+ 采用了将不同的数据库分别放在不同的区域中的方法来将它们分隔开。这些不同的区域就称为 FoxBASE+ 的数据工作区(Work Area),共有 10 个。在一个数据工作区中,只允许打开一个数据库文件,这样,在 FoxBASE+ 中能同时使用的数据库文件最多为 10 个。

数据工作区是用编号或名称来标识和区别的,编号分别为 1~10,名称分别为 A~J 的单个字母,在后面我们还会看到数据库的别名,这也是一种标识。

在 FoxBASE+ 中用户虽然可同时使用多达 10 个数据工作区或打开 10 个数据库文件,但为管理和使用的方便,在任一时刻只有一个工作区是主要的,称之为当前工作区(Currently Selected Work Area),如不特别声明,一切操作均是指当前工作区。当前工作区中的数据库文件称为当前数据库文件(Currently Selected Database)。

如果需要同时使用其它工作区中的数据库文件的数据,必须在该数据前冠以其所在工作区的编号或名称或别名(下面将要介绍)。

FoxBASE+ 在刚进入时默认 1 号工作区为当前工作区,可使用工作区选择命令(SELECT)来指定其他工作区为当前工作区。

3. 数据库别名

FoxBASE+ 对非当前工作区的数据进行操作时,虽可使用该工作区的名称来标识,但由于工作区的名称是固定的,因而不利于数据与操作功能的相对独立。因此,FoxBASE+ 让每个已打开的数据库文件都有一个别名(Alias),用户可用别名代替工作区名来标识工作区中的数据库。别名的建立可由用户在打开数据库文件的同时任意指定一个字符,不但灵活,而且与所在工作区没有必然的联系,因此容易实现数据与操作功能的独立。另外,由于别名与数据库文件名的联系,用户不需专门记忆要用的数据库是在哪个工作区打开的,只需调用别名就行了。如:某个数据库文件的别名为 SSS,无论它在哪个工作区打开,使用 SELECT SSS 就能选择该文件所在的工作区为当前工作区。

数据库的别名,由一个不超过 8 个字符的不含空格的字符串组成。在数据库文件打开时由

用户命名,如果用户不命名,则 FoxBASE+ 自动把该数据库的文件名(不包括该文件的扩展名)指定为它的别名。

4. 记录指针

在一个数据库中,一般都有多个记录,FoxBASE+ 进行数据库操作时,虽然可对所有记录或其中一部分记录进行连续处理,但在处理过程中总是逐个记录进行的。这样,对于一个已打开的数据库文件来说,在任一时刻,总是只有一个记录被当作当前数据处理的对象,它或者是正在被处理或者是正等待处理,这个记录被称作当前记录(Current Record)。FoxBASE+ 用于标识当前记录的一个指针就被称作是记录指针。在每个数据工作区 FoxBASE+ 都设置了一个记录指针(Record Pointer),以此来标识和区别各工作区中的当前记录。记录指针中存放的是该记录在数据库文件中的序号,这个序号是在数据库建立时决定的,是录入时的顺序号,叫记录号,也可以称之为绝对记录号(物理顺序号)。如无特殊声明,本书中所指的记录号都是指的记录在数据库中的真正的位置。在某个工作区中没有数据库文件打开时,该工作区中记录指针的值为 0;如果有数据库文件打开,则记录指针的值就是当前记录的记录号。

FoxBASE+ 对所有记录进行的操作是通过移动记录指针而实现的,因此,有专门移动记录指针的命令和函数。我们定义记录指针移动的方向为:向前移动表示记录号增大,向后移动表示记录号减少。当记录指针向前移动,超过数据库文件中最后一个记录,指向文件尾(End of File)时,记录指针的值比数据库文件中的最大记录号(等于数据库文件中记录的个数)大 1;当记录指针向后移动,移过数据库文件中最前的一个记录(首记录),指向文件头(Beginning of File)时,它的值仍为 1。

