Curso Avançado de Natural

Luciano Perdigão

natural_2-owner@yahoo.com.br

Grupo no Yahoo da Linguagem Natural 2

Para mais informações: **http://br.groups.yahoo.com/group/natural_2**

Enviar mensagem.....: **natural_2@yahoo.com.br**

Assinar.....: **natural_2-subscribe@yahoo.com.br**

Cancelar Assinatura..: **natural_2-unsubscribe@yahoo.com.br**

Proprietário da Lista: **natural_2-owner@yahoo.com.br**

HISTÓRICO DE ATUALIZAÇÕES

DATA	AUTOR	ATUALIZAÇÃO

ÍNDICE

1. DEFINIÇÃO DE VARIÁVEIS

1.1. Tipo de Dados

Os tipos de dados existentes em Natural são:

```
#A1     (A10)  /* 10 posições alfanuméricas.
#A2     (B4)   /*  4 posições binárias.
#A3     (P4)   /* Numérico compactado, 4 posições e 1 sinal.
#A4     (N7.2) /* Numérico descompactado, 7 posições e 2 decimais.
#A6     (P7.2) /* Numérico compactado, 7 posições e 2 decimais + o sinal.
#INT1   (I1)   /* Inteiro, 1 byte.
#INT2   (I2)   /* Inteiro, 2 bytes.
#INT4   (I4)   /* Inteiro, 4 bytes.
#FLT4   (F4)   /* Ponto flutuante, 4 bytes.
#FLT8   (F8)   /* Ponto flutuante, 8 bytes.
#DATE   (D)    /* Date (Formato e tamanho interno P6).
#TIME   (T)    /* Time (Formato e tamanho interno P12).
#SWITCH (L)    /* Lógico, 1 byte (TRUE or FALSE).
```

Os tipos de dados abaixo contêm definição inválida.

```
#A5   (N7.)     /* definição inválida!!!
#INT3 (I3)      /* definição inválida!!!
#INT5 (I5)      /* definição inválida!!!
#FLT2 (F2)      /* definição inválida!!!
```

1.2. Init

Podemos inicializar uma variável com o valor de uma variável de sistema do Natural.

```
     0010 DEFINE DATA LOCAL
     0020 1 #DATA      (D) INIT <*DATX>
     0030 END-DEFINE
     0040 DISPLAY #DATA
     0050 END

Resultado:

#DATA
--------

18112004
```

A opção FULL LENGTH preenche um campo alfanumérico com um caracter especificado. Se múltiplos caracteres são especificados, o grupo é repetido até que o campo fique completamente preenchido.

A opção LENGTH *n* preenche as *n* primeiras posições de um campo com o caracter ou grupo de caracteres especificados.

Estas opções também são válidas para estruturas de ARRAY.

```
     0010 DEFINE DATA LOCAL
     0020 1 #CAMPO     (A25) INIT FULL LENGTH <'*'>
     0030 END-DEFINE
     0040 DISPLAY #CAMPO
     0050 END
```

Resultado:

```
         #CAMPO
-------------------------

************************
```

```
     0010 DEFINE DATA LOCAL
     0020 1 #CAMPO     (A25) INIT LENGTH 6 <'*'>
     0030 END-DEFINE
     0040 DISPLAY #CAMPO
     0050 END
```

Resultado:

```
         #CAMPO
------------------------

******
```

1.3. Const

Define o conteúdo de uma variável como uma constante. O conteúdo desta variável não muda de valor durante a execução do programa. Exemplo:

```
     0010 DEFINE DATA LOCAL
     0020 1 #VAR      (N1) CONST<5>
     0030 END-DEFINE
     0040 END
```

1.4. Arrays

Utilize arrays para armazenar tabelas na memória, onde a grande maioria dos módulos terá acesso.

Exemplos de definição de Array:

```
#TAB1(A5/6)                    /* Array de uma dimensão
#TAB2(A5/1:7,1:7)              /* Array de duas dimensões
#TAB3(A5/1:4,1:4,1:4)          /* Arrau de três dimensões
```

Uma variável definida como array pode ser referenciada da seguinte maneira:

```
#TABX(*)
#TABY(*,*)
#TABX(1)
#TAB(5,#FIELDX)
```

Deve-se escrever rotinas padronizadas para armazenar e pesquisar tabelas.

Para qualquer referência a uma tabela genérica por parte da rotina padronizada, pode utilizar a técnica de parametrização de dimensões:

```
0010 DEFINE DATA LOCAL
0020 1 #T          (N3) CONST <50>
0030 1 #A1         (A3/#T)
0040 1 #A2         (I4/#T)
0050 END-DEFINE
```

Para inicialização de arrays bidimensionais e tri-dimensionais utilize a notação “V”. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #VAR1 (A1/2,2) INIT (1,V) <'A','B'>
0030 1 #VAR2 (N5/2,3) INIT (1,2) <200>
0040 1 #VAR3 (A1/4,3) INIT (V,2:3) <'W','X','Y','Z'>
0050 END-DEFINE
0070 WRITE NOTITLE '=' #VAR1 (1,1) '=' #VAR1 (1,2)
0080             / '=' #VAR1 (2,1) '=' #VAR1 (2,2)
0100 WRITE ///     '=' #VAR2 (1,1) '=' #VAR2 (1,2) '=' #VAR2 (1,3)
0110             / '=' #VAR2 (2,1) '=' #VAR2 (2,2) '=' #VAR2 (2,3)
```

```
     0130 WRITE ///     '=' #VAR3 (1,1) '=' #VAR3 (1,2) '=' #VAR3 (1,3)
     0140 WRITE       / '=' #VAR3 (2,1) '=' #VAR3 (2,2) '=' #VAR3 (2,3)
     0150 WRITE       / '=' #VAR3 (3,1) '=' #VAR3 (3,2) '=' #VAR3 (3,3)
     0160 WRITE       / '=' #VAR3 (4,1) '=' #VAR3 (4,2) '=' #VAR3 (4,3)
     0180 END

Resultado:

#VAR1: A #VAR1: B
#VAR1:   #VAR1:

#VAR2:      0 #VAR2:    200 #VAR2:      0
#VAR2:      0 #VAR2:      0 #VAR2:      0

#VAR3:   #VAR3: W #VAR3: W
#VAR3:   #VAR3: X #VAR3: X
#VAR3:   #VAR3: Y #VAR3: Y
#VAR3:   #VAR3: Z #VAR3: Z
```

Exemplo de inicialização de intervalos de array com diferentes valores:

```
     0010 DEFINE DATA LOCAL
     0020 1 #VAR1 (A1/3,5) INIT
     0030                        (1,1:3) <'A'>  (1,4:5) <'B'>
     0040                        (2,1:2) <'C'>  (2,3:5) <'D'>
     0050                        (3,1:3) <'E'>  (3,4:5) <'F'>
     0060 END-DEFINE
     0070 WRITE #VAR1(1,*) /
     0080       #VAR1(2,*) /
     0090       #VAR1(3,*)
     0100 END

Resultado:

A A A B B
C C D D D
E E E F F
```

Exemplo de inicialização de ocorrências do array com o mesmo valor:

```
     0010 DEFINE DATA LOCAL
     0020 1 #VAR1 (A1/3,5) INIT ALL <'A'>
     0060 END-DEFINE
     0070 WRITE #VAR1(1,*) /
     0080       #VAR1(2,*) /
     0090       #VAR1(3,*)
     0100 END

Resultado:

A A A A A
A A A A A
```

```
A A A A A
```

1.5. Inicialização de Arrays no Editor de Dados

```
Local     LCARRO    Library XXXPRG                          DBID    29 FNR     4
Command                                                                   > +
I T L Name                            F Leng Index/Init/EM/Name/Comment
All - ------------------------------- - ---- ---------------------------------
    1 #PLACA_DO_CARRO                 A    7
    1 #MODELO_CARRO                   A   15
    1 #ANO_CARRO                      N    4 (10)









----- Current Source Size: 145  Free: 100800 -------------------- S 3    L 1
```

```
Local     LCARRO    Library XXXPRG                          DBID    29 FNR     4
Command                                                                   > +
I T L Name                            F Leng Index/Init/EM/Name/Comment
All - ------------------------------- - ---- ---------------------------------
    1 #PLACA_DO_CARRO                 A    7
    1 #MODELO_CARRO                   A   15
  . e #ANO_CARRO                      N    4 (10)









----- Current Source Size: 145  Free: 100800 -------------------- S 3    L 1
```

Com a execução do comando de linha ".E", uma tela com um menu de opções é exibida:

```
08:53:10               *****  EDIT FIELD  *****                    17/11/2004
                     - Initial Values and Edit Mask -
```

```
Local     LCARRO    Library XXXPRG                          DBID    29 FNR     4


         Code  Function                        Definition
        -----  ------------------------------  ----------
          S     Single Value Initialization    no
          F     Free Mode Initialization       no
          E     Edit Mask Definition           no
          P     Parameter Type Definition      no
          D     Delete all Definitions
          ?     Help
          .     Exit
        -----  ------------------------------  ----------

   Code   ?  for Field: #ANO_CARRO(N4/10)
```

A opção “S” abre uma tela para preenchimento dos valores de inicialização do ARRAY, por ocorrência, como segue:

```
08:53:10               *****  EDIT FIELD  *****                    17/11/2004
                     - Initial Values - Single Mode -
Local     LCARRO    Library XXXPRG                     DBID    29 FNR     4
Command  +

       Index            #ANO_CARRO(N4/10)
---------------------- --------------------------------------------------
(1)
(2)
(3)
(4)
(5)
(6)
(7)
(8)
(9)
(10)
```

```
08:55:20               *****  EDIT FIELD  *****                    17/11/2004
                     - Initial Values - Single Mode -
Local     LCARRO    Library XXXPRG                     DBID    29 FNR     4
Command  +

       Index            #ANO_CARRO(N4/10)
---------------------- --------------------------------------------------
(1)                    1995
(2)                    1996
(3)                    1997
(4)                    1998
(5)                    1999
(6)                    2000
(7)                    2001
(8)                    2002
(9)                    2003
(10)                   2004
```

Na volta ao editor de Data Area, o campo estará com o label “I” na coluna I, mostrando que este campo tem inicialização de valores.

```
Local     LCARRO    Library XXXPRG                          DBID    29 FNR     4
Command                                                                   > +
I T L Name                            F Leng Index/Init/EM/Name/Comment
All - ------------------------------- - ---- --------------------------------
    1 #PLACA_DO_CARRO                 A    7
    1 #MODELO_CARRO                   A   15
I   1 #ANO_CARRO                      N    4 (10)









----- Current Source Size: 145  Free: 100840 -------------------- S 3    L 1
```

O comando GENERATE ou GEN gera um fonte do tipo COPYCODE com a definição do ARRAY e sua inicialização, como segue:

```
Local     LCARRO    Library XXXPRG                          DBID    29 FNR     4
Command gen                                                               > +
I T L Name                            F Leng Index/Init/EM/Name/Comment
All - ------------------------------- - ---- --------------------------------
    1 #PLACA_DO_CARRO                 A    7
    1 #MODELO_CARRO                   A   15
I   1 #ANO_CARRO                      N    4 (10)









----- Current Source Size: 145  Free: 100840 -------------------- S 3    L 1
```

```
>                                     > +  Copycode             Lib XXXPRG
All    ....+....1....+....2....+....3....+....4....+....5....+.Mode Reporting..
```

```
   0010 DEFINE DATA LOCAL
   0020 1 #PLACA_DO_CARRO(A7)
   0030 1 #MODELO_CARRO(A15)
   0040 1 #ANO_CARRO(N4/10)
   0050   INIT
   0060   (1)<1995>
   0070   (2)<1996>
   0080   (3)<1997>
   0090   (4)<1998>
   0100   (5)<1999>
   0110   (6)<2000>
   0120   (7)<2001>
   0130   (8)<2002>
   0140   (9)<2003>
   0150   (10)<2004>
   0160 END-DEFINE
   0170
   0180
   0190
   0200
        ....+..Current Source Size: 268 Char. Free: 100677 ...+... S 16   L 1
```

Para transformar este copycode em programa, basta utilizar o comando SET TYPE P.

```
> set type p                            > +  Copycode             Lib XXXPRG
 All    ....+....1....+....2....+....3....+....4....+....5....+.Mode Reporting..
   0010 DEFINE DATA LOCAL
   0020 1 #PLACA_DO_CARRO(A7)
   0030 1 #MODELO_CARRO(A15)
   0040 1 #ANO_CARRO(N4/10)
   0050   INIT
   0060   (1)<1995>
   0070   (2)<1996>
   0080   (3)<1997>
   0090   (4)<1998>
   0100   (5)<1999>
   0110   (6)<2000>
   0120   (7)<2001>
   0130   (8)<2002>
   0140   (9)<2003>
   0150   (10)<2004>
   0160 END-DEFINE
   0170
   0180
   0190
   0200
        ....+..Current Source Size: 268 Char. Free: 100677 ...+... S 16   L 1
```

```
>                                      > +  Program              Lib XXXPRG
 All    ....+....1....+....2....+....3....+....4....+....5....+.Mode Reporting..
   0010 DEFINE DATA LOCAL
   0020 1 #PLACA_DO_CARRO(A7)
   0030 1 #MODELO_CARRO(A15)
   0040 1 #ANO_CARRO(N4/10)
   0050   INIT
   0060   (1)<1995>
   0070   (2)<1996>
   0080   (3)<1997>
   0090   (4)<1998>
   0100   (5)<1999>
```

```
  0110   (6)<2000>
  0120   (7)<2001>
  0130   (8)<2002>
  0140   (9)<2003>
  0150   (10)<2004>
  0160 END-DEFINE
  0170
  0180
  0190
  0200
       ....+..Current Source Size: 268 Char. Free: 100677 ...+... S 16   L 1
```

1.6. Inicialização de Variáveis de Controle no Editor de Dados

```
Local     LCARRO    Library XXXPRG                          DBID    29 FNR     4
Command                                                                   > +
I T L Name                            F Leng Index/Init/EM/Name/Comment
All - ------------------------------- - ---- ---------------------------------
    1 #PLACA_DO_CARRO                 A    7
    1 #MODELO_CARRO                   A   15
    1 #ANO_CARRO                      N    4 (10)
  . e #CV_ANO_CARRO                   C









----- Current Source Size: 185  Free: 100912 ------------------- S 4    L 1
```

```
09:04:00               *****  EDIT FIELD  *****                     17/11/2004
                    - Initial Values and Edit Mask -

Local     LCARRO    Library XXXPRG                          DBID    29 FNR     4


        Code  Function                        Definition
       -----  -----------------------------  ----------
         S     Single Value Initialization    no
         F     Free Mode Initialization       no
         E     Edit Mask Definition           no
         P     Parameter Type Definition      no
         D     Delete all Definitions
         ?     Help
         .     Exit
       -----  -----------------------------  ----------

   Code   s  for Field: #CV_ANO_CARRO(C)
```

```
09:04:00                *****  EDIT FIELD  *****                  17/11/2004
                         - Initial Values and Edit Mask -

Local     LCARRO    LIBRARY XXXPRG                        DBID    29 FNR     4


          I  Attribute             I  Colour
          -  --------------------  -  ------------
          _  D  Default Intensity  _  NE  Neutral
          _  N  Non-Displayable    _  BL  Blue
          _  B  Blinking           _  RE  Red
          _  I  Intensified        _  GR  Green
          _  C  Italic             _  YE  Yellow
          _  V  Reversed Video     _  PI  Pink
          _  U  Underlined         _  TU  Turquoise
          _  P  Write Protection
          -  --------------------  -  ------------

Mark attributes and/or colour for field:
and hit ENTER, to return to the editor.
```

```
09:04:00                *****  EDIT FIELD  *****                  17/11/2004
                         - Initial Values and Edit Mask -

Local     LCARRO    LIBRARY XXXPRG                        DBID    29 FNR     4


          I  Attribute             I  Colour
          -  --------------------  -  ------------
          _  D  Default Intensity  _  NE  Neutral
          _  N  Non-Displayable    _  BL  Blue
          _  B  Blinking           _  RE  Red
          x  I  Intensified        _  GR  Green
          _  C  Italic             x  YE  Yellow
          _  V  Reversed Video     _  PI  Pink
          _  U  Underlined         _  TU  Turquoise
          _  P  Write Protection
          -  --------------------  -  ------------

Mark attributes and/or colour for field:
and hit ENTER, to return to the editor.
```

```
09:05:40                *****  EDIT FIELD  *****                  17/11/2004
                      - Initial Values - Attributes -
Local     LCARRO    Library XXXPRG                        DBID    29 FNR     4
Command  +

#CV_ANO_CARRO(C/10)

        Index                 Attributes                 Colours
 ----------------------    D  N  B  I  C  V  U  P   NE BL RE GR YE PI TU
 (1)                       _  _  _  x  _  _  _  _    _  x  _  _  _  _  _
 (2)                       _  _  _  _  _  _  _  _    _  _  x  _  _  _  _
 (3)                       _  _  _  x  _  _  _  _    _  _  _  x  _  _  _
 (4)                       _  _  _  _  _  _  _  _    _  _  _  _  x  _  _
 (5)                       _  _  _  x  _  _  _  _    _  _  _  _  _  x  _
 (6)                       _  _  _  _  _  _  _  _    _  _  _  _  _  _  x
 (7)                       _  _  _  x  _  _  _  _    _  _  _  _  _  x  _
 (8)                       _  _  _  _  _  _  _  _    _  _  _  _  x  _  _
```

```
 (9)                    _  _  _  x  _  _  _  _    _  _  _  x  _  _  _
 (10)                   _  _  _  _  _  _  _  _    _  _  x  _  _  _  _
```

```
Local     LCARRO    Library XXXPRG                          DBID    29 FNR     4
Command                                                                  > +
I T L Name                            F Leng Index/Init/EM/Name/Comment
All - ------------------------------- - ---- --------------------------------
    1 #PLACA_DO_CARRO                 A    7
    1 #MODELO_CARRO                   A   15
    1 #ANO_CARRO                      N    4 (10)
I   1 #CV_ANO_CARRO                   C      (10)









----- Current Source Size: 208  Free: 100489 ------------------- S 4    L 1
```

```
>                                      > +  Program              Lib XXXPRG
All    ....+....1....+....2....+....3....+....4....+....5....+.Mode Structured.
  0010 DEFINE DATA LOCAL
  0020 1 #PLACA_DO_CARRO(A7)
  0030 1 #MODELO_CARRO(A15)
  0040 1 #ANO_CARRO(N4/10)
  0050   INIT
  0060   (1)<1995> (2)<1996> (3)<1997> (4)<1998> (5)<1999>
  0070   (6)<2000> (7)<2001> (8)<2002> (9)<2003> (10)<2004>
  0080 1 #CV_ANO_CARRO(C/10)
  0090   INIT
  0100   (1)<(AD=I CD=BL)> (2)<(CD=RE)> (3)<(AD=I CD=GR)> (4)<(CD=YE)>
  0110   (5)<(AD=I CD=PI)> (6)<(CD=TU)> (7)<(AD=I CD=PI)> (8)<(CD=YE)>
  0120   (9)<(AD=I CD=GR)> (10)<(CD=RE)>
  0130 END-DEFINE
  0140 END
  0150
  0160
  0170
  0180
  0190
  0200
       ....+..Current Source Size: 441 Char. Free: 100520 ...+... S 14   L 1
```

1.7. AIVs – Application Independent Variables

Permitidas somente no modo estruturado, estas variáveis devem ser definidas com o prefixo "+", e devem estar na cláusula INDEPENDENT da instrução DEFINE DATA.

A cláusula INDEPENDENT deve ser codificada por último dentro da estrutura do DEFINE DATA.

Podem ser utilizadas para comunicação entre programas que tenham GDAs (GLOBAL DATA AREA) diferentes. Isto porque, quando um programa é chamado via FETCH e sua GDA é diferente da GDA do chamador, os dados da GDA do programa chamador são perdidos, permanecendo as AIVs definidas nos dois programas.

2. MAPAS

2.1. Considerações Básicas

Todas as aplicações que utilizem Mapas e Windows devem estabelecer um padrão de desenho, preenchimento e/ou apresentação, de forma a facilitar o trabalho do usuário.

Estes padrões, em conjunto com outros critérios técnicos, servirão para tomar a aplicação produtiva para o usuário e com performance para o ambiente.

No caso de Mapas recomenda-se padronizar o seguinte:

- Títulos
- Sub-Títulos
- Réguas de PF's e suas funções
- Linha de mensagens
- Mensagens
- Texto que precede o início da área de preenchimento por campo.

Esta padronização proporciona ganhos de performance, pois mapas quase iguais exigem um mínimo de transmissão e de I/O de terminal.

Outro importante mecanismo, que melhora a performance e facilita as atividades dos usuários é a utilização de comandos diretos, evitando a navegação excessiva por vários menus.

A participação do usuário final é muito importante no desenho de telas que contenham um grande número de campos de preenchimento. A ordem de colocação destes pode favorecer, e muito, o trabalho operacional, minimizando a freqüência de erros e assim melhorando a performance.

No caso de Windows as características básicas são:

- Pouca informação
- Facilidade na comunicação com o usuário
- Muito utilizada para Help de preenchimento de campos.

Algumas padronizações criadas para os mapas devem ser também utilizadas para windows, como:

- Títulos
- Réguas de PFs e suas funções
- Linha de mensagens
- Mensagens

Deve-se tomar cuidado ao decidir por uma windows, levando-se em conta o percentual de espaço que ela irá ocupar na tela.

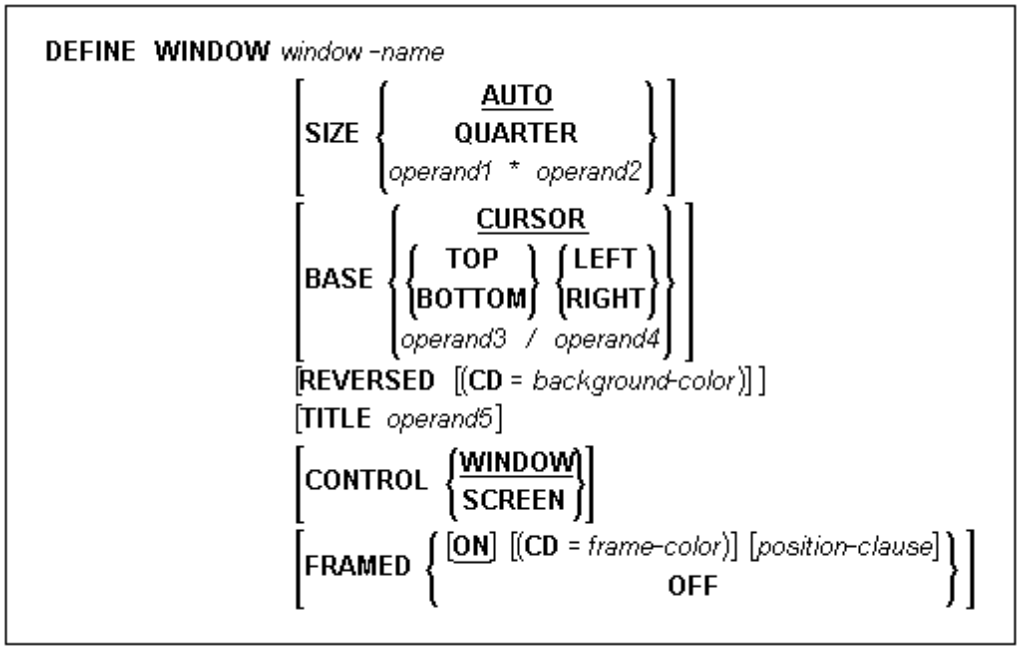
Para os casos em que se exibe windows sobre window (pop-up), recomenda-se que todas tenham o mesmo tamanho e sejam exibidas cobrindo umas as outras apenas parcialmente, para que o usuário saiba qual foi a sua seqüência de navegação.

As windows contribuem para uma boa performance de aplicação, uma vez que utilizam buffers menores.

O tamanho da linha de mensagens de uma window fica restrito ao tamanho da mesma, criando-se, às vezes, dificuldades de comunicação com o usuário.

Se múltiplas windows estão visíveis na tela física, somente a última se encontra ativa.

2.2. Definição de Janelas



- SIZE → Tamanho da window.

- AUTO → o tamanho da window é determinado automaticamente pelo Natural. Para isso, são utilizados alguns dados, como segue:
    - O número de linhas da window será o número de linhas do input, acrescido, se for o caso, de linha de PF-KEYs e linha de mensagem.
    - O número de colunas da window é determinado pelo tamanho da linha do input. O Natural procura, do começo ao fim da linha, bytes significativos. Isto pode causar a truncagem de um campo de input ou output modificável. Para que isso não aconteça, deve-se colocar um caracter texto no fim da linha onde estão estes campos, ou informar explicitamente o tamanho da window no comando "SIZE *op1 * op2*".

- QUARTER → o tamanho da window será ¼ da tela física.
- *OP1 * OP2* → o número de linhas da window é determinado pelo *op1*, e o número de colunas pelo *op2*. Devem ser inteiros. Se uma window contiver frame, o tamanho especificado será inclusive com o frame. O tamanho mínimo possível é duas linhas por dez colunas sem frame, e a quatro linhas por treze colunas com frame. O tamanho máximo será o tamanho definido para a tela física.
  Se a cláusula SIZE for omitida, o SIZE AUTO será aplicado como default.
- BASE → determina a posição da window na tela física.
    - CURSOR → posiciona o topo à esquerda da window na posição corrente do cursor. Se o tamanho da window impossibilita a sua colocação na posição do cursor, o Natural tenta ajustá-la para uma posição possível.
    - TOP / BOTTOM / LEFT / RIGHT → posiciona a window no topo à esquerda, rodapé à esquerda, topo à direita, rodapé à direita, respectivamente, na tela física.
    - *OP3 / OP4* → posiciona o topo à esquerda da window na especificação de linha/coluna da tela física. O número da linha é determinado pelo *op3*, e a coluna pelo *op4*. Nenhum dos operandos pode ter dígito decimal. Se o tamanho da window impossiblitar a colocação na posição da tela física determinada, será retornada uma mensagem de erro.
      Se a cláusula BASE for omitida, o BASE CURSOR será aplicado como default.
- REVERSED (CD=*background-color*) → mostra a área definida para a janela em vídeo reverso. As possíveis cores são:

```
BL = Azul
GR = Verde
NE = Neutro
PI = Rosa
RE = Vermelha
TU = Azul Claro
YE = Amarela
```

- TITLE → com esta cláusula poderá ser definido um cabeçalho para a window. O título especificado (*op5*) será exibido centralizado na linha de frame do topo da window. Este título pode ser definido como texto entre apóstrofes, ou com o conteúdo de uma variável auxiliar. Se o título for maior que a linha da window, será truncado. O título só será exibido se a window estiver com frame, caso contrário será ignorado.
- FRAMED → por default, se esta cláusula for omitida, a window terá frame. Se for especificado FRAMED OFF, todas as cláusulas definidas para o frame (título e informação de posição) serão desligadas.
- POSITION OFF → não exibe informação de posição. Se a cláusula não for especificada, será mostrado na linha do frame, à direita, a posição da window na página lógica.

SET WINDOW 'nome window'

ou

SET WINDOW OFF

O comando DEFINE WINDOW apenas especifica a janela. Para mostrá-la, utiliza-se o comando SET WINDOW 'nome-da-window'. Para desligar é só emitir o comando SET WINDOW OFF. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #I           (I2)
0030 END-DEFINE
0040 *
0050 DEFINE WINDOW TESTE-JANELA
0060   SIZE 15 * 50
0070   BASE BOTTOM RIGHT
0080   FRAMED ON
0090 *
0100 SET WINDOW 'TESTE-JANELA'
0110 FOR #I 1 5
0120   WRITE NOTITLE 'TESTE JANELA' #I
0130 END-FOR
0140 END
```

2.3. Atributos Dinâmicos

Os atributos dinâmicos são definidos com o parâmetro DY.

