# 第一章　数据库的基本概念

## 1.1　数据库管理系统

### 1.1.1　数据库及数据库管理系统

数据库(Data Base）是计算机存储设备上合理存放的相互关联的数据集合。它们具有如下特点：

(1)数据库中的数据无不必要的冗余,即尽可能不重复。

(2）数据共享，即数据库中的数据以最优的方式服务于一个或多个应用程序。

(3)数据的存储独立于使用它的应用程序。

(4)用一个软件统一管理这些数据,例如增加、修改、删除和检索数据。

数据库的结构分为层次型、网状型和关系型三种。FoxBASE ⁺系统建立的数据库为关系型数据库。

数据库管理系统(DBMS-DateBase Management System),顾名思义,即是管理数据库的系统。它具有定义、建立、管理和维护数据库的功能，具有与操作系统联机处理的能力。它由如下三部分组成：

1. 数据描述语言及其翻译模块；
2. 数据操作语言及其编译模块；
3. 数据库管理例行程序。

需要全本请访问 www.ertongbook.com

根据不同需要已研制出 dBASE、FoxBASE、FoxPro、ORACLE 等许多适于微机使用的数据库管理系统。其中，dBASE 系统已过时陶汰，FoxBASE 系统已发展许多版本，较新版本的 FoxBASE＋系统功能很强，又对计算机的软硬件支撑环境要求较低，是国内目前应用最广泛的数据库管理系统，并且是学习和使用 FoxPro 的基础。

利用数据库管理系统开发的数据库应用系统，或称为信息管理系统，已是各单位离不开的办公自动化系统。它具有数据的录入、修改、查询、加工和报表打印等各种操作功能。它由一个或多个数据库，以及管理数据库和加工信息的程序组成。因此，我们首先要了解数据库结构，学习管理数据库的知识，然后掌握设计与编写管理数据库和数据处理程序的方法。

在数据库应用系统中，每个数据库称为一个数据库文件，每个程序称为程序文件，它由 FoxBASE 命令组成，因此也称为命令文件。

### 1.1.2 FoxBASE＋系统及其启动与退出

#### 1.1.2.1 FoxBASE ＋系统组成

FoxBASE＋数据库系统主要由如下文件组成：

| | |
|---|---|
| MFOXPLUS.EXE | 主控程序 |
| MFOXPLUS.OVL | 覆盖程序 |
| FOXPCOMP.EXE | 编译程序 |
| FOXPHELP.HLP | 帮助程序 |
| FOXBIND.EXE | 过程文件生成器 |
| CONFIG.FX | 系统参数设置文件 |

其中，主控程序 MFOXPLUS.EXE 和覆盖程序 MFOXPLUS.OVL 是 FoxBASE＋系统的基本文件，只要有这两个文件，系统即可以正常运行。

主控程序 MFOXPLUS.EXE 是系统启动的核心文件。有时，为了操作方便，常常把该文件的主文件名简化为 FOXPLUS 或 FOX。覆盖程序 MFOXPLUS.OVL 是系统运行的必备文件,由它对系统中使用的各种命令进行解释。编译程序 FOXCOMP.EXE 用于编译命令文件，产生伪编译目标文件（扩展名为 .FOX)。帮助程序 FOXPHELP.HLP 提供各种命令和函数的语法规则和使用格式的说明。过程文件生成器能够自动生成过程文件,将在 7.7 节中介绍。用户可以在文件 CONFIG.FX 中设置系统参数,见第十章。只要具备 MFOXPLUS.EXE 和 MFOXPLUS.OVL 两个文件即可构成基本的 FoxBASE＋系统,完成开发数据库应用系统的任务。

1.1.2.2 FoxBASE+系统的启动与退出

将 FoxBASE ＋系统装入微机后,即可在 MS-DOS 系统下启动 FoxBASE＋系统。为了开发中文信息管理系统，应该在启动中文 CCDOS 系统后启动 FoxBASE$^+$ 系统。