在一个数据库文件中,记录指针可以任意移动;还可以通过关联,使一个数据库为父辈数据库,几个数据库为子辈数据库,且当父辈数据库的记录指针移动时,各个子辈数据库的记录指针同时作相应的移动。

5. 数据类型

在各种计算机语言中,为了便于管理和应用,都规定了参加运算的两个数据的类型必须是相同的,至少也是相容的(即可以把一种类型强制或自动转化成另一种类型),FoxBASE+ 也不例外。相同类型的数据之间才能进行各种操作,这是基本原则。因此,数据类型(Data Type)是一个非常重要的基本概念。

在 FoxBASE+ 中,数据类型有:数字型、字符型、日期型、逻辑型、备注型、屏幕型等六种。它们的构成和使用规则如下:

(1) 数字型(Numeric) 类型代号为 N。只能由阿拉伯数字、小数点和正负号组成,可以进行各种数学运算,最大精度为 16 位有效数字。

(2) 字符型(Character) 类型代号为 C。由所有可打印的文字或符号组成。不能参加数学运算,但可用“+”或“-”号将几个字符型数据连接起来;还可以按其字符的 ASCII 码的值进行大小比较。在一个字符型数据中含有的字符个数称为该字符型数据的宽度。一个字符型数据的长度最大是 254 个字节。

(3) 日期型(Date) 类型代号为 D。日期型用于存放表示日期的数据。为便于统一处理,系统规定了 8 种日期格式,可由用户自己选定。如果用户不指定日期格式,系统默认的日期型数据的格式为“月/日/年”(MM/DD/YY),长度固定为 8 个字符。两个日期型数据可以进行大小比较,日期在前者(离公元一年近)为小,日期在后者(离公元一年远)为大;两个日期型数据

可进行减法运算,结果是两个日期型数据所表示的日期相差的天数,为数字型;一个日期型数据可以加或减一个数字型数据,结果是日期型。

(4) 逻辑型(Logical) 类型代号为 L。逻辑型数据只有两个值:逻辑真和逻辑假。它的长度固定为 1。逻辑真可用 .T.、.t.、.Y.、.y. 表示,逻辑假可用 .F.、.f.、.N.、.n. 表示,它们之间只能进行逻辑运算(布尔运算)。

(5) 备注型(Memo) 类型代号为 M。FoxBASE+ 规定了一个字符型数据的长度最大为 254 个字符,但在实际情况中往往需要在一个数据中保存更多的信息,所以,为解决这一问题, FoxBASE+ 增加了一个辅助文件(称为备注文件,扩展名为 .DBT),专门用来存放这些长度较大而又不等长的字符信息块,而在数据库中只存放这些信息块的地址指针。这样,该数据的类型就称为备注型,其长度固定为 10。备注型的数据可以被编辑、显示、打印,但不能参加任何类型的运算。

(6) 屏幕型(Screen) 类型代号为 S。用来存放屏幕信息。在 FoxBASE+ 中,屏幕型只在屏幕的字符方式下使用,它存放的是整幅屏幕的信息。一幅屏幕的信息包括 25×80 个字符,每个字符还带有一个字节的颜色指示码,这样一个屏幕型数据需占内存 4KB。屏幕型数据除了存放或显示屏幕信息外,不能进行其它任何类型的操作或运算。

除上述 6 种数据类型外,还有一种 A 型(Array),用来标识数组。但 A 型不是一种独立的数据类型,因为数组中的各个单元可以各有自己的类型,且可以互不相同。

在 FoxBASE+ 中的数据类型也是可以进行转换的。通过类型转换,可以使不同类型的数据能够进行运算或操作。主要的类型转换有:字符型与数字型之间的相互转换(这要求字符型数据只能由数字字符、正负号组成);日期型与字符型之间的相互转换(这要求字符型数据的格式符合日期型数据的格式要求)。进行数据转换的目的不仅仅是为了进行运算或操作,有的是为了进行显示或打印的需要。如:打印或显示一个数字型数据,要想控制它所占的宽度,就必须将它转换成字符型,同时定义它所占的宽度,以形成美观的格式、整齐统一。

