



Manual Prático de Programação

**Enviado por :
Rafael Dias Corrêa**

Manual Prático de Programação

(Versão 1.0)

- ◆ COBOL - COmmon Business-Oriented Language
- ◆ CICS - Customer Information Control System
- ◆ SQL - Structured Query Language
- ◆ JCL - Job Control Language
- ◆ TSO - Time Sharing Option
- ◆ ENDEVOR

Índice

COBOL II.....	1
1 INTRODUÇÃO AO COBOL	1
1.1 <i>Estrutura Geral de um Programa</i>	1
1.2 <i>Instruções Básicas</i>	5
1.2.1 IF e PERFORM	5
1.2.2 PERFORM UNTIL, MOVE e ADD	6
1.2.3 COMPUTE e COMPUTE ROUNDED	6
1.2.4 DIVIDE e SUBTRACT	6
1.2.5 EVALUATE	6
2 PROGRAMAÇÃO BATCH.....	8
2.1 <i>Instruções Batch</i>	8
2.1.1 ACCEPT	8
2.1.2 STOP RUN	8
2.1.3 DISPLAY	8
2.1.4 CALL	8
2.1.5 GOBACK	9
2.2 <i>Ficheiros Sequenciais</i>	9
2.2.1 Definição	9
2.2.2 Abertura	9
2.2.3 Leitura.....	10
2.2.4 Escrita.....	10
2.2.5 Fecho	10
2.3 <i>Ficheiros Indexados</i>	11
2.3.1 Definição	11
2.3.2 Abertura	11
2.3.3 Posicionamento.....	11
2.3.4 Leitura.....	11
2.3.5 Escrita.....	11
2.3.6 Eliminação de Registos.....	11
2.3.7 Fecho	12
2.4 <i>Mapas</i>	12
2.5 <i>Variáveis de Sistema</i>	13
2.5.1 RETURN-CODE	13
ANEXO COM FILE STATUS	14
CICS.....	1
1 PROGRAMAÇÃO ONLINE.....	1
1.1 <i>Estrutura dos Programas</i>	1
1.1.1 Programa chamado via LINK	1
1.1.2 Programa chamado via START	1
1.2 <i>Instruções Básicas</i>	4
1.2.1 SEND.....	4
1.2.2 RECEIVE	6
1.2.3 SYNCPOINT e ROLLBACK	6
1.2.4 LINK.....	6
1.2.5 START.....	6
1.2.6 RETURN	7
1.2.7 HANDLE AID.....	8
1.2.8 ASKTIME	9
1.2.9 FORMATTIME	9
1.2.10 HANDLE CONDITION	9
1.2.11 IGNORE CONDITION	11
1.2.12 RETRIEVE.....	11
1.3 <i>Ficheiros Indexados</i>	11
1.3.1 Definição	11
1.3.2 Abertura.....	12
1.3.3 Posicionamento.....	12
1.3.4 Leitura.....	12
1.3.5 Escrita	13
1.3.6 Eliminação de Registos.....	14
1.3.7 Fecho	14

1.4 Mapas	14
1.5 Temporary Storage	15
1.5.1 Leitura.....	16
1.5.2 Escrita.....	16
1.5.3 Eliminação	16
1.6 Transient Data.....	17
1.6.1 Leitura.....	17
1.6.2 Escrita.....	17
1.6.3 Eliminação	17
1.7 Ecrãs.....	18
1.7.1 Sufixos das Variáveis	18
1.7.2 Ecrã de Manipulação de Dados sem Ocorrências	19
1.7.3 Ecrã de Manipulação de Dados com Ocorrências.....	21
1.7.4 Ecrã de Consulta de Dados sem Ocorrências.....	23
1.7.5 Ecrã de Consulta de Dados com Ocorrências	28
1.8 Variáveis de Sistema.....	29
1.8.1 DFHCOMMAREA	29
1.8.2 EIBCALEN	29
1.8.3 EIBTRMID.....	29
1.8.4 EIBCPOSN.....	29
1.8.5 EIB Aid.....	29
1.8.6 EIBRESP	29
1.8.7 Estrutura DFHAID.....	30
1.8.8 Estrutura DFHBMSCA.....	31
2 COMANDOS ONLINE	33
2.1 CEMT	33
2.1.1 CEMT SET PROGRAM	33
2.1.2 CEMT SET TASK.....	33
2.1.3 CEMT SET TRANSACTION	33
2.1.4 CEMT SET TERM	33
2.1.5 CEMT SET FILE.....	34
2.2 CECI.....	34
2.3 CEDF.....	34
2.4 CMAC.....	34
2.5 DSNCLISP STAT	34
ANEXO COM ERROS CICS.....	35

SQL..... 1

1 COMANDOS BÁSICOS.....	1
1.1 INCLUDE.....	1
1.2 WHENEVER SQLERROR	1
1.3 SELECT.....	2
1.4 INSERT.....	3
1.5 UPDATE.....	4
1.6 DELETE	4
1.7 COMMIT	4
1.8 ROLLBACK.....	4
2 CURSORES.....	5
2.1 Definição	5
2.2 Abertura.....	6
2.3 Seleção de Dados.....	6
2.4 Actualização de Dados.....	6
2.5 Fecho.....	6
3 FUNÇÕES	7
3.1 SUM, COUNT.....	7
3.2 MAX, MIN, AVG.....	7
3.3 DATE, YEAR, MONTH, DAY, YEARS, MONTHS, DAYS	7
3.4 HOUR, MINUTE, SECOND, MICROSECOND.....	9
3.5 SUBSTR, CHAR, DECIMAL.....	9
4 BIND PACKAGE E BIND PLAN	10
5 TABELAS DE SISTEMA	11
6 ERROS SQL MAIS COMUNS.....	12

ANEXO COM ESTRUTURA SQLCA	18
JCL.....	1
1 JOBS	1
1.1 Job de Compilação	3
1.2 Job de Bind	3
1.3 Job de Load com SYSPUNCH	4
1.4 Job de Unload	4
1.5 Job de Manipulação de Ficheiros	5
1.5.1 Fecho de ficheiro no CICS	5
1.5.2 Eliminação de ficheiros de trabalho	5
1.5.3 Cópia de um ficheiro	5
1.5.4 Ordenação de um ficheiro	5
1.5.5 Impressão de avisos constantes num ficheiro	5
1.5.6 Carregamento (Load) de um ficheiro para uma tabela	6
1.5.7 Inicialização de um ficheiro sequencial	6
1.5.8 Eliminação e realocação de um ficheiro indexado	6
1.5.9 Inicialização de um ficheiro indexado	6
1.5.10 Separação de um ficheiro em vários	7
1.5.11 Listagem de um ficheiro	7
1.5.12 Abertura de ficheiro no CICS	7
1.6 Job de Execução de Programas	7
1.6.1 Programa sem ficheiros	7
1.6.2 Programa que lê um ficheiro	7
1.6.3 Programa que imprime um mapa	8
1.6.4 Programa que escreve num ficheiro temporário	8
1.6.5 Programa que escreve num ficheiro e tem um parâmetro de SYSIN	8
2 PARMLIBS	9
2.1 Execução de Programas	9
2.2 Ordenação de Ficheiros	9
2.3 Bind Package	10
2.4 Bind Plan	10
2.5 Load de Tabelas	10
3 PROCS	11
3.1 Compilação Batch sem DB2	11
3.2 Compilação Batch com DB2	12
3.3 Compilação Online sem DB2	13
3.4 Compilação Online com DB2	14
3.5 Execução de Programas sem DB2	14
3.6 Execução de Programas com DB2	14
3.7 Ordenação de Ficheiros	14
3.8 Execução de Bind Package	14
3.9 Execução de Bind Plan	14
3.10 Load (Carregamento) de Tabelas	14
ANEXO COM SYSTEM ABENDS	16
TSO	1
1 PARÂMETROS DO TERMINAL E DO UTILIZADOR	1
2 VISUALIZAÇÃO DE DATA SETS	2
3 EDIÇÃO DE DATA SETS	2
4 UTILITÁRIOS	3
4.1 Manipulação de Membros	4
4.2 Manipulação de Data Sets	4
4.3 Movimentação e Cópia de Membros	5
4.4 Listagem de Data Sets	5
4.5 Comparação de Membros	6
4.6 Pesquisa em Membros	6
5 COMANDOS TSO	7
6 VISUALIZAÇÃO DE JOBS	7

ENDEVOR.....	1
1 OPÇÕES DE VISUALIZAÇÃO	1
1.1 Peças de Software (Fontes)	1
1.2 Resultados de Compilação	2
1.3 Ambientes	2
1.4 Sistemas	3
1.5 Subsistemas.....	3
1.6 Tipos de Elementos.....	3
1.7 Grupos de Processadores.....	3
2 OPÇÕES FOREGROUND	4
2.1 Passagem de Elementos para o ENDEVOR.....	4
2.2 Obtenção de Elementos existentes no ENDEVOR.....	4
2.3 Recompilação de Elementos no ENDEVOR.....	5
2.4 Movimentação de Elementos no ENDEVOR.....	5
2.5 Eliminação de Elementos no ENDEVOR	5
3 OPÇÕES BATCH	6

COBOL

Capítulo 1 - Introdução ao COBOL

1.1 Estrutura de um Programa

Os programas em COBOL são obrigatoriamente constituídos por quatro divisões:

1. A IDENTIFICATION DIVISION marca o início de um programa COBOL e contém informações gerais sobre o mesmo, tais como a sua identificação, a identificação do autor, a data em que foi escrito, etc.
2. A ENVIRONMENT DIVISION contém instruções ou comandos que descrevem o ambiente físico em que o programa é executado. Esta divisão é usada principalmente para descrever fisicamente os ficheiros usados pelo programa, através da indicação dos seus nomes (internos e externos) e do modo como estão organizados.
3. A DATA DIVISION contém instruções que descrevem os dados usados pelo programa, quer eles existam em ficheiros (o seu *layout*) ou sejam internos do programa (variáveis).
4. A PROCEDURE DIVISION contém as instruções COBOL que o programa irá executar.

Cada divisão de um programa COBOL subdivide-se em unidades mais pequenas denominadas SECTIONS. As secções podem conter parágrafos (os procedimentos) e estes, por sua vez, contém instruções.

```

0          1          2          3          4          5          6          7
12345678901234567890123456789012345678901234567890123456789012
000001 IDENTIFICATION DIVISION.
000002
000003 PROGRAM-ID.          PTNE2550.
000004
000005 AUTHOR.              FMMM.
000006
000007 DATE-WRITTEN.        23/05/96.
000008
000009*****
000010* COMENTARIOS SOBRE O PROGRAMA                                     *
000011*****
000012
000013 ENVIRONMENT DIVISION.
000014
000015 CONFIGURATION SECTION.
000016 SPECIAL-NAMES.
000017     DECIMAL-POINT IS COMMA.
000018
000019 INPUT-OUTPUT SECTION.
000020
000021 FILE-CONTROL.
000022
000023*****
000024*FICHEIROS SEQUENCIAIS (SELECT NOME-INTERNO ASSIGN NOME-EXTERNO)*
000025*****
000026
000027     SELECT  FICH01          ASSIGN          FICH01
000028                                FILE STATUS  STATUS-FICH.
000029
000030     SELECT  FICH02          ASSIGN          FICH02
000031                                FILE STATUS  STATUS-FICH.
000032

```

```

000033      SELECT  FICH03              ASSIGN      FICH03
000034              FILE STATUS          STATUS-FICH.
000035
000036      SELECT  MAPA01              ASSIGN      MAPA01
000037              FILE STATUS          STATUS-FICH.
000038
000039*****
000040* FICHEIRO INDEXADO
000041*****
000042
000043      SELECT  FICH04              ASSIGN      FICH04
000044              ORGANIZATION          INDEXED
000045              ACCESS                DYNAMIC
000046              RECORD KEY            FICH04-CHAVE
000047              FILE STATUS          STATUS-FICH.
000048
000049 DATA DIVISION.
000050*****
000051
000052*****
000053* LAYOUTs DOS FICHEIROS (FICH01, ..., FICH04, MAPA01)
000054*****
000055
000056 FILE SECTION.
000057
000058 FD FICH01.
000059 01 REG-FICH01.
000060     10 COD-BALCAO                PIC 9(04).
000061     10 NUM-BOLETIM                PIC 9(07).
000062     10 ESPECIE                    PIC X(09).
000063     10 CONTA-DO                   PIC 9(14).
000064     10 NOME                       PIC X(40).
000065     10 MORADA                     PIC X(40).
000066     10 COD-POSTAL                 PIC X(30).
000067     10 COD-PAIS                   PIC 9(04).
000068     10 NUMERARIO                  PIC 9(12)V99.
000069     10 QTD-ATRIB                  PIC 9(09).
000070     10 PRECO                      PIC 9(05).
000071
000072 FD FICH02.
000073 COPY FDFICH02.
000074
000075 FD FICH03.
000076 COPY FDFICH03.
000077
000078 FD FICH04
000079 01 REG-FICH04.
000080     05 FICH04-CHAVE.
000081         10 FICH04-DTMOVTO          PIC X(10).
000082         10 FICH04-NCONTIT          PIC 9(14).
000083         10 FICH04-CESPECIE          PIC X(09).
000084     05 FICH04-QTD                  PIC 9(07).
000085
000086 FD MAPA01                      RECORD 132
000087                                LABEL RECORDS STANDARD
000088                                RECORDING MODE IS F.
000089 01 LINHA-MAPA                    PIC  X(132).
000090
000091*****
000092* DECLARAÇÃO E INICIALIZAÇÃO DE VARIÁVEIS
000093* NIVEL 77 : VARIÁVEIS ELEMENTARES (NAO SAO ESTRUTURAS)
000094* NIVEL 88 : FLAGS QUE SE TORNAM VERDADEIRAS QUANDO A VARIÁVEL
000095* DO NIVEL IMEDIATAMENTE ANTERIOR ASSUME O VALOR
000096* ASSOCIADO 'A FLAG, E FALSAS CASO CONTRARIO
000097* OUTROS NIVEIS : ESTRUTURAS DE DADOS
000098*
000099* PICTURE X : VARIÁVEIS ALFANUMERICAS
000100* PICTURE 9 : VARIÁVEIS NUMERICAS
000101* PICTURE Z : VARIÁVEIS DE DISPLAY (O DIGITO APENAS APARECE SE
000102* 'A ESQUERDA HOUVEREM DIGITOS DIFERENTES DE ZERO)
000103* PICTURE S9, -9, -Z, 9-, Z- : VARIÁVEIS COM SINAL
000104* PICTURE 9(i)V9(d) : VARIÁVEIS COM PARTE INTEIRA (i DIGITOS) E
000105* PARTE DECIMAL (d DIGITOS)
000106* PICTURE 9 COMP : VARIÁVEIS COMPACTADAS
000107* (MAIS DE UM DIGITO POR BYTE)
000108*****
000109
000110 WORKING-STORAGE SECTION.
000111
000112 77 WS-ARRED-ESC                PIC 9(13)V.
000113 77 W-MSG                       PIC X(70)          VALUE SPACES.
000114 77 W-PARAGRAFO                 PIC X(30)          VALUE SPACES.
000115 77 W-COMANDO                   PIC X(35)          VALUE SPACES.

```



```

000116 77 W-TIPO-ACESSO          PIC X(30)          VALUE SPACES.
000117 77 W-FICHEIRO             PIC X(08)          VALUE SPACES.
000118 77 W-TABELA               PIC X(08)          VALUE SPACES.
000119 77 W-ROTINA               PIC X(10)          VALUE SPACES.
000120 77 WS-IMP-01              PIC 9(15)          VALUE ZEROS.
000121 77 WS-NUM-PAG             PIC 9(04)          VALUE ZEROS.
000122 77 WS-NUM-LINHAS          PIC 9(04)          VALUE 60.
000123 77 WS-INDICE              PIC 99.
000124 77 WS-LIDOS               PIC 9(15)          VALUE ZEROS.
000125 77 WS-VFINANC             PIC 9(14)V99.
000126 77 WS-POS-CURSOR          PIC 9(03).
000126 77 WS-LINHA-CURSOR        PIC 9(03).
000128 77 WS-RESTO               PIC 9(03).
000129 77 N-DIGITOS-LINK         PIC 99.
000130 77 TITULO                 PIC X(30).
000131 77 CODTAB                 PIC XX.
000132 77 USER-ID-LINK           PIC X(8).
000133 77 W-QNEGOC               PIC S9(09) COMP VALUE ZEROS.
000134 77 W-NUMLINHAS            PIC S9(09) COMP VALUE ZEROS.
000135 77 WS-DUMMY               PIC X.
000136 77 WS-ZTIMESTP            PIC X(26)          VALUE SPACES.
000137 77 W-VPUORDEM-MIN         PIC S9(10)V99 COMP-3.
000139 77 W-VPUORDEM-AVG         PIC S9(10)V99 COMP-3.
000140 77 W-VPUORDEM-MAX         PIC S9(10)V99 COMP-3.
000141 77 W-DATA                 PIC X(10).
000142 77 W-YEAR                 PIC S9(09) COMP.
000143 77 W-MONTH                PIC S9(09) COMP.
000144 77 W-DAY                  PIC S9(09) COMP.
000145 77 W-DT-YEAR              PIC X(10).
000146 77 W-DT-MONTH             PIC X(10).
000147 77 W-DT-DAY               PIC X(10).
000148 77 W-STRING               PIC X(04).
000149 77 W-HOUR                 PIC S9(04) COMP.
000150 77 W-MINUTE               PIC S9(04) COMP.
000151 77 W-SECOND               PIC S9(04) COMP.
000152 77 W-MSECOND              PIC S9(09) COMP.
000153
000154 77 W-SQLCODE              PIC -999          VALUE ZEROS.
000155      88 BOM-IO-DB2          VALUE 0.
000156      88 INEXISTENTE-DB2     VALUE +100 -305.
000157      88 DUPLICADO-DB2        VALUE -803.
000158      88 VARIAS-LINHAS-DB2    VALUE -811.
000159
000160 77 STATUS-FICH              PIC X(02)          VALUE ZEROS.
000161      88 STATUS-FIM            VALUE '10'.
000162      88 STATUS-OK             VALUE '00' '97'.
000163      88 STATUS-DUP            VALUE '22'.
000164      88 STATUS-INEXISTENTE    VALUE '23'.
000165
000166 01 LINHAS-TEXTO.
000167      05 LINHA-TEXTO OCCURS 13 TIMES.
000168          10 NUM-LINHA          PIC 9(2).
000169          10 TEXTO-LINHA        PIC X(78).
000170
000171 01 WS-DATA-DIA                PIC 9(06).
000172 01 FILLER REDEFINES WS-DATA-DIA.
000173      10 WS-ANO                PIC 9(02).
000174      10 WS-MES                 PIC 9(02).
000175      10 WS-DIA                 PIC 9(02).
000176
000177 01 WS-HORA-DIA                 PIC 9(08).
000178 01 FILLER REDEFINES WS-HORA-DIA.
000179      10 WS-HORA                PIC 9(02).
000180      10 WS-MIN                  PIC 9(02).
000181      10 WS-SEG                 PIC 9(02).
000182      10 WS-DSEG                PIC 9(02).
000183
000184***** VARIAVEIS PARA ACESSO AO MODULO PTNEM013 *****
000185 77 WS-PTNEM013                PIC X(08) VALUE 'PTNEM013'.
000186 01 CAREA-PTNEM013.
000187 COPY CAMOD013.
000188
000189*****
000190* AREA DE MAPA *
000191*****
000192
000193 01 BINE1.
000194      05 BIN1                      PIC X(23) VALUE
000195          '*** DJDE JDE=FSL018,END;'.
000196 01 BINE2.
000197      05 BIN2                      PIC X(56) VALUE
000198          '*** DJDE JDE=FSP02,FORMS=AVILAN, FONTS=((2721BP,6.00 LPI)),'.
000199      05 BIN3                      PIC X(19) VALUE
000200          '(XSYSP,6.00 LPI)),;'.

```

```

000201 01 BINE3.
000202     05 BIN4                                PIC X(49) VALUE
000203     '*** DJDE OTEXT=('MONTE IMPRESSO MOD.737',WAIT),;'.
000204 01 BINE4.
000205     05 BIN5                                PIC X(08) VALUE
000206     '*** DJDE '.
000207     05 BIN6                                PIC X(44) VALUE
000208     'OTEXT=('MONTE PAPEL BRANCO',END,WAIT),END;'.
000209
000210 01 MAPA.
000211
000212     05 LINHA00.
000213         10 FILLER                            PIC X(132) VALUE SPACES.
000214
000215     05 LINHA01.
000216         10 FILLER                            PIC X(10) VALUE 'PTNE2550'.
000217         10 FILLER                            PIC X(39) VALUE SPACES.
000218         10 FILLER                            PIC X(66) VALUE
000219         '*** NOME DA INSTITUICAO ***'.
000220         10 FILLER                            PIC X(06) VALUE 'PAG.:'.
000221         10 PAG-CAB                            PIC Z.ZZ9.
000222
000223     05 LINHA02.
000224         10 WS-SEculo                        PIC 9(02) VALUE 19.
000225         10 WDATAH                          PIC X(08) VALUE SPACES.
000226         10 FILLER                          PIC X(01) VALUE SPACES.
000227         10 WTEMPOH                         PIC X(08) VALUE SPACES.
000228         10 FILLER                          PIC X(27) VALUE SPACES.
000229         10 FILLER                          PIC X(69) VALUE
000230         '***** NOME DA APLICACAO *****'.
000231         10 FILLER                          PIC X(06) VALUE 'REF.:'.
000232         10 DATA-REF                        PIC X(10).
000233
000234     05 LINHA03.
000235         10 FILLER                            PIC X(49) VALUE SPACES.
000236         10 FILLER                            PIC X(83) VALUE
000237         '***** NOME DO MAPA *****'.
000238
000239     05 LINHA04.
000240         10 FILLER                            PIC X(06) VALUE 'DEST: '.
000241         10 BALC-DEST                        PIC X(04) VALUE SPACES.
000242         10 FILLER                            PIC X(1).
000243         10 DESC-BALC                        PIC X(30) VALUE SPACES.
000244
000245     05 LINHA05.
000246         10 FILLER                            PIC X(05) VALUE
000247         'BALC'.
000248         10 FILLER                            PIC X(08) VALUE
000249         'BOLETIM'.
000250         10 FILLER                            PIC X(18) VALUE
000251         'CONTA'.
000252         10 FILLER                            PIC X(10) VALUE
000253         'ESPECIE'.
000254         10 FILLER                            PIC X(05) VALUE
000255         'CORR'.
000256         10 FILLER                            PIC X(10) VALUE
000257         'QTDE'.
000258         10 FILLER                            PIC X(14) VALUE
000259         'PRECO'.
000260         10 FILLER                            PIC X(14) VALUE
000261         'CORRETAGEM'.
000262         10 FILLER                            PIC X(14) VALUE
000263         'TAXA BOLSA'.
000264         10 FILLER                            PIC X(14) VALUE
000265         'VALOR TITULOS'.
000266         10 FILLER                            PIC X(14) VALUE
000267         'VLR A DEBITAR'.
000268         10 FILLER                            PIC X(06) VALUE
000269         'N.OPE.'.
000270
000271     05 LINHA06.
000272         10 FILLER                            PIC X(04) VALUE ALL '-'.
000273         10 FILLER                            PIC X VALUE SPACES.
000274         10 FILLER                            PIC X(07) VALUE ALL '-'.
000275         10 FILLER                            PIC X VALUE SPACES.
000276         10 FILLER                            PIC X(17) VALUE ALL '-'.
000277         10 FILLER                            PIC X VALUE SPACES.
000278         10 FILLER                            PIC X(09) VALUE ALL '-'.
000279         10 FILLER                            PIC X VALUE SPACES.
000280         10 FILLER                            PIC X(04) VALUE ALL '-'.
000281         10 FILLER                            PIC X VALUE SPACES.
000282         10 FILLER                            PIC X(09) VALUE ALL '-'.
000283         10 FILLER                            PIC X VALUE SPACES.
000284         10 FILLER                            PIC X(13) VALUE ALL '-'.

```

```

000285      10  FILLER              PIC X          VALUE SPACES.
000286      10  FILLER              PIC X(13)       VALUE ALL '-'.
000287      10  FILLER              PIC X          VALUE SPACES.
000288      10  FILLER              PIC X(13)       VALUE ALL '-'.
000289      10  FILLER              PIC X          VALUE SPACES.
000290      10  FILLER              PIC X(13)       VALUE ALL '-'.
000291      10  FILLER              PIC X          VALUE SPACES.
000292      10  FILLER              PIC X(13)       VALUE ALL '-'.
000293      10  FILLER              PIC X          VALUE SPACES.
000294      10  FILLER              PIC X(05)       VALUE ALL '-'.
000295
000296      05  LINHA-DET.
000297          10  BALCAO-MAP        PIC 9999.
000298          10  FILLER              PIC X          VALUE SPACES.
000299          10  BOLETIM-MAP       PIC ZZZZZZ9.
000300          10  FILLER              PIC X          VALUE SPACES.
000301          10  CONTA-MAP         PIC 9999B999B999999.9.
000302          10  FILLER              PIC X          VALUE SPACES.
000303          10  ESPECIE-MAP       PIC X(09).
000304          10  FILLER              PIC XX         VALUE SPACES.
000305          10  CORRETOR-MAP      PIC 999.
000306          10  FILLER              PIC X          VALUE SPACES.
000307          10  QTDE-MAP         PIC Z.ZZZ.ZZ9 VALUE ZEROS.
000308          10  FILLER              PIC X          VALUE SPACES.
000309          10  PRECO-MAP        PIC ZZ.ZZZ.ZZ9,99 VALUE ZEROS.
000310          10  FILLER              PIC X          VALUE SPACES.
000311          10  CORRETAGEM-MAP   PIC ZZ.ZZZ.ZZ9,99 VALUE ZEROS.
000312          10  FILLER              PIC X          VALUE SPACES.
000313          10  TXBOLSA-MAP     PIC ZZ.ZZZ.ZZ9,99 VALUE ZEROS.
000314          10  FILLER              PIC X          VALUE SPACES.
000315          10  VLR-TIT-MAP      PIC ZZ.ZZZ.ZZ9,99 VALUE ZEROS.
000316          10  FILLER              PIC X          VALUE SPACES.
000317          10  VLR-DEBITADO-MAP PIC ZZ.ZZZ.ZZ9,99 VALUE ZEROS.
000318          10  FILLER              PIC X          VALUE SPACES.
000319          10  NOPER-MAP       PIC ZZZZ9        VALUE ZEROS.
000320
000321      05  LINHA-FIM.
000322          10  FILLER              PIC X(56) VALUE SPACES.
000323          10  FILLER              PIC X(76) VALUE
000324              '*** FIM DE MAPA ***'.
000325
000326  PROCEDURE DIVISION.
000327  *****
000328
000329      .....
000330
000331      PERFORM FIM-PROGRAMA.

```

1.2 Instruções Básicas

1.2.1 IF e PERFORM

Um programa tem de ser capaz de tomar decisões sobre os dados e, em função delas, executar diferentes secções de código. O verbo IF permite alterar o fluxo de um programa em função dos seus dados.

O verbo PERFORM permite estruturar os programas e identificar os seus procedimentos. Ao encontrar o verbo PERFORM, o código a executar será o constante no parágrafo “chamado” e, uma vez executado o parágrafo, o programa executará a instrução seguinte ao PERFORM (caso o parágrafo “chamado” não tenha as instruções GO TO ou STOP RUN). Desta forma, para além de se conseguirem programas mais legíveis, é possível escrever rotinas genéricas que podem ser executadas em diferentes pontos do programa.

```

000001      PERFORM READ-FICH01.
000002      IF STATUS-FIM
000003          PERFORM FIM-PROGRAMA
000004      ELSE
000005          PERFORM TRATA-REG-FICH01
000006      END-IF.

```

1.2.2 PERFORM UNTIL, MOVE e ADD

O PERFORM UNTIL permite que se executem repetidamente secções de código até que se verifique uma condição de paragem (testada antes da execução).

O verbo MOVE é usado para armazenar valores em variáveis.

O verbo ADD é usado para adicionar valores aos já existentes nas variáveis.

```
000001      MOVE 1                      TO WS-INDICE.
000002      PERFORM UNTIL WS-INDICE > 13
000003          MOVE WS-INDICE          TO NUM-LINHA(WS-INDICE)
000004          MOVE SPACES              TO TEXTO-LINHA(WS-INDICE)
000005          ADD 1                    TO WS-INDICE
000006      END-PERFORM.
```

1.2.3 COMPUTE e COMPUTE ROUNDED

O verbo COMPUTE é usado para calcular os resultados de expressões aritméticas e armazená-los em variáveis. As expressões aritméticas, para além de parêntesis, podem conter os operadores + (adição), - (subtracção), * (multiplicação) e / (divisão). A cláusula ROUNDED permite que o resultado obtido na avaliação da expressão seja arredondado de acordo com a definição da variável. Se esta cláusula não for indicada, o resultado será truncado de acordo com a definição da variável.

```
000001      MOVE 0,15                  TO CAMOD013-TCTTAXA.
000002      COMPUTE CAMOD013-VTXCORR = WS-VFINANC * (CAMOD013-TCTTAXA / 100).
000003      COMPUTE WS-ARRED-ESC        ROUNDED = CAMOD013-VTXCORR.
000004      MOVE WS-ARRED-ESC            TO CAMOD013-VTXCORR.
```

1.2.4 DIVIDE e SUBTRACT

Os verbos DIVIDE e SUBTRACT permitem, respectivamente, efectuar as operações de divisão e subtracção.

```
000001      DIVIDE WS-POS-CURSOR BY 80 GIVING WS-LINHA-CURSOR REMAINDER WS-RESTO.
000002      SUBTRACT 4 FROM WS-LINHA-CURSOR.
```

1.2.5 EVALUATE

A instrução EVALUATE, à semelhança do verbo IF, permite alterar o fluxo do programa em função dos seus dados. Ao encontrar o EVALUATE, cada uma das suas condições será avaliada até se encontrar uma que seja verdadeira. Seguidamente, o código associado a essa condição é executado e, posteriormente, é executada a instrução seguinte ao EVALUATE.

```
000001      EVALUATE TRUE
000002      WHEN HELP-LINK-ALFA = 'POST'
000003          MOVE ' TABELA DE CODIGOS POSTAIS' TO TITULO
000004          MOVE '07' TO CODTAB
000005          MOVE 4 TO N-DIGITOS-LINK
000006      WHEN HELP-LINK-ALFA = 'BALC'
000007          MOVE ' TABELA DE BALCOES' TO TITULO
000008          MOVE '04' TO CODTAB
000009          MOVE 4 TO N-DIGITOS-LINK
000010      WHEN HELP-LINK-ALFA = 'PAIS'
000011          MOVE ' TABELA DE PAISES' TO TITULO
000012          MOVE 'F2' TO CODTAB
000013          MOVE 3 TO N-DIGITOS-LINK
000014      WHEN HELP-LINK-ALFA = 'ACTE'
000015          MOVE ' TABELA DE ACTIV. ECONOMICAS' TO TITULO
000016          MOVE '98' TO CODTAB
000017          MOVE 5 TO N-DIGITOS-LINK
000018      WHEN HELP-LINK-ALFA = 'GART'
000019          MOVE ' TABELA DE GARANTIAS' TO TITULO
000020          MOVE 'E5' TO CODTAB
000021          MOVE 6 TO N-DIGITOS-LINK
000022      WHEN HELP-LINK-ALFA = 'SECE' OR 'SECR'
000023          MOVE ' TABELA DE SECTORIZACOES' TO TITULO
000024          MOVE 'F1' TO CODTAB
000025          MOVE 8 TO N-DIGITOS-LINK
```

```
000026      WHEN HELP-LINK-ALFA = 'VINC'
000027          MOVE '      TABELA DE VINCULOS'          TO TITULO
000028          MOVE 'E4'                                TO CODTAB
000029          MOVE 2                                     TO N-DIGITOS-LINK
000030      WHEN HELP-LINK-ALFA = 'CDBP'
000031          MOVE 'CODIGOS DO BANCO DE PORTUGAL'        TO TITULO
000032          MOVE 'TN'                                  TO CODTAB
000033          MOVE 4                                     TO N-DIGITOS-LINK
000034      WHEN OTHER
000035          MOVE ALL '*'                                TO TITULO
000036          MOVE SPACES                                TO CODTAB
000037      END-EVALUATE.
```

Capítulo 2 - Programação BATCH

2.1 Instruções Batch

2.1.1 ACCEPT

O verbo ACCEPT é usado para obter valores de variáveis do ambiente, tais como, a data da máquina (ACCEPT ... FROM DATE), a hora da máquina (ACCEPT ... FROM TIME) e parâmetros para execução do programa (ACCEPT ... FROM SYSIN¹).

```
000001    ACCEPT WS-DATA-DIA FROM DATE.
000002    IF WS-ANO < 50
000003        MOVE 20 TO WS-SEculo
000004    END-IF.
000005    ACCEPT WS-HORA-DIA FROM TIME.
000006    ACCEPT W-DATA FROM SYSIN.
```

2.1.2 STOP RUN

A instrução STOP RUN é usada para terminar a execução do programa.

```
000001 FIM-PROGRAMA.
000002     PERFORM DISPLAYS.
000003     STOP RUN.
```

2.1.3 DISPLAY

O verbo DISPLAY é usado para produzir mensagens relativas à execução do programa.

```
000001 DISPLAYS.
000002     DISPLAY '***** PROGRAMA PTNE2550 *****'.
000003     DISPLAY '* LIDOS NO FICHEIRO - ' WS-LIDOS.
000004     DISPLAY '* IMPRESSOS CORRECTOS - ' WS-IMP-01.
000005     DISPLAY '*****'.
```

2.1.4 CALL

O verbo CALL é usado para, a partir de um programa COBOL, se executarem outros programas COBOL. Ao encontrar o verbo CALL, o programa chamador é suspenso e o programa chamado é executado. Uma vez terminado o programa chamado, o controlo é retornado ao programa chamador e a execução continua na linha seguinte ao verbo CALL. O CALL a um programa pode ser feito de duas formas:

- CALL estático, ou seja, o código do programa chamado é incluído no programa chamador na altura da LINKEDIÇÃO. Deste modo, sempre que o programa chamado for alterado, é necessário recompilar todos os programas que o chamam.

Ex: CALL 'PTNEM013' USING CAREA-PTNEM013.

- CALL Dinâmico, ou seja, o código do programa chamado apenas é obtido durante a execução do programa chamador, pelo que, sempre que o programa chamado for alterado, o programa chamador “apanha” a nova versão.

Ex: CALL WS-PTNEM013 USING CAREA-PTNEM013.

```
000001 OBTEM-PRECARIO.
000002     COMPUTE WS-VFINANC = QTD-ATRIB * PRECO.
000003     INITIALIZE CAREA-PTNEM013.
000004     MOVE WS-VFINANC          TO CAMOD013-VFINANC.
```

¹ Ver JCL - 1.6.5 Programa que escreve num ficheiro e tem um parâmetro de SYSIN

```

000005      MOVE  52                      TO   CAMOD013-COPER.
000006      MOVE  SPACES                  TO   CAMOD013-RETORNO.
000007
000008      CALL  WS-PTNEM013 USING CAREA-PTNEM013.
000009      IF   CA-RETORNO NOT EQUAL SPACES
000010          MOVE 'OBTEM-PRECARIO' TO W-PARAGRAFO
000011          MOVE 'PTNEM013'      TO W-ROTINA
000012          MOVE CA-MSG          TO W-MSG
000013          PERFORM FIM-ERRO-CALL
000014      END-IF.

