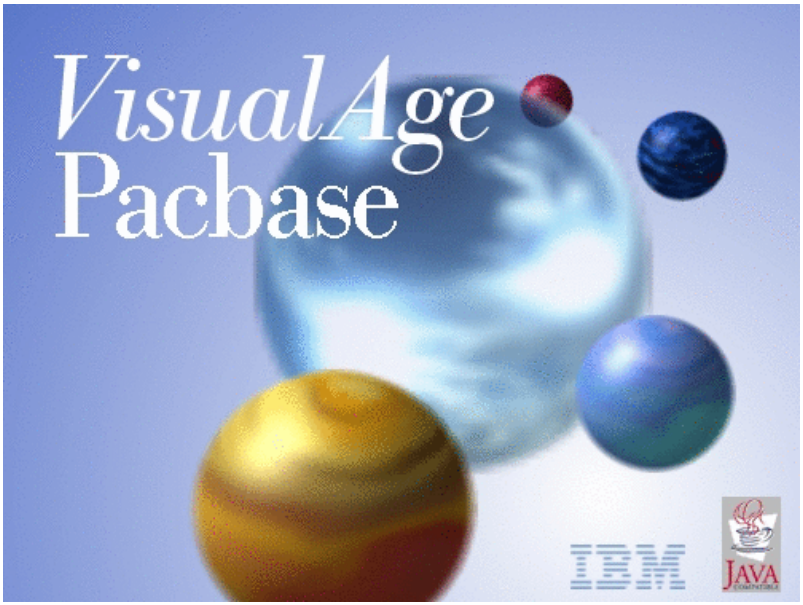
VisualAge Pacbase

Batch Applications

Version 3.5



VisualAge Pacbase

Batch Applications

Version 3.5

Note

Before using this document, read the general information under "Notices" on page v.

You may consult or download the complete up-to-date collection of the VisualAge Pacbase documentation from the VisualAge Pacbase Support Center at:

http://www.ibm.com/support/docview.wss?rs=37&uid=swg27005477

Consult the Catalog section in the Documentation home page to make sure you have the most recent edition of this document.

Second Edition (July 2006)

This edition applies to the following licensed programs:
- VisualAge Pacbase Version 3.5

Comments on publications (including document reference number) should be sent electronically through the Support Center Web site at: http://www.ibm.com/software/awdtools/vapacbase/support.html or to the following postal address:

IBM Paris Laboratory
1, place Jean–Baptiste Clément
93881 Noisy-le-Grand, France.

Contents

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk NY 10504–1785, U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Paris Laboratory, SMC Department, 1 place J.B.Clément, 93881 Noisy-Le-Grand Cedex. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

IBM may change this publication, the product described herein, or both.

Trademarks

IBM is a trademark of International Business Machines Corporation, Inc. AIX, AS/400, CICS, CICS/MVS, CICS/VSE, COBOL/2, DB2, IMS, MQSeries, OS/2, PACBASE, RACF, RS/6000, SQL/DS, TeamConnection, and VisualAge are trademarks of International Business Machines Corporation, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

All other company, product, and service names may be trademarks of their respective owners.

Chapter 1. Introduction

Purpose of the manual

The purpose of this manual is to describe the entire tire range of the entities managed by the Batch Systems Development function.

This manual is not a User's Guide or a textbook, but a reference document to be consulted for complete information concerning this function.

PREREQUISITES

For a basic knowledge of all the possibilities the system has to offer and specifically, the command language used to access the different screens, the user must consult:

.The Character Mode User Interface Guide,

.The Data Dictionary Manual,

.The Structured code Manual.

Principles of description

In this manual, the entities and screens managed by VisualAge Pacbase are described in two parts:

- An introductory comment explaining the purpose and the general characteristics of the entity or screen,
- A detailed description of each screen, including the input fields for on-line screens data entry into the Database.

For the description of batch input, refer to the 'Developer's Procedures' manual.

All on-line fields described in this manual are assigned an order number. These numbers are displayed on the screen examples which appear before the input field descriptions and allow for easy identification of a given field.

NOTE: If you use Developer workbench, refer to the on-line Help.

NOTE: If you use the VisualAge Pacbase WorkStation, refer to the 'WorkStation User Interface' guide which documents the corresponding windows.

Batch Systems Development Function

The purpose of the Batch Systems Development (BSD) function is to describe and generate batch systems.

The general principle is to describe the batch procedures that are most often used:

- File access,

- Loading of tables,

- Data validation,

- Updates,

- Reports.

From the description of these procedures, the BSD function ensures the generation of the corresponding programs. All programs have the same structure, which contains all or some of the procedures described above.

GENERAL DESCRIPTION

Each batch procedure is described as to what can be done automatically.