6. 数据种类

FoxBASE+ 的数据有常量、变量、函数,共三种。

常量(Constant)是固定不变的数据;变量(Variable)是其内容可能要变化的数据;函数(Function)是一种由一些数据根据某种规则运算或操作后得出的结果数据。

7. 常量

FoxBASE+ 的常量可以是数字型、字符型、日期型、逻辑型。

(1) 数字型常量 可以是整数或小数,不能用分数,可用科学记数法,如 -9, +3, 123.4567, 1.56789E2 等都是正确的常量。

(2) 字符型常量 用定界符(指单引号、双引号或方括号)括起来的字符数据,又称为字符串(Character String)。定义字符串的定界符必须成对匹配,如“数据库管理系统”、“新颖的关系数据库系统 FoxBASE+”、[QWERT]等。如果某种定界符本身也是字符串内容的一部分,则需用另一种定界符括起来。如:‘这是一个特定的字符串——[数据库]’。’ ’或” ”也是字符型常量,它表示长度为零的空字符串(Null String)。

(3) 日期型常量 是用于表示日期的常量。在 FoxBASE+ 系统中无法直接表示日期型常量,一般是通过转换函数 CTOD()实现。该函数是将符合日期格式的字符串转换成日期型数据,如 CTOD('12/30/93')表示 1993 年 12 月 30 日。

(4) 逻辑型常量 只有两个值:真或假。

8. 变量

FoxBASE+ 的变量有 2 种:字段变量(Field variable)、内存变量(Memory variable)。

字段变量就是数据库的字段;内存变量又分标量变量(Scalar variable)和数组变量(Array Variable)。变量的要素是变量名称、类型、长度(宽度)、作用域、保存方式等。变量的名称可以由字母、汉字、数字和下划线组成,必须以字母或汉字开头,名称的长度最多不能超过 10 个字符。

(1) 字段变量 字段变量的类型有数字型、字符型、日期型、逻辑型、备注型。

1) 数字型 整数或小数,长度为 1~19 位。如是小数,小数点也占一位,小数位数为 1~15 位,并且至少应比字段长度少 2 位,数字型字段的初始值为 0。

2) 字符型 长度为 1~254 字节。字符型字段的初始值是与字符型字段所定义宽度等长的由空格组成的字符串。

3) 日期型 长度固定为 8 字节。初始值不表示任何日期,显示形式为:

“ / / ”。

4) 逻辑型 长度固定为 1 字节,初始值为.F.。

5) 备注型 长度固定为 10 字节。其实际内容存放在备注文件(.DBT)中,在字段中存放的是该内容的地址,在显示时仅显示“Memo”。

字段变量只存在于数据库中,是在建立数据库文件时产生的。一个数据库文件中字段变量的个数最多为 255 个,且它们的总长度不得超过 4000 字节。

字段变量的保存形式就是数据库文件,故它的作用域也就随文件的打开而建立,随文件的关闭而撤消。在一般情况下,数据操作都是针对当前选择的数据工作区中的字段进行的,因此,数据工作区也可看作是字段变量的作用域。

对于一个数据库文件中的某一字段,该数据库文件有多少条记录,它就有多个值,也就是说字段变量是一个多值变量,但它又相对固定。这是因为,一旦数据库不再改动时,则对应哪条记录的字段的值就被确定了。字段变量的值随记录的不同而不同,只有记录指针当前所指的记录的字段内容才是它的当前值。

也可以调用非当前工作区中的字段,但要在这些字段名前冠以所在工作区的名称或数据库别名。

(2) 内存变量 内存变量是一种临时信息存储单元,它独立于数据库文件而存在,使用时可随时建立,也可随时释放。共有四种类型:数字型、字符型、日期型、逻辑型。屏幕型的内存变量我们不做考虑。