Utiliza caracteres especiais para indicar o início e o fim de uma string que terá atributos diferentes dentro de um texto.

Possibilita o controle dinâmico de:

- Tamanho da string de um texto na tela;
- Intensificação;
- Proteção especial do texto designado.

```
       $XXXXXXXX                Politec Informática Ltda.                $MMMMMMMMM
       $XXXXXXXX               @Cadastro@de@Funcionários                   $XXXXXXXX
        ---------------------- Funcionários - Incluir -------------------------------

        Código.............:.eXXX$XXX

        Nome................:&XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

        Endereço............:&XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX




       $XXXXXXXXX&X$XXXXXXXXXXXXXX
        ------------------------------------------------------------------------------
       @F1 Ajuda @F3 Sai @F4 Seleciona(+) @F5 Encerra
001    --010---+----+----+---030---+----+----+---050---+----+----+---070---+----
```

```
       Fld #CD-FUNC                                                          Fmt A6
       ------------------------------------------------------------------------------
       AD= MDLT'_'___     ZP=         SG=        HE= +_____________________   Rls 0
       AL= _____          CD= __      CV= _____________________________________   Mod User
       PM= __  DF=                    DY= ?_____________________________
       EM= ____________________________________________________________

       001   --010---+----+----+---030---+----+----+---050---+----+----+---070---+----
       $XXXXXXXX                Politec Informática Ltda.                $MMMMMMMMM
       $XXXXXXXX               @Cadastro@de@Funcionários                   $XXXXXXXX
        ---------------------- Funcionários - Incluir -------------------------------

        Código.............:.EXXXXX$XXX

        Nome................:&XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

        Endereço............:&XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX


       Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
             HELP  Mset  Exit  <---  --->  --    -     +           <     >     Let
```

```
       15:42:20                  Design Dynamic Attributes               19/11/2004
       ------------------------------------------------------------------------------
       #_  Specify escape character                D   Default/Dark ....... _
           (special character or hex value)        I   Intensified ........ x
                                                   B   Blinking ........... _
           and select one attribute                N   Nondisplay ......... _
                                                   V   Reversed Video ..... _
                                                   U   Underlined ......... _
                                                   C   Cursive/Italic ..... _

           and/or one color                        NE  Neutral ............ _
                                                   RE  Red ................ _
```

```
                                                BL  Blue ............... x
                                                GR  Green .............. _
                                                YE  Yellow ............. _
                                                PI  Pink ............... _
                                                TU  Turquoise .......... _

    and/or write protection                     P   Protect ............ _
-------------------------------------------------------------------------------
@_  to finish specify stop character         DY=
```

```
Fld #CD-FUNC                                                          Fmt A6
-------------------------------------------------------------------------------
AD= MDLT'_'___     ZP=         SG=        HE= +_____________________    Rls 0
AL= _____          CD= __      CV= _____________________________     Mod User
PM= __  DF=                    DY= #BLI@_____________________________
EM= ____________________________________________________________

001   --010---+----+----+---030---+----+----+---050---+----+----+---070---+----
$XXXXXXXX                Politec Informática Ltda.                $MMMMMMMMM
$XXXXXXXX               @Cadastro@de@Funcionários                   $XXXXXXXX
 ---------------------- Funcionários - Incluir -------------------------------

 Código.............:.EXXXXX$XXX

 Nome................:&XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

 Endereço............:&XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      HELP  Mset  Exit  <---  --->  --    -     +           <     >     Let
```

2.4. Variável de Controle

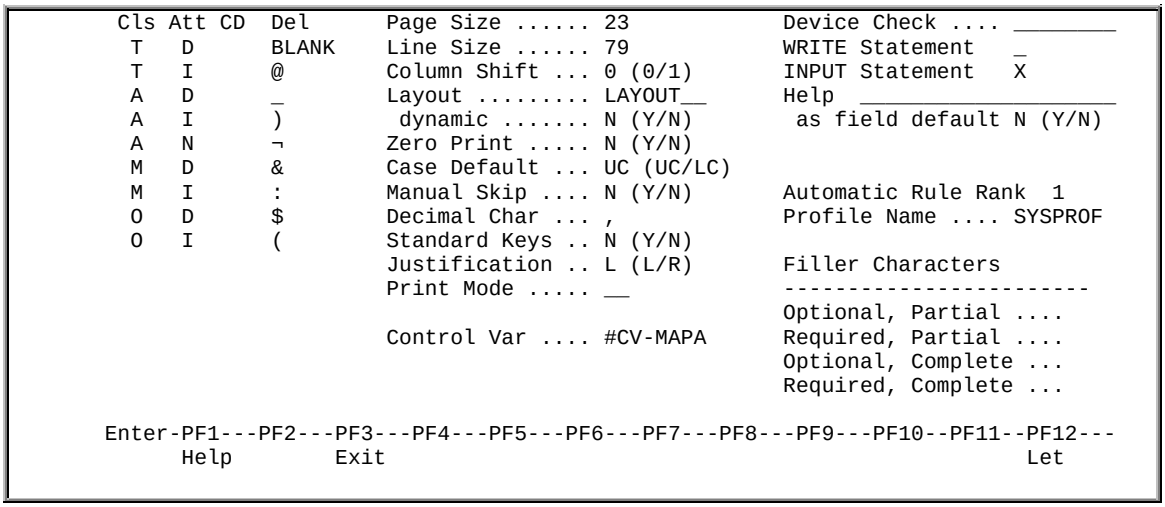
Além de proporcionar definição dinâmica de atributos para o campo, a variável de controle serve também para checar se algum campo sofreu alguma modificação.

```
IF #CV-MAPA MODIFIED

        Ou

IF #CV-MAPA NOT MODIFIED
```

Poderá ser verdadeiro ou falso, dependendo se o(s) campo(s) for(em) modificado(s) após a transmissão para a tela do terminal.

A definição do campo #CV-MAPA na profile do mapa permite testar se houve alguma alteração em qualquer campo do mapa.

```
15:52:04              Define Map Settings for MAP                19/11/2004

Delimiters           Format                          Context
-----------------    --------------------------      -------------------------
```



2.5. Regras de Validação

São blocos de instruções Natural que podem ser incorporados a uma variável do mapa ou à variável de sistema *PF-KEY.

As aplicações que utilizam Regras de Validação possuem uma estrutura mais legível, pois a validação dos campos é efetuada pelo mapa, ficando no programa apenas o processo principal.

As características principais do uso das Regras de Validação são:

- Programas mais legíveis;
- Facilidades de manutenção;
- Padrão de consistência;
- Consistências logicamente independentes, com níveis de validação bem definidos;
- Facilidades no desenvolvimento.

As regras de validação possuem uma ordem de processamento:

- Rank;
- Posição do campo na tela. A ordem é da esquerda para a direita e de cima para baixo. As PF-KEYs são consideradas como primeira posição.

Algumas considerações sobre o Rank:

- Pode ser de 0 (primeiro) a 99 (último);
- Várias regras podem ter o mesmo Rank;
- O Rank das regras pode ser alterado pelo comando P=nn.

Eles são invocados por:

- ..p para regras relacionadas às PF-KEYs;
- .p no campo para regras relacionadas ao campo;
- p digitado na coluna "CMD" da edição estendida de campo (Extended Field Edting Summary).

O "&" pode substituir o nome de uma variável nas regras. Pode também substituir o nome da view (qualificador) de um campo de banco de dados nas regras.

As regras podem ser removidas via comando UNLINK.

A instrução REINPUT deve ser codificada sempre que possível dentro das Regras de Validação, pois, desta forma seu tempo de processamento é reduzido. Esta instrução num programa pode provocar mais de uma carga do mapa no Buffer Pool.

Recomenda-se fazer o máximo de validações no mapa. Porém, deve-se evitar validações que envolvam acessos complexos à Base de Dados.

3. PROGRAMAS

Os programas tem execução independente, acionada pela digitação do nome do módulo na linha de comando do Natural, ou através de chamadas FETCH.

Devem sempre estar no primeiro nível de execução de um sistema, pois é a partir deles que podemos dar início às suas transações.

Podemos criar programas que sejam parte de uma grande função (como se fosse uma subrotina) e sua execução deve ser feita através de FETCH RETURN. Esta situação pode ocorrer se tivermos, por exemplo, uma função que ora faça parte de uma opção de menu e, num outro ponto do sistema, seja parte de uma outra função de menu onde termos então um outro programa.

Devemos ter como objetivo programas enxutos, legíveis, modularizados, de forma tal que a manutenção seja rápida sem prejudicar a performance.

4. SUB-PROGRAMAS

São módulos que não podem ser executados independentemente. Sua chamada é através do comando CALLNAT.

Os sub-programas devem utilizar a área de dados global disponível para resolver problemas complexos de armazenagem de dados em memória, já que não tem acesso aos dados globais ativos no sistema.

Normalmente utilizados como rotinas padronizadas que podem ser chamadas de qualquer ponto do sistema, os sub-programas têm como característica favorável o fato de não obrigarem a recatalogação dos módulos que o referenciam em caso de alteração no seu processamento. Esta recatalogação só se faz necessário se houver inclusão e/ou exclusão de campos na área de parâmetros do sub-programa (PARAMETER DATA AREA).

A passagem de dados para os sub-programas é muito eficiente, pois não há movimentação de campos, apenas o endereço em memória é passado.

Há que considerar um OVERHEAD nas chamadas via CALLNAT, pois o Natural tem que controlar as áreas dos objetos ativos, ocupando para isso uma parte da USIZE, e também a ordem de processamento do empilhamento ocasionado por várias chamadas CALLNAT.

Programa Chamador: XXXPSOMA

```
0010 DEFINE DATA LOCAL
0020 1 #NUM1     (N2) INIT<25>
0030 1 #NUM2     (N2) INIT<35>
0040 1 #SOMA     (N5)
0050 END-DEFINE
0060 *
0070 CALLNAT 'XXXNSOMA' #NUM1 #NUM2 #SOMA
0080 *
0090 WRITE '=' #NUM1 / '=' #NUM2 // '=' #SOMA
0100 *
0110 END
```

Subprograma Chamado: XXXNSOMA

```
0010 DEFINE DATA PARAMETER
0020 1 #A        (N2)
0030 1 #B        (N2)
0040 1 #S        (N5)
0050 END-DEFINE
0060 *
0070 #S := #A + #B
0080 END
```

Resultado:

```
#NUM1:  25
#NUM2:  35

#SOMA:     60
```

5. SUB-ROTINA EXTERNA

Assim como o sub-programa, a sub-rotina externa não pode ser executada independentemente, pois deve ser acionada através do comando PERFORM.

A grande novidade é a passagem de parâmetros no próprio comando PERFORM, o que só era possível através da GDA (*Global Data Area).*

Os dados são recebidos na sub-rotina, através de uma PDA (*Parameter Data Area)* interna ou externa, onde os campos deverão estar na mesma ordem, formato e tamanho. Igualmente aos sub-programas, não há movimentação de campos e somente o endereço é passado.

Programa Chamador

```
0010 DEFINE DATA LOCAL
0020 1 #NUM1     (N2) INIT<25>
0030 1 #NUM2     (N2) INIT<35>
0040 1 #SOMA     (N5)
0050 END-DEFINE
0060 *
0070 PERFORM SOMA #NUM1 #NUM2 #SOMA
0080 *
0090 WRITE '=' #NUM1 / '=' #NUM2 // '=' #SOMA
0100 *
0110 END
```

Sub-Rotina Chamada

```
0010 DEFINE DATA PARAMETER
0020 1 #NUM1     (N2)
0030 1 #NUM2     (N2)
0040 1 #SOMA     (N5)
0050 END-DEFINE
0060 *
0070 DEFINE SUBROUTINE SOMA
0080 #SOMA := #NUM1 + #NUM2
0090 END-SUBROUTINE
0100 *
0110 END
```

Resultado:

```
#NUM1:  25
#NUM2:  35

#SOMA:     60
```

6. COPYCODE

São módulos que são inseridos no código objeto do fonte que está sendo catalogado, não causando, portanto, nenhum OVERHEAD em tempo de execução.

Os copycodes podem ser bem aproveitados para codificações que se façam necessárias em vários pontos do sistema, que tem a tendência de não sofrerem alterações com o passar dos tempos.

A desvantagem está que cada vez que se altera o COPYCODE a mudança não reflete nos programas que usam COPYCODE até uma nova recatalogação.

Programa Principal

```
0010 DEFINE DATA LOCAL
0020 1 #NUM1     (N2)
0030 1 #NUM2     (N2)
0040 1 #SOMA     (N5)
0050 END-DEFINE
0060 INCLUDE XXXC0001
0070 *
0080 INPUT (AD=MDLT'_' ZP=OFF) 'ENTRE COM UM NÚMERO...:' #NUM1 /
0090                           'ENTRE COM OUTRO NÚMERO:' #NUM2 // '-' (27) /
0100                           'PF3 - SAI'
0110 *
0120 WRITE '=' #NUM1 / '=' #NUM2
0130 *
0140 END
```

Copycode: XXXC0001

```
0010 SET KEY ALL
0020 SET KEY PF3 = 'XXXP9999'
```

Programa acionado pela Copycode: XXXP9999

```
0010 TERMINATE
0020 END
```

7. HELPROUTINES

É recomendável que todo sistema on-line tenha informações eficientes de help. Isto é facilmente implementado com o uso de helproutines e helpmaps.

Um helpmap é criado através da opção H do menu principal do Editor de mapas e é mantido como um outro mapa qualquer.

Exemplo de helpmaps:

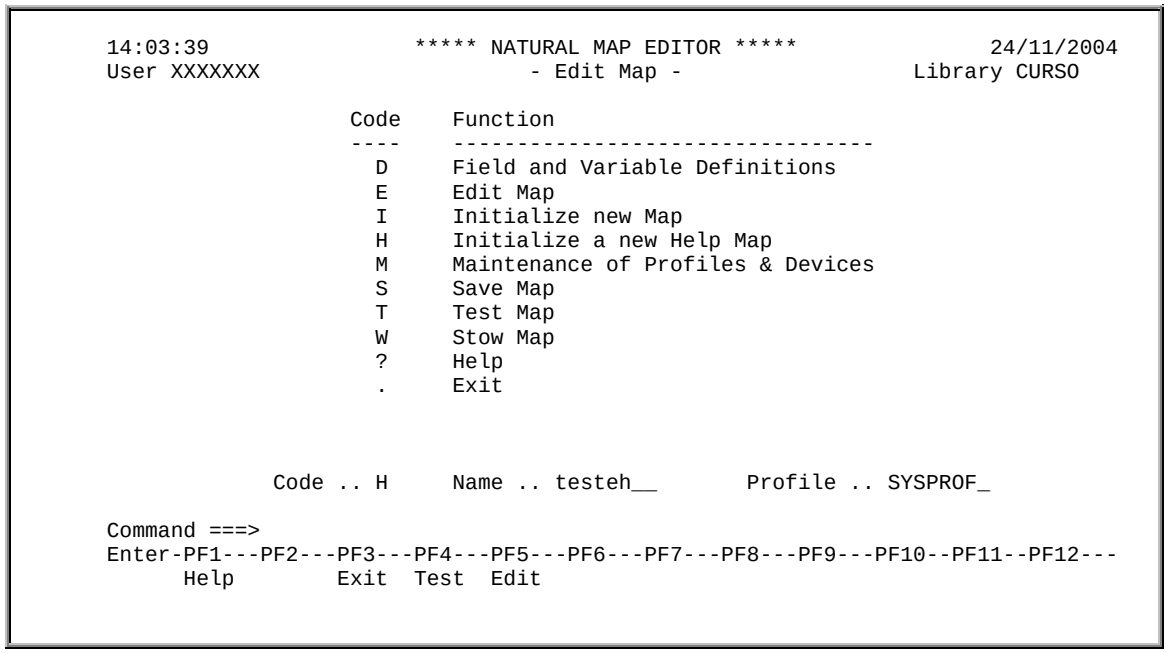
```
14:03:39                ***** NATURAL MAP EDITOR *****              24/11/2004
User XXXXXXX                       - Edit Map -                  Library CURSO

                     Code    Function
                     ----    --------------------------------
                       D     Field and Variable Definitions
                       E     Edit Map
                       I     Initialize new Map
                       H     Initialize a new Help Map
                       M     Maintenance of Profiles & Devices
                       S     Save Map
                       T     Test Map
                       W     Stow Map
                       ?     Help
                       .     Exit



                Code .. H     Name .. testeh__       Profile .. SYSPROF_

Command ===>
```

```
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit  Test  Edit
```

```
14:06:38              Define Map Settings for HELPMAP              24/11/2004

 Delimiters           Format                          Context
 -----------------    --------------------------      -------------------------
 Cls Att CD  Del      Page Size ...... 5              Device Check .... ________
  T   D      BLANK    Line Size ...... 30             WRITE Statement   _
  T   I      ?        Column Shift ... 0 (0/1)        INPUT Statement   X
  A   D      _        Layout ......... ________       Help
  A   I      )         dynamic ....... N (Y/N)         as field default   (Y/N)
  A   N      ¬        Zero Print ..... N (Y/N)        Position Line    Col
  M   D      &        Case Default ... UC (UC/LC)
  M   I      :        Manual Skip .... N (Y/N)        Automatic Rule Rank  1
  O   D      +        Decimal Char ... ,              Profile Name .... SYSPROF
  O   I      (        Standard Keys .. N (Y/N)
                      Justification .. L (L/R)        Filler Characters
                      Print Mode ..... __             ----------------------
                                                      Optional, Partial ....
                      Control Var .... ________       Required, Partial ....
                                                      Optional, Complete ...
                                                      Required, Complete ...

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit                                                  Let
```

```
                                ..............................................
                                ..............................................
      Exemplo de Helpmap        ..............................................
                                ..............................................
                                ..............................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
.....................................................................................
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Mset  Exit  Test  Edit  --    -     +     Split <     >     Let
001   --010---+----+----+---030---+----+----+---050---+----+----+---070---+----
```

Exemplo de help de campos:



Uma helproutine é um tipo especial de sub-programa que pode ter acesso a área de dados global. É criada através do editor de programas, especificando-se SET TYPE HELPROUTINE.

7.1. Utilização

- Fornecer informações a respeito do conteúdo/preenchimento de um determinado campo.
- Exibir os valores possíveis de serem colocados no campo, com opção de seleção de um dos valores apresentados.
- Apresentar informações detalhadas de um campo, considerando o seu valor.

7.2. Definição

- Helpmaps menores que o mapa principal são automaticamente colocados em windows.
- Helpmaps menores que o mapa principal devem ser controlados pela rotina.
- O Natural controla automaticamente o posicionamento da window

7.3. Passagem de Controle

- O nome da rotina deve ser especificado:
  - No nível de mapa na profile do mapa
  - No nível de campo no parâmetro HE do campo
- A helproutine é invocada:

- Digitando-se “X” no campo
- Pressionando-se a tecla de função do help
- Com REINPUT USING HELP

7.4. Passagem de Dados

- Através da GDA (*Global Data Area)*
- O campo ao qual está se requerendo o Help é enviado automaticamente
- Além desse campo um outro pode ser passado. Deve ser especificado junto com o nome da rotina na profile do mapa no parâmetro HE
- A helproutine deve ter uma PDA para ambos os campos

7.5. Passagem de Parâmetros

O campo que está associado a uma Helproutine não pode ser alterado na tela antes da chamada da mesma, pois isso faria com que a helproutine não fosse acionada.

Um campo que esteja na tela e tenha seu valor alterado antes da chamada da helproutine não terá esta alteração refletida na mesma, ainda que se faça parte da GDA (*Global Data Area*) e/ou seja passado como parâmetro adicional.

Igualmente, se este campo tiver seu valor alterado dentro da helproutine, no retorno da mesma, esta alteração não será refletida na tela, que voltará com os valores originais anteriores à chamada do help.

Para entender este funcionamento é necessário explicar o que ocorre quando o controle é passado para uma helproutine.

Supondo uma tela onde já foram preenchidos alguns valores de campos e então é digitado um “?” em um campo associado a uma helproutine. Neste momento, a tela é congelada e os valores já preenchidos não são movidos para suas respectivas posições de memória. O controle é então transferido para a helproutine. No retorno, a tela que havia sido congelada é restaurada com os valores originais dos campos.

Devido à forma de funcionamento das helproutines, toda vez que as mesmas são acionadas, o Natural emite comandos ADABAS de atualização para seus arquivos de controle.

Desta forma, o uso indiscriminado de helproutines, que viciem seu acionamento pelos usuários finais, deve ser evitado, pois causaria um overhead que não pode ser desconsiderado.

7.6. Considerações

Quando da definição de um sistema, algumas funções aparecem em mais de um ponto do processamento. Se essas funções são dependentes de outras, como por exemplo, cálculo de dígito verificador, cálculo de vencimento de faturas, etc., poderemos adotar os seguintes critérios:

- Se esta função pode sofrer alterações com o passar do tempo (devido à legislação e outros motivos), é necessário que se faça opção por subprogramas ou subrotinas externas.
- Se a função não sofre modificações com o passar do tempo, por exemplo, ações a serem tomadas em caso de ON ERROR, pode-se optar por uma codificação através de copycode, que não tem OVERHEAD nenhum em tempo de execução.

O uso de Helproutine deve ser considerado quando a comunicação for feita com mapas. Há que se saber que o uso de helproutines tem um overhead a ser considerado, pois o Natural faz I/O ao System File para congelar informações da tela sobreposta pela tela helproutine.

8. MANIPULAÇÃO DE DADOS

8.1. Operações Aritméticas

Os operandos dos comandos ADD, SUBTRACT, MULTIPLY, DIVIDE e COMPUTE podem ser ARRAYs aritméticos. Isto evita a codificação de loops em vários casos.