```

2.1.5 GOBACK

O verbo GOBACK é usado nos programas chamados por CALL para terminar a sua execução e retornar ao programa chamador.

```

000001 IDENTIFICATION DIVISION.
000002 PROGRAM-ID. PTNEM013.
000003
000004 ENVIRONMENT DIVISION.
000005 CONFIGURATION SECTION.
000006 SPECIAL-NAMES.
000007     DECIMAL-POINT IS COMMA.
000008
000009 INPUT-OUTPUT SECTION.
000010
000011 FILE-CONTROL.
000012     .....
000013 DATA DIVISION.
000014
000015 FILE SECTION.
000016     .....
000017 WORKING-STORAGE SECTION.
000018     .....
000019 LINKAGE SECTION.
000020
000021 01 CAREA-PTNEM013.
000022 COPY CAMOD013.
000023
000024 PROCEDURE DIVISION USING CAREA-PTNEM013.
000025     .....
000026     GOBACK.

```

2.2 Ficheiros Sequenciais

Um ficheiro sequencial é um conjunto de registos armazenados em disco de tal forma que, para que um programa COBOL consiga ler um determinado registo, tem obrigatoriamente de ler sequencialmente todos os registos anteriores.

2.2.1 Definição

Ver 1.1 Estrutura de um Programa (INPUT-OUTPUT SECTION e FILE SECTION)

2.2.2 Abertura

Para abrir ficheiros em COBOL usa-se o verbo OPEN, e existem quatro modos de abertura de ficheiros:

- INPUT - O ficheiro é aberto apenas para leitura.
- OUTPUT - O ficheiro é aberto para escrita, sendo previamente eliminados todos os seus registos.
- I-O - O ficheiro é aberto para leitura, mas os registos podem ser rescritos.
- EXTEND - O ficheiro é aberto para escrita de registos após os nele existentes.

```

000001      OPEN INPUT FICH01.
000002      IF NOT STATUS-OK
000003          MOVE 'ABRIR-FICHEIROS ' TO W-PARAGRAFO

```

```

000004      MOVE  'OPEN      '           TO  W-TIPO-ACESSO
000005      MOVE  'FICH01 '           TO  W-FICHEIRO
000006      PERFORM  FIM-ERRO-FICH
000007      END-IF.
000008
000009      OPEN OUTPUT MAPA01.
000010      IF NOT STATUS-OK
000011          MOVE  'ABRIR-FICHEIROS '   TO  W-PARAGRAFO
000012          MOVE  'OPEN      '           TO  W-TIPO-ACESSO
000013          MOVE  'MAPA01 '           TO  W-FICHEIRO
000014          PERFORM  FIM-ERRO-FICH
000015      END-IF.
000016
000017      OPEN I-O FICH02.
000018      IF NOT STATUS-OK
000019          MOVE  'ABRIR-FICHEIROS '   TO  W-PARAGRAFO
000020          MOVE  'OPEN      '           TO  W-TIPO-ACESSO
000021          MOVE  'FICH02 '           TO  W-FICHEIRO
000022          PERFORM  FIM-ERRO-FICH
000023      END-IF.
000024
000025      OPEN EXTEND FICH03.
000026      IF NOT STATUS-OK
000027          MOVE  'ABRIR-FICHEIROS '   TO  W-PARAGRAFO
000028          MOVE  'OPEN      '           TO  W-TIPO-ACESSO
000029          MOVE  'FICH03 '           TO  W-FICHEIRO
000030          PERFORM  FIM-ERRO-FICH
000031      END-IF.

```

2.2.3 Leitura

Para ler ficheiros usa-se o verbo READ.

```

000001      READ FICH01.
000002      IF (NOT STATUS-FIM) AND (NOT STATUS-OK)
000003          MOVE  'LER-FICHEIRO '      TO  W-PARAGRAFO
000004          MOVE  'READ      '          TO  W-TIPO-ACESSO
000005          MOVE  'FICH01 '           TO  W-FICHEIRO
000006          PERFORM  FIM-ERRO-FICH
000007      END-IF.

```

2.2.4 Escrita

Para rescrever o registo corrente de um ficheiro usa-se o verbo REWRITE, e para escrever um novo registo usa-se o verbo WRITE. De referir que, após a escrita, as variáveis associadas à descrição do ficheiro (constantes na FILE-SECTION) podem perder o seu conteúdo. Para contornar este problema, é necessário usar a cláusula FROM do verbo WRITE. Deste modo, o conteúdo do registo deve ser previamente escrito numa estrutura da WORKING-STORAGE com o mesmo *layout* do ficheiro.

```

000001      REWRITE REG-FICH02.
000002      IF NOT STATUS-OK
000003          MOVE  'ESCREVE-FICHEIROS '  TO  W-PARAGRAFO
000004          MOVE  'WRITE      '          TO  W-TIPO-ACESSO
000005          MOVE  'FICH02 '           TO  W-FICHEIRO
000006          PERFORM  FIM-ERRO-FICH
000007      END-IF.
000008
000009      WRITE LINHA-MAPA FROM LINHA-DET.
000010      IF NOT STATUS-OK
000011          MOVE  'ESCREVE-FICHEIROS '  TO  W-PARAGRAFO
000012          MOVE  'WRITE      '          TO  W-TIPO-ACESSO
000013          MOVE  'MAPA01 '           TO  W-FICHEIRO
000014          PERFORM  FIM-ERRO-FICH
000015      END-IF.
000016
000017      WRITE REG-FICH03.
000018      IF NOT STATUS-OK
000019          MOVE  'ESCREVE-FICHEIROS '  TO  W-PARAGRAFO
000020          MOVE  'REWRITE'           TO  W-TIPO-ACESSO
000021          MOVE  'FICH04 '           TO  W-FICHEIRO
000022          PERFORM  FIM-ERRO-FICH
000023      END-IF.

```


2.2.5 *Fecho*

Para fechar ficheiros usa-se o verbo CLOSE.

```
000001      CLOSE FICH01.
000002      IF NOT STATUS-OK
000003          MOVE 'FECHAR-FICHEIROS ' TO W-PARAGRAFO
000004          MOVE 'CLOSE ' TO W-TIPO-ACESSO
000005          MOVE 'FICH01' TO W-FICHEIRO
000006          PERFORM FIM-ERRO-FICH
000007      END-IF.
```

2.3 Ficheiros Indexados

Os ficheiros indexados, para além da zona de dados, contêm um índice de acesso aos dados. Deste modo, se preenchermos a chave correspondente ao índice, podemos aceder directamente ao registo do ficheiro com aquela chave, sem ter necessidade de ler sequencialmente todos os registos anteriores.

2.3.1 *Definição*

Ver 1.1 Estrutura de um Programa (INPUT-OUTPUT SECTION e FILE SECTION)

2.3.2 *Abertura*

Ver 2.2.2 Abertura de Ficheiros Sequenciais

2.3.3 *Posicionamento*

Por vezes é conveniente ler vários registos de um ficheiro indexado, em vez de apenas um. Para tal, é necessário usar o verbo START. Este verbo posiciona-nos (não lê) no primeiro registo do ficheiro que obedece às condições por nós indicadas.

```
000001      MOVE '0001-01-01' TO FICH04-DTMOVTO.
000002      MOVE ZEROS TO FICH04-NCONTIT.
000003      MOVE SPACES TO FICH04-CESPECIE.
000004      START FICH04 KEY IS GREATER THAN FICH04-CHAVE.
```

2.3.4 *Leitura*

Para ler ficheiros indexados usa-se o verbo READ com a cláusula NEXT.

```
000001      READ FICH04 NEXT.
000002      IF (NOT STATUS-FIM) AND (NOT STATUS-OK)
000003          MOVE 'LER-FICHEIRO ' TO W-PARAGRAFO
000004          MOVE 'READ ' TO W-TIPO-ACESSO
000005          MOVE 'FICH04' TO W-FICHEIRO
000006          PERFORM FIM-ERRO-FICH
000007      END-IF.
```

2.3.5 *Escrita*

Ver 2.2.4 Escrita de Ficheiros Sequenciais

2.3.6 *Eliminação de Registos*

Para eliminar o registo corrente de um ficheiro indexado usa-se o verbo DELETE.

```
000001      DELETE FICH04.
000002      IF NOT STATUS-OK
000003          MOVE 'APAGAR-REGISTO ' TO W-PARAGRAFO
000004          MOVE 'DELETE ' TO W-TIPO-ACESSO
000005          MOVE 'FICH04' TO W-FICHEIRO
```

```
000006      PERFORM  FIM-ERRO-FICH
000007      END-IF.
```

2.3.7 Fecho

Ver 2.2.5 Fecho de Ficheiros Sequenciais

2.4 Mapas

A elaboração de mapas faz-se de forma análoga à escrita em ficheiros sequenciais (abertura, escrita e fecho). Contudo existem formulários especiais que requerem a execução de comandos apropriados antes de se começar a escrever num mapa (após o OPEN). Estes comandos podem variar de sistema para sistema, mas não diferirão muito dos exemplos apresentados de seguida:

- Elaboração de um mapa com 160 colunas²

```
000001      WRITE LINHA-MAPA FROM BINE1 AFTER PAGE.
```

- Elaboração de um mapa em papel especial (um modelo previamente definido)

```
000001      WRITE LINHA-MAPA FROM BINE2 AFTER PAGE.
000002      WRITE LINHA-MAPA FROM BINE3 AFTER 1.
000003      WRITE LINHA-MAPA FROM BINE4 AFTER 1.
```

O exemplo seguinte ilustra o processo habitual de elaboração de mapas (neste caso, a partir de registos lidos num ficheiro)

```
000001  PROCESSA-FICHEIRO.
000002  *****
000003
000004      PERFORM LER-FICHEIRO.
000005      PERFORM UNTIL STATUS-FIM
000006          PERFORM TRATA-REGISTO
000007          PERFORM GERAR-MAPA
000008          PERFORM LER-FICHEIRO
000009      END-PERFORM.
000010
000011  GERAR-MAPA.
000012  *****
000013
000014      MOVE NOPER      OF VTN02801  TO NOPER-MAP.
000015      MOVE NCUPAO     OF VTN01501  TO BALCAO-MAP.
000016      MOVE NOPRORIG  OF VTN01501  TO BOLETIM-MAP.
000017      MOVE NCONTIT   OF VTN01501  TO CONTA-MAP.
000018      MOVE CESPECIE   OF VTN01501  TO ESPECIE-MAP.
000019      MOVE CINTERM   OF VTN01501  TO CORRETOR-MAP.
000020      MOVE QNEGOC     OF VTN01501  TO QTDE-MAP.
000021      MOVE VPUOPER   OF VTN01501  TO PRECO-MAP.
000022      MOVE '$'      TO PRECO-MAP(11:1).
000023
000024      MOVE W-CORRETAGEM TO CORRETAGEM-MAP.
000025      MOVE '$'         TO CORRETAGEM-MAP(11:1).
000026      MOVE W-TXBOLSA  TO TXBOLSA-MAP.
000027      MOVE '$'         TO TXBOLSA-MAP(11:1).
000028      MOVE VDESREM OF VTN04701  TO VLR-TIT-MAP.
000029      MOVE '$'         TO VLR-TIT-MAP(11:1).
000030      COMPUTE W-VFINANC =
000031          VFINANC OF VTN01501 - VNUMERAR OF VTN01501 -
000032          W-CORRETAGEM + W-TXBOLSA + VDSFIXA OF VTN04701.
000033      MOVE W-VFINANC    TO VLR-DEBITADO-MAP.
000034      MOVE '$'         TO VLR-DEBITADO-MAP(11:1).
000035
000036      IF WS-NUM-LINHAS > 55
000037          PERFORM IMPRIME-CABECALHO
000038      END-IF.
000039      WRITE LINHA-MAPA      FROM LINHA-DET AFTER 1.
000040      ADD 1                 TO WS-NUM-LINHAS.
000041
000042  IMPRIME-CABECALHO.
000043  *****
000044
000045      ADD 1                 TO WS-NUM-PAG.
000046      MOVE WS-NUM-PAG      TO PAG-CAB.
000047      WRITE LINHA-MAPA     FROM LINHA01 AFTER PAGE.
```

² Considere-se que a variável LINHA-MAPA foi definida com PIC X(160)

```

000048      WRITE LINHA-MAPA          FROM LINHA02 AFTER 1.
000049      WRITE LINHA-MAPA          FROM LINHA03 AFTER 1.
000050      WRITE LINHA-MAPA          FROM LINHA04 AFTER 2.
000051      WRITE LINHA-MAPA          FROM LINHA05 AFTER 2.
000052      WRITE LINHA-MAPA          FROM LINHA06 AFTER 1.
000053      WRITE LINHA-MAPA          FROM LINHA00 AFTER 1.
000054      MOVE 9                      TO WS- NUM-LINHAS.

```

2.5 Variáveis de Sistema

A execução automática de programas Batch conduz-nos frequentemente à necessidade de condicionar a execução de alguns programas em função da execução com sucesso de outros. A variável RETURN-CODE oferece-nos a possibilidade de controlar as saídas dos programas (ver JCL). Usualmente, utilizam-se os seguintes RETURN-CODEs:

RETURN-CODE	SIGNIFICADO
0	Programa terminou correctamente
4	Programa terminou anormalmente, mas os seguintes podem continuar
16	Programa terminou anormalmente, e os seguintes devem parar

2.5.1 RETURN-CODE

```

000001 FIM-ERRO-FICH.
000002      DISPLAY '***** PTNE2550 *****'.
000003      DISPLAY '* ACESSO          ' W-TIPO-ACESSO.
000004      DISPLAY '* FICHEIRO        ' W-FICHEIRO.
000005      DISPLAY '* ERRO            ' STATUS-FICH.
000006      MOVE +16      TO RETURN-CODE.
000007      PERFORM DISPLAYS.
000008      STOP RUN.

```

ANEXO - FILE STATUS

CICS

Capítulo 1 - PROGRAMAÇÃO ONLINE

Existem duas formas básicas³ de executar um programa Online: via LINK⁴ e via START⁵. Para iniciar um programa via START, é necessário que ele tenha uma transacção associada e que os diversos objectos a ele associados estejam recenseados no CICS, nomeadamente:

- O nome do programa
- O nome do ecrã
- O nome do plano DB2
- O nome da transacção (associando-a ao programa e ao plano)

1.1 Estrutura dos Programas

Em termos de estrutura, os programas Online apenas diferem dos Batch por não terem INPUT-OUTPUT SECTION nem FILE SECTION, uma vez que, os ficheiros têm de ser recenseados no CICS e, como tal, este já conhece as suas características.

1.1.1 Programa chamado via LINK

Um programa chamado via LINK (Ver 1.2.4 LINK) suspende o programa chamador até que acabe de ser executado. Para terminar e devolver o controlo ao programa chamador, deve usar o comando RETURN (Ver 1.2.6 RETURN).

```

0          1          2          3          4          5          6          7
12345678901234567890123456789012345678901234567890123456789012
000001 IDENTIFICATION DIVISION.
000002 PROGRAM-ID. PTNUM013.
000003
000004 ENVIRONMENT DIVISION.
000005 CONFIGURATION SECTION.
000006 SPECIAL-NAMES.
000007     DECIMAL-POINT IS COMMA.
000008
000009 DATA DIVISION.
000010
000011 WORKING-STORAGE SECTION.
000012     .....
000013 LINKAGE SECTION.
000014
000015 01 DFHCOMMAREA.
000016 COPY CAMOD013.
000017
000018 PROCEDURE DIVISION.
000019     .....
000020     EXEC CICS RETURN  END-EXEC.
```

1.1.2 Programa chamado via START

Um programa chamado via START não tem uma execução sequencial e, como tal, tem de distinguir em que fase da sua execução é que se encontra. Assim sendo, é necessária uma área de

³ Ver 1.1 Estrutura dos Programas

⁴ Ver 1.2.4 LINK

⁵ Ver 1.2.5 START

comunicação onde se guardem todas as informações necessárias para detectar e tratar em conformidade as diversas fases do programa.

Várias situações diferentes podem ocorrer:

1. O programa está a começar pela primeira vez e tem de mostrar o ecrã “limpo” ao utilizador para que este o preencha. Nesta situação, o programa geralmente recebe a área de comunicação do programa que o “iniciou” (Ver 1.2.12 RETRIEVE), limpa as variáveis do ecrã, envia-o e retorna a ele próprio (Ver 1.2.1 SEND).
2. O programa vai executar tarefas em função dos dados digitados pelo utilizador. Nesta situação, o programa já tem a área de comunicação em DFHCOMMAREA, limpa as variáveis do ecrã, executa os comandos HANDLE AID (Ver 1.2.7 HANDLE AID), HANDLE CONDITION (Ver 1.2.10 HANDLE CONDITION) e/ou IGNORE CONDITION (Ver 1.2.11 IGNORE CONDITION) para posicionar as respectivas *flags*, recebe o ecrã (Ver 1.2.2 RECEIVE) e trata os dados.
3. O programa terminou e enviou uma mensagem (operação efectuada, erro ocorrido, etc.) ao utilizador, e este voltou a reiniciá-lo (quer fazer uma nova operação). Nesta situação, o programa já tem a área de comunicação em DFHCOMMAREA, limpa as variáveis do ecrã, envia-o e retorna a ele próprio (Ver 1.2.1 SEND).
4. O utilizador, por exemplo, carregou numa tecla de HELP e, portanto, o programa tem de lhe mostrar o HELP. Nesta situação, o programa comporta-se como na situação 2, mas depois tem geralmente de fazer START (Ver 1.2.5 START) ao outro programa (neste caso, o programa de HELP), guardando antes o estado actual na área de comunicação.
5. O utilizador, por exemplo, volta do HELP. Neste caso o programa comporta-se como na situação 1, à excepção do facto de, em vez de enviar o ecrã vazio, ter de enviar o ecrã tal como estava antes de ter sido chamado o HELP.
6. O utilizador carregou numa tecla para voltar à opção anterior. Nesta situação, o programa comporta-se como na situação 2, à excepção do facto de, em vez de tratar dados, ter de fazer START (Ver 1.2.5 START) a outro programa (neste caso, o programa que o tinha iniciado).
7. Etc.

```

0          1          2          3          4          5          6          7
123456789012345678901234567890123456789012345678901234567890123456789012
000001 IDENTIFICATION DIVISION.
000002 PROGRAM-ID. PTNU255A.
000003
000004 ENVIRONMENT DIVISION.
000005 CONFIGURATION SECTION.
000006 SPECIAL-NAMES.
000007     DECIMAL-POINT IS COMMA.
000008
000009 DATA DIVISION.
000010
000011 WORKING-STORAGE SECTION.
000012
000013 77 LINHA-CORRENTE          PIC S9(4) COMP.
000014 77 LINHA-CURSOR          PIC S9(4) COMP.
000015 77 WS-RESP               PIC S9(08) COMP VALUE +0.
000016 77 WS-LEN                PIC S9(04) COMP VALUE +7260.
000017 77 WS-NLINHA             PIC 99          VALUE +10.
000018 77 NUMERO                PIC 9(8)        VALUE ZEROS.
000019 77 W-TIPO-ACESSO         PIC X(30)       VALUE SPACES.
000020 77 W-FICHEIRO            PIC X(08)       VALUE SPACES.
000021 77 W-TABELA             PIC X(08)       VALUE SPACES.
000022 77 W-ROTINA             PIC X(10)       VALUE SPACES.
000023 77 WS-TRANS             PIC X(04)       VALUE SPACES.
```

```

000024 77 WS-PRINTER          PIC X(04)      VALUE SPACES.
000025 77 WS-NUM-PAG          PIC 999        VALUE ZEROS.
000026 77 W-SQLCODE           PIC -999       VALUE ZEROS.
000027      88 BOM-IO-DB2      VALUE 0.
000028      88 INEXISTENTE-DB2 VALUE +100 -305.
000029      88 DUPLICADO-DB2    VALUE -803.
000030      88 VARIAS-LINHAS-DB2 VALUE -811.
000031 77 W-EIBRESP           PIC 99.
000032      88 EIB-ENDFILE      VALUE 20.
000033      88 EIB-OK           VALUE 00.
000034      88 EIB-NOTFND       VALUE 13.
000035
000036 01 TS-AREA.
000037      05 TS-NUM-ITEM        PIC S9(04) COMP.
000038      05 TS-TAMAREA        PIC S9(04) COMP VALUE +79.
000039      05 TS-TSNAME.
000040          10 FILLER          PIC X(04)      VALUE 'TNF2'.
000041          10 TS-TERMID       PIC X(04)      VALUE SPACES.
000042      05 REG-TS            PIC X(79).
000043
000044 01 REG-FICH04.
000045      05 FICH04-CHAVE.
000046          10 FICH04-DTMOVTO PIC X(10).
000047          10 FICH04-NCONTIT  PIC 9(14).
000048          10 FICH04-CESPECIE  PIC X(09).
000049      05 FICH04-QTD         PIC 9(07).
000050
000051 01 WS-PAGINA.
000052      05 FILLER OCCURS 55 TIMES.
000053          10 WS-TEXTO       PIC X(79).
000054          10 FILLER          PIC X(53).
000055
000056 01 AREA-TRATAMENTO-DATA-HORA-CICS.
000057      02 W-ABSTIME           PIC S9(15) COMP-3.
000058      02 W-YYMMDD.
000059          05 ANO-CURR        PIC 99.
000060          05 FILLER          PIC X.
000061          05 MES-CURR        PIC 99.
000062          05 FILLER          PIC X.
000063          05 DIA-CURR        PIC 99.
000064      02 W-TIME              PIC X(8).
000065      02 R-TIME REDEFINES W-TIME.
000066          05 TIME-HH         PIC 99.
000067          05 FILLER          PIC X.
000068          05 TIME-MM         PIC 99.
000069          05 FILLER          PIC X.
000070          05 TIME-SS         PIC 99.
000071      02 TIME-HHMMSS.
000072          05 TIME-HH         PIC 99.
000073          05 TIME-MM         PIC 99.
000074          05 TIME-SS         PIC 99.
000075      02 R-TIME-HHMMSS REDEFINES TIME-HHMMSS
000076          PIC 9(6).
000077      02 W-YYYYMMDD.
000078          05 SEC-CURR        PIC 99.
000079          05 ANO-CURR        PIC 99.
000080          05 MES-CURR        PIC 99.
000081          05 DIA-CURR        PIC 99.
000082      02 R-YYYYMMDD REDEFINES W-YYYYMMDD
000083          PIC 9(8).
000084      02 W-DATA-FORM.
000085          05 SEC-CURR        PIC 99.
000086          05 ANO-CURR        PIC 99.
000087          05 FILLER          PIC X VALUE '/'.
000088          05 MES-CURR        PIC 99.
000089          05 FILLER          PIC X VALUE '/'.
000090          05 DIA-CURR        PIC 99.
000091
000092 01 AREA-LIGACAO.
000093      05 SITUACAO-LINK        PIC X.
000094      05 TOTAL-LINHAS-LINK    PIC S9(4) COMP.
000095      05 LINHA-INICIAL-LINK  PIC S9(4) COMP.
000096      05 COMPRI-LINK         PIC S9(4) COMP.
000097      05 DATA-GLOBAL-LINK  PIC X(10).
000098      05 BANCO-LINK          PIC X(06).
000099      05 TITULO-LINK         PIC X(30).
000100      05 CURS-LINK           PIC S9(4) COMP.
000101      05 TIPO-MAPA-LINK      PIC X(04).
000102      05 SITUACAO-LINK-ANT  PIC X.
000103      05 TRANS-LINK          PIC X(04).
000104      05 ITEMH-LINK          PIC S9(4) COMP.
000105      05 DABRV-LINK          PIC X(15).
000106      05 DCOMP-LINK          PIC X(30).
000107      05 HELP-LINK           PIC S9(4) COMP.

```

```

000108      05 NCOD-LINK          PIC 9(10).
000109      05 PAGINA-LINK        PIC S9(4) COMP.
000110      05 ECRA-LINK          PIC X(1109).
000111
000112 COPY DFHAID.
000113 COPY DFHBMSCA.
000114
000115 LINKAGE SECTION.
000116
000117 01 DFHCOMMAREA              PIC X(1134).
000118
000119 PROCEDURE DIVISION.
000120
000121      MOVE LOW-VALUES TO ECRAO.
000122      MOVE EIBTRMID   TO TS-TERMID.
000123      MOVE +1134     TO COMPRI-LINK.
000124      IF EIBCALEN = ZEROS
000125          EXEC CICS IGNORE CONDITION LENGERR END-EXEC
000126          EXEC CICS RETRIEVE
000127              INTO (AREA-LIGACAO)
000128              LENGTH(COMPRI-LINK)
000129          END-EXEC
000130      ELSE
000131          MOVE DFHCOMMAREA TO AREA-LIGACAO
000132      END-IF.
000133      .....

```

1.2 Instruções Básicas

1.2.1 SEND

É usado para enviar ecrãs e texto para terminais.

```

EXEC CICS SEND Map()
    << FROM() > < Length() > < Dataonly > | MAPOnly >
    < MAPSet() >
    < FMhparm() >
    < Reqid() >
    < LdC() | < ACTpartn() > < Outpartn() > >
    < MSr() >
    < Cursor() >
    < Set() | PAging | Terminal < Wait > < LAsT > >
    < PRint >
    < FREkb >
    < ALArm >
    < L40 | L64 | L80 | Honeom >
    < NLeom >
    < ERASE < Default | ALternate > | ERASEup >
    < ACCum >
    < FRSet >
    < NOflush >
    < FOrmfeed >
END-EXEC.

```

```

000001 SEND-GRELHA-POS.
000002
000003      EXEC SQL INCLUDE BTAP0002 END-EXEC.
000004
000005      MOVE W-TIME          TO HORA2550.
000006      MOVE DATA-GLOBAL-LINK TO DATA2550.
000007      MOVE EIBTRMID       TO ITER2550.
000008      MOVE BANCO-LINK     TO BANC2550.
000009      MOVE TITULO-LINK    TO DESC2550.
000010      MOVE CURS-LINK      TO EIBCPOSN.
000011
000012      EXEC CICS SEND      MAP('TNG255')
000013                      FROM(TNG2550)
000014                      MAPSET('ETNA255')
000015                      FRSET
000016                      ERASE
000017                      FREEKB
000018                      CURSOR(EIBCPOSN)
000019      END-EXEC.
000020
000021      EXEC CICS RETURN TRANSID('TNT1')
000022                      COMMAREA(AREA-LIGACAO)
000023                      LENGTH(COMPRI-LINK)
000024      END-EXEC.

000001 P005-MOVE-GRELHA.
000002

```



```

000003      IF TIPO-MAPA-LINK EQUAL 'TODO'
000004          MOVE 'DADO'                TO TIPO-MAPA-LINK
000005          MOVE EIBTRMID                TO TERM6340
000006          MOVE TITULO-LINK            TO CAB6340
000007          EXEC CICS SEND              MAP('TNG634')
000008                                      MAPSET('ETNA634')
000009                                      ERASE
000010                                      CURSOR
000011          END-EXEC
000012      ELSE
000013          EXEC CICS SEND              MAP('TNG634')
000014                                      MAPSET('ETNA634')
000015                                      DATAONLY
000016                                      CURSOR
000017          END-EXEC
000018      END-IF.
000019
000020 P010-SEND-GRELHA.
000021
000022      PERFORM P005-MOVE-GRELHA.
000023
000024      EXEC CICS RETURN TRANSID ('TNW7')
000025                      COMMAREA (AREA-LIGACAO)
000026                      LENGTH (COMPRI-LINK)
000027      END-EXEC.

EXEC CICS SEND PAGE
    << TRANSid() > RELease | RETain >
    < TRAILer() >
    < FMhparm() >
    < Set() >
    < Noautopage | AUtopage < Current | ALl > >
    < Operpurge >
    < Last >
END-EXEC.

000001 P200-SEND-PAGE.
000002
000003      MOVE 'AGUARDE. MOVIMENTOS A SEREM PESQUISADOS.'
000004                      TO MSG6340.
000005      PERFORM P005-MOVE-GRELHA.
000006      EXEC CICS SEND PAGE END-EXEC.
000007      MOVE SPACES
000008                      TO MSG6340.

EXEC CICS SEND Text
    FROM()
    < Length() >
    < FMhparm() >
    < Regid() >
    < Cursor() >
    < LDc() | < ACTpartn() > < Outpartn() > >
    < MSr() >
    < Set() | PAging | TERminal < Wait > < LAsT > >
    < PRint >
    < FREekb >
    < ALArm >
    < L40 | L64 | L80 | HOneom >
    < Erase < Default | ALTernate > >
    < NLeom >
    < NOedit < MApped > | ACCum << JUSFirst | JUSLast | JUSTify() >
    < HEder() > < TRailer() > > >
    < FOrmfeed >
END-EXEC.

000001 INICIO.
000002
000003      PERFORM OBTEM-DADOS.
000004
000005      IF INEXISTENTE-DB2
000006          MOVE SPACES                TO WS-PAGINA
000007          MOVE 'NENHUMA ORDEM FOI ENVIADA' TO WS-TEXTO (20)
000008
000009          EXEC CICS SEND TEXT FROM (WS-PAGINA)
000010                      PRINT HONEOM
000011          END-EXEC
000012          EXEC CICS SEND PAGE          END-EXEC
000013          EXEC CICS PURGE MESSAGE END-EXEC
000014          EXEC CICS RETURN END-EXEC
000015      END-IF.

```

1.2.2 RECEIVE

É usado para receber ecrãs.

```
EXEC CICS RECeive Map()
  < Set() | INTO() >
  < Mapset() >
  < From() < Length() > | Terminal < Asis > < INPartn() > >
END-EXEC.

000001 RECEIVE-GRELHA.
000002
000003     EXEC CICS RECEIVE MAP('TNG255')
000004             INTO(TNG255I)
000005             MAPSET('ETNA255')
000006     END-EXEC.
```

1.2.3 SYNCPOINT e ROLLBACK

São usados, respectivamente, para tornar permanentes as alterações efectuadas, e para desfazer as alterações efectuadas após o último SYNCPOINT.

```
000001     EXEC CICS SYNCPOINT END-EXEC.

000001     EXEC CICS SYNCPOINT ROLLBACK END-EXEC.
```

1.2.4 LINK

O comando LINK é usado para, a partir de um programa COBOL, se executarem outros programas COBOL. Ao encontrar o comando LINK, o programa chamador é suspenso e o programa chamado é executado. Uma vez terminado o programa chamado, o controlo é retornado ao programa chamador e a execução continua na linha seguinte ao comando LINK.

```
EXEC CICS Link
  Program()
  < Commarea() < Length() > < Datalength() > >
  < SYSid() >
  < SYNconreturn >
  < Transid() >
  < INPUTMSG() < INPUTMSGLen() > >
END-EXEC.

000001 CHAMA-MODULO-PTNUM031.
000002
000003     EXEC CICS LINK PROGRAM ('PTNUM031')
000004             COMMAREA (VTN01501)
000005     END-EXEC.
000006     IF CTERMID OF VTN01501 EQUAL 'XXXX'
000007         MOVE 'Z' TO SITUACAO-LINK
000008         MOVE VTN01501 TO MSG444AO
000009         PERFORM ENVIAR-TELA
000010     END-IF.
```

1.2.5 START

O comando START é usado para, a partir de um programa COBOL, se iniciarem outros programas COBOL. Ao encontrar o comando START, um novo programa passará a ser executado, pelo que, o programa inicial deverá terminar imediatamente após o START.

```
EXEC CICS START
  TRansid()
  < Interval( +0000000 ) | TIme() | ( After | AT ) < Hours() > < Minutes() >
  < SEconds() > >
  < FFrom() < Length() < FMh > > >
  < TErmid() | Userid() >
  < SYSid() >
  < RTRansid() >
  < RTErmid() >
  < Queue() >
  < Nocheck >
  < Protect >
```

```

    < REqid() >
END-EXEC.

000001 CHAMA-HELP.
000002
000003     MOVE SITUACAO-LINK             TO SITUACAO-LINK-ANT.
000004     MOVE 'M'                     TO SITUACAO-LINK.
000005     MOVE 'TNT1'                   TO TRANS-LINK.
000006     MOVE EIBCPOSN                 TO CURS-LINK.
000007     MOVE TNG255I                   TO ECRA-LINK.
000008     MOVE 'TN02'                   TO WS-TRANS.
000009     MOVE ZEROS                     TO ITEMH-LINK.
000010     MOVE SPACES                   TO DABRV-LINK.
000011     MOVE SPACES                   TO DCOMP-LINK.
000012     EVALUATE TRUE
000013     WHEN EIBCPOSN = 500 OR 501
000014         MOVE 231                     TO HELP-LINK
000015         MOVE CCOR255I                 TO NCOD-LINK
000016     WHEN EIBCPOSN = 550 OR 551 OR 552
000017         MOVE 232                     TO HELP-LINK
000018         MOVE ELIQ255I                 TO NCOD-LINK
000019     WHEN OTHER
000020         MOVE SITUACAO-LINK-ANT        TO SITUACAO-LINK
000021         MOVE 'HELP INACTIVO NESTA POSICAO' TO MSG2550
000022         PERFORM SEND-GRELHA-POS
000023     END-EVALUATE.
000024
000025     EXEC CICS START TRANSID(WS-TRANS)
000026             TERMIN(EIBTRMID)
000027             FROM(AREA-LIGACAO)
000028             LENGTH(COMPRI-LINK)
000029     END-EXEC.
000030     EXEC CICS RETURN
000031     END-EXEC.

000001 INICIA-IMPRESSAO.
000002
000003     EXEC CICS INQUIRE TERMINAL (WS-PRINTER)
000004     END-EXEC.
000005     MOVE EIBRESP                       TO W-EIBRESP.
000006     IF NOT EIB-OK
000007         MOVE 'E'                       TO SITUACAO-LINK
000008         MOVE 'NAO HA IMPRESSORA DISPONIVEL, PRIMA <ENTER>.'
000009                                         TO MSG10
000010         PERFORM P010-SEND-GRELHA
000011     END-IF.
000012
000013     EXEC CICS START TRANSID ('TNL2')
000014             TERMIN (WS-PRINTER)
000015     END-EXEC.
000016     MOVE EIBRESP                       TO W-EIBRESP.
000017     IF EIB-OK
000018         MOVE 'I'                       TO SITUACAO-LINK
000019         MOVE 'IMPRESSAO INICIADA COM SUCESSO, PRIMA <ENTER>.'
000020                                         TO MSG10
000021     ELSE
000022         MOVE 'E'                       TO SITUACAO-LINK
000023         MOVE 'OCORREU UM ERRO NA IMPRESSAO, PRIMA <ENTER>.'
000024                                         TO MSG10
000025     END-IF.
000026     PERFORM P010-SEND-GRELHA.