1) 数字型 存储数字型数据,即整数或小数,数值精度为 16 位有效数字。

2) 字符型 存储可打印字符组成的串,长度最大 254 字节。

3) 日期型 存储日期型数据,长度固定为 8 字节。

4) 逻辑型 存储逻辑型数据,长度固定为 1 字节。

屏幕型为存储当前整幅屏幕信息,用得较少。

内存变量中的标量一般是通过各种赋值命令建立的,这些命令同时完成变量的建立和赋值,且自动给变量赋予类型,如:STORE 命令,由值的类型决定内存变量的类型;ACCEPT 命令,所建立的全部变量的类型都是字符型;INPUT 命令,通过键盘输入建立内存变量,内存变

量的类型依所输入的数据类型而定。

内存变量中的数组只能是一维或二维的,一般通过数组定义命令(DIMENSION)来建立,其初始值为逻辑假。通过各种赋值命令可对整个数组赋同一个值,也可分别进行赋值,且类型可任意。

内存变量的作用域有两种:全局内存变量和局部内存变量。全局内存变量通过命令 PUBLIC 来说明并建立,除非删除它,否则始终存在并起作用,任何程序都可对它进行调用并改变它的值;局部内存变量的作用域仅限于建立它的那个程序(包括被该程序调用的程序)内有效,退出该程序,自行释放(删除)。

内存变量的保存形式是内存变量文件,可将变量存入文件,也可从该文件中再恢复出来。

内存变量的个数默认为 256 个,用户可通过系统配置文件 CONFIG. FX 中的 MV-COUNT 配置项,将内存变量的个数扩大至 3600 个。一个数组型的内存变量的单元个数也可达 3600 个,且仅占一个内存变量的名额。所有字符型变量的总长度不得超过 6KB(可配置为最多 31KB),其它各类变量的总长度只受当前内存的最大可用空间的限制。

(3) 字段变量和内存变量的关系 字段变量和内存变量只要类型相同,可以互相传递,并可参与相应类型的数据操作。若类型不同,如能通过类型转换函数将其中之一的类型转换成另一个类型,则仍可进行传递或运算。

数据传递的方式有以下几种:

- 1) 字段变量传递给标量变量;
- 2) 标量变量传递给字段变量;
- 3) 字段变量传递给数组变量;
- 4) 数组变量传递给字段变量。

字段与数组相互传递时,一个字段对应数组中的一个单元,一个记录对应一个一维数组。有关相互传递的命令在后面有详细论述。

字段变量和内存变量的不同之处,主要表现在以下几点:

- 1) 字段变量隶属于数据库文件,不能独立存在;内存变量独立于数据库,自由度较大。
- 2) 字段变量的保存形式是数据库文件,内存变量的保存形式是内存变量文件。但在保存时,前者采用的 ASCII 码符,后者采用的是 IEEE 长实数(数字型、日期型)和一字节的二进制(逻辑型)。
- 3) 字段变量的类型和长度是相对固定的,除非改动数据库文件结构,否则无法改变;内存变量的类型和长度随所赋数据的类型和长度而定,只要重新赋值,便可予以改变。
- 4) 字段变量是一种多值变量,即使不重新赋值,只要记录指针一移动,它的当前值就会随之改变;内存变量是一种单值变量,只要不重新赋值,它的当前值就保持不变。
- 5) 在处理的顺序上,字段变量比内存变量优先,即两者如果同名时,系统只认字段变量,而不认内存变量,为区别同名的内存变量与字段变量,应在内存变量名之前冠以“M->”。

关于内存变量在后面还有更详细的讨论和介绍。如无特殊声明,将标量变量和数组变量统称为内存变量。

9. 运算符

运算符(Operator)是表示数据之间运算方式的符号。FoxBASE+ 的运算方式有 4 种:数学运算、字符串运算、关系运算、逻辑运算。因此,运算符也有 4 种:数学运算符、字符串运算符、

关系运算符、逻辑运算符,各运算方式和运算说明如下:

(1) 数学运算(Mathematical Operation):数学运算的对象是数字型数据,运算的结果仍是数字型数据。数学运算符按其运算优先级从高到低排列如下:

()	括号		
+	正号	-	负号
^	乘方		
*	乘	/	除
+	加	-	减

(2) 字符串运算(String Operation):字符串运算是指把两个字符型数据连接起来,形成一个新的字符串。字符串运算符有两种:

- ④ 两字符串相接
- ① 两字符串相接,并把前一个字符串尾部空格移到新字符串的尾部。

(3) 关系运算(Relational Operation):关系运算是指比较两个数据的大小或前后。关系运算只能在数字型数据之间、字符型数据之间、日期型数据之间进行。

※数字型数据之间:按数值的大小进行比较;

※字符型数据之间:按字符的 ASCII 码的大小进行比较。

汉字进行比较时,按其内码的 ASCII 码值进行。从外观上看,一级字库按其拼音字母的排列顺序,二级字库按其笔划(首尾码)的顺序进行。

字符型数据的比较还分为精确匹配和不精确匹配。在要求精确匹配时,参加比较的两个字符串必须完全相等;不精确匹配时,只要等号前的字符串包含(从第一个字符开始)等号后面的字符串就行了。如:“ABCDE”与“AB”,在精确匹配时不相等;在不精确匹配时,可以认为是相等的。

字符型数据之间还有一种关系运算是子串比较,即在一个字符串中检测另一个字符串是否存在,完全包含的关系。

※日期型数据之间:日期在前的为小,日期在后的为大。

关系运算后的结果是逻辑型数据,只有两种值:逻辑真或逻辑假。关系运算符有如下几种:

=	等于	# 或 <>	不等于
>	大于	<	小于
>=	大于等于	<=	小于等于
==	字符串精确匹配	\$	子字符串比较

(4) 逻辑运算(Logical Operation):逻辑运算只能对逻辑型数据进行,它的结果依然是逻辑型数据。逻辑运算符按其运算的优先级别从高到低排列如下:

()	括号
! 或 .NOT.	逻辑非
.AND.	逻辑与
.OR.	逻辑或

当有多种运算符出现在一起时,括号优先级最高(括号无大、中、小括号之分,一律用小括号),最内层的括号最优先;其余各运算符的优先级从高到低排列如下(相同优先级的运算,按

从左到右的顺序进行):

数字运算符或字符运算符

关系运算符

逻辑运算符

10. 表达式

表达式(Expression)是数据之间运算关系的表达式形式,它可由常量、字段变量、内存变量、函数等数据和运算符组成。作为特例,单个数据也是表达式。

表达式经过运算,总是得到一个具体的结果值,因此表达式也是数据,也有类型、也可以参加各种运算和操作。它的数据类型有:数字型、字符型、日期型、逻辑型等4种。表达式一般是按其值的数据类型分类的,这样 FoxBASE+ 的表达式有:数字型表达式(Numeric Expression)、字符型表达式(Character Expression)、日期型表达式(Date Expression)、逻辑型表达式(Logical Expression)等四种类型。逻辑表达式因为经常用来作为数据检索或程序分支的条件,因此又称为条件(Condition)。

表达式在 FoxBASE+ 的各类数据操作和运算中,起着非常重要的作用。它不仅常常用于规定各类数据操作的对象,而且用于限制数据处理的范围、定义数据过滤的条件、设计数据输出的格式、设置系统状态和参数,有时可用一个表达式来描述一个复杂的数据管理事务或过程。因此,正确合理地应用表达式是非常重要的。用得好,可以收到事半功倍的效果;用得不好,也可能会事倍功半。

11. 函数

函数也是一种数据,且这种数据的值是由某些数据通过某种特定的操作后得到的。因此,函数一方面是一种不同于常量和变量的特殊数据,另一方面它又代表了一些特定的数据操作功能,每取一次函数的值就是进行一次数据操作。FoxBASE+ 中有近 100 个函数,也就有近 100 种特殊功能。

