```
    @ 1,COL() SAY''
  ENDIF
  IF ch2=6
    @ 1,COL() SAY''
  ENDIF
  @ 1,COL() GET jg PICTURE'9999999.99'
  READ
  IF ch2=1
    tj1=tj1+'jg='+STR(jg,10,2)
  ENDIF
  IF ch2=2
    tj1=tj1+'jg>'+STR(jg,10,2)
  ENDIF
  IF ch2=3
    tj1=tj1+'jg<'+STR(jg,10,2)
  ENDIF
  IF ch2=4
    tj1=tj1+'jg<>'+STR(jg,10,2)
  ENDIF
  IF ch2=5
    tj1=tj1'jg>='+STR(jg,10,2)
  ENDIF
  IF ch2=6
    tj1=tj1'jg<='+STR(jg,10,2)
  ENDIF
  x=x+20
 ENDIF
ENDDO
```

用 FoxBASE+ 实现基于数据库的窗口编辑器

林　民　（内蒙师大计算机系）

摘要：本文针对 FoxBASE+ 中 MEMO 字段处理变长信息功能差的问题，用 FoxBASE+ 设计了一个基于数据库的窗口编辑器，克服了 MEMO 字段的缺点，较好的解决了变长信息的存储、编辑、显示、检索、打印等处理问题。

一、引言

在开发微机数据库应用系统中，经常会遇到处理信息量大、伸缩性也大的文字信息。例如，档案管理中个人简历、大事记，题库管理中说明性问题的答案等。若采用 FoxBASE+ 中提供的 MEMO 字段来处理这类信息，会遇到下述难以解决的问题：

1. 编辑屏幕格式单一，无法控制
2. 编辑时存在半个汉字的处理问题
3. 显示输出会破坏原有屏幕格式
4. 难于进行检索
5. 打印报表实现复杂
6. 磁盘空间利用率不高

因此，为解决这些问题，笔者开发了基于数据库的窗口编辑器，能较好的适应变长信息的存储、编辑、显示、检索、打印等方面的需要。考虑到 FoxBASE+ 与其它语言接口的复杂性以及需占用很大内存空间的限制，这里直接采用 FoxBASE+ 自身的命令和函数来实现编辑器。

二、有关数据库结构的设计

1. 辅助数据库结构

辅助数据库用来代替存放 MEMO 字段信息的辅助文件（*.DBT），其结构如下：

NO1 字段：(N)用来存储变长信息字段对应的记录号

NO2 字段：(N)用来存储变长信息字段在记录中的序号

TEXT 字段：(C)用来存储相应的变长信息

2. 编辑数据库结构

供编辑器使用，用来暂时存放变长信息，其结构如下：

TEXT 字段：(C)存储输入的变长信息

这样，编辑器直接对编辑数据库记录进行操作，再将编辑好的信息添加到辅助数据库。一个数据库里的所有变长信息字段内容都存储在一个辅助数据库中，并可以按 NO1、NO2 字段建立索引，便于检索和显示。

三、编辑器的设计方法

1. 调用方法及参数说明

DO EDITOR WITH ROW1, COL1, ROW2, COL2, MFILE

其中，EDITOR：编辑器过程文件名

ROW1，COL1：窗口左上角坐标

ROW2，COL2：窗口右下角坐标

MFILE：编辑数据库文件名

2. 主要变量说明

TEXTCOLOR：文本显示颜色

BLACKCOLOR：背景颜色

SS(400)：编辑缓冲区数组

SS-MAX：编辑数组的实用最大行号

KEY：存放按键值

XX，YY：当前行在文本中的绝对行、列号

SS-XX:当前行在编辑数组中的行号
X,Y:当前行在显示屏幕上的行、列坐标
SER:当前光标处的字符序数
M:左右移屏的屏号
BP:每次移屏的字符数
INS:是否为插入状态标志变量
CHG:文本是否修改过标志变量
OLD:区别新、旧文件的标志变量
FP-END:旧文件读完的标志变量
TTL-X:文本末行行号
FP-RD:文本已读出行的最大行号
WRA(100),WRB(100):记录临时文件读写地址数组
OA,OB:供 WRA、WRB 数组使用的下标变量