```

1.2.6 RETURN

É usado para terminar a execução de programas. Ao construir um programa deve ser levado em conta o facto de este comando tornar permanentes todas as eventuais alterações que tenham sido feitas a dados.

```

EXEC CICS RETURN
    < Transid() < Commarea() < Length() > > < Immediate > >
    < INPUTMSG() < INPUTMSGLen() > >
END-EXEC.

000001 P010-SEND-GRELHA.
000002
000003     PERFORM P005-MOVE-GRELHA.
000004
000005     EXEC CICS RETURN TRANSID ('TNW7')
000006             COMMAREA (AREA-LIGACAO)
000007             LENGTH (COMPRI-LINK)

```

```

000008      END-EXEC.

000001 VOLTA-ANTERIOR.
000002
000003      MOVE AREA-PTNUM045    TO DFHCOMMAREA.
000004      EXEC CICS RETURN  END-EXEC.

```

1.2.7 HANDLE AID

É usado para associar procedimentos a teclas. De referir, no entanto, que o procedimento apenas é chamado após a recepção do ecrã.

```

EXEC CICS Handle Aid
  < Anykey() >
  < CLear() >
  < CLRpartn() >
  < Enter() >
  < Lightpen() >
  < Operid() >
  < Trigger() >
  < PA1() >
  < PA2() >
  < PA3() >
  < PF1() >
  < PF2() >
  < PF3() >
  < PF4() >
  < PF5() >
  < PF6() >
  < PF7() >
  < PF8() >
  < PF9() >
  < PF10() >
  < PF11() >
  < PF12() >
  < Enter() >
  < Lightpen() >
  < PF13() >
  < PF14() >
  < PF15() >
  < PF16() >
  < PF17() >
  < PF18() >
  < PF19() >
  < PF20() >
  < PF21() >
  < PF22() >
  < PF23() >
  < PF24() >
END-EXEC.

000001 PROCEDURE DIVISION.
000002      EXEC CICS HANDLE AID
000003              PF1      (CHAMA-HELP)
000004              PF3      (VOLTA-PROG)
000005              PF7      (SAIDA-PF7)
000006              PF8      (SAIDA-PF8)
000007      END-EXEC.
000008      .....
000009 SAIDA-PF7.
000010
000011      IF PAGINA-LINK = 1
000012          MOVE 'NAO EXISTE PAGINA ANTERIOR' TO MSG634O
000013      ELSE
000014          COMPUTE PAGINA-LINK = PAGINA-LINK - 1
000015          COMPUTE LINHA-INICIAL-LINK = ((PAGINA-LINK - 1) * 11) + 1
000016
000017          PERFORM P350-LIMPA-GRELHA
000018          PERFORM P410-LER-TS
000019          MOVE SPACES TO MSG634O
000020      END-IF.
000021      MOVE -1 TO FUND634L.
000022      PERFORM P010-SEND-GRELHA.
000023*
000024 SAIDA-PF8.
000025
000026      IF LINHA-INICIAL-LINK >= TOTAL-LINHAS-LINK
000027          MOVE 'NAO EXISTE PAGINA POSTERIOR' TO MSG634O
000028      ELSE

```

```

000029      COMPUTE PAGINA-LINK = PAGINA-LINK + 1
000030      PERFORM P350-LIMPA-GRELHA
000031      PERFORM P410-LER-TS
000032      MOVE SPACES                TO MSG6340
000033      END-IF.
000034      MOVE -1                      TO FUND634L.
000035      PERFORM P010-SEND-GRELHA.

```

1.2.8 ASKTIME

É usado para obter a data e a hora do sistema.

```

000001      EXEC CICS ASKTIME
000002          ABSTIME(W-ABSTIME)
000003      END-EXEC.

```

1.2.9 FORMATTIME

É usado para formatar a data e a hora do sistema.

```

EXEC CICS Formattime
  Abstime()
  < YYDDD() >
  < YYMdd() >
  < YYDDMm() >
  < DDMYY() >
  < MMDDYY() >
  < YYYYDDD() >
  < YYYYMdd() >
  < YYYYDDMm() >
  < DDMYYYY() >
  < MMDDYYYY() >
  < DATE() >
  < DATEForm() >
  < DATESep() >
  < DAYCount() >
  < DAYOFWeek() >
  < DAYOFMonth() >
  < MOnthofyear() >
  < YEar() >
  < TIME() < TIMESEP() > >
END-EXEC.

000001  FORMATAR-HORA.
000002
000003      EXEC CICS FORMATTIME
000004          ABSTIME(W-ABSTIME)
000005          DATESEP(' / ')
000006          YYMMDD(W-YYMMDD)
000007          TIME(W-TIME)
000008          TIMESEP
000009      END-EXEC.
000010      MOVE CORR W-YYMMDD TO W-YYYYMMDD.
000011      IF ANO-CURR OF W-YYYYMMDD > 90
000012          MOVE 19 TO SEC-CURR OF W-YYYYMMDD
000013      ELSE
000014          MOVE 20 TO SEC-CURR OF W-YYYYMMDD
000015      END-IF.
000016      MOVE CORR W-YYYYMMDD TO W-DATA-FORM.
000017      MOVE CORR R-TIME TO TIME-HHMMSS.
000018      MOVE W-TIME TO WHORA.
000019      MOVE W-DATA-FORM TO WDATA.

```

1.2.10 HANDLE CONDITION

É usado para associar procedimentos a códigos de retorno do CICS.

```

EXEC CICS Handle Condition
  < Allocerr >
  < CBiderr >
  < Disabled >
  < DSNnotfound >
  < DSStat >
  < DUPKey >
  < DUPRec >
  < END >
  < ENDData >
  < ENDFile >
  < ENDInpt >

```

< ENQbusy >
< ENVdeferr >
< EOC >
< EODs >
< EOF >
< ERror >
< ENDData >
< ENDFile >
< ENDInpt >
< ENQbusy >
< ENVdeferr >
< EXpired >
< Filenotfound >
< FUncerr >
< IGREQCd >
< IGREQId >
< ILlogic >
< INBfmh >
< INVERrterm >
< INVEXitreq >
< INVLdc >
< INVMpsz >
< INVPARTN >
< INVPARTNSet >
< INVReq >
< INVTsreq >
< IOerr >
< IScinvreq >
< ITemerr >
< Jiderr >
< LEgerr >
< LOading >
< MApfail >
< MOdeliderr >
< NEtnameiderr >
< NODeiderr >
< NOJbufsp >
< NONval >
< NOPASSBKRD >
< NOPASSBKWr >
< NOSPAce >
< NOSPOol >
< NOSTArt >
< NOSTG >
< NOTAlloc >
< NOTAUth >
< NOTFnd >
< NOTOpen >
< OPeerr >
< OUtdescrerr >
< OVerflow >
< PARTNEriderr >
< PARTNFail >
< PGmiderr >
< PRofileiderr >
< QBusy >
< QIderr >
< QZero >
< RDatt >
< RESiderr >
< RETpage >
< ROLledback >
< RTEFail >
< RTESome >
< SELnerr >
< SESSBusy >
< SESSIonerr >
< Signal >
< SPOLBusy >
< SPOLerr >
< STrelerr >
< Suppressed >
< SYSBusy >
< SYSIderr >
< TAskiderr >
< TCiderr >
< TERMerr >
< TERMIiderr >
< TRansiderr >
< TSioerr >
< UNexpin >
< USeriderr >
< Voliderr >
< Wrbrk >

```
END-EXEC.
```

```
000001 PROCEDURE DIVISION.
000002     EXEC CICS HANDLE CONDITION
000003             ERROR(SAIDA-ERRO)
000004     END-EXEC.
```

1.2.11 IGNORE CONDITION

É usado para ignorar códigos de retorno do CICS, quando o programa prevê algum tratamento para as situações em causa.

```
EXEC CICS Ignore COndition
      < as mesmas condições de HANDLE CONDITION >
END-EXEC.
```

```
000001 PROCEDURE-DIVISION.
000002
000003     EXEC CICS IGNORE CONDITION MAPFAIL
000004                                 QIDERR
000005                                 ITEMERR
000006                                 NOTOPEN
000007                                 INVREQ
000008                                 NOTFND
000009                                 ENDFILE
000010                                 LENGERR
000011     END-EXEC.
```

1.2.12 RETRIEVE

É usado para obter a área de comunicação passada pelo programa anterior, aquando do comando START.

```
EXEC CICS RETRIEve
      < Set() | Into() >
      < Length() >
      < RTRansid() >
      < RTRmid() >
      < Queue() >
      < Wait >
END-EXEC.
```

```
000001 PROCEDURE DIVISION.
000002
000003     MOVE LOW-VALUES TO ECRAO.
000004     MOVE +1134      TO COMPRI-LINK.
000005     IF EIBCALEN = ZEROS
000006         EXEC CICS RETRIEVE
000007                 INTO (AREA-LIGACAO)
000008                 LENGTH(COMPRI-LINK)
000009         END-EXEC
000010     ELSE
000011         MOVE DFHCOMMAREA TO AREA-LIGACAO
000012     END-IF.
```

1.3 Ficheiros Indexados

Os ficheiros indexados, para além da zona de dados, contêm um índice de acesso aos dados. Deste modo, se preencheremos a chave correspondente ao índice, podemos aceder directamente ao registo do ficheiro com aquela chave, sem ter necessidade de ler sequencialmente todos os registos anteriores.

1.3.1 Definição

Os ficheiros indexados são criados no BATCH (Ver JCL - 1.5.8 Eliminação e realocação de um ficheiro indexado) e têm de ser definidos no CICS (pela equipa responsável). Para definir um ficheiro indexado, é usual fornecerem-se várias indicações, tais como:

- O nome lógico do ficheiro (a usar pelos programas)
- O nome físico do DATA SET
- O tamanho da chave
- O tamanho do registo
- Os tipos de acessos (Read, Update, Delete, Browse, Add)
- O tipo de LOG (se necessário "Rollback" ou não)

1.3.2 Abertura

Os ficheiros indexados são normalmente abertos no BATCH (Ver JCL - 1.5.12 Abertura de ficheiro no CICS), mas também o podem ser através de comandos Online (Ver 2.1.5 CEMT SET FILE).

1.3.3 Posicionamento

Por vezes é conveniente ler vários registos de um ficheiro indexado, em vez de apenas um. Para tal, é necessário usar o comando STARTBR FILE. Este comando posiciona-nos (não lê) no primeiro registo do ficheiro que obedece às condições por nós indicadas.

```
000001 INICIAR-PESQUISA.
000002
000003     EXEC CICS STARTBR FILE ('FICH04')
000004                     RIDFLD (FICH04-CHAVE)
000005                     KEYLENGTH (+10)
000006                     GENERIC
000007                     REQID (0)
000008                     GTEQ
000009     END-EXEC.
000010
000011     MOVE EIBRESP          TO W-EIBRESP.
000012     IF (NOT EIB-OK) AND (NOT EIB-NOTFND)
000013         MOVE 'STARTBR'    TO W-TIPO-ACESSO
000014         MOVE 'FICH04'     TO W-TABELA
000015         PERFORM SAIDA-ERRO-VSAM
000016     END-IF.
```

1.3.4 Leitura

Para ler ficheiros indexados usa-se o comando READ FILE ou READNEXT FILE (se tivermos usado STARTBR FILE).

```
000001 P100-READ-FICH04.
000002
000003     EXEC CICS
000004         READ FILE ('FICH04')
000005         INTO (REG-FICH04)
000006         RIDFLD (FICH04-CHAVE)
000007         GTEQ
000008     END-EXEC.
000009
000010     MOVE EIBRESP          TO W-EIBRESP.
000011     IF (NOT EIB-OK) AND (NOT EIB-NOTFND)
000012         MOVE 'READ'      TO W-TIPO-ACESSO
000013         MOVE 'FICH04'    TO W-TABELA
000014         PERFORM SAIDA-ERRO-VSAM
000015     END-IF.

000001 CONSULTA-FICH04.
000002
000003     EXEC CICS IGNORE CONDITION ERROR END-EXEC.
000004     EXEC CICS READ FILE('FICH04')
000005         INTO (REG-FICH04)
000006         RIDFLD (FICH04-CHAVE)
000007         KEYLENGTH(+33)
000008         EQUAL
000009     END-EXEC.
000010
000011     MOVE EIBRESP          TO W-EIBRESP.
000012     IF (NOT EIB-OK) AND (NOT EIB-NOTFND)
```



```

000013      MOVE 'READ'          TO W-TIPO-ACESSO
000004      MOVE 'FICH04'        TO W-TABELA
000015      PERFORM SAIDA-ERRO-VSAM
000016      END-IF.

000001 READNEXT-FICH04.
000002
000003      EXEC CICS READNEXT FILE ('FICH04')
000004                      INTO (REG-FICH04)
000005                      RIDFLD (FICH04-CHAVE)
000006      END-EXEC.
000007
000008      MOVE EIBRESP              TO W-EIBRESP.
000009      EVALUATE TRUE
000010      WHEN EIB-OK
000011          MOVE FICH04-NCONTIT TO NCONTIT OF VTN02001
000012          MOVE FICH04-QTD     TO QDISPON OF VTN02001
000013          MOVE FICH04-CESPECIE TO CESPECIE OF VTN02001
000014          IF FICH04-DTMOVTO NOT = W-DATA
000015              MOVE +13        TO W-EIBRESP
000016          END-IF
000017      WHEN (NOT EIB-NOTFND) AND (NOT EIB-ENDFILE)
000018          MOVE 'READNEXT'      TO W-TIPO-ACESSO
000019          MOVE 'FICH04'        TO W-TABELA
000020          PERFORM SAIDA-ERRO-VSAM
000021      END-EVALUATE.

```

1.3.5 Escrita

Para escrever em ficheiros indexados usa-se o comando **REWRITE FILE** (se o registo já existir) ou **WRITE FILE** (se o registo ainda não existir).

```

000001 ACTUALIZAR-FICHEIRO.
000002
000003      EXEC CICS IGNORE CONDITION ERROR END-EXEC.
000004      EXEC CICS READ FILE('FICH04')
000005                      INTO (REG-FICH04)
000006                      RIDFLD (FICH04-CHAVE)
000007                      KEYLENGTH(+33)
000008                      EQUAL
000009                      UPDATE
000010      END-EXEC.
000011
000012      MOVE EIBRESP              TO W-EIBRESP.
000013      EVALUATE TRUE
000014      WHEN EIBRESP = ZEROS
000015          EXEC CICS IGNORE CONDITION ERROR END-EXEC
000016          EXEC CICS REWRITE FILE('FICH04')
000017                      FROM (REG-FICH04)
000018          END-EXEC
000019      MOVE EIBRESP              TO W-EIBRESP
000020      IF NOT EIB-OK
000021          MOVE 'REWRITE'        TO W-TIPO-ACESSO
000022          MOVE 'FICH04'        TO W-TABELA
000023          PERFORM SAI-ERRO-VSAM
000024      END-IF
000025      WHEN EIB-NOTFND
000026          PERFORM CRIA-FICH04
000027      WHEN OTHER
000028          MOVE 'READ-UPD'       TO W-TIPO-ACESSO
000029          MOVE 'FICH04'        TO W-TABELA
000030          PERFORM SAI-ERRO-VSAM
000031      END-EVALUATE.

000001 CRIA-FICH04.
000002
000003      MOVE W-QNEGOC              TO FICH04-QTD.
000004
000005      EXEC CICS WRITE
000006                      FROM (REG-FICH04)
000007                      FILE ('FICH04')
000008                      RIDFLD(FICH04-CHAVE)
000009      END-EXEC.
000010
000011      MOVE EIBRESP              TO W-EIBRESP.
000012      IF (NOT EIB-OK) AND (NOT EIB-NOTFND)
000013          MOVE 'WRITE'          TO W-TIPO-ACESSO
000014          MOVE 'FICH04'        TO W-TABELA
000015          PERFORM SAIDA-ERRO-VSAM
000016      END-IF.

```

1.3.6 Eliminação de Registos

Para eliminar registos em ficheiros indexados usa-se o comando DELETE FILE.

```

000001 DELETE-UM-REG.
000002
000003     EXEC CICS
000004         DELETE FILE      ('FICH04')
000005         RIDFLD  (FICH04-CHAVE)
000006         KEYLENGTH (+33)
000007         EQUAL
000008     END-EXEC.
000009
000010     MOVE EIBRESP          TO W-EIBRESP.
000011     IF (NOT EIB-OK) AND (NOT EIB-NOTFND)
000012         MOVE 'DELETE'      TO W-TIPO-ACESSO
000013         MOVE 'FICH04'      TO W-TABELA
000014     PERFORM SAIDA-ERRO-VSAM
000015     END-IF.

000001 DELETE-REGISTOS.
000002
000003     EXEC CICS
000004         DELETE FILE      ('FICH04')
000005         RIDFLD  (FICH04-CHAVE)
000006         KEYLENGTH (+10)
000007         GENERIC
000008     END-EXEC.
000009
000010     MOVE EIBRESP          TO W-EIBRESP.
000011     IF (NOT EIB-OK) AND (NOT EIB-NOTFND)
000012         MOVE 'DELETE'      TO W-TIPO-ACESSO
000013         MOVE 'FICH04'      TO W-TABELA
000014     PERFORM SAIDA-ERRO-VSAM
000015     END-IF.

```

1.3.7 Fecho

Os ficheiros indexados são normalmente fechados no BATCH (Ver JCL - 1.5.1 Fecho de ficheiro no CICS), mas também o podem ser através de comandos Online (Ver 2.1.5 CEMT SET FILE). Contudo, se tivermos usado o comando STARTBR FILE, também devemos finalizar a pesquisa com o comando ENDBR FILE.

```

000001 FINALIZAR-PESQUISA.
000002
000003     EXEC CICS ENDBR FILE ('FICH04')
000004     END-EXEC.
000005
000006     MOVE EIBRESP          TO W-EIBRESP.
000007     IF (NOT EIB-OK) AND (NOT EIB-NOTFND)
000008         MOVE 'ENDBR'      TO W-TIPO-ACESSO
000009         MOVE 'FICH04'      TO W-TABELA
000010     PERFORM SAIDA-ERRO-VSAM
000011     END-IF.

```

1.4 Mapas

A impressão de mapas é feita, página a página, à base do comando SEND TEXT (Ver 1.2.1 SEND). No entanto, para que o texto seja direccionado para uma impressora, é necessário que a transacção tenha sido iniciada para essa impressora (Ver 1.2.5 START). A transacção de impressão tanto pode ser uma criada especificamente para o efeito, como pode ser a própria transacção através do qual o utilizador desencadeou a impressão. Neste último caso, a transacção, após ter sido iniciada para a impressora, deve conseguir reconhecer que se encontra numa etapa em que está a imprimir um mapa para uma impressora e, como tal, todas as eventuais mensagens de erro a emitir nesta fase devem ser feitas para o mapa.

O exemplo seguinte ilustra o processo habitual de elaboração de mapas.

```

000001 IMPRIME-MAPA.
000002*****
000003
000004      MOVE SPACES                TO WS-PAGINA.
000005      PERFORM OBTEM-DADOS.
000006      PERFORM UNTIL INEXISTENTE-DB2
000007          PERFORM ESCRIVE-REGISTO
000008              PERFORM OBTEM-DADOS
000009      END-PERFORM.
000010      MOVE LINHA-FIM                TO WS-TEXT(55).
000011      PERFORM IMPRIME-PAGINA.
000012
000013
000014 ESCRIVE-REGISTO.
000015*****
000016
000017      MOVE NOPER      OF VTN02801 TO NOPER-MAP.
000018      MOVE NCUPAO    OF VTN01501 TO BALCAO-MAP.
000019      MOVE NOPRORIG  OF VTN01501 TO BOLETIM-MAP.
000020      MOVE NCONTIT   OF VTN01501 TO CONTA-MAP.
000021      MOVE CESPECIE  OF VTN01501 TO ESPECIE-MAP.
000022      MOVE CINTERM   OF VTN01501 TO CORRETOR-MAP.
000023      MOVE QNEGOC    OF VTN01501 TO QTDE-MAP.
000024      MOVE VPUPER    OF VTN01501 TO PRECO-MAP.
000025      MOVE '$'       TO PRECO-MAP(11:1).
000026
000027      MOVE W-CORRETAGEM TO CORRETAGEM-MAP.
000028      MOVE '$'         TO CORRETAGEM-MAP(11:1).
000029      MOVE W-TXBOLSA   TO TXBOLSA-MAP.
000030      MOVE '$'         TO TXBOLSA-MAP(11:1).
000031      MOVE VDESREM OF VTN04701 TO VLR-TIT-MAP.
000032      MOVE '$'         TO VLR-TIT-MAP(11:1).
000033      COMPUTE W-VFINANC =
000034          VFINANC OF VTN01501 - VNUMERAR OF VTN01501 -
000035          W-CORRETAGEM + W-TXBOLSA + VDSFIXA OF VTN04701.
000036      MOVE W-VFINANC TO VLR-DEBITADO-MAP.
000037      MOVE '$'       TO VLR-DEBITADO-MAP(11:1).
000038
000039      IF WS-NLINHA > 54
000040          PERFORM IMPRIME-PAGINA
000041      END-IF.
000042
000043      MOVE LINHA-DET                TO WS-TEXT(WS-NLINHA).
000044      ADD 1                          TO WS-NLINHA.
000045 IMPRIME-PAGINA.
000046*****
000047
000048      ADD 1                          TO WS-NUM-PAG.
000049      MOVE WS-NUM-PAG              TO PAG-CAB.
000050      MOVE LINHA01                 TO WS-TEXT(01).
000051      MOVE LINHA02                 TO WS-TEXT(02).
000052      MOVE LINHA03                 TO WS-TEXT(03).
000053      MOVE LINHA04                 TO WS-TEXT(05).
000054      MOVE LINHA05                 TO WS-TEXT(07).
000055      MOVE LINHA06                 TO WS-TEXT(08).
000056      EXEC CICS SEND TEXT FROM (WS-PAGINA)
000057                          LENGTH (WS-LEN)
000058                          HONEOM PRINT
000059      END-EXEC.
000060      EXEC CICS SEND PAGE          END-EXEC.
000061      EXEC CICS PURGE MESSAGE      END-EXEC.
000062      MOVE SPACES                  TO WS-PAGINA.
000063      MOVE 10                      TO WS-NLINHA.

```

1.5 Temporary Storage

As TSs constituem uma forma de armazenar (em memória ou disco) dados de trabalho de um programa, e caracterizam-se por terem um nome e por associarem a cada registo um número de registo. Para se obter um nome único é usual concatenar-se a identificação do terminal (EIBTRMID) em que o programa está a correr e a transacção que lhe está associada. Normalmente as TSs são usadas em programas que têm opções de <PAGE UP> e <PAGE DOWN>, para armazenar as ocorrências dos ecrãs.

1.5.1 Leitura

Para se ler uma TS, usa-se o comando READQ TS, indicando o número do registro a ler.

```
EXEC CICS READQ TS
  Queue()
  < Sysid() >
  ( SET() | INTO() )
  < Length() >
  < Item() | NExt >
  < NUmitems() >
END-EXEC.

000001 LE-TS.
000002     EXEC CICS READQ TS QUEUE (TS-TSNAME)
000003                     INTO (REG-TS)
000004                     ITEM (TS-NUM-ITEM)
000005     END-EXEC.
000006     IF EIBRESP NOT = ZEROS
000007         MOVE TS-TSNAME          TO W-TABELA
000008         MOVE 'READQ'             TO W-TIPO-ACesso
000009         MOVE EIBRESP             TO W-EIBRESP
000010         PERFORM SAIDA-ERRO-TS
000011     END-IF.
```

1.5.2 Escrita

Para se escrever numa TS, usa-se o comando WRITEQ TS. Este comando, quando usado sem a cláusula REWRITE, escreve sempre no fim da TS e devolve o número do registro escrito.

```
EXEC CICS WRITEQ TS
  Queue()
  < Sysid() >
  From()
  < Length() >
  < NUmitems() | Item() < Rewrite > >
  < Main | Auxiliary >
  < NOSuspend >
END-EXEC.

000001 ESCREVE-TS.
000002     EXEC CICS WRITEQ TS QUEUE (TS-TSNAME)
000003                     FROM (REG-TS)
000004                     ITEM (TS-NUM-ITEM)
000005     END-EXEC.
000006     IF EIBRESP NOT = ZEROS
000007         MOVE TS-TSNAME          TO W-TABELA
000008         MOVE 'WRITEQ'           TO W-TIPO-ACesso
000009         MOVE EIBRESP             TO W-EIBRESP
000010         PERFORM SAIDA-ERRO-TS
000011     END-IF.

000001 ATUALIZA-TS.
000002     EXEC CICS WRITEQ TS QUEUE (TS-TSNAME)
000003                     FROM (REG-TS)
000004                     ITEM (TS-NUM-ITEM)
000005                     REWRITE
000006     END-EXEC.
000007     IF EIBRESP NOT = ZEROS
000008         MOVE TS-TSNAME          TO W-TABELA
000009         MOVE 'REWRITE'          TO W-TIPO-ACesso
000010         MOVE EIBRESP             TO W-EIBRESP
000011         PERFORM SAIDA-ERRO-TS
000012     END-IF.
```

1.5.3 Eliminação

Para se apagar uma TS, usa-se o comando DELETEQ TS.

```
EXEC CICS DELETEQ TS
  Queue()
  < Sysid() >
END-EXEC.

000001     EXEC CICS DELETEQ TS QUEUE (TS-TSNAME) END-EXEC.
```

1.6 Transient Data

As TDs são semelhantes às TSs, diferenciando-se delas por poderem ter associadas a elas (através do JES) um DATA SET. Desta forma, é possível, por exemplo, ter um programa Online a escrever numa TD que depois será lida por um programa Batch.

1.6.1 Leitura

Para se ler uma TD, usa-se o comando READQ TD, indicando o número do registo a ler.

```
EXEC CICS READQ TD
  Queue()
  < Sysid() >
  ( SET() | Into() )
  < Length() >
  < Nosuspend >
END-EXEC.

000001 LE-TD.
000002 EXEC CICS READQ TD QUEUE ('IRDR')
000003                               INTO (REG-TS)
000004                               ITEM (TS-NUM-ITEM)
000005 END-EXEC.
000006 IF EIBRESP NOT = ZEROS
000007   MOVE 'IRDR' TO W-TABELA
000008   MOVE 'READQ' TO W-TIPO-ACesso
000009   MOVE EIBRESP TO W-EIBRESP
000010   PERFORM SAIDA-ERRO-TS
000011 END-IF.
```

1.6.2 Escrita

Para se escrever numa TD, usa-se o comando WRITEQ TD.

```
EXEC CICS WRITEQ TD
  Queue()
  < Sysid() >
  From()
  < Length() >
END-EXEC.

000001 ESCREVE-TD.
000002 EXEC CICS
000003   WRITEQ TD QUEUE ('IRDR')
000004   FROM (REG-TS)
000005   LENGTH (80)
000006 END-EXEC.
000007 IF EIBRESP NOT = ZEROS
000008   MOVE 'IRDR' TO W-TABELA
000009   MOVE 'WRITEQ' TO W-TIPO-ACesso
000010   MOVE EIBRESP TO W-EIBRESP
000011   PERFORM SAIDA-ERRO-TS
000012 END-IF.
```

1.6.3 Eliminação

Para se apagar uma TD, pode usar-se o comando DELETEQ TD.

```
EXEC CICS DELETEQ TD
  Queue()
  < Sysid() >
END-EXEC.

000001 EXEC CICS DELETEQ TD QUEUE ('IRDR') END-EXEC.
```

No entanto, não é usual apagar-se uma TD, uma vez que, deverá ser usada por um programa Batch. Em vez disso, procede-se do seguinte modo:

Antes de se escrever na TD, esta é limpa através do comando

```
000001 EXEC CICS
000002   SET TDQUEUE('IRDR') OPENSTATUS('18')
000003 END-EXEC.
```

Após se ter escrito na TD, esta é fechada através do comando

```
000001 EXEC CICS
```

```
000002      SET TDQUEUE('IRDR') OPENSTATUS('19')
000003      END-EXEC.
```

A este comando pode estar associado um *trigger* para desencadear um JOB Batch. Se na TD tiver sido escrito o JCL de um JOB, pode usar-se uma TD genérica para desencadear JOBS - os escritos na TD.

1.7 Ecrãs

Os ecrãs são construídos através de uma linguagem própria e, uma vez ensamblados, o Copy resultante é incluído no programa. De uma forma simplista, pode dizer-se que a sintaxe de descrição de um campo é: **NomeCampo** DFHMDF POS=(**linha,coluna**),**Atributos** (Ver 1.7.2 a 1.7.5).

1.7.1 Sufixos das Variáveis

Após a ensamblagem do BMS é gerado um Copy (a incluir no programa associado ao ecrã) com os campos do ecrã sufixados por várias letras, cada uma delas originando uma variável com uma função específica. Assim, temos:

Sufixo	Função da variável correspondente
L	Usada para posicionar o cursor no campo (movendo -1 para a variável)
A	Usada para alterar os atributos do campo (movendo-os ⁶ para a variável)
C	Usada para alterar a cor do campo (movendo-a ⁷ para a variável)
P	
H	
V	
I e O	Usadas para transmitir o valor do campo. Estas variáveis são REDEFINES uma da outra, pelo que, podem ter tipos de dados diferentes. Para o efeito, devem usar-se na construção do BMS as cláusulas PICIN (sufixo I) e PICOUT (sufixo O) de acordo com a conveniência.

O exemplo seguinte resultou da ensamblagem do ecrã apresentado em 1.7.5 Ecrã de Consulta de Dados com Ocorrências.

```
01  MAPAI.
    02  FILLER PIC X(12).
    02  DATEL  COMP PIC S9(4).
    02  DATEF  PICTURE X.
    02  FILLER REDEFINES DATEF.
    03  DATEA  PICTURE X.
    02  FILLER PICTURE X(4).
    02  DATEI  PIC X(10).
    02  TITULL COMP PIC S9(4).
    02  TITULF PICTURE X.
    02  FILLER REDEFINES TITULF.
    03  TITULA PICTURE X.
    02  FILLER PICTURE X(4).
    02  TITULI PIC X(30).
    02  TIMEF  COMP PIC S9(4).
    02  TIMEF  PICTURE X.
    02  FILLER REDEFINES TIMEF.
    03  TIMEA  PICTURE X.
    02  FILLER PICTURE X(4).
    02  TIMEI  PIC X(8).
    02  TERML  COMP PIC S9(4).
    02  TERMF  PICTURE X.
    02  FILLER REDEFINES TERMF.
    03  TERMA  PICTURE X.
    02  FILLER PICTURE X(4).
    02  TERMI  PIC X(4).
    02  LIN005D OCCURS 15 TIMES.
    03  LIN005L COMP PIC S9(4).
    03  LIN005F PICTURE X.
    03  FILLER PICTURE X(4).
    03  LIN005I PIC X(79).
```

⁶ Ver 1.8.8 Estrutura DHHBMSCA

⁷ Ver 1.8.8 Estrutura DHHBMSCA (Cores)

```

02 ERRMSG1 COMP PIC S9(4).
02 ERRMSGF PICTURE X.
02 FILLER REDEFINES ERRMSGF.
03 ERRMSGA PICTURE X.
02 FILLER PICTURE X(4).
02 ERRMSGI PIC X(70).
01 MAPAO REDEFINES MAPAI.
02 FILLER PIC X(12).
02 FILLER PICTURE X(3).
02 DATEC PICTURE X.
02 DATEP PICTURE X.
02 DATEH PICTURE X.
02 DATEV PICTURE X.
02 DATEO PIC X(10).
02 FILLER PICTURE X(3).
02 TITULC PICTURE X.
02 TITULP PICTURE X.
02 TITULH PICTURE X.
02 TITULV PICTURE X.
02 TITULO PIC X(30).
02 FILLER PICTURE X(3).
02 TIMEC PICTURE X.
02 TIMEP PICTURE X.
02 TIMEH PICTURE X.
02 TIMEV PICTURE X.
02 TIMEO PIC X(8).
02 FILLER PICTURE X(3).
02 TERMC PICTURE X.
02 TERMP PICTURE X.
02 TERMH PICTURE X.
02 TERMV PICTURE X.
02 TERMO PIC X(4).
02 DFHMS1 OCCURS 15 TIMES.
03 FILLER PICTURE X(2).
03 LIN005A PICTURE X.
03 LIN005C PICTURE X.
03 LIN005P PICTURE X.
03 LIN005H PICTURE X.
03 LIN005V PICTURE X.
03 LIN005O PIC X(79).
02 FILLER PICTURE X(3).
02 ERRMSGC PICTURE X.
02 ERRMSGP PICTURE X.
02 ERRMSGH PICTURE X.
02 ERRMSGV PICTURE X.
02 ERRMSGO PIC X(70).