函数是数据,当然也可用在表达式中。每个函数的值都有固定的类型,我们在介绍函数时,对它们的类型都作了介绍。

函数同时也是一种操作功能,所以需要操作数。操作数的情况分为三种:1)函数的操作数由系统规定;2)由用户给出;3)由用户指定,也可默认。对于 2)、3)这两种情况,其操作数又称为自变量(Argument)。每个函数对其自变量的个数、类型、值域都有一定的规定,在进行函数调用时,必须遵守这些规定。在自变量中可以使用表达式,而表达式中也可以含有函数,这就实现了函数的嵌套调用。

在 FoxBASE+ 的函数中按功能或用途可分为 6 类:数字运算、日期时间、字符串操作、类型转换、数据库函数、测试函数。除此之外,FoxBASE+ 还允许用户自己定义函数,称为用户定义函数(User Defined Function,有时缩写为 UDF)。

无论是 FoxBASE+ 的函数或用户自己定义的函数,它们的表示形式都是:在函数名称后跟一对圆括号,圆括号内给出函数的自变量。

12. 命令

命令是 FoxBASE+ 用以完成各种数据库操作、程序控制、系统状态或参数的设置等功能的指令,它是 FoxBASE+ 的重要组成部分。在 FoxBASE+ 中,几乎所有的功能都是由命令(包括通过命令调用各种函数)来完成的。

命令一般由命令动词(Command Verb)和限定该命令动词的若干选择项来组成的。命令动词规定该命令所要完成的“动作”,选择项说明该“动作”的对象、范围、条件。一般采用英文术语表示。

命令可以由用户直接从键盘输入执行,也可以通过编程来执行。

FoxBASE+ 的命令有近 180 条,按其用途可分为如下 15 类:

- (1) 文件建立与编辑的命令
- (2) 文件打开和关闭的命令
- (3) 文件其它操作的命令
- (4) 数据库关系操作的命令
- (5) 数据库记录检索的命令
- (6) 数据库记录增加的命令
- (7) 数据库记录修改的命令
- (8) 数据库记录删除的命令
- (9) 数据计算的命令
- (10) 数据输出(显示或打印)的命令
- (11) 用于内存变量的操作的命令
- (12) 用于键盘管理的命令
- (13) 用于程序设计的命令
- (14) 用于调试的命令
- (15) 用于设置系统状态或参数的命令

13. 程序

FoxBASE+ 的命令可通过键盘直接输入并执行,也可编成一个命令文件(Command file)来执行,这个文件就是程序(Program)文件,是一个 ASCII 文本文件,可用 FoxBASE+ 的文本编辑命令(MODIFY COMMAND)来编辑,也可用任意其他的字处理软件来编辑。

FoxBASE+ 的程序是解释执行的。它的解释机制除了像一般的解释方法那样解释一句执行一句外,还有一个重要的特点:解释时先把命令编译成可执行代码再执行,且这些代码执行后不消除,仍然保留在内存的程序缓冲区(Compiled Program Buffer)里,如果要重复执行该程序段,FoxBASE+ 就不再回到磁盘上搜寻、装载、解释该程序,而是执行已有的代码,因而大大提高了程序运行速度。这是 FoxBASE+ 比其他数据库软件快几倍的原因之一。

FoxBASE+ 的程序可以进行准编译,编译后生成的目标程序虽然不能独立运行,还得由 FoxBASE+ 执行,但具有程序装载快、内容可保密、长度减小等优点。

FoxBASE+ 是一种结构化的数据库编程语言,没有无条件转移语句,也不使用语句标号,它的程序流向是用逻辑结构来控制的。FoxBASE+ 的基本程序逻辑结构有三种:顺序结构、分支结构(单分支和多分支)、循环结构。

FoxBASE+ 的程序按用途和构造可分为四种程序:

- (1) 一般程序:一个命令文件中只有一个程序,且该程序不是屏幕格式程序。
- (2) 过程:一个命令文件中含有多个彼此独立、分别由标识符 PROCEDURE 开头的程序。该文件称为过程文件,文件中的各个程序分别称为过程。
- (3) 屏幕格式程序:定义全屏幕编辑时屏幕格式的程序,称为屏幕格式程序,又称为屏幕