Specific procedures are described in functions written in Structured Code (refer to the corresponding manual).

The BSD function automatically generates the following:

- File retrieval, especially sequential files, with synchronization and control break detection; the matching and control break criteria are indicated when the file is called in a program,
- Automatic loading of files into program tables,
- Validation of transactional information in the batch input stream. This is done by adding information on the segment description made during the analysis phase. Validations include presence, class, and value validations (coding, tables, etc.),
- Update of permanent data of the system accomplished by conditional substitution, subtraction or addition, following the same principle as that adapted for validation processing,
- Report printing. This is accomplished with the description of a report layout, as it will be seen by the end-user. This will assist in determining

both the report composition (headings, detail lines, page bottom, etc.) and the structure of the output (data elements making up each line, position in the line, source, condition, etc.).

The coding of the report is accomplished using the layout. There will be no difference between the layout and the report once it is programmed.

Report printing automatically generates the processing of totals, to be executed at each control-break.

GENERATION

Once the above data is defined, the VisualAge Pacbase system ensures:

- The automatic generation of batch COBOL programs containing one or more of the procedures described above,
- The ability to generate and incorporate additional functional procedures that have not been taken into account. These additional procedures must be written in Structured Code.

Therefore, these programs are completely generated in COBOL.

CROSS-REFERENCES

The Batch Systems Development function is used in conjunction with the Specifications Dictionary and Structured Code functions, and benefits from all the advantages associated with them (keywords, cross-references, documentation, use of macro-structures, etc.).

Managed entities

All VisualAge Pacbase information is grouped into homogeneous families called ENTITIES.

Entities are made up of one or more associated screens. The three basic types of screens are:

- DEFINITION,

- DESCRIPTION,

- DOCUMENTATION.

Each screen is made up of fields. Definition screens define a single ″line″ whereas the other two may contain more than one line. Certain fields function as keys to these lines.

The entities managed by the BSD function are the following:

. Programs,

. Reports.

The automatic generation of BSD procedures is obtained from data structure and report calls in the programs:

- The Definition screen of a program determines the repetitive structure characteristic of a batch procedure,
- Data from the Program Call of Data Structures Screen (-CD) provokes the generation of file retrieval functions: open, read, detection of control breaks, file matching, write and close,
- Validation and update processing are generated from the definition and description of segments,
- Print procedures are generated from the definition and description of reports.

The Structured Code also allows to:

- Add work and linkage areas (-W),
- Complete or modify the beginning of the program (-B),
- Add specific procedures (-P).

REVERSE ENGINEERED PROGRAMS

Programs that have been "reverse engineered" include only the following:

- Work Area (-W) lines,
- Source Code (-SC) lines (COBOL source code).

It is possible to add Structured Code (-W and -P lines) and Calls of Marcro-Structures (-CP lines) to these programs, and then regenerate them. Call of Data Structures (-CD) and Beginning Insertions (-B) lines are ignored.

Chapter 2. Programs

Definition (P)

The purpose of the 'Program' entity is to develop and implement all procedures defined in the detailed analysis phase.

GENERAL CHARACTERISTICS

The Program entity contains:

- A Definition, required, giving general characteristics (Program code on six characters, keywords, Type of COBOL to generate, etc.),
- Comments entered on the Comments screen or batch form providing useful data related to the program (programmer's name, etc.),
- Several types of description lines:
  - Call of Data Structures lines make up the Data Division and most of the Procedure Division in the generated program,
  - Beginning Insertions lines, allowing the user to modify the Environment Division up to and including the 'Data Division' and 'File Section' statements,
  - Work Area lines used to supplement the DATA DIVISION, procedures" manual.
  - Call of PMS lines used to call pre-defined macros into the program.

NOTE

For more information concerning Beginning Insertions, Procedural Code, Work Areas, and Parameterized Macro- Structures, see the Structured Code Manual.

The Batch codes are to be found in the "Developer's Procedures" Manual.

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
PROGRAM CODE.............. PO0001     1

PROGRAM NAME.....................: VENDOR REPORTS  2

CODE FOR SEQUENCE OF GENERATION....: PO0001          3

TYPE OF CODE TO GENERATE...........: 0    4
COBOL NUMBERING AND ALIGNMENT OPT..:      5
CONTROL CARDS IN FRONT OF PROGRAM..: B    6
CONTROL CARDS IN BACK OF PROGRAM...: B    7
COBOL PROGRAM-ID...................: PO0001        8
MODE OF PROGRAMMING................: P    9
TYPE AND STRUCTURE OF PROGRAM......: B   10
PROGRAM CLASSIFICATION CODE........: P   PROGRAM  11
TYPE OF PRESENCE VALIDATION........:     12
SQL INDICATORS GENERATION WITH '-'.:     13