```

1.7.2 Ecrã de Manipulação de Dados sem Ocorrências

0	1	2	3	4	5	6	7
123456789012345678901234567890123456789012345678901234567890123456789012							
PTNM503	DFHMSD	TYPE=DSECT,					X
		LANG=COBOL,					X
		STORAGE=AUTO,					X
		EXTATT=YES,					X
		MODE=INOUT,					X
		CTRL=FREEKB,					X
		TIOAPFX=YES					
TNG503A	DFHMDI	SIZE=(24,80),					X
		HEADER=YES,					X
		JUSTIFY=FIRST					
BANC1	DFHMDF	POS=(1,3),					X Início do cabeçalho
		ATTRB=(ASKIP,BRT),					X
		COLOR=NEUTRAL,					X
		LENGTH=10					
	DFHMDF	POS=(1,16),					X
		ATTRB=(ASKIP,NORM),					X
		LENGTH=43,COLOR=BLUE,					X
		INITIAL='***** IDENTIFICACAO DA APLICACAO *****'					
	DFHMDF	POS=(1,60),					X
		ATTRB=(ASKIP,NORM),					X
		LENGTH=03,COLOR=BLUE,					X
		INITIAL='***'					
DATA1	DFHMDF	POS=(1,69),					X
		ATTRB=(ASKIP,NORM),COLOR=BLUE,					X
		LENGTH=10					
	DFHMDF	POS=(2,5),					X
		ATTRB=(ASKIP,NORM),					X
		LENGTH=5,COLOR=BLUE,					X
		INITIAL='TN503'					
CAB1	DFHMDF	POS=(2,25),					X
		ATTRB=(ASKIP,BRT),					X

	COLOR=NEUTRAL,	X
	LENGTH=30	
HORA1	DFHMD F POS=(2,70),	X
	ATTRB=(ASKIP,NORM),	X
	LENGTH=8,COLOR=BLUE	
TERM1	DFHMD F POS=(3,5),	X
	ATTRB=(ASKIP,NORM),COLOR=BLUE,	X
	LENGTH=4	
	DFHMD F POS=(11,05),	X 1ª linha após cabeçalho
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=13,	X
	INITIAL='CORRETOR....:'	
CORRET	DFHMD F POS=(11,19),	X Campo numérico
	ATTRB=(UNPROT,BRT,NUM),	X
	COLOR=RED,	X
	LENGTH=3,PICOUT='999'	
	DFHMD F POS=(11,23),	X
	ATTRB=(ASKIP,BRT),	X
	LENGTH=1,	X
	COLOR=NEUTRAL,	X
	INITIAL=' '	
DCORRET	DFHMD F POS=(11,38),	X Campo de display
	ATTRB=(ASKIP,BRT),	X
	LENGTH=40,	X
	COLOR=TURQUOISE	
	DFHMD F POS=(14,05),	X 2ª linha após cabeçalho
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=13,	X
	INITIAL='ESPECIE.....:'	
ESPECIE	DFHMD F POS=(14,19),	X Campo alfanumérico
	ATTRB=(UNPROT,BRT),	X
	COLOR=RED,	X
	LENGTH=9	
	DFHMD F POS=(14,29),	X
	ATTRB=(ASKIP,BRT),	X
	COLOR=RED,	X
	LENGTH=1	
DESP	DFHMD F POS=(14,38),	X Campo de display
	ATTRB=(ASKIP,NORM),	X
	COLOR=TURQUOISE,	X
	LENGTH=40	
	DFHMD F POS=(17,05),	X 3ª linha após cabeçalho
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=13,	X
	INITIAL='PRECO.....:'	
PRECO	DFHMD F POS=(17,19),	X Campo com casas decimais
	ATTRB=(UNPROT,BRT,NUM),	X
	COLOR=RED,	X
	LENGTH=8,PICIN='9(6)V99'	
	DFHMD F POS=(17,28),	X
	ATTRB=(ASKIP,BRT),	X
	COLOR=RED,	X
	LENGTH=1,INITIAL=' '	
DPRECO	DFHMD F POS=(17,38),	X Campo de display
	ATTRB=(ASKIP,NORM),	X
	COLOR=TURQUOISE,	X
	LENGTH=10,PICOUT='ZZZ.ZZ9,99'	
	DFHMD F POS=(23,02),	X Início Rodapé
	ATTRB=(ASKIP,NORM),	X
	COLOR=YELLOW,	X
	LENGTH=07,	X
	INITIAL='OPCAO -'	
OPCAO	DFHMD F POS=(23,10),	X
	ATTRB=(IC,UNPROT,BRT),	X
	LENGTH=1,COLOR=RED	
	DFHMD F POS=(23,12),	X
	ATTRB=(ASKIP,NORM),	X
	LENGTH=1,COLOR=RED,INITIAL=' '	
	DFHMD F POS=(23,18),	X
	ATTRB=(ASKIP,NORM),	X
	LENGTH=3,	X
	COLOR=YELLOW,	X
	INITIAL='C -'	
	DFHMD F POS=(23,22),	X
	ATTRB=(ASKIP,NORM),	X
	LENGTH=14,COLOR=BLUE,	X
	INITIAL='CONFIRMA ENVIO'	
MSG1	DFHMD F POS=(24,02),	X
	ATTRB=(ASKIP,BRT),	X
	COLOR=NEUTRAL,	X
	LENGTH=70	


```
DFHMSD TYPE=FINAL
END
```

1.7.3 Ecrã de Manipulação de Dados com Ocorrências

0	1	2	3	4	5	6	7
123456789012345678901234567890123456789012345678901234567890123456789012							
PTNM444	DFHMSD	TYPE=DSECT,					X Início do cabeçalho
		LANG=COBOL,					X
		STORAGE=AUTO,					X
		EXTATT=YES,					X
		MODE=INOUT,					X
		CTRL=FREEKB,					X
		TIOAPFX=YES					
TNG444A	DFHMDI	SIZE=(24,80),					X
		HEADER=YES,					X
		COLOR=BLUE,					X
		JUSTIFY=FIRST					
BAN444A	DFHMDI	POS=(1,1),					X
		ATTRB=(ASKIP,BRT),					X
		COLOR=NEUTRAL,					X
		LENGTH=10					
	DFHMDI	POS=(1,17),					X
		ATTRB=(ASKIP,NORM),					X
		LENGTH=45,					X
		INITIAL='***** IDENTIFICACAO DA APLICACAO *****'					
DAT444A	DFHMDI	POS=(1,69),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=10					
	DFHMDI	POS=(2,1),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=5, INITIAL='TN444'					
CAB444A	DFHMDI	POS=(2,25),					X
		ATTRB=(ASKIP,BRT),					X
		COLOR=NEUTRAL,					X
		LENGTH=30					
HOR444A	DFHMDI	POS=(2,70),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=8					
TER444A	DFHMDI	POS=(3,1),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=4					
	DFHMDI	POS=(3,70),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=5,					X
		INITIAL=' PAG. : '					
PAG444A	DFHMDI	POS=(3,76),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		PICOUT='Z9'					
	DFHMDI	POS=(5,1),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=7,					X
		INITIAL=' BALCAO : '					
BALC444	DFHMDI	POS=(5,09),					X
		ATTRB=(ASKIP,FSET),					X
		COLOR=TURQUOISE,					X
		LENGTH=4, PICOUT='9999'					
DBAL444	DFHMDI	POS=(5,15),					X
		ATTRB=(ASKIP,NORM,FSET),					X
		COLOR=TURQUOISE,					X
		LENGTH=40					
	DFHMDI	POS=(6,65),					X
		ATTRB=(ASKIP,BRT),					X
		COLOR=NEUTRAL,					X
		LENGTH=07, INITIAL=' CATIVA? '					
	DFHMDI	POS=(7,01),					X
		ATTRB=(ASKIP,BRT),					X
		COLOR=NEUTRAL,					X
		LENGTH=22, INITIAL=' CONTA D.O. NOME '					
	DFHMDI	POS=(7,59),					X
		ATTRB=(ASKIP,BRT),					X
		COLOR=NEUTRAL,					X
		LENGTH=12, INITIAL=' VALOR S/N/E '					
DETB444	DFHMDI	POS=(9,01),					X 1ª Ocorrência
		ATTRB=(ASKIP,FSET,NORM),					X

```

        COLOR=TURQUOISE,
        LENGTH=66
CATB444  DFHMD  POS=(9,68),
        ATTRB=(FSET,BRT,ASKIP),
        COLOR=RED,
        LENGTH=01
KEYB444  DFHMD  POS=(9,70),
        ATTRB=(FSET,ASKIP,DRK),
        COLOR=RED,
        LENGTH=08
        DFHMD  POS=(09,79),ATTRB=ASKIP,LENGTH=01

.....

DET444  DFHMD  POS=(21,01),
        ATTRB=(ASKIP,FSET,NORM),
        COLOR=TURQUOISE,
        LENGTH=66
CAT444  DFHMD  POS=(21,68),
        ATTRB=(FSET,BRT,ASKIP),
        COLOR=RED,
        LENGTH=01
KEY444  DFHMD  POS=(21,70),
        ATTRB=(FSET,ASKIP,DRK),
        COLOR=RED,
        LENGTH=08
        DFHMD  POS=(21,79),ATTRB=ASKIP,LENGTH=01
        DFHMD  POS=(23,01),
        ATTRB=(ASKIP,NORM),
        COLOR=YELLOW,
        INITIAL='PF7 - ',
        LENGTH=5
        DFHMD  POS=(23,07),
        ATTRB=(ASKIP,NORM),
        COLOR=BLUE,
        INITIAL='PAG. ANTERIOR',
        LENGTH=13
        DFHMD  POS=(23,25),
        ATTRB=(ASKIP,NORM),
        COLOR=YELLOW,
        INITIAL='PF8 - ',
        LENGTH=5
        DFHMD  POS=(23,31),
        ATTRB=(ASKIP,NORM),
        COLOR=BLUE,
        INITIAL='PAG. SEGUINTE',
        LENGTH=15
MSG444A DFHMD  POS=(24,1),
        ATTRB=(ASKIP,BRT),
        COLOR=NEUTRAL,
        LENGTH=79
        DFHMSD TYPE=FINAL
        END

```

X Última ocorrência

X Início Rodapé

Para simplificar a programação relativa a este género de ecrãs, é usual redefinir-se na WORKING-STORAGE do programa o Copy que resultou da assemblagem do BMS, substituindo as variáveis todas por ocorrências.

Para o ecrã anterior, fez-se a seguinte redefinição:

```

000001 COPY PTNM444.
000002 01 FILLER REDEFINES TNG444AI.
000003 05 FILLER PIC X(176).
000004 05 LINHAS-DA-TELA.
000005 07 LIN-TELA OCCURS 13 TIMES.
000006 10 FILLER PIC X(07).
000007 10 RPREF444.
000008 15 NCONTA444 PIC 9999B999B999999.9.
000009 15 FIL01 PIC X(01).
000010 15 NOME444 PIC X(33).
000011 15 FIL02 PIC X(01).
000012 15 VALOR444 PIC ZZZ.ZZZ.ZZ9,99.
000013 10 CATIVA444L PIC S9(04) COMP.
000014 10 CATIVA444A PIC X(01).
000015 10 FILLER PIC X(04).
000016 10 CATIVA444 PIC X(01).
000017 10 SENHA444L PIC S9(04) COMP.
000018 10 SENHA444A PIC X(01).
000019 10 FILLER PIC X(04).
000020 10 SENHA444 PIC X(08).
000021 05 FILLER PIC X(86).

```

1.7.4 Ecrãs de Consulta de Dados sem Ocorrências

O ecrã apresentado nesta secção tem a particularidade de englobar dois ecrãs. Um primeiro ecrã que é um misto entre o ecrã apresentado em 1.7.2 e o que vai ser apresentado na secção seguinte (1.7.5), e um segundo ecrã que tem apenas campos de consulta.

0	1	2	3	4	5	6	7
123456789012345678901234567890123456789012345678901234567890123456789012							
PTNM258	DFHMSD	TYPE=DSECT,					X Início 1º ecrã
		LANG=COBOL,					X
		STORAGE=AUTO,					X
		EXTATT=YES,					X
		MODE=INOUT,					X
		CTRL=FREEKB,					X
		TIOAPFX=YES					
TNG258A	DFHMDI	SIZE=(24,80),					X
		HEADER=YES,					X
		JUSTIFY=FIRST					
BANC1	DFHMDF	POS=(1,1),					X Início cabeçalho 1º ecrã
		ATTRB=(ASKIP,BRT),					X
		COLOR=NEUTRAL,					X
		LENGTH=10					
	DFHMDF	POS=(1,16),					X
		ATTRB=(ASKIP,NORM),					X
		LENGTH=43,COLOR=BLUE,					X
		INITIAL='***** IDENTIFICACAO DA APLICACAO ****'					
	DFHMDF	POS=(1,60),					X
		ATTRB=(ASKIP,NORM),					X
		LENGTH=03,COLOR=BLUE,					X
		INITIAL='***'					
DATA1	DFHMDF	POS=(1,70),					X
		ATTRB=(ASKIP,NORM),COLOR=BLUE,					X
		LENGTH=10					
	DFHMDF	POS=(2,2),					X
		ATTRB=(ASKIP,NORM),					X
		LENGTH=7,COLOR=BLUE,					X
		INITIAL='TN258/1'					
CAB1	DFHMDF	POS=(2,25),					X
		ATTRB=(ASKIP,BRT),					X
		COLOR=NEUTRAL,					X
		LENGTH=30					
HORA1	DFHMDF	POS=(2,71),					X
		ATTRB=(ASKIP,NORM),					X
		LENGTH=8,COLOR=BLUE					
TERM1	DFHMDF	POS=(3,2),					X
		ATTRB=(ASKIP,NORM),COLOR=BLUE,					X
		LENGTH=4					
	DFHMDF	POS=(3,71),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=5,					X
		INITIAL=' PAG.: '					
PAG1	DFHMDF	POS=(3,77),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=2,					X
		PICOUT='Z9'					
	DFHMDF	POS=(5,03),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=21,					X
		INITIAL='COD. ESPECIE INTERNO: '					
ESPECIE	DFHMDF	POS=(05,25),					X
		ATTRB=(IC,UNPROT,FSET,BRT),					X
		COLOR=RED,					X
		LENGTH=9					
	DFHMDF	POS=(05,35),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=RED,					X
		LENGTH=1					
DESP	DFHMDF	POS=(05,38),					X
		ATTRB=(ASKIP,NORM),					X
		COLOR=TURQUOISE,					X
		LENGTH=40					
	DFHMDF	POS=(6,03),					X 1ª linha do 1º ecrã
		ATTRB=(ASKIP,NORM),					X
		COLOR=BLUE,					X
		LENGTH=10,					X
		INITIAL=' SEGMENTO : '					
SEGM	DFHMDF	POS=(06,14),					X
		ATTRB=(IC,UNPROT,BRT,FSET,NUM),					X
		COLOR=RED,					X

	PICIN='9(1)',	X	
	LENGTH=1		
DSEGM	DFHMDF POS=(06,16),	X	
	ATTRB=(ASKIP,NORM),	X	
	COLOR=TURQUOISE,	X	
	LENGTH=40		
	DFHMDF POS=(6,66),	X	
	ATTRB=(ASKIP,NORM),	X	
	COLOR=BLUE,	X	
	LENGTH=08,	X	
	INITIAL='BALCAO :'		
BALCAO	DFHMDF POS=(06,75),	X	
	ATTRB=(IC,UNPROT,BRT,FSET,NUM),	X	
	COLOR=RED,	X	
	PICIN='9(4)',	X	
	LENGTH=4		
	DFHMDF POS=(06,80),	X	
	ATTRB=(ASKIP,NORM),	X	
	COLOR=RED,	X	
	LENGTH=1		
	DFHMDF POS=(7,03),	X	2ª linha do 1º ecrã
	ATTRB=(ASKIP,NORM),	X	
	COLOR=BLUE,	X	
	LENGTH=13,	X	
	INITIAL='DATA RECOLHA:'		
DATA REC	DFHMDF POS=(07,17),	X	
	ATTRB=(IC,UNPROT,BRT,FSET,NUM),	X	
	COLOR=RED,	X	
	PICIN='9(8)',	X	
	LENGTH=8		
DDTREC	DFHMDF POS=(07,26),	X	
	ATTRB=(ASKIP,NORM),	X	
	COLOR=TURQUOISE,	X	
	PICOUT='9999/99/99',	X	
	LENGTH=10		
	DFHMDF POS=(9,02),	X	
	ATTRB=(ASKIP,NORM),	X	
	COLOR=BLUE,	X	
	LENGTH=78,	X	
	INITIAL='SEG. BALC. NUM. BOL. CONTA A DEBITAR QTD PEX		
	DIDA QTD REALIZ. PRECO SIT'		
LIN258	DFHMDF POS=(10,01),	X	Ocorrências do 1º ecrã
	ATTRB=(ASKIP,NORM),	X	
	COLOR=TURQUOISE,	X	
	LENGTH=79, OCCURS=12		
	DFHMDF POS=(23,02),	X	Início Rodapé do 1º ecrã
	ATTRB=(ASKIP,NORM),	X	
	COLOR=YELLOW,	X	
	LENGTH=07,	X	
	INITIAL='ENTER -'		
	DFHMDF POS=(23,10),	X	
	ATTRB=(ASKIP,NORM),	X	
	LENGTH=07,COLOR=BLUE,	X	
	INITIAL='DETALHE'		
	DFHMDF POS=(23,20),	X	
	ATTRB=(ASKIP,NORM),	X	
	COLOR=YELLOW,	X	
	LENGTH=05,	X	
	INITIAL='PF7 -'		
	DFHMDF POS=(23,26),	X	
	ATTRB=(ASKIP,NORM),	X	
	LENGTH=12,COLOR=BLUE,	X	
	INITIAL='PAG.ANTERIOR'		
	DFHMDF POS=(23,41),	X	
	ATTRB=(ASKIP,NORM),	X	
	LENGTH=05,	X	
	COLOR=YELLOW,	X	
	INITIAL='PF8 -'		
	DFHMDF POS=(23,47),	X	
	ATTRB=(ASKIP,NORM),	X	
	LENGTH=12,COLOR=BLUE,	X	
	INITIAL='PAG.SEGUINTE'		
MSG1	DFHMDF POS=(24,02),	X	
	ATTRB=(ASKIP,BRT),	X	
	COLOR=NEUTRAL,	X	
	LENGTH=70		
TNG258B	DFHMDF SIZE=(24,80),	X	Início do 2º ecrã
	HEADER=YES,	X	
	JUSTIFY=FIRST		
BANC2	DFHMDF POS=(1,1),	X	Início cabeçalho 2º ecrã
	ATTRB=(ASKIP,BRT),	X	
	COLOR=NEUTRAL,	X	
	LENGTH=10		
	DFHMDF POS=(1,16),	X	

	ATTRB=(ASKIP,NORM),	X
	LENGTH=43,COLOR=BLUE,	X
	INITIAL='***** IDENTIFICACAO DA APLICACAO *****'	
	DFHMD F POS=(1,60),	X
	ATTRB=(ASKIP,NORM),	X
	LENGTH=03,COLOR=BLUE,	X
	INITIAL='***'	
DATA2	DFHMD F POS=(1,70),	X
	ATTRB=(ASKIP,NORM),COLOR=BLUE,	X
	LENGTH=10	
	DFHMD F POS=(2,2),	X
	ATTRB=(ASKIP,NORM),	X
	LENGTH=7,COLOR=BLUE,	X
	INITIAL='TN258/2'	
	DFHMD F POS=(2,22),	X
	ATTRB=(ASKIP,BRT),	X
	COLOR=NEUTRAL,	X
	LENGTH=36,	X
	INITIAL='ORDEM PARA OPERACAO PUBLICA DE VENDA'	
HORA2	DFHMD F POS=(2,71),	X
	ATTRB=(ASKIP,NORM),	X
	LENGTH=8,COLOR=BLUE	
TERM2	DFHMD F POS=(3,2),	X
	ATTRB=(ASKIP,NORM),COLOR=BLUE,	X
	LENGTH=4	
	DFHMD F POS=(4,46),	X 1ª linha do 2º ecrã
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=13,	X
	INITIAL='ENT. LIQUID.: '	
ELIQ258	DFHMD F POS=(4,60),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	COLOR=RED,	X
	PICOUT='9(3)',	X
	LENGTH=03	
	DFHMD F POS=(4,66),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=08,	X
	INITIAL='BALCAO : '	
BALC258	DFHMD F POS=(4,75),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	COLOR=RED,	X
	PICOUT='9(4)',	X
	LENGTH=04	
	DFHMD F POS=(5,5),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=22,	X
	INITIAL='COD. ESPECIE INTERNO : '	
ESP258	DFHMD F POS=(5,28),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	LENGTH=9	
DESP258	DFHMD F POS=(5,38),	X
	ATTRB=(ASKIP,NORM,FSET),	X
	COLOR=TURQUOISE,	X
	LENGTH=30	
	DFHMD F POS=(6,5),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='SEGMENTO : '	
SEGM258	DFHMD F POS=(6,21),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	COLOR=RED,	X
	PICOUT='9',	X
	LENGTH=1	
DSEGM258	DFHMD F POS=(6,23),	X
	ATTRB=(ASKIP,NORM,FSET),	X
	COLOR=TURQUOISE,	X
	LENGTH=30	
	DFHMD F POS=(6,57),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=12,	X
	INITIAL='BOLETIM N. : '	
NBOL258	DFHMD F POS=(6,70),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	COLOR=RED,	X
	PICOUT='9(9)',	X
	LENGTH=09	
	DFHMD F POS=(7,05),	X
	ATTRB=(ASKIP,NORM),	X

	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='NACIONALIDADE : '	
NACI258	DFHMDF POS=(7,21),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	COLOR=RED,	X
	PICOUT='9',	X
	LENGTH=1	
DNACI258	DFHMDF POS=(7,23),	X
	ATTRB=(ASKIP,NORM,FSET),	X
	COLOR=TURQUOISE,	X
	LENGTH=16	
	DFHMDF POS=(8,5),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='TIPO TRABALHA. : '	
INVE258	DFHMDF POS=(8,21),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	PICOUT='9',	X
	COLOR=RED,	X
	LENGTH=1	
	DFHMDF POS=(10,05),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='NOME : '	
NOME258	DFHMDF POS=(10,21),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	LENGTH=40	
	DFHMDF POS=(11,05),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='MORADA : '	
MORA258	DFHMDF POS=(11,21),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	LENGTH=40	
	DFHMDF POS=(12,05),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='LOCALIDADE : '	
LOCA258	DFHMDF POS=(12,21),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	LENGTH=40	
	DFHMDF POS=(13,05),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='CODIGO POSTAL : '	
CPOS258	DFHMDF POS=(13,21),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	LENGTH=4	
DPOS258	DFHMDF POS=(13,26),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	LENGTH=35	
	DFHMDF POS=(14,5),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='NUMERO B.I. : '	
BI258	DFHMDF POS=(14,21),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	COLOR=RED,	X
	LENGTH=11	
	DFHMDF POS=(15,5),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='SECTOR INSTIT. : '	
SECT258	DFHMDF POS=(15,21),	X
	ATTRB=(ASKIP,NUM,BRT,FSET),	X
	COLOR=RED,	X
	PICIN='9(6)',	X
	LENGTH=6	
DSECT258	DFHMDF POS=(15,28),	X
	ATTRB=(ASKIP,NORM,FSET),	X
	COLOR=TURQUOISE,	X

	LENGTH=27	
	DFHMD F POS=(16,5),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=15,	X
	INITIAL='TIPO DOCUMENTO: '	
TDOC258	DFHMD F POS=(16,21),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	PICOUT='9',	X
	LENGTH=1	
DTDOC258	DFHMD F POS=(16,25),	X
	ATTRB=(ASKIP,NORM,FSET),	X
	COLOR=TURQUOISE,	X
	LENGTH=30	
	DFHMD F POS=(16,56),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=09,	X
	INITIAL='NUM. DOC: '	
NDOC258	DFHMD F POS=(16,66),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	LENGTH=12	
	DFHMD F POS=(17,5),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=17,	X
	INITIAL='NUM. TRABALHADOR: '	
NTRA258	DFHMD F POS=(17,23),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	COLOR=RED,	X
	PICOUT='9(9)',	X
	LENGTH=09	
	DFHMD F POS=(19,5),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=04,	X
	INITIAL='QTD: '	
QTDE258	DFHMD F POS=(19,10),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	COLOR=RED,	X
	PICOUT='9(9)',	X
	LENGTH=09	
	DFHMD F POS=(20,5),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=12,	X
	INITIAL='TIPO PAGAM.: '	
TPAG258	DFHMD F POS=(20,18),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	PICOUT='9',	X
	LENGTH=1	
DTPAG258	DFHMD F POS=(20,20),	X
	ATTRB=(ASKIP,NORM,FSET),	X
	COLOR=TURQUOISE,	X
	LENGTH=19	
	DFHMD F POS=(21,05),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=21,	X
	INITIAL='MONT. TIT. DIV. PUB.: '	
TDIV258	DFHMD F POS=(21,27),	X
	ATTRB=(ASKIP,BRT,FSET,NUM),	X
	COLOR=RED,	X
	PICOUT='9(12)',	X
	LENGTH=12	
DTDIV258	DFHMD F POS=(21,40),	X
	ATTRB=(ASKIP,NORM,FSET),	X
	COLOR=TURQUOISE,	X
	LENGTH=13	
	DFHMD F POS=(22,05),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X
	LENGTH=14,	X
	INITIAL='DATA RECOLHA: '	
DATAORD	DFHMD F POS=(22,20),	X
	ATTRB=(ASKIP,BRT,FSET),	X
	COLOR=RED,	X
	LENGTH=10	
	DFHMD F POS=(22,40),	X
	ATTRB=(ASKIP,NORM),	X
	COLOR=BLUE,	X

```

                LENGTH=12,                                X
                INITIAL=' SITUACAO      : '
SIT258  DFHMDF POS=(22,53),                                X
                ATTRB=(ASKIP,BRT,FSET,NUM),                X
                COLOR=RED,                                  X
                LENGTH=10
                DFHMDF POS=(23,02),                          X Início Rodapé 2º ecrã
                ATTRB=(ASKIP,NORM),                          X
                COLOR=YELLOW,                                X
                LENGTH=07,                                    X
                INITIAL=' ENTER - '
                DFHMDF POS=(23,10),                          X
                ATTRB=(ASKIP,NORM),                          X
                LENGTH=07,COLOR=BLUE,                       X
                INITIAL=' RETORNA '
MSG2    DFHMDF POS=(24,02),                                X
                ATTRB=(ASKIP,BRT),                          X
                COLOR=NEUTRAL,                              X
                LENGTH=70
                DFHMSD TYPE=FINAL
                END

```

1.7.5 Ecrã de Consulta de Dados com Ocorrências

```

0      1      2      3      4      5      6      7
12345678901234567890123456789012345678901234567890123456789012
PTNM005 DFHMSD TYPE=DSECT,CTRL=FRECKB,LANG=COBOL,          X
                MODE=INOUT,STORAGE=AUTO,TIOAPFX=YES,EXTATT=YES
MAPA    DFHMDI SIZE=(24,80),HEADER=YES,JUSTIFY=FIRST,      X
                COLOR=BLUE
BANC     DFHMDF POS=(1,1),                                  X Início cabeçalho
                ATTRB=(ASKIP,BRT),                          X
                COLOR=NEUTRAL,                              X
                LENGTH=10
                DFHMDF POS=(1,16),                          X
                ATTRB=(ASKIP,NORM),                          X
                LENGTH=43,COLOR=BLUE,                       X
                INITIAL=' ***** IDENTIFICACAO DA APLICACAO ***** '
                DFHMDF POS=(1,60),                          X
                ATTRB=(ASKIP,NORM),                          X
                LENGTH=03,COLOR=BLUE,                       X
                INITIAL=' *** '
DATE     DFHMDF POS=(01,70),LENGTH=10,ATTRB=(ASKIP,NORM,FSET)
                DFHMDF POS=(02,04),LENGTH=5,ATTRB=(ASKIP,NORM), X
                INITIAL=' TN005 '
TITUL    DFHMDF POS=(02,25),LENGTH=30,ATTRB=(ASKIP,FSET,BRT), X
                COLOR=NEUTRAL
TIME     DFHMDF POS=(02,71),LENGTH=8,ATTRB=(ASKIP,NORM)
TERM     DFHMDF POS=(03,04),LENGTH=4,ATTRB=(ASKIP,NORM,FSET)
LIN005   DFHMDF POS=(06,01),LENGTH=79,ATTRB=(PROT,FSET),OCCURS=15 Ocorrências do ecrã
                DFHMDF POS=(23,02),LENGTH=5,ATTRB=(PROT,FSET), X Início do Rodapé
                INITIAL=' PF7 - ',                          X
                COLOR=YELLOW
                DFHMDF POS=(23,08),LENGTH=15,ATTRB=(PROT,FSET), X
                INITIAL=' PAGINA ANTERIOR ',                  X
                COLOR=BLUE
                DFHMDF POS=(23,30),LENGTH=5,ATTRB=(PROT,FSET), X
                INITIAL=' PF8 - ',                          X
                COLOR=YELLOW
                DFHMDF POS=(23,36),LENGTH=15,ATTRB=(PROT,FSET), X
                INITIAL=' PAGINA SEGUINTE ',                  X
                COLOR=BLUE
                DFHMDF POS=(23,58),LENGTH=6,ATTRB=(PROT,FSET), X
                INITIAL=' PF3 - ',                          X
                COLOR=YELLOW
                DFHMDF POS=(23,65),LENGTH=3,ATTRB=(PROT,FSET), X
                INITIAL=' SAI ',                              X
                COLOR=BLUE
ERRMSG   DFHMDF POS=(24,02),LENGTH=70,ATTRB=(ASKIP,BRT),    X
                COLOR=NEUTRAL
                DFHMSD TYPE=FINAL
                END

```

Quando se trabalha com ecrãs que têm ocorrências, é usual ter as ocorrências gravadas numa TS, por forma a que, quando o utilizador tecle <PAGE UP> ou <PAGE DOWN>, o programa se limite a ler da TS os registos na página anterior ou seguinte. Para o efeito, é necessário guardar na área de comunicação o número de total de registos da TS e o número do actual 1º registo do ecrã.

O procedimento para passar os registos da TS para o ecrã é semelhante ao apresentado de seguida.

```
000004  FORMATA-PAG.
000005
000006      PERFORM VARYING IND FROM 1 BY 1 UNTIL IND > 15
000007          MOVE SPACES          TO LIN0050(IND)
000008      END-PERFORM.
000009      MOVE LINHA-INICIAL-LINK      TO LINHA-CORRENTE.
000000      PERFORM VARYING IND FROM 1 BY 1 UNTIL
000001          IND > 15 OR LINHA-CORRENTE > TOTAL-LINHAS-LINK
000002          PERFORM LE-TS
000003          MOVE REG-TS              TO LIN0050(IND)
000006          ADD 1 TO LINHA-CORRENTE
000007      END-PERFORM.
```

Se houver necessidade de validar se na posição em que o utilizador colocou o cursor existia uma linha, procede-se de uma forma semelhante à seguinte:

```
000001  VALIDA-MAP.
000002
000003      EXEC CICS RECEIVE MAP('MAPA')
000004          MAPSET('PTNM005')
000005          INTO(MAPAO)
000006      END-EXEC.
000007
000008      DIVIDE EIBCPOSN BY 80 GIVING LINHA-CURSOR.
000009      SUBTRACT 4 FROM LINHA-CURSOR.
000010      COMPUTE LINHA-CORRENTE = LINHA-INICIAL-LINK +
000011          LINHA-CURSOR - 1.
000012
000013      IF LINHA-CURSOR < 1 OR LINHA-CURSOR > 15 OR
000014          LINHA-CORRENTE > TOTAL-LINHAS-LINK
000015          MOVE 'CURSOR EM POSICAO INVALIDA' TO ERRMSGO
000016          PERFORM ENVIA-MAP
000017      END-IF.
```

1.8 Variáveis de Sistema

1.8.1 DFHCOMMAREA

Esta variável, quando declarada na LINKAGE SECTION, é usada para passar dados entre os programas Online (Ver 1.2.12 Retrieve).

1.8.2 EIBCALEN

Esta variável contém o tamanho da área de comunicação passada ao programa. Quando um programa é chamado via START, este valor é zero (Ver 1.2.12 Retrieve).

1.8.3 EIBTRMID

Esta variável contém a identificação do terminal em que o programa está a ser executado.

1.8.4 EIBCPOSN

Esta variável contém, após o *receive* do ecrã, a posição em que o utilizador deixou o cursor. Esta é uma posição absoluta, começando-se a contar do canto superior esquerdo.

1.8.5 EIBRID

Esta variável contém, após o *receive* do ecrã, a tecla que o utilizador premiu (Ver 1.8.7 Estrutura DFHAID).

1.8.6 EIBRESP

Esta variável contém o código de retorno da execução dos comandos CICS.

Código	Condição	Código	Condição	Código	Condição
00	NORMAL	36	MAPFAIL	72	SUPRESSED

01	ERROR	37	INVERRTERM	73	
02	RDATT	38	INVMPsz	74	
03	WRBRK	39	IGREQID	75	
04	EOF	40	OVERFLOW	76	
04	EODS	41	INVLDC	77	
06	EOC	42	NOSTG	78	
07	INBFMH	43	JIDERR	79	
08	ENDINPT	44	QIDERR	80	
09	NONVAL	45	NOJBUFSP	81	TERMERR
10	NOSTART	46	DSSTAT	82	ROLLEDBACK
11	TERMIDERR	47	SELNERR	83	
12	FILENOTFOUND	48	FUNCERR	84	DISABLED
13	NOTFND	49	UNEXPIN	85	
14	DUPREC	50	NOPASSBKRD	86	
15	DUPKEY	51	NOPASSBKWR	87	OPENERR
16	INVREQ	52		88	
17	IOERR	53	SYSIDERR	89	
18	NOSPACE	54	ISCINVREQ	90	
19	NOTOPEN	55	ENQBUSY	91	
20	ENDFILE	56	ENVDEFERR	92	
21	ILLOGIC	57	IGREQCD	93	
22	LENGERR	58	SESSIONERR	94	LOADING
23	QZERO	59	SYSBUSY		
24	SIGNAL	60	SESSBUSY		
25	QBUSY	61	NOTALLOC		
26	ITEMERR	62	CBIDERR		
27	PGMIDERR	63	INVEXITREQ		
28	TRANSIDERR	64	INVPARTNSET		
29	ENDDATA	65	INVPARTN		
30	INVTSREQ	66	PARTNFAIL		
31	EXPIRED	67			
32	RETPAGE	68			
33	RTEFAIL	69	USERIDERR		
34	RTESOME	70	NOTAUTH		
35	TSIOERR	71			

1.8.7 Estrutura DFHAID

Esta estrutura contém os possíveis valores da variável EIBAID (Ver 1.8.5 EIBAID).

```

01  DFHAID.
02  DFHNULL    PIC X VALUE IS ' '.
02  DFHENTER   PIC X VALUE IS ' '.
02  DFHCLEAR   PIC X VALUE IS '_ '.
02  DFHCLRP    PIC X VALUE IS '| '.
02  DFHPEN     PIC X VALUE IS '='.
02  DFHOPID    PIC X VALUE IS 'W'.
02  DFHMSRE    PIC X VALUE IS 'X'.
02  DFHSTRF    PIC X VALUE IS 'h'.
02  DFHTRIG    PIC X VALUE IS '"'.
02  DFHPA1     PIC X VALUE IS '%'.
02  DFHPA2     PIC X VALUE IS '>'.
02  DFHPA3     PIC X VALUE IS ','.
02  DFHPF1     PIC X VALUE IS '1'.
02  DFHPF2     PIC X VALUE IS '2'.
02  DFHPF3     PIC X VALUE IS '3'.
02  DFHPF4     PIC X VALUE IS '4'.
02  DFHPF5     PIC X VALUE IS '5'.
02  DFHPF6     PIC X VALUE IS '6'.
02  DFHPF7     PIC X VALUE IS '7'.
02  DFHPF8     PIC X VALUE IS '8'.
02  DFHPF9     PIC X VALUE IS '9'.
02  DFHPF10    PIC X VALUE IS ':'.
02  DFHPF11    PIC X VALUE IS '#'.
02  DFHPF12    PIC X VALUE IS '@'.
02  DFHPF13    PIC X VALUE IS 'A'.
02  DFHPF14    PIC X VALUE IS 'B'.
02  DFHPF15    PIC X VALUE IS 'C'.
02  DFHPF16    PIC X VALUE IS 'D'.
02  DFHPF17    PIC X VALUE IS 'E'.
02  DFHPF18    PIC X VALUE IS 'F'.
02  DFHPF19    PIC X VALUE IS 'G'.

```

```

02 DFHPF20 PIC X VALUE IS 'H'.
02 DFHPF21 PIC X VALUE IS 'I'.
02 DFHPF22 PIC X VALUE IS 'ç'.
02 DFHPF23 PIC X VALUE IS '.'.
02 DFHPF24 PIC X VALUE IS '<'.