启动 FoxBASE$^+$系统需要利用主控程序的主文件名，例如主控程序为 FOX.EXE 时，启动操作为：

C)FOX ↙

此时，DOS 运行主控程序 FOX.EXE，经暂短时间后，机器进入 FoxBASE+系统,转换屏幕后,给出 FoxBASE ＋系统屏幕,且在屏幕下方给出：

.—

圆点‘ . ’即 FoxBASE ＋系统提示符，其后闪动的小横线即为光标。此时即可执行 FoxBASE ＋的一切命令和操作。

当完成 FoxBASE＋系统的操作后，需要退出 FoxBASE+系统。退出命令和操作如下：

.QUIT↙

屏幕上立即转换成：

```
FoxBASE+ mormal shutdown
C>
```

表示 FoxBASE＋正常退出,并返回 DOS 系统。

## 1.2 数据库的结构

FoxBASE＋系统建立的关系型数据库是一个二维数据表，由若干行和列组成。例如，表 1.2.1 为一组学生的档案简表。

表 1.2.1 学生档案简表

| 学号 | 姓名 | 性别 | 出生日期 | 英语 | 计算机 | 体育达标 | 简历 |
|---|---|---|---|---|---|---|---|
| 01 | 王小明 | 男 | 11/23/75 | 86.0 | 75.0 | .T. | |
| 02 | 张兰莉 | 女 | 05/23/76 | 95.0 | 80.0 | .T. | |
| 03 | 何　欢 | 女 | 10/12/75 | 90.0 | 86.0 | .F. | |
| 04 | 王　默 | 男 | 07/30/74 | 63.0 | 47.0 | .T. | |
| 05 | 周利莎 | 女 | 08/11/73 | 45.0 | 63.0 | .T. | |
| 06 | 诸葛丰 | 男 | 02/18/75 | 78.0 | 48.0 | .T. | |
| 07 | 周伟兰 | 女 | 09/18/75 | 69.0 | 28.0 | .F. | |
| 08 | 辛　蔚 | 女 | 04/28/75 | 93.0 | 90.0 | .T. | |

在 FoxBASE＋系统中,将表 1.2.1 存储于计算机中，即建立一个数据库文件。

### 1.2.1 数据库文件名

任何一个数据库文件必须且只能有一个文件名，其格式如下：

```
××××××.DBF
↑      ↑
主文件名扩展名
```

主文件名：或称文件名，可由英文字母或汉字组成，最多不超过 8 个字符，每个汉字占两个字符位。在文件名中不得包含空格和 * ，等特殊符号。

扩展名："DBF"(Data Base File 的缩写）为扩展名。在书写和输入时，扩展名与主文件名之间必须用圆点“ . ”分隔。

文件的扩展名代表文件的类型，在 FoxBASE+系统中,扩展名“ DBF”代表数据库文件，其他类型的文件都有不同的扩展名，以后将逐步介绍。在 FoxBASE +系统中建立和调用数据库，可直接使用主文件名，省略扩展名。

例如，我们要建立一个如表 1.2.1 的学生档案库，可以使用 STUDENT.DBF 作为数据库文件名。在同一个系统中，至少在同一个子目录下，不同数据库不得重名。

### 1.2.2 字段(Field)

在表 1.2.1 中,数据从第二行起逐行排列,第一行为对应各列数据内容的说明,即学号、姓名、性别……等等,分别标记它们对应列数据的意义。在通常的数据表中,将它们叫做表头,或叫做栏目。在数据库中，把第一行中的诸项叫做字段，由它们标记对应列数据的意义及其属性，由以下四个参数描述：

1.字段名(Field Name)

每个字段必须有一个名字，它是给每列数据起的名字。因此，字段名的字面含义应与对应列数据的意义一致。

字段名必须以字母或汉字开头，其余可以是字母、汉字、数字或下划线，但结尾不能是下划线。一个字段名至多不能超过 10 个字符。我们建议使用英文字母或汉语拼音字母作为字段名，而不是汉字。因为这些字段名将在程序或操作命令中作为变量使用，若重复输入汉字将带来许多麻烦。因此,应把表 1.2.1 改为表 1.2.2 的格式,称为 STUDENT.DBF 数据库。