EXPLICIT KEYWORDS..:  14
UPDATED BY.........:            ON :           AT :   :  :    LIB :
SESSION NUMBER.....: 0059       LIBRARY......: CIV    LOCK....:

O: C1 CH: Ppo0001                  ACTION:
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
1	6		PROGRAM CODE (REQUIRED)
			Code identifying the program in the library.
2	30		PROGRAM NAME (REQUIRED IN CREAT)
			It must be as explicit as possible since the implicit keywords are created from this name.
3	6		CODE FOR SEQUENCE OF GENERATION
			Default option: PROGRAM CODE in the VisualAge Pacbase Library.
			Programs are sorted on this code in the generated program stream.
4	1		TYPE OF COBOL TO GENERATE
			Specifies the COBOL variant for the generated Program.
			The default value at creation is the value of the GENERATED LANGUAGE field in the Library Definition.
			Compatibility of Programs generated with Cobol 85, Cobol II, Cobol/370, Cobol OS/390 operates according to the value of the GENERATED LANGUAGE in the Library.
		'N'	No adaptation to a language variant.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			It is used to prevent program generation.
		'0'	IBM MVS/ESA OS/390
		'1'	IBM DOS/VSE
		'3'	UNIX, WINDOWS
		'4'	BULL GCOS7
		'5'	BULL GCOS8
		'8'	UNISYS A Series
		'F'	TANDEM
		'I'	DEC/VAX VMS
		'K'	ICL 2900
		'O'	AS/400
		'U'	UNISYS 2200 Series
		'X'	IBM MVS/ESA OS/390
		'Q'	ACUCOBOL
		'C'	Extraction of COBOL Source Code. (Refer to chapter 'Appendix: Pure COBOL Source Code' in the 'Structured Code' manual).
5	1		COBOL NUMBERING AND ALIGNMENT OPTION
			This option can be used to suppress numbering or the identification of a program or to modify the justification of the generated program lines.
		blank	Numbering, justification and identification of program in accordance with the standard COBOL line (default value).
		'1'	Suppression of numbering.
		'2'	Suppression of numbering and justification of statements (columns 8 to 71 inclusive) in column 1.
		'3'	Standard numbering and justification, suppression of program identification.
		'4'	Suppression of numbering and program identification.
		'5'	Suppression of numbering and of program identification justification of instructions (columns 8 to 71 inclusive) in column 1.
6	1		Control cards in front of programs
			Enter the one-character code that identifies the job card to be inserted before the generated program.
			Default: Code entered on the Library Definition Screen