```

1.8.8 Estrutura DFHBMSCA

Esta estrutura contém os possíveis atributos dos campos dos ecrãs (Ver 1.7 Ecrãs).

```

01 DFHBMSCA.
02 DFHBMPPEM PICTURE X VALUE IS '.'. END OF MESSAGE
02 DFHBMPNL PICTURE X VALUE IS '.'. NEW LINE
02 DFHBMPFF PICTURE X VALUE IS '.'.
02 DFHBMPCCR PICTURE X VALUE IS '.'.
02 DFHBMASK PICTURE X VALUE IS '0'. ASKIP
02 DFHBMMUNP PICTURE X VALUE IS ' '. UNPROT,NORM
02 DFHBMMUNN PICTURE X VALUE IS '&'. UNPROT,NUM (para campos de escrita numérica)
02 DFHBMPRO PICTURE X VALUE IS '-'. PROT,NORM
02 DFHBMBRY PICTURE X VALUE IS 'H'. UNPROT,BRT,PEN
02 DFHBMDAR PICTURE X VALUE IS '<'. UNPROT,DRK (para introdução de passwords)
02 DFHBMFSE PICTURE X VALUE IS 'A'. UNPROT,FSET (para campos de escrita)
02 DFHBMPRF PICTURE X VALUE IS '/'. PROT,FSET (para proteger campos)
02 DFHBMASF PICTURE X VALUE IS '1'. ASKIP,FSET (para campos de visualização)
02 DFHBMASB PICTURE X VALUE IS '8'. ASKIP,BRT,PEN
02 DFHBMEOF PICTURE X VALUE IS 'Ø'.
02 DFHBMCUR PICTURE X VALUE IS '.'.
02 DFHBMEC PICTURE X VALUE IS 'b'.
02 DFHBMFLG PICTURE X.
88 DFHERASE VALUES ARE 'Ø', 'b'.
88 DFHCURSR VALUES ARE '.', 'b'.
02 DFHBMDDET PICTURE X VALUE IS '.'.
02 DFHBMPPO-BIN PIC 9(4) COMP VALUE 3599.
* ABOVE VALUE 3599 = X'0E0F' ADDED BY PTM 81385 (APAR PN2
02 FILLER REDEFINES DFHBMPPO-BIN.
03 DFHBMPPO PICTURE X.
03 DFHBMPPI PICTURE X.
02 DFHSA PICTURE X VALUE IS '.'.
02 DFHCOLOR PICTURE X VALUE IS 'â'.
02 DFHPS PICTURE X VALUE IS 'ä'.
02 DFHHLT PICTURE X VALUE IS '.'.
02 DFH3270 PICTURE X VALUE IS '{'.
02 DFHVAL PICTURE X VALUE IS 'A'.
02 DFHOUTLN PICTURE X VALUE IS 'B'.
02 DFHBKTRN PICTURE X VALUE IS 'ä'.
02 DFHALL PICTURE X VALUE IS '.'.
02 DFHERROR PICTURE X VALUE IS '.'.
02 DFHDFT PICTURE X VALUE IS '.'.
02 DFHDFCOL PICTURE X VALUE IS '.'.
02 DFHBLUE PICTURE X VALUE IS '1'. AZUL
02 DFHRED PICTURE X VALUE IS '2'. VERMELHO
02 DFHPINK PICTURE X VALUE IS '3'. COR DE ROSA
02 DFHGREEN PICTURE X VALUE IS '4'. VERDE
02 DFHTURQ PICTURE X VALUE IS '5'. TURQUESA
02 DFHYELLO PICTURE X VALUE IS '6'. AMARELO
02 DFHNEUTR PICTURE X VALUE IS '7'.
02 DFHBASE PICTURE X VALUE IS '.'.
02 DFHDFHI PICTURE X VALUE IS '.'.
02 DFHBLINK PICTURE X VALUE IS '1'.
02 DFHREVRP PICTURE X VALUE IS '2'.
02 DFHUNDLN PICTURE X VALUE IS '4'.
02 DFHMFIL PICTURE X VALUE IS '.'.
02 DFHMENT PICTURE X VALUE IS '.'.
02 DFHMFEE PICTURE X VALUE IS '.'.
02 DFHUNNOD PICTURE X VALUE IS '('.
02 DFHUNIMD PICTURE X VALUE IS 'I'.
02 DFHUNNUM PICTURE X VALUE IS 'J'.
02 DFHUNNUB PICTURE X VALUE IS 'Q'.
* ABOVE VALUE DFHUNNUB ADDED BY APAR PN67669
02 DFHUNINT PICTURE X VALUE IS 'R'.
02 DFHUNNON PICTURE X VALUE IS ')'.
02 DFHPROTI PICTURE X VALUE IS 'Y'.
02 DFHPROTN PICTURE X VALUE IS '%'.
02 DFHMT PICTURE X VALUE IS '.'.
02 DFHMFT PICTURE X VALUE IS '.'.
02 DFHMET PICTURE X VALUE IS '.'.
02 DFHMFET PICTURE X VALUE IS '.'.
02 DFHDFFR PICTURE X VALUE IS '.'.
02 DFHLEFT PICTURE X VALUE IS '.'.
02 DFHOVER PICTURE X VALUE IS '.'.
02 DFHRIGHT PICTURE X VALUE IS '.'.
02 DFHUNDER PICTURE X VALUE IS '.'.
02 DFHBOX-BIN PIC 9(4) COMP VALUE 15.

```

```
02      FILLER REDEFINES DFHBOX-BIN.  
    03  FILLER      PICTURE X.  
    03  DFHBOX      PICTURE X.  
02      DFHSOSI     PICTURE X    VALUE IS  '.'.  
02      DFHTRANS    PICTURE X    VALUE IS  '0'.  
02      DFHOPAQ      PICTURE X    VALUE IS  '.'. 
```

Capítulo 2 - COMANDOS ONLINE

2.1 CEMT [SET|INQUIRE]

2.1.1 CEMT SET PROG

Esta comando é normalmente usado para “dar a conhecer” ao CICS novas versões de programas ou ecrãs.

Ex: CEMT S PROG(**Programas** ou **Mapsets**) NEW

```
CEMT Set PProgram()
  < CClass() | ALI >
  < Enabled | Disabled >
  < Shared | PRivate >
  < NEwcopy | PHasein >
  < CEDf | NOCedf >
  < DPbsubset | Fullapi >
```

2.1.2 CEMT SET TASK

Esta comando é normalmente usado para “terminar” programas em execução.

```
CEMT Set TAsk() | < All >
  < PRiority() >
  < PUrge | FOrcepurge >
```

2.1.3 CEMT SET TRANS

Esta comando é normalmente usado para visualizar a relação entre as transacções e os programas.

```
CEMT Set TRAnsaction()
  < CClass() | ALI >
  < PRIority() >
  < TClass() >
  < Enabled | Disabled >
  < PUrgeable | NOTpurgeable >
```

2.1.4 CEMT SET TERM

Esta comando é normalmente usado para activar impressoras.

Ex: CEMT SET TERM(**Impressora**) ACQ

```
CEMT Set TErMinal()
  < CClass() | ALI >
  < PRiority() >
  < PAgeable | AUtopageable >
  < Inservice | Outservice >
  < ATi | NOAti >
  < TTi | NOTti >
  < PUrge | Forcepurge | CAnceL >
  < ACquired | COldacq | RELeased >
  < CReate | NOCreate >
  < REMotesystem() >
```

2.1.5 CEMT SET FILE

Esta comando é normalmente usado para abrir e fechar ficheiros no CICS.
Ex: CEMT SET FILE(**Ficheiro**) OPE

CEMT Set File()

< ALI >
< OPen | CLosed | Forceclose >
< ENabled | DIabled >
< REAd | NORead >
< UPdate | NOUpdate >
< ADdable | NOAddable >
< BRowse | NOBrowse >
< DElete | NODelete >
< EXclusive | NOEXclusive >
< EMptyreq | NOEMptyreq >
< OLd | Share >
< DSname() >
< Maxnumrecs() >
< NOTable | CIcstable | USertable >

2.2 CECI

Este comando é usado para executar as instruções CICS usadas nos programas (Ver 1.1 Instruções Básicas).

Ex. de envio de um ecrã: CECI SEND MAP(**Ecrã**) MAPSET(**Mapset**) ERASE

2.3 CEDF

Este comando é usado para fazer DEBUG de programas em execução.
Ex: CEDF **Terminal**

De referir que, se não nos interessar seguir o fluxo do programa, e apenas estivermos interessados no valor de uma variável num determinado ponto do programa, é mais prático alterar o programa para colocar as instruções seguintes (supondo que NUMERO é a variável) do que fazer CEDF.

```
000001      EXEC CICS SEND TEXT FROM(NUMERO)  END-EXEC
000002      EXEC CICS RECEIVE INTO(NUMERO)  END-EXEC.
```

2.4 DSNCR DISP STAT

Esta comando é usado para visualizar os planos associados às transacções.

2.5 CMAC

Esta comando é usado para visualizar das descrições dos códigos de erro.

DFHCMC01 Display On-line Messages and Codes

Type the required message identifier, then press Enter.

```
Component ID. . . .      (for example, TC for Terminal Control
                          FC for File Control, etc.)
                          This field is required for messages in the
                          form DFHxyyyy, where xx is the Component ID.

Message Number. . . AEI1 (for example, 1060, 5718, or Abend Code
                          such as ASRA, etc.)
```

F3=Exit to CICS

ANEXO - ERROS CICS

SQL

O SQL é uma linguagem de definição, pesquisa e manipulação de dados para Bases de Dados Relacionais.

Capítulo 1 - COMANDOS BÁSICOS

1.1 INCLUDE

O comando INCLUDE é usado para incluir peças de *Software* (*copys* com declarações de variáveis e instruções) nos programas. Estes *copys* serão expandidos pelo pré-compilador de DB2 e, uma vez que este pré-compilador “passa” pelos programas antes do pré-compilador de CICS, este deve ser o comando usado para incluir nos programas *copys* com comandos CICS. Contudo, este comando é usado principalmente para incluir os *copys* com os *layouts* das tabelas (DCLGENs). Os DCLGENs contêm as declarações COBOL dos campos das tabelas e podem ser gerados manual ou automaticamente.

Para os gerar manualmente deve obedecer-se à seguinte correspondência:

Declaração SQL	Declaração COBOL
CHAR(n)	X(n)
SMALLINT	S9(4) COMP
INTEGER	S9(9) COMP
DECIMAL(m,n)	S9(m-n)V9(n) COMP-3
DATE	X(10)
TIME	X(8)
TIMESTAMP	X(26)

Para os gerar automaticamente devem ser usados utilitários apropriados. Por exemplo:

```
DCLGEN                      SSID: DB2D
====>

Enter table name for which declarations are required:
 1 SOURCE TABLE NAME ==> TTIT089_POSICOES          (Unqualified)
 2 TABLE OWNER ..... ==> SDB2D                    (Optional)
 3 AT LOCATION ..... ==>                           (Optional)

Enter destination data set:          (Can be sequential or partitioned)
 4 DATA SET NAME ... ==> 'DATIT.COPY.COBOLO(TNT0089)'
 5 DATA SET PASSWORD ==>                      (If password protected)

Enter options as desired:
 6 ACTION ..... ==> ADD          (ADD new or REPLACE old declaration)
 7 COLUMN LABEL .... ==> NO      (Enter YES for column label)
 8 STRUCTURE NAME .. ==> VTN08901 (Optional)
 9 FIELD NAME PREFIX ==>         (Optional)
10 DELIMIT DBCS .... ==> YES     (Enter YES to delimit DBCS identifiers)
11 COLUMN SUFFIX ... ==> NO      (Enter YES to append column name)
12 INDICATOR VARS .. ==> NO      (Enter YES for indicator variables)

000001      EXEC SQL INCLUDE SQLCA END-EXEC.
```

1.2 WHENEVER SQLERROR

Usado para indicar que o programa se encarregará de tratar os erros de acesso à Base de Dados.


```
000001      EXEC SQL WHENEVER SQLERROR CONTINUE END-EXEC.
```

1.3 SELECT

O conceito fundamental em SQL é denominado Bloco de Pesquisa (“Query Block”):

```
SELECT <lista-atributos>
FROM <lista-tabelas>
[WHERE <expressão-lógica>]
[GROUP BY <atributo> [, ...] [HAVING <condição>]]
[ORDER BY <atributo> [ASC|DESC] [, ...]]
```

Pesquisa a uma tabela, seleccionando todos os atributos.

```
000001 SELECT-T01.
000002
000003      EXEC SQL
000004          SELECT *
000005          INTO   :VTN00101
000006          FROM   VTN00101_TITGERAL
000007          WHERE  CESPECIE = :VTN00101.CESPECIE
000008      END-EXEC.
000009      MOVE SQLCODE                TO W-SQLCODE.
000010      IF (NOT BOM-IO-DB2) AND (NOT INEXISTENTE-DB2)
000011          MOVE 'SELECT'          TO W-TIPO-ACESSO
000012          MOVE 'VTN00101'        TO W-TABELA
000013          MOVE 'SELECT-T01'       TO W-PARAGRAFO
000014          PERFORM FIM-ERRO-DB2
000015      END-IF.
```

Pesquisa a uma tabela, seleccionando um atributo da tabela e a data e hora actuais.

```
000001 SELECT-T28.
000002
000003      EXEC SQL
000004          SELECT ZMOVTO, CURRENT TIMESTAMP
000005          INTO   :VTN02801.ZMOVTO, :WS-ZTIMESTP
000006          FROM   VTN02801_GLOBAL
000007      END-EXEC.
000008      MOVE SQLCODE                TO W-SQLCODE.
000009      IF (NOT BOM-IO-DB2) AND (NOT INEXISTENTE-DB2)
000010          MOVE 'SELECT'          TO W-TIPO-ACESSO
000011          MOVE 'VTN02801'        TO W-TABELA
000012          MOVE 'SELECT-T28'       TO W-PARAGRAFO
000013          PERFORM FIM-ERRO-DB2
000014      END-IF.
```

Pesquisa a uma tabela, sem seleccionar quaisquer atributos (apenas para verificar se existem registos que satisfazem a cláusula WHERE).

```
000001 VALIDA-EXISTENCIA.
000002
000003      EXEC SQL
000004          SELECT '1'
000005          INTO   :WS-DUMMY
000006          FROM   VTN00101_TITGERAL
000007          WHERE  CESPECIE = :VTN00101.CESPECIE
000008      END-EXEC.
000009      MOVE SQLCODE                TO W-SQLCODE.
000010      IF (NOT BOM-IO-DB2) AND (NOT INEXISTENTE-DB2)
000011          MOVE 'SELECT'          TO W-TIPO-ACESSO
000012          MOVE 'VTN00101'        TO W-TABELA
000013          MOVE 'VALIDA-EXISTENCIA' TO W-PARAGRAFO
000014          PERFORM FIM-ERRO-DB2
000015      END-IF.
```

Pesquisa a duas tabelas, seleccionando ordenadamente atributos de ambas (o atributo CESPECIE é usado para fazer a junção das tabelas).

```
000001 JOIN-T01-T11.
000002
000003      EXEC SQL
000004          SELECT T11.CESPECIE, T11.VPUORDEM, T11.IORDEM,
000005                  T11.QORDEM, T11.ZEMISORD, T11.NORDEM,
000006                  T01.CESPLISB, T01.CESPPORT
```

```

000006      INTO      :VTN01101.CESPECIE, :VTN01101.VPUORDEM,
000007      :VTN01101.IORDEM,      :VTN01101.QORDEM,
000008      :VTN01101.ZEMISORD, :VTN01101.NORDEM,
000009      :VTN00101.CESPLISB, :VTN00101.CESPPORT
000010      FROM      VTN01101_ORDCOMVEN T11,
000011      VTN00101_TITGERAL T01
000012      WHERE     T11.CSTATUS IN (2,9,10,12)
000013      AND      T01.CESPECIE = T11.CESPECIE
000014      ORDER BY
000015      T01.CESPLISB, T11.VPUORDEM, T11.IORDEM
000016      END-EXEC.
000017      MOVE SQLCODE TO W-SQLCODE.
000018      IF (NOT BOM-IO-DB2) AND (NOT INEXISTENTE-DB2)
000019      MOVE 'SELECT' TO W-TIPO-ACESSO
000020      MOVE 'T11/T01' TO W-TABELA
000021      MOVE 'JOIN-T11-T01' TO W-PARAGRAFO
000022      PERFORM FIM-ERRO-DB2
000023      END-IF.

```

Pesquisa a uma tabela, seleccionando todos os atributos de registos que não têm correspondência noutra tabela (o atributo NCONTIT é usado para fazer a junção das tabelas).

```

000001 SELECT-NOT-EXISTS.
000002
000003      EXEC SQL
000004      SELECT *
000005      INTO :VTN00401
000006      FROM VTN00401_CONTATIT T4
000007      WHERE NOT EXISTS (
000008      SELECT *
000009      FROM VTN02001_CARTEIRA T20
000010      WHERE T20.NCONTIT = T4.NCONTIT)
000011      END-EXEC.
000012      MOVE SQLCODE TO W-SQLCODE.
000013      IF (NOT BOM-IO-DB2) AND (NOT INEXISTENTE-DB2)
000014      MOVE 'SELECT' TO W-TIPO-ACESSO
000015      MOVE 'VTN00401' TO W-TABELA
000016      MOVE 'SELECT-NOT-EXISTS' TO W-PARAGRAFO
000017      PERFORM FIM-ERRO-DB2
000018      END-IF.

```

1.4 INSERT

A introdução de dados numa tabela é feita através do comando:

```

INSERT INTO <nome_tabela> [(<lista_colunas>)]
VALUES (<lista_valores>) | <comando_select>

```

```

000001 INSERT-T15.
000002
000003      EXEC SQL INSERT
000004      INTO      VTN01501_MOVIMENTA
000005      VALUES (:VTN01501)
000006      END-EXEC.
000007      MOVE SQLCODE TO W-SQLCODE.
000008      IF NOT BOM-IO-DB2
000009      MOVE 'INSERT' TO W-TIPO-ACESSO
000010      MOVE 'VTN01501' TO W-TABELA
000011      MOVE 'INSERT-T15' TO W-PARAGRAFO
000012      PERFORM FIM-ERRO-DB2
000013      END-IF.

000001 INSERT-LOG.
000002
000003      MOVE LENGTH OF VTN07001 TO DREGISTO-LEN OF VTN20001.
000004      MOVE VTN07001 TO DREGISTO-TEXT OF VTN20001.
000005      MOVE USER-ID-LINK TO CUSERID OF VTN20001.
000006      EXEC SQL
000007      INSERT INTO VTN20001_TABLOGSIS
000008      (CTABELA,
000009      ZTIMESTP,
000010      CPROGRAMA,
000011      CUSERID,
000012      IOPER,
000013      DREGISTO)
000014      VALUES
000015      ('VTN07001',
000016      CURRENT TIMESTAMP,

```

```

000017          'PTNU255A',
000018          :VTN20001.CUSERID,
000019          :VTN20001.IOPER,
000020          :DREGISTO)
000021      END-EXEC.
000022      MOVE SQLCODE                      TO W-SQLCODE.
000023      IF NOT BOM-IO-DB2
000024          MOVE 'INSERT'                TO W-TIPO-ACESSO
000025          MOVE 'VTN20001'              TO W-TABELA
000026          MOVE 'INSERT-LOG'            TO W-PARAGRAFO
000027          PERFORM FIM-ERRO-DB2
000028      END-IF.

```

1.5 UPDATE

A actualização de dados de uma tabela é feita através do comando:

```

UPDATE <nome_tabela>
SET <nome_coluna> = <valor> [, ...]
[WHERE <condição>]

```

```

000001 UPDATE-T20.
000002
000003      EXEC SQL UPDATE VTN02001_CARTEIRA
000004          SET QDISPON = QDISPON + :W-QNEGOC
000005          WHERE NCONTIT = :VTN01501.NCONTIT
000006              AND CESPECIE = :VTN01501.CESPECIE
000007      END-EXEC.
000008      MOVE SQLCODE                      TO W-SQLCODE.
000009      IF NOT BOM-IO-DB2
000010          MOVE 'UPDATE'                TO W-TIPO-ACESSO
000011          MOVE 'VTN02001'              TO W-TABELA
000012          MOVE 'UPDATE-T20'            TO W-PARAGRAFO
000013          PERFORM FIM-ERRO-DB2
000014      END-IF.

```

1.6 DELETE

A eliminação de registos de uma tabela é feita através do comando:

```

DELETE FROM <nome_tabela>
[WHERE <condição>]

```

```

000001 DELETE-T70.
000002
000003      EXEC SQL DELETE
000004          FROM VTN07001_CARACTOPV
000005          WHERE CESPPRIV = :VTN07001.CESPPRIV
000006      END-EXEC.
000007      MOVE SQLCODE                      TO W-SQLCODE.
000008      IF NOT BOM-IO-DB2
000009          MOVE 'DELETE'                TO W-TIPO-ACESSO
000010          MOVE 'VTN07001'              TO W-TABELA
000011          MOVE 'DELETE-T70'            TO W-PARAGRAFO
000012          PERFORM FIM-ERRO-DB2
000013      END-IF.

```

1.7 COMMIT

O comando COMMIT é normalmente usado (nos programas batch) para tornar permanentes todas as actualizações que o programa efectuou na Base de Dados, até ao momento do COMMIT. No entanto, também pode ser usado em programas que apenas fazem pesquisa, servindo neste caso para libertar recursos.

```

000001      EXEC SQL COMMIT WORK END-EXEC.

```

1.8 ROLLBACK

O comando ROLLBACK é usado (nos programas batch) para desfazer todas as actualizações que o programa efectuou na Base de Dados, após o último COMMIT (caso exista).

```
000001      EXEC SQL ROLLBACK WORK END-EXEC.
```

Capítulo 2 - CURSORES

Os cursores são usados para, dentro dos programas, fazer pesquisas à Base de Dados quando estas podem retornar mais de um registro.

2.1 Declaração

Os cursores obedecem à sintaxe do comando SELECT, e devem ser declarados na WORKING-STORAGE (após a inclusão dos *copys* com os *layouts* das tabelas envolvidas).

Pesquisa a uma tabela, seleccionando ordenadamente alguns atributos dos registos que satisfazem a cláusula WHERE, com indicação de que os registos não serão actualizados.

```
000001      EXEC SQL DECLARE
000002          CURS-T020 CURSOR FOR
000003              SELECT NCONTIT, QDISPON, QBLOQUEA, QCAUCAO,
000004                  QBLQOBRI, QBOLDEB, QPENDCRE
000005              FROM   VTN02001_CARTEIRA
000006              WHERE  CESPECIE = :VTN02001.CESPECIE
000007              ORDER BY NCONTIT
000007              FOR FETCH ONLY
000008      END-EXEC.
```

Pesquisa a duas tabelas, seleccionando alguns atributos (comuns a ambas) dos registos que satisfazem as respectivas cláusulas WHERE.

```
000001      EXEC SQL DECLARE CURS-UNION CURSOR FOR
000002          SELECT NCONTIT, CESPECIE, COPER
000003          FROM   VTN00501_MOVFUNDOS
000004          WHERE  ZOPER = :VTN02801.ZMOVTO
000005          UNION
000006          SELECT NCONTIT, CESPECIE, COPER
000007          FROM   VTN01501_MOVIMENTA
000008          WHERE  ZOPER = :VTN02801.ZMOVTO
000009      END-EXEC.
```

Pesquisa a uma tabela, seleccionando alguns atributos dos registos que satisfazem a cláusula WHERE, com indicação de que os registos serão actualizados.

```
000001      EXEC SQL DECLARE CURS-FOR-UPDT CURSOR FOR
000002          SELECT QDISPON, QCORRCR, QCORRDB,
000003              QBOLDEB, QBOLCRE, QBLOQUEA
000004          FROM   VTN02001_CARTEIRA
000005          WHERE  NCONTIT = :VTN02001.NCONTIT
000006              AND NIC    = :VTN02001.NIC
000007              AND CESPECIE = :VTN02001.CESPECIE
000008          FOR UPDATE OF QDISPON, QCORRCR, QCORRDB,
000009              QBOLDEB, QBOLCRE, QBLOQUEA
000010      END-EXEC.
```

Pesquisa a uma tabela, seleccionando todos os atributos de todos os registos da tabela, com indicação para manter o cursor aberto até que o programa o feche explicitamente.

```
000001      EXEC SQL DECLARE
000002          CURS-T001 CURSOR WITH HOLD FOR
000003          SELECT *
000004          FROM   VTN00101_TITGERAL
000005      END-EXEC.
```

Pesquisa a uma tabela, seleccionando a média dos valores do atributo VJURDIA, agrupando os registos pelas três primeiras posições do atributo CESPECIE, e devolvendo apenas os grupos que satisfazem as condições do GROUP BY.

```
000001      EXEC SQL DECLARE
000002          CURS-T003 CURSOR FOR
000003          SELECT SUBSTR(CESPECIE,1,3), AVG(VJURDIA)
000004          FROM   VTN00301_TITRNDFIX
```

```

000005      GROUP BY 1 HAVING COUNT(*) > 1 AND MIN(VJURDIA) > 0
000006      END-EXEC.

```

2.2 Abertura

Para abrir cursores usa-se o comando OPEN.

```

000001      EXEC SQL OPEN CURS-T020 END-EXEC.
000002      MOVE SQLCODE                      TO W-SQLCODE.
000003      IF NOT BOM-IO-DB2
000004          MOVE 'OPEN'                      TO W-TIPO-ACESSO
000005          MOVE 'VTN02001'                  TO W-TABELA
000006          MOVE 'OPEN-CURS-T20'            TO W-PARAGRAFO
000007          PERFORM FIM-ERRO-DB2
000008      END-IF.

```

2.3 Selecção de Dados

Uma vez o cursor aberto, usa-se o comando FETCH para obter os dados seleccionados. De referir que os tipos de dados das variáveis para as quais se lêem os dados seleccionados devem ser compatíveis com os tipos de dados da tabela (ver 1.1 INCLUDE).

```

000001      EXEC SQL FETCH CURS-T001 INTO :VTN00101
000002      END-EXEC.
000003      MOVE SQLCODE                      TO W-SQLCODE.
000004      IF NOT BOM-IO-DB2
000005          MOVE 'FETCH'                      TO W-TIPO-ACESSO
000006          MOVE 'VTN00101'                  TO W-TABELA
000007          MOVE 'OPEN-CURS-T001'            TO W-PARAGRAFO
000008          PERFORM FIM-ERRO-DB2
000009      END-IF.

```

2.4 Actualização de Dados

Para actualizar o registo de uma tabela correspondente ao registo corrente do cursor, usa-se o comando UPDATE ... WHERE CURRENT OF De referir que, apesar de não ser obrigatório, é conveniente declarar este tipo de cursores como sendo FOR UPDATE.

```

000001      EXEC SQL UPDATE
000002          VTN02001_CARTEIRA
000003      SET QDISPON = :VTN02001.QDISPON,
000004          QCORRCR = :VTN02001.QCORRCR,
000005          QCORRDB = :VTN02001.QCORRDB,
000006          QBOLDEB = :VTN02001.QBOLDEB,
000007          QBOLCRE = :VTN02001.QBOLCRE,
000008          QBLOQUEA = :VTN02001.QBLOQUEA
000009      WHERE CURRENT OF CURS-FOR-UPDT
000010      END-EXEC.
000011      MOVE SQLCODE                      TO W-SQLCODE.
000012      IF NOT BOM-IO-DB2
000013          MOVE 'UPDATE'                      TO W-TIPO-ACESSO
000014          MOVE 'VTN02001'                  TO W-TABELA
000015          MOVE 'UPDATE-CURS-T020'            TO W-PARAGRAFO
000016          PERFORM FIM-ERRO-DB2
000017      END-IF.

```

2.5 Fecho

Para fechar cursores usa-se o comando CLOSE.

```

000001      EXEC SQL CLOSE CURS-T020 END-EXEC.
000002      MOVE SQLCODE                      TO W-SQLCODE.
000003      IF NOT BOM-IO-DB2
000004          MOVE 'CLOSE'                      TO W-TIPO-ACESSO
000005          MOVE 'VTN02001'                  TO W-TABELA
000006          MOVE 'OPEN-CURS-T20'            TO W-PARAGRAFO
000007          PERFORM FIM-ERRO-DB2
000008      END-IF.

```

Capítulo 3 - FUNÇÕES

O SQL providencia um conjunto de funções que se podem aplicar aos dados, por forma a ser possível obter outras informações sobre esses mesmos dados.

3.1 SUM, COUNT

Pesquisa a uma tabela, seleccionando a soma dos valores de um atributo, relativamente aos registos que satisfazem a cláusula WHERE.

```
000001 SELECT-SUM.
000002
000003     EXEC SQL SELECT SUM(QORDEM)
000004         INTO :W-QNEGOC
000005         FROM VTN01101_ORDCOMVEN
000006         WHERE NCONTIT = :VTN01101.NCONTIT
000007             AND CESPECIE LIKE 'EDP%'
000008             AND IORDEM = 'C'
000009     END-EXEC.
000010     MOVE SQLCODE TO W-SQLCODE.
000011     IF (NOT BOM-IO-DB2) AND (NOT INEXISTENTE-DB2)
000012         MOVE 'SELECT' TO W-TIPO-ACESSO
000013         MOVE 'VTN01101' TO W-TABELA
000014         MOVE 'SELECT-SUM' TO W-PARAGRAFO
000015         PERFORM FIM-ERRO-DB2
000016     END-IF.
```

Pesquisa a uma tabela, seleccionando o número de registos que satisfazem a cláusula WHERE.

```
000001 SELECT-COUNT.
000002
000003     EXEC SQL SELECT COUNT(*)
000004         INTO :W-NUMLINHAS
000005         FROM VTN01101_ORDCOMVEN
000006         WHERE NCONTIT = :VTN01101.NCONTIT
000007             AND SUBSTR(CESPECIE,6,1) IN ('I','M','N')
000008             AND IORDEM = 'C'
000009     END-EXEC.
000010     MOVE SQLCODE TO W-SQLCODE.
000011     IF (NOT BOM-IO-DB2) AND (NOT INEXISTENTE-DB2)
000012         MOVE 'SELECT' TO W-TIPO-ACESSO
000013         MOVE 'VTN01101' TO W-TABELA
000014         MOVE 'SELECT-COUNT' TO W-PARAGRAFO
000015         PERFORM FIM-ERRO-DB2
000016     END-IF.
```

3.2 MAX, MIN e AVG

Pesquisa a uma tabela, seleccionando o mínimo, a média e o máximo dos valores de um atributo, relativamente aos registos que satisfazem a cláusula WHERE.

```
000001 SELECT-MIN-AVG-MAX.
000002
000003     EXEC SQL SELECT MIN(VPUORDEM),AVG(VPUORDEM),MAX(VPUORDEM)
000004         INTO :W-VPUORDEM-MIN, :W-VPUORDEM-AVG, :W-VPUORDEM-MAX
000005         FROM VTN01101_ORDCOMVEN
000006         WHERE NCONTIT = :VTN01101.NCONTIT
000007             AND SUBSTR(CESPECIE,6,1) NOT IN ('I','M','N')
000008             AND IORDEM = 'C'
000009     END-EXEC.
000010     MOVE SQLCODE TO W-SQLCODE.
000011     IF (NOT BOM-IO-DB2) AND (NOT INEXISTENTE-DB2)
000012         MOVE 'SELECT' TO W-TIPO-ACESSO
000013         MOVE 'VTN01101' TO W-TABELA
000014         MOVE 'SELECT-MIN-AVG-MAX' TO W-PARAGRAFO
000015         PERFORM FIM-ERRO-DB2
000016     END-IF.
```

3.3 DATE, YEAR, MONTH, DAY, YEARS, MONTHS, DAYS

Pesquisa a uma tabela, seleccionando o ano, o mês e o dia do valor de um atributo (de tipo **TIMESTAMP**), relativamente a um registo que satisfaz a cláusula **WHERE**.

```

000001 SELECT-YMD.
000002
000003     EXEC SQL SELECT YEAR(ZTIMESTP), MONTH(ZTIMESTP), DAY(ZTIMESTP)
000004         INTO :W-YEAR, :W-MONTH, :W-DAY
000005         FROM VTN01501_MOVIMENTA
000006         WHERE ZOPER = :VTN01501.ZOPER
000007             AND NOPER = :VTN01501.NOPER
000008     END-EXEC.
000009     MOVE SQLCODE                                TO W-SQLCODE.
000010     IF NOT BOM-IO-DB2
000011         MOVE 'SELECT'                            TO W-TIPO-ACESSO
000012         MOVE 'VTN01501'                          TO W-TABELA
000013         MOVE 'SELECT-YMD'                        TO W-PARAGRAFO
000014         PERFORM FIM-ERRO-DB2
000015     END-IF.

```

Pesquisa a uma tabela, seleccionando a data do valor de um atributo (de tipo **TIMESTAMP**), relativamente a um registo que satisfaz a cláusula **WHERE**, e fazendo operações com datas de modo a obter respectivamente:

- A data que resulta da adicção a outra data de um prazo em anos
- A data que resulta da adicção a outra data de um prazo em meses
- A data que resulta da adicção a outra data de um prazo em dias

```

000001 SELECT-DATAS.
000002
000003     EXEC SQL SELECT DATE(ZTIMESTP),
000004         DATE(:VTN01401.ZEVENTO) + :VTN00301.NPRZJUR YEAR,
000005         DATE(:VTN01401.ZEVENTO) + :VTN00301.NPRZJUR MONTH,
000006         DATE(:VTN01401.ZEVENTO) + :VTN00301.NPRZJUR DAY
000007     INTO :W-DATA, :W-DT-YEAR, :W-DT-MONTH, :W-DT-DAY
000008     FROM VTN01401_EVENTOS
000009     WHERE ZEVENETO = :VTN01401.ZEVENETO
000010         AND CEVENETO = :VTN01401.CEVENETO
000011         AND CESPECIE = :VTN01401.CESPECIE
000012     END-EXEC.
000013     MOVE SQLCODE                                TO W-SQLCODE.
000014     IF NOT BOM-IO-DB2
000015         MOVE 'SELECT'                            TO W-TIPO-ACESSO
000016         MOVE 'VTN01401'                          TO W-TABELA
000017         MOVE 'SELECT-DATAS'                      TO W-PARAGRAFO
000018         PERFORM FIM-ERRO-DB2
000019     END-IF.

```

Acesso a uma tabela, fazendo apenas operações datas de modo a obter respectivamente:

- A diferença entre duas datas em anos
- A diferença entre duas datas em meses
- A diferença entre duas datas em dias

```

000001 SELECT-PAZOS.
000002
000003     EXEC SQL SELECT YEARS(:VTN00301.ZULTJUR) - YEARS(:VTN00301.ZPRIMJUR),
000004         MONTS(:VTN00301.ZULTJUR) - MONTHS(:VTN00301.ZPRIMJUR),
000005         DAYS(:VTN00301.ZULTJUR) - DAYS(:VTN00301.ZPRIMJUR)
000006     INTO :W-YEAR, :W-MONTH, :W-DAY
000007     FROM VTN02801_GLOBAL
000008     END-EXEC.
000009     MOVE SQLCODE                                TO W-SQLCODE.
000010     IF NOT BOM-IO-DB2
000011         MOVE 'SELECT'                            TO W-TIPO-ACESSO
000012         MOVE 'VTN02801'                          TO W-TABELA
000013         MOVE 'SELECT-PAZOS'                      TO W-PARAGRAFO
000014         PERFORM FIM-ERRO-DB2
000015     END-IF.