- Operação aritmética com todas a ocorrências de array:

```
0010 DEFINE DATA LOCAL
0020 1 #A           (N2/3,2) INIT
0030   (1,1) <1> (1,2) <1>
0040   (2,1) <2> (2,2) <0>
0050   (3,1) <3> (3,2) <2>
0060 1 #B           (N2/3,2) INIT
0070   (1,1) <3> (1,2) <6>
0080   (2,1) <4> (2,2) <8>
0090   (3,1) <8> (3,2) <3>
0100 END-DEFINE
0110 *
0120 WRITE 'VALORES INICIAIS DE #A' / '-'(22)
0130 WRITE '=' #A(1,1) '=' #A(1,2) /
0140       '=' #A(2,1) '=' #A(2,2) /
0150       '=' #A(3,1) '=' #A(3,2)
0160 *
0170 WRITE / 'VALORES INICIAIS DE #B' / '-'(22)
0180 WRITE '=' #B(1,1) '=' #B(1,2) /
0190       '=' #B(2,1) '=' #B(2,2) /
0200       '=' #B(3,1) '=' #B(3,2)
0210 *
0220 ADD #A(*,*)  TO  #B(*,*)
0230 *
0240 WRITE / 'VALORES FINAIS DE #B' / '-'(20)
0250 WRITE '=' #B(1,1) '=' #B(1,2) /
0260       '=' #B(2,1) '=' #B(2,2) /
0270       '=' #B(3,1) '=' #B(3,2)
0280 END
```

Resultado:

```
VALORES INICIAIS DE #A
----------------------
#A:   1 #A:   1
#A:   2 #A:   0
#A:   3 #A:   2

VALORES INICIAIS DE #B
----------------------
#B:   3 #B:   6
#B:   4 #B:   8
#B:   8 #B:   3

VALORES FINAIS DE #B
--------------------
#B:   4 #B:   7
#B:   6 #B:   8
#B:  11 #B:   5
```

- Operação aritmética com uma linha do array:

```
0010 DEFINE DATA LOCAL
0020 1 #A            (N2/3,2) INIT
0030   (1,1) <1> (1,2) <1>
0040   (2,1) <2> (2,2) <0>
0050   (3,1) <3> (3,2) <2>
0060 1 #B            (N2/3,2) INIT
0070   (1,1) <3> (1,2) <6>
0080   (2,1) <4> (2,2) <8>
0090   (3,1) <8> (3,2) <3>
0100 1 #C            (N2) INIT<6>
0110 END-DEFINE
0120 *
0130 WRITE 'VALORES INICIAL DE #C' / '-'(21)
0140 WRITE '=' #C /
0150 *
0160 WRITE 'VALORES INICIAIS DE #A' / '-'(22)
0170 WRITE '=' #A(1,1) '=' #A(1,2) /
0180       '=' #A(2,1) '=' #A(2,2) /
0190       '=' #A(3,1) '=' #A(3,2)
0200 *
0210 WRITE / 'VALORES INICIAIS DE #B' / '-'(22)
0220 WRITE '=' #B(1,1) '=' #B(1,2) /
0230       '=' #B(2,1) '=' #B(2,2) /
0240       '=' #B(3,1) '=' #B(3,2)
0250 *
0260 ADD #C  TO  #B(1,*)
0270 *
0280 WRITE / 'VALORES FINAIS DE #B' / '-'(20)
0290 WRITE '=' #B(1,1) '=' #B(1,2) /
0300       '=' #B(2,1) '=' #B(2,2) /
0310       '=' #B(3,1) '=' #B(3,2)
0320 END
```

Resultado:

```
VALORES INICIAL DE #C
---------------------
#C:   6

VALORES INICIAIS DE #A
----------------------
```

```
#A:   1 #A:   1
#A:   2 #A:   0
#A:   3 #A:   2

VALORES INICIAIS DE #B
----------------------
#B:   3 #B:   6
#B:   4 #B:   8
#B:   8 #B:   3

VALORES FINAIS DE #B
--------------------
#B:   9 #B:  12
#B:   4 #B:   8
#B:   8 #B:   3
```

- Operação aritmética com uma coluna do array:

```
0010 DEFINE DATA LOCAL
0020 1 #A           (N2/3,2) INIT
0030   (1,1) <1> (1,2) <1>
0040   (2,1) <2> (2,2) <0>
0050   (3,1) <3> (3,2) <2>
0060 END-DEFINE
0070 *
0080 WRITE 'VALORES INICIAIS DE #A' / '-'(22)
0090 WRITE '=' #A(1,1) '=' #A(1,2) /
0100       '=' #A(2,1) '=' #A(2,2) /
0110       '=' #A(3,1) '=' #A(3,2)
0120 *
0130 ADD 2  TO  #A(*,2)
0140 *
0150 WRITE / 'VALORES FINAIS DE #A' / '-'(20)
0160 WRITE '=' #A(1,1) '=' #A(1,2) /
0170       '=' #A(2,1) '=' #A(2,2) /
0180       '=' #A(3,1) '=' #A(3,2)
0190 END
```

Resultado:

```
VALORES INICIAIS DE #A
----------------------
#A:   1 #A:   1
#A:   2 #A:   0
#A:   3 #A:   2

VALORES FINAIS DE #A
--------------------
#A:   1 #A:   6
#A:   2 #A:   5
#A:   3 #A:   7
```

Os demais comandos aritméticos também possuem a mesma facilidade.

8.2. Passagem Variável de Parâmetros (Array)

Os Arrays podem ser definidos numa PDA (*Parameter Data Area*) com um número variável de ocorrências, utilizando-se a notação “1:V”. Exemplo:

PROGRAMA CHAMADOR

```
----------------

0010 DEFINE DATA LOCAL
0020 1 #TAB            (A2/3,2) INIT
0030                                (1,1) <'11'> (1,2) <'12'>
0040                                (2,1) <'21'> (2,2) <'22'>
0050                                (3,1) <'31'> (3,2) <'32'>
0060 1 #I              (N1) INIT<3>
0070 1 #J              (N1) INIT<2>
0080 END-DEFINE
0090 *
0100 CALLNAT 'NARRAY2' #TAB(*,*) #I #J
0110 END
```

SUBPROGRAMA

```
-----------

0010 DEFINE DATA PARAMETER
0020 1 #X              (A2/1:V,1:V)
0030 1 #I              (N1)
0040 1 #J              (N1)
0050 LOCAL
0060 1 #A              (N1)
0070 1 #B              (N1)
0080 END-DEFINE
0090 *
0100 FOR #A 1 TO #I
0110   FOR #B 1 TO #J
0120     WRITE #X(#A,#B)
0130   END-FOR
0140 END-FOR
0150 END
```

9. Manipulação de Strings/Array

9.1. Instrução EXAMINE

A instrução EXAMINE pesquisa uma string em um campo, alfanumérico ou array, através de uma opção, e altera, deleta ou conta a quantidade de ocorrência de um campo.

9.1.1. Opção FULL

Com a opção FULL, o caracter " " (espaço em branco) é considerado como um outro qualquer. Sem esta opção, o comando EXAMINE não considera os brancos finais do campo. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A        (A15) INIT<'XYZ'>
0030 1 #B        (A10) INIT<'     XPTO1'>
0040 1 #CONT     (N2)
0050 END-DEFINE
0060 *
0070 EXAMINE #A FOR ' ' GIVING NUMBER #CONT
0080 WRITE 'QUANTIDADE DE ESPAÇO EM BRANCO SEM FULL:' #CONT
```

```
0090 *
0100 EXAMINE FULL #A FOR ' ' GIVING NUMBER #CONT
0110 WRITE 'QUANTIDADE DE ESPAÇO EM BRANCO COM FULL:' #CONT
0120 *
0130 EXAMINE #B FOR ' ' GIVING NUMBER #CONT
0140 WRITE 'QUANTIDADE DE ESPAÇO EM BRANCO SEM FULL:' #CONT
0150 END
```

Resultado:

```
QUANTIDADE DE ESPAÇO EM BRANCO SEM FULL:   0
QUANTIDADE DE ESPAÇO EM BRANCO COM FULL:  12
QUANTIDADE DE ESPAÇO EM BRANCO SEM FULL:   5
```

9.1.2. Opção DELETE/REPLACE

Com a opção DELETE/REPLACE é possível pesquisar o campo pelo valor especificado e excluir ou substituir as ocorrências encontradas pelo valor desejado. Se for especificado FIRST para qualquer das opções, apenas a primeira ocorrência encontrada será excluída ou substituída. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A        (A15) INIT<'XYZ'>
0030 1 #B        (A15) INIT<'XYZ'>
0040 END-DEFINE
0050 *
0060 EXAMINE #A FOR 'Y' DELETE
0070 WRITE '#A DEPOIS DO EXAMINE / DELETE.:' #A
0080 *
0090 EXAMINE #B FOR 'Y' REPLACE ' '
0100 WRITE '#B DEPOIS DO EXAMINE / REPLACE:' #B
0110 END
```

Resultado:

```
#A DEPOIS DO EXAMINE / DELETE.: XZ
#B DEPOIS DO EXAMINE / REPLACE: X Z
```

Observe que DELETE e REPLACE com “ “ funcionam de forma diferente conforme o exemplo.

Se for especificado FULL na opção REPLACE, os brancos no final do campo utilizado para substituição também serão contados. Portanto, o campo fonte pode não suportar o REPLACE devido ao tamanho. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A        (A8) INIT<'ABCDEFGH'>
0030 END-DEFINE
0040 *
0050 EXAMINE #A FOR 'B' AND REPLACE WITH FULL 'X '
0060 WRITE '#A DEPOIS DO EXAMINE / DELETE.:' #A
```

```
0070 END
```

Resultado:

```
Erro: NAT1308 Replace string does not fit into variable.
```

Nesta situação ocorreria uma mensagem de erro informando que o valor da substituição não cabe na variável.

Senão fosse especificado FULL o EXAMINE seria executado e o valor de #A seria "AXCDEFG"..

```
0010 DEFINE DATA LOCAL
0020 1 #A        (A8) INIT<'ABCDEFGH'>
0030 END-DEFINE
0040 *
0050 EXAMINE #A FOR 'B' AND REPLACE 'X '
0060 WRITE '#A DEPOIS DO EXAMINE / DELETE.:' #A
0070 END
```

Resultado:

```
#A DEPOIS DO EXAMINE / DELETE.: AXCDEFGH
```

9.1.3. Opção GIVING NUMBER

A opção GIVING NUMBER fornece o número de ocorrências do valor pesquisado no campo fonte. Esta informação refere-se ao campo ante das possíveis operações com as opções REPLACE ou DELETE. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A        (A15) INIT<'XYZXYZXYZXYZXYZ'>
0030 1 #CONT     (N2)
0040 END-DEFINE
0050 *
0060 EXAMINE #A FOR 'Z' REPLACE WITH 'A' GIVING NUMBER #CONT
0070 WRITE '#A DEPOIS DO EXAMINE / REPLACE:' #A //
0080       'NUMERO DE OCORRÊNCIA DE B.....:' #CONT
0090 END
```

Resultado:

```
#A DEPOIS DO EXAMINE / REPLACE: XYAXYAXYAXYAXYA

NUMERO DE OCORRÊNCIA DE B.....:   5
```

9.1.4. Opção GIVING POSITION

A opção GIVING POSITION fornece a posição da primeira ocorrência do valor pesquisado no campo fonte. Esta posição refere-se ao campo antes de qualquer operação com as opções REPLACE / DELETE. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A        (A20) INIT<'XYZXYZXYZXYZXYZXYZXY'>
0030 1 #CONT     (N2)
0040 END-DEFINE
0050 *
0060 EXAMINE #A FOR 'YZ' DELETE GIVING POSITION #CONT
0070 WRITE '#A DEPOIS DO EXAMINE / DELETE........:' #A //
0080       'POSIÇÃO DO PRIMEIRO B ANTES DO DELETE:' #CONT
0090 END
```

Resultado:

```
#A DEPOIS DO EXAMINE / DELETE........: XXXXXXXY

POSIÇÃO DO PRIMEIRO B ANTES DO DELETE:   2
```

9.1.5. Opção GIVING LENGTH

A opção GIVING LENGTH fornece o tamanho do campo pesquisado depois que todas as operações de REPLACE / DELETE forem feitas. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A        (A10) INIT<'XYZX'>
0030 1 #CONT     (N2)
0040 END-DEFINE
0050 *
0060 EXAMINE #A FOR 'Z' REPLACE 'K K' GIVING LENGTH #CONT
0070 WRITE '#A DEPOIS DO EXAMINE / REPLACE.......:' #A //
0080       'O NUMERO DE CARACTERES APOS O REPLACE:' #CONT
0090 END
```

Resultado:

```
#A DEPOIS DO EXAMINE / REPLACE.......: XYK KX

O NUMERO DE CARACTERES APÓS O REPLACE:   6
```

9.1.6. Opção GIVING INDEX

Com a opção GIVING INDEX é possível localizar um determinado valor em um ARRAY e fornecer a primeira ocorrência do ARRAY em que o valor de pesquisa foi encontrado. Esta opção também é válida para arrays multidimensionais. Exemplo 1:

```
0010 DEFINE DATA LOCAL
0020 1 #TAB      (A1/5,3,3) INIT
0030   (1,2,3) <'A'> (5,1,2) <'B'>
0040 1 #LINHA    (N2)
```

```
0050 1 #COLUNA   (N2)
0060 1 #PLANO    (N2)
0070 END-DEFINE
0080 *
0090 EXAMINE #TAB(*,*,*) FOR 'A' GIVING INDEX #LINHA #COLUNA #PLANO
0100 WRITE 'A OCORRÊNCIA DO VALOR "A" é' //
0110       'LINHA.:' #LINHA  /
0120       'COLUNA:' #COLUNA /
0130       'PLANO.:' #PLANO  /
0140 *
0150 EXAMINE #TAB(*,*,*) FOR 'B' GIVING INDEX #LINHA #COLUNA #PLANO
0160 WRITE 'A OCORRÊNCIA DO VALOR "B" é' //
0170       'LINHA.:' #LINHA  /
0180       'COLUNA:' #COLUNA /
0190       'PLANO.:' #PLANO  /
0200 END
```

Resultado:

```
A OCORRÊNCIA DO VALOR 'A' é

LINHA.:   1
COLUNA:   2
PLANO.:   3

A OCORRÊNCIA DO VALOR 'B' é

LINHA.:   5
COLUNA:   1
PLANO.:   2
```

Exemplo 2:

```
0010 DEFINE DATA LOCAL
0020 1 #MARCA      (A10/5) INIT <'FIAT','CITROEN','VOLVO','AUDI','FORD'>
0030 1 #VEICULO    (A15/5) INIT <'PALIO','PICASSO','REVOLUTION','A6','KA'>
0040 *
0050 1 #MARCA_AUX  (A10)
0060 1 #INDICE     (N1)
0070 END-DEFINE
0080 *
0090 #MARCA_AUX := 'AUDI'
0100 *
0110 EXAMINE #MARCA(*) #MARCA_AUX GIVING INDEX #INDICE
0120 DISPLAY #MARCA(*) #VEICULO(*) #MARCA_AUX #INDICE #MARCA(#INDICE)
0130   #VEICULO(#INDICE)
0140 END
```

Resultado:

```
#MARCA      #VEICULO     #MARCA_AUX #INDICE   #MARCA      #VEICULO
---------- --------------- ---------- ------- ---------- ---------------

FIAT       PALIO           AUDI        4      AUDI       A6
CITROEN    PICASSO
VOLVO      REVOLUTION
AUDI       A6
FORD       KA
```

9.1.7. Opção SUBSTRING

A opção SUBSTRING permite que uma porção do campo seja examinada. Depois do nome (*operando1*), deve ser especificada a primeira posição a ser examinada (*operando2*) e então o tamanho (*operando3*) da parte do campo a ser pesquisada. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #D          (A26) INIT <'TESTECOMANDOEXAMINENOCURSO'>
0030 1 #NUMBER     (N2)
0040 1 #POSICAO    (N2)
0050 END-DEFINE
0060 *
0070 EXAMINE SUBSTRING(#D,13,7) 'MINE' GIVING #NUMBER POSITION #POSICAO
0080 WRITE #D / '=' #NUMBER / '=' #POSICAO
0090 END
```

Resultado:

```
TESTECOMANDOEXAMINENOCURSO
#NUMBER:   1
#POSICAO:   4
```

Na opção SUBSTRING pode-se codificar apenas a posição do campo onde a pesquisa deve iniciar. Neste caso, a procura começará na posição determinada e seguirá até o final do campo. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #D          (A26) INIT <'TESTECOMANDOEXAMINENOCURSO'>
0030 1 #NUMBER     (N2)
0040 1 #POSICAO    (N2)
0050 END-DEFINE
0060 *
0070 EXAMINE SUBSTRING(#D,13,7) 'MINE' GIVING #NUMBER POSITION #POSICAO
0080 WRITE #D / '=' #NUMBER / '=' #POSICAO
0090 END
```

Resultado:

```
TESTECOMANDOEXAMINENOCURSO
#NUMBER:   1
#POSICAO:   4
```

Processamento de elementos de um ARRAY utilizando a opção SUBSTRING. Exemplo:

```
DEFINE DATA LOCAL
1 #NOME       (A15/8) INIT
  <'LUCIANO','LUIZA','ANA','LUCIA','PERDIGAO','ROSA','MONICA','LUCAS'>
1 #NUMBER     (N2)
END-DEFINE
*
EXAMINE SUBSTRING(#NOME(*),1,2) 'LU' GIVING #NUMBER
```

```
WRITE #NOME(*) / '=' #NUMBER
END

Resultado:

LUCIANO         LUIZA           ANA             LUCIA           PERDIGAO
ROSA            MONICA          LUCAS
#NUMBER:   4
```

9.1.8. Opção PATTERN

A opção PATTERN tem a capacidade de pesquisar um campo excluindo posições selecionadas.

Os caracteres PATTERN "." (ponto), "?" (interrogação) e "_" (underscore), representam uma posição a ser ignorada. Os caracteres "*" (asterisco) e "%" (percentual), indicam qualquer número de posições a serem ignoradas. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #TAB         (A10/4) INIT<'CARRO','CARROÇA','GARRAFA','CAVALO'>
0030 1 #I           (N3)
0040 1 #N           (N3)
0050 END-DEFINE
0060 *
0070 FOR #I 1 4
0080   EXAMINE #TAB(#I) PATTERN 'A.O' GIVING NUMBER #N
0090   WRITE #TAB(#I) '-' #N
0100 END-FOR
0110 *
0120 SKIP 1
0130 *
0140 FOR #I 1 4
0150   EXAMINE #TAB(#I) PATTERN 'A*O' GIVING NUMBER #N
0160   WRITE #TAB(#I) '-' #N
0170 END-FOR
0180 END

Resultado:

CARRO      -    0
CARROÇA    -    0
GARRAFA    -    0
CAVALO     -    1

CARRO      -    1
CARROÇA    -    1
GARRAFA    -    0
CAVALO     -    1
```

9.1.9. Opção WITH DELIMITERS

A pesquisa tem como default o "*absolute scan*", ou seja, o resultado fornecido pelo EXAMINE é dado após uma pesquisa em todo o campo.

A opção WITH DELIMITERS é utilizada para um valor que está entre delimitadores " " (espaço em branco), ou qualquer outro que não seja letra ou número.

WITH DELIMITERS *operando* é utilizado para procurar um valor delimitado pelo caracter especificado pelo caracter especificado no *operando*. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #T           (A40) INIT<'-A-BC A B C  .A. .B. .C.  -A- -B-AB'>
0030 1 #P1          (N3)
0040 1 #P2          (N3)
0050 1 #P3          (N3)
0060 END-DEFINE
0070 *
0080 EXAMINE #T 'A' GIVING NUMBER #P1
0090 EXAMINE #T 'A' WITH DELIMITERS GIVING NUMBER #P2
0100 EXAMINE #T 'A' WITH DELIMITERS '-' GIVING NUMBER #P3
0110 *
0120 WRITE '=' #P1 / '=' #P2 / '=' #P3
0130 END
```

Resultado:

```
#P1:    5
#P2:    4
#P3:    2
```

9.1.10. Opção TRANSLATE

A opção TRANSLATE é utilizada para traduzir caracteres de um campo em maiúsculo ou minúsculo, ou em outro caracter especificado.

Se for especificado INTO LOWER CASE, o conteúdo do operando utilizado na pesquisa será traduzido para minúsculo.

Se for especificado INTO UPPER CASE, o conteúdo do operando utilizado na pesquisa será traduzido para maiúsculo.

A sintaxe EXAMINE ... TRANSLATE USING *operando*, tem como valores de tradução os caracteres contidos no *operando*, que é uma tabela de tradução definida com formato (A2) ou (B2).

Quando EXAMINE encontrar o primeiro caracter da tabela no campo, faz a troca do caracter do campo para o segundo caracter definido na tabela.

A opção INVERTED altera a direção de troca no *operando*. Exemplo:

```
0010 DEFINE DATA LOCAL
```

```
0020 1 #T           (A40)  INIT<'-A-BC A B C  .A. .B. .C.  -A- -B-AB'>
0030 1 #T1          (A2/3) INIT<'A1','B2','C3'>
0040 *
0050 1 #NOME        (A8)   INIT<'PERDIGÃO'>
0060 END-DEFINE
0070 *
0080 WRITE 'VALOR DE #T ANTES PRIMEIRO EXAMINE:' #T
0090 EXAMINE #T TRANSLATE USING #T1(*)
0100 *
0110 WRITE 'VALOR DE #T ANTES SEGUNDO EXAMINE.:' #T
0120 EXAMINE #T TRANSLATE USING INVERTED #T1(*)
0130 *
0140 WRITE 'VALOR DE #T FINAL.................:' #T
0150 *
0160 SKIP 1
0170 WRITE 'NOME ANTES DO EXAMINE.............:' #NOME
0180 EXAMINE #NOME TRANSLATE INTO LOWER CASE
0190 WRITE 'NOME APÓS EXAMINE.................:' #NOME
0200 END
```

Resultado:

```
VALOR DE #T ANTES PRIMEIRO EXAMINE: -A-BC A B C  .A. .B. .C.  -A- -B-AB
VALOR DE #T ANTES SEGUNDO EXAMINE.: -1-23 1 2 3  .1. .2. .3.  -1- -2-12
VALOR DE #T FINAL.................: -A-BC A B C  .A. .B. .C.  -A- -B-AB

NOME ANTES DO EXAMINE.............: PERDIGÃO
NOME APÓS EXAMINE.................: perdigÃo
```

9.2. Instrução SEPARATE

Separa o conteúdo de um campo alfanumérico em dois ou mais campos alfanuméricos ou ocorrências de um ARRAY alfanumérico. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A30) INIT<'XXXXXXXXXXXXXXX.YYYY.ZZZZZZZZZ'>
0030 1 #A1          (A15)
0040 1 #A2          (A15)
0050 1 #A3          (A15)
0060 END-DEFINE
0070 *
0080 SEPARATE #A INTO #A1 #A2 #A3
0090 WRITE '=' #A // '=' #A1 / '=' #A2 / '=' #A3
0100 END
```

Resultado:

```
#A: XXXXXXXXXXXXXXX.YYYY.ZZZZZZZZZ

#A1: XXXXXXXXXXXXXXX
#A2: YYYY
#A3: ZZZZZZZZZ
```

Este exemplo mostra um SEPARATE onde há mais campos resultantes do que os possíveis a partir do campo fonte. Observe que o Natural anula o valor do campo excedente após o SEPARATE.

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A32) INIT<'1111111111 222'>
0030 1 #A1          (A10) INIT<'LUCIANO'>
0040 1 #A2          (A10) INIT<'LUCIANO'>
0050 1 #A3          (A10) INIT<'LUCIANO'>
0060 END-DEFINE
0070 *
0080 WRITE 'ANTES DO SEPARATE'
0090 WRITE '-----------------'
0100 WRITE '=' #A // '=' #A1 / '=' #A2 / '=' #A3
0110 SEPARATE #A INTO #A1 #A2 #A3
0120 SKIP 1
0130 WRITE 'APOS DO SEPARATE'
0140 WRITE '----------------'
0150 WRITE '=' #A1 / '=' #A2 / '=' #A3
0160 END
```

Resultado:

```
ANTES DO SEPARATE
-----------------
#A: 1111111111 222

#A1: LUCIANO
#A2: LUCIANO
#A3: LUCIANO

APOS DO SEPARATE
----------------
#A1: 1111111111
#A2: 222
#A3:
```

Este exemplo mostra um SEPARATE onde há tratamento de dois delimitadores consecutivos. O campo é anulado, ou seja, seu conteúdo é espaços em branco.