表 1.2.2 数据库文件

| NO | NAME | SEX | BIRTH | ENGLISH | COMPUTER | PASS | VITA |
|---|---|---|---|---|---|---|---|
| 01 | 王小明 | 男 | 11/23/75 | 86.0 | 75.0 | .T. | Memo |
| 02 | 张兰莉 | 女 | 05/23/76 | 95.0 | 80.0 | .T. | Memo |
| 03 | 何 欢 | 女 | 10/12/75 | 90.0 | 86.0 | .F. | Memo |

需要全本请访问 www.ertongbook.com

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 04 | 王 默 | 男 | 07/30/74 | 63.0 | 47.0 | .T. | Memo |
| 05 | 周利莎 | 女 | 08/11/73 | 45.0 | 63.0 | .T. | Memo |
| 06 | 诸葛丰 | 男 | 02/18/75 | 78.0 | 48.0 | .T. | Memo |
| 07 | 周伟兰 | 女 | 09/18/75 | 69.0 | 28.0 | .F. | Memo |
| 08 | 辛 蔚 | 女 | 04/28/75 | 93.0 | 90.0 | .T. | Memo |

在表 1.2.2 中,用英文代替了表 1.2.1 中的汉字作为字段名。

在同一个数据库中，不能使用相同的字段名命名不同的字段，但在不同的数据库中，允许使用同一个字段名。

2. 字段类型(Fleld Type)

在数据库中，同列数据必须具有相同的类型，它由字段的类型来确定。在 FoxBASE ＋系统中,字段可分为五种类型

(1)字符型(Character)

字符型字段存放字符或汉字数据，即字符串，如表 1.2.2 中的 NO,NAME 和 SEX 等字段。字符型用字母“ C ”表示。

(2) 数值型(Numeric)

数值型字段存储正、负整数或小数，如表 1.2.2 中的 ENGLISH 和 COMPUTER 字段。数值型用字母“N”代表。

(3)日期型 (Date)

日期型字段存储日期数据，有三种格式：

美 式 ： mm/dd/yy　即月／日／年

欧 式 ： dd/mm/yy　即日／月／年

标准式：yy・mm・dd　即　年・月・日

通常，默认的格式为美式。

日期型字段用字母“ D ”代表,如 STUDENT.DBF 数据库中的 BIRTH 字段。

(4) 逻辑型(Logical)

逻辑型字段只存储.T.(True)或.F.(False)两种逻辑值。如表 1.2.2 中的 PASS 字段,存储‘体育达标’情况,已达标为.T.,否则为.F..

逻辑值是计算机中一种特殊数据,只有“真”“假”两种值。

逻辑型字段用字母“L”代表。

(5) 记忆型(Memory)

记忆型又称备注型,用字母“M”代表。这是一种特殊类型的字段,用于存储不定长且数量较大的字符型数据。比如,人事档案中的“个人简历”,每个人的情况不同,简历的字数不同。若使用字符型字段,需要按档案中某个最长简历的人确定字段的宽度。这对于简历短的人,则是很大的浪费。对此,可以选用记忆型字段,如表 1.2.2 中的 Vita 字段。

记忆型是一种字段类型,而不是数据类型。它的数据类型为字符型。

3. 字段宽度(Field with)

在建立数据库结构时必须指定每个字段所存放数据的宽度,即各字段能存储数据的最大长度。

字符型字段最大宽度为 254 个字符位,数值型字段最大宽度为 19 位,日期型固定为 8 位,逻辑型固定为 1 位,记忆型在结构设计时假定为 10 位,实际可存储至多达 64K 个字节。

4. 小数位数(Decimal)