```


3.4 HOUR, MINUTE, SECOND, MICROSECOND

Pesquisa a uma tabela, seleccionando a hora, o minuto, o segundo e o microsegundo do valor de um atributo (de tipo TIMESTAMP), relativamente a um registo que satisfaz a cláusula WHERE.

```

000001 SELECT-HMSM.
000002
000003      EXEC SQL SELECT HOUR(ZTIMESTP), MINUTE(ZTIMESTP),
000004                      SECOND(ZTIMESTP), MICROSECOND(ZTIMESTP)
000005                      INTO :W-HOUR, :W-MINUTE, :W-SECOND, :W-MSECOND
000006                      FROM VTN01501_MOVIMENTA
000007                      WHERE ZOPER = :VTN01501.ZOPER
000008                      AND NOPER = :VTN01501.NOPER
000009      END-EXEC.
000010      MOVE SQLCODE                                TO W-SQLCODE.
000011      IF NOT BOM-IO-DB2
000012          MOVE 'SELECT'                            TO W-TIPO-ACESSO
000013          MOVE 'VTN01501'                          TO W-TABELA
000014          MOVE 'SELECT-HMSM'                        TO W-PARAGRAFO
000015          PERFORM FIM-ERRO-DB2
000016      END-IF.

```

3.5 SUBSTR, CHAR, DECIMAL

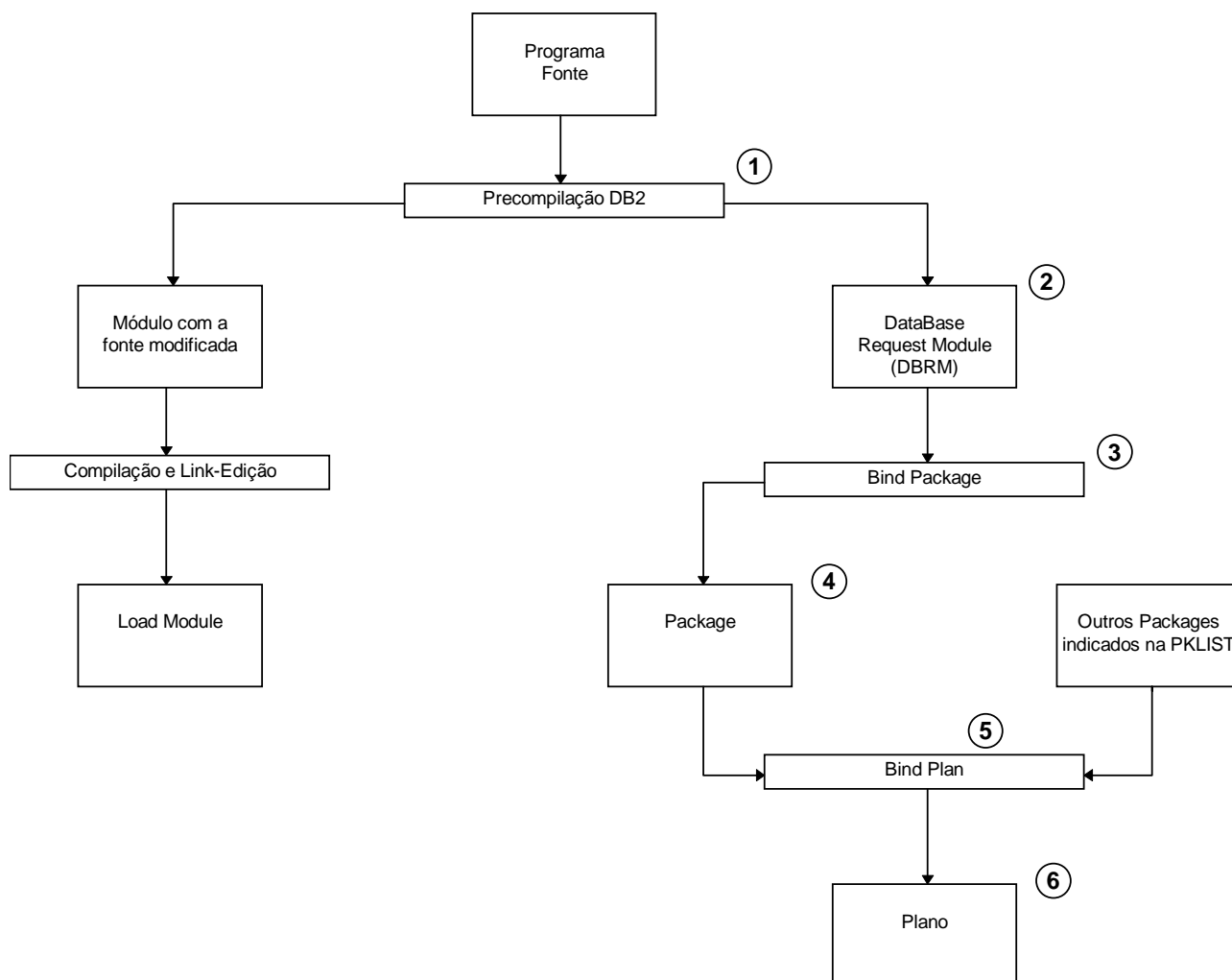
Pesquisa a uma tabela, seleccionando uma *substring* de um campo numérico, relativamente a um registo que satisfaz a cláusula WHERE.

```

000001 SELECT-SCD.
000002
000003      EXEC SQL SELECT SUBSTR(CHAR(DECIMAL(NCONTIT)),3,4)
000004                      INTO :W-STRING
000005                      FROM VTN01501_MOVIMENTA
000006                      WHERE ZOPER = :VTN01501.ZOPER
000007                      AND NOPER = :VTN01501.NOPER
000008      END-EXEC.
000009      MOVE SQLCODE                                TO W-SQLCODE.
000010      IF NOT BOM-IO-DB2
000011          MOVE 'SELECT'                            TO W-TIPO-ACESSO
000012          MOVE 'VTN01501'                          TO W-TABELA
000013          MOVE 'SELECT-SCD'                        TO W-PARAGRAFO
000014          PERFORM FIM-ERRO-DB2
000015      END-IF.

```

Capítulo 4 - BIND PACKAGE e BIND PLAN



(1) Processo que transforma as instruções EXEC SQL em CALLs à linguagem de *interface* do DB2.

(2) Módulo que contém as instruções SQL extraídas do programa.

(3) Processo que chama o otimizador para que este determine os melhores caminhos de acesso aos dados, em função das instruções SQL contidas no DBRM.

(4) Unidade que contém os resultados do BIND PACKAGE e cujo conteúdo está associado a um único DBRM. O package resultante do BIND PACKAGE fica inserido na coleção especificada pelo utilizador.

(5) Processo que inclui num único plano os *packages* indicados pelo utilizador na PKLIST. Por exemplo, se tivermos PKLIST(ColecçãoA.PackageA, ColecçãoA.PackageB, ColecçãoB.*), será gerado um plano com:

- PackageA da ColecçãoA
- PackageB da ColecçãoA
- Todos os *packages* da ColecçãoB

(6) Conjunto de *packages* gerado pelo BIND PLAN.

Capítulo 5 - TABELAS DE SISTEMA

O Sistema de Gestão de Bases de Dados DB2 mantém um conjunto de tabelas (o catálogo do DB2) com dados relativos aos vários objectos DB2. Estes dados podem ser consultados através do comando SELECT (usando um interface interactivo - SPUFI ou QMF), tal como se tratassem de quaisquer outras tabelas. De entre as tabelas do catálogo DB2 destacam-se:

SYSTABLES - Contém informações sobre as Tabelas das aplicações.

```
SELECT NAME, CREATOR, DBNAME, COLCOUNT, REMARKS
FROM SYSIBM.SYSTABLES
WHERE NAME LIKE 'TTIT%' AND CREATOR = 'SDB2D'
ORDER BY NAME
;
```

SYSINDEXES - Contém informações sobre os Índices das Tabelas das aplicações.

```
SELECT NAME, CREATOR, DBNAME, COLCOUNT, REMARKS
FROM SYSIBM.SYSINDEXES
WHERE NAME LIKE 'ITIT063%'
;
```

SYSKEYS - Contém informações sobre as Colunas dos Índices.

```
SELECT *
FROM SYSIBM.SYSKEYS
WHERE IXNAME LIKE 'ITIT063%'
;
```

SYSCOLUMNS - Contém informações sobre as Colunas das Tabelas das aplicações.

```
SELECT COLNO, NAME, COLTYPE, LENGTH, SCALE, NULLS, DEFAULT, REMARKS
FROM SYSIBM.SYSCOLUMNS
WHERE TBNAME LIKE 'TTIT%'
AND TBCreator = 'SDB2D'
ORDER BY COLNO
;
```

SYSPACKAGE - Contém informações sobre os Packages associados aos programas.

```
SELECT *
FROM SYSIBM.SYSPACKAGE
WHERE NAME = 'PTNU146A'
;
```

SYSPLAN - Contém informações sobre os Planos das aplicações.

```
SELECT *
FROM SYSIBM.SYSPLAN
WHERE NAME = 'DTNU146A'
;
```

SYSPACKLIST - Contém informações sobre as bibliotecas em que os Planos estão incluídos.

```
SELECT *
FROM SYSIBM.SYSPACKLIST
WHERE PLANNAME = 'ATNU6190'
;
```

Capítulo 6 - ERROS SQL MAIS COMUNS

Código ... +100

Descricao: Row not found for Fetch, Update or Delete, or result of a query is an empty table

----- Descricao Completa -----
Explanation: One of the following conditions occurred:

- . No row met the search conditions specified in an Update or Delete statement
 - . The result of a SELECT INTO statement was an empty table
 - . A FETCH statement was executed when the curso was positioned after the last row of the result table
 - . The result of the subselect of an Insert statement is empty
- When a SELECT statement is executed using SPUFI, this SQL code indicates normal completion

System Action: No data was retrived, updated, or deleted

Código ... -180

Descricao: The string representation of a datetime value has invalid syntax

----- Descricao Completa -----
Explanation: The string representation of a datetime value does not conform to the syntax for the specified or implied data type.

System Action: The statement cannot be executed.

Programmer Response: Check that the datetime value conforms to the syntax for the data type it represents.

Código ... -181

Descricao: The string representation of a datetime values is not a valid datetime value

----- Descricao Completa -----
Explanation: The string representation of a datetime is not in the acceptable range. The proper ranges for datetime values are as follows:

Datetime	Numeric	range	
Years	0001 to 9999	
Months	1 to 12	
Days			
April, June, September, November	...	1 to 30	
February	...	1 to 28	*
January, March, May, July, August, October, December	...	1 to 31	
Hours	0 to 24	**
Minutes	0 to 59	
Seconds	0 to 59	
Microseconds	0 to 999999	

Note:

- * .. Except leap years, when the proper range is 1 to 29
- ** .. If the hour is 24, the other parts of the time value are zeros. If the time format is USA, the hour cannot be greater then 12.

System Action: The statement cannot be executed.

Programmer Response: Check whether the value is within the valid range.

Código ... -301

Descricao: The vallue of input host variable number 'position-number' cannot be used as specified because of its data type

----- Descricao Completa -----
Explanation: The input host variable in the input SQLDA whose entry number is indicated by 'position-number', could not be used as specified in the statement because its data type is incompatible with the request function.

System Action: The statement cannot be executed.

Programmer Response: Verify that the data type of the indicated input host variable in the statement is compatible with the manner in which it is used.

Codigo ... -302

Descricao: The value of input variable number 'position-number' is invalid or too large for target column

----- Descricao Completa -----
 Explanation: The value of the input host variable, whose entry in the SQLDA is indicated by 'position-number', was found to be invalid or to be too large to fit in the corresponding column of the table. One of the following has occurred:

- . The column is defined as string and the host variable contains a string that is too long for the column
- . The column is defined as numeric and the host variable contains a numeric value too large for the definition of the column
- . The host variable is defined as decimal, but contains bad decimal data

System Action: The statement cannot be executed.

Programmer Response: Correct the application program. Check the column type and length of the value or the data type and contents of input host variable 'position-number'. Ensure that the value of the host variable will fit in the column or contains valid decimal data. Valid decimal data is a System/370 packed decimal number.

Codigo ... -303

Descricao: A value cannot be assigned to output host variable number 'position-number' because the data types are not comparable

----- Descricao Completa -----
 Explanation: A FETCH or SELECT into the output host variable, whose entry in the output SQLDA as indicated by 'position-number', could not be performed because the data type of the variable was not compatible with the data type of the corresponding SELECT-list element. The output host variable and the corresponding SELECT-list element must fall in one of the following categories:

- . Both values must be numbers
- . Both values must be character strings
- . Both values must be graphic strings.

In addition, for datetime, and timestamp values, the host variable must be a character string variable with a proper length.

System Action: The FETCH or SELECT cannot be executed. No data was retrieved.

Programmer Response: Verify that table definitions are current, and that the host variable has the proper data type.

Codigo ... -304

Descricao: A value with data type 'data-type1' cannot be assigned to a host variable because the value is not within the range of the host variable in position 'position-number' with data type 'data-type2'

----- Descricao Completa -----
 Explanation: A FETCH or SELECT into a host variable list or structure, position number 'position-number' failed because the host variable having data type 'data-type2' was not large enough to hold the retrieved value having data type 'data-type1'.

System Action: The statement cannot be executed. No data was retrieved. If the statement was FETCH then the CURSOR remains open.

Programmer Response: Verify that table definitions are current, and that the host variable has the proper data type. See the explanation for SQL return code -405 for ranges of SQL data type.

Codigo ... -305

Descricao: The null value cannot be assigned to output host variable number 'position-number' because no indicator variable is specified

----- Descricao Completa -----
 Explanation: A FETCH or embedded SELECT operation resulted in the retrieval of a null value to be inserted into the 'position-number' of the

output SQLDA, for which no indicator variable was provided. An indicator variable must be supplied if a column returns a null value.

System Action: The statement cannot be executed. No data was retrieved.

Programmer Response: Examine the definition of the table that is the object of the FETCH or SELECT, and correct the application program so provide indicator variables for all host variable into which values from columns that can contain null values are retrieved.

Codigo ... -313

Descricao: The number of host variables specified is not equal to the number of parameter markers

----- Descricao Completa -----
Explanation: The number of host variables specified in the EXECUTE or OPEN statement is not the same as the number of parameter markers (question marks) appearing in the prepared SQL statement.

System Action: The statement cannot be executed.

Programmer Response: Correct the application program so that the number of host variables specified in the EXECUTE or OPEN statement is the same as the number of parameter markers appearing in the prepared SQL statement.

Codigo ... -401

Descricao: The operands of an arithmetic or comparison operation are not comparable

----- Descricao Completa -----
Explanation: An arithmetic operation appearing within the SQL statement has mixture of numeric and nonnumeric operands, or the operands of a comparison operation are not compatible

System Action: The statement cannot be executed.

Programmer Response: Check the data types of all operands to ensure that their data types are comparables and compatible with their usage in the statement.
If all the operands of the SQL statement are correct, then, if a view is being accessed, check the data types of all the operands in the view definition.

Codigo ... -405

Descricao: The numeric literal 'literal' cannot be used as specified because it is out of range

----- Descricao Completa -----
Explanation: The specified numeric literal is not in the proper range.

The proper ranges for SQL values are as follows:

- . 5.4e-79 to 7.2e+75 for FLOAT values
- . -9999999999999999 to 9999999999999999 for DECIMAL values
- . -2147483648 to 2147483647 for INTEGER values
- . -32768 to +32767 for small integer (SMALLINT) values

System Action: The statement cannot be executed.

Programmer Response: The value of the literal should be reduced to the appropriate size for this data type. Perhaps you left out a decimal point.

Codigo ... -407

Descricao: An Update or Insert value is null, but the object column 'column-name' cannot contain null values

----- Descricao Completa -----
Explanation: The update or insert values was NULL, but the object column was declared as NOT NULL in the table definition.

Consequently:

- . Null values cannot be inserted into that column.
- . Values in that column cannot be set to NULL by an update.

System Action: The statement cannot be executed.

Note: 'column-name' may or may not be returned in SQLCA, depending on the syntax of the SQL statement in which the error was detected.

Programmer Response: Examine the definition of the object table to determine which columns of the table have the NOT NULL attribute, and correct the SQL statement accordingly.

Codigo ... -408

Descricao: An Update or Insert value is not comparable with the data type of its object column 'column-name'

----- Descricao Completa -----
 Explanation: The data type of the value to be inserted into or set in the column 'column-name' by an INSERT or UPDATE statement is incompatible with the declared data type of that column. Both must be numeric or both must be graphic string:

- . Dates or character
- . Times or character
- . Timestamps or character.

However, dates, times, or timestamps cannot be assigned to a character column that has a field procedure.

System Action: The statement cannot be executed. No data was inserted or update.

Programmer Response: Examine the current definition for the object table, ensure that the host variable or literal value assigned to the specified column has the proper data type.

Codigo ... -501

Descricao: The cursor identified in a fetch or close statement is not open

----- Descricao Completa -----
 Explanation: The application program attempted to either:

1. FETCH using a cursor, or
 2. CLOSE a cursor
- at time when the specified cursor was not open.

System Action: The statement cannot be executed.

Programmer Response: Check for a previous SQL return code that may have closed the cursor. Commit and rollback operations close cursor. SQL codes -404, -625, -679, -901, -904, -909, -910, -911, and -913 will force the cursor to close. After the cursor is closed, any fetches or close cursor statement will receive this SQL code (-501). If no previous SQL codes have been issued, correct the logic of the application program to ensure that the cursor is open at the time the FETCH or CLOSE statement is executed.

Codigo ... -502

Descricao: The cursor identified in an open statement is already open

----- Descricao Completa -----
 Explanation: The application program attempted to execute an OPEN statement for a cursor that was already open.

System Action: The statement cannot be executed. The cursor was unchanged (that is, if it was not 'reopened').

Programmer Response: Correct the logic of the application program to ensure that it does not attempt to execute an OPEN statement for a cursor that is already open.

Codigo ... -803

Descricao: An inserted or updated value is invalid because the index 'index-name' constrains columns of the table such that no two rows can contain duplicate values in those columns. Rowid of existing row is x'rid'

----- Descricao Completa -----
 Explanation: The table that is the object of the INSERT or UPDATE operation is constrained (by UNIQUE INDEX 'index-name') to have unique values in certain columns. Completion of the requested INSERT or UPDATE would result in duplicate values occurring in row 'rid'.

If a view is the object of the INSERT or UPDATE statement, the table that defines the view is constrained. The update might also be caused by a DELETE operation of a parent row that cascades to a dependent row with a delete rule of SET NULL.

System Action: The INSERT, UPDATE, OR delete STATEMENT CANNOT BE EXECUTED. The object table is unchanged.

Programmer Response: Examine the definitions for UNIQUE INDEX 'index-name' to determine the uniqueness constraint imposed.

For an UPDATE statement, verify that the specified operation is consistent with the uniqueness constraint. If this does not indicate the error, examine the object table to determine the cause of the problem.

For an INSERT statement, examine the object table to determine which values violate the uniqueness constraint. If the INSERT statement contains a subquery, match the contents of the object table to determine the cause of the problem.

For a DELETE statement, examine the index key columns in the table that defines the index. These columns contain a foreign key, which when set NULL on a cascade delete from the object table, causes the duplicate values.

Codigo ... -805

Descricao: Program name 'program-name' not found in plan 'plan-name'

----- Descricao Completa -----
Explanation: An attempted was made to execute the application program 'program-name' with a DBRM that has not been bound as part of the application plan 'plan-name'.

System Action: The statement cannot be executed.

Programmer Response: Rebind the application. Be sure that the DBRMs for all SQL bearing application programs executed in the application are specified in the BIND command.

Codigo ... -811

Descricao: The result of an embedded select statement is a table of more than one row, or the result of the subquery of a basic predicate is more than one value

----- Descricao Completa -----
Explanation: Execution of an embedded SELECT statement has resulted in a result table containing more than one row. Alternatively, a subquery contained in a basic predicate has produced more than one value.

System Action: The statement cannot be executed.

Programmer Response: Examine the syntax of the statement to ensure that it contains the proper condition specifications. If it does, there may be a problem with the data that is causing more than one row or value to be returned when you do not expect it.

Codigo ... -904

Descricao: Unsuccessful execution caused by an unavailable resource, reason 'reason-code', type of resource 'resource-type', and resource name 'resource-name'

----- Descricao Completa -----
Explanation: The SQL statement could not be executed because resource 'resource-name' of type 'resource-type' was not available at the time for the reason indicated by 'reason-code'. Refer to Figure 4 in Appendix B, "Problem Determination" on page x-5 for an explanation of resource type codes. Refer to "Section 4. DB2 Codes" on page 4-1 for an explanation of the given reason code.

System Action: The SQL statement cannot be executed.

Programmer Response: Verify the identity of the resource that was not available. The reason the resource was unavailable can be determined by checking the specified 'reason-code' in "Section 4. DB2 Codes" on page 4-1.

Codigo ... -911

Descricao: The current unit of work has been rolled back due to deadlock or timeout. Reason 'reason-code', type of resource 'resource-type', and resource name 'resource-name'

----- Descricao Completa -----
Explanation: The current unit of work was the victim in a deadlock, or experienced a timeout, and had to be rolled back.

The reason code indicated whether a deadlock, or timeout occurred. Refer to

message DSNT500I under "Service Controller and Installation Messages (DSNT...)" on page 3-121 for an explanation of resource type and resource name. Refer to Figure 4 in Appendix B, "Problem Determination" on page x-5 for an explanation of resource type codes.

Note: The changes associated with the unit of work must be entered again.

System Action: The statement cannot be executed. The application is rolled back to the previous COMMIT.

Programmer Response: A long-running application, or an application that is likely to encounter a deadlock, should (if possible) issue frequent COMMIT commands. This can lessen the possibility of a deadlock occurring. On receipt of the -911 return code, the application should, in general, terminate.

For more information about how IMS, CICS, and TSO handle deadlocks, see Section 4 of Application Programming and SQL Guide.

Codigo ... -922

Descricao: Connection authorization failure: 'error-type' error

----- Descricao Completa -----
Explanation: Connection authorization failed because of the error indicated by 'error-type', which may be one of the following:

- . User validation
- . Resource access
- . Subsystem name
- . Installation error.

System Action: The statement cannot be executed. The connection to DB2 is not established.

Programmer Response: If 'error-type' is 'user validation', the authorization-ID specified to DB2 through your attachment facility is not valid for DB2. See your system programmer or your CICS, IMS/VS, or TSO system administrator.

If 'error-type' is 'plan access', allocation of a requested plan is not allowed. Refer to Figure 4 in Appendix B, "Problem Determination" on pages x-5 for a list of other possible resources. See your system administrator.

If 'error-type' is 'subsystem-name', then you specified an invalid subsystem name.

If 'error-type' is 'installation-error', a logon validation exit has denied your request. See your system programmer.

ANEXO - ESTRUTURA SQLCA

```
01 SQLCA.  
  05 SQLCAID      PIC X(8).  
  05 SQLCABC      PIC S9(9) COMP-4.  
  05 SQLCODE      PIC S9(9) COMP-4.  
  05 SQLERRM.  
    49 SQLERRML PIC S9(4) COMP-4.  
    49 SQLERRMC PIC X(70).  
  05 SQLERRP      PIC X(8).  
  05 SQLERRD      OCCURS 6 TIMES  
                  PIC S9(9) COMP-4.  
  05 SQLWARN.  
    10 SQLWARN0 PIC X.  
    10 SQLWARN1 PIC X.  
    10 SQLWARN2 PIC X.  
    10 SQLWARN3 PIC X.  
    10 SQLWARN4 PIC X.  
    10 SQLWARN5 PIC X.  
    10 SQLWARN6 PIC X.  
    10 SQLWARN7 PIC X.  
  05 SQLEXT.  
    10 SQLWARN8 PIC X.  
    10 SQLWARN9 PIC X.  
    10 SQLWARNA PIC X.  
    10 SQLSTATE PIC X(5).
```

JCL

Capítulo 1 - JOBS

É através dos JOBS que se executam os programas e utilitários BATCH, pelo que, pode afirmar-se que eles são a unidade de trabalho BATCH. Eles podem ser executados manualmente (através do comando SUB) ou automaticamente (através de um sistema apropriado - OPC).

Os JOBS são constituídos por STEPs cuja sintaxe é:

//NomeStep TipoCartão Parâmetros Comentários

No entanto, os STEPs podem estender-se por mais de uma linha. Para o efeito, termina-se a linha anterior com o carácter ',' (vírgula) e usa-se o cartão de continuação. Este cartão começa com // e é seguido da restante codificação do STEP (iniciada entre as colunas 2 e 16, inclusive).

Cada tipo de cartão (JOB, EXEC e DD) tem os seus parâmetros específicos, pelo que, é conveniente analisar separadamente cada um deles. Assim, temos:

♦ Cartão JOB

O cartão JOB indica ao sistema o início de uma unidade de trabalho e, de entre os seus parâmetros destacam-se:

- **NOTIFY=userid**

Indica ao sistema que avise o utilizador (**userid**) quando o JOB terminar. Se **userid=&SYSUID**, é avisado o utilizador TSO que submeteu o JOB.

- **REGION=memsize**

Estabelece a dimensão máxima (512K, 2M, 4M, etc) da memória virtual que o JOB poderá utilizar.

- **MSGLEVEL=(m,n)**

Estabelece o nível de detalhe das informações sobre o JOB que o sistema listará.

m=0 ⇒ Só Cartão JOB ; **m=1** ⇒ JCL e Procedimentos ; **m=2** ⇒ JCL de Entrada

n=0 ⇒ Não saem mensagem de alocação ; **n=1** ⇒ Todas as mensagens

- **MSGCLASS=classe**

Permite designar uma classe de saída para as mensagens produzidas pelo JOB.

- **CLASS=classe**

Permite designar uma classe para a execução do JOB.

- **TYPERUN=tipo_exec**

Pede um tratamento especial ao JES.

tipo_exec=HOLD ⇒ Espera uma indicação do operador (comando RELEASE) para fazer executar o JOB

tipo_exec=SCAN ⇒ Pede uma análise sintáctica do JCL, sem execução do JOB

♦ **Cartão EXEC**

O cartão EXEC é usado para executar programas e utilitários, e para chamar procedimentos catalogados. De entre os seus parâmetros destacam-se:

- **PARM=parâmetros**

Permite passar uma informação a um programa que tenha previsto essa possibilidade.

- **TIME=tempo**

Permite fixar o tempo máximo e efectivo de CPU para o STEP.

Obs: Se **tempo=NOLIMIT**, o STEP não tem limite de tempo para execução.

- **COND=(valor,operador,nome_um_step_anterior)**

Estabelece as condições de execução do STEP. Se **valor** é **operador** que o código de retorno do step referenciado, então o STEP não é executado.

valor=0, ..., 4095 e **operador**=GT,LT,EQ,NE,GE,LE

Existem duas condições especiais: COND=EVEN (o STEP é executado mesmo que hajam erros anteriores) e COND=ONLY (o STEP apenas é executado se não houverem erros anteriores).

♦ **Cartão DD**

O cartão DD é usado a seguir ao cartão EXEC para descrever um ficheiro que vai ser usado, e para fazer a ligação entre os ficheiros lógicos (referenciados nos programas) e os ficheiros físicos. De entre os seus parâmetros destacam-se:

- **DSN=ficheiro**

Especifica o nome físico de um ficheiro. Esse nome pode ter várias formas, de entre as quais:

ficheiro=NomeDataSet - até 5 qualificadores de 8 posições cada, separados por '.'

ficheiro=NomeFichParticionado(NomeMembro)

ficheiro=&&NomeFichTemp - apenas existe durante a execução do JOB

- **DISP=(sit_inicial,fim_normal,fim_anormal)**

Especifica as acções a executar sobre o ficheiro em três momentos: quando o JOB se inicia, quando o STEP termina normalmente, e quando o STEP termina anormalmente.

sit_inicial=NEW - Cria o ficheiro

sit_inicial=OLD - Abre o ficheiro em modo Escrita (apaga o conteúdo)

sit_inicial=SHR - Abre o ficheiro em modo Leitura/Escrita (mantém o conteúdo)

sit_inicial=MOD - Abre o ficheiro em modo Append (mantém o conteúdo)

Em caso de omissão, **sit_inicial**=NEW.

fim_normal=KEEP - Mantém o ficheiro

fim_normal=CATLG - Cataloga o ficheiro

fim_normal=PASS - Passa o ficheiro a outros STEPs e elimina-o no fim do JOB

fim_normal=DELETE - Elimina o ficheiro

fim_normal=UNCATLG - Descataloga o ficheiro

Em caso de omissão, **fim_normal**=NEW se **sit_inicial**=NEW, e **fim_normal**=KEEP c.c..

fim_anormal=DELETE - Elimina o ficheiro

fim_anormal=KEEP - Mantém o ficheiro

fim_anormal=CATLG - Cataloga o ficheiro

fim_anormal=UNCATLG - Descataloga o ficheiro

Em caso de omissão, **fim_anormal**=**fim_normal** se **fim_normal**≠PASS. Se **fim_normal**=PASS, procede-se como na omissão de **fim_normal**.

- **DCB=(RECFM=tipo_reg,LRECL=num_bytes_reg,BLKSIZE= num_bytes_bloco)**

Caracteriza fisicamente o ficheiro.

tipo_reg=FB ⇒ Registos de tamanho fixo

tipo_reg=VB ⇒ Registos de tamanho variável

tipo_reg=UB ⇒ Registos de tamanho indefinido

num_bytes_bloco = n * **num_bytes_reg** (com n>0)

- **SPACE=(unidade,(tam_inicial,tam_adicional),RLSE)**

Estabelece o tamanho máximo (**tam_inicial** + 15***tam_adicional**) de espaço em disco a alocar para o ficheiro.

unidade=CYL ⇒ **tam_inicial** e **tam_adicional** medidos em Cilindros

unidade=TRK ⇒ **tam_inicial** e **tam_adicional** medidos em Pistas

unidade=**num_bytes_bloco** ⇒ **tam_inicial** e **tam_adicional** medidos em Blocos

tam_inicial é o número de **unidades** a alocar inicialmente para o ficheiro

tam_adicional é o número de **unidades** a alocar secundariamente para o ficheiro

RLSE é uma indicação para libertar o espaço que sobrar

- *** e DUMMY**

São dois parâmetros especiais do cartão DD e significam, respectivamente, "Input Stream" e inexistência de ficheiro.

1.1 Job de Compilação

```
//DB03041J JOB 'BATCH SEM DB2',MSGCLASS=X,NOTIFY=&SYSUID,CLASS=D,
//          MSGLEVEL=(1,1),REGION=4M
//PROCLIB  JCLLIB ORDER=DATIT.COMP.COBOL
//*
//*-----*
//*      ESTE STEP COMPILA PROGRAMAS BATCH SEM DB2      *
//*-----*
//PTNC001A EXEC PROC=BSDB2,MEM=PTNC001A
//*
//*-----*
//*      ESTE STEP COMPILA PROGRAMAS BATCH COM DB2      *
//*-----*
//PTNE7290 EXEC PROC=BCDB2,MEM=PTNE7290
//*
//*-----*
//*      ESTE STEP COMPILA PROGRAMAS ONLINE SEM DB2    *
//*-----*
//PTNWM503 EXEC PROC=OSDB2,MEM=PTNWM503
//*
//*-----*
//*      ESTE STEP COMPILA PROGRAMAS ONLINE COM DB2    *
//*-----*
//PTNUM041 EXEC PROC=OCDB2,MEM=PTNUM041
```

1.2 Job de Bind

```
//DB03041B JOB 'BIND PACKAGE',MSGCLASS=X,NOTIFY=&SYSUID,CLASS=D,
//          MSGLEVEL=(1,1),REGION=2M
//PROCLIB  JCLLIB ORDER=DATIT.COMP.COBOL
//*
//*-----*
```

```

/*      ESTE STEP EFECTUA O BIND PACKAGE                                *
/*-----*
//PTNU742A EXEC PROC=BINDPKG, MEM=PTNU742A
/*
/*-----*
/*      ESTE STEP EFECTUA O BIND PLAN                                  *
/*-----*
//PTNU745A EXEC PROC=BINDPLN, MEM=DTNU745A

```

1.3 Job de Load com SYSPUNCH

Quando existem diferentes versões de Tabelas e se deseja copiar registos de uma das versões para a outra, é necessário produzir o ficheiro de SYSPUNCH no processo de UNLOAD. Este ficheiro contém informações sobre o *layout* dos registos produzidos no UNLOAD, e deve ser usado ao fazer-se o carregamento desses registos através do LOAD.

```

//DB03041A JOB 'LOAD', MSGLEVEL=(1,1), MSGCLASS=X, NOTIFY=&SYSUID, CLASS=D
/*
/*-----*
//*****
//JOB LIB DD DISP=SHR,
//          DSN=SYS1.DB2.SDSNLOAD
//LOAD EXEC PGM=DSNUTILB, PARM='DB2D, LOADTS2',
// REGION=4M
//SORTLIB DD DISP=SHR, DSN=SYS1.SORTLIB
//SYSPRINT DD SYSOUT=*
//UTPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSUT1 DD DSN=DATIT.LOAD1.SYSUT1.TEMP, DISP=(NEW,DELETE,CATLG),
//          UNIT=3390, SPACE=(CYL,(5,2)),
//          DCB=(BUFNO=20)
//SORTOUT DD DSN=DATIT.LOAD1.SORTOUT.TEMP, DISP=(NEW,DELETE,CATLG),
//          UNIT=3390, SPACE=(CYL,(5,2)),
//          DCB=(BUFNO=20)
//SORTWK01 DD DSN=DATIT.LOAD1.SORTWK1.TEMP, DISP=(NEW,DELETE,DELETE),
//          UNIT=3390, SPACE=(CYL,(5,2)),
//          DCB=(BUFNO=20)
//SORTWK02 DD DSN=DATIT.LOAD1.SORTWK2.TEMP, DISP=(NEW,DELETE,DELETE),
//          UNIT=3390, SPACE=(CYL,(5,2)),
//          DCB=(BUFNO=20)
//SYSDISC DD DSN=DATIT.LOAD1.DISCARD1.TEMP, DISP=(NEW,CATLG,DELETE),
//          UNIT=3390, SPACE=(CYL,(2,2)),
//          DCB=(BUFNO=20)
//SYSERR DD DSN=DATIT.LOAD1.SYSERR.TEMP, DISP=(NEW,CATLG,DELETE),
//          UNIT=3390, SPACE=(CYL,(5,2)),
//          DCB=(BUFNO=20)
//SYSMAP DD DSN=DATIT.LOAD1.SYSMAP.TEMP, DISP=(NEW,CATLG,DELETE),
//          UNIT=3390, SPACE=(CYL,(5,2)),
//          DCB=(BUFNO=20)
//SYSREC DD DISP=SHR,
//          DSN=DATIT.TTIT004T.TAB
//SYSIN DD DSN=DATIT.TTIT004D.SYSPUNCH.DADOS, DISP=SHR

```

1.4 Job de Unload

```

//DB09407U JOB 'UNLOAD SEM VARCHAR', MSGCLASS=X, CLASS=D, NOTIFY=&SYSUID,
//          MSGLEVEL=(1,1), REGION=4M
/*
/*-----*
/*      ESTE JOB DESCARREGA REGISTOS DE UMA TABELA PARA O FICHEIRO    *
/*      INDICADO EM SYSREC00                                           *
/*-----*
//JOB LIB DD DISP=SHR,
//          DSN=SYS1.DB2.SDSNLOAD
//STEP1 EXEC PGM=IKJEFT01, DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
//          DSN SYSTEM(DB2T)
//          RUN PROGRAM(DSNNTIAUL) PLAN(DSNNTIAUL) PARM('SQL') -
//          LIB('GTADM.DB2T.RUNLIB.LOAD')
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSREC00 DD UNIT=3390, SPACE=(CYL,(1,1)), DISP=MOD,
//          DSN=DATIT.TTIT020T.TAB
/*
//SYSPUNCH DD DUMMY
//SYSIN DD *
//          SELECT *
//          FROM SDB2T.VTN02001_CARTEIRA

```

```

WHERE NCONTIT BETWEEN 99565400000000 AND 99565409999999
AND QDISPON + QBOLDEB > 0;
/*

```

1.5 Job de Manipulação de Ficheiros

```

//TTIT0455 JOB (TATIT),CLASS=E,MSGCLASS=P,USER=TTIT,
//          MSGLEVEL=(1,1),REGION=4M
//
//JOB LIB      DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.EMER.LOAD
//            DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.GERAIS.LOAD
//            DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.CONTAS.LOAD
//            DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.CLIENT.LOAD
//            DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.DIVERS.LOAD
//            DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.OPCRED.LOAD
//            DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.FINANC.LOAD
//            DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.GESTAO.LOAD
//            DD DISP=SHR,DSN=SIBS.FTS.LOADLIB
//            DD DISP=SHR,DSN=GTADM.SYSTEM.LOADLIB
//            DD DISP=SHR,DSN=GTADM.APPL.LOAD
//            DD DISP=SHR,DSN=GTADM.FTPCLI.LOAD
//            DD DISP=SHR,DSN=SYS2.ISP.SISPLOAD
//
//PROCLIB      JCLLIB ORDER=TATIT.APPL.PROCLIB
//

```

1.5.1 Fecho de ficheiro no CICS

```

/*-----*
/*  ESTE STEP FECHA UM FICHEIRO NO CICS                      *
/*-----*
//FICH CLO    EXEC  PGM=IEFBR14
//  COMMAND 'MODIFY PBCICT02,CEMT S FILE(CE2TI) CLO'
//

```

1.5.2 Eliminação de ficheiros de trabalho

```

/*-----*
/*  ESTE STEP ELIMINA FICHEIROS DE TRABALHO                 *
/*-----*
//STEP005 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
//          DELETE TATIT.CE2TIT.TEMP
//          DELETE TATIT.LOAD.DISCAL61.TEMP
//          SET MAXCC = 0
//

```

1.5.3 Cópia de um ficheiro

```

/*-----*
/*  ESTE STEP COPIA TODO O CONTEUDO DE UM FICHEIRO (INFILE) PARA *
/*  OUTRO (OUTFILE)                                           *
/*-----*
//STEP010 EXEC PGM=IDCAMS,COND=(4,LT)
//SYSPRINT DD SYSOUT=*
//INFILE  DD DSN=TATIT.CE2TI.DATA,DISP=SHR
//OUTFILE DD DSN=TATIT.CE2TIT.DATA,DISP=MOD
//SYSIN   DD *
//          REPRO INFILE (INFILE) OUTFILE (OUTFILE)
//

```

1.5.4 Ordenação de um ficheiro

```

/*-----*
/*  ESTE STEP GERA UM FICHEIRO (SORTOUT) CONTENDO OS REGISTOS DE *
/*  OUTRO (SORTIN) ORDENADOS PELA CONDICAÇÃO INDICADA EM STNC4990 *
/*-----*
//STEP020 EXEC PROC=PTISORT, MEMBER='STNC4990', COND=(4,LT)
//SORTIN  DD DSN=TATIT.CE2TI.DATA, DISP=SHR
//SORTOUT DD DSN=TATIT.CE2TIT.TEMP, DISP=(NEW,CATLG,DELETE),
//          DCB=(RECFM=FB, LRECL=118),
//          SPACE=(TRK,(30,15),RLSE)
//

```

1.5.5 Impressão de avisos constantes num ficheiro

```

/*-----*
/*  ESTE STEP EXECUTA O PROGRAMA PTNC4990 PARA IMPRIMIR AVISOS *
/*  EM PAPEL ESPECIAL                                           *
/*-----*
//STEP030 EXEC PROC=PTITRCOB, MEMBER='PTNC4990', COND=(0,NE)

```

```
//PRNTDS  OUTPUT LINECT=0,FCB=90,TITLE='TITULOS / CLIENTES',
//        FORMS=M737
//ENT001  DD DSN=TATIT.CE2TIT.TEMP,DISP=SHR
//REL001  DD SYSOUT=R,DEST=LOCAL,OUTPUT=(*.PRNTDS)
//*
```

1.5.6 Carregamento (Load) de um ficheiro para uma tabela

```
//*-----*
/* ESTE STEP CARREGA O CONTEUDO DO FICHEIRO TATIT.CE2TIT.TEMP  *
/* PARA A TABELA INDICADA EM LDTIT161                          *
/*-----*
//TESTE   IF STEP030.RUNCOB.RC = 0 THEN
//STEP050 EXEC PROC=PTITLOAD,MEMBER='LDTIT161'
//SYSDISC DD DSN=TATIT.LOAD.DISCA161.TEMP,DISP=(NEW,CATLG,CATLG),
//        SPACE=(TRK,(30,15),RLSE),DCB=(BUFNO=20)
//SYSREC  DD DSN=TATIT.CE2TIT.TEMP,DISP=SHR
//FIMTESTE ENDIF
//*
```

1.5.7 Inicialização de um ficheiro sequencial

```
//*-----*
/* ESTE STEP COPIA UM FICHEIRO VAZIO (INFILE) PARA OUTRO      *
/* (OUTFILE), INICIALIZANDO ESTE ULTIMO                       *
/*-----*
//TESTE   IF STEP030.RUNCOB.RC = 0 ] STEP050.LOAD.RC <= 4 THEN
//STEP060 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//INFILE  DD DSN=TATIT.INICE2TI.DATA,DISP=SHR
//OUTFILE DD DSN=TATIT.CE2TIT.DATA,DISP=SHR
//SYSIN   DD *
//        REPRO INFILE (INFILE) OUTFILE (OUTFILE)
//FIMTESTE ENDIF
//*
```

1.5.8 Eliminação e realocação de um ficheiro indexado

```
//*-----*
/* ESTE STEP ELIMINA UM FICHEIRO INDEXADO (TATIT.CE2TI.DATA) E *
/* VOLTA A FAZER A SUA ALOCAÇÃO                                *
/*-----*
//TESTE   IF STEP030.RUNCOB.RC = 0 ] STEP050.LOAD.RC <= 4 THEN
//STEP070 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN   DD *
//        DELETE (TATIT.CE2TI.DATA) -
//        PURGE
//        DEFINE CLUSTER -
//        (NAME(TATIT.CE2TI.DATA) -
//        IXD -
//        KEYS(46 0) -
//        FREESPACE(01 01) -
//        RECSZ(118 118) -
//        TRK (15 15) -
//        SHR(3) -
//        SPEED -
//        VOLUMES(DES002) -
//        ) -
//        DATA -
//        (NAME(TATIT.CE2TI.DATA.DADOS) -
//        CISZ(4096) -
//        ) -
//        INDEX -
//        (NAME(TATIT.CE2TI.DATA.INDICE) -
//        )
//        SET MAXCC = 0
//*
```

1.5.9 Inicialização de um ficheiro indexado

```
//*-----*
/* ESTE STEP EXECUTA O PROGRAMA PTNC9940 PARA ESCREVER E APAGAR *
/* UM REGISTO NO FICHEIRO TATIT.CE2TI.DATA, POR FORMA A QUE O  *
/* CICS O RECONHEÇA (TINHA SIDO ELIMINADO NO STEP ANTERIOR)    *
/*-----*
//STEP080 EXEC PROC=PTITRCOB,MEMBER='PTNC9940'
//SAICE2  DD DSN=TATIT.CE2TI.DATA,DISP=SHR
//*
//FIMTESTE ENDIF
//*
```


1.5.10 Separação de um ficheiro em vários

```

/*-----*
/* ESTE STEP COPIA O CONTEUDO DE UM FICHEIRO (INPUT1) PARA TRES *
/* OUTROS FICHEIROS (OUTPUT1,OUTPUT2 E OUTPUT3), COLOCANDO 4000 *
/* REGISTOS EM CADA UM DOS DOIS PRIMEIROS E O RESTANTE NO 3º *
/*-----*
//STEP090 EXEC PGM=IDCAMS,COND=(2,LT)
//SYSPRINT DD SYSOUT=*
//INPUT1 DD DSN=TATIT.FTN047.DATA,DISP=SHR
//OUTPUT1 DD DSN=DATIT.FTN047.D1.DATA,DISP=OLD
//OUTPUT2 DD DSN=DATIT.FTN047.D2.DATA,DISP=OLD
//OUTPUT2 DD DSN=DATIT.FTN047.D3.DATA,DISP=OLD
//SYSIN DD *
        REPRO INFILE(INPUT1) OUTFILE(OUTPUT1) COUNT(4000)
        REPRO INFILE(INPUT1) OUTFILE(OUTPUT1) SKIP(4000) COUNT(4000)
        REPRO INFILE(INPUT1) OUTFILE(OUTPUT2) SKIP(8000)
//*

```

1.5.11 Listagem de um ficheiro

```

/*-----*
/* ESTE STEP LISTA O FICHEIRO TATIT.VSK31001.DATA *
/*-----*
//STEP100 EXEC PGM=IDCAMS
//IN DD DSN=TATIT.VSK31001.DATA,DISP=SHR
//SYSPRINT DD SYSOUT=* A,DEST=R10
//SYSIN DD *
        PRINT -
            INFILE (IN)
//*

```

1.5.12 Abertura de ficheiro no CICS

```

/*-----*
/* ESTE STEP ABRE UM FICHEIRO NO CICS *
/*-----*
//STEP110 EXEC PGM=IEFBR14
// COMMAND 'MODIFY PBCICT02,CEMT S FILE(CE2TI) OPE '
//*

```

1.6 Job de Execução de Programas

```

//TTIT0200 JOB (TATIT),CLASS=E,MSGCLASS=P,USER=TTIT,
//          MSGLEVEL=(1,1),REGION=4M
//*
//JOBLIB DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.EMER.LOAD
//        DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.GERAIS.LOAD
//        DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.CONTAS.LOAD
//        DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.CLIENT.LOAD
//        DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.DIVERS.LOAD
//        DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.OPCRED.LOAD
//        DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.FINANC.LOAD
//        DD DISP=SHR,DSN=PBPRDSW.NDVR.PR.GESTAO.LOAD
//        DD DISP=SHR,DSN=SIBS.FTS.LOADLIB
//        DD DISP=SHR,DSN=GTADM.SYSTEM.LOADLIB
//        DD DISP=SHR,DSN=GTADM.APPL.LOAD
//        DD DISP=SHR,DSN=GTADM.FTPCLI.LOAD
//        DD DISP=SHR,DSN=SYS2.ISP.SISPLOAD
//*
//PROCLIB JCLLIB ORDER=TATIT.APPL.PROCLIB
//*

```

1.6.1 Programa sem ficheiros

```

/*-----*
/* ESTE STEP EXECUTA UM PROGRAMA QUE NAO ACEDE A FICHEIROS *
/*-----*
//STEP010 EXEC PROC=PTITRDB2,MEMBER='RTNE5120'
//*

```

1.6.2 Programa que lê um ficheiro

```

/*-----*
/* ESTE STEP EXECUTA UM PROGRAMA QUE LE UM FICHEIRO *
/*-----*
//STEP020 EXEC PROC=PTITRDB2,MEMBER='RTNE5310',COND=(4,LT)
//FWK522 DD DSN=TATIT.FWK522.DATA,DISP=SHR
//*

```

1.6.3 Programa que imprime um mapa

```

//*-----*
//*  ESTE STEP EXECUTE UM PROGRAMA QUE IMPRIME UM MAPA  *
//*-----*
//STEP030 EXEC PROC=PTITRDB2, MEMBER='RTNE2130', COND=(4,LT)
//REL001   DD SYSOUT=F,FREE=CLOSE,COPIES=2
//*

```

1.6.4 Programa que escreve num ficheiro temporário

```

//*-----*
//*  ESTE STEP EXECUTA UM PROGRAMA QUE ESCRIVE NUM FICHEIRO TEMP.  *
//*-----*
//STEP040 EXEC PROC=PTITRDB2, MEMBER='RTNE5300', COND=(4,LT)
//SAI009   DD DSN=&&TEMP, DISP=(NEW,,DELETE),
//          DCB=(LRECL=032, RECFM=FB),
//          SPACE=(TRK,(2,1),RLSE)
//*

```

1.6.5 Programa que escreve num ficheiro e tem um parâmetro de SYSIN

```

//*-----*
//*  ESTE STEP EXECUTA UM PROGRAMA QUE ESCRIVE NUM FICHEIRO E TEM  *
//*  UM PARAMETRO DE SYSIN  *
//*-----*
//STEP050 EXEC PROC=PTITRDB2, MEMBER='RTNE5280', COND=(4,LT)
//SAI528   DD DSN=TATIT.FWK528.DATA, DISP=SHR          /* OUTPUT
//SYSIN    DD *
ZMOVTOD2
//*

```

Capítulo 2 - PARMLIBS

As PARMLIBs são parâmetros existentes em Bibliotecas para serem usados pelas PROCs chamadas nos JOBS. Cada uma das PROCs ao ser executada acederá à respectiva Biblioteca para expandir a PARMLIB que lhe foi passada como parâmetro.

2.1 Execução de Programas

Esta Parmlib é usada para executar programas Batch com DB2. A localização do programa é indicada na JOBLIB do JOB que executa este programa.

```
DSN SYS(DB2T)
      RUN PROGRAM(PTNE5120) -
      PLAN      (TTNE5120)
END
```

2.2 Ordenação de Ficheiros

Esta Parmlib é usada para ordenar um ficheiro pelas nove primeiras posições.

```
SORT  FIELDS=(01,9,A),FORMAT=CH
```

Esta Parmlib é usada para seleccionar de um ficheiro os registos que têm 'T' na posição 147 e 'N' na posição 148.

```
SORT FIELDS=COPY
      INCLUDE COND=(147,2,CH,EQ,C'TN')
```

Esta Parmlib é usada para ordenar um ficheiro e omitir os registos que têm zeros nas posições 66 a 74.

```
SORT FIELDS=(24,12,A,15,9,A,1,4,A,5,10,A,55,2,A,270,2,A),FORMAT=BI
      OMIT COND=(66,9,ZD,EQ,0)
```

Esta Parmlib é usada para seleccionar de um ficheiro os registos que têm 'TN' ou 'PF' nas posições 147 e 148.

```
SORT FIELDS=COPY
      INCLUDE COND=(147,2,CH,EQ,C'TN',OR,147,2,CH,EQ,C'PF')
```

2.3 Bind Package

Esta Parmlib é usada para fazer o BIND PACKAGE a um programa Online.

```
DSN SYSTEM(DB2D)
      BIND PACKAGE(DCOLTITU) +
      LIBRARY( 'PBPRDSW.NDVR.TU.FINANC.DBRM') +
      OWNER(SDB2X) QUALIFIER(SDB2D) +
      ACTION(REPLACE) VALIDATE(BIND) ISOLATION(CS) +
      EXPLAIN(YES) FLAG(I) +
      MEMBER(PTNU740A)
END
```

Esta Parmlib é usada para fazer o BIND PACKAGE a um programa Batch.

```
DSN SYSTEM(DB2D)
      BIND PACKAGE(DCOLTITE) +
      LIBRARY( 'PBPRDSW.NDVR.TU.FINANC.DBRM') +
      OWNER(SDB2X) QUALIFIER(SDB2D) +
      ACTION(REPLACE) VALIDATE(BIND) ISOLATION(CS) +
      EXPLAIN(YES) FLAG(I) +
      MEMBER(PTNE255A)
END
```

2.4 Bind Plan

Esta Parmlib é usada para fazer o BIND PLAN a um programa Online.

```
DSN SYSTEM(DB2D)
BIND PLAN(DTNU740A) +
  OWNER(SDB2X) +
  PKLIST(DCOLTITU.* ,+
    DCOLALTU.* ,+
    DCOLUTLG.* ) +
  ACTION(REPLACE) VALIDATE(BIND) ISOLATION(CS) FLAG(I) +
  ACQUIRE(USE) RELEASE(COMMIT)
END
```

Esta Parmlib é usada para fazer o BIND PLAN a um programa Batch.

```
DSN SYSTEM(DB2D)
BIND PLAN(DTNE255A) +
  OWNER(SDB2X) +
  PKLIST(DCOLTITE.* ,+
    DCOLALTU.* ,+
    DCOLUTLG.* ) +
  ACTION(REPLACE) VALIDATE(BIND) ISOLATION(CS) FLAG(I) +
  ACQUIRE(ALLOCATE) RELEASE(DEALLOCATE)
END
```

2.5 Load de Tabelas

Esta Parmlib é usada para carregar uma tabela.

```
LOAD DATA RESUME YES LOG YES ENFORCE CONSTRAINTS
  INTO TABLE SDB2T.TTIT065_HISTCE2
```

Capítulo 3 - PROCS

As PROCs são procedimentos genéricos usados nos JOBS, que aceitam como parâmetros as PARMLIBs. Para que os JOBS consigam expandir as PROCs, é necessário indicar na PROCLIB do JOB o nome da Biblioteca onde se encontram. O uso de PROCs tem a vantagem de tornar os JOBS mais sucintos e, portanto, mais legíveis.

3.1 Compilação Batch sem DB2

```
//BSDB2   PROC MEM='          '
//*-----*
//* ESTA PROC COMPILA PROGRAMAS BATCH SEM DB2
//* O PROGRAMA FONTE DEVE ESTAR NA BIBLIOTECA DATIT.SRC.COBOL
//*-----*
//*-----*
//*                                COMPILACAO COBOL II
//*-----*
//COMP    EXEC PGM=IGYCRCTL,COND=(4,LT),
//      PARM=('SOURCE,LIB,RES,APOST,NOLIST,XREF,NODYN,OFFSET ')
//STEPLIB DD DSN=GTADM.COB2.BATCH,DISP=SHR
//      DD DSN=SYS1.COB2COMP,DISP=SHR
//SYSLIB  DD DSN=DATIT.COPY.COBOL,DISP=SHR
//      DD DSN=DGALT.COPY.COBOL,DISP=SHR
//      DD DSN=DGSYS.COPY.COBOL,DISP=SHR
//SYSIN   DD DSN=DATIT.SRC.COBOL(&MEM),DISP=SHR
//SYSLIN  DD DSN=&LSET,DISP=(NEW,PASS),SPACE=(CYL,(1,1)),
//      UNIT=SYSDA
//SYSPRINT DD SYSOUT=*
//SYSUT1  DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT2  DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT3  DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT4  DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT5  DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT6  DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT7  DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//*-----*
//*                                LINKEDICAO
//*-----*
//LKED    EXEC PGM=IEWL,PARM='LIST,XREF,LET,CALL,MAP',
//      COND=(4,LT)
//SYSUT1  DD SPACE=(CYL,(1,1)),UNIT=SYSDA
//SYSPRINT DD SYSOUT=*
//SYSLIB  DD DSN=SYS1.COB2LIB,DISP=SHR
//      DD DSN=DATIT.APPL.LOAD,DISP=SHR
//      DD DSN=DGSYS.APPL.LOAD,DISP=SHR
//OBJPASS DD DSN=&LSET,DISP=(OLD,DELETE,DELETE)
//SYSLMOD DD DSN=DATIT.APPL.LOAD(&MEM),DISP=SHR
//SYSLIN  DD DSN=DATIT.LKDBSDB2.DATA,DISP=SHR
```

3.2 Compilação Batch com DB2

```
//BCDB2   PROC MEM='          '
//*-----*
//* ESTA PROC COMPILA PROGRAMAS BATCH COM DB2
//* O PROGRAMA FONTE DEVE ESTAR NA BIBLIOTECA DATIT.DB2.COBOL
//*-----*
//*-----*
//*                                TRANSLATE DB2
//*-----*
//PREDB2  EXEC PGM=DSNHPC,
//      PARM='HOST(COB2),APOST,SOURCE,XREF,LEVEL(00)'
//STEPLIB DD DSN=SYS1.DB2.SDSNLOAD,DISP=SHR
//DBRMLIB DD DSN=DATIT.DBRM.OBJ(&MEM),DISP=OLD
//SYSLIB  DD DSN=DATIT.COPY.COBOL,DISP=SHR
//      DD DSN=DGALT.COPY.COBOL,DISP=SHR
//      DD DSN=DGSYS.COPY.COBOL,DISP=SHR
//*      DD DSN=DACTA.COPY.COBOL,DISP=SHR
//SYSIN   DD DSN=DATIT.DB2.COBOL(&MEM),DISP=SHR
//SYSCIN  DD DSN=&DSNHOUT,DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1)),
//      DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120)
//SYSPRINT DD SYSOUT=*
//SYSTEM  DD SYSOUT=*
//SYSUT1  DD SPACE=(CYL,(1,1)),UNIT=SYSDA
//SYSUT2  DD SPACE=(CYL,(1,1)),UNIT=SYSDA
```

```

// *-----*
// *                COMPILACAO COBOL II                *
// *-----*
// COMP      EXEC  PGM=IGYCRCTL,COND=(4,LT),
//      PARM=( ' LIB,RES,APOST,NOLIST,XREF,NODYN,RENT,OFFSET  ' )
// STEPLIB DD   DSN=GTADM.COB2.BATCH,DISP=SHR
//          DD   DSN=SYS1.COB2COMP,DISP=SHR
// SYSLIB DD   DSN=DATIT.COPY.COBOLE,DISP=SHR
//          DD   DSN=DGALT.COPY.COBOLE,DISP=SHR
//          DD   DSN=DGSYS.COPY.COBOLE,DISP=SHR
// SYSIN DD   DSN=&DSNHOUT,DISP=(OLD,DELETE)
// SYSLIN DD   DSN=&LSET,DISP=(NEW,PASS),SPACE=(CYL,(1,1)),
//          UNIT=SYSDA
// SYSPRINT DD  SYSOUT=*
// SYSUT1 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT2 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT3 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT4 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT5 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT6 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT7 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// *-----*
// *                LINKEDICAO                          *
// *-----*
// LKED      EXEC  PGM=IEWL,PARM='LIST,XREF,LET,CALL,MAP,AMOD=31,RMOD=ANY',
//          COND=(4,LT)
// SYSUT1 DD   SPACE=(CYL,(1,1)),UNIT=SYSDA
// SYSPRINT DD  SYSOUT=*
// SYSLIB DD   DSN=SYS1.COB2LIB,DISP=SHR
//          DD   DSN=SYS1.DB2.SDSNLOAD,DISP=SHR
//          DD   DSN=DATIT.APPL.LOAD,DISP=SHR
//          DD   DSN=DGALT.APPL.LOAD,DISP=SHR
//          DD   DSN=DGSYS.APPL.LOAD,DISP=SHR
// OBJPASS DD   DSN=&LSET,DISP=(OLD,DELETE,DELETE)
// SYSLMOD DD   DSN=DATIT.APPL.LOAD(&MEM),DISP=SHR
// SYSLIN DD   DSN=DATIT.LKDBCDB2.DATA,DISP=SHR

```

3.3 Compilação Online sem DB2

```

// OSDB2      PROC MEM=' '
// *-----*
// * ESTA PROC COMPILA PROGRAMAS ONLINE SEM DB2        *
// * O PROGRAMA FONTE DEVE ESTAR NA BIBLIOTECA DATIT.SRC.COBOLE *
// *-----*
// *                TRANSLATE CICS                      *
// *-----*
// CICS      EXEC  PGM=DFHECP1$,PARM='COBOLE2,SP'
// STEPLIB DD   DSN=SYS1.CICS.SDFHLOAD,DISP=SHR
// SYSIN DD   DSN=DATIT.SRC.COBOLE(&MEM),DISP=SHR
// SYSPRINT DD  SYSOUT=*
// SYSPUNCH DD  DISP=(NEW,PASS),DSN=&CICSOUT0,SPACE=(400,(4000,1000)),
//          DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120),UNIT=SYSDA
// *-----*
// *                COMPILACAO COBOL II                *
// *-----*
// COMP      EXEC  PGM=IGYCRCTL,COND=(4,LT),
//      PARM=( ' LIB,RENT,APOST,LIST,XREF,NOOPT,NOSEQ  ',
//          ' NODYNAM,OFFSET                          ' )
// STEPLIB DD   DSN=GTADM.COB2.CICS,DISP=SHR
//          DD   DSN=SYS1.COB2COMP,DISP=SHR
// SYSLIB DD   DSN=SYS1.CICS.SDFHCOB,DISP=SHR
//          DD   DSN=DATIT.COPY.COBOLE,DISP=SHR
//          DD   DSN=DATIT.BMS.ASM,DISP=SHR
//          DD   DSN=DGALT.COPY.COBOLE,DISP=SHR
//          DD   DSN=DGSYS.COPY.COBOLE,DISP=SHR
// SYSIN DD   DSN=&CICSOUT0,DISP=(OLD,DELETE)
// SYSLIN DD   DSN=&LSET,DISP=(NEW,PASS),SPACE=(CYL,(1,1)),
//          UNIT=SYSDA
// SYSPRINT DD  SYSOUT=*
// SYSUT1 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT2 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT3 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT4 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT5 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT6 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// SYSUT7 DD   SPACE=(CYL,(2,2)),UNIT=SYSDA
// *-----*
// *                LINKEDICAO                          *
// *-----*
// LKED      EXEC  PGM=IEWL,PARM='LIST,XREF,LET,CALL,MAP',
//          COND=(4,LT)

```

```
//SYSUT1 DD SPACE=(CYL,(1,1)),UNIT=SYSDA
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DSN=SYS1.CICS.SDFHLOAD,DISP=SHR
// DD DSN=SYS1.COB2LIB,DISP=SHR
// DD DSN=DATIT.CICS.LOAD,DISP=SHR
// DD DSN=DGALT.APPL.LOAD,DISP=SHR
// DD DSN=DGSYS.APPL.LOAD,DISP=SHR
//OBJPASS DD DSN=&LSET,DISP=(OLD,DELETE,DELETE)
//SYSLMOD DD DSN=DATIT.CICS.LOAD(&MEM),DISP=SHR
//SYSLIN DD DSN=DATIT.LKDOSDB2.DATA,DISP=SHR
```

3.4 Compilação Online com DB2

```
//OCDB2 PROC MEM=' '
//*-----*
//* ESTA PROC COMPILA PROGRAMAS ONLINE COM DB2 *
//* O PROGRAMA FONTE DEVE ESTAR NA BIBLIOTECA DATIT.DB2.COBOL *
//*-----*
//*-----*
//* TRANSLATE DB2 *
//*-----*
//PREDB2 EXEC PGM=DSNHPC,
// PARM='HOST(COB2),APOST,SOURCE,XREF,LEVEL(00)'
//STEPLIB DD DSN=SYS1.DB2.SDSNLOAD,DISP=SHR
//DBRMLIB DD DSN=DATIT.DBRM.OBJ(&MEM),DISP=OLD
//SYSLIB DD DSN=DATIT.COPY.COBOL,DISP=SHR
// DD DSN=DGALT.COPY.COBOL,DISP=SHR
// DD DSN=DGSYS.COPY.COBOL,DISP=SHR
//SYSIN DD DSN=DATIT.DB2.COBOL(&MEM),DISP=SHR
//SYSCIN DD DSN=&DSNHOUT,DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1)),
// DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120)
//SYSPRINT DD SYSOUT=*
//SYSTEM DD SYSOUT=*
//SYSUT1 DD SPACE=(CYL,(1,1)),UNIT=SYSDA
//SYSUT2 DD SPACE=(CYL,(1,1)),UNIT=SYSDA
//*-----*
//* TRANSLATE CICS *
//*-----*
//CICS EXEC PGM=DFHECP1$,PARM='COBOL2,SP'
//STEPLIB DD DSN=SYS1.CICS.SDFHLOAD,DISP=SHR
//SYSIN DD DSN=&DSNHOUT,DISP=(OLD,DELETE)
//SYSPRINT DD SYSOUT=*
//SYSPUNCH DD DISP=(NEW,PASS),DSN=&CICSOUT0,SPACE=(400,(4000,1000)),
// DCB=(LRECL=80,RECFM=FB,BLKSIZE=3120),UNIT=SYSDA
//*-----*
//* COMPILACAO COBOL II *
//*-----*
//COMP EXEC PGM=IGYCRCTL,COND=(4,LT),
// PARM=('LIB,RENT,APOST,LIST,XREF,NOOPT,NOSEQ',
// 'NODYNAM,OFFSET,FDUMP')
//STEPLIB DD DSN=GTADM.COB2.CICS,DISP=SHR
// DD DSN=SYS1.COB2COMP,DISP=SHR
//SYSLIB DD DSN=SYS1.CICS.SDFHCOB,DISP=SHR
// DD DSN=DATIT.COPY.COBOL,DISP=SHR
// DD DSN=DATIT.BMS.ASM,DISP=SHR
// DD DSN=DGALT.COPY.COBOL,DISP=SHR
// DD DSN=DGSYS.COPY.COBOL,DISP=SHR
//SYSIN DD DSN=&CICSOUT0,DISP=(OLD,DELETE)
//SYSLIN DD DSN=&LSET,DISP=(NEW,PASS),SPACE=(CYL,(1,1)),
// UNIT=SYSDA
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT2 DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT3 DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT4 DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT5 DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT6 DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//SYSUT7 DD SPACE=(CYL,(2,2)),UNIT=SYSDA
//*-----*
//* LINKEDICAO *
//*-----*
//LKED EXEC PGM=IEWL,PARM='LIST,XREF,LET,CALL,MAP,AMOD=31,RMOD=ANY',
// COND=(4,LT)
//SYSUT1 DD SPACE=(CYL,(1,1)),UNIT=SYSDA
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DSN=SYS1.CICS.SDFHLOAD,DISP=SHR
// DD DSN=SYS1.COB2LIB,DISP=SHR
// DD DSN=SYS1.DB2.SDSNLOAD,DISP=SHR
// DD DSN=DATIT.CICS.LOAD,DISP=SHR
// DD DSN=DGALT.APPL.LOAD,DISP=SHR
// DD DSN=DGSYS.APPL.LOAD,DISP=SHR
//OBJPASS DD DSN=&LSET,DISP=(OLD,DELETE,DELETE)
```

```
//SYSLMOD DD DSN=DATIT.CICS.LOAD(&MEM),DISP=SHR
//SYSLIN DD DSN=DATIT.LKDOCDB2.DATA,DISP=SHR
```

3.5 Execução de Programas Batch sem DB2

```
//PTITRCOB PROC
//RUNCOB EXEC PGM=&MEMBER
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DSNTRACE DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*,DCB=BLKSIZE=133
//SYSOUT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSDBOUT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
```

3.6 Execução de Programas Batch com DB2

```
//PTITRDB2 PROC
//RUNDB2 EXEC PGM=IKJEFT01,DYNAMNBR=20,REGION=4M
//SYSPUNCH DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DSNTRACE DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*,DCB=BLKSIZE=133
//SYSOUT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSDBOUT DD SYSOUT=*
//SYSTSIN DD DSN=TATIT.APPL.PARMLIB(&MEMBER),DISP=SHR
```

3.7 Ordenação de Ficheiros

```
//PTITSORT PROC
//SORT EXEC PGM=ICEMAN
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSOUT DD SYSOUT=X
//SORTWK01 DD DSN=&&SORTWK03,DISP=(NEW,DELETE,DELETE),
// SPACE=(CYL,(80,20),RLSE)
//SORTWK02 DD DSN=&&SORTWK04,DISP=(NEW,DELETE,DELETE),
// SPACE=(CYL,(80,20),RLSE)
//SYSIN DD DSN=TATIT.APPL.PARMLIB(&MEMBER),DISP=SHR
```

3.8 Execução de Bind Package

```
//BINDPKG PROC MEM=' '
//BIND EXEC PGM=IKJEFT01,DYNAMNBR=20 COND=(0,NE)
//STEPLIB DD DSN=SYS1.DB2.SDSNLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSIN DD DUMMY
//SYSTSIN DD DSN=DATIT.DB2.PKG.PLAN(&MEM),DISP=SHR
//
```

3.9 Execução de Bind Plan

```
//BINDPLN PROC MEM=' '
//*
//BIND EXEC PGM=IKJEFT01,DYNAMNBR=20 COND=(0,NE)
//STEPLIB DD DSN=SYS1.DB2.SDSNLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSIN DD DUMMY
//SYSTSIN DD DSN=DATIT.DB2.PLAN(&MEM),DISP=SHR
//
```

3.10 Load (Carregamento) de Tabelas

```
//PTITLOAD PROC
//LOAD EXEC PGM=DSNUTILB,REGION=2048K,PARM='DB2T'
//STEPLIB DD DISP=SHR,DSN=SYS1.DB2.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//UTPRINT DD SYSOUT=*
```



```
//SYSUDUMP DD SYSOUT=*
//SYSUT1 DD DSN=TATIT.SYSUT2.ATIT.DATA,DISP=(NEW,DELETE,CATLG),
//          SPACE=(CYL,(80,20),RLSE)
//SORTWK01 DD DSN=TATIT.SORTWK01.ATIT.DATA,DISP=(NEW,DELETE,DELETE),
//          SPACE=(CYL,(40,20),RLSE)
//SORTWK02 DD DSN=TATIT.SORTWK02.ATIT.DATA,DISP=(NEW,DELETE,DELETE),
//          SPACE=(CYL,(40,20),RLSE)
//SORTOUT DD DSN=TATIT.SORTOUT.ATIT.DATA,DISP=(NEW,DELETE,CATLG),
//          SPACE=(CYL,(80,20),RLSE)
//SYSMAP DD DSN=TATIT.SYSMAP.ATIT.DATA,DISP=(NEW,DELETE,CATLG),
//          SPACE=(CYL,(30,9),RLSE),DCB=(BUFNO=20)
//SYSERR DD DSN=TATIT.SYSERR.ATIT.DATA,DISP=(NEW,DELETE,CATLG),
//          SPACE=(CYL,(30,9),RLSE),DCB=(BUFNO=20)
//SYSIN DD DSN=TATIT.APPL.PARMLIB(&MEMBER),DISP=SHR
```

ANEXO - SYSTEM ABENDS

TSO

O TSO disponibiliza aos utilizadores, através de um terminal, um ambiente com as ferramentas adequadas ao desenvolvimento e gestão de aplicações e sistemas informáticos.

```

Menu  Utilities  Compilers  Options  Status  Help
-----
Option ==>
                                ISPF Primary Option Menu                                SYSB

1  Settings      Terminal and user parameters      User ID . : DB03041
2  View          Display source data or listings    Time. . . : 17:47
3  Edit          Create or change source data      Terminal. : 3278
4  Utilities     Perform utility functions         Screen. . : 1
5  Command       Enter TSO or Workstation commands Language. : ENGLISH
6  SDSF          System Display and Search Facility Appl ID . : ISP
                                           TSO logon : GRPSTD
                                           TSO prefix: DB03041
                                           System ID : SYSB
                                           MVS acct. : SYSACCT
                                           Release . : ISPF 4.1
  
```

Enter X to Terminate using log/list defaults

Capítulo 1 - PARÂMETROS DO TERMINAL E UTILIZADOR

```

Log/List  Function keys  Colors  Environ  GUI  Temporary  Help
-----
                                ISPF Settings
Command ==>

Select Options:
  Enter "/" to select option
  Command line at bottom
  / Panel display CUA mode
  Long message in pop-up
  Tab to action bar choices
  Tab to point-and-shoot fields
  / Restore TEST/TRACE options
  Session Manager mode
  / Jump from leader dots
  Edit PRINTDS Command

Print Graphics Params:
  Family printer type 2
  Device name . . .
  Aspect ratio . . . 0

General:
  Input field pad . . N
  Command delimiter . ;

Terminal Characteristics:
  Screen format  1  1. Data    2. Std    3. Max    4. Part
  Terminal Type  3  1. 3277    3. 3278    5. 3290A   7. 3278CF  9. 3278KN
                  2. 3277A    4. 3278A    6. 3278T   8. 3277KN
  
```

Executando o comando <KEYS> é possível personalizar as teclas de função nos vários menus do TSO.

```

----- Keylist Utility -----
File
-----
                                ISR Keylist ISRSAB Change                                Row 1 to 11 of 24
Command ==>                                                                Scroll ==> PAGE

Make changes and then select File action bar.

Keylist Help Panel Name . . .

Key      Definition      Format  Label
F1 . . .  HELP          SHORT  Help
F2 . . .  SPLIT         LONG   Split
F3 . . .  EXIT          SHORT  Exit
F4 . . .
F5 . . .
F6 . . .
F7 . . .  BACKWARD      LONG   Backward
F8 . . .  FORWARD       LONG   Forward
F9 . . .  SWAP          LONG   Swap
  
```

F10	.	.	ACTIONS	SHORT	Actions
F11	.	.			
F12	.	.	RETRIEVE	SHORT	Cancel
F13	.	.	HELP	SHORT	Help
F14	.	.	SPLIT	LONG	Split
F15	.	.	END	SHORT	End
F16	.	.	RETURN	SHORT	Return
F17	.	.	RFIND	SHORT	Rfind
F18	.	.	RCHANGE	SHORT	Rchange
F19	.	.	UP	LONG	Up
F20	.	.	DOWN	LONG	Down
F21	.	.	SWAP	LONG	Swap
F22	.	.	LEFT	SHORT	Left
F23	.	.	RIGHT	SHORT	Right
F24	.	.	CRETRIEV	SHORT	Cretriev

Capítulo 2 - VISUALIZAÇÃO DE DATA SETS

Menu RefList RefMode Utilities Help

View Entry Panel

Command ==>

ISPF Library:

```
Project . . . DATIT
Group . . . DB2
Type . . . COBOL
Member . . . (Blank or pattern for member selection list)
```

Other Partitioned or Sequential Data Set: 'DATIT.FTN047.DATA'

```
Data Set Name . . .
Volume Serial . . . (If not cataloged)
```

```
Initial Macro . . . . Enter "/" to select option
Profile Name . . . . Browse Mode
Format Name . . . . / Confirm Cancel/Move/Replace
Mixed Mode
```

Data Set Password . . (If password protected)

Capítulo 3 - EDIÇÃO DE DATA SETS

Menu RefList RefMode Utilities Help

Edit Entry Panel

Command ==>

ISPF Library:

```
Project . . . DATIT
Group . . . DB2
Type . . . COBOL
Member . . . PTNE255A (Blank or pattern for member selection list)
```

Other Partitioned or Sequential Data Set:

```
Data Set Name . . .
Volume Serial . . . (If not cataloged)
```

```
Initial Macro . . . . Enter "/" to select option
Profile Name . . . . / Confirm Cancel/Move/Replace
Format Name . . . . Mixed Mode
```

```
Data Set Password . . (If password protected) LMF Lock
3 1. Never
2. No
3. Yes
```

Esta opção disponibiliza-nos um editor de texto com várias funcionalidades:

Funcionalidade	Modo de Fazer
Inserir linhas para edição	Posiciona-se o cursor nas colunas 1 a 6 da linha imediatamente anterior e digita-se In em que n é o nº de linhas a inserir.
Gravar as alterações	Digita-se SAVE na linha de comando
Sair do editor	Pode sair-se com a tecla de função designada para o efeito <PF3> ou com CAN . No primeiro caso as alterações serão gravadas, no segundo caso serão desprezadas.

Copiar linhas para posterior colagem	Posiciona-se o cursor nas colunas 1 a 6 da linha a copiar, digita-se C e depois CUT (acumula a outros CUTs) ou CUT R (substitui outros CUTs) na linha de comando. Para colar a linha copiada, posiciona-se o cursor nas colunas 1 a 6 da linha imediatamente antes/depois à zona desejada, digita-se A/B e depois PASTE na linha de comando. Para copiar blocos, em vez de C, digita-se CC na primeira e última linha do bloco.
Copiar linhas para outra zona	Posiciona-se o cursor nas colunas 1 a 6 da linha a copiar e digita-se C. De seguida, posiciona-se o cursor nas colunas 1 a 6 da linha imediatamente antes/depois à zona desejada e digita-se A/B. Para copiar blocos, em vez de C, digita-se CC na primeira e última linha do bloco.
Mover linhas para outra zona	Posiciona-se o cursor nas colunas 1 a 6 da linha a mover e digita-se M. De seguida, posiciona-se o cursor nas colunas 1 a 6 da linha imediatamente antes/depois à zona desejada e digita-se A/B. Para mover blocos, em vez de M, digita-se MM na primeira e última linha do bloco.
Eliminar linhas	Posiciona-se o cursor nas colunas 1 a 6 da linha a eliminar e digita-se D. Para eliminar blocos, em vez de D, digita-se DD na primeira e última linha do bloco.
Repetir linhas	Posiciona-se o cursor nas colunas 1 a 6 da linha a repetir e digita-se Rn, em que n é o nº de repetições a fazer. Para repetir blocos, em vez de Rn, digita-se RRn na primeira e RR na última linha do bloco.
Mover o conteúdo das linhas para a direita	Posiciona-se o cursor nas colunas 1 a 6 da linha a deslocar e digita-se)n, em que n é o nº de caracteres a deslocar. Para deslocar blocos, em vez de)n, digita-se))n na primeira e)) na última linha do bloco.
Mover o conteúdo das linhas para a esquerda	Posiciona-se o cursor nas colunas 1 a 6 da linha a deslocar e digita-se (n, em que n é o nº de caracteres a deslocar. Para deslocar blocos, em vez de (n, digita-se ((n na primeira e ((na última linha do bloco.
Localizar termos	Digita-se na linha de comando F 'termo' ou F 'termo' N°Coluna (para localizar apenas os termos que se encontram na coluna indicada). Para localizar o termo seguinte usa-se a tecla de função apropriada <PF5>.
Substituir termos	Digita-se na linha de comando C 'termo' 'novo-termo' ou C 'termo' 'novo-termo' N°Coluna (para substituir apenas os termos que se encontram na coluna indicada). Para substituir o termo seguinte usa-se a tecla de função apropriada <PF6> ou, caso desejemos substituir todos os termos do documento, acrescenta-se a cláusula ALL ao comando a digitar.
Alterar numeração das linhas	Para numerar as linhas de um programa COBOL de acordo com a numeração standard, digita-se o NUM STD COB na linha de comando. Se desejarmos retirar a numeração da direita (nas colunas 73 a 80), digita-se UNNUM na linha de comando.
Trabalhar com letras minúsculas	Digita-se CAPS OFF na linha de comando.
Inserir caracteres numa linha	Usa-se a tecla <INSERT> do teclado. Se não for suficiente, digita-se NULLS ON na linha de comando.
Colocar régua	Posiciona-se o cursor nas colunas 1 a 6 de uma linha e digita-se COLS.
Retirar todas as informações que não sejam texto do documento	Digita-se RES na linha de comando.

Capítulo 4 - UTILITÁRIOS

Menu Help

Utility Selection Panel

Command ===>

1	<u>Library</u>	Compress or print data set. Print index listing. Print rename, delete, browse, edit or view members
2	<u>Data Set</u>	Allocate, rename, delete, catalog, uncatalog, or display information of an entire data set
3	<u>Move/Copy</u>	Move, copy, or promote members or data sets
4	<u>Dslist</u>	Print or display (to process) list of data set names. Print or display VTOC information
5	Reset	Reset statistics for members of ISPF library
6	Hardcopy	Initiate hardcopy output
7	ISPF C/S	Install ISPF C/S workstation code from MVS to your workstation.
8	Outlist	Display, delete, or print held job output
9	Commands	Create/change an application command table
*	Reserved	This option reserved for future expansion.
11	Format	Format definition for formatted data Edit/Browse
12	<u>SuperC</u>	Compare data sets (Standard Dialog)
13	<u>SuperCE</u>	Compare data sets and Search-for strings (Extended Dialog)
14	<u>Search-For</u>	Search data sets for strings of data (Standard Dialog)

4.1 Manipulação de Membros

```

Menu  RefList  Utilities  Help
-----
                                Library Utility

Option ==> R                                                    More:  +

blank Display member list                E Edit member
  C Compress data set                    V View member
  X Print index listing                  B Browse member
  L Print entire data set                D Delete member
  I Data set information                  R Rename member
  S Short data set information            P Print member

ISPF Library:
Project . . . DATIT
Group . . . DB2 . . . . .
Type . . . COBOL
Member . . . PTNE2550 (If B, D, E, P, R, V, or blank selected)
New name . . PTNE255A (If R selected)

Other Partitioned or Sequential Data Set:
Data Set Name . . .
Volume Serial . . . (If not cataloged)

```

4.2 Manipulação de Bibliotecas

```

Menu  RefList  Utilities  Help
-----
                                Data Set Utility

Option ==>

  A Allocate new data set                C Catalog data set
  R Rename entire data set                U Uncatalog data set
  D Delete entire data set                S Data set information (short)
blank Data set information                M Enhanced data set allocation
                                          V VSAM Utilities

ISPF Library:
Project . . DATIT
Group . . . DB2
Type . . . COBOL

Other Partitioned, Sequential or VSAM Data Set:
Data Set Name . . .
Volume Serial . . . (If not cataloged, required for option "C")

Data Set Password . . (If password protected)

```

Para alocar um novo DATA SET é usual obterem-se antes as informações de outro já existente que seja análogo. Para tal, escreve-se o nome do DATA SET existente, tecla-se <ENTER> e, uma vez no ecrã com as características do DATA SET, tecla-se <PF3>. De seguida, usa-se a opção de alocação para o novo DATA SET, alterando as características que desejarmos.

```

----- Data Set Information -----
|
| Command ==>
|
|                                                    More:  +
|

```

```

Data Set Name . . . . : DATIT.FTN047.DATA

General Data
Management class . . . : MCUSERS
Storage class . . . . : SC65
Volume . . . . . : BPG088
Device type . . . . : 3390
Data class . . . . . : DCSAM
Organization . . . . : PS
Record format . . . : FB
Record length . . . : 110
Block size . . . . : 27940
1st extent cylinder : 1
Secondary cylinders : 1
Data set name type. :

Current Allocation
Allocated cylinders : 1
Allocated extents : 1

Current Utilization
Used cylinders . . : 0
Used extents . . . : 0

Creation date . . : 1997/11/14
Expiration date . . : ***None***

```

4.3 Movimentação e Cópia de Membros

1º ecrã

Menu RefList Utilities Help

Move/Copy Utility

Option ==>

More: +

```

C Copy data set or member(s)      CP Copy and print
M Move data set or member(s)      MP Move and print
L Copy and LMF lock member(s)     LP Copy, LMF lock, and print
P LMF Promote data set or member(s) PP LMF Promote and print

```

Specify "From" Data Set below, then press Enter key

From ISPF Library:

```

Project . . . DATIT      (--- Options C, CP, L, and LP only ---)
Group . . . . DB2
Type . . . . COBOL
Member . . . PTNE255A    (Blank or pattern for member list,
                        "*" for all members)

```

From Other Partitioned or Sequential Data Set:

```

Data Set Name . . .
Volume Serial . . .      (If not cataloged)

```

2º ecrã

Menu RefList Utilities Help

COPY From DATIT.DB2.COBOL(PTNE255A)

Command ==>

More: +

Specify "To" Data Set Below

To ISPF Library:

```

Project . . . AATIT
Group . . . . DB2
Type . . . . COBOL
Member . . . PTNE255A    (Blank unless member is to be renamed)

```

To Other Partitioned or Sequential Data Set:

```

Data Set Name . . .
Volume Serial . . .      (If not cataloged)

```

Data Set Password . . . (If password protected)

To Data Set Options:

```

Sequential          Pack          Enter "/" to select option
Disposition 2 1. Mod Option 3 1. Yes Replace like-named
                2. Old           2. No PDS members
                3. Default

```

4.4 Listagem de DATA SETs

Data Set List Utility

Command ==>

```

blank Display data set list      P Print data set list
V Display VTOC information       PV Print VTOC information

```

Enter one or both of the parameters below:

Dsname Level . . . **DATIT.FTN047***
Volume serial . . .

Data set list options:

Initial View . . . 1	1. Volume	Enter "/" to select option
	2. Space	/ Confirm Delete
	3. Attrib	
	4. Total	

The following actions will be available when the list is displayed:

Enter a "/" on the data set list command field for command prompt pop-up.
Enter TSO commands, CLIST, REXX execs, or "=" to execute previous command.

4.5 Comparação de Membros

1º ecrã

Menu RefList Utilities Help

SuperC Utility

Command ==>

Specify "NEW" Data Set to be compared, then press the ENTER key.

Project . . . **DATIT**
Group . . . **DB2**
Type . . . **COBOL**
Member . . . **PTNE255A** (Blank or pattern for member selection list,
"*" for all members)

"NEW" Other Partitioned or Sequential Data Set:

Data Set Name . . .
Volume Serial . . . (If not cataloged)

Profile DSN

Data Set Password . . . (If password protected)

Mode . . 1	1. Foreground	Enter "/" to select option
	2. Batch	Mixed Mode

2º ecrã

Compare DATIT.DB2.COBOL(PTNE255A)

Command ==>

Specify "OLD" Data Set to be compared, then press the ENTER key.

Project . . . **AATIT**
Group . . . **DB2**
Type . . . **COBOL**
Member . . . **PTNE255A**

"OLD" other Partitioned or Sequential Data Set:

Data Set Name . . .
Volume Serial . . . (If not cataloged)

Data Set Password . . . (If password protected)

Listing Type DELTA	(DELTA/CHNG/LONG/OVSUM/NOLIST)
Listing DS Name . . . SUPERC.LIST	
Sequence Numbers . . .	(blank/SEQ/NOSEQ/COBOL)

4.6 Pesquisa em Membros

Menu RefList Utilities Help

Search-For Utility

Command ==>

Search String . . **DSN=TATIT**

Enter "/" to select option	Mode. . 1	1. Foreground
Specify additional search strings		2. Batch
Mixed Mode		

ISPF Library:

Project . . . **TATIT**
Group . . . **APPL**
Type . . . **CNTL**
Member . . . ***** (Blank or pattern for member selection list,


```

                                """ for all members)
Other Partitioned or Sequential Data Set:
  Data Set Name . . .
  Volume Serial . . .           (If not cataloged)

Listing Data Set . . . SRCHFOR.LIST
Data Set Password . . .       (If password protected)

```

Capítulo 5 - COMANDOS TSO

Menu List Mode Functions Utilities Help

```

-----
                                ISPF Command Shell
Enter TSO or Workstation commands below:

```

```

====> SEND 'ISTO E UM EXEMPLO' USER(DB09999) NOW

```

Place cursor on choice and press enter to Retrieve command

```

=> IND$FILE GET 'DATIT.DB2.COBOL(PTNE728A)' ASCII CRLF
=> IND$FILE PUT 'DATIT.FTN052.DATA' ASCII CRLF RECFM(F)

```

Capítulo 6 - VISUALIZAÇÃO DE JOBS

Uma vez submetidos os JOBS, é possível acompanhar a sua evolução. Assim:

- usa-se a opção I para visualizar os JOBS que ainda não iniciaram a sua execução
- usa-se a opção DA para visualizar os JOBS que estão em execução
- usa-se a opção H para visualizar os JOBS que terminaram a sua execução
- usa-se a opção O para visualizar os mapas produzidos pelos JOBS

```

V1R4M0 ----- SDSF PRIMARY OPTION MENU -----
COMMAND INPUT ==>                                SCROLL ==> CSR

```

Type an option or command and press Enter.

```

LOG      - Display the system log
DA       - Display active users of the system
I        - Display jobs in the JES2 input queue
O        - Display jobs in the JES2 output queue
H        - Display jobs in the JES2 held output queue

TUTOR    - Short course on SDSF (ISPF only)
END      - Exit SDSF

```

Licensed Materials - Property of IBM

5665-488 (C) Copyright IBM Corp. 1981, 1993. All rights reserved.
 US Government Users Restricted Rights - Use, duplication or
 disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

A lista de JOBS tem o seguinte aspecto:

```

SDSF HELD OUTPUT DISPLAY ALL CLASSES  1,294      LINES LINE 1-1 (1)
COMMAND INPUT ==>                                SCROLL ==> CSR
NP  JOBNAME  TYPE JNUM      TOT REC PROGRAMMER NAME      ST DATE ST TIME  END TIM
    DB03041Z JOB  25309      1,294                      97.331  17:35:31  17:38:3

```

Para restringir o aparecimento de JOBS nesta lista podem usar-se vários comandos:

- OWNER **nome_utilizador** faz aparecer apenas os JOBS do utilizador TSO referido
 - PRE **nome_job** faz aparecer apenas os JOBS com o nome referido
- (podem usar-se curingas para fazer aparecer um conjunto de JOBS)

Os spool dos JOBS desta lista pode ser visualizado ou destruído, através de comandos colocados à esquerda do JOB desejado. Assim:

- ? - Para listar os cartões de um JOB

S - Para visualizar um JOB ou um dos seus cartões

D e P - eliminam o spool do referido JOB

ENDEVOR

O ENDEVOR é um sistema de gestão de *software* que permite consultar, obter para alteração, compilar e armazenar as diferentes versões pelas quais o *software* vai passando até ser dado como pronto para execução nos ambientes de Produção.

Gestao de Software

```

XXXXXXXX  XX  XX  XXXXXXX  XXXXXXX  XX  XX  XXXX  XXXXX
XX      XXX  XX  XX  XX  XX  XX  XX  XX  XX  XX
XX      XX  X  XX  XX  XX  XX  XX  XX  XX  XX
XXXXX   XX  XXX  XX  XX  XXXX  XX  XX  XX  XX  XXXXX
XX      XX  XX  XX  XX  XX  XX  XX  XX  XX  XX
XX      XX  XX  XX  XX  XX  XX  XX  XX  XX  XX
XXXXXXX  XX  XX  XXXXXXX  XXXXXXX  X  XXXX  XX  XX

```

<enter>

----- ENDEVOR 3.7.2 Primary Options Menu -----
Option ===>

```

0  DEFAULTS      - Specify ENDEVOR ISPF default parameters
1  DISPLAY      - Perform Display functions
2  FOREGROUND   - Execute Foreground Actions
3  BATCH        - Perform Batch Action processing
4  BATCH PACKAGE - Perform Batch Package SCL Generation
T  TUTORIAL     - Display information about ENDEVOR
C  CHANGES    - Display summary of changes for this release of ENDEVOR
X  EXIT        - Exit the ENDEVOR/MVS dialog

```

Current environment: BDES

(C) 1987,1995 Computer Associates International, Inc.

Capítulo 1 - OPÇÕES DE VISUALIZAÇÃO

----- DISPLAY OPTIONS MENU -----
OPTION ===>

```

1  ELEMENT      - Display element/component list information
2  FOOTPRINT    - Display footprinted members and compressed listings
3  SITE         - Display site information
4  STAGE       - Display stage information
5  SYSTEM      - Display system definitions
6  SUBSYSTEM   - Display subsystem definitions
7  TYPE        - Display type definitions
8  PROCESSOR GROUP - Display processor group definitions
9  APPROVER GROUP - Display approver groups
A  RELATE GROUP - Display inventory area/approver group relationships
E  ENVIRONMENT - Display information about the current environment

```

1.1 Peças de Software (Fontes)

----- Display Elements/Component Lists -----
OPTION ===> B

blank - Display selection list B - Browse element current level

S - Display summary of levels C - Display changes current level
M - Display element master info H - Display history current level

Enter SX, BX, CX or HX to display component list information

```
FROM ENDEVOR:
ENVIRONMENT ===> BDES
SYSTEM       ===> FINANC
SUBSYSTEM    ===> TN
ELEMENT      ===> PTNE728A
TYPE         ===> COBII
STAGE        ===> 1

LIST OPTIONS:
DISPLAY LIST      ===> Y (Y/N)
WHERE CCID EQ     ===>
WHERE PROC GRP EQ ===>
DISPLAY SYS/SBS LIST ===> N (Y/N)
BUILD USING MAP   ===> N (Y/N)
1 - TU           2 - TI
```

1.2 Resultados de Compilações

```
----- ENDEVOR - FOOTPRINT DISPLAY -----
Option ===> L
```

blank - Member selection list
I - Display load module CSECTS and ENDEVOR footprints
L - Display the library member

```
FROM ISPF LIBRARY:
PROJECT ===>
LIBRARY ===>
TYPE ===>
MEMBER ===> THRU MEMBER ===>
```

```
OTHER PARTITIONED DATA SET:
DATA SET NAME ===> 'PBPRDSW.NDVR.TU.FINANC.RPT(PTNE728A)'
```

1.3 Ambientes

O ENDEVOR disponibiliza quatro ambientes de trabalho:

Ambiente	Finalidade
STAGE 1 : TU	<i>Software</i> em desenvolvimento
STAGE 2 : TI	<i>Software</i> em fase de testes integrados
STAGE 3 : CQ	<i>Software</i> em controlo de qualidade
STAGE 4 : PR	<i>Software</i> em Produção

Uma peça de *software* é inicialmente colocada no STAGE 1 e, para chegar ao STAGE 4, tem obrigatoriamente de passar por todos os STAGES intermédios.

```
----- STAGE INFORMATION -----
COMMAND ===>
```

```
CURRENT ENV ===> BDES
NEXT ENV: BPROD STAGE ID: 3
```

```
STAGE 1 INFORMATION:
ID: 1
Name: TU
Title: APPL - TEST UNITARIO
MCF data set name: PBSYSSW.NDVR.TU.DES.MCF
```

```
STAGE 2 INFORMATION:
ID: 2
Name: TI
Title: APPL - TEST PROJECTO
MCF data set name: PBSYSSW.NDVR.TI.DES.MCF
```

```
----- STAGE INFORMATION -----
COMMAND ===>
```

```
CURRENT ENV ===> BPROD
NEXT ENV: STAGE ID:
```

```
STAGE 1 INFORMATION:
ID: 3
Name: CQ
Title: APPL - CTR.QUALIDADE
MCF data set name: PBSYSSW.NDVR.CQ.PROD.MCF
```

```
STAGE 2 INFORMATION:
ID: 4
Name: PR
```

```
Title:          APPL - PRODUCAO
MCF data set name: PBSYSSW.NDVR.PR.PROD.MCF
```

1.4 Sistemas

As diversas aplicações de uma instituição estão integradas em sistemas do ENDEVOR.

```
----- SYSTEM SELECTION LIST ----- Row 1 to 8 of 8
COMMAND ===>                               SCROLL ===> CSR

CURRENT ENV:  BDES
NEXT      ENV:  BPROD

SYSTEM      SYSTEM TITLE
CLIENT      CLIENTES, BALANCOS E GARANTIAS
CONTAS      CONTAS, OP. EM ME, TELECOMPENSACAO E CHEQUES
DB2         GESTAO DE OBJECTOS DB2
DIVERS      MEIOS PAGAMENTO, TRANSFERENCIAS E APLIC. DIVERSAS
FINANC      PRODUTOS FINANCEIROS
GERAIS      SOFTWARE DE UTILIZACAO GERAL
GESTAO      CONTABILIDADE E INFORMACAO DE GESTAO
OPCRED      EMPRESTIMOS, CREDITO E LETRAS
```

1.5 Subsistemas

Cada um dos sistemas ENDEVOR pode ainda comportar várias sub-aplicações.

```
----- SUBSYSTEM SELECTION LIST ----- Row 1 to 3 of 3
COMMAND ===>                               SCROLL ===> CSR

CURRENT ENV:  BDES      SYSTEM:  FINANC
NEXT      ENV:  BPROD    SYSTEM:  FINANC

SUBSYSTEM SUBSYSTEM TITLE
AF         APLICACOES FINANCEIRAS
PF         PRODUTOS FINANCEIROS
TN         TITULOS
```

1.6 Tipos de Elementos

O ENDEVOR classifica as peças de *software* da seguinte forma:

Tipo	Objecto
COBII	Programas COBOL
BMS	Ecrãs
BK	<i>Copys</i>
DCLGEN	<i>Layouts</i> de Tabelas

1.7 Grupos de Processadores

A cada uma das quatro possíveis classes de programas corresponde um grupo de processador, e apenas este deve ser usado. Assim temos:

Processador	Classe de Programa
CIINBL2	Para Compilar programas Batch sem DB2
CIIDBL2	Para Compilar programas Batch com DB2
CIINCL	Para Compilar programas Online sem DB2
CIIDCL1	Para Compilar programas Online com DB2

De referir que os BMSs (por serem assemblados) e os *Copys* (por não sofrerem qualquer alteração) não têm grupo de processador associado.

```
----- PROCESSOR GROUP SELECTION LIST ----- Row 1 to 10 of 10
COMMAND ===>                               SCROLL ===> CSR

CURRENT ENV:  BDES      STAGE ID:  1  SYSTEM:  FINANC      TYPE:  COBII
NEXT      ENV:  BDES      STAGE ID:  2  SYSTEM:  FINANC      TYPE:  COBII

PROCESSOR
GROUP      PROCESSOR GROUP DESCRIPTION
```

```

CIIDBL   P DB2 BAT -NOOPT DYN NORENT * A(31) R(ANY)
CIIDBL1  M DB2 BAT -NOOPT DYN NORENT * A(24) R(24)
CIIDBL2 P DB2 BAT -NOOPT DYN NORENT * A(ANY) R(24)
CIIDCL   P DB2 CICS-NOOPT NODYN RENT * A(31) R(ANY)
CIIDCL1 P DB2 CICS-NOOPT NODYN RENT * A(31) R(ANY) DSNCLI
CIINBL   P BAT      -NOOPT DYN NORENT * A(31) R(ANY)
CIINBL1  M BAT      -NOOPT DYN NORENT * A(24) R(24)
CIINBL2 P BAT      -NOOPT DYN NORENT * A(ANY) R(24)
CIINCL  P CICS     -NOOPT NODYN RENT * A(31) R(ANY)
CIINCL1 P CICS     -NOOPT NODYN RENT * A(24) R(24)

```

Capítulo 2 - OPÇÕES FOREGROUND

```

----- Foreground Options Menu -----
Option ==>

```

```

1  DISPLAY   - Display an element
2  ADD/UPDATE - Add or update an element into stage 1
3  RETRIEVE  - Retrieve or copy an element
4  GENERATE  - Execute the Generate Processor for this element
5  MOVE      - Move an element to the next inventory location
6  DELETE    - Delete an element
7  PRINT     - Print elements, changes and detail change history
8  SIGNIN    - Explicitly sign-in an element

```

2.1 Passagem de Elementos para o ENDEVOR

Ao colocar no ENDEVOR (STAGE 1) os programas e BMSs, estes são, respectivamente, compilados e ensamblados. Eventuais erros que aconteçam neste processo podem ser visualizados tal como indicado em 1.2 Resultados de Compilações.

```

----- ADD/UPDATE ELEMENTS -----
OPTION ==> A

blank - Member list      A - Add an element      U - Update an element

TO ENDEVOR:
ENVIRONMENT ==> BDES
SYSTEM      ==> FINANC
SUBSYSTEM   ==> TN
ELEMENT     ==> PTNE255A
TYPE        ==> COBII
STAGE:      1
COMMENT     ==> ALTERADO PARA ...

ACTION OPTIONS:
CCID                ==>
GENERATE ELEMENT    ==> Y (Y/N)
DELETE INPUT SOURCE ==> Y
NEW VERSION         ==>
OVERRIDE SIGNOUT    ==> N (Y/N)
PROCESSOR GROUP     ==> *
UPDATE IF PRESENT   ==> Y (Y/N)

FROM ISPF LIBRARY:
PROJECT ==> DB03041
LIBRARY ==> PGM
TYPE    ==> SOURCE
MEMBER  ==> PTNE255A THRU MEMBER ==>

LIST OPTIONS:
DISPLAY LIST ==> Y (Y/N)

```

2.2 Obtenção de Elementos existentes no ENDEVOR

```

----- RETRIEVE ELEMENTS -----
OPTION ==> R

blank - Element list
R - Retrieve element

ELEMENT DISPLAY OPTIONS:
S - Summary  B - Browse  H - History
M - Master   C - Changes

FROM ENDEVOR:
ENVIRONMENT ==> BDES
SYSTEM      ==> FINANC
SUBSYSTEM   ==> TN
ELEMENT     ==> PTNE728A
TYPE        ==> COBII
STAGE       ==> 1          1 - TU          2 - TI
COMMENT     ==> PARA ALTERACAO DE ....

ACTION OPTIONS:
CCID                ==>
EXPAND INCLUDES     ==> N (Y/N)
SIGNOUT ELEMENT     ==> N (Y/N)
OVERRIDE SIGNOUT    ==> Y (Y/N)
REPLACE MEMBER      ==> Y (Y/N)

FROM ISPF LIBRARY:
PROJECT ==> DATIT
LIBRARY ==> DB2
TYPE    ==> COBOL
MEMBER  ==>

LIST OPTIONS:
DISPLAY LIST ==> Y (Y/N)
WHERE CCID EQ ==>
WHERE PROC GRP EQ ==>
BUILD USING MAP ==> N (Y/N)
FIRST FOUND     ==> Y (Y/N)

```

Para listar todas as versões anteriores de um elemento, usa-se a opção 'S'. De seguida pode usar-se qualquer uma das opções apresentadas no ecrã anterior.

2.3 Recompilação de Elementos no ENDEVOR

```
----- GENERATE ELEMENTS -----
OPTION  ==> G

                ELEMENT DISPLAY OPTIONS:
blank - Element list      S - Summary   B - Browse   H - History
G - Generate element      M - Master    C - Changes

FROM ENDEVOR:                ACTION OPTIONS:
ENVIRONMENT ==> BDES          CCID              ==>
SYSTEM       ==> FINANC      COPYBACK         ==> N (Y/N)
SUBSYSTEM    ==> TN          OVERRIDE SIGNOUT ==> N (Y/N)
ELEMENT      ==> PTNE255A    PROCESSOR GROUP  ==> *
TYPE         ==> COBII
STAGE        ==> 1          1 - TU              2 - TI

COMMENT      ==> RECOMPILADO POR ...

LIST OPTIONS:
DISPLAY LIST      ==> Y (Y/N)
WHERE CCID EQ     ==>
WHERE PROC GRP EQ ==>
BUILD USING MAP   ==> N (Y/N)
```

2.4 Movimentação de Elementos no ENDEVOR

Para que se possa passar um elemento para o STAGE 3, é necessário colocá-lo no STAGE 2.

```
----- MOVE ELEMENTS -----
OPTION  ==> O

                ELEMENT DISPLAY OPTIONS:
blank - Element list      S - Summary   B - Browse   H - History
O - Move element          M - Master    C - Changes

FROM ENDEVOR:                ACTION OPTIONS:
ENVIRONMENT ==> BDES          CCID              ==>
SYSTEM       ==> FINANC      SYNC              ==> Y (Y/N)
SUBSYSTEM    ==> TN          WITH HISTORY     ==> Y (Y/N)
ELEMENT      ==> PTNE255A    RETAIN SIGNOUT  ==> Y (Y/N)
TYPE         ==> COBII      SIGNOUT TO      ==>
STAGE        ==> 1          ACKNOWLEDGE ELM JUMP ==> N (Y/N)
                1 - TU      DELETE 'FROM' ELEMENT ==> Y (Y/N)
                2 - TI

COMMENT      ==> ALTERADO PARA ...

LIST OPTIONS:
DISPLAY LIST      ==> Y (Y/N)
WHERE CCID EQ     ==>
WHERE PROC GRP EQ ==>
```

2.5 Eliminação de Elementos no ENDEVOR

```
----- DELETE ELEMENTS -----
OPTION  ==> #

                ELEMENT DISPLAY OPTIONS:
blank - Element list      S - Summary   B - Browse   H - History
# - Delete element        M - Master    C - Changes

FROM ENDEVOR:                ACTION OPTIONS:
ENVIRONMENT ==> BDES          CCID              ==>
SYSTEM       ==> FINANC      OVERRIDE SIGNOUT ==> N (Y/N)
SUBSYSTEM    ==> TN          ONLY COMPONENT   ==> N (Y/N)
ELEMENT      ==> PTNE255A
TYPE         ==> COBII
STAGE        ==> 1          1 - TU              2 - TI

COMMENT      ==> ELIMINADO POR ...

LIST OPTIONS:
DISPLAY LIST      ==> Y (Y/N)
WHERE CCID EQ     ==>
```

```
WHERE PROC GRP EQ ==>
```

Capítulo 3 - OPÇÕES BATCH

```
BATCH ----- BATCH OPTIONS MENU -----
OPTION ==> 3

1 BUILD SCL - Build batch SCL actions
2 EDIT      - Edit request data set
3 SUBMIT    - Submit job for batch processing
4 VALIDATE  - Check request data set for syntax errors
5 BUILD JCL - Enter additional JCL to be included with the job

REQUEST DATA SET:
PROJECT ==> DATIT          APPEND      ==> N (Y/N)
GROUP   ==> FMMM          INCLUDE JCL ==> N (Y/N)
TYPE    ==> DADOS
MEMBER  ==> PROCURA

OTHER PARTITIONED OR SEQUENTIAL DATA SET:
DSNAME ==>

JOB STATEMENT INFORMATION:
==> //DB03041J JOB 'NDVR',MSGCLASS=X,NOTIFY=&SYSUID,CLASS=C,
==> //                MSGLEVEL=(1,1),REGION=4M
```

Todas as tarefas descritas no capítulo anterior podem (e devem) ser executadas em BATCH. Para o efeito deve ser usada a opção <BUILD SCL>. Esta opção conduz-nos através dos menus apresentados no capítulo anterior, por forma a que seja gerado o SCL apropriado à acção que desejamos. Uma vez gerado o SCL, podemos editá-lo (opção EDIT), verificar a sua correcção sintáctica (opção VALIDATE) e, finalmente, submeter o JOB (opção SUBMIT) para processamento BATCH.

Vejamos, por exemplo, um SCL para listar os elementos que contêm uma determinada palavra.

```
SET FROM ENVIRONMENT 'BPROD' SYSTEM 'FINANC' SUBSYSTEM 'TN'
TYPE 'COBII' STAGE '4'
.
SET WHERE TEXT ("PCTR0400" COLUMN 08 72)
.
SET TO SYSOUT
.
LIST ELEMENT 'PTNU*'
.
```

Mais concretamente, este SCL lista os programas COBOL do STAGE 4 cujo nome começa por <PTNU>, que contêm a palavra <PCTR0400>, e que pertencem ao ambiente BPROD, ao sistema FINANC e ao subsistema TN.