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
7	1		CONTROL CARDS IN BACK OF PROGRAMS
			Enter the one-character code that identifies the job card to be inserted after the generated program.
			Default: Code entered on the Library Definition Screen
8	8		COBOL PROGRAM-ID
			(Default value at generation: CODE FOR SEQUENCE OF GENERATION.)
			This code identifies the generated program:
			.in the IDENTIFICATION DIVISION,
			.in a source module library,
			.in the library of executable modules.
			This code intervenes (totally or partially) in the job control language lines generated before or after the program.
9	1		MODE OF PROGRAMMING
			Structured Code
		'P'	Default value when creating a Library. Programming in Structured Code on '-P' lines (Procedural Code).
			Cobol generator (in conjunction with the Reverse Engineering function)
		'S'	Specific procedures composed of Source Code (-SC) and Procedural Code (-P).
			With this value, the Type and structure of Program field must also be 'S'.
		'8'	Programming with '-8' type of lines.
			Used only to maintain applications written with former VisualAge Pacbase versions.
			The value entered on the Definition line of the Library is channeled down by default to the Definition line of a Program when it is created.
			At the Program level, the programming type can be modified.
			The combination of '-P' and '-8' lines called in the same Program, either directly, or via Macro-structures, is rejected.
10	1		TYPE AND STRUCTURE OF PROGRAM
			This identifies the structure of the generated Program or the type of the Program in the Library.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'B'	Structure of a batch Program (default option).
			It provides the general structure of an iterative program:
			.beginning of the loop (F05),
			.end of run (F20),
			.end of the loop (F9099. GO TO F05).
		'S'	Suppress automatic structure generation
			STRUCTURED CODE FUNCTION
			This type can be used to describe the TDS 'system generation', the IDS II 'schema', ...
			.suppression of COBOL divisions,
			.the program is made up of Beginning Insertions
			(-B), Work Areas (-W) and Call of Data Structures (-CD) lines.
			COBOL GENERATOR FUNCTION
			.the program is made up of '-W', -P', '-SC' and '-CP' lines.
		'T'	On-line Program structure.
			Suppression of the loop, i.e:
			.no beginning of loop (F05),
			.no end of run (F20),
			.no end of loop (F9099. GO TO F05).
		'C'	C.I.C.S. on-line Program structure.
			Suppression of the loop, i.e:
			.no beginning of loop (F05),
			.no end of run (F20),
			.no end of loop (F9099. GO TO F05).
			Same as 'T' but also with:
			.generation, at the beginning of the PROCEDURE DIVISION, of the line: MOVE CSACDTA TO TCACBAR,
			.generation in F9099 of: DFHPC TYPE=RETURN,
			.no line numbering in the generated program.
		'M'	Parameterized Macro-Structure type. (For documentation purposes only).
			This is used for programs to be inserted into other programs. This type of program cannot be generated alone.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'F'	Program composed of Call of Data Structures (-CD) and Pure COBOL Source Code (-9) lines.
			This option permits the manipulation of the Pure COBOL Source Code (-9) lines that invoke the structural description of the automatically generated D.S.'s, according to the characteristics assigned to that D.S. on the Call of Data Structures (-CD) screen.
			For more information see chapter 'Appendix: Pure COBOL Source Code' in the 'Structured Code' Manual.
		'D'	Program composed of Call of Data Structures (-CD), Beginning Insertions (-B), Work Areas (-W) and Pure COBOL Source Code (-9) lines. This option provides the automatic generation of the IDENTIFICATION, ENVIRONMENT and DATA DIVISIONS.
			The PROCEDURE DIVISION is written entirely on Pure COBOL Source Code (-9) lines.
		'P'	Program composed of Call of Data Structures (-CD), Beginning Insertions (-B), Work Areas (-W) and Procedural Code (-P) lines. This option provides the automatic generation of the IDENTIFICATION, ENVIRONMENT and DATA DIVISIONS. The PROCEDURE DIVISION is entirely written in Structured Code.
		'Y'	Program written in C LANGUAGE and composed of Work Areas (-W), Source Code (-SC) and Call of P.M.S.'s (-CP) lines.
11	1		PROGRAM CLASSIFICATION CODE
			This value is used primarily for documentation purposes. The label corresponding to the selected code will be displayed on Reports and Screens.
			It is also used to select the non-expansion option for Macro-Structures.
		'A'	TP System
		'D'	Sub-program
		'G'	Screen map
		'M'	Macro-structure
		'N'	Non-expanded Macro-Structure
		'P'	Program
		'S'	Schema
		'T'	On-line Program (Screen)

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'U'	Utility
		'V'	Sub-schema
12	1		TYPE OF PRESENCE VALIDATION
			In validation Programs, the presence of numeric Data Element will be determined according to this code:
			For numeric fields:
		blank	Field present if not blank (default value).
		'0'	Field present if not zero.
			For alphabetic and numeric fields:
		'L'	Field present if not low-value.
13	1		SQL INDICATORS GENERATION WITH '-'
			Cross-references available for the use of SQL indicators in Structured Language.
		BLANK	SQL indicators generated in the format: VXXNNCORUB:
		'-'	SQL indicators generated in the format: V-XXNN-CORUB.
14	55		EXPLICIT KEYWORDS
			This field allows you to enter additional (explicit) keywords. By default, keywords are generated from the instance's name (implicit keywords).
			Keywords must be separated by at least one space. Keywords have a maximum length of 13 characters which must be alphanumeric. However, '=' and '*' are reserved for special usage and are therefore ignored in keywords.
			Keywords are not case-sensitive: uppercase and lower-case letters are equivalent.
			NOTE: Accented and special characters can be declared as equivalent to an internal value in order to optimize the search of instances by keywords (Administrator workbench, 'Window' menu, 'Parameters browser' choice, in 'Special Characters' tab).
			A maximum of ten explicit keywords can be assigned to one entity. For more details, refer to the 'Character Mode User Interface' guide, chapter 'Search for Instances', subchapter 'Searching by Keywords'.

Call of Data Structures (-CD)

The purpose of the Call of Data Structures is to identify all Data Structures used in a Program, specifying their physical characteristics as well as the way these files are to be used in the Program.

The Call of Data Structures screen is accessed by entering '-CD' in the CHOICE field from any screen within the Program entity's network.

GENERAL CHARACTERISTICS

Each Data Structure may be described on as many continuation lines as needed. Certain information must be entered on the first line of the call, as opposed to being entered on a continuation line, and vice versa.

The system assigns default values to required information areas of the Data Structure call line. By default, a Data Structure will look like a sequential file with fixed-length records. The Data Structure Description will contain all of the Data Structure records, with the Data Elements in internal format, without the optional Data Elements.

ORGANIZATION

Data Structures are 'organized' into three basic types:

. Standard Files,