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A30) INIT<'111..1111'>
0030 1 #A1          (A20)
0040 1 #A2          (A20)
0050 1 #A3          (A20)
0060 END-DEFINE
0070 *
0080 SEPARATE #A INTO #A1 #A2 #A3
0090 WRITE '=' #A // '=' #A1 / '=' #A2 / '=' #A3
0100 END
```

Resultado:

```
#A: 111..1111

#A1: 111
#A2:
#A3: 1111
```

Este exemplo mostra um SEPARATE com manipulação de ARRAYS.

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A20) INIT<'LUCIANO PERDIGAO'>
0030 1 #ARRAY       (A10/2)
0040 END-DEFINE
0050 *
0060 SEPARATE #A INTO #ARRAY(*)
0070 WRITE '=' #A / '=' #ARRAY(*)
0080 END
```

Resultado:

```
#A: LUCIANO PERDIGAO
#ARRAY: LUCIANO    PERDIGAO
```

9.2.1. Opção WITH DELIMITERS

Este exemplo mostra o SEPARATE de um campo fonte em três campos resultantes, com utilização do delimitador "@".

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A32) INIT<'1111111111@2222222222@CCCCCCCCCC'>
0030 1 #A1          (A10)
0040 1 #A2          (A10)
0050 1 #A3          (A10)
0060 END-DEFINE
0070 *
0080 SEPARATE #A INTO #A1 #A2 #A3 WITH DELIMITERS '@'
0090 WRITE '=' #A // '=' #A1 / '=' #A2 / '=' #A3
0100 END
```

Resultado:

```
#A: 1111111111@2222222222@CCCCCCCCCC

#A1: 1111111111
#A2: 2222222222
#A3: CCCCCCCCCC
```

9.2.2. Opção GIVING NUMBER

A cláusula GIVING NUMBER fornece o número de campos resultantes diferentes de brancos. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A32) INIT<'1111111111.22233'>
0030 1 #A1          (A10) INIT<'LUCIANO'>
0040 1 #A2          (A10) INIT<'LUCIANO'>
0050 1 #A3          (A10) INIT<'LUCIANO'>
0060 1 #CONT        (N3)
```

```
0070 END-DEFINE
0080 *
0090 WRITE 'ANTES DO SEPARATE'
0100 WRITE '-----------------'
0110 WRITE '=' #A // '=' #A1 / '=' #A2 / '=' #A3
0120 SEPARATE #A INTO #A1 #A2 #A3 GIVING NUMBER #CONT
0130 SKIP 1
0140 WRITE 'APOS DO SEPARATE'
0150 WRITE '----------------'
0160 WRITE '=' #A1 / '=' #A2 / '=' #A3 //
0170   'NÚMEROS DE CAMPOS RESULTANTES:' #CONT
0180 END
```

Resultado:

```
ANTES DO SEPARATE
-----------------
#A: 1111111111.22233

#A1: LUCIANO
#A2: LUCIANO
#A3: LUCIANO

APOS DO SEPARATE
----------------
#A1: 1111111111
#A2: 22233
#A3:

NÚMEROS DE CAMPOS RESULTANTES:    2
```

9.2.3. Opção LEFT JUSTIFIED

A cláusula LEFT JUSTIFIED remove brancos à esquerda dos sub-campos. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A30) INIT<'111.  1111.             22233'>
0030 1 #A1          (A20)
0040 1 #A2          (A20)
0050 1 #A3          (A20)
0060 END-DEFINE
0070 *
0080 SEPARATE #A INTO #A1 #A2 #A3 WITH DELIMITERS '.'
0090 WRITE 'ALINHAMENTO SEM LEFT JUSTIFIED:' /
0100 #A1 '#A1' / #A2 '#A2' / #A3 '#A3' //
0110 *
0120 SEPARATE #A LEFT JUSTIFIED INTO #A1 #A2 #A3 WITH DELIMITERS '.'
0130 WRITE 'ALINHAMENTO COM LEFT JUSTIFIED:' /
0140 #A1 '#A1' / #A2 '#A2' / #A3 '#A3' //
0150 END
```

Resultado:

```
ALINHAMENTO SEM LEFT JUSTIFIED:
111                 #A1
  1111              #A2
              22233  #A3


ALINHAMENTO COM LEFT JUSTIFIED:
```

```
111                 #A1
1111                #A2
22233               #A3
```

9.2.4. Opção SUBSTRING

A opção SUBSTRING permite que seja feito um SEPARATE em parte de um campo. Depois do nome do campo (*operando1*), especifica-se a posição inicial (*operando2*), e então o tamanho a pesquisar (*operando3*) da porção do campo a ser separada. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A20) INIT<'11111.22222.33333'>
0030 1 #A1          (A5)
0040 1 #A2          (A5)
0050 1 #A3          (A5)
0060 END-DEFINE
0070 *
0080 SEPARATE SUBSTRING(#A,4,6) INTO #A1 #A2 #A3
0090 WRITE '=' #A // '=' #A1 / '=' #A2 / '=' #A3
0100 END
```

Resultado:

```
#A: 11111.22222.33333

#A1: 11
#A2: 222
#A3:
```

9.2.5. Opção IGNORE

Com a opção IGNORE o Natural ignorará se não houverem campos suficientes para receber o resultado do SEPARATE. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A20) INIT<'11111.22222.33333'>
0030 1 #A1          (A5)
0040 1 #A2          (A5)
0050 END-DEFINE
0060 *
0070 SEPARATE #A INTO #A1 #A2 IGNORE
0080 WRITE '=' #A // '=' #A1 / '=' #A2
0090 END
```

Resultado:

```
#A: 11111.22222.33333

#A1: 11111
#A2: 22222
```

9.2.6. Opção REMAINDER

Com a opção REMAINDER *operando*, a seção de valor que não for carregada nos campos receptores, será carregada no *operando*. Este conteúdo do *operando* poderá ser utilizado, por exemplo, para um subseqüente SEPARATE. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A20) INIT<'RRRRR.OOOOO.AAAAA'>
0030 1 #A1          (A5)
0040 1 #A2          (A5)
0050 1 #R           (A5)
0060 END-DEFINE
0070 *
0080 SEPARATE #A INTO #A1 #A2 REMAINDER #R
0090 WRITE '=' #A // '=' #A1 / '=' #A2 / '=' #R
0100 END
```

Resultado:

```
#A: RRRRR.OOOOO.AAAAA

#A1: RRRRR
#A2: OOOOO
#R: AAAAA
```

9.2.7. Opção RETAINED WITH DELIMITERS

Normalmente os caracteres delimitadores não são carregados nos campos receptores do SEPARATE. Com a opção WITH RETAINED DELIMITERS, o delimitador será também carregado no campo receptor. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A20) INIT<'RRRRR.OOOOO.AAAAA'>
0030 1 #A1          (A5)
0040 1 #A2          (A5)
0050 1 #A3          (A5)
0060 END-DEFINE
0070 *
0080 SEPARATE #A INTO #A1 #A2 #A3 IGNORE WITH RETAINED DELIMITERS
0090 WRITE '=' #A // '=' #A1 / '=' #A2 / '=' #A3
0100 END
```

Resultado:

```
#A: RRRRR.OOOOO.AAAAA

#A1: RRRRR
#A2: .
#A3: OOOOO
```

9.3. Instrução COMPRESS

A instrução COMPRESS serve para concatenar o conteúdo de dois ou mais campos em um outro campo.

Os operandos a serem combinados podem ser de qualquer formato, porém o campo resultante deve ser alfanumérico.

Pode-se utilizar as variáveis de sistema, como *NUMBER ou *COUNTER, diretamente no comando COMPRESS.

```
0010 DEFINE DATA LOCAL
0020 1 #A           (A40) INIT<'O PERDIGAO FEZ O CURSO DE NATURAL DIA'>
0030 1 #B           (N2)  INIT<10>
0040 1 #C           (A50)
0050 END-DEFINE
0060 *
0070 COMPRESS #A #B TO #C LEAVING NO
0080 WRITE '=' #A / '=' #B // '=' #C
0090 END
```

Resultado:

```
#A: O PERDIGAO FEZ O CURSO DE NATURAL DIA
#B:  10

#C: O PERDIGAO FEZ O CURSO DE NATURAL DIA10
```

Exemplo de utilização de (PM=I), para inversão dos dados. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #NOME           (A8)  INIT<'LUCIANO'>
0030 1 #SOBRENOME      (A8)  INIT<'PERDIGÃO'>
0040 1 #NOME_COMPLETO  (A30)
0050 END-DEFINE
0060 *
0070 COMPRESS #NOME(PM=I) #SOBRENOME INTO #NOME_COMPLETO
0080 WRITE '=' #NOME / '=' #SOBRENOME // '=' #NOME_COMPLETO
0090 *
0100 END
```

Resultado:

```
#NOME: LUCIANO
#SOBRENOME: PERDIGÃO

#NOME_COMPLETO: ONAICUL PERDIGÃO
```

9.3.1. Opção DELIMITER(S)

A opção DELIMITER(S) permite especificar um caracter delimitador entre cada valor no campo resultante.

Se não for especificada a opção DELIMITERS(S) é colocado um espaço em branco entre os valores. Se for especificado DELIMITER(S) sem caracter algum, será assumido o caracter especificado no parâmetro ID=’delimitador’ do Natural, cujo valor default é “,” (vírgula).

Se um dos operandos do comando COMPRESS for numérico e tiver seu valor zerado, o zero aparecerá no campo resultante.

```
0010 DEFINE DATA LOCAL
0020 1 #DIA         (N2) INIT<10>
0030 1 #MES         (N2) INIT<02>
0040 1 #ANO         (N4) INIT<2004>
0050 1 #DATA        (A10)
0060 END-DEFINE
0070 *
0080 COMPRESS #DIA #MES #ANO INTO #DATA WITH DELIMITERS '/'
0090 WRITE '=' #DIA / '=' #MES / '=' #ANO // '=' #DATA
0100 END
```

Resultado:

```
#DIA:  10
#MÊS:   2
#ANO:  2004

#DATA: 10/2/2004
```

9.3.2. Opção LEAVING NO SPACE

A opção LEAVING NO SPACE serve para suprimir o espaço em branco entre dois ou mais campos concatenados.

```
0010 DEFINE DATA LOCAL
0020 1 #DIA         (N2) INIT<10>
0030 1 #MES         (N2) INIT<02>
0040 1 #ANO         (N4) INIT<2004>
0050 1 #DATA        (A10)
0060 END-DEFINE
0070 *
0080 COMPRESS #DIA #MES #ANO INTO #DATA LEAVING NO
0090 WRITE '=' #DIA / '=' #MES / '=' #ANO // '=' #DATA
0100 *
0110 END
```

Resultado:

```
#DIA:  10
#MÊS:   2
#ANO:  2004
```

```
#DATA: 1022004
```

9.3.3. Opção NUMERIC

A opção NUMERIC inclui os caracteres de sinal e decimal na concatenação de valores.

Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #NUM1     (N3,2) INIT<-123,45>
0030 1 #NUM2     (N2,2) INIT<-9,1>
0040 1 #NUM3     (A12)
0050 END-DEFINE
0060 *
0070 COMPRESS #NUM1 #NUM2 INTO #NUM3 WITH DELIMITER '*'
0080 WRITE 'COMPRESS SEM NUMERIC:' #NUM3
0081 *
0090 COMPRESS NUMERIC #NUM1 #NUM2 INTO #NUM3 WITH DELIMITER '*'
0100 WRITE 'COMPRESS COM NUMERIC:' #NUM3
0110 *
0120 END

Resultado:

COMPRESS SEM NUMERIC: 12345*910
COMPRESS COM NUMERIC: -123,45*-9,1
```

9.4. Concatenação com Hífen

```
0010 DEFINE DATA LOCAL
0020 1 #S            (A79)
0030 END-DEFINE
0040 *
0050 MOVE 'Este exemplo mostra uma frase concatenada com o símbolo "-" no'-
0060 ' programa Natural' TO #S
0070 *
0080 WRITE #S
0090 END

Resultado:

Este exemplo mostra uma frase concatenada com o símbolo '-' no programa Natural
```

10. VARIÁVEIS E CONSTANTES

10.1. Variáveis de Sistema

- ***APPLIC-ID (A008)**

Nome da biblioteca Natural.

- ***COM (A128)**

  Representa uma área de comunicação que pode ser utilizada para processar dados de saída de uma window. Normalmente, quando uma window está ativa, nenhum dado pode ser *inputado* na tela sobreposta pela window. Contudo, se o mapa contém *COM como uma variável modificável, ele estará disponível para que o usuário entre com dados quando uma window estiver ativa na tela.

- ***COUNTER (P10)**

  Número de vezes que o loop foi executado em um FIND, READ, SELECT ou HISTOGRAM. Não é incrementado se o registro não passa pela cláusula WHERE. É incrementado antes do ACCEPT ou REJECT.

- ***CURSOR (N6)**

  Posição do cursor (linha e coluna). Exemplo: 005024 – linha 5, coluna 24.

- ***CURS-COL (P3)**

  Coluna do cursor. Pode ser modificada.

- ***CURS-FIELD (I4)**

  Contém a identificação interna do campo de input no qual o cursor está posicionado. Só pode ser utilizado com a função POS, que serve para checar em que campo o cursor está posicionado.

- ***CURS-LINE (P3)**

  Contém o número da linha em que o cursor está posicionado. Pode conter também os seguintes valores de posicionamento:

  0 → topo ou rodapé do frame de uma windows;
  -1 → linha de mensagens;
  -2 → linha de informações / estatísticas;
  -3 → linha de PF's (número);
  -4 → linha de PF's (nome).

- ***DATA (N3)**

  Número de elementos de dados no STACK Natural para a próxima instrução INPUT. Pode conter ainda:

  0 → nenhum dado no Stack;
  -1 → próximo elemento no Stack é um comando.

- ***DEVICE (A8)**

  Tipo / modo do recurso de onde o Natural foi invocado.

- ***ERROR-LINE (N4)**

  Número da linha no programa onde ocorreu o erro.

- ***ERROR-NR (N7)**

Número do erro. Pode ser modificado.

- ***ERROR-TA (A8)**

  Nome do programa que recebe o controle quando ocorre um erro. Pode ser modificado. Permanece ativo durante a sessão. Isto evita a codificação do ON ERROR em cada programa. O *PROGRAM não está disponível para o programa que trata o erro, então considerar MOVE *PROGRAM TO GDA em todos os programas.

- ***ISN (P10)**

  Número do ISN (número seqüencial único) do registro corrente referenciado um FIND/READ ou STORE. Pode ser muito eficiente em procedimentos de ler ou reler um registro quando o ISN é conhecido. Num HISTOGRAM o *ISN contém o número de ocorrências associadas no valor corrente de um descritor de grupo periódico.

- ***LANGUAGE (I1)**

  Esta variável contém o código da linguagem de instalação do Natural.

- ***LEVEL (N1)**

  Contém o nível onde um módulo está sendo executado.

- ***LINE-COUNT (P5)**

  Número da linha corrente na página corrente. Se múltiplos relatórios está sendo produzidos, o (*rep*) deve ser especificado para identificar cada um deles. O valor máximo para *LINE-COUNT é 250.

- ***LOG-LS (N3)**

  Contém o valor do parâmetro LS (Line Size) da página lógica do relatório primário (*rep=0*).

- ***LOG-PS (N3)**

  Contém o valor do parâmetro PS (Page Size) da página lógica do relatório primário (*rep=0*).

- ***NUMBER (P10)**

  Para um FIND NUMBER, é o número de registros encontrados. Para um FIND, é o número de registros encontrados como resultado de um critério WITH (anterior à avaliação de um critério WHERE) Para um HISTOGRAM, é o número de registros que contém o valor retornado.

- ***PAGE-NUMBER (P5)**

  Número de página corrente. Pode ser alterado. Se múltiplos relatórios estão sendo produzidos, o controle desta numeração é mantido através de *PAGE-NUMBER(*rep*).

- ***PF-NAME (A10)**

  Contém o nome da função de um PF-KEY, que foi definida através da cláusula NAMED da instrução SET KEY.

- ***PF-KEY (A4)**

Tecla utilizada pelo usuário em resposta a uma instrução INPUT. Valores contidos: PA1 até PA3, PF1 até PF24, CLR, PEN, ENTR.

- ***PROGRAM (A8)**

  Contém o nome do programa Natural que está sendo executado.

- ***SUBROUTINE (A32)**

  Contém o nome interno da subrotina Natural em execução.

```
*APPLIC-ID....: XXXPRG
*APPLIC-NAME..: SYSTEM
*COM..........:
*CONVID.......:           0
*CURSOR.......:    1004
*CURS-COL.....:    4
*CURS-FIELD...:           0
*CURS-LINE....:    1
*DATA.........:   -1
*DEVICE.......: COLOR
*ERROR-LINE...:     0
*ERROR-NR.....:        0
*ERROR-TA.....:
*ETID.........: X?hÂ?  ?
*GROUP........:
*HARDCOPY.....:
*HARDWARE.....: 2084
*INIT-ID......: B1OQ
*INIT-PROGRAM.: NTRN
*INIT-USER....: X028862
*LANGUAGE.....:   38
*LEVEL........:   1
*LIBRARY-ID...: XXXPRG
*LINE-COUNT...:      8
*LINESIZE.....:       80
*LOG-LS.......:  250
*LOG-PS.......:   20
*MACHINE-CLASS: MAINFRAME
*OPSYS........: MVS/ESA
*OS...........: MVS/ESA
*OSVERS.......: SP7.0.4
*PAGESIZE.....:       24
*PAGE-NUMBER..:      2
*PF-KEY.......: ENTR
*PF-NAME......:
*PROGRAM......: VARSIS
*SCREEN-IO....: X
*SERVER-TYPE..:
*STARTUP......:
*STEPLIB......: SISTEM
*SUBROUTINE...:
*THIS-OBJECT..: 0000000000000000
*TPSYS........: CICS
*UI...........: CHARACTER
*USER.........: X028862
*USER-NAME....: SYSTEM
*WINDOW-LS....:   80
*WINDOW-POS...:       0
*WINDOW-PS....:   24
```

10.1.1. Variáveis Date

```
*DATD : 24.11.04         → (A8)  → Formato DD.MM.YY
*DATE : 24/11/04         → (A8)  → Formato DD/MM/YY
*DATI : 04-11-24         → (A8)  → Formato YY-MM-DD
*DATJ : 04329            → (A5)  → Data Juliana no formato YYDDD
*DATU : 11/24/04         → (A8)  → Formato MM/DD/YY
*DATG : 24Novembro 2004  → (A15) → Formato Dd nome-mês YYYY
*DATN :  20041124        → (N8)  → Formato YYYYMMDD
*DATX : 24112004         → (D)   → Formato interno
*DAT4D: 24.11.2004       → (A10) → Formato DD.MM.YYYY
*DAT4E: 24/11/2004       → (A8)  → Formato DD/MM/YYYY
*DAT4I: 2004-11-24       → (A10) → Formato YYYY-MM-DD
*DAT4J: 2004329          → (A7)  → Data Juliana no formato YYYYDD
*DAT4U: 11/24/2004       → (A10) → Formato MM/DD/YYYY
```

10.1.2. Variáveis Time

```
*TIME....: 15:18:13.8       → (A10) → Formato HH:MM:SS:T
*TIMESTMP: BC2B48F0F378858E → (B8)  → Valor do relógio interno da máquina
*TIME-OUT:      0           → (N5)  → Contém o nº de segundos restantes da transação
                                      corrente em execução (somente c/Natural Security)
*TIMN....:  1518138         → (N7)  → Formato HHMMSST
*TIMX....: 15:18:13         → (T)   → Formato interno
*TIMD:        0             → (N7)  → Tempo decorrido durante um comando SETTIME
                                      (HHMMSST)
```

10.2. Operações Aritméticas com Variáveis Date e Time

Números podem ser adicionados ou subtraídos de variáveis com formato (D), sendo que o resultado uma data válida.

Os números adicionados em variáveis de formato (D) são dias, e em formato (T) são décimos de segundo.

```
0010 DEFINE DATA LOCAL
0020 1 #DATA              (D) INIT<*DATX>
0030 1 #HORA              (T) INIT<*TIMX>
0040 END-DEFINE
0050 *
0060 ADD 5 TO #DATA
0070 WRITE 'DATA DE HOJE:' *DATX(EM=DD/MM/YYYY) '+ 5 DIAS ='
0080   #DATA(EM=DD/MM/YYYY)
0090 *
0100 SKIP 1
0110 *
0120 ADD 600 TO #HORA
0130 WRITE 'HORA ATUAL:' *TIMX '+ 60 SEGUNDOS =' #HORA
0140 END
```

Resultado:

```
DATA DE HOJE: 24/11/2004 + 5 DIAS = 29/11/2004

HORA ATUAL: 16:10:37 + 60 SEGUNDOS = 16:11:37
```

10.3. Movimentação com Máscara de Edição Date e Time

As variáveis definidas com formatos DATE (D) e TIME (T) podem ser exibidas de formas diversas, utilizando as máscaras de edição. Exemplo:

```
0010 WRITE NOTITLE
0020      'DATA INTERNA  :' *DATX (DF=L) /
0030      '              :' *DATX (EM=N(9)' 'ZW.'SEMANA(S) 'YY)  /
0040      '              :' *DATX (EM=ZZJ'.DIA 'YYYY)        /
0050      '    ROMANA    :' *DATX (EM=R)  /
0060      '    AMERICANA :' *DATX (EM=MM/DD/YYYY)      12X 'OR  ' *DAT4U /
0070      '    JULIANA   :' *DATX (EM=YYYYJJJ)         15X 'OR  ' *DAT4J /
0080      '    GREGORIANA:' *DATX (EM=ZD.''L(10)''YYYY) 5X 'OR  ' *DATG ///
0090 *
0100      'HORA INTERNA  :' *TIMX                       14X 'OR  ' *TIME /
0110      '              :' *TIMX (EM=HH.II.SS.T) /
0120      '              :' *TIMX (EM=HH.II.SS' 'AP) /
0130      '              :' *TIMX (EM=HH)
0140 END

Resultado:

DATA INTERNA  : 24/11/2004
              : Quarta 48.SEMANA(S) 04
              : 329.DIA 2004
    ROMANA    : MMIV
    AMERICANA : 11/24/2004            OR   11/24/2004
    JULIANA   : 2004329               OR   2004329
    GREGORIANA: 24.Novembro2004       OR   24Novembro 2004


HORA INTERNA  : 16:01:02              OR   16:01:02.5
              : 16.01.02.5
              : 04.01.02 PM
              : 16
```

10.4. Variável Lógica

Utilizada para verificar se uma condição é verdadeira (TRUE) ou falsa (FALSE).

É mais fácil de entender a lógica de programas que utilizam variáveis lógicas do que trabalhar com flags e controles. Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #FLAG              (L) INIT<TRUE>
0030 1 #I                 (N2)
0040 END-DEFINE
0050 *
```

```
0060 FOR #I 1 3
0070   WRITE #FLAG(EM=FALSE/TRUE) '/' #FLAG(EM=ON/OFF) 5X 'INDICE' #I
0080   IF #FLAG
0090     #FLAG := FALSE
0100   ELSE
0110     #FLAG := TRUE
0120   END-IF
0130 END-FOR
0140 END
```

Resultado:

```
TRUE  / OFF     INDICE   1
FALSE / ON      INDICE   2
TRUE  / OFF     INDICE   3
```

11. FUNÇÕES MATEMÁTICAS

```
ABS(campo)     → Valor absoluto
ATN(campo)     → Arco tangente
COS(campo)     → Coseno
EXP(campo)     → Exponencial
FRAC(campo)    → Parte fracionária
INT(campo)     → Parte inteira
LOG(campo)     → Logaritmo natural (base e¹⁰)
SGN(campo)     → Sinal (+,0,-)
SIN(campo)     → Seno
SQRT(campo)    → Raiz quadrada. Se o campo informado for negativo, se-
                 rá tratado como positivo
TAN(campo)     → Tangente
VAL(campo)     → Valor numérico de campo alfanumérico
```

```
0010 DEFINE DATA LOCAL
0020 1 #A           (N2,1) INIT <10>
0030 1 #B           (N2,1) INIT <-6,3>
0040 1 #C           (N2,1) INIT <0>
0050 1 #LOGA        (N2,6)
0060 1 #SQRTA       (N2,6)
0070 1 #TANA        (N2,6)
0080 1 #ABS         (N2,1)
0090 1 #FRAC        (N2,1)
0100 1 #INT         (N2,1)
0110 1 #SGN         (N1)
0120 END-DEFINE
0130 COMPUTE #LOGA  = LOG(#A)
0140 WRITE NOTITLE '=' #A 5X 'LOG' 40T #LOGA
0150 *
0160 COMPUTE #SQRTA = SQRT(#A)
0170 WRITE         '=' #A 5X 'RAIZ QUADRADA' 40T #SQRTA
0180 *
```

```
0190 COMPUTE #TANA  = TAN(#A)
0200 WRITE         '='  #A 5X 'TANGENTE' 40T #TANA
0210 *
0220 COMPUTE #ABS   = ABS(#B)
0230 WRITE      /  '='  #B 5X 'VALOR ABSOLUTO' 40T #ABS
0240 *
0250 COMPUTE #FRAC  = FRAC(#B)
0260 WRITE         '='  #B 5X 'FRAÇÃO' 40T #FRAC
0270 *
0280 COMPUTE #INT   = INT(#B)
0290 WRITE         '='  #B 5X 'INTEIRO' 40T #INT
0300 *
0310 COMPUTE #SGN   = SGN(#A)
0320 WRITE       / '='  #A 5X 'SINAL'     40T #SGN
0330 *
0340 COMPUTE #SGN   = SGN(#B)
0350 WRITE         '='  #B 5X 'SINAL'     40T #SGN
0360 *
0370 COMPUTE #SGN   = SGN(#C)
0380 WRITE         '='  #C 5X 'SINAL'     40T #SGN
0390 END
```

Resultado:

```
#A:  10,0     LOG                        2,302585
#A:  10,0     RAIZ QUADRADA              3,162277
#A:  10,0     TANGENTE                   0,648360

#B:  -6,3     VALOR ABSOLUTO             6,3
#B:  -6,3     FRAÇÃO                    -0,3
#B:  -6,3     INTEIRO                   -6,0

#A:  10,0     SINAL                      1
#B:  -6,3     SINAL                     -1
#C:   0,0     SINAL                      0
```

12. INSTRUÇÃO MASK

Permite que o conteúdo das posições de um campo seja checado. MASK é justificado à esquerda.

Sinais “*” (asterisco) ou “%” (percentual), inseridos na definição do MASK indicam que qualquer número de posições podem ser ignoradas na verificação. O “.” (ponto), “,” (vírgula) ou o “_” (underscore), significam que esta posição não será verificada.

A checagem pode ser feita por constantes, variáveis, tipos, datas e intervalos.

```
IF #X = MASK(..'VA')
```

13. INSTRUÇÃO IF campo IS...

Verifica o formato e tamanho de uma variável alfanumérica.

Formatos:

- N*xx* → numérico com tamanho *xx*
- F*xx* → ponto flutuante com tamanho *xx*
- D → data
- T → hora
- P*xx* → numérico compactado com tamanho *xx*
- I*xx* → numérico inteiro com tamanho *xx*

```
ABS(campo)     → Valor absoluto
ATN(campo)     → Arco tangente
COS(campo)     → Cosseno
EXP(campo)     → Exponencial
FRAC(campo)    → Parte fracionária
INT(campo)     → Parte inteira
LOG(campo)     → Logaritmo natural (base e¹⁰)
SGN(campo)     → Sinal (+,0,-)
SIN(campo)     → Seno
SQRT(campo)    → Raiz quadrada. Se o campo informado for negativo, se-
                 rá tratado como positivo
TAN(campo)     → Tangente
VAL(campo)     → Valor numérico de campo alfanumérico
```

Exemplo:

```
0010 DEFINE DATA LOCAL
0020 1 #CAMPOA       (A10)
0030 1 #CAMPOB       (N5)
0040 1 #DATA         (A10)
0050 END-DEFINE
0060 INPUT #DATA #CAMPOA
0070 IF #DATA IS (D)
0080   IF #CAMPOA IS (N5)
0090     COMPUTE #CAMPOB = VAL(#CAMPOA)
0100     WRITE NOTITLE 'FUNÇÃO VAL OK' // '=' #CAMPOA '=' #CAMPOB
0110   ELSE
0120     REINPUT 'FORMATO DO CAMPO INVALIDO (N5)' MARK *#CAMPOA
0130   END-IF
0140 ELSE
0150 REINPUT 'DATA INVALIDA. NO FORMATO (DD/MM/YYYY) ' MARK *#DATA
```

```
0160 END-IF
0170 END
```

14. Instrução MOVE

14.1. Opção EDITED

O comando MOVE EDITED é utilizado para fazer movimentações com edição ou de-edição.

Pode ser especificada uma máscara de edição para o campo fonte ou para o campo destino:

- CAMPO FONTE → a máscara é aplicada ao conteúdo do campo fonte e então o valor editado é movido para o campo destino.
- CAMPO DESTINO → a máscara de edição é aplicada sobre o conteúdo do campo fonte para a *remoção* dos caracteres de edição e o valor resultante, sem os caracteres de edição, é movido para o campo destino.

Outra forma de processar o MOVE EDITED com *de-edição* é a utilização para inicializar variáveis do tipo (D) a partir de variáveis alfanuméricas com caracter de edição.

```
0010 DEFINE DATA LOCAL
0020 1 #A                (A6) INIT<'811123'>
0030 1 #B                (A8)
0040 1 #C                (A6)
0050 END-DEFINE
0060 *
0070 MOVE EDITED #A(EM=XX/XX/XX) TO #B
0080 *
0090 MOVE EDITED #B              TO #C(EM=XX/XX/XX)
0100 *
0110 WRITE '=' #A // '=' #B // '=' #C
0120 END
```

Resultado:

```
#A: 811123

#B: 81/11/23

#C: 811123
```

```
0010 DEFINE DATA LOCAL
0020 1 #A                (A10) INIT<'23/11/2004'>
0030 1 #D                (D)
0040 END-DEFINE
```

```
0050 *
0060 MOVE EDITED #A TO #D(EM=DD/MM/YYYY)
0070 *
0080 WRITE '=' #A // '=' #D
0090 END
```

Resultado:

```
#A: 23/11/2004

#D: 23112004
```

14.2. Opção LEFT JUSTIFIED

Move os dados fazendo alinhamento à esquerda do campo receptor.

Esta opção é útil para mover campos de entrada, evitando ter que pesquisar por brancos digitados à esquerda do valor.

```
0010 DEFINE DATA LOCAL
0020 1 #NUM          (N5)  INIT<10>
0030 1 #A            (A10) INIT<'     ABCDE'>
0040 1 #B            (A10)
0050 END-DEFINE
0060 MOVE LEFT JUSTIFIED #NUM TO #B
0070 DISPLAY '12345' #NUM '1234567890' #B
0080 *
0090 MOVE LEFT JUSTIFIED #A TO #B
0100 WRITE // '#A: +' #A '+' /
0110          '#B: +' #B '+' /
0120 END
```

Resultado:

```
12345  1234567890
------ ----------

    10 10


#A: +      ABCDE +
#B: + ABCDE      +
```

A opção LEFT JESTIFIED não é válida quando o campo destino é numérico. Nesta situação, ocorre um erro de compilação.

14.3. Opção RIGHT JUSTIFIED

Movimenta os dados alinhando à direita do campo destino.

```
0010 DEFINE DATA LOCAL
0020 1 #NUM          (N5)  INIT<10>
0030 1 #A            (A10) INIT<'ABCDE'>
```

```
0040 1 #B           (A10)
0050 END-DEFINE
0060 MOVE RIGHT JUSTIFIED #NUM TO #B
0070 DISPLAY '12345' #NUM '1234567890' #B
0080 *
0090 MOVE RIGHT JUSTIFIED #A TO #B
0100 WRITE // '#A: +' #A '+' /
0110          '#B: +' #B '+' /
0120 END
```

Resultado:

```
12345  1234567890
------ ----------

    10         10


#A: + ABCDE      +
#B: +      ABCDE +
```

14.4. Opção BY NAME

Movimenta os campos individuais (itens elementares) de uma estrutura de dados para outra, sendo que apenas os campos cujos nomes aparecem em ambas às estruturas são movimentadas.

A segunda estrutura (destino) deve ser de nível 01.

Esta opção é muito útil no caso de codificação de múltiplos MOVEs campo a campo. Por exemplo, quando se deseja mover os campos de uma view para um registro de workfile.

```
0010 DEFINE DATA LOCAL
0020 1 XXX VIEW OF PERSONNEL
0030   2 NAME
0040   2 SEX
0050   2 AGE
0060   2 PHONE
0070   2 SALARY
0080 *
0090 1 #REG-XXX
0100   2 NAME            (A20)
0110   2 SEX             (A1)
0120   2 AGE             (N2)
0130   2 PHONE           (A8)
0140   2 SALARY          (N6)
0150 END-DEFINE
0160 *
0170 READ XXX
0180   MOVE BY NAME XXX TO #REG-XXX
0190 END-READ
0200 END
```

Não esquecer de sempre qualificar o nome dos campos com o nome da view, mesmo dentro do loop de leitura. Isto porque o Natural sempre assume, em princípio, que se trata de uma variável do usuário e não um campo de view.

14.5. Opção BY POSITION

A opção BY POSITION do MOVE permite mover campos de um grupo para outro, apesar dos nomes destes campos poderem estar diferentes.

As regras são as seguintes:

- O número de campos nos dois grupos precisa ser o mesmo;
- Os níveis dos campos precisam ser iguais

Se houver a presença de estruturas de ARRAY, eles precisam ter as mesmas dimensões.

```
0010 DEFINE DATA LOCAL
0020 1 XXX VIEW OF PERSONNEL
0030   2 NAME
0040   2 SEX
0050   2 AGE
0060   2 PHONE
0070   2 SALARY
0080 *
0090 1 #REG-XXX
0100   2 NOME            (A20)
0110   2 SEXO            (A1)
0120   2 IDADE           (N2)
0130   2 TELEFONE        (A8)
0140   2 SALARIO         (N6)
0150 END-DEFINE
0160 *
0170 READ XXX
0180   MOVE BY POSITION XXX TO #REG-XXX
0190 END-READ
0200 END
```

14.6. Opção SUBSTRING

Permite mover a porção de um campo para outro campo.

```
MOVE SUBSTRING(#A,1,5 ) TO #B
```

Permite mover o valor de um campo para a porção de um outro campo.

```
MOVE #A TO SUBSTRING(#B,1,5)
```

Permite mover a porção de um campo para a porção de um outro campo.

```
MOVE SUBSTRING(#A,7,3) TO SUBSTRING(#B,3,3)
```

```
0010 DEFINE DATA LOCAL
0020 1 #NOME         (A30) INIT<'LUCIANO  PERDIGÃO'>
0030 1 #SOBRENOME    (A30)
0040 END-DEFINE
0050 MOVE SUBSTRING(#NOME,10,8) TO #SOBRENOME
0060 *
0070 WRITE '=' #NOME // '=' #SOBRENOME //
0080 *
0090 MOVE #SOBRENOME TO SUBSTRING(#NOME,1,8)
0100 *
0110 WRITE '=' #NOME
0120 END
```

Resultado:

```
#NOME: LUCIANO  PERDIGÃO

#SOBRENOME: PERDIGÃO


#NOME: PERDIGÃO PERDIGÃO
```

14.7. Opção ALL

Move repetidas vezes o valor do campo fonte para o campo destino, até que esteja completo.

Este comando pode ser usado para expressar um gráfico de barras, por exemplo:

```
MOVE ALL '/****' TO #A UNTIL #N
```

O tamanho da barra seria calculado pelo programa e atribuído à #N, e o comando MOVE ALL desenharia o gráfico.

Para #N = 3, #A conteria "/**".

Para #N=12, #A conteria "/****/****/*"

Não utilizar este comando para "branquear" uma variável alfa. Um RESET ou MOVE ALL é muito mais eficiente e o efeito é o mesmo.

A opção UNTIL controla o número de caracteres a serem movidos e não o número de repetições do campo fonte.

```
0010 DEFINE DATA LOCAL
0020 1 #A            (A30)
0030 END-DEFINE
0040 MOVE ALL '#****' TO #A
0050 WRITE '=' #A /
0060 *
0070 RESET #A
0080 MOVE ALL '#****' TO #A UNTIL 10
0090 WRITE '=' #A
0100 END
```

Resultado:

```
#A: #****#****#****#****#****#****

#A: #****#****
```

15. CONSIDERAÇÕES DECIDE x IF

O comando IF tem algumas considerações importantes quanto à performance:

- As condições são analisadas da esquerda para a direita;
- Assim que o Natural "determine" um IF, isto é, encontre uma cláusula FALSA entre cláusulas combinadas por AND ou uma VERDADEIRA entre cláusulas combinadas por OR, as demais condições do IF não são avaliadas.

Consideremos a seguinte situação:

Se uma série de quatro condições é verdadeira, deve-se executar alguma rotina, caso contrário nada deve ser feito.

A melhor codificação do comando IF será:

```
IF COND1 AND COND2 AND COND3 AND COND4
```

Onde COND1 é a condição com maior probabilidade de ser FALSA e COND4 a mais provável de ser VERDADEIRA. Assim o Natural poderia interromper a avaliação quando encontrasse a primeira condição FALSA.

Consideremos agora o uso do comando DECIDE para esta mesma situação. Desta forma, poderíamos ter a seguinte codificação:

```
DECIDE FOR FIRST CONDITION
   WHEN COND1 AND COND2 AND COND3 AND COND4
      ...
   WHEN NONE IGNORE
END-DECIDE
```

Ou ainda:

```
DECIDE FOR EVERY CONDITION
   WHEN COND1
   WHEN COND2
   WHEN COND3
   WHEN COND4
```

```
        WHEN ALL
        WHEN NONE IGNORE
    END-DECIDE
```

A primeira codificação é melhor e teria o mesmo nível de performance que o exemplo com o comando IF. A segunda codificação funciona, porem será menos eficiente.

A grande vantagem do uso dos comandos DECIDE FOR e DECIDE ON é quando se pode fazer uso das cláusulas ANY, ALL e NONE.

Nestes casos, a codificação com o comando IF é recomendável em casos apropriados, principalmente quando a cláusula ELSE não é relevante. Entretanto, na grande maioria dos casos pode-se usar o comando DECIDE, obtendo-se não só melhor legibilidade como também maior eficiência.

16. COMANDOS DE ACESSO AO BANCO DE DADOS ADABAS

16.1. Find

```
FIND |operando1|   RECORDS IN FILE view-name

     PASSWORD=operando2
     CIPHER=operando3
     WITH-clause
     COUPLED-clause
     STARTING WITH ISN=operando5
     SORTED BY-clause
     RETAIN-clause
     WHERE-clause
     IF NO RECORDS FOUND-clause
     Statement...
END-FIND
```

A instrução FIND é usada para selecionar registros de arquivos da base de dados, baseado em um critério de pesquisa que consiste de campos descritores (chaves).

Esta instrução inicia um LOOP que é executado a cada vez que um registro é selecionado. Cada campo do registro pode ser referenciado dentro deste LOOP.

16.1.1. Processo de limite (ALL/*operando1*)

O número de registros selecionados pode ser limitado através do *operando1*, que pode ser tanto uma constante numérica ou uma variável entre parênteses. ALL pode ser especificado opcionalmente para enfatizar que todos os registros selecionados devem ser processados.

Quando um limite for especificado via *operando1*, este limite se aplica ao LOOP que está sendo executado. O número especificado não pode exceder o valor do limite global definido tanto pela instrução LIMIT quanto pela instrução SET GLOBALS.

Registros rejeitados pela cláusula WHERE não são levados em conta para controlar o limite.

```
0010 DEFINE DATA LOCAL
0020 1 XXX VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 SEX
0060   2 CITY
0070 END-DEFINE
0080 *
0090 FIND(2) XXX WITH NAME = 'ARTHUR'
0100   DISPLAY PERSONNEL-ID
0110     NAME
0120     SEX
0130     CITY
0140 END-FIND
0150 END
```

Resultado:

```
PERSONNEL         NAME         S         CITY
   ID                          E
                               X
--------- -------------------- - --------------------

50005500  ARTHUR               M BRASILIA
50005300  ARTHUR               F BRASILIA
```

16.1.2. Find First, Find Number, Find Unique

Estas opções são usadas para selecionar o primeiro registro de um conjunto selecionados (FIND FIRST), para determinar o número de registros de um conjunto selecionado (FIND NUMBER), ou para assegurar que apenas um registro satisfaz um critério de seleção (FIND UNIQUE).

16.1.3. View-name

O nome de uma “view” é definida em um área de dados global ou local (“global data area” ou “local data area”).

Em “structured mode”, o nome da “view” precisa ser definido na área de dados global ou local, usando ou o editor de área de dados ou a instrução DEFINE DATA.

16.1.4. Cláusula PASSWORD

A cláusula PASSWORD é usada para fornecer uma senha (password) para acessar dados de um arquivo ADABAS protegido por password.

Se a PASSWORD for especificada como uma constante, ela deve ser sempre ser codificada na coluna 1 (um) de uma linha para que não apareça. Quando a PASSWORD for fornecida através do terminal, deve ser digitado primeiro ”%*” para que a PASSWORD não apareça.

O valor da PASSWORD não pode ser alterado durante a execução de um LOOP.

```
0010 DEFINE DATA LOCAL
0020 1 XXX VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 SEX
0060   2 CITY
0070 END-DEFINE
0080 *
0090 FIND(2) XXX
0100 PASSWORD=
0110     WITH NAME = 'ARTHUR'
0120   DISPLAY PERSONNEL-ID
0130     NAME
0140     SEX
0150     CITY
0160 END-FIND
0170 END
```

16.1.5. Cláusula CIPHER

Esta cláusula é usada para acessar arquivos ADABAS cifrados. O seu uso é análogo a da cláusula PASSWORD.

A chave de cifragem pode ser especificado pode ser especificada como uma constante numérica ou como uma variável definida pelo usuário com o formato “(N8)”.

O valor da chave não pode ser alterado durante a execução de um LOOP.

```
0010 DEFINE DATA LOCAL
0020 1 XXX VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 SEX
0060   2 CITY
0070 *
0080 1 #C                (N8)
0090 END-DEFINE
0100 *
0110 INPUT (AD=MDLT'_') 'ENTRE COM A CHAVE CIFRADA:' #C
0120 *
0130 FIND(2) XXX
```

```
0140 PASSWORD=
0150   CIPHER = #C
0160     WITH NAME = 'ARTHUR'
0170   DISPLAY PERSONNEL-ID
0180     NAME
0190     SEX
0200     CITY
END-FIND
END
```

16.1.6. Cláusula WITH

```
WITH LIMIT (operand4) basic-search-criterion
```

Esta cláusula é obrigatória e é usada para especificar o critério de pesquisa, que consiste de campos chaves (descritores) definidos na base de dados.

Dentro do WITH podem ser usados descritores, sub-descritores, super-descritores ou descritores fonéticos.

O número de registros selecionados como resultado de uma cláusula WITH pode ser limitado com o uso da palavra LIMIT junto com uma constante numérica ou uma variável definida pelo usuário, entre parênteses, contendo o valor de limite. Se o número de registros selecionados exceder o limite, o programa terminará com uma mensagem de erro.

```
0010 DEFINE DATA LOCAL
0020 1 XXX VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 SEX
0060   2 CITY
0070 *
0080 1 #C               (N8)
0090 END-DEFINE
0100 *
0110 FIND XXX WITH LIMIT (20) NAME = 'ARTHUR'
0120   DISPLAY PERSONNEL-ID
0130     NAME
0140     SEX
0150     CITY
0160 END-FIND
0170 DISPLAY *NUMBER(0110) *COUNTER(0110)
0180 END
```

Resultado:

```
NAT1005 More records found than specified in search limit.
```

16.1.6.1. Critério de básico de pesquisa

```
                                =
1)  descritor[(i)]            EQ       value
                            EQUAL TO

                              OR =         value
                          THRU value BUT NOT value

                                =
                                EQ
                              EQUAL TO
                                ¬=
                                NE
                              NOT EQUAL
                                <
2) descritor[(i)]           LT LESS THAN
                                <=
                                LE
                             LESS EQUAL
                                >
                                GT
                            GREATER THAN
                                >=
                                GE
                            GREATER EQUAL

3) set-name
```

- descritor → descrito, sub-descritor, super-descritor ou descrito fonético ADABAS.
- i → Um descritor contido dentro de um grupo periódico pode ser especificado com ou sem índice. Se o índice não for especificado, o registro será selecionado se o valor especificado estiver em qualquer ocorrência.

  Se o índice for especificado, o registro é selecionado apenas se o valor estiver na ocorrência espcificada pelo índice.

  O índice deve ser uma constante e não pode ser especificado um intervalo.

  Se um descritor estiver dentro de um campo múltiplo, o índice não pode ser especificado. O registro será selecionado se o valor estiver em qualquer posição do campo.
- Valor → valor de pesquisa. Os formatos do descritor e o valor de pesquisa devem se compatíveis.
- set-name → identifica um conjunto de registros previamente selecionados com um FIND com a cláusula RETAIN. O conjunto referenciado no FIND precisa ter sido criado do mesmo arquivo ADABAS físico. O "set-name" deve ser uma constante alfanumérica (máximo de 32 caracteres) ou o conteúdo de uma variável alfanumérica.

16.1.6.2. Critério de pesquisa utilizando conexão

Critérios de pesquisa podem ser combinados usando operadores "booleanos" AND, OR e NOT. Podem ser usados parênteses para controlar a ordem de avaliação, que é:

```
()  → Parênteses (mais alta)
NOT → Negação
AND → Conexão E
OR  → Conexão OU (mais baixa)
```

Exemplos:

```
FIND EMPLOYEES WITH NAME = 'LUCIANO'

FIND EMPLOYEES WITH CITY NE 'BRASILIA'

FIND EMPLOYEES WITH BIRTH = 710501

FIND EMPLOYEES WITH BIRTH = 710501 THRU 710531

FIND EMPLOYEES WITH NAME = 'PERDIGAO' OR = 'LUCIANO' OR = 'ROGERIO'

FIND EMPLOYEES WITH PERSONNEL-ID = 1000 THRU 1009
               BUT NOT             1005 THRU 1007

FIND EMPLOYEES WITH BIRTH LT 750105 AND DEPT = 'DEPT05'

FIND EMPLOYEES WITH JOB-TITLE = 'ANALISTA'
               AND (BIRTH GT 700221 OR LANG = 'PORTUGUESE'

FIND EMPLOYEES WITH JOB-TITLE = 'ANALISTA'
               AND NOT (BIRTH GT 700221 OR LANG = 'PORTUGUESE')

FIND EMPLOYEES WITH DEPT = 'ABC' THRU 'DEF'
               AND (CITY = 'BRASILIA' OR = 'CORONEL FABRICIANO')
               AND BIRTH GT 600101

FIND VEHICLES WITH MAKE = 'FIAT'
               AND COLOR = 'AZUL' OR = 'AMARELO' OR = 'VERMELHO'
```

16.1.7. Uso de descritores

Os usuário podem usar campos de base de dados definidos como descritores para construírem critérios básicos de pesquisa. Os valores usados com os descritores devem ser compatíveis com o formato interno do descritor.

16.1.8. Sub-descritores, Super-descritores e Descritores Fonéticos

Um sub-descritor é um descritor formado de parte de um campo e o seu formato interno é o mesmo do campo do qual ele deriva.

Um super-descritor é um descritor cujo valor é formado de um ou mais campos ou partes de campos. O seu formato interno é binário se todos os campos dos quais ele deriva forem definidos com o formato numérico. Caso contrário, o formato é alfanumérico.

Uma pesquisa fonética retorna todos os valores cujo som seja semelhante ao do valor procurado. Um descritor fonético tem sempre o formato alfanumérico.

Valores para sub-descritores e super-descritores podem ser especificados das seguintes formas:

- Constantes numéricas ou hexadecimais. Uma constante hexadecimal deve ser usada para super-descritores com formato binário.
- Variáveis definidas pelo usuário podem ser especificadas usando a instrução REDEFINE para selecionar as partes que formam o valor de um sub ou super-descritor.

16.1.9. Uso de Descritores contidos dentro de um “Array” do DB

Estes descritores podem ser um campo múltiplo ou estar contido em um grupo periódico.

Um descritor contido dentro de um grupo periódico pode ser especificado com ou sem índice. Se o índice não for especificado, o registro será selecionado se o valor estiver em qualquer ocorrência.

Se o índice for especificado, o registro é selecionado apenas se o valor estiver na ocorrência especificada pelo índice.

O índice deve ser uma constante e não pode ser especificado um intervalo.

Se um descritor estiver dentro de um campo múltiplo, o índice não pode ser especificado. O registro será selecionado se o valor estiver em qualquer posição do campo.

Obs.: os seguintes exemplos assumem que o campo “SALARY” é um descritor contido dentro de um grupo periódico e o campo “LANG” é um campo múltiplo.

Exemplos:

```
FIND EMPLOYEES WITH SALARY LT 20000
(resulta na pesquisa de todas as ocorrências de SALARY)

FIND EMPLOYEES WITH SALARY(1) LT 20000
(resulta na pesquisa somente da primeira ocorrência)

FIND EMPLOYEES WITH SALAY (1:4) LT 20000     (inválido)
(não pode ser especificado um intervalo para campos dentro de grupo
 periódico usados como chave)

FIND EMPLOYEES WITH LANG = 'BRAZIL'
(resulta na pesquisa de todos os valores de LANG)

FIND EMPLOYEES WITH LANG(1) = 'BRAZIL'       (inválido)
(não pode ser especificado índice para campos múltiplos)
```

16.1.10. Cláusula COUPLED

```
AND COUPLED TO FILE view-name   VIA descriptor1  EQ     descriptor2
OR                                               EQUAL
                                                 =

                                WITH basic-search-criterion
```

Está cláusula é usada para especificar uma pesquisa que envolva o uso da facilidade de acoplamento do ADABAS, que permite que descritores de diferente arquivos sejam especificados no critério de pesquisa de um único FIND.

Os arquivos usados com a cláusula COUPLED devem ser fisicamente acoplados, usando o utilitário ADABAS apropriado.

Em um único FIND, podem ser usadas, no máximo, 4 cláusulas COUPLED.

O mesmo arquivo não pode ser usado em duas diferentes cláusulas COUPLED dentro do mesmo FIND.

Um "set-name" (ver cláusula RETAIN) não pode ser especificado no critério básico de pesquisa.

A conexão de critérios, usando parênteses, não é permitida no critério básico de pesquisa.

Campos de base de dados especificados em uma cláusula COUPLED não estão disponíveis para referências posteriores, a menos que um outro READ ou FIND seja usado separadamente para o arquivo acoplado.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 SEX
0060   2 CITY
0070 *
0080 1 VEI VIEW OF VEHICLES
0090   2 MAKE
0100 END-DEFINE
0110 *
0120 FIND EMP WITH CITY = 'BARCELONA'
0130     AND COUPLED TO VEI WITH MAKE = 'FIAT'
0140   DISPLAY NAME
0150 END-FIND
0160 END
```

A referência a NAME é válida pois este campo está contido no arquivo EMPLOYEES e a referência a MAKE é inválida proque MAKE está contido no arquivo VEHICLES que foi especificado na cláusula COUPLED.

16.1.11. Cláusula SORTED BY

```
SORTED BY descriptor ...3  DESCENDING
```

A cláusula SORTED BY é usada para que o ADABAS classifique os registros selecionados baseado na seqüência de um a três descritores, que podem ser diferentes dos usados para a seleção. Pode ser especificada classificação ascendente (default) ou descendente. Para fazer esta classificação, o ADABAS utiliza as listas invertidas, sem fazer a leitura dos registros.

Se os descritores contiverem um grande número de valores ocorrerá um grande "OVERHEAD" quando do uso do SORTED, porque será preciso pesquisar a lista inteira até que todos os registros selecionados tenham sido localizados. Esta cláusula não deve ser usada quando for grande o número de registros a serem classificados.

Não se pode usar campos periódicos para realizar a classificação. São aceitos campos múltiplos sem índice.

As cláusulas SORTED BY e RETAIN não pode ser usadas juntas.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 FIRST-NAME
0050   2 NAME
0060   2 SEX
0070   2 CITY
0080 END-DEFINE
```

```
0090 *
0100 FIND(15) EMP WITH CITY = 'BRASILIA'
0110     SORTED BY NAME PERSONNEL-ID
0120   DISPLAY NAME(IS=ON) FIRST-NAME PERSONNEL-ID
0130 END-FIND
0140 END
```

Resultado:

```
NAME              FIRST-NAME      PERSONNEL
                                        ID
------------------- ------------------- ---------

                    FULANO DE TAL        12000002
                    FULANA DE TAL        12000003
ADRIANA             SILVA                12111
ANTONIA PAULINA                          24242425
ANTUNES             MARCIO               777
ARTHUR              MOURA                50001700
                    MOURA                50001900
                    MOURA                50002100
                    MOURA                50002300
                    MOURA                50002500
                    MOURA                50002700
                    MOURA                50002900
                    MOURA                50003000
                    MOURA                50003100
                    MOURA                50003300
```

16.1.12. Cláusula RETAIN

```
RETAIN AS operando
```

Usa-se esta cláusula para guardar os ISN's de um grupo de registros selecionados pela cláusula WITH para processamento posterior.

A lista de ISN's dos registros selecionados é armazenada com um conjunto de ISN's no ADABAS WORK FILE. O conjunto pode ser usado em FIND's subseqüentes com critério básico de pesquisa para posterior refinamento do conjunto ou para posterior processamento dos registros. O conjunto criado é específico do arquivo e só pode ser usado em outro FIND que processa o mesmo arquivo. O conjunto pode ser referenciado por qualquer programa NATURAL.

A grande vantagem desta cláusula é que o resultado de pesquisas muito grandes podem ser "retidas" e usadas para um posterior processamento sem necessidade de repetir toda a pesquisa.

- Set-Name (operado) → O nome através do qual se identifica estes registros "retidos" pode ser uma constante alfanumérica ou conteúdo de uma variável alfanumérica. Não é checada a duplicidade de nomes de conjuntos no RETAIN. O nome do conjunto de registros sobrepõe ao antigo, caso se use o mesmo nome.

- "Set-Name" liberados → Não há limite para o número de conjuntos que podem ser retidos ou o número de ISN's em um conjunto. Recomenda-se reter o menor número de registros necessários num determinado momento a fim de não encher a WORK FILE. A lista de ISN (dos registros retidos) que não for mais utilizada pode ser liberada com a instrução RELEASE SETS. Os conjuntos retidos valem para uma sessão NATURAL e não são liberados automaticamente. Um conjunto criado por um programa poder ser referenciado por um outro.

16.1.12.1. Acessos/Atualizações por outros usuários

Os registros identificados pelos ISN's em conjuntos retidos não são protegidos contra acessos e atualizações efetuadas por outros usuários.

16.1.12.2. Restrições

As cláusulas RETAIN e SORTED BY não podem ser usadas juntas.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 NAME
0040   2 BIRTH
0050   2 CITY
0060 END-DEFINE
0070 *
0080 FIND NUMBER EMP WITH BIRTH = 710230
0090     RETAIN AS 'TESTEEMP'
0100 *
0110 IF *NUMBER(0080) = 0
0120   STOP
0130 END-IF
0140 *
0150 FIND EMP WITH 'TESTEEMP' AND CITY = 'MADRID'
0160   DISPLAY NAME CITY BIRTH
0170 END-FIND
0180 *
0190 RELEASE SET 'TESTEEMP'
0200 END
```

Resultado:

```
NAME                 CITY           DATE
                                      OF
                                    BIRTH
-------------------- -------------------- --------

FREIXOMIL            MADRID              71/02/30
```

16.1.13. Cláusula WHERE

```
WHERE logical-condition
```

Esta cláusula é usada para especificar critério de pesquisa adicional que é levado em conta após a leitura dos registros que atendam a cláusula WITH, para depois em cima deste conjunto, fazer a seleção pelo WHERE. Isto ocorre antes que qualquer processamento seja feito (incluindo AT BREAK).

Os campos usados na condição lógica podem ser ou não descritores.

Os registros rejeitados (que não atendam a cláusula WHERE) não são levados em conta para o limite especificado dentro da instrução FIND. Entretanto, são considerados para qualquer limite estabelecido numa instrução GLOBALS, LIMIT ou num limite que seja estabelecido no momento da geração do NATURAL.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 NAME
0040   2 BIRTH
0050   2 CITY
0060 END-DEFINE
0070 *
0080 FIND EMP WITH CITY = 'BRASILIA'
0090     WHERE NAME = 'IVAN'
0100   DISPLAY NAME CITY
0110 END-FIND
0120 END
```

Resultado:

```
NAME                 CITY


-------------------- --------------------

IVAN                 BRASILIA
```

16.1.14. Cláusula IF NO RECORDS FOUND

```
IF NO RECORDS FOUND

    ENTER
    Statement ..

END-NOREC
```

Esta cláusula é usada para executar instruções se nenhum registro atender ao critério de pesquisa especificado na cláusula WITH e/ou WHERE.

As instruções estabelecidas no IF NO RECORDS FOUND são executadas antes de se iniciar as instruções do LOOP. Se nenhum processamento for necessário deverá ser usada a palavra chave ENTER.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 NAME
0040   2 BIRTH
0050   2 CITY
0060 END-DEFINE
0070 *
0080 FIND EMP WITH CITY = 'GOIANIA'
0090   IF NO
0100     WRITE 'REGISTRO INEXISTENTE.'
0110     ESCAPE BOTTOM
0120   END-NOREC
0130   DISPLAY NAME CITY BIRTH
0140 END-FIND
0150 END
```

16.1.14.1. Valores da Base de Dados

Os campos da base de dados que referenciam o arquivo no LOOP serão resetados, a menos que outros valores sejam assinalados dentro do IF NO RECORDS FOUND.

16.1.14.2. Avaliação das Funções do Sistema NATURAL

As funções são avaliadas uma vez para o restrito vazio que é criado como resultado da cláusula IF NO RECORDS FOUND.

16.1.14.3. Restrições

Esta cláusula não está disponível para as opções FIND FIRST, FIND NUMBER e FIND UNIQUE.

16.1.15. Variáveis do Sistema Disponíveis com FIND

As variáveis *ISN, *NUMBER e *COUNTER são automaticamente criadas para cada instrução FIND. Quando se usa estas variáveis fora do LOOP ou com as instruções FIND UNIQUE, FIND FIRST ou FIND NUMBER deve-se referenciar o número da linha da instrução FIND.

O formato e tamanho default de cada uma destas variáveis do Sistema é (P8). Este default não pode ser alterado.

- *ISN → Contém o ISN (internal sequence number) do registro que está sendo processado no LOOP. Esta variável não está disponível para a instrução FIND NUMBER.
- *NUMBER → Contém o número de registros que foram selecionados pela cláusula WITH.
- *COUNTER → Contém o número de vezes que o LOOP do FIND foi executado.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 NAME
0040   2 BIRTH
0050   2 CITY
0060 END-DEFINE
0070 *
0080 FIND EMP WITH CITY = 'MADRID'
0090   DISPLAY NAME CITY *ISN *NUMBER *COUNTER
0100 END-FIND
0110 END
```

Resultado:

```
NAME                 CITY                    ISN         NMBR        CNT
-------------------- -------------------- ----------- ----------- -----------

DE JUAN              MADRID                      400          40           1
DE LA MADRID         MADRID                      401          40           2
PINERO               MADRID                      405          40           3
TRANQUILINO          MADRID                      407          40           4
GOMEZ                MADRID                      408          40           5

...                  ...                         ...         ...         ...


SERRANO              MADRID                      495          40          37
DE LA IGLESIA        MADRID                      496          40          38
OSEA                 MADRID                      498          40          39
MARTINEZ             MADRID                      499          40          40
```

16.1.16. Várias instruções FIND

É possível criar ninhos de FIND's onde o LOOP mais interno é executado para cada registro selecionado no LOOP externo.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
```

```
0040   2 NAME
0050   2 FIRST-NAME
0060 1 VEI VIEW OF VEHICLES
0070   2 PERSONNEL-ID
0080   2 MAKE
0090 END-DEFINE
0100 *
0110 FIND EMP WITH NAME = 'IVAN'
0120   FIND VEI WITH PERSONNEL-ID = EMP.PERSONNEL-ID
0130     DISPLAY EMP.NAME VEI.MAKE
0140   END-FIND
0150 END-FIND
0160 END
```

Resultado:

```
NAME                 MAKE
-------------------- --------------------

IVAN                 VOLVO
```

16.1.17. FIND NUMBER

A instrução FIND NUMBER é usada para determinar o número de registros que satisfaçam o critério de seleção do WITH/WHERE. Não é gerado LOOP e nenhum campo do arquivo pesquisado estará disponível.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 FIRST-NAME
0060 END-DEFINE
0070 *
0080 FIND NUMBER EMP WITH NAME = 'MARIA'
0090 DISPLAY *NUMBER(0080)
0100 END
```

Resultado:

```
NMBR
-----------

          6
```

16.1.17.1. Restrições

As cláusulas SORTED BY e IF NO RECORDS FOUND não podem ser usadas com a instrução FIND NUMBER.

16.1.18. Variáveis do Sistema Disponíveis com FIND

As variáveis do Sistema disponíveis com FIND NUMBER são:

- *ISN → Contém o número de registros selecionados com a cláusula WITH.

16.2. Read

```
READ    operand1 RECORDS IN FILE view-name
BROWSE

   PASSWORD=operand2
   CIPHER=operand3
   WITH REPOSITION
   sequence/range-specification
   STARTING WITH ISN=operand4
   WHERE-clause
   statement ...

END-READ
```

A instrução READ é usada para ler registros de um arquivo de base de dados. Os registros podem ser lidos na seqüência física, na seqüência de ISN ou na seqüência lógica através de valores de pesquisa por um determinado descritor. Esta instrução gera um LOOP de processamento.

16.2.19. Processo de limite (*operando1*)

O número de registros a serem lidos pode ser limitado por uma constante numérica (0 a 99999999) ou uma variável, colocada entre parênteses, seguindo a palavra READ.

Qualquer limite especificado deve ser menor que o limite definido durante a instalação do NATURAL ou com a instrução LIMIT ou SET GLOBALS. Este limite sobrepõe qualquer outro limite que tenha sido definido anteriormente.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 FIRST-NAME
0060 END-DEFINE
0070 *
0080 READ(3) EMP
0090   DISPLAY PERSONNEL-ID NAME FIRST-NAME
0100 END-READ
0110 END
```

Resultado:

```
PERSONNEL         NAME              FIRST-NAME
   ID
--------- -------------------- --------------------
```

```
50005600  MIRIAM               MOURA
50005500  ARTHUR               MOURA
50005300  ARTHUR               MOURA
```

16.2.20. View-name

Nome da “view” como definida na área globals ou local. No modo estruturado, “view-name” tem que ser definida na área global ou local, usando o editor ou a instrução DEFINE DATA.

16.2.21. PASSWORD/CHIPER

A cláusula PASSWORD é usada para acessar arquivos ADABAS protegidos. A cláusula CIPHER é usada para acessar arquivos ADABAS cifrados.

16.2.22. Especificação de Seqüências

```
1. IN PHYSICAL ASCENDING           SEQUENCE
               DESCENDING
               VARIABLE operand5

2. BY    ISN   =              operand6  THRU      operand7
   WITH        EQ                       ENDING AT
               EQUAL TO
               STARTING FROM

3. IN LOGICAL ASCENDING          SEQUENCE
              DESCENDING
              VARIABLE operand5

     BY    descriptor  =              operand6  THRU      operand7
     WITH              EQ                       ENDING AT
                       EQUAL TO
                       STARTING FROM
```

PHYSICAL SEQUENCE é usada para ler registros na ordem em que ele estão fisicamente armazenados na base de dados. Esta é a seqüência default.

BY ISN é usada para ler registros na seqüência de ISN (Internal Sequence Number).

LOGICAL SEQUENCE é usada para ler registros na ordem ascendente dos valores de um descritor. Se LOGICAL for especificado com um descritor, os registros serão lidos na seqüência dos valores do descritor. Um descritor, sub-descritor ou super-descritor podem ser usados para controle de seqüência. Um descritor fonético ou um descritor dentro de um grupo periódico não pode ser usado.

Se o descritor usado for definido com NULL VALUE SUPRESSION, aqueles que tem valores nulos não serão lidos.

Se o descritor usado for um campo múltiplo o mesmo registro será lido múltiplas vezes dependendo do número de valores armazenados.

As cláusulas STARTING FROM e ENDING AT são usadas para especificar limites de valores para pesquisa. Os termos THRU e ENDING AT implicam em leitura do limite superior inclusive. Se um valor inicial for especificado, a leitura começará com o valor especificado. Se o valor inicial não for encontrado, inicia-se a partir do valor imediatamente superior. Se até ao limite superior não for encontrado valores, o LOOP não será executado. Valores ENDING AT não podem ser especificados com o uso de sub ou super-descritores. Um campo múltiplo não pode ser usado com a opção ENDING AT. A palavra chave EQ ou "=" somente estabelece o valor inicial para a operação de READ. O valor final deve ser especificado com a opção ENDING AT/THRU.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 FIRST-NAME
0060 END-DEFINE
0070 *
0080 WRITE 'READ FISICO'
0090 READ(5) EMP
0100   DISPLAY *ISN PERSONNEL-ID NAME FIRST-NAME
0110 END-READ
0120 *
0130 WRITE 'READ POR ISN'
0140 READ(5) EMP BY ISN
0150   DISPLAY *ISN PERSONNEL-ID NAME FIRST-NAME
0160 END-READ
0170 *
0180 WRITE 'READ POR NOME'
0190 READ(5) EMP BY NAME
0200   DISPLAY *ISN PERSONNEL-ID NAME FIRST-NAME
END-READ
*
WRITE 'READ POR NOME INICIANDO DE L'
READ(5) EMP BY NAME = 'L'
  DISPLAY *ISN PERSONNEL-ID NAME FIRST-NAME
END-READ
END
```

Resultado:

```
ISN      PERSONNEL         NAME              FIRST-NAME
               ID
---------- --------- -------------------- --------------------

READ FISICO
          2 50005600  MIRIAM               MOURA
          3 50005500  ARTHUR               MOURA
          4 50005300  ARTHUR               MOURA
          5 50004900  ARTHUR               MOURA
          6 50004600  ARTHUR               MOURA
READ POR ISN
          1 9990001   MIRIAM               MOURA
          2 50005600  MIRIAM               MOURA
          3 50005500  ARTHUR               MOURA
          4 50005300  ARTHUR               MOURA
          5 50004900  ARTHUR               MOURA
READ POR NOME
```

```
          1302 34356739
          1413 12000001                         LISBELA
          1421 12000002                         FULANO DE TAL
          1429 12000003                         FULANA DE TAL
          1437 12000004                         RUTINEIA FREI
    READ POR NOME INICIANDO DE L
           658 F6113745  LAERCIO                TERUYA
           181 50001800  LAFON                 CHRISTIANE
           312 11400320  LANDMANN              HARRY
           938 30008544  LANE                  JACQUELINE
           256 11100309  LANKATILLEKE          LALITH
```

16.2.23. Cláusula WHERE

```
WHERE condição-lógia
```

A cláusula WHERE é usada para especificar outros critérios de seleção que serão executados depois dos registros lidos e antes que qualquer processamento seja executado com o registro (incluindo processo de quebra).

Se a instrução LIMIT ou um processo de limite for especificado na instrução READ que contém a cláusula WHERE, os registros rejeitados não são levados em conta para efeito do limite especificado.

Variáveis do Sistema NATURAL disponíveis com READ:

- *ISN → Contém o ISN (internal sequence number) do registro que está sendo processado no LOOP.
- *COUNTER → Contém o número de vezes que o LOOP do READ foi executado.

16.3. Combinação de instruções READ e FIND

O exemplo abaixo lê registros do arquivo EMPLOYEES em seqüência lógica baseado no descritor NAME. A instrução FIND é então usada para o arquivo VEHICLES usando PERSONNEL-ID do arquivo EMPLOYEES como critério de pesquisa. O relatório mostra o nome (obtido do arquivo EMPLOYEES) de cada pessoa lida e o modelo do automóvel (obtido do arquivo VEHICLES) pertencente a estas pessoas. Múltiplas linhas com o mesmo nome serão produzidas se a pessoa tiver mais de um automóvel.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
```

```
0040   2 NAME
0050   2 FIRST-NAME
0060 *
0070 1 VEI VIEW OF VEHICLES
0080   2 PERSONNEL-ID
0090   2 MAKE
0100 END-DEFINE
0110 *
0120 LIMIT 10
0130 *
0140 READ EMP BY NAME = 'MARIA'
0150   FIND VEI WITH PERSONNEL-ID = EMP.PERSONNEL-ID
0160     IF NO
0170       ESCAPE BOTTOM
0180     END-NOREC
0190     DISPLAY EMP.PERSONNEL-ID EMP.NAME VEI.MAKE
0200   END-FIND
0210 END-READ
0220 END
```

Resultado:

```
PERSONNEL         NAME                 MAKE
   ID
--------- -------------------- --------------------

20017100  MARKUSH              MERCEDES-BENZ
20005100  MARKUSH              FORD
```

16.4. Get

```
                            GET IN FILE view-name

                                PASSWORD = operand1

                                CIPHER = operand2

                                RECORD operand3
                                *ISN (r)
```

A instrução GET é usada para ler um registro a partir de seu ISN. Esta instrução não gera LOOP.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 FIRST-NAME
0100 END-DEFINE
0110 *
0120 GET EMP 1117
0130 DISPLAY *ISN PERSONNEL-ID NAME
0140 END
```

Resultado:

```
ISN      PERSONNEL         NAME
               ID
----------- --------- --------------------

       1117 12312312  MARIA
```

16.5. Histogram

```
HISTOGRAM operand1 VALUE IN FILE view-name

    PASSWORD=operand2
    SEQUENCE-clause
    VALUE FOR FIELD operand4
    STARTING/ENDING-clause
    WHERE logical-condition

    Statement ...

END-HISTOGRAM
```

A instrução HISTOGRAM é usada para ler valores de campos descritores, sub-descritores ou super-descritores. Estes valores são lidos diretamente das listas invertidas do ADABAS e retornam em ordem crescente.

Esta instrução gera LOOP, mas não é feito nenhum acesso a registro de dados. Desta forma apenas os campos descritores usados na instrução HISTOGRAM estão disponíveis.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 NAME
0040 END-DEFINE
0050 *
0060 HISTOGRAM EMP NAME STARTING FROM 'MARIA'
0070   DISPLAY NAME
0080 END-HISTOGRAM
0090 END

Resultado:

NAME
--------------------

MARIA
MARIA SILVA
MARIO
MARKUSH
```

16.5.1. Limite do processo de LOOP (operand1)

O número de registros a serem processados com a instrução HISTOGRAM pode ser limitado especificando uma constante numérica (0 até 99999999) ou uma variável do usuário (somente formato N ou P). O limite sobrepõe qualquer limite em efeito definido previamente. O limite não pode exceder qualquer limite global que tenha sido estabelecido em parâmetros ativados no NATURAL ou especificados com o comando GLOBALS ou instrução SET GLOBALS.

16.5.2. Cláusula PASSWORD

A cláusula PASSWORD é usada para acessar arquivos ADABAS protegidos.

16.5.3. Cláusula STARTING/ENDING

Pode-se estabelecer intervalos de pesquisa com o uso da palavra chave THRU ou STARTING (iniciando em) ou ENDING (terminando em), seguidos por uma constante ou uma variável que identificar critério adicionais que serão executados após a leitura dos valores antes de qualquer processamento (incluindo a execução do AT BREAK).

Caso o valor de início não seja encontrado a pesquisa começará no valor imediatamente superior. Se o limite superior não for achado a pesquisa terminará no valor imediatamente anterior aquele limite estabelecido.

Valores hexadecimais são aceitos como limite de pesquisa se os descritores forem de formato alfanumérico (A) ou binário (B).

16.5.4. Cláusula WHERE

```
WHERE condição-lógia
```

A cláusula WHERE é usada para especificar critérios de seleção adicionais que serão executados após a leitura dos valores e antes de qualquer processamento (incluindo a execução do AT BREAK).

O descritor usado no WHERE precisa ser o mesmo usado no HISTOGRAM.

16.5.5. Variáveis do Sistema Disponíveis com HISTOGRAM

O formato e tamanho default de cada uma destas variáveis do Sistema é (P8). Este default não pode ser alterado.

- *ISN → Indica o número da(s) ocorrência(s) nas quais o critério de pesquisa foi encontrado. Retornará zero se o descritor não estiver em um grupo periódico.
- *NUMBER → Indica o número de registros correspondente a cada quebra ocorrida no critério de pesquisa.
- *COUNTER → Contém o número de LOOP's executados pelo HISTOGRAM.

16.6. Store

```
STORE  RECORD  IN  FILE view-name
       PASSWORD=operand1
       CIPHER=operand2

       USING   NUMBER operand3  (r)
       GIVING
```

A instrução STORE é usada para adicionar um registro na base de dados.

16.6.1. View-name

O nome da "view" como definido na área de dados local ou global.

No modo estruturado, "view-name" deve ser o nome da "view" como definido na área de dados local ou globals, usando o editor da área de dados ou a instrução DEFINE DATA.

16.6.2. Cláusula PASSWORD/CIPHER

A cláusula PASSWORD é usada para prover uma "password" quando for fazer alterações nos dados de arquivos ADABAS que são protegidos.

A cláusula CIPHER é usada para prover um código quando for fazer alterações nos dados de arquivos ADABAS cifrados.

16.6.3. Cláusula USING/GIVING NUMBER

Esta opção é usada para gravar o registro com um número de ISN fornecido pelo usuário.

Se já existir um registro com o ISN especificado ocorrerá um erro e o programa terminará a menos que ON ERROR seja especificado.

16.6.4. Variável do Sistema *ISN

A variável do Sistema NATURAL *ISN contém um ADABAS ISN atribuído ao novo registro como resultado da execução da instrução STORE. Uma referência subseqüente ao *ISN deve conter o número da linha da instrução STORE.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 CITY
0060 END-DEFINE
0070 *
0080 PERSONNEL-ID := 212121
0090 NAME := 'PERDIGAO'
0100 CITY := 'IPABA'
0110 *
0120 STORE EMP
0130 END TRANSACTION
0140 END
```

16.7. Update

```
UPDATE RECORD IN STATEMENT (r)
```

A instrução UPDATE é usada para atualizar um ou mais registros na base de dados. O registro a ser atualizado deve ter sido previamente selecionado usando a instrução FIND, GET, READ ou STORE.

16.7.1. Restrições

A instrução UPDATE não pode estar na mesma linha da instrução que obteve o registro .

16.7.2. Notação de Referência

A notação “(r)“ é usada para indicar a instrução na qual o registro a ser modificado foi lido. Se nenhuma referência for especificada, será assumido o registro selecionado pelo LOOP mais interno. “(r)” pode ser especificado como o número da linha da instrução ou o “label”.

16.7.3. Hold Status

O uso da instrução UPDATE faz com que cada registro lido, no correspondente FIND ou READ, seja colocado em HOLD.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 CITY
0060 END-DEFINE
0070 *
0080 FIND EMP WITH PERSONNEL-ID = '230'
0090   WRITE 'ANTES DO UPDATE'
0100   WRITE '=' PERSONNEL-ID '=' NAME '=' CITY
0110 *
0120   NAME := 'PERDIGAO'
0130   CITY := 'BRASILIA'
0140 *
0150   UPDATE (0080)
0160 *
0170   WRITE // 'APOS O UPDATE'
0180   WRITE '=' PERSONNEL-ID '=' NAME '=' CITY
0190 END-FIND
0200 END TRANSACTION
0210 END
```

Resultado:

```
ANTES DO UPDATE
PERSONNEL ID: 230      NAME: QQQQQQQQQQQQQQ       CITY: SAO PAULO


APOS O UPDATE
PERSONNEL ID: 230      NAME: PERDIGAO             CITY: BRASILIA
```

16.8. Delete

```
DELETE RECORD IN STATEMENT (r)
```

A instrução DELETE é usada para deletar registros de uma base de dados.

16.8.4. Restrições

A instrução DELETE não pode ser especificado na mesma linha da instrução FIND, READ ou GET.

16.8.5. Notação de Referência

A notação “(r)“ “(label ou número)” é usada para referenciar a linha onde o registro foi obtido.

Se nenhuma referência for especificada, assume-se o LOOP ativo mais interno, no qual o registro foi obtido.

16.8.6. Hold Status

Todos os registros acessados pelo FIND ou READ são colocados em HOLD, até que seja executado um END TRANSACTION a cada 100 registros, para que estes sejam deletados fisicamente, evitando que a fila onde ficam armazenados os registros a serem atualizados/deletados se sature.

```
0010 DEFINE DATA LOCAL
0020 1 EMP VIEW OF EMPLOYEES
0030   2 PERSONNEL-ID
0040   2 NAME
0050   2 CITY
0060 END-DEFINE
0070 *
0080 FIND EMP WITH PERSONNEL-ID = '230'
0090   DELETE (0080)
0100 END-FIND
0110 END TRANSACTION
0120 END
```

16.9. End Transaction

A instrução END TRANSACTION é usada para indicar o fim de um transação lógica. Uma transação lógica é a menor unidade de trabalho (como definida pelo usuário) que precisa ser executada em sua totalidade para assegurar que informação contida na base de dados seja logicamente consistente.

A execução bem sucedida do END TRANSACTION assegura que todas as atualizações feitas durante a transação sejam fisicamente aplicadas as bases de dados sem levar em conta subseqüentes usuários, sessões NATURAL, ADABAS ou interrupções do sistema operacional.

Atualizações para as quais o END TRANSACTION não foi completado, serão abandonadas pelo ADABAS automaticamente.

Esta instrução libera os registros retidos pelo usuário durante a transação e pode ser executada sob uma condição lógica.

16.10. Backout Transaction

A instrução BACKOUT TRANSACTION é usada para abandonar todas as atualizações feitas numa transação lógica, isto é, antes do END TRANSACTION.

Também libera todos os registros mantidos em “hold” durante a transação.

17. READ WORK FILE

```
READ WORK FILE work-file-number ONCE
   AT END OF FILE
      Statement ...
   END-ENDFILE
   Statement...
END-WORK
```

A instrução READ WORK FILE é usada para ler dados de um arquivo seqüencial não ADABAS/DB2. É necessário o uso do JCL quando o arquivo de trabalho for lido. Esta instrução somente pode ser usada dentro de um programa que será executado em "batch mode". Causa o início de um laço de processamento (LOOP) e todos os registros do arquivo de trabalho serão lidos e executados.

17.1. Work-file-number

É o número do arquivo de trabalho (como definido para o NATURAL) a ser lido.

17.2. Opção ONCE

ONCE é usada para indicar que somente um registro será lido. Nenhum processo de LOOP é iniciado. Se ONCE for especificado, a cláusula AT END OF FILE pode, também, ser usada.

17.3. AT END OF FILE

Se a opção ONCE for usada, esta cláusula deve ser especificada para indicar a ação a ser tomada, quando a condição de final de arquivo for detectada.

```
0010 DEFINE DATA LOCAL
0011 1 EMP VIEW OF EMPLOYEES
0012   2 PERSONNEL-ID
0013   2 NAME
0014   2 CITY
0015 *
0020 1 #REGISTRO
0030   2 #CODIGO       (N5)
```

```
0040   2 #NOME         (A40)
0050   2 #CIDADE       (A25)
0060 END-DEFINE
0070 *
0080 READ EMP BY NAME
0081   #CODIGO := EMP.PERSONNEL-ID
0082   #NOME   := EMP.NAME
0083   #CITY   := EMP.CITY
0084 *
0086   WRITE WORK FILE 1 #REGISTRO
0100 END-READ
0120 END
```

18. WRITE WORK FILE

```
WRITE WORK FILE work-file-number
```

A instrução WRITE WORK FILE é usada para gravar registros em arquivos de trabalhos seqüencial. Esta instrução é aplicável em "batch mode". É possível criar e ler um arquivo de trabalho no mesmo programa.

18.1. Work-file-number

É o número do arquivo de trabalho (como definido para o NATURAL) a ser usado.

```
0010 DEFINE DATA LOCAL
0070 1 #REGISTRO
0080   2 #CODIGO       (N5)
0090   2 #NOME         (A40)
0100   2 #CIDADE       (A25)
0110 END-DEFINE
0120 *
0130 READ WORK FILE 1 #REGISTRO
0140   DISPLAY #CODIGO #NOME #CIDADE
0190 END-WORK
0200 END
```

19. PERFORMANCE DE APLICAÇÕES

19.1. Princípios Básicos de Programação Natural 2

Esta é uma síntese dos critérios que devem ser seguidos para uma programação Natural eficiente:

- Utilize os comandos mais adequados para acesso ao ADABAS
- Evite repetir ou omitir comandos desnecessários
- Evite ler/atualizar mais campos do Banco de Dados do que os estritamente necessários
- Minimizem o número de comandos ADABAS por transação (entre um Enter e outro)
- Utilize os limites disponíveis no Natural para os comandos de acesso ao ADABAS
- Otimize o ciclo de atualização
- Otimize as validações de campos
- Minimize os LOOPs
- Otimize o processamento condicional
- Parametrize os ARRAYs
- Otimize as instruções simples
- Otimize seu uso da precisão nas operações aritméticas
- Modularize corretamente
- Desenhe corretamente seus mapas e windows
- Utilize tabelas residentes na memória para otimizar a utilização repetitiva das mesmas

19.2. Otimização de Comandos de Acesso ao ADABAS

A otimização dos comandos de acesso ao ADABAS começa na definição das views, que serão utilizadas por estes comandos.

19.2.1. Estudo de alguns casos freqüentemente encontrados em PGMs

19.2.1.1. Views

Raramente se justifica, do ponto de vista da aplicação, o uso de views muito extensas. Quando isto ocorrer, desconfie, não se justifica apresentar ao usuário uma quantidade muito grande de informações.

É melhor para os usuários e muito melhor para o desempenho dos sistemas, ainda que exija uma programação mais trabalhosa, fornecer telas menos poluídas, contendo apenas as informações estritamente necessárias.

Somente quando o usuário solicitar algumas informações adicionais, a aplicação as deverá ler.

Seguem algumas dicas:

- Defina uma view para cada comando de acesso, evitando usar a mesma view para mais de um comando
- Se uma view exigir um número muito elevado de campos, revise a aplicação
- Use labels nos comandos (para evitar referências às linhas)

Os nomes das views devem ser pequenos

19.2.1.2. Pesquisa de Existência

Temos as seguintes opções de programação:

```
1. FIND NUMBER view WITH chave = valor
2. FIND (1) view WITH chave = valor
3. READ (1) view BY chave = valor
4. HISTOGRAM (1) view FOR chave STARTING valor
```

A alternativa número 4 é sempre a mais eficiente.

19.2.1.3. Leitura de um único registro com condição *chave = valor*

Temos as seguintes opções de programação:

```
1. FIND (1) view WITH chave = valor
2. READ (1) view BY chave = valor THRU valor
```

Se o registro tiver uma alta probabilidade de não existir, use a opção número 1, pois:

- O FIND pesquisará a lista invertida de chave procurando valor, se não o encontrar encerra o comando
- O READ pesquisará a lista invertida de chave procurando valor, caso não o achar, procura o valor seguinte, lendo o registro correspondente

Se o registro tiver uma alta probabilidade de existir, use a opção número 2, pois o comando READ será neste caso mais eficiente que o FIND.

19.2.1.4. Leitura de todos os registros com condição *chave = valor*

Temos as seguintes opções de programação:

```
1. FIND view WITH chave = valor
2. READ view BY chave = valor THRU valor
```

Se o arquivo estiver fisicamente classificado por chave escolha a opção número 2.

Em todos os outros casos, escolha a opção número 1, pois a opção 2 executa sempre uma leitura a mais.

Se o arquivo estiver classificado por chave, a opção 2 será muito eficiente, caso o mesmo estiver classificado por ISN a melhor opção será a número 1. Reorganize freqüentemente seus arquivos, pela chave mais utilizada ou pelo ISN.

19.2.1.5. Leitura dos registros com condição chave = faixa-valores

Temos as seguintes opções de programação:

```
1. FIND view WITH chave = valor-i THRU valor-f
2. READ view BY chave = valor-i THRU valor-f
```

Se o arquivo estiver fisicamente classificado por chave escolha a opção número 2.

Em todos os outros casos, escolha a opção 1, pois a opção 2 executa sempre uma leitura a mais.

19.2.1.6. Leitura dos registros com condição chave=faixa-valores, classificados por chave

Temos as seguintes opções de programação:

```
1. READ view BY chave = valor-i THRU valor-f
2. READ view BY PHYSICAL WHERE chave = valor-I THRU valor-
   f SORT BY chave USING…
3. FIND view WITH chave = valor-i THRU valor-f SORT BY
   chave USING…
4. FIND view WITH chave = valor-i THRU valor-f SORTED BY
   chave
```

Se o arquivo estiver fisicamente classificado por chave escolha sempre a opção número 1.

Se o número de registros lidos for pequeno, escolha a opção número 1.

Se o número de registros lidos for grande, escolha as opções 2 ou 3, dependendo de:

- Se o total de registros lidos for muito maior que o total de blocos do arquivo no DATA, escolha a opção número 2.
- Caso contrário, escolha a opção número 3.

19.2.1.7. Seleção de Registros Segundo Condição Complexa

Primeiro deve ser analisada a possibilidade e conveniência de simplificar a expressão pelo uso de super-descritores.

Vencida esta etapa, deve ser considerada a possibilidade de separar a expressão complexa em duas, sendo que a primeira deverá ter um nível de filtragem alto, quase equivalente à original, e a segunda será apenas complementar.

Temos as seguintes opções de programação:

```
1. FIND view WITH expressão
2. READ view PHYSICAL WHERE expressão
3. FIND view WITH sub-expressão-1 WHERE sub-expressão-2
```

Se todos os descritores de expressão apresentarem um nível de filtragem equivalente, devemos adotar a opção número 1.

Se sub-expressão-1 selecionar um número não muito elevado de registros, devemos adotar a opção número 3.

Se o número de registros selecionados por expressão for muito maior que o total de blocos do arquivo no DATA, deveremos utilizar a opção número 2.

19.2.1.8. Leitura batch de todos os registros de um arquivo

Teremos as seguintes opções:

```
1. READ view PHYSICAL
2. READ view BY ISN...
3. READ view BY descriptor
```

Nunca escolha a opção número 1 se o programa for atualizar os registros lidos ou se o mesmo arquivo pode estar sendo atualizado concorrentemente.

Escolha a opção número 2 ou a número 3, dependendo de qual seja a ordem física do arquivo (por ISN ou pelo descritor da opção 3).

19.2.1.9. Utilização dos comandos GET e GET SAME

Existem as seguintes possibilidades:

- GET para acesso pelo ISN
- GET falso para evitar que tenhamos que voltar a ler o mesmo registro para atualizá-lo
- GET para ler campos adicionais

GET SAME para se ler mais ocorrências dos mesmos campos múltiplos, periódicos e/ou múltiplos em periódicos, das que tenham sido lidas, por um comando anterior, utilizando a mesma View.

19.2.1.10. Utilização de limites

Se ao selecionar e/ou ler registros de um arquivo, ou valores de uma lista invertida, soubermos da existência de limites (número máximo de registros que cumprem uma determinada condição ou que tem um certo valor), deve-se aproveitar esta informação nos programas, inserindo esses limites nos comandos apropriados.

Desta forma, estaremos incluindo no programa, sem a necessidade de código adicional, um mecanismo importante para detectar possíveis erros de integridade lógica no arquivo.

Especifique o parâmetro LE=ON.

Para evitar os erros que, indevidamente, este parâmetro pode provocar, utilize ESCAPE BOTTOM.

Exemplo:

```
FIND(1) VIEW
   Processa
   ESCAPE BOTTOM
END-FIND
```

Parametrize os limites e insira os parâmetros nos comandos correspondentes.

Exemplo:

```
01 #LIM-FIND (P2) CONST <50>
...
FIND ( #LIM-FIND ) view WITH condição
   Processa
```

```
END-FIND
```

O erro NAT1005 significa que o arquivo está, possivelmente, com algum problema de integridade.

19.3. Ciclo de atualização – Opções e Recomendações

Critério para definir freqüência de END TRANSACTION

O comando END TRANSACTION, essencial para a manutenção da integridade física e lógica do banco de dados ADABAS, é um grande consumidor de recursos.

O objetivo desta análise é minimizar sua freqüência sem por em risco a integridade do banco de dados. Tal análise deverá considerar tanto os processos on-line quanto os batch.

19.3.1. Batch

A não emissão de comandos de END TRANSACTION para todas as alterações lógicas efetivadas por um programa batch terá as seguintes conseqüências:

No caso do processo finalizar anormalmente, o seu reinicio sofrerá um atraso equivalente à duração de cada transação. Este atraso não deverá ser, em média, maior do que 1 minuto, se adotarmos um END TRANSACTION para cada 100 transações (não é significativo).

Se o processamento batch e on-line for concorrente, os registros que permanecerão em HOLD por aproximadamente 1 minuto, poderão provocar conseqüências nefastas no tempo de resposta de um número bastante elevado de usuários. Neste caso, será obrigatória a opção WH=OFF para o NATURAL on-line, pois evitará que os usuários deste ambiente concorrentes com o batch, ao ficarem em WAIT, prendam as THREADS e consequentemente engarrafem o ambiente todo.

Se houver concorrência Batch e On-line, o Natural On-line deverá usar WH=OFF e os processos Batch não poderão adotar menos do que um END TRANSACTION para cada 10 transações lógicas.

19.3.2. On-line

Os programas on-line devem emitir exatamente um END TRANSACTION para todas as transações lógicas do Banco de Dados, resultantes do ingresso, pelo operador do terminal, de uma transação lógica de usuário (uma ou mais telas).

Exemplo:

- Um usuário pode digitar um pedido com muitos itens utilizando várias telas e a menos que se adote um modelo que aceite um pedido parcial, o END TRANSACTION deverá ser emitido tão somente quando forem ingressados todos os itens do pedido.

- Outro usuário que ingresse numa tela vários lançamentos contábeis, lidos de um formulário, deverá provocar que as muitas transações resultantes da tela sejam efetivadas em conjunto por um único END TRANSACTION.

Não há, portanto, neste caso, muita flexibilidade para que o programador possa definir a forma mais eficiente de emitir comandos END TRANSACTION. Cabe ao projetista definir a relação entre os dados ingressados nas telas e o Modelo de Dados da organização. Não se pode porém esquecer que o HOLD de registros que forem solicitados por outros usuários afeta o tempo de resposta e pode até parar o ambiente.

19.4. Técnicas de programação que melhoram o desempenho

Um registro que já foi lido, em qualquer local do programa, pode ser atualizado sem repetir a leitura.

Evite sempre que possível, excluir e/ou incluir muitos registros por programa. Substitua pela execução do utilitário ADALOD.

Evite, sempre que possível, processar por programa alterações demoradas numa grande quantidade de registros. Considere a possibilidade de alterar uma cópia seqüencial do arquivo e posteriormente efetivar a sua carga (ADACMP e ADALOD).

Evite processar on-line o que pode ser diferido e processado em batch.

Se tiver que incluir e/ou alterar registros, e muitos dos campos previstos nestas operações forem raramente informados, programe os STORE e/ou UPDATE para serem executados em duas etapas. Primeiro execute os comandos contendo os campos que serão informados em 90% dos casos ou mais, e quando necessário (menos do que 10% das vezes), emita os comandos de UPDATE com os campos restantes.

19.5. Acesso ótimo a campos múltiplos e periódicos

Campos múltiplos e periódicos são recursos muito importantes para a melhora de performance das aplicações via desnormalização.

Como contrapartida, o seu desempenho pode não atender às expectativas e a programação e manutenção pode tornar-se mais complexa se não forem tomados certos cuidados:

- Ler e atualizar um número mínimo de ocorrências
- Ter rotinas padronizadas para manuseio dos campos, simplificando assim a sua programação e manutenção
- Parametrizar o número de ocorrências processadas, para poder adaptar-se às mudanças ambientais e facilitar a sua manutenção.

Ler, incluir e atualizar um número mínimo de ocorrências

Podem ser desenvolvidos utilitários muito simples para minimizar o número de ocorrências lidas, calculando a média de ocorrências, desvio padrão, curva de Gauss, etc.

Para atualizar/incluir, deve-se utilizar também o número médio de ocorrências e se houverem mais serão emitidos comandos adicionais de atualização.

Rotinas padronizadas

Quaisquer dos casos apresentados podem ser tratados por “esqueletos” padronizados que podem (devem) ser utilizadas pelas equipes de programação.

Isto acompanha a nova tecnologia de programação por objetos. Os campos múltiplos e/ou periódicos são elementos dos arquivos que pela sua vez são objetos ou parte de objetos. Estes objetos

devem ser complementados por funções (rotinas) padronizadas que concentrarão toda e qualquer solicitação a esses objetos.

19.6. Processamento de REPEAT e FOR

O FOR é 50% menos eficiente em termos de consumo de CPU, que o REPEAT equivalente:

```
0010 DEFINE DATA LOCAL
0020 1 #I       (N2)
0030 END-DEFINE
0040 *
0050 REPEAT
0060   UNTIL #I > 9
0070   ADD 1 TO #I
0080   WRITE '=' #I
0090 END-REPEAT
0100 *
0110 SKIP 1
0120 *
0130 FOR #I 1 10
0140   WRITE '=' #I
0150 END-FOR
0160 END

Resultado:

#I:   1
#I:   2
#I:   3
#I:   4
#I:   5
#I:   6
#I:   7
#I:   8
#I:   9
#I:  10

#I:   1
#I:   2
#I:   3
#I:   4
#I:   5
#I:   6
#I:   7
#I:   8
#I:   9
#I:  10
```

Se precisar usar o FOR descendente, não utilize a solução abaixo e sim a última:

```
0010 DEFINE DATA LOCAL
0020 1 #I       (N2)
0030 1 #J       (N2)
0040 END-DEFINE
0050 *
0060 FOR #I 1 10
0070   #J := 11 - #I
0080   WRITE '=' #J
```

```
0090 END-FOR
0100 *
0110 SKIP 1
0120 *
0130 FOR #J 10 1 -1
0140   WRITE '=' #J
0150 END-FOR
0160 END

Resultado:

#J:  10
#J:   9
#J:   8
#J:   7
#J:   6
#J:   5
#J:   4
#J:   3
#J:   2
#J:   1

#J:  10
#J:   9
#J:   8
#J:   7
#J:   6
#J:   5
#J:   4
#J:   3
#J:   2
#J:   1
```

19.7. Processamento de ACCEPT e REJECT

Ainda que a diferença de performance não é significativa, ACCEPT e REJECT são comandos que apresentarão melhor performance se não forem misturados.

O que sim pode sim pode interessar é o perigo de misturá-los.

19.8. Precisão em operações Aritméticas

Nunca utilize a opção ROUNDED, pois consome uma quantidade exagerada de CPU e não é otimizável pelo Natural nem pelo NOC (Natural Optimizer Compiler).

Some `0,...5` ao valor que se quer arredondar. Sempre utilize uma posição a mais do que a precisão máxima do resultado.

Esta alternativa ao ROUNDED não se aplica se o resultado tiver mais de 7 casas decimais.

```
0010 DEFINE DATA LOCAL
0020 1 #N     (N3,1) INIT<1,4>
0030 1 #N1    (N3,2) INIT<1,46>
0040 *
0050 1 #R       (N3)
0060 1 #R1      (N3,1)
0070 END-DEFINE
```

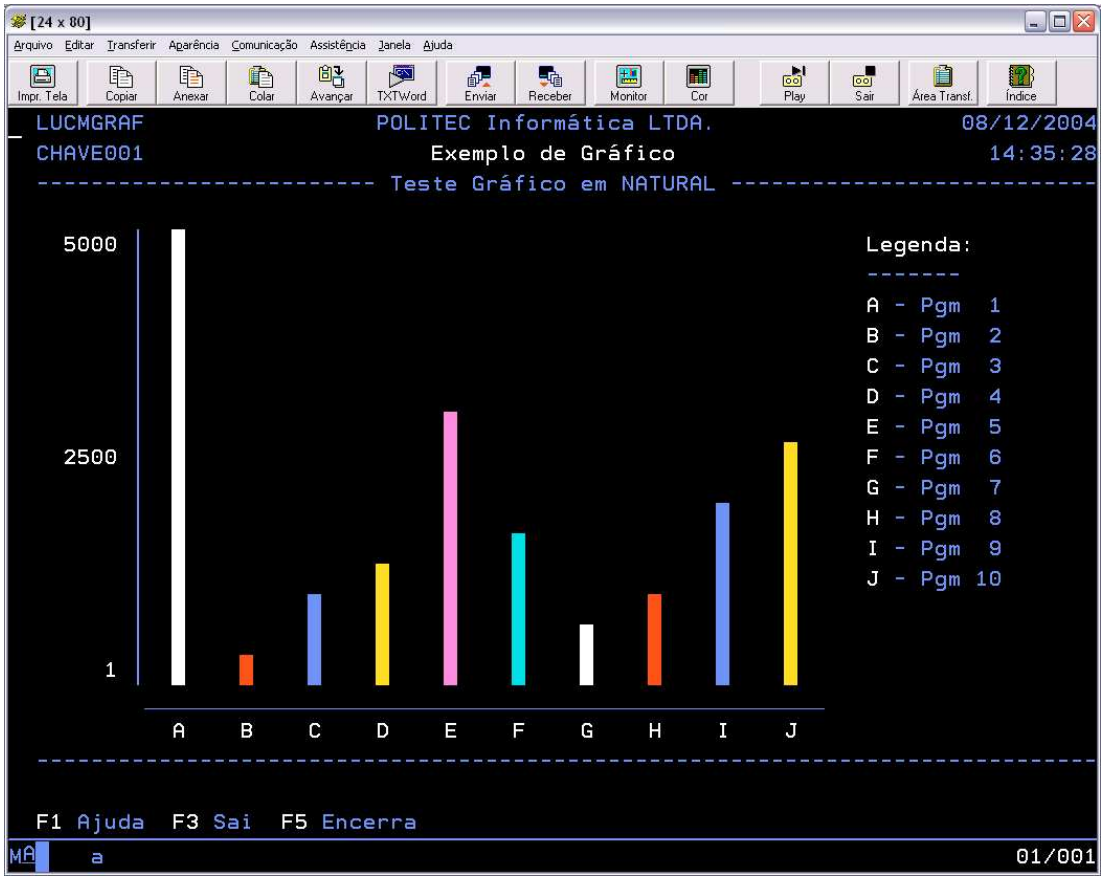
```
0080 *
0090 WRITE '=' #N
0100 #R := #N + 0,5
0110 WRITE '=' #R //
0120 *
0130 WRITE '=' #N1
0140 #R1 := #N1 + 0,05
0150 WRITE '=' #R1
0160 END
```

Resultado:

```
#N:    1,4
#R:    1


#N1:    1,46
#R1:    1,5
```

20. EXEMPLO DE GRÁFICO EM NATURAL

Abaixo é mostrado um exemplo de um programa e um mapa que mostra um gráfico em NATURAL.




```
0010 ************************************************************************
0020 ** Programa : PERPGRAF
0030 ** Sistema  : Curso Avançado de Natural.
0040 **
0050 ** Função   : Exemplo de Gráfico em Natural.
0060 **
0070 ** Analista : Luciano Perdigão
0080 ** Autor    : Luciano Perdigão
0090 ** Data     : 16/12/2002
0100 ************************************************************************
0110 * VRS001 - 16/12/2002 - Perdigão - Implantação
0120 *
0130 DEFINE DATA LOCAL
0140 * --------------------------
0150 * Definição Variáveis do mapa
0160 * --------------------------
0170 1 #GRAFICO       (A2/15,10)
0180 1 #CV-GRAFICO    (C/15,10)
0190 * ---------------------
0200 * Variáveis de trabalho
```

```
0210 * --------------------
0220 1 #I                 (I2)
0230 1 #J                 (I2)
0240 1 #TAMANHO-BARRA-AUX (N2,2/10)
0250 1 #TAMANHO-BARRA     (N2/10)
0260 1 #A                 (N15,2/10)
0270 1 #COR               (A1/10)
0280   INIT <'@','#','$','&','*','(','@','#','$','&'>
0290 END-DEFINE
0300 *
0310 SET KEY ALL
0320 SET CONTROL 'M23'
0330 *
0340 #A(1)  := 4000,50
0350 #A(2)  :=    0,00
0360 #A(3)  := 1000,05
0370 #A(4)  := 1100,00
0380 #A(5)  := 2500,00
0390 #A(6)  := 1400,90
0400 #A(7)  :=  600,00
0410 #A(8)  :=  950,20
0420 #A(9)  := 1800,45
0430 #A(10) := 2300,00
0440 *
0450 FOR #I 1 10
0460   #TAMANHO-BARRA-AUX(#I) := (#A(#I) * 15) / 4000
0470   #TAMANHO-BARRA(#I)     := #TAMANHO-BARRA-AUX(#I)
0480   IF #TAMANHO-BARRA(#I) = 0
0490     #TAMANHO-BARRA(#I)   := 1
0500   END-IF
0510   #TAMANHO-BARRA(#I)     :=  15 - (#TAMANHO-BARRA(#I) - 1)
0520 END-FOR
0530 *
0540 DISPLAY #TAMANHO-BARRA(*) #TAMANHO-BARRA-AUX(*)
0550 *
0560 RESET #J
0570 *
0580 REPEAT
0590   ADD 1 TO #J
0600 * -------------------------
0610 * Carrega variáveis do mapa
0620 * -------------------------
0630   FOR #I #TAMANHO-BARRA(#J) 15
0640     #CV-GRAFICO(#I,#J) := (AD=V)     /** Coloca a barra em vídeo reverso
0650     #GRAFICO(#I,#J)    := #COR(#J)   /** Coloca a cor na barra
0660   END-FOR
0670   IF #J < 10
0680     ESCAPE TOP
0690   END-IF
0700 * ----------
0710 * EXIBE MAPA
0720 * ----------
0730   INPUT USING MAP 'LUCMGRAF'
0740 * ------------------
0750 * TRATA PF'S DO MAPA
0760 * ------------------
0770   IF *PF-KEY = 'ENTR' OR= 'PF3'
0780     ESCAPE BOTTOM
0790   END-IF
0800 END-REPEAT
0810 END
```

Mapa para exibição do Gráfico - **LUCMGRAF**

```
+XXXXXXXX                XXXXXXX Informática LTDA.                +MMMMMMMMMM
+XXXXXXXX                   ?Exemplo?de?Gráfico                     +XXXXXXXX
 ------------------------ Teste Gráfico em NATURAL --------------------------

  ?5000 |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?Legenda:
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX     -------
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?A - Pgm  1
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?B - Pgm  2
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?C - Pgm  3
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?D - Pgm  4
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?E - Pgm  5
  ?2500 |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?F - Pgm  6
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?G - Pgm  7
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?H - Pgm  8
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?I - Pgm  9
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?J - Pgm 10
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX
     ?1 |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX
         ____________________________________________________
          ?A   ?B   ?C   ?D   ?E   ?F   ?G   ?H   ?I   ?J
 ------------------------------------------------------------------------------
?F1 Ajuda ?F3 Sai ?F5 Encerra
```

```
Arr #GRAFICO                                                        Fmt A2
------------------------------------------------------------------------------
AD= OY________     ZP=         SG=        HE= ____________________   Rls 0
AL= _____          CD= __      CV= #CV-GRAFICO(*,*)_________________   Mod User
PM= __  DF=                    DY= @NE#RE$BL&YE*PI(TU=_____________
EM= ____________________________________________________

001   --010---+----+----+---030---+----+----+---050---+----+----+---070---+----
+XXXXXXXX                XXXXXXX Informática LTDA.                +MMMMMMMMMM
+XXXXXXXX                   ?Exemplo?de?Gráfico                     +XXXXXXXX
 ------------------------ Teste Gráfico em NATURAL --------------------------

  ?5000 |.EX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?Legenda:
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX     -------
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?A - Pgm  1
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?B - Pgm  2
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?C - Pgm  3
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?D - Pgm  4
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?E - Pgm  5
  ?2500 |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?F - Pgm  6
        |+XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX  +XX    ?G - Pgm  7
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      HELP  Mset  Exit  <---  --->  --    -     +           <     >     Let
```

21. BIBLIOGRAFIA

- Apostila de Natural Avançado – Politec
- http://www.consist.com
- Manual de Natural da Consist.

22. LICENÇA PÚBLICA GERAL GNU

22.1. Licença Pública Geral GNU (Tradução)

Esta é uma tradução não oficial da Licença de Documentação Livre GNU em Português Brasileiro. Ela não é publicada pela Free Software Foundation, e não se aplica legalmente a distribuição de textos que usem a GFDL - apenas o texto original em Inglês da GNU FDL faz isso. Entretanto, nós esperamos que esta tradução ajude falantes de português a entenderem melhor a GFDL.

Licença de Documentação Livre GNU

Versão 1.1, Março de 2000

0. INTRODUÇÃO

O propósito desta Licença é deixar um manual, livro-texto ou outro documento escrito “livre” no sentido de liberdade: assegurar a qualquer um a efetiva liberdade de copiar ou redistribuí-lo, com ou sem modificações, comercialmente ou não. Secundariamente, esta Licença mantém para o autor e editor uma forma de ter crédito por seu trabalho, sem ser considerado responsável pelas modificações feitas por terceiros.

Esta licença é um tipo de “copyleft” (“direitos revertidos”), o que significa que derivações do documento precisam ser livres no mesmo sentido. Ela complementa a GNU Licença Pública Geral (GNU GPL), que é um copyleft para software livre.

Nós fizemos esta Licença para que seja usada em manuais de software livre, porque software livre precisa de documentação livre: um programa livre deve ser acompanhado de manuais que forneçam as mesmas liberdades que o software possui.

Mas esta Licença não está restrita a manuais de software; ela pode ser usada para qualquer trabalho em texto, independentemente do assunto ou se ele é publicado como um livro impresso. Nós recomendamos esta Licença principalmente para trabalhos cujo propósito seja de instrução ou referência.

1. APLICABILIDADE E DEFINIÇÕES

Esta Licença se aplica a qualquer manual ou outro texto que contenha uma nota colocada pelo detentor dos direitos autorais dizendo que ele pode ser distribuído sob os termos desta Licença. O “Documento”, abaixo, se refere a qualquer tal manual ou texto. Qualquer pessoa do público é um licenciado e é referida como “você”.

Uma “Versão Modificada” do Documento se refere a qualquer trabalho contendo o documento ou uma parte dele, quer copiada exatamente, quer com modificações e/ou traduzida em outra língua.
Uma “Seção Secundária” é um apêndice ou uma seção inicial do Documento que trata exclusivamente da relação dos editores ou dos autores do Documento com o assunto geral do Documento (ou assuntos relacionados) e não contém nada que poderia ser incluído diretamente nesse assunto geral. (Por exemplo, se o Documento é em parte um livro texto de matemática, a Seção Secundária pode não explicar nada de

matemática). Essa relação poderia ser uma questão de ligação histórica com o assunto, ou matérias relacionadas, ou de posições legais, comerciais, filosóficas, éticas ou políticas relacionadas ao mesmo.
As “Seções Invariantes” são certas Seções Secundárias cujos títulos são designados, como sendo de Seções Invariantes, na nota que diz que o Documento é publicado sob esta Licença.

Os “Textos de Capa” são certos trechos curtos de texto que são listados, como Textos de Capa Frontal ou Textos da Quarta Capa, na nota que diz que o texto é publicado sob esta Licença.

Uma cópia “Transparente” do Documento significa uma cópia que pode ser lida automaticamente, representada num formato cuja especificação esteja disponível ao público geral, cujos conteúdos possam ser vistos e editados diretamente e sem mecanismos especiais com editores de texto genéricos ou (para imagens compostas de pixels) programas de pintura genéricos ou (para desenhos) por algum editor de desenhos grandemente difundido, e que seja passível de servir como entrada a formatadores de texto ou para tradução automática para uma variedade de formatos que sirvam de entrada para formatadores de texto. Uma cópia feita em um formato de arquivo outrossim Transparente cuja constituição tenha sido projetada para atrapalhar ou desencorajar modificações subseqüentes pelos leitores não é Transparente. Uma cópia que não é “Transparente” é chamada de “Opaca”.
Exemplos de formatos que podem ser usados para cópias Transparentes incluem ASCII simples sem marcações, formato de entrada do Texinfo, formato de entrada do LaTeX, SGML ou XML usando uma DTD disponibilizada publicamente, e HTML simples, compatível com os padrões, e projetado para ser modificado por pessoas. Formatos opacos incluem PostScript, PDF, formatos proprietários que podem ser lidos e editados apenas com processadores de texto proprietários, SGML ou XML para os quais a DTD e/ou ferramentas de processamento e edição não estejam disponíveis para o público, e HTML gerado automaticamente por alguns editores de texto com finalidade apenas de saída.

A “Página do Título” significa, para um livro impresso, a página do título propriamente dita, mais quaisquer páginas subseqüentes quantas forem necessárias para conter, de forma legível, o material que esta Licença requer que apareça na página do título. Para trabalhos que não tenham uma tal página do título, “Página do Título” significa o texto próximo da aparição mais proeminente do título do trabalho, precedendo o início do corpo do texto.

2. FAZENDO CÓPIAS EXATAS

Você pode copiar e distribuir o Documento em qualquer meio, de forma comercial ou não comercial, desde que esta Licença, as notas de copyright, e a nota de licença dizendo que esta Licença se aplica ao documento estejam reproduzidas em todas as cópias, e que você não acrescente nenhuma outra condição quaisquer que sejam às desta Licença.

Você não pode usar medidas técnicas para obstruir ou controlar a leitura ou confecção de cópias subseqüentes das cópias que você fizer ou distribuir. Entretanto, você pode aceitar compensação em troca de cópias. Se você distribuir uma quantidade grande o suficiente de cópias, você também precisa respeitar as condições da seção 3.

Você também pode emprestar cópias, sob as mesmas condições colocadas acima, e você também pode exibir cópias publicamente.

1. FAZENDO CÓPIAS EM QUANTIDADE

Se você publicar cópias do Documento em número maior que 100, e a nota de licença do Documento obrigar Textos de Capa, você precisa incluir as cópias em capas que tragam, clara e legivelmente, todos esses Textos de Capa: Textos de Capa da

Frente na capa da frente, e Textos da Quarta Capa na capa de trás. Ambas as capas também precisam identificar clara e legivelmente você como o editor dessas cópias. A capa da frente precisa apresentar o título completo com todas as palavras do título igualmente proeminentes e visíveis. Você pode adicionar outros materiais às capas. Fazer cópias com modificações limitadas às capas, tanto quanto estas preservem o título do documento e satisfaçam essas condições, pode tratado como cópia exata em outros aspectos.

Se os textos requeridos em qualquer das capas for muito volumoso para caber de forma legível, você deve colocar os primeiros (tantos quantos couberem de forma razoável) na capa verdadeira, e continuar os outros nas páginas adjacentes.

Se você publicar ou distribuir cópias Opacas do Documento em número maior que 100, você precisa ou incluir uma cópia Transparente que possa ser lida automaticamente com cada cópia Opaca, ou informar em ou com cada cópia Opaca a localização de uma cópia Transparente completa do Documento acessível publicamente em uma rede de computadores, à qual o público usuário de redes tenha acesso a download gratuito e anônimo utilizando padrões públicos de protocolos de rede. Se você utilizar o segundo método, você precisa tomar cuidados razoavelmente prudentes, quando iniciar a distribuição de cópias Opacas em quantidade, para assegurar que esta cópia Transparente vai permanecer acessível desta forma na localização especificada por pelo menos um ano depois da última vez em que você distribuir uma cópia Opaca (diretamente ou através de seus agentes ou distribuidores) daquela edição para o público.

É pedido, mas não é obrigatório, que você contate os autores do Documento bem antes de redistribuir qualquer grande número de cópias, para lhes dar uma oportunidade de prover você com uma versão atualizada do Documento.

4. MODIFICAÇÕES

Você pode copiar e distribuir uma Versão Modificada do Documento sob as condições das seções 2 e 3 acima, desde que você publique a Versão Modificada estritamente sob esta Licença, com a Versão Modificada tomando o papel do Documento, de forma a licenciar a distribuição e modificação da Versão Modificada para quem quer que possua uma cópia da mesma. Além disso, você precisa fazer o seguinte na versão modificada:

A. Usar na Página de Título (e nas capas, se alguma) um título distinto daquele do Documento, e daqueles de versões anteriores (que deveriam, se houvesse algum, estarem listados na seção Históricos do Documento). Você pode usar o mesmo título de uma versão anterior se o editor original daquela versão lhe der permissão.

B. Listar na Página de Título, como autores, uma ou mais das pessoas ou entidades responsáveis pela autoria das modificações na Versão Modificada, conjuntamente com pelo menos cinco dos autores principais do Documento (todos os seus autores principais, se ele tiver menos que cinco).

C. Colocar na Página de Título o nome do editor da Versão Modificada, como o editor.

D. Preservar todas as notas de copyright do Documento.

E. Adicionar uma nota de copyright apropriada para suas próprias modificações adjacente às outras notas de copyright.

F. Incluir, imediatamente depois das notas de copyright, uma nota de licença dando ao público o direito de usar a Versão Modificada sob os termos desta Licença, na forma mostrada no Adendo abaixo.

G. Preservar nessa nota de licença as listas completas das Seções Invariantes e os Textos de Capa requeridos dados na nota de licença do Documento.

H. Incluir uma cópia inalterada desta Licença.

I. Preservar a seção intitulada “Histórico”, e seu título, e adicionar à mesma um item dizendo pelo menos o título, ano, novos autores e editor da Versão Modificada como dados na Página de Título. Se não houver uma sessão denominada “Histórico”; no Documento, criar uma dizendo o título, ano, autores, e editor do Documento como dados em sua Página de Título, então adicionar um item descrevendo a Versão Modificada, tal como descrito na sentença anterior.

J. Preservar o endereço de rede, se algum, dado no Documento para acesso público a uma cópia Transparente do Documento, e da mesma forma, as localizações de rede dadas no Documento para as versões anteriores em que ele foi baseado. Elas podem ser colocadas na seção “Histórico”. Você pode omitir uma localização na rede para um trabalho que tenha sido publicado pelo menos quatro anos antes do Documento, ou se o editor original da versão a que ela ser refira der sua permissão.

K. Em qualquer seção intitulada “Agradecimentos”; ou “Dedicatórias”;, preservar o título da seção e preservar a seção em toda substância e tim de cada um dos agradecimentos de contribuidores e/ou dedicatórias dados.

L. Preservar todas as Seções Invariantes do Documento, inalteradas em seus textos ou em seus títulos. Números de seção ou equivalentes não são considerados parte dos títulos da seção.

M. Apagar qualquer seção intitulada “Endossos”;. Tal sessão não pode ser incluída na Versão Modificada.

N. Não re-intitular qualquer seção existente com o título “Endossos”; ou com qualquer outro título dado a uma Seção Invariante.

Se a Versão Modificada incluir novas seções iniciais ou apêndices que se qualifiquem como Seções Secundárias e não contenham nenhum material copiado do Documento, você pode optar por designar alguma ou todas aquelas seções como invariantes. Para fazer isso, adicione seus títulos à lista de Seções Invariantes na nota de licença da Versão Modificada. Esses títulos precisam ser diferentes de qualquer outro título de seção.

Você pode adicionar uma seção intitulada “Endossos”;, desde que ela não contenha qualquer coisa além de endossos da sua Versão Modificada por várias pessoas ou entidades - por exemplo, declarações de revisores ou de que o texto foi aprovado por uma organização como a definição oficial de um padrão.

Você pode adicionar uma passagem de até cinco palavras como um Texto de Capa da Frente , e uma passagem de até 25 palavras como um Texto de Quarta Capa, ao final da lista de Textos de Capa na Versão Modificada. Somente uma passagem de Texto da Capa da Frente e uma de Texto da Quarta Capa podem ser adicionados por (ou por acordos feitos por) qualquer entidade. Se o Documento já incluir um texto de capa para a mesma capa, adicionado previamente por você ou por acordo feito com alguma entidade para a qual você esteja agindo, você não pode adicionar um outro; mas você pode trocar o antigo, com permissão explícita do editor anterior que adicionou a passagem antiga.

O(s) autor(es) e editor(es) do Documento não dão permissão por esta Licença para que seus nomes sejam usados para publicidade ou para assegurar ou implicar endossamento de qualquer Versão Modificada.

5. COMBINANDO DOCUMENTOS

Você pode combinar o Documento com outros documentos publicados sob esta Licença, sob os termos definidos na seção 4 acima para versões modificadas, desde que você inclua na combinação todas as Seções Invariantes de todos os documentos originais, sem modificações, e liste todas elas como Seções Invariantes de seu trabalho combinado em sua nota de licença.

O trabalho combinado precisa conter apenas uma cópia desta Licença, e Seções Invariantes Idênticas com múltiplas ocorrências podem ser substituídas por apenas uma cópia.Se houver múltiplas Seções Invariantes com o mesmo nome mas com conteúdos distintos, faça o título de cada seção único adicionando ao final do mesmo, em parênteses, o nome do autor ou editor original daquela seção, se for conhecido, ou um número que seja único. Faça o mesmo ajuste nos títulos de seção na lista de Seções Invariantes nota de licença do trabalho combinado.
Na combinação, você precisa combinar quaisquer seções intituladas “Históricas”; dos diversos documentos originais, formando uma seção intitulada “Histórico”; da mesma forma combine quaisquer seções intituladas “Agradecimentos”, ou “Dedicatórias”. Você precisa apagar todas as seções intituladas como “Endosso”.

6. COLETÂNEAS DE DOCUMENTOS

Você pode fazer uma coletânea consistindo do Documento e outros documentos publicados sob esta Licença, e substituir as cópias individuais desta Licença nos vários documentos com uma única cópia incluída na coletânea, desde que você siga as regras desta Licença para cópia exata de cada um dos Documentos em todos os outros aspectos.

Você pode extrair um único documento de tal coletânea, e distribuí-lo individualmente sob esta Licença, desde que você insira uma cópia desta Licença no documento extraído, e siga esta Licença em todos os outros aspectos relacionados à cópia exata daquele documento.

7. AGREGAÇÃO COM TRABALHOS INDEPENDENTES

Uma compilação do Documento ou derivados dele com outros trabalhos ou documentos separados e independentes, em um volume ou mídia de distribuição, não conta como uma Versão Modificada do Documento, desde que não seja reclamado nenhum copyright de compilação seja reclamado pela compilação. Tal compilação é chamada um “agregado”, e esta Licença não se aplica aos outros trabalhos auto-contidos compilados junto com o Documento, só por conta de terem sido assim compilados, e eles não são trabalhos derivados do Documento.

Se o requerido para o Texto de Capa na seção 3 for aplicável a essas cópias do Documento, então, se o Documento constituir menos de um quarto de todo o agregado, os Textos de Capa do Documento podem ser colocados em capas adjacentes ao Documento dentro do agregado. Senão eles precisam aparecer nas capas de todo o agregado.

8. TRADUÇÃO

A tradução é considerada como um tipo de modificação, então você pode distribuir traduções do Documento sob os termos da seção 4. A substituição de Seções Invariantes por traduções requer uma permissão especial dos detentores do copyright das mesmas, mas você pode incluir traduções de algumas ou de todas as Seções Invariantes em adição as versões originais dessas Seções Invariantes. Você pode incluir uma tradução desta Licença desde que você também inclua a versão original em Inglês desta Licença. No caso de discordância entre a tradução e a versão original em Inglês desta Licença, a versão original em Inglês prevalecerá.

9. TÉRMINO

Você não pode copiar, modificar, sublicenciar, ou distribuir o Documento exceto como expressamente especificado sob esta Licença. Qualquer outra tentativa de copiar, modificar, sublicenciar, ou distribuir o Documento é nulo, e resultará automaticamente no término de seus direitos sob esta Licença. Entretanto, terceiros que tenham recebido cópias, ou direitos, de você sob esta Licença não terão suas licenças terminadas tanto quanto esses terceiros permaneçam em total acordo com esta Licença.

10. REVISÕES FUTURAS DESTA LICENÇA

A Free Software Foundation pode publicar novas versões revisadas da Licença de Documentação Livre GNU de tempos em tempos. Tais novas versões serão similares em espírito à versão presente, mas podem diferir em detalhes ao abordarem novos problemas e preocupações. Veja http://www.gnu.org/copyleft/.

A cada versão da Licença é dado um número de versão distinto. Se o Documento especificar que uma versão particular desta Licença "ou qualquer versão posterior" se aplica ao mesmo, você tem a opção de seguir os termos e condições daquela versão específica, ou de qualquer versão posterior que tenha sido publicada (não como rascunho) pela Free Software Foundation. Se o Documento não especificar um número de Versão desta Licença, você pode escolher qualquer versão já publicada (não como rascunho) pela Free Software Foundation.

ADENDO: Como usar esta Licença para seus documentos

Para usar esta Licença num documento que você escreveu, inclua uma cópia desta Licença no documento e ponha as seguintes notas de copyright e licenças logo apos a página de título:

> Copyright (c) ANO SEU NOME.
> É dada permissão para copiar, distribuir e/ou modificar este documento
> sob os termos da Licença de Documentação Livre GNU, Versão 1.1 ou qualquer
> versão posterior publicada pela Free Software Foundation;
> com as Seções Invariantes sendo LISTE SEUS TÍTULOS, com os Textos da
> Capa da Frente sendo LISTE, e com os Textos da Quarta-Capa sendo LISTE.
> Uma cópia da licença em está inclusa na seção entitulada
> "Licença de Documentação Livre GNU".

Se você não tiver nenhuma Seção Invariante, escreva "sem Seções Invariantes" ao invés de dizer quais são invariantes. Se você não tiver Textos de Capa da Frente, escreva "sem Textos de Capa da Frente" ao invés de "com os Textos da Capa da Frente sendo LISTE"; o mesmo para os Textos da Quarta Capa.

Se o seu documento contiver exemplos não triviais de código de programas, nós recomendamos a publicação desses exemplos em paralelo sob a sua escolha de licença de software livre, tal como a GNU General Public License, para permitir o seu uso em software livre.

22.2. GNU Free Documentation License (original em inglês)

GNU Free Documentation License
Version 1.2, November 2002

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant.

The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may
differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
> A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the
> Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

A tradução para o português encontra-se disponível em http://www.cipsga.org.br. A tradução não é oficial. Ela não é publicada pela Free Software Foundation, e não se aplica legalmente a distribuição de textos que usem a GFDL - apenas o texto original em Inglês da GNU FDL faz isso. Entretanto, nós esperamos que esta tradução ajude a entender melhor a GFDL.