对于数值型字段,还需要指定小数所占的位数。因此,对于整数,所定义的字段宽度即为整数的位数(对于负整数则含一位符号位);对于含小数的字段,所定义的字段宽度为整数位数加小数位数及一位小数点“.”之和。

一个数据库文件至多可达 128 个字段。然而 FoxBASE$^+$ 数据库的字段总宽度限制在 4K 位内,因此,建立数据库结构时,必须满足字段个数和各字段宽度总和两个因素的要求。

### 1.2.3 记录(Record)

数据库文件中,每一行数据叫做一个记录。例如,表 1.2.2 作

为数据库,取名为 STUDENT. DBF,其格式列于表 1. 2. 3。

表 1. 2. 3　　STUDENT. DBF 数据库

| Record # | NO | NAME | SEX | BIRTH | ENGLISH | COMPUTER | PASS | VITA |
|---|---|---|---|---|---|---|---|---|
| 1 | 01 | 王小明 | 男 | 11/23/75 | 86.0 | 75.0 | .T. | Memo |
| 2 | 02 | 张兰莉 | 女 | 05/23/76 | 95.0 | 80.0 | .T. | Memo |
| 3 | 03 | 何　欢 | 女 | 10/12/75 | 90.0 | 86.0 | .F. | Memo |
| 4 | 04 | 王　默 | 男 | 07/30/74 | 63.0 | 47.0 | .T. | Memo |
| 5 | 05 | 周利莎 | 女 | 08/11/73 | 45.0 | 63.0 | .T. | Memo |
| 6 | 06 | 诸葛丰 | 男 | 02/18/75 | 78.0 | 48.0 | .T. | Memo |
| 7 | 07 | 周伟兰 | 女 | 09/18/75 | 69.0 | 28.0 | .F. | Memo |
| 8 | 08 | 辛　蔚 | 女 | 04/28/75 | 93.0 | 90.0 | .T. | Memo |

在 STUDENT. DBF 数据库中有 8 个记录，每个记录有 8 个字段,或者说,该数据库有 8 个字段,在每个字段中,各记录可以存储在宽度限制内且类型相同的数据。一个记录最多可包含 128 个字 段 。

在 1. 2. 2 节中已知，一个数据库字段宽度总和不大于 4K 位，即一个记录最大长度限制为 4K 字符（不包括记忆型字段中存储的字符数据)。

在表 1. 2. 3 中左侧第一列为记录号。每个记录有一个记录号。记录号从 1 起顺序编号，每增加一个记录，自动增加一个记录号。许多材料介绍，一个数据库文件可以存储多达 10 亿个记录。实际上，受到硬件资源，主要是内存容量和速度以及磁盘容量的限制，特别是数据管理和加工速度的限制，很少建立上亿个记录的数据库。

FoxBASE＋系统为每个数据库提供一个记录指针。在刚打开数据库时，记录指针总是指向第一个记录。要调用、修改和删除哪个记录，必须先把记录指针指向那个记录，记录指针所指的记录为当前记录。

## 1.3 FoxBASE +系统数据

在 FoxBASE +系统中，除数据库中各记录数据外，还可以使用常量、变量及函数等形式的数据。本节将介绍这几种数据的格式及其要求。

### 1.3.1 常量(Constant)

常量也称为常数，是在程序中输入、输出的数据，以及数据库中存储的数据，分为三种类型。

#### 1.3.1.1 字符型常量

字符型常量是包含在界限符内的字符串，顾名思义，即一串字符。此后我们将使用字符串一词。

所谓界限符即作为字符串起点和终点的界限的符号。FoxBASE +系统提供三种可供选用的界限符：

1. 单引号：' '，例如 'CHINA'，'北京'
2. 双引号：" "，例如"CHINA"，"北京"
3. 方括号：[]，例如 [CHINA]，[沈阳]

上述三种界限符可以任选，但在同一个字符串中必须相同且成对。例如"姓名 ' '，[性别]，' 英语 ' 等等都是正确的字符串。但是，"姓名 '，" 性别]，" 北京 ' 等都是非法的。有时，字符串中需要包含某个界限符，即需要把界限符作为字符串中的字符，此时必须用另一种界限符定义该字符串的界限。例如，"我们已经学会使用 [FoxBASE+] 研制信息管理系统"，其中 " " 为界限符；我们……系统 '为字符串，方括号 [] 为字符串中的两个字符。

字符串的最大长度为 254 个字符。注意，一个汉字占用两个字符位。

#### 1.3.1.2 数值型常量

数值型常量即人们常说的“数”,包括正、负整数和小数。

在 FoxBASE＋中,最高精度为 16 位有效数字,数值范围:

最大绝对值约:$1\times10^{308}$

最小绝对值约:$1\times10^{-307}$

1.3.1.3 逻辑型常量

逻辑型常量包括“真值”和“假值”两种,其表示字符和形式如下:

真值:.T.,.t.,.Y.,.y.

假值:.F.,.f.,.N.,.n.

在输入逻辑常量时,必须在字母两侧加上“.”号。但是,在计算机内存储时,仅存一个字符 T(真值)或 F(假值),在输出时,也只有.T.或.F.两种情况。

日期型数据没有常量格式,仅当向数据库的日期型字段或日期型变量输入数据时,按日期格式要求输入,将在下节中介绍。

### 1.3.2 变量(Variable)

所谓变量即可以改变数值的量。在计算机中的变量是一组存储单元的代表符号,向变量中存储数据,即向它所代表的存储单元中存储数据,调用变量即从相应的存储单元中读取数据。

在 FoxBASE＋系统中使用的变量分为字段名变量和内存变量。

在数据库管理系统中,我们输入和输出数据都是通过变量实现的。与数据库交换数据和修改数据库数据需要利用字段名变量;在程序中输入临时数据,存储计算结果数据等都需要使用内存变量。

#### 1.3.2.1 字段名变量

当数据库打开后,它的全部字段名都可以作为变量使用,其类型和宽度已在建立数据库结构时定义。但是,一定要注意,字段名

和数据库是联系在一起的，对字段名输入、修改和删除数据都将影响数据库。字段名变量中的数据就是记录指针当时所指记录在该字段的数据。

1.3.2.2 内存变量

在 FoxBASE+系统中，内存变量是由用户定义用于临时存储数据的变量。内存变量的命名规则与字段名相同。在同一个程序中，或在命令操作时，选用的内存变量名不能与已打开的数据库的字段名重名。若二者重名，将遵循‘字段名优先’原则，就是把它看作字段名，而不是内存变量。

内存变量与字段名变量一样，也有四种数据类型：字符型、数值型、逻辑型和日期型，分别用 C、N、L 和 D 代表。它们各自存储对应类型的数据。它们的类型决定于存入数据的类型，比如变量 CHARAC 存入字符串“ CHINA”，则为字符型，然后再存入数 125.25，变量 CHARAC 中原字符串被清除，且存入了数值，则变为数值型。在上节介绍常量时已知，没有日期型常量，因此，不能直接利用输入变量的方法定义日期型内存变量。必须通过日期型字段或日期型函数(将在 1.3.4 节介绍) 定义日期型内存变量。

各种类型内存变量的宽度限制与对应类型字段宽度的限制一致，详见 1.2.2 节。

### 1.3.3 内存变量的赋值和数据输出

1.3.3.1 内存变量的赋值

已知内存变量的类型决定于所赋数据的类型。为内存变量赋值的命令有如下两种格式：

1. 直接赋值

内存变量＝〈表达式〉

它把右边表达式的值赋给左边的变量。例如

SEX＝男′

需要全本请访问 www.ertongbook.com

```
NAME='司马三郎'
NUM=2512*3
LOG=.T.
```

上述各赋值命令行为每个变量赋一个值，同时也定义了变量的类型。表达式的类型即是变量的类型，变量 SEX 和 NAME 为字符型，NUM 为数值型，LOG 为逻辑型。

此命令是算术运算、逻辑运算和字符运算的主要工具。

2. 命令赋值

上述赋值命令，每个命令行只能为一个变量赋值。此节介绍的 STORE 命令可以给若干内存变量同时赋一个相同的值，其格式为：

STORE 〈表达式〉 TO 〈内存变量表〉

内存变量表为一个内存变量或多个用逗号‘，’分隔的内存变量。该命令把表达式的值赋给变量表中的每个变量。例如

```
STORE 0 TO A,B,C
STORE 'O' TO K,M
STORE .F. TO MK
```

在上述各例中，内存变量 A,B,C 均被赋以数零，且同时为数值型内存变量。K 和 M 被同时赋以字符‘'O”，因此成为字符型。

以上两种赋值命令都只能对内存变量赋值，不能为字段名变量赋值。

1.3.3.2 变量中数据的输出

有多种命令可以输出变量中的数据，本节只介绍最简单的输出命令 和 ??此类命令称为非格式输出命令，其格式为：

？（表达式表）

?? 〈表达式表〉

前一种在下一行起始位置输出表达式的值，后一种从光标(或打印机头）当前的位置开始输出表达式的值。简单地说，前一种回

车换行后输出，后一种不回车换行输出。

所谓“表达式表”是指一个或多个表达式。表中各表达式之间用逗号分隔。例如

```
.N=123↙
.M=12345↙
.? N↙
123
.? M↙
12345
.? N,M↙
123   12345
 ? "工资",250+430↙
工资   680
```

在上述各命令行中，行首的圆点“.”为 FoxBASE ＋系统提示符，进入系统后机器自动产生，行尾的“↙”符代表按回车键操作，此后均同，不再说明。

### 1.3.4 常用函数(Function)

函数是各种计算机语言常用数据之一。所谓函数是一个标准程序，在调用它时，对应提供的参数，给出所需要的结果，即函数值。

本节介绍一些在数据库操作中最常用的函数。

#### 1.3.4.1 数据类型转换函数

1. STR(〈数值表达式〉,N[,K])

该函数把数值表达式的值转换成字符串。其中

N 为字符串长度

K 为 N 位中含小数的位数

式中方括号中的内容，可以省略不用(以后遇同种情况不再说明)。如果省略 K 值，则只转换整数值。该函数首先计算表达式的值，然

后取 N 个字符(含 K 位小数)组成字符串。例如

STR(45.78,5,2)="45.78"

STR(25/8,5,3)="3.125"

如果数值的小数位数多于第三个参数 K 值,或者,虽然数值有小数,但是只取整数而无 K 值,都对多余位的小数进行(四舍五入)处理,然后转换成字符串,例如

STR (765.459,6,2) ="765.46"

STR (32.463,5,2)="32.46"

STR (4.34,2)=" 4"

STR (37.56,2) =" 38"

如果数值的整数位数多于指定长度中所限定的整数位数,则不能转换,例如

STR(3546.25,6,2)

给出错误标记'＊＊＊＊＊＊ ",星号的个数取决于指定的长度 N。

凡在函数的自变量位置写有表达式者,均先计算表达式的值,然后将此值作为自变量进行函数计算。

2. VAL(〈字符表达式〉)

该函数是 STR 函数的反函数。它把' 数字式 '的字符串转换成数值。所谓数字式字符串,是由数字、小数点和正负号组成的字符串。允许在数字的前、后有空格,但是中间不得插有空格。例如

VAL(' −123.25' ) =−123.25

VAL(" 23.54" ) = 23.54

M="357.39"

VAL(M) =357.39

3. CTOD(〈字符表达式〉)

该函数把字符型数据转换为日期型数据。它要被转换的字符串必须具有日期格式,且符合日期要求,即' 月月／日日／年年 ",例如

DATE=CTOD("06/25/37")

将字符串“ 06/25/37 ”转换成 1937 年 6 月 25 日，并存入变量 DATE 中,则 DATE 为日期型内存变量。

4. DTOC(〈日期型表达式〉)

该函数把日期型数据转换成字符串，其形式不变。例如，已在变量 DATE 中存入日期 1937年 6 月 25 日,则

CHAR=DTOC(DATE)

将变量 DATE 中的日期 06/25/37 作为字符串存入变量 CHAR 中。

1.3.4.2 字符型函数

1. CHR(<数值表达式>)

该函数以表达式的值作为 ASCII 码，转换成它对应的字符。数值表达式的限制为 0～255之间的正整数,若有小数,则舍去,只截取整数部分。例如

CHR(65 )为字符"A"

CHR(78 )为字符"N"

I=45

CHR(3*I/2) 为字符"C"

2. LEN(〈字符表达式>)

LEN("ABCD" =4

L="

LEN(L)=16

LEN(" ")=0

3. SUBSTR ,I,[,K])

该函数是取子串函数。它从给定的字符串的第 I 个字符起,取 K 个字符组成子串。例如

SUBSTR('ABCDE',3,2)='CD'

SUBSTR("辽宁大学",5,2)="大"

若不指定第 3个参数 K,则从第 I 个字符起一直取至串尾组成子

串。例如

SUBSTR("辽宁省沈阳市",7)= "沈阳市"

请特别注意，一个汉字占两个字符位。在对由汉字组成的字符串取子串时,一定要取整个汉字。若起始位置 I 不在汉字的前一个字符位上，取得的子串将面目全非，不再是原汉字字形，若子串的最后一个字符取半个汉字，则最后一个汉字将变形。

若起点 (I 值）超过指定的字符串长度，则超出范围不能产生子串,系统将自动指示 " 执行出错 "。

4. TRIM（〈字符表达式〉）

该函数压缩掉字符串尾部的空格。例如

TRIM （' 计算机系 '）= ' 计算机系 '

单位= ' 辽宁大学

TRIM(单位)= '辽宁大学'

该函数不能去掉字符串中间和前面的空格。

5. LTRIM(〈字符表达式〉）

该函数与 TRIM 函数相似，它删除字符串首部的空格。例子略。

6. UPPER(〈字符表达式〉）

该函数把字符表达式中的全部小写字母转换为大写字母。例如

UPPER('abc')='ABC'

UPPER('aBC')='ABC'

UPPER('y')='Y'

7. LOWER(〈字符表达式〉）

该函数与 UPPER 函数相反，它把字符表达式中全部大写字母转换成小写字母，而小写字母不变。

8. SPACE（〈数值表达式〉）

此函数为产生空格字符串的函数。产生空格字符串的长度(即

空格个数)取决于表达式的值。例如

SPACE(5)=' '

1.3.4.3 数值型函数和日期型函数

1. ASC(〈字符表达式〉)

该函数是函数 CHR 的反函数。它给出指定字符串的第一个字符的 ASCII 码。例如

ASC("A")=65

ASC("计算中心")= 188

K="ABCD"

ASC(K)=65

2. 取整函数:INT(〈数值表达式〉)

该函数取数值表达式的值的整数部分。例如

INT(34.45)=34

INT(45.99)=45

INT(-36.10)=-36

INT(25/3)=8

3. 日期函数:DATE( )

该函数给出机器的当前系统日期,其格式与日期型数据相同,即月月/日日/年年。例如,03/13/89。

注意,该函数无自变量,函数值取自于机器工作的时钟。PC 计算机在启动 DOS 系统时可由操作员设置日期,如果机器装有时钟电池,则只在安装机器时一次设置并校准日期和时间,此后机器自动计时,再开机启动 DOS 时就不必设置时间了。

1.3.4.4 宏代换函数

宏代换函数:&〈字符变量〉代换出字符变量中的字符串。

例 1 设 单位 = "辽宁大学计算中心"

则 "寄 & 单位"的结果为字符串:"寄辽宁大学计算中心"

例 2 设 MK ="GS"

则 &MK+YY

需要全本请访问 www.ertongbook.com