

## Curso Básico de Natural

Joca Moraes  
Nov/2002



## Objetos Natural

Um objeto Natural nada mais é do que um componente de um sistema. Os diversos tipos de objetos Natural são:

COPYCODE - Pedaço de código fonte, que pode ser aproveitado em vários programas. Único objeto Natural que não é compilado. Nos programas onde o COPYCODE vai ser aproveitado, através do comando INCLUDE, ele será compilado juntamente com o programa.

GLOBAL - Objeto utilizado para definição de variáveis. Todos os objetos que se referenciarem a uma GLOBAL, terão as mesmas variáveis e, estas manterão os valores após o término de um objeto, permitindo que um objeto altere valores e outro reconheça as alterações.

LOCAL - Objeto utilizado para definição de variáveis. A diferença entre uma LOCAL e uma GLOBAL é que os valores das variáveis locais somente serão reconhecidos pelo objeto onde foram declaradas, deixando de existir quando do término do mesmo.

PARAMETER - Objeto utilizado para definição de variáveis que serão utilizadas entre programas e subprogramas.

MAP - Utilizado para entrada e exibição de dados. Permite a definição de Regras de Validação para os campos, inclusive com acesso à Base de Dados.

HELPROUTINE - Tipo de objeto que permite ser definido como ajuda a um campo de entrada de dados. Quando o cursor estiver posicionado num campo e for teclado "?", será chamada a HELPROUTINE que estiver associada ao campo.

PROGRAM - Principal objeto Natural. É a partir dele que são chamados os outros componentes.

SUBPROGRAM (N) - Objeto chamado por outro, com passagem de parâmetros que permite o retorno de dados ao objeto chamador.

SUBROUTINE - Uma subrotina é um pedaço de código geralmente executado mais de uma vez ou de diversas partes de um programa. Pode ser interno, definido dentro de um programa ou fora dele (externo). Quando escolhe-se a segunda opção, o tipo de objeto que conterá as subrotinas é o SUBROUTINE.

As letras sublinhadas nos nomes dos objetos podem ser utilizadas logo após o comando EDIT, evitando a digitação completa do tipo de objeto.

### Comandos Natural

CATALOG - Cataloga o objeto na área de trabalho. Este comando checa a sintaxe e grava a versão executável do objeto. Não grava o fonte. Pode ser substituído por apenas "cat".

CHECK - Checa a sintaxe do objeto atualmente na área de trabalho. Pode ser substituído por apenas "c".

CLEAR - Limpa a área de trabalho.

EDIT - Permite editar um objeto (programa, mapa, subrotina, etc). Pode-se informar o nome do objeto desejado, desde que ele exista. Para novos programas, por exemplo, basta "edit program" ou simplesmente "e p".

FIN - Termina a execução do Natural. Se não houver sido salvo, o objeto presente na Área de Trabalho será perdido.

HELP - Permite a exibição de mensagens de erro e comandos do Natural. Também pode ser acessado teclando "?". Ambas as opções permitem a passagem do código diretamente (? 1001, por exemplo, exibe o conteúdo do código de erro 1001).

LIST - Permite a listagem de um fonte especificado. Pode-se digitar "\*" para que sejam listados todos os nomes de todos os objetos. Também pode-se listar todos os objetos de determinado tipo, como "LIST P\*", por exemplo, para listar todos os programas.

LOGON - Permite selecionar uma biblioteca específica. O nome da biblioteca deve ser informado após o comando Logon. Se a biblioteca não existir, a mesma será criada. Caso não seja gravado nada, será apagada ao sair do Natural.

PURGE - Deleta o fonte do objeto.

QUIT - Finaliza o editor de programas. Pode ser substituído por "..".

READ - Lê e transfere para a área de trabalho o objeto definido após o comando.

RENUMBER - Renumara as linhas do programa.

RUN - Executa o programa atualmente na área de trabalho. O comando compila o programa antes de executá-lo. Pode ser substituído por apenas "r".

SAVE - Salva o objeto. Pode-se especificar o nome desejado logo após o comando. Pode se substituído por apenas "s".

SCRATCH - Apaga o fonte e o módulo objeto informado logo após o comando.

STOW - Checa a sintaxe do objeto na área de trabalho, salva o fonte e cataloga o objeto. Este comando executa internamente as funções dos comandos CHECK, SAVE e CATALOG. Porém, ele somente salva o objeto se este não contiver nenhum erro.

STRUCT - Estrutura o programa existente na área de trabalho.

UNCATALOG - Apaga apenas o módulo objeto, mantendo o módulo fonte.

## Editor de Programas

Após escolher a opção D no menu inicial, digitar EDIT PROGRAM (ou simplesmente EP) e será exibido o editor de programas. O editor de programas possui o seguinte layout:

```

> + Program
All .....+....1.....+....2.....+....3.....+....4.....+....5.....+....6.
0010
0020
0030 Linha de Comando
0040
0050
0060
0070
0080
0090
0100
0110
0120
0130
0140
0150
0160
0170
0180
0190
0200
.....+....1.....+....2.....+....3.....+....4.....+....5....S 0 L 1

```

Na linha de comando, além dos comandos do Natural já citados, podemos digitar os seguintes, exclusivos à área do editor de programas:

ADD - Adiciona linhas no final do fonte.

BOTTOM - Pode ser substituído por B. Posiciona a janela de edição no final do fonte sendo editado.

CHANGE - São passados dois parâmetros: o valor a trocar e o novo valor. O comando procura a primeira ocorrência do valor e efetua a troca. Caracteres especiais devem ser informados entre aspas simples (''). Na linha em que tiver sido efetuada a troca, na área destinada à numeração das linhas, aparecerá a informação da substituição.

Globals SM (ON / OFF) - Muda o modo de programação entre estruturado e report.

LET - Ignora todas as alterações feitas no fonte após o último <ENTER>.

PROFILE - Exibe e permite alterar as informações de edição da seção atual.

RESET - Limpa as marcas de bloco e qualquer outra informação presente na área de numeração das linhas.

SCAN - Procura a string no fonte. Pode ser substituído por SC.

SPLIT <tipo de objeto> <nome do objeto> - Divide a tela em duas partes: na superior, mantém o editor ativo; na inferior, apresenta apenas para visualização o objeto solicitado.

TOP - Pode ser substituído por T. Posiciona a janela da edição na primeira linha do fonte sendo editado.

A seguir, a relação dos principais comandos de movimentação de texto do editor de programas. Estes comandos devem ser digitados na primeira posição da linha:

.C - Copy. Copia a linha atual, duplicando-a. Se especificado um número entre parênteses, duplica a linha tantas vezes especificado.

.CX-Y - Copia o bloco marcado com .X e .Y.

.D - Delete. Apaga a linha corrente. Pode-se especificar um número de linhas a apagar entre parênteses.

.I - Insert. Insere linhas após a linha corrente. Pode ser informado um número, entre parênteses, que representa a quantidade de linhas que serão inseridas. Se informado .I(objeto), o fonte do objeto será inserido no fonte atual.

.J - Join. Concatena a linha corrente com a linha seguinte, desde que o tamanho da linha resultante seja suportado pelo editor (127 caracteres por linha).

.L - Ignora as mudanças da linha, desde que não pressionado <ENTER>.

.MX-Y - Move o bloco marcado para a linha seguinte à atual.

.P - Posiciona a linha como sendo a primeira da tela.

.S - Split. Divide a linha corrente, quebrando-a em duas. Após colocar .S no início da linha, colocar o cursor sob o caracter que deverá ser o primeiro da nova linha e teclar <ENTER>.

.X e .Y - Marcam o bloco.

Após a marcação de um bloco, aparecem as marcas correspondentes (X e Y) na primeira coluna do fonte. Quando definimos o .X e o .Y para a mesma linha, na primeira coluna é exibido Z.

Quando existir um bloco marcado, podem ser digitados os comandos DX-Y na linha de comandos para apagar o bloco e EX-Y para editar apenas o bloco, ou seja, apaga o resto do fonte.

As teclas de função podem ser configuradas pelo usuário, através do comando PROFILE, para executar diversas funções.

### Editor de Mapas

Após escolher a opção D no menu inicial, digitar EDIT MAP (ou simplesmente E M) e será exibido o submenu do editor de mapas:

```
10:29:42      ***** NATURAL MAP EDITOR *****          30/10/2002
User X033501           - 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 .. I      Name .. _____      Profile .. SYSPROF_
```

Command ==>

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

O Natural pressupõe a criação de um mapa novo; por esse motivo, o CODE já vem preenchido com I. Se formos editar, testar, etc. um mapa, devemos trocar o código.

Opções:

D - Exibe todos os campos definidos no mapa, com respectiva posição, formato, nome e outras, carregado.

E - Edita um mapa já existente. Se informado EDIT MAP <NOME>, o mapa já será carregado.

I - Inicializa um novo mapa. Existe uma tela de parâmetros para construção de um novo mapa. No caso de edição de um mapa já existente, a tela de parametrização não é exibida automaticamente.

H - Inicializa um novo mapa de Help.

M - Manutenção de profiles. Profiles são mapas que podem ser usados com modelo de novos mapas.

S - Salva o mapa que estiver na área de trabalho.

T - Testa o mapa. Permite visualizar o mapa e simular seu preenchimento.

W - Gera o módulo objeto e salva o módulo fonte do mapa (STOW).

? - Help

. - Sai do submenu de mapas.

Se formos inicializar um novo mapa, será exibida a seguinte tela:

Define Map Settings for MAP				30/10/2002
Delimiters		Format	Context	
Cls	Att	CD	Del	Page Size .... 23
T	D		BLANK	Device Check ....
T	I		?	Line Size .... 79
A	D		-	WRITE Statement
A	I		)	Column Shift...0 (0/1)
A	N		-	INPUT Statement
M	D		&	X
M	I		:	Layout .....
O	D		+	Help
O	I		(	dynamic ... N (Y/N)
				as field default N (Y/N)
				Zero Print .. N (Y/N)
				Case Default .UC (UC/LC)
				Manual Skip ..N (Y/N)
				Automatic Rule Rank 1
				Decimal Char.,
				Profile Name ... SYSPROF
				Standard Keys.N (Y/N)
				Justification.L (L/R)
				Filler Characters
				Print Mode ...
				-----
				Optional, Partial ....
				Required, Partial ....
				Optional, Complete ...
				Required, Complete ...
ControlVar... _____				
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10-----				
Help		Exit		

Delimiters:

Os delimitadores servem para informar ao Natural qual tipo de campo e seus atributos serão definidos a seguir.

Tipo de Campos	
T	Texto constante
A	Campo somente de entrada
M	Campo modificável. Serve para entrada e saída de dados
O	Campo somente de saída

Atributos de Campos			
B	Biking (piscante)	V	Modo de vídeo reverso
C	Cursivo / Itálico	U	Sublinhado(Underlined)
D	Default (cor normal)	Y	Dinâmico
I	Indentificado (high light)	N	Não exibido

Caracteres Default de Prefixo de Campos		
Campo	Atributo	Caracter Prefixo
T	Default	Espaço em branco
T	Intensificado	?
A	Default	-
A	Intensificado	)
A	Não exibido	-
M	Default	&
M	Intensificado	:
O	Default	+
O	Intensificado	(

Format:

Especifica características de entrada/saída do mapa.

Page Size	Número de linhas do mapa. Pode variar de 1 a 250.
Line Size	Número de linhas do mapa. Pode variar de 5 a 249.
Column Shift	Se o conteúdo for 1, o mapa é deslocado uma coluna para a esquerda em tempo de execução.
Layout	Nome do mapa que contém o layout pré-definido para o mapa.
Dynamic	Se especificado Y, os mapas são combinados em tempo de execução. Senão, são combinados durante a inicialização.
Zero Print	Se especificado Y, campos zerados serão exibidos como 0. Senão, campos zerados serão mostrados como espaços.
Case Default	UC indica que todos os campos terão as minúsculas transformadas em maiúsculas; LC, as minúsculas permanecem minúsculas.
Manual Skip	Se especificado Y, o cursor não passará para o próximo campo quando ele estiver todo preenchido. Com N, o cursor passará para o campo seguinte assim que o campo atual estiver completo.
Decimal Character	Caracter a ser utilizado como separador das casas decimais.
Standard Keys	Se informado Y, nas duas últimas linhas serão exibidas as teclas de função.
Justification	Se informado L, todos os campos serão justificados à esquerda (LEFT). Se informado R, à direira (RIGHT).
Print Mode	Modo de impressão do campo I - Invertido C - Utiliza caracteres alternativos N - Campo não será impresso D - Modo de caracter duplo-byte
Control Variable	Variável de controle para todos os campos que sejam controlados dinamicamente e que não tenham variável de controle definida.

Context:

Especifica o contexto do mapa.

Device Check	Se um nome de um dispositivo é informado neste campo, todos os atributos do mapa serão checados para verificar compatibilidade com o dispositivo.
Write Statement	Marcando este campo, o mapa será usado apenas para saída de dados.
Input Statement	Marcando este campo, o mapa será usado para entrada/saída de dados.
Help	Nome da helproutine associada ao mapa.
As field default	Se especificado Y, o nome do campo onde for pressionada a tecla de help será passado como parâmetro default para o helproutine.
Automatic Rule Rank	Define o número default de uma regra de validação automática.
Profile Name	Nome da profile ativa para inicialização do mapa.

Filler Characters:

Especifica caracteres de preenchimento para os campos.

Optional, Partial	Opcional, não precisa ser preenchido completamente
Required, Partial	Preenchimento requerido, não precisa ser preenchido completamente.
Optional, Complete	Opcional, deve ser preenchido completamente.
Required, Complete	Preenchimento requerido, deve ser preenchido completamente.

Após a alteração dos parâmetros desejados, aparece a tela de edição de mapas:

ob

A large circle with a black outline, centered on the page. Inside the circle, the number "1" is written in a black serif font.

Ob	D	CLS	ATT	DEL		CLS	ATT	DEL
.		T	D	Blnk		T	I	?
.		A	D	-		A	I	)
.		A	N	¬		M	D	&
.		M	I	:		O	D	+
.		O	I	(				

001--010---+----+----+----030---+----+----+----050---+----+----+----070-

3

Enter-PF1--PF2--PF3--PF4--PF5--PF6--PF7--PF8--PF9--PF10-PF11--P  
Help Mset Exit Test Edit -- - + Full < > L

## Áreas do Editor:

1 - Área de split. Permite visualizar e inserir variáveis de views e de outros objetos.

2 - Área de prefixos. Mostra quais são os prefixos associados a cada tipo de campo.

3 - Área de edição. Permite definir o layout do mapa.

Para elaborarmos um mapa, basta digitarmos os campos desejados, precedidos ou não do prefixo correspondente, de acordo com critérios pré-estabelecidos.

Para criarmos um campo alfanumérico modificável, por exemplo, basta digitarmos o código &, seguido do número de X correspondentes ao desejado. Outra forma é digitar &X (gtde).

Para criarmos um campo numérico, digitar o código, seguido de 9.  
Exemplo: &9999.

A entrada de campos do tipo data não é possível diretamente da tela, sendo necessário digitarmos a data num campo alga ou numérico e depois movermos para um campo DATE.

?Administracao?de?Recursos?Humanos

?Matricula : . . . . : &999999999

?Nome : . . . . . : &xxxxxxxxxxxxxxxxxxxxxxxxxxxx

?Ultimos Salários : &9999999,99

Enter-PF1--PF2--PF3--PF4--PF5--PF6--PF7--PF8--PF9--PF10-PF11--P  
Help Mset Exit Test Edit -- - + Full < > L

Teclas de função na tela de edição d mapas:

<b>Teclas de função</b>			
PF1	Invoca o help do editor de mapa	PF7	Move metade da página para baixo
PF2	Permite alterar definições do mapa	PF8	Move metade da página para cima
PF3	Retorna ao menu do editor	PF9	Permite edição em tela cheia
PF4	Testa o mapa corrente	PF10	Move janela para a esquerda
PF5	Edita o campo sob o cursos	PF11	Move janela para a direita
PF6	Topo - move para o início do mapa	PF12	Ignora as mudanças após o ENTER

Para colocarmos um campo diretamente de uma view ou de um programa, basta, na área de split, digitar V e o nome da view para exibir os respectivos campos, ou P e o nome do programa.

Os caracteres + e - permitem movimentar a tela de split para cima ou para baixo.

Para inserirmos o campo desejado, colocar na tela o prefixo desejado e o número do campo que aparece na tela de split. Para desativar, colocar espaço em branco.

### Comandos de Linha

Os comandos de linha devem ser digitados no início da linha. Alguns comandos de linha não podem ser digitados ao mesmo tempo, pois o Natural não interpreta todos os comandos ao mesmo tempo.

..C - Centraliza a linha corrente. Deve ter sido dado <ENTER> previamente na linha a centralizar.

..D - Deleta a linha.

..D(n) - Apaga as "n" linhas, contando com a linha atual.

..E - Permite a edição de campos em tela cheia.

..Fc - Preenche a linha inteira com o caracter "c".

..I - Insere uma linha.

..I(n) - Insere "n" linhas.

..J - Join. Concatena a linha corrente com a linha de baixo.

..M - Move a linha. Colocar o cursor na linha de destino e teclar <ENTER>.

..P - Abre o editor de regras de validação do mapa.

..Q - Sair do editor de mapas.

..R - Duplica a linha.

..S - Split. Permite dividir a linha. Posicionar o cursor no ponto de quebra e teclar <ENTER>.

## Comandos de campo

Os comandos de campo devem ser digitados em cima do prefixo do campo.

.A - Definição de array (matriz). Permite definir o número de ocorrências, matrizes bidimensionais e qual elemento deverá ser o primeiro a ser apresentado.

.C - Centraliza o campo, levando em consideração os campos adjacentes.

.D - Deleta o campo.

.E - Permite editar o campo no modo estendido.

.M - Move o campo. Posicionar o cursor no novo lugar e teclar <ENTER>

.R - Repete o campo marcado na posição atual do cursor.

.P - Permite editar as regras de validação do campo.

Exemplo de edição extendida de campo.

Fld#001 Fmt N11  
-----  
AD= MDLT \_\_\_\_\_ ZP= OFF SG= OFF HE= \_\_\_\_\_ Rls 0  
NL= \_\_\_\_\_ CD= \_\_\_ CV= \_\_\_\_\_ Mod Undef  
PM= \_\_\_ DY= \_\_\_\_\_  
FM= \_\_\_\_\_

001 010---+----+----+---030---+----+----+---050---+----+----+---070---

?Administracao?de?Recursos?Humanos

?Matricula .....: E9999999999

?Nome ..... :&XXXXXXXXXXXXXXXXXXXXXX

?Ultimos Salarios :&999999,99

Enter-PF1--PF2--PF3--PF4--PF5--PF6--PF7--PF8--PF9--PF10-PF11--P  
Help Mset Exit < -> -- - + < >

Quando da criação do campo, o Natural estabelece #001 para o primeiro campo e vai nomeando sucessivamente os campos à medida em que são criados, seqüencialmente. Para alterar o nome do campo, digitar ".E" no campo.

Outros campos na tela de edição extendida do campo:

Fmt: Formato. Permite alterar o formato (N, A, P, D, etc.)

AD: Definição de atributos do campo. Digitar o atributo diretamente no campo ou teclar "?" para preencher as opções disponíveis.

ZP: Zero print. Disponível apenas se for campo numérico. Se informado Y, imprime 0 quando for valor zerado. Se informado N, o campo é mostrado como espaços em branco.

SG: Signal. Especifica se o sinal do campo será mostrado ou não. Disponível apenas se for campo numérico.

HE: Helproutine. Especifica uma Helproutine a ser executado caso seja pressionada a tecla de HELP no campo. Outra forma de ativar o Help é digitar "?" na primeira posição do campo.

NL / AL: Numeric Length ou Alphabetic Length. Especifica o número de caracteres a serem exibidos. O campo pode ser A9, por exemplo, mas ser necessário mostrar apenas as 4 primeiras: AL = 4.

CD: Color Atributes. Permite atribuir uma cor específica para o campo.

CV: Control Variable. Variável de controle. Estabelece a variável de controle (tipo C) do mapa.

PM: Print Mode. I - Invertido, C - conjunto de caracteres alternativo, N - sem impressão.

DY - Atributos dinâmicos. Permite definir atributos para alterar parte do campo.

EM: Edit Mask. Permite especificar uma máscara para o campo.

Máscara para números:

9 - apresenta o número, mesmo que seja zero.

Z - suprime os zeros não significativos. Ex.: ZZZ.ZZ9,99

S - sinal do campo. Ex.: ZZ9,99S

Máscara para campos data:

DD - dia

MM - mês

YYYY - ano

Ex.: DD/MM/YYYY

Máscara para campos hora:

HH - hora

II - minutos

SS - segundos

Ex.: HH:II:SS

Máscara para campos alfa:

X - especifica um caracter alfa. Ex.: XXX-XX

Se for digitado .A num campo, será aberta a tela de edição de array:

Name #003	Upper Bnds	1	1	1
-----				
Dimensions	Occurrences	Starting from	Spacing	
0 . Index vertical	1__	_____	0 Lines	
0 . Index horizontal	1__	_____	1 Columns	
0 . Index (h/v) V	1__	_____	0 Cls/Ls	
001	--010-----+----+---+030-----+----+----+050-----+--			
?Administração?de?Recursos?Humanos				
?Matrícula .....: E999999999				
?Nome .....:&XXXXXXXXXXXXXXXXXXXXXX				
?Últimos Salários :.A999999,99				
Enter-PF1--PF2--PF3--PF4--PF5--PF6--PF7--PF8--PF9--PF10-PF11--P				
Help Mset Exit -- - + < >				

Upper Boundaries: estabelece o tamanho máximo de ocorrências do array.

Occurrences: a linha que for preenchida nesta coluna. Estabelece o número de ocorrências verticais e/ou horizontais do array.

Starting from: permite estabelecer uma variável para controlar qual o número do primeiro elemento da tela em relação ao array definido no programa.

Por exemplo, podemos definir um array de 100 ocorrências internamente e 10 ocorrências sendo exibidas na tela. Definimos então uma variável que terá o número do primeiro elemento. Se a variável tiver o conteúdo 70, serão mostradas as ocorrências de 70 a 80. É útil para controle de paginações.

## Comandos

### ➤ DEFINE DATA

Para a definição de variáveis, utilizamos a instrução DEFINE DATA. Deve ser o primeiro comando de um programa. Somente pode exibir uma DEFINE DATA por objeto. As definições podem ser internas ou externas.

Definições internas:

```
INDEPENDENT  
LOCAL  
PARAMETER
```

Definições externas:

```
GLOBAL USING GDA  
LOCAL USING LDA  
LOCAL USING PDA
```

Quando múltiplas áreas de dados são utilizadas pelo mesmo programa, elas devem ser codificadas na seguinte ordem:

- GLOBAL
- PARAMETER
- LOCAL
- INDEPENDENT

Após a definição da última variável, fechar o laço com END-DEFINE.  
Exemplo:

```
DEFINE DATA LOCAL  
1 #AREA      (A20)  
1 #TELEFONE  
    2 #DDD      (N3)  
    2 #NUMERO   (N7)  
1 #VALOR     (P11,2)  
1 #TAB_MÊS   (A3/12)  
END-DEFINE
```

A estrutura para definição é composta de:

- Nível
- Nome
- Formato/tamanho
- Complemento

Podemos definir 7 níveis de variáveis. O tamanho máximo para um nome de variável é 32 caracteres. Os caracteres válidos para formação do nome são os alfabéticos (A-Z), números (0-9) e os especiais (#, -, \_, /, @, \$, &). O carácter + também pode ser utilizado, desde que apenas na primeira posição. Para efeito de padronização, utilizamos o carácter # como carácter inicial para designar variáveis definidas pelo usuário.

Formato	Código	Tamanho Máximo	Observação
Alfanumérico	A	253	
Numérico	N	27	Máximo de 7 decimais
Compactado	P	27	Máximo de 7 decimais
Inteiro	I	1, 2 ou 4	I1 - valores de 1 até 127 I2 - valores de 1 até 32767
Data	D		
Hora	T		
Lógico	L		TRUE ou FALSE
Binário	B	126	
Controle	C		Atributo controle de mapas
Ponto flutuante	F	4.8	

## Matrizes

JAN	FEV	MAR	ABR	MAI	...	DEZ
-----	-----	-----	-----	-----	-----	-----

Para definir a ocorrência de múltiplos elementos, basta colocar uma barra, seguida do número de vezes que o campo deve ser repetido. Opcionalmente, podemos definir 1:n como quantidade. Variáveis de qualquer tipo pode ser transformadas em matriz.

```
1 #MESES-ANO      (A3/12)
1 #DIAS-SEMANA    (A7/1:7)
```

## Redefinição

Podemos redefinir um campo, ou parte dele, em campos menores, de acordo com a necessidade. Os campos de menor nível podem ser de formato diferente, desde que tenham o mesmo tamanho.

TELEFONE	
DDD	NÚMERO

```
1 #TELEFONE      (N10)
1 REDEFINE #TELEFONE
    2 #DDD        (N3)
    2 #NUMERO     (N7)
```

#### Valores iniciais

A atribuição de valores iniciais pode ser feita quando da definição da variável. Durante a execução do programa, pode-se atribuir novos valores às mesmas. Quando a variável for alfanumérica, o valor inicial deve vir entre aspas simples.

```
1 #CABECALHO   (A7) INIT <'POLITEC'>
1 #CGC          (N14) INIT <42278796000199>
1 #FIM-ARQUIVO (L) INIT <FALSE>
```

Não se pode atribuir valores a uma variável que é redefinição de outra, mas apenas à variável de nível acima dela.

```
1 #CGC          (N14) INIT <42278796000199>
1 REDEFINE #CGC
    2 #CGC-BASE    (N8)
    2 #CGC-FILIAL  (N4)
    2 #CGC-CONTROLE (N2)
```

#### Registros

Um registro é uma coleção de variáveis, com características em comum. O Natural permite a definição de registros, bastando para isso criar variáveis com níveis diferentes.

```
1 #FICHA-REGISTRO
    2 #NOME (A36)
    2 #ENDERECO
        3 #LOGRADOURO (A35)
        3 #NUMERO      (N5)
    2 #BAIRRO   (A20)
    2 #CIDADE   (A20)
    2 #SALARIO  (P13,2)
```

## Views

A declaração de uma view (visão) é uma ligação entre o Natural e o arquivo correspondente na base de dados. Não é necessário definir formato/tamanho dos campos da view, pois eles irão possuir as mesmas características de seus correspondentes no banco de dados.

```
1 ARH215 VIEW OF ARH215-CADASTRO-BASICO
  2 MATRICULA-215
  2 NOME-215
  2 DATA-POSSE-215
```

## ➤ RESET

Comando utilizado para anular ou inicializar o valor de variáveis.

*Formato:*

RESET (INITIAL) operando-1 .... operando-n
--

Restabelece o valor nulo, de acordo com o tipo da variável:

DEFAULT DE INICIALIZAÇÃO		
<b>A</b>	Alfanumérico	Branco
<b>B</b>	Binário	0
<b>N</b>	Numérico	0
<b>P</b>	Numérico compactado	0
<b>I</b>	Inteiro	0
<b>L</b>	Lógico	False
<b>F</b>	Ponto flutuante	0
<b>D</b>	Data	0
<b>T</b>	Hora	00:00:00

A cláusula INITIAL restabelece o valor original definido na DEFINE DATA pela cláusula INIT.

```
DEFINE DATA LOCAL
1 #ALFA          (A10)
1 #PERCENTUAL   (N3,2)
1 #MAT-MÊS       (N2/12)
1 #REGISTRO
    2 #NOME        (A36)
    2 #MATRICULA   (N9)
1 #MAX-REG       (P4)  INIT <100>
END-DEFINE
...
```

```
RESET #ALFA  #PERCENTUAL  #REGISTRO  #MAT-MÊS (*)  #MAX-REG  
...  
RESET INITIAL  #MAX-REG  
...
```

➤ SET KEY

Comando utilizado para ativar/desativar as teclas de função

*Formato:*

```
SET KEY = { ON / OFF / ALL}  
SET KEY PFn = {programa / ON / OFF / HELP} [NAMED {OFF / operando}]
```

SET KEY ALL - Ativa todas as PF's para uso  
SET KEY PF2 = PGM - Ativa PF2 para executar o programa especificado  
SET KEY OFF - Desativa todas as PF's  
SET KEY ON - Reativa todas as PF's  
SET KEY PF4 = HELP - Ativa a tecla PF4 para servir de HELP de campo  
SET KEY PF7 = 'PRGP0045' NAMED 'PEND' - Ativa tecla PF7 para executar  
o programa PRGP0045 e coloca  
o string PEND na régua de  
PF's.

```
...  
END-DEFINE  
SET KEY ALL  
SET KEY PF4 = HELP  
    PF6 = 'PRGP0010'  
    PF8 = 'PRGP0045' NAMED 'MENU'  
...
```

➤ SET GLOBALS

Permite especificar parâmetros de entrada/saída. Funciona somente no modo REPORT.

*Formato:*

```
SET GLOBALS parâmetros ...
```

Alguns parâmetros que podem ser setados:

Parâmetros		
EJ	PAGE EJECT	EJ=ON - a página será ejetada EJ=OFF - a página não será ejetada
EM	EDIT MASK	9 - para campos numéricos Z - não mostra zeros à esquerda X - valores alfanuméricicos
FS	FORMAT SPECIFICATION	FS=ON - exige definição do formato para as variáveis FS=OFF - atribui automaticamente o formato às variáveis
IC	INSERTION CHARACTERS	Insere um caracter à esquerda do valor
IS	IDENTICAL SUPPRESS	Suprime valores já impressos na linha anterior
LS	LINE SIZE	Número de caracteres da linha
MP	MAXIMUM NUMBER PAGES	Número máximo de páginas a serem impressas
OS	PAGE SIZE	Número de linhas da página
SF	SPACING FACTOR	Especifica o número de caracteres que separam as colunas de um DISPLAY
TC	TRAILING CHARACTERS	Insere um caracter à direita do valor
ZD	ZERO DIVISION CHECK	ZD=ON - retorna mensagem de erro caso haja divisão por zero ZD=OFF - retorna zero no caso de uma divisão por zero
ZP	ZERO PRINTING	ZP=ON - se o campo for zerado, será impresso 0 ZP=OFF - se o campo for zerado, será impresso brancos

...  
SET GLOBALS LS=80 OS=66

...

➤ INPUT

Instrução para **obter** dados para o programa ou para **mostrar** os dados de programa.

Formato:

```
INPUT [(parâmetros)] [MARK *operando]
```

onde **X** significa espaços em branco, **T** posição de tabulação, **x/y** linha e coluna de tela.

Exemplos:

```
INPUT #VAR-ALFA    #VAR-NUM(SG=OFF)  
INPUT '          ' #AGENCIA  
INPUT     //// 10T 'AGENCIA....: ' #AGENCIA
```

Parâmetros

AD = [r][a][i/o][c][f]  
r - apresentação: Blink, Default, Intensified, Non-display  
a - alinhamento: Left, Right  
i/o - modo: Input (A), Modifiable, Output  
c - tratamento de caracteres: T minúsculas para maiúsculas,  
 W aceita minúsculas  
f - caracter de preenchimento (filler): definido pelo usuário

```
INPUT (AD=ILMT '..')     //// 10t 'AGENCIA....:' #AGENCIA
```

```
INPUT /// 'AGENCIA' (I) #AGENCIA (AD=M)
```

INPUT com campo não escoando na tela:

---

INPUT com campo de entrada alfanumérico, brilho, caracter de preenchimento '.', e ajuste à direita:

---

```
INPUT (AD=M) ' ' #AGENCIA ' ' #CONTA
```

```
INPUT MARK *#CONTA /// #AGENCIA #CONTA
```

```
INPUT MARK 2 /// #AGENCIA #CONTA
```

INOUT com mensagem na linha de mensagem:

```
INPUT WITH TEXT 'Digite a agência.' 'AGENCIA:'  
#AGENCIA (AD=MI'_')
```

INPUT com utilização de mapa externo:

```
INPUT USING MAP 'FDEM001'
```

Apresentando mensagem na linha de mensagem:

```
INPUT WITH TEXT #MSG USING MAP 'GRHM0014'
```

Marcando um determinado campo:

```
INPUT WITH TEXT #MSG MARK #IDC-MARK USING MAP 'TTEM5400'
```

#### ➤ REINPUT

Instrução para voltar a executar a última instrução INPUT

Formato:

```
REINPUT [FULL] operando1 MARK *operando2 [ALARM]
```

Operando1: variável ou constante com a mensagem a ser apresentada no REINPUT.

Operando2: campo para posicionamento do cursor.

Entre uma instrução INPUT e o REINPUT correspondente, não pode haver nenhuma instrução WRITE ou DISPLAY, pois para a execução do REINPUT, a última comunicação com a tela tem obrigatoriamente que ter sido com um INPUT.

A cláusula FULL altera todos os campos da tela, até onde encontra-se o campo a redigitar. Se não for especificada, não atualiza os campos.

```
DEFINE DATA LOCAL  
1 #USER      (A8)  
1 #PASSW     (A8)  
END-DEFINE  
INPUT (AD=M) WITH TEXT 'Preencher os campos.' ////  
10T 'Usuário :' (I) #USER /
```

```
10T 'Password :' (I) #passw (AD=N) /
IF #USER = ''
    REINPUT 'Informe código do usuário.' MARK *#USER ALARM
END-IF
IF #PASSW = ''
    REINPUT 'Informe a password.' MARK *#PASSW ALARM
END-IF
END
```

Utilizando um mapa para a entrada de dados, o programa anterior ficaria da seguinte maneira:

```
DEFINE DATA LOCAL
1 #USER (A8)
1 #PASSW (A8)
END-DEFINE
INPUT WITH TEXT 'Preencher os campos.' USING MAP 'SENHA'
IF #USER = ''
    REINPUT 'Informe código de usuário.' MARK *#USER ALARM
END-IF
IF #PASSW = ''
    REINPUT 'Informe a password.' MARK *#PASSW ALARM
END-IF
END
```

➤ WRITE

Instrução para imprimir dados do programa, em formato de linhas

*Formato:*

```
WRITE [(rel)] [NOTITLE] [{parâmetros}]
[{nX}] [{'='}]
[{NT}] [{`'texto`}]} operando [(parâmetros)]
[{'c' n}]
```

Opções:

NOTITLE: suprime a linha de cabeçalho da página.

Rel: número do relatório. O Natural pode controlar até 32 relatórios simultâneos, numerados de 0 a 31.

Parâmetros:

ES (Empty Line Supress): Supressão de linhas vazias. ON/OFF

IS (Identical Supress): Suprime a impressão do campo na linha se este tiver o mesmo conteúdo da linha anterior. ON/OFF

ZP (Zero Printing): Imprime ou não campos numéricos zerados. ON/OFF

```
WRITE (ES=ON)
```

```
WRITE 'ORGAO' SIGLA_ORGAO
```

```
WRITE 'MATRICULA/COLABORADOR' NUM_MATRIC_EMPREG
```

Parâmetros de variáveis:

AL (Alphanumeric Length): Tamanho máximo para exibição de campo alfanumérico.

NL (Numeric Length): Tamanho máximo para exibição de campo numérico.

```
WRITE NUM_MATRIC_EMPREG NOME_EMPREGADO (AL=20)
```

```
WRITE '=' NUM_MATRIC_EMPREG
```

```
WRITE NUM_MATRIC_EMPREG / NOME_EMPREGADO
```

Vejamos o seguinte programa como exemplo de utilização WRITE:

```
DEFINE DATA LOCAL
1 EMP VIEW OF AD-EMPREGADO-INTERNET
  2 NUM_MATRIC_EMPREG          /* A6
  2 NOME_EMPREGADO            /* A36
  2 SIGLA_ORGAO                /* A10
  2 DATA_ADMISS_EMPREG        /* A10
END-DEFINE

*
WRITE '-----1-----2-----3-----4-----5---'
*

SELECT NUM_MATRIC_EMPREG, NOME_EMPREGADO, SIGLA_ORGAO,
       DATA_ADMISS_EMPREG
INTO   VIEW EMP
FROM   AD-EMPREGADO-INTERNET
WHERE  SIGLA_ORGAO = 'DIVISA'
END-SELECT
```

A execução gera o seguinte resultado:

Page	1	
-----1-----2-----3-----4-----5---		
046115	GERALDO GUILHERME CARVALHO CALDEIRA	DIVISA
046479	CLARINO DE ABREU FILHO	DIVISA
064801	ROMAN DARIO CUATTRIN	DIVISA
065113	PAULO EMILIO HARTUNG	DIVISA
067127	ANAMARIA LOPES FERREIRA .	DIVISA

➤ REPEAT

Instrução para iniciar um processamento de loop. Se não for especificada condição de finalização, o loop será eterno. A codificação de cláusula de término pode ser no início ou no final do laço.

*Formatos:*

```
REPEAT
    Instruções
    [UNTIL] condição lógica
    [WHILE]
END-REPEAT
```

```
REPEAT
    [UNTIL] condição lógica
    [WHILE]
    Instruções
END-REPEAT
```

UNTIL - o laço será executado até a condição lógica ser satisfeita.  
WHILE - o laço será executado enquanto a condição lógica for verdadeira.

É permitido sair do laço com o comando ESCAPE BOTTOM ou com o comando de término do programa (STOP) ou do próprio Natural (TERMINATE). O ESCAPE BOTTOM entrega o controle do programa à próxima instrução após o END-REPEAT.

O comando ESCAPE TOP permite uma nova execução do laço, antes dele atingir o comando END-REPEAT. A execução de um ESCAPE TOP força novo teste lógico, desde que a cláusula WHILE/UNTIL esteja no início do laço.

```
...
REPEAT
    INPUT USING MAP 'TESTE'
    IF #MATRICULA = 0
        ESCAPE BOTTOM
    END-IF

SELECT NOME_EMPREGADO
    INTO VIEW EMP
    FROM AD-EMPREGADO-INTERNET
    WHERE MATRICULA = #MATRICULA
END-SELECT
END-REPEAT
...

REPEAT UNTIL #OPCAO = 9
    PERFORM EXIBE-TELA-INICIAL
END-REPEAT

REPEAT
    PERFORM TRATA-PROCESSO
    UNTIL #SITUACAO = 'OK'
END-REPEAT
...

REPEAT
    INPUT USING MAP 'CCSM0050'
    PERFORM #TRATA-MATRICULA
    IF #CONDICAO = 'S'
        ESCAPE TOP
    END-IF
    PERFORM GERA-COMANDO
    IF #COD-COMANDO = 4
        ESCAPE BOTTOM
    END-IF
END-REPEAT
```

➤ IF

A instrução IF é utilizada para controlar a execução de uma instrução ou de um grupo de instruções, baseado numa condição lógica. A condição lógica será utilizada para estabelecer se o bloco de código deverá ser executado.

Formato:

```
IF condição - lógica
{ IGNORE
  instruções }
[else
  { IGNORE
    instruções}]
END-IF
```

Expressões relacionais:

Igualdade: =, EQ, EQUAL

Diferença: NE, NOTEQUAL

Menor que: <, LT, LESS THAN

Maior que: >, GT, GREATER THAN

Menor ou igual: <=, LE, LESS EQUAL

Maior ou igual: >=, GE, GREATER EQUAL

Condição-lógica: estrutura válida, inclusive para outras instruções condicionais:

operando-1 (expressão relacional) operando-2

operando-1 (EQ operando-2 THRU operando-3)

operando-1 EQ operando-2 OR = operando-3

operando-1 EQ MASK(máscara)

operando-1 NE SCAN operando-2

IF #DEPTO EQ 'DES' THRU 'PRO'

IF #DEPTO = 'DES' OR = 'PRO'

IF #DEPTO = MASK('A')

IF #DATA EQ MASK(DDMMYYYY)

```
IF #CAMPO EQ MASK(NNNN)
IF #CAMPO EQ SCAN `*'
DEFINE DATA LOCAL
1 #VALOR    (P5)
END-DEFINE
INPUT 'DIGITE UM VALOR' #VALOR
IF #VALOR <25
    WRITE 'VALOR DIGITADO MENOR QUE 25.'
ELSE
    IF #VALOR <=50
        WRITE 'VALOR ENTRE 25 E 50.'
    ELSE
        WRITE 'VALOR MAIOR QUE 50.'
    END-IF
END-IF
END
```

A cláusula ELSE estabelece um bloco de código que será executado caso a condição lógica não seja satisfeita.

A cláusula IGNORE permite uma maior legibilidade ao código, não executando nada caso a condição seja satisfeita.

```
IF #SEXO = 'M' AND #IDADE <=21
    IGNORE
ELSE
    WRITE 'DEPENDENTE PERDEU DIREITO A I.R.'
END-IF
```

é igual ao código abaixo:

```
IF NOT (#SEXO = 'M' AND #IDADE <=21)
    WRITE 'DEPENDENTE PERDEU DIREITO A I.R.'
END-IF
```

➤ DECIDE

A instrução DECIDE é uma estrutura de múltipla escolha (case).

Formatos:

```
DECIDE FOR { FIRST
              EVERY } CONDITION

WHEN condição lógica
      Instruções

[ WHEN ANY
    Instruções ]

[ WHEN ALL .
    Instruções ]

WHEN NOME
      Instruções

END-DECIDE
```

```
DECIDE FOR { FIRST
              EVERY } VALUE OF operando-1
VALUE operando-2 [ ,... ]
      Instruções
VALUE operando-2 : operando-3
      Instruções

WHEN condição lógica
      Instruções

[ ANY VALUE
    Instruções ]

[ ALL VALUE
    Instruções ]

NOME VALUE
      Instruções

END-DECIDE
```

FIRST / EVERY: válido somente para o primeiro ou para todos.

ANY: executar se alguma condição/valor anterior for executado.

NONE: nenhuma condição/valor anterior for executado. É obrigatória sua codificação.

Pode ser utilizada a cláusula IGNORE em qualquer um dos blocos.

#### **DECIDE FOR EVERY CONDITION**

```
DEFINE SUBROUTINE CONSISTE-TELA
DECIDE FOR EVERY CONDITION
    WHEN #PRONTUARIO = 0
        REINPT 'Digitar prontuário.' MARK *# PRONTUARIO ALARM
    WHEN #PRONTUARIO NE 0
        CALLNAT 'DIGCHK' #PRONTUARIO #RC
        IF #RC NE 0
            REINPUT 'DV não cofere.' MARK *#PRINTUARIO ALARM
        END-IF
    WHEN #NASC NE MASK (DDMMYY)
        REINPUT 'Data nascimento inválida.' MARK #NASC ALARM
    WHEN ANY
        MOVE TRUE TO #ERRO
    WHEN NONE
        IGNORE
END DECIDE
END-SUBROUTINE
```

#### **DECIDE FOR FIRST CONDITION**

```
DECIDE FOR FIRST CONDITION
    WHEN #TIPO = 'A'
        PERFORM TRATA-CONTA-A
    WHEN #TIPO = 'C'
        PERFORM TRATA-CONTA-C
    WHEN NONE
        REINPUT 'Tipo de conta inválido.' MARK *#TIPO ALARM
END-DECIDE
```

#### **DECIDE ON FIRST VALUE**

```
DECIDE ON FIRST VALUE OF #COD
    VALUE 1
        PERFORM VE-COD-A
    VALUE2:4
        PERFORM VE-COD-2-3-4
```

```
    VALUE 5, 9, 11
        PERFORM VE-COD-5-9-11
    VALUE 8
        STOP
    NONE
        IGNORE
END-DECIDE
```

#### **DECIDE ON EVERY VALUE**

```
DECIDE ON EVERY VALUE OF #COD
    VALUE 1:6
        PERFORM VERIFICA-DATA
    VALUE 1:3
        PERFORM VERIFICA-NOME
    VALUE 4:6
        PERFORM VERIFICA-ENDEREÇO
    NONE
        IGNORE
END-DECIDE
```

#### **DECIDE DENTRO DE DECIDE**

```
DECIDE ON FIRST VALUE OF #CODIGO
    VALUE 1
        DECIDE FOR FIRST CONDITION
            WHEN #TIPO > #TIPOGR
                REINPUT 'Tipo inválido.' MARK ...
            WHEN #TIPO < #TIPOGR
                PERFORM GRAVA-TIPO
            WHEN NONE
                IGNORE
        END-DECIDE
    VALUE 2
        WRITE 'Tipo: ' #TIPO
    NONE VALUE
        IGNORE
END-DECIDE
```

➤ MOVE

Copia o conteúdo de um campo para outro ou atribui um valor a uma variável.

Formato:

[ROUNDED]	.	.
MOVE [BY NAME]	operando -1 TO operando -2	
[ALL]		
[EDITED]		

ROUNDED: faz a movimentação e arredonda, de acordo com o tamanho do campo receptor.

BY NAME: movimenta todos os campos de uma estrutura/registro para outro, desde que haja campos nas duas estruturas com o mesmo nome.

ALL: efetua preenchimento de uma variável *string* com o parâmetro especificado.

EDITED: efetua a movimentação de um campo com uma máscara para outro campo.

MOVE #VAR TO #VAR-2

MOVE ROUNDED 1,95 TO #N1                           /\*N1

#N1 =

MOVE BY NAME DATA TO DATA-INV

01 DATA

  02 D (N2)  
  02 M (N2)  
  02 A (N4)

01 DATA-INV

  02 A (N4)  
  02 M (N2)  
  02 D (N2)

MOVE ALL '\*' TO #LINHA                           /\* A10

#LINHA =

MOVE EDITED DATX(EM=DD/MM/YYYY) TO #A                           /\* A (A10)

#A =

➤ PERFORM

Executa uma chamada a uma rotina, interna ou externa. O controle é devolvido à instrução seguinte ao PERFORM, após a execução da subrotina.

*Formato:*

PERFORM subrotina [ parâmetro-1 ... parâmetro-n ]

O comando ESCAPE ROUTINE permite interromper a execução de uma subrotina antes do seu término, devolvendo o controle para a instrução PERFORM.

```
DEFINE DATA LOCAL
 1 FUNC VIEW OF FUNCIONARIO
   2 NOME-FUNC
 1 #COD-FUNC  (N9)
END-DEFINE
INPUT USING MAP 'FGSM0111'
PERFORM VALIDA-MAPA
PERFORM ALTERA-REGISTRO
...
DEFINE SUBROUTINE VALIDA-MAPA
IF COD-FUNC = 0
  REINPUT 'Código inválido.' MARK *#COD-FUNC ALARM
END-IF
...
END-SUBROUTINE
...
END
```

➤ ADD

Comando utilizado para efetuar a operação de soma.

*Formatos:*

```
ADD [ROUNDED] operando-1 ... operando-n TO operando-x
```

```
ADD [ROUNDED] operando-1 ... operando-n GIVING operando-x
```

O Comando ADD permite adicionar constantes ou variáveis a uma variável. Se for utilizada a opção TO, os valores do operando-1 até o operando-n serão adicionados ao operando-x, inclusive.

Se for utilizada a opção GIVING, o resultado da soma dos valores do operando-1 até o operando-n será armazenada no operando-x.

A cláusula ROUNDED permite que o resultado seja arredondado, ajustando-se ao tamanho do operando-x.

```

ADD #VAR #VAR-1 #VAR-2 TO #VAR-3
ADD #TAB(*) TO #SOMA
ADD ROUNDED #X #Y #Z GIVING #W

DEFINE DATA LOCAL
1 #A (N3) INIT <4>
1 #B (N3) INIT <7>
1 #C (N3) INIT <9>
1 #D (N3,2)
END-DEFINE
ADD #A #B TO #C
WRITE '=' #C
ADD 3A #B GIVING #C
WRITE '=' #C
ADD ROUNDED 50,775 45,881 TO #D
WRITE '=' #D
END

```

A execução gera o seguinte resultado:

```
#C: 20
#C: 11
#D: 96,66
```

➤ SUBTRACT

Comando utilizado para efetuar a operação de subtração.

Formato:

```
SUBTRACT [ ROUNDED ] oper-1 ... FROM oper-2 [ GIVING oper-3]
```

Quando for utilizada somente a cláusula FROM, o valor de oper-2 será atualizado com a subtração do oper-1 do oper-2.

Quando for utilizada a cláusula GIVING, o resultado da subtração do oper-1 do oper-2 será armazenado no oper-3.

Com a cláusula ROUNDED, o resultado é arredondado de acordo com o campo receptor.

```
SUBTRACT 10 FROM #VAR  
SUBTRACT #VAR1 #VAR2 FROM #VAR3  
SUBTRACT 50 FROM #VAR1 GIVING #VAR2
```

```
DEFINE DATA LOCAL  
1 #A (P2) INIT <50>  
1 #B (P2)  
1 #C (P1,1) INIT <2,4>  
END-DEFINE  
SUBTRACT 6 FROM #A  
WRITE '=' #A  
SUBTRACT 6 FROM 11 GIVING #A  
WRITE '=' #A  
SUBTRACT 3 4 FROM #A GIVING #B  
WRITE '=' #A '=' #B  
SUBTRACT -3 -4 FROM #A GIVING #B  
WRITE '=' #A '=' #B  
SUBTRACT ROUNDED 2,06 FROM #C  
WRITE '=' #C  
END
```

A execução gera o seguinte resultado:

```
#A: 44  
#A: 5  
#A: 5 #B: -2  
#A: 5 #B: 12  
#C: 0,3
```

➤ MULTIPLY

Comando utilizado para efetuar a operação de multiplicação.

Formato:

```
MULTIPLY [ ROUNDED ] ope-1 BY op2-2 [ GIVING ope-3 ]
```

Quando for utilizado somente BY, o resultado da operação será armazenado em ope-2.

Quando for utilizado o GIVING, o resultado será armazenado em ope-3.

A opção ROUNDED faz o arredondamento da operação.

```
MULTIPLY #TAB(*) BY 2  
MULTIPLY ROUNDED 3 BY 3,5 GIVING #RESULTADO
```

```
DEFINE DATA LOCAL  
1 #A (N3) INIT <20>  
1 #B (N5)  
1 #C (N3)  
1 #D (N2)  
END-DEFINE  
MULTIPLY #A BY 3  
WRITE '=' #A  
MULTIPLY #A BY 3 GIVING #B  
WRITE '=' #B  
MULTIPLY ROUNDED 3 BY 3,5 GIVING #C  
WRITE '=' #C  
MULTIPLY 3 BY -4 GIVING #D  
WRITE '=' #D  
MULTIPLY -3 BY -4 GIVING #D  
WRITE '=' #D  
MULTIPLY 3 BY 0 GIVING #D  
WRITE '=' #D  
END
```

A execução gera o seguinte resultado

```
#A: 60  
#B: 180  
#C: 10  
#D: -12  
#D: 12  
#D: 0
```

➤ DIVIDE

Comando utilizado para efetuar a operação de divisão.

Formato:

```
DIVIDE [ ROUNDED ] ope-1 INTO ope-2 [ GIVING ope-3 ] [ REMAINDER ope-4 ]
```

Se for utilizada a cláusula GIVING, o resultado da divisão será armazenado em ope-3, senão o resultado será armazenado em ope-2.

Se for utilizada a cláusula REMAINDER, o resto da divisão será armazenado em ope-4.

```
DIVIDE 5 INTO 20  
DIVIDE 5 INTO #VAR REMAINDER #RESTO
```

```
DEFINE DATA LOCAL  
1 #A (N7) INIT <20>  
1 #B (N7)  
1 #C (N3,2)  
1 #D (N1)  
1 #E (N1) INT <3>  
1 #F (N1)  
END-DEFINE  
DIVIDE 5 INTO #A  
WRITE '=' #A  
RESET INITIAL #A  
DIVIDE 5 INTO #A GIVING #B  
WRITE '=' #B  
DIVIDE 3 INTO 3,10 GIVING #C  
WRITE '=' #C  
DIVIDE 3 INTO 3,10 GIVING #D  
WRITE '=' #D  
DIVIDE 2 INTO #E REMAINDER #F  
WRITE '=' #E '=' #F  
END
```

A execução gera o seguinte resultado

```
#A: 4  
#B: 4  
#C: 1,03  
#D: 1  
#E: 1 #F: 1
```

➤ COMPUTE

A instrução COMPUTE executa operações aritméticas e permite a utilização de funções do NATURAL.

Formato:

```
[COMPUTE [ ROUNDED ]] ope-1 = expressão matemática
```

A cláusula ROUNDED faz com que o resultado seja arredondado.

A palavra COMPUTE é opcional, desde que o sinal de igualdade seja precedido de dois pontos:

```
#VAR := 4 * #MULT
```

Expressão aritmética:

( ) - parênteses	** - exponenciação	* - multiplicação
/ - divisão	+ - adição	- - subtração

ABS(operando) - valor absoluto(módulo) EXP(operando) - exponencial  
INT(operando) - parte inteira LOG(operando) - logaritmo natural  
SQRT(operando) - raiz quadrada  
VAL(operando) - extrai valor de parâmetro alfanumérico

```
COMPUTE #X = SQRT(#VALOR)
COMPUTE #X = VAL(#NUMERO)
COMPUTE #X = (#A * #B) / (#C ** 5-1)
    DEFINE DATA LOCAL
    1 #A (P4)
    1 #B (N3,4)
    1 #C (N3,4)
    END-DEFINE
    COMPUTE #A = 3 * 2 + 4 / 2 -1
    WRITE '=' #A
    COMPUTE ROUNDED 3B = 3 - 4 / 2 * ,89
    WRITE '=' #B
    COMPUTE #C = SQRT (#B)
    WRITE '=' #C
    END
A execução gera o seguinte resultado:
#A: 7
#B: 1,2200
#C: 1,1045
```

➤ COMPRESS

Concatena dois ou mais campos em um único campo.

Formato:

```
COMPRESS operando -1 ... operando - n [IN]TO  
operando - resultado  
[ { LEAVING NO [SPACE] } ]  
[ { WITH DELIMITER ope-3 } ]
```

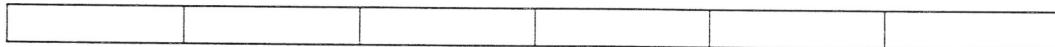
operando-1 ... operando-n: qualquer tipo de variável ou constante.

operando-resultado: variável alfanumérica que receberá o resultado da concatenação.

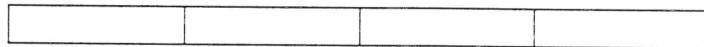
LEAVING NO: concatena sem deixar espaço em branco entre os dados.

WITH DELIMITER: especifica um caracter para servir de separador entre os campos.

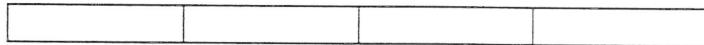
COMPRESS 'AA' '01' INTO #AREA



MOVE 1 TO #DIA /\* #DIA (N2)  
MOVE 5 TO #MÊS /\* #MÊS (N2)  
COMPRESS #DIA #MÊS TO #DIA-MÊS /\* #DIA-MÊS (A5)



COMPRESS #DIA ' / ' #MÊS TO #DIA-MÊS LEAVING NO



COMPRESS #DIA #MÊS TO #DIA-MÊS WITH DELIMITER ' / '

➤ DISPLAY

Instrução para imprimir dados do programa, em formato de colunas. A instrução DISPLAY automaticamente inclui os cabeçalhos, com o mesmo nome da variável, caso não seja informado um cabeçalho diferente.

Formato:

```
DISPLAY [( rel )] [NOTITLE] [( parâmetros )]
[ { nX } ] [{ '=' }]
[ { nT } ] [{ 'texto' }] operando [( parâmetros )]
[ { 'c' n } ]
```

Opções:

NOTITLE: suprime a linha de cabeçalho da página.

rel: número do relatório. O Natural pode controlar até 32 relatórios simultâneos, numerados de 0 a 31.

parâmetros:

SF(Spacing Factor): Fator de espaçamento das colunas. Valores: de 1 a 30

HC(Heading Centering): Ajuste do cabeçalho das colunas - Center / Left / Right

FC(Filler Character): Caracter de preenchimento do cabeçalho. Default: brancos

UC(Underling Character): Caracter utilizado para sublinhar os cabeçalhos. Default: '-'

```
DISPLAY (SF=25) HC=L UC=*) NUM-MATRIC-EMPREG NOME-EMPREGADO  
SIGLA-ORGAO
```

```
DISPLAY 'MATRICULA DO EMPREGADO' NUM-MATRIC-EMPREG
```

```
DISPLAY 'MATRIC' NUM_MATRIC_EMPREG
```

```
DISPLAY 'MATRICULA/DO/EMPREGADO' NUM-MATRIC-EMPREG
```

Parâmetros de variáveis:

AL(Alphanumeric Length): Tamanho máximo para exibição de campo alfanumérico

NL(Numeric Length): Tamanho máximo para exibição de campo numérico.

```
DISPLAY NUM-MATRIC-EMPREG NOME-EMPREGADO (AL=20)
```

```
DISPLAY NUM-MATRIC-EMPREG (AL=3) NOME-EMPREGADO  
SIGLA-ORGAO (AL=4)
```

Vejamos os seguintes programas como exemplo de utilização da instrução DISPLAY:

```
DEFINE DATA LOCAL  
1 DEFINE DATA LOCAL  
1 EMP VIEW OF AD-EMPREGADO-INTERNET  
    2 NUM-MATRIC-EMPREG          /* A6  
    2 NOME-EMPREGADO           /* A36  
    2 SIGLA-ORGAO              /* A10  
END-DEFINE
```

```
SELECT  NUM_MATRIC_EMPREG, NOME_EMPREGADO, SIGLA_ORGAO  
INTO    VIEW EMP  
FROM    AD-EMPREGADO-INTERNET  
WHERE   SIGLA_ORGAO = 'DIVISA'  
END-SELECT  
END
```

A execução gera o seguinte resultado

Page	1	5
-----	-----	-----
046115	GERALDO GUILHERME CARVALHO CALDEIRA	DIVISA
046479	CLARINO DE ABREU FILHO	DIVISA
064801	ROMAN DARIO CUATTRIN	DIVISA
065113	PAULO EMILIO HARTUNG	DIVISA
067127	ANAMARIA LOPES FERREIRA	DIVISA

Diferenças entre DISPLAY e WRITE

Funções	DISPLAY	WRITE
Título de página	Sim	Sim
Título de campo	Sim	Não
Tamanho ocupado pelo campo	Determinado pelo título ou campo, o que for maior	Campo
Espacejamento entre campos	SF, nX ou nT	NX ou nT
Estouro de linha	Não	Sim
Uso de '/*'	Válido para próximo campo	Cria nova linha
Impressão de array (matriz)	Vertical	Horizontal
Orientação	Coluna	Linha

➤ **FETCH**

Este comando é utilizado para execução de outro programa.

*Formato:*

```
FETCH [ RETURN ] programa [ ope-1 ... ope-40 ]
```

Ope-1 ... ope-40: parâmetros que serão passados ao programa

A cláusula RETURN faz com que, ao término do programa chamado, o controle retorne à instrução seguinte ao FETCH. Caso não seja especificada, o controle retorna ao NATURAL após a execução do programa.

```
DEFINE DATA LOCAL
1 #OPCAO (N1)
END-DEFINE
INPUT USING MAP 'OPCM0001'
IF NOT (#OPCAO = 1 OR = 2 OR = 3)
    REINPUT 'Opção inválida.' MARK *#OPCAO ALARM
END-IF
DECIDE ON FIRST VALUE OF #OPCAO
    VALUE 1
        FETCH RETURN 'PRGP0001'
    VALUE 2
        FETCH RETURN 'PRGP0002'
    NONE
        FETCH RETURN 'PRGP0003'
END-DECIDE
END
```

➤ FOR

Instrução utilizada para iniciar um processamento de loop e controlar o número de vezes que o loop será executado.

*Formato:*

```
FOR ope-1 = ope-2 TO ope-4 [ STEP ope-4 ]
    Instruções
END-FOR
```

ope-1: pode ser uma variável de usuário ou um campo de arquivo, que terá o valor alterador a cada ocorrência do loop.

ope-2 e ope-3: variáveis ou constantes, de valor inicial e final do loop.

ope-4: incremento, podendo ser positivo ou negativo.

```
DEFINE DATA LOCAL
1 #I      (P3)
1 #RAIZ   (P3, 4)
END-DEFINE
FOR #I = 1 TO 101 STEP 2
    IF #I = 51
        ESCAPE TOP
    END-IF
    COMPUTE #RAIZ = SQRT(#I)
    WRITE #I #RAIZ
END-FOR
END
```

➤ AT BREAK

Comando usado para detectar uma "quebra" do valor de um campo ordenado.

*Formato:*

```
[ AT ] BREAK [ OF ] campo [ /n/ ]
      Instruções
      END-BREAK
```

Campo: atributo do arquivo, cujo conteúdo será utilizado para teste de quebra, ou seja o AT BREAK terá os comandos internos executados toda vez que ocorrer uma alteração do valor do campo em relação ao conteúdo anterior.

/n/: pode-se especificar a quebra de um certo número de caracteres iniciais do campo, através da cláusula /n/. Por exemplo, se quisermos uma quebra quando houver alteração das primeiras letras de um nome, basta codificar AT BREAK OF NONE /1/.

O bloco de comandos é finalizado com END-BREAK.

Mais de um AT BREAK pode ser codificado numa operação de leitura, e quando acontece uma quebra, as quebras de níveis anteriores também acontecem.

Funções disponíveis na quebra:

COUNT: quantidade de registros lidos

SUM: soma dos valores

OLD: conteúdo do campo de quebra antes de ocorrer a quebra

AVER: média dos valores

MAX: maior valor lido

MIN: menor valor lido

Quebra simples:

```
SELECT  NUM_MATRIC_EMPREG, NOME_EMPREGADO, SIGLA_ORGAO,
        VL_SALARIO
INTO    VIEW EMP
FROM    AD-EMPREGADO-INTERNET
WHERE   SIGLA_ORGAO = 'DIVISA'
AT BREAK OF SIGLA_ORGAO
      WRITE '=' (70) /
      10T 'SIGLA: 'OLD(SIGLA_ORGAO) (AD=I) /
      40T 'SALÁRIO MÉDIO: ' AVER(VL_SALARIO) /
      40T 'NR FUNCIONARIOS: 'COUNT (VL_SALARIO) /
      40T 'TOTAL SALARIO: ' SUM(VL_SALARIO)
END-BREAK
DISPLAY NUM_MATRIC NOME_EMPREGADO SIGLA_ORGAO VL_SALARIO
END-SELECT
END
```

Quebra composta:

```
SELECT  NUM_MATRIC_EMPREG, NOME_EMPREGADO, SIGLA_ORGAO,
        SIGLA_SETOR ,VL_SALARIO
INTO    VIEW EMP
FROM    AD-EMPREGADO-INTERNET
WHERE   SIGLA_ORGAO > ' '
AND    SIGLA-SETOR > ' '
AT BREAK OF SIGLA_SETOR
      WRITE
      10T 'SIGLA_SETOR :OLD(SIGLA_SETOR) (AD=I) /
      40T 'NR FUNCIONARIOS: 'COUNT (VL_SALARIO) /
      40T 'TOTAL SALARIO: ' SUM(VL_SALARIO)
END-BREAK

AT BREAK OF SIGLA_ORGAO
      WRITE '=' (70) /
      10T 'SIGLA ORGÃO: 'OLD(SIGLA_ORGAO) (AD=I) /
      40T 'SALÁRIO MÉDIO: ' AVER(VL_SALARIO) /
      40T 'NR FUNCIONARIOS: 'COUNT (VL_SALARIO) /
      40T 'TOTAL SALARIO: ' SUM(VL_SALARIO)
END-BREAK
DISPLAY NUM_MATRIC NOME_EMPREGADO SIGLA_SETOR VL_SALARIO
END-SELECT
END
```

➤ AT START OF DATA

Comando que executa uma série de comandos no início da leitura de um file.

*Formato:*

```
[ AT ] START .[ OF ] DATA  
      Instruções  
END-START
```

Deve ser codificado dentro do laço de leitura.

```
SELECT  NUM_MATRIC_EMPREG, NOME_EMPREGADO, SIGLA_ORGAO,  
        VL_SALARIO  
INTO    VIEW EMP  
FROM    AD-EMPREGADO-INTERNET  
WHERE   SIGLA_ORGAO = 'DIVISA'  
AT START OF DATA  
      WRITE 'INICIO DO RELATORIO'  
END-START  
.....  
END-SELECT
```

➤ AT END OF DATA

Comando que executa uma série de comandos ao término da leitura de um file.

*Formato:*

```
[ AT ] END .[ OF ] DATA  
      Instruções  
END-ENDDATA
```

Deve ser codificado dentro do laço de leitura.

```
SELECT NUM_MATRIC_EMPREG, NOME_EMPREGADO, SIGLA_ORGAO,  
      VL_SALARIO  
INTO   VIEW EMP  
FROM   AD-EMPREGADO-INTERNET  
WHERE  SIGLA_ORGAO = 'DIVISA'  
.....  
AT END OF DATA  
    WRITE 'FINAL DO RELATORIO'  
END-ENDDATA  
  
END-SELECT
```

➤ AT TOP OF PAGE

Comando que executa uma série de comandos toda vez que uma página for inicializada para impressão de um relatório.

*Formato:*

```
[ AT ] TOP [ OF ] PAGE  
Instruções  
END-TOPPAGE
```

Pode ser codificado em qualquer lugar do programa.

...

**AT TOP OF PAGE**

```
WRITE '*** RELATÓRIO DE VENDAS POR TRIMESTRE ***'  
END-TOPPAGE
```

...

➤ AT END OF PAGE

Comando que executa uma série de comandos quando uma condição de fim de página for detectada.

*Formato:*

```
[ AT ] END [ OF ] PAGE  
Instruções  
END-ENDPAGE
```

Pode ser codificado em qualquer parte do programa.

...

**AT END OF PAGE**

```
WRITE 'PAGINA: ' #PAG (EM=ZZ9)  
ADD 1 TO #PAG  
END-ENDPAGE
```

...

➤ DEFINE SUBROUTINE

Este comando é utilizado para definir um subrotina, interna ou externa.

*Formato:*

```
DEFINE SUBROUTINE nome-da-rotina
    Instruções
END-SUBROUTINE
```

nome-da-rotina: pode conter até 32 caracteres.

A subrotina é chamada através do comando PERFORM. Uma alteração na subrotina externa não condiciona à recompilação do objeto que a chama.

Subrotina Interna:

```
...
END-SELECT
*
DEFINE SUBROUTINE
IF #CAMPO <= 0
    WRITE 'Campo inválido.'
    ESCAPE ROUTINE
END-IF
FOR #X = 1 TO #CAMPO
    WRITE '=' #X
END-FOR
END-SUBROUTINE
END
```

Subrotina Externa:

```
DEFINE DATA PARAMETER
1 #SENHA (A5)
1 #RETORNO (L)
END-DEFINE
DEFINE SUBROUTINE TESTA-SENHA-ENTRADA
IF #SENHA = '26511'
    #RETORNO := TRUE
ELSE
    #RETORNO := FALSE
END-IF
END-SUBROUTINE
END
```

➤ FORMAT

A instrução FORMAT especifica parâmetros de entrada e saída.

*Formato:*

```
FORMAT [ (rel) ] LS = nn PS = mm
          SF, AL, NL, HC, UC, ...
- nn: 10 a 250
- mm: 10 a 250
```

```
DEFINE DATA LOCAL
1 EMP VIEW OF AD-EMPREGADO-INTERNET
  2 NUM-MATRIC-EMPREG           /* A6
  2 NOME-EMPREGADO             /* A36
  2 SIGLA-ORGAO                /* A10
  2 DATA-ADMISS-EMPREG         /* A10
END-DEFINE
FORMAT(1) LS=100 PS=50
WRITE(1) TITLE 'RELAÇÃO DE FUNCIONÁRIOS' (I)
  90T *PAGE-NUMBER(1)
SELECT  NUM_MATRIC_EMPREG, NOME_EMPREGADO, SIGLA_ORGAO,
INTO    VIEW EMP
FROM    AD-EMPREGADO-INTERNET
WHERE   SIGLA_ORGAO = 'DIVISA'
        DISPLAY NUM_MATRIC_EMPREG NOME_EMPREGADO SIGLA_ORGAO
END-SELECT
END
```

➤ WRITE TITLE

Provê título ao relatório especificado, gerado através de DISPLAY ou WRITE.

*Formato:*

```
WRITE [ (rel) ] TITLE [ LEFT ] [ (I) ]
{ [{nT}] }
{ [/] [{nX}] operando [ (parâmetros) ] }
{ [{x/y}] }
SKIP variável/constante
```

Somente pode ser definido um WRITE TITLE por relatório, no máximo. O WRITE TITLE é executado toda vez que ocorre um estouro de variável do Natural \*LINE-COUNT, que contém armazenado o número da última linha impressa, ou por uma instrução NEWPAGE, que força o salto para a próxima página.

```
WRITE TITLE 'RELAÇÃO DE FUNCIONÁRIOS' (I)
    70T 'PAG' *PAGE-NUMBER
        SKIP 2
```

\*LINE-COUNT: contador de linha. Ocorre estouro quando ultrapassa o valor definido na variável PS (Page Size) - tamanho da página, em linhas.

\*PAGE-NUMBER: variável do Natural, responsável pelo contador de páginas.

```
DEFINE DATA LOCAL
1 EMP VIEW OF AD-EMPREGADO-INTERNET
    2 NUM-MATRIC-EMPREG          /* A6
    2 NOME-EMPREGADO            /* A36
    2 SIGLA-ORGAO              /* A10
END-DEFINE
FORMAT(1) LS=100 PS=50
WRITE(1) TITLE LEFT 'RELAÇÃO DOS FUNCIONÁRIOS' (I)
    90T 'PAG: ' *PAGE-NUMBER(1) (EM=999)
SELECT NUM_MATRIC_EMPREG, NOME_EMPREGADO, SIGLA_ORGAO,
INTO   VIEW EMP
FROM   AD-EMPREGADO-INTERNET
WHERE  SIGLA_ORGAO = 'DIVISA'
DISPLAY NUM_MATRIC_EMPREG NOME_EMPREGADO SIGLA_ORGAO
END-SELECT
END
```

A execução gera o seguinte resultado:

Relação de funcionários

NUM-MATRIC-EMPREG	NOME-EMPREGADO	SIGLA-ORGAO
046115	GERALDO GUILHERME CARVALHO CALDEIRA	DIVISA
046479	CLARINO DE ABREU FILHO	DIVISA
064801	ROMAN DARIO CUATTRIN	DIVISA
065113	PAULO EMILIO HARTUNG	DIVISA
067127	ANAMARIA LOPES FERREIRA	DIVISA

➤ NEWPAGE

Este comando provoca um avanço de página do relatório.

Formato:

```
NEW PAGE [(R)] [[{ IF }] [LESS] [THAN] ope-1 [LINES] [LEFT]]  
          { WHEN }
```

O NEWPAGE avança a página e provoca a impressão do cabeçalho na página seguinte.

IF LESS THAN: se especificada a cláusula, ocorrerá quebra de página somente quando houver menos linhas do que o especificado no parâmetro.

```
...  
NEWPAGE (1) IF LESS THAN 5 LEFT  
...
```

➤ EJECT

Este comando provoca um avanço de página do relatório, sem gerar cabeçalho na próxima página.

Formato:

```
EJECT [(R)] [[{ IF }] [LESS].[THAN] ope-1 [LINES] [LEFT]
          { WHEN}]
```

O EJECT avança página, mas não provoca a impressão do cabeçalho na página seguinte.

IF LESS THAN: se especificada a cláusula, ocorrerá quebra de página somente quando houver menos linhas do que o especificado no parâmetro.

```
...
EJECT (1) IF LESS THAN 5 LEFT
...
```

➤ SET CONTROL

Instrução efetuada para executar um comando de terminal.  
Formato:

```
SET CONTROLE operando-1
```

Quando o comando de terminal é executado pelo comando SET CONTROL, não é necessário especificar o caracter de terminal (%).

```
...
SET CONTROL 'M23'
SET CONTROL 'L'
...
```

Principais comandos de terminal

%H	HardCopy: Efetua uma cópia da tela na impressora associada
%Knn	Simula o pressionamento de uma tecla de PF
%L	Lowercase. Desabilita a transformação de minúsculas para maiúsculas
%N	Simula o pressionamento da tecla <ENTER>
%Mnn	Estabelece a linha nn como linha para mensagens
%U	Uppercase. Habilita a transformação de minúsculas para maiúsculas
%W	Window. Permite especificar uma janela

➤ STOP

Termina a execução de um programa e retorna o controle para o Natural.

```
...
IF #OPCAO = 9
  STOP
END-IF
```

```
...
```

➤ TERMINATE

O comando termina a execução do programa e do Natural, retornando o controle ao monitor de teleprocessamento (CICS, TSO, COMPLETE, ETC.)

```
...
IF *PF-KEY = 'PF5' OR = 'PF17'
  TERMINATE
END-IF
```

```
...
```

➤ SKIP

Gera linhas em branco em um relatório.

*Formato:*

SKIP operando-1 [ LINES ]

```
...
WRITE 'TOTAL DE AGENCIAS: ' #TOT-AGENCIA
SKIP2
```

```
...
```

➤ EXAMINE

Instrução para pesquisar o conteúdo de um campo alfanumérico ou uma faixa de campos de uma variável array, através de uma *string*. Permite apagar ou trocar a ocorrência.

*Formato:*

```
EXAMINE operando-1 FOR operando-2
{           DELETE [FIRST ]      }
{REPLACE [ FIRST ] operando-3   }
{GIVING POSITION operando-3   }
{GIVING INDEX operando-3     }
```

operando-1: representa o campo a ser examinado

operando-2: representa o valor a ser utilizado na operação de pesquisa.

DELETE: apaga a ocorrência especificada, concatenando o resto do string. Se FIRST for especificado, apaga apenas a primeira ocorrência.

REPLACE: troca a ocorrência especificada em operando-2 pela especificada em operando-3. Se FIRST for especificado, apenas a primeira ocorrência será alterada.

GIVING POSITION: usado para obter a partir de qual byte o operando-2 inicia no operando-1. Se o operando-2 não for encontrado, retorna 0 (zero).

GIVING INDEX: usado para obter o número da ocorrência (índice) em que o operando-2 ocorre no operando-1, que deverá ser uma matriz. Caso não seja encontrado, retorna 0 (zero).

```
DEFINE DATA LOCAL
1 #DEP1      (A18)
1 #DEP       (A18)  INIT <'AAABBCCDDDEEEFFF' >
1 REDEFINE #DEP
2 #TAB_DEP   (A3/6)
1 #I         (I1)
1 #J         (I1)
END-DEFINE
MOVE #DEP TO #DEP1
EXAMINE #TAB_DEP(*) FOR 'BBB' GIVING INDEX #I
EXAMINE #DEP FOR 'BBB' GIVING POSITION #J
EXAMINE #DEP FOR 'DE' REPLACE 'XY'
EXAMINE #DEP FOR 'C' REPLACE FIRST 'W'
EXAMINE #DEP FOR 'F' REPLACE 'Z'
EXAMINE #DEP FOR 'B' DELETE
DISPLAY #I #J #DEP1 #DEP
END
```

A execução gera o seguinte resultado:

#I	#J	#DEP1	#DEP
2	4	AAABBCCDDDEEEFFF	AAAWCCDDXYEEZZZ

---

➤ **DEFINE WINDOW**

Comando utilizado para especificar o tamanho, posição e atributos de uma janela.

*Formato:*

```
DEFINE WINDOW nome
  SIZE lin * col
  BASE lin / col
  [ TITLE ope-1 ]

  FRAMED [ ON ]
          [ OFF ]
  [ POSITION [ ON ]
            [ OFF ] ]
```

**SIZE:** tamanho da janela, em linhas \* colunas

**BASE:** posição de abertura da janela, em relação a uma janela normal (24 \* 80), em linhas / colunas

**TITLE:** título da janela. Pode ser uma variável alfanumérica ou uma constante.

**FRAMED:** se informado ON, a janela terá moldura. Se informado OFF, a janela não terá.

**POSITION:** deve ser informado apenas se a janela a ser exibida for maior do que a tela física. Se a janela for maior, ON mostrará na última linha uma informação de que existe mais janela a ser exibida. OFF desabilita a linha de posição.

O comando **DEFINE WINDOW** apenas define uma janela, sendo necessário o comando **SET WINDOW** para tornar a janela ativa.

```
DEFINE DATA LOCAL
1 #I    (P3)
END-DEFINE
DEFINE WINDOW WIND1
  SIZE 13 * 10
  BASE 5 / 12
  FRAMED ON POSITION OFF
RESET #I
...
```

➤ SET WINDOW

Comando utilizado para ativar / desativar uma janela.

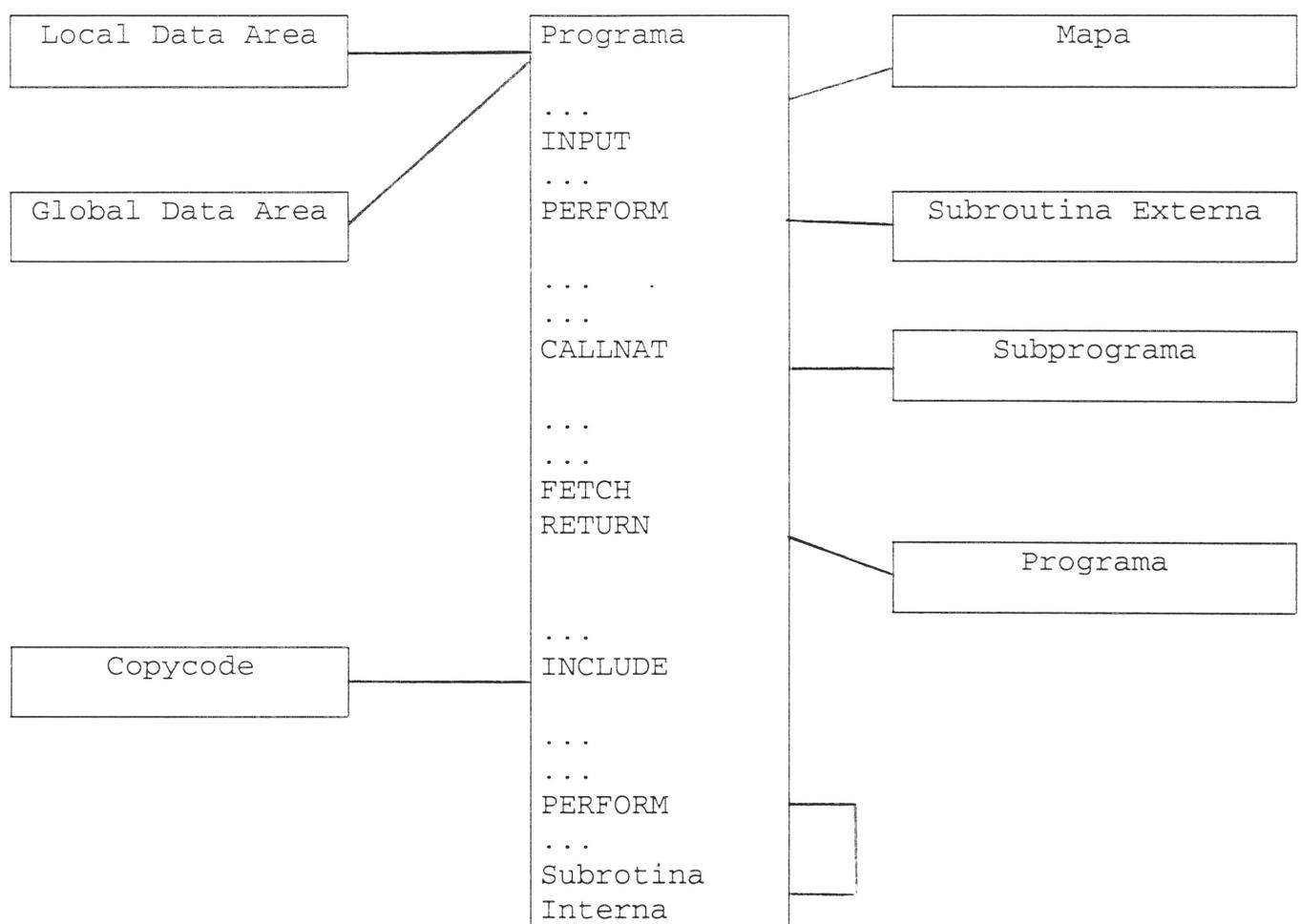
*Formato:*

```
SET WINDOW [ nome-da-janela ]
[ OFF ]
```

Com o comando SET WINDOW nome-da-janela, ativa-se a janela especificada, o que significa que as próximas instruções de entrada / saída utilizarão essa janela, até que a mesma seja desativada ou uma nova seja ativada.

OFF: desativa a janela lógica aberta atualmente.

```
DEFINE DATA LOCAL
1 #I      (P3)
END-DEFINE
DEFINE WINDOW WIND1
    SIZE 13 * 10
    BASE 5 / 12
    FRAMED ON POSITION OFF
RESET #I
SET WINDOW 'WIND1'
INPUT USING MAP 'TTRM0010'
SET WINDOW OFF
END
```



15:04:52 \*\*\*\*\* NATURAL LIST COMMAND \*\*\*\*\* 31/10/2002  
User X033501 - List DDM DB2TST-CADASTRO\_CURSO - Library CURSO

DDM DBID 250 DDM FNR 1 VSAM Name Default Sequence ? Page 1

T L	DB Name	F	Leng	S	D	Remarks
1 OA	CD_USU	A	11	D		
1 AA	COD_CPF_CGC	P	14,0	D		
1 AB	NOM	A	60	D		
1 AC	NR_SEQL_END	I	2	D		
1 AD	SEXO	A	1	D		
1 AE	NVL_ECLD	I	2	D		
1 AF	DT_ATL	A	10	D		

End of List.

Command ===>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---

Help Print Exit Long -- - + ++ Canc

16:01:27 \*\*\*\*\* NATURAL LIST COMMAND \*\*\*\*\* 31/10/2002  
User X033501 - List DDM DB2TST-ENDRECO\_CURSO - Library CURSO

DDM DBID 250 DDM FNR 1 VSAM Name Default Sequence ? Page 1

T L DB Name F Leng S D Remarks

---

1 OA	CD_USU	A	11	D
1 OB	NR_SEQL_END	I	2	D
1 AA	CD_TIP_END	I	2	D
1 AB	NM_CID	A	60	D
1 AC	SG_UF	A	2	D
1 AD	NM_RUA	A	60	D
1 AE	NM_BAI	A	60	D
1 AF	NR_CEP	I	4	D
1 AG	NR_DDD	I	2	D
1 AH	TEL_RSDC	I	4	D
1 AI	TCEL	I	4	D
1 AJ	TEL_CML	I	4	D
1 AK	DT_ATL	A	10	D

End of List.

Command ===>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---

Help Print Exit Long -- - + ++ Canc

**AREA RESERVADA PARA SEUS APONTAMENTOS**

---

**AREA RESERVADA PARA SEUS APONTAMENTOS**

---

**AREA RESERVADA PARA SEUS APONTAMENTOS**

---