3.关键模块实现技术

①字符(汉字)输入模块 CHR

```
chg=1 && 文本已修改标志置 1
g=len(ss(ss-x))
if ins#0.or.yy=g && 若为插入状态或写在行末
.
.
.
*插入处以后字符后移一列
ss(ss-x)=stuff(ss(ss-x),yy+1,1,chr(key)-;
substr(ss(ss-x),yy+1,1))
else && 若为修改状态
*若为半角覆盖全角,全角字符后半字用空格覆盖
if key<127.and.asc(substr(ss(ss-x),yy+1,1))>160
ss(ss-x)=stuff(ss(ss-x),yy+1,2,chr(key)+'')
endif
*若为全角覆盖半角和一个全角字符前半字,全角字符后半字用空格覆盖
if vs(yy)=0.and.key>160.and.asc(subs(ss(ss-x),yy+1,1))<127;
.and.asc(substr(ss(ss-x),yy+2,1))>160
ss(ss-x)=stuff(ss(ss-x),yy+2,2,'')
endif
ss(ss-x)=stuff(ss(ss-x),yy+1,1,chr(key))
endif
*当前光标位置右移一格
yy=yy+1
y=y+1
ser=ser+1
.
.
.
*重新显示屏幕
if y>zs-1
m=m+1
do disp-t
else
do disp with ss-x,x
endif
.
.
```

②显示一行文本模块 DISP

```
k=int(m*bp) && 本屏首列号
s=ss(a)
g=len(s)
n=iif(g>zs+k,zs+k,g) && 计算本屏当前行尾的列号
if g<=k && 如行尾在以前屏,本屏为空串
s=''
else && 行尾在本屏
*处理本屏第一个字符是汉字情况
f=1
do while j<=n
if asc(substr(s,j,1))>160 && 计算汉字字节数
u=u+1
endif
*本屏第一字节是汉字后半字节时,本屏第一字节用空格代替
if m#0.and.j=k.and.mod(u,2)#0
s=stuff(s,k+1,1,'')
u=u+1
j=j+1
endif
j=j+1
enddo
if mod(u,2)#0
n=n+1
endif
s=substr(s,k+1,n-k)
endif
q=len(s)
*显示当前行
@row1+i,col1 say s
@row1+i,col1+q say space(zs-q)
```

③删除光标处字符(汉字)模块 DELC

```
do while .t.
if y>0          && 当前列号在屏中
*光屏处开始的字符串前移一格,覆盖光标前一字符
ss(ss-x)=stuff(ss(ss-x),yy,1,'')
y=y-1           && 修改相应列坐标
yy=yy-1
ser=ser-1       && 字序数-1
if vs(yy-1)=0 && 若不在全角字符前半字
if k=0          && 若不在屏幕最左列,返回 0,否则返回 1
return 0
```

```
else
return 1
endif
endif
else                    && 当前列号在屏幕最左列
if m#0                  && 若不在第0屏,退到前一屏
m=m-1
y=yy-m*bp               && 重新计算坐标
.
.
.
if xx#0                 && 若不在文本首行
g=len(ss(ss-x))         && 测行长
ss-x=ss-x-1             && 上移一行
yy=len(ss(ss-x))
.
.
.
xx=xx-1
*当前行改为上一行,下行文本接到本行尾
ss(ss-x)=ss(ss-x)-ss(ss-x+1)
*数组后续各行前移,覆盖原行
do movbk with ss-x+1,1
附:VS模块 计算全角字节数,返回奇偶值
j=0
i=1
o while i<=a+1
*统计全角字节数
if asc(substr(ss(ss-x),i,1))>160
j=j+1
endif
i=i+1
enddo
return mod(j,2) && 返回奇偶值

④回车换行模块 ENTER
chg=1                   && 文本已修改标志置1
if ins=0                && 若为修改状态
g=len(ss(ss-x))         && 计算当前行长
.
.
.
*修改有关坐标
ss-x=ss-x+1
xx=xx+1
*计算字符数
ser=ser+(g-yy)
*显示屏幕
.
.
.
else                    && 若为插入状态
g=m
k=x
do intercep with yy && 从当前光标处折断字符串
*显示屏幕
if g=0
if k<h3
scroll row1+x,coll,row1+h3,co12,-1
else
scroll rowl,coll,row1+h3,co12,+1
endif
do disp with ss-x=1,x-1
do disp with ss-x,x
else
do disp-t
endif
endif
store 0 to y,yy
```

四、结束语

本编辑器基本功能齐全,响应速度较快,使用方法与一般较流行的编辑器 WS、WPS 一致,便于掌握和使用。由于采用 FoxBase+ 开发,程序短小、易懂、易维护,在此基础上可以再扩充块操作有关功能,使编辑器功能更加完备。此外,对编辑器程序稍做修改可成为一个实用的窗口显示器,可以应用在文本浏览显示,打印报表的预先显示等方面。这些技术将会大大提高数据库应用系统的开发质量。

参考文献:

①史济民,《FoxBase+及其应用系统开发》,清华大学出版社

②沈建威等,《一个编辑软件的设计与编制》,清华大学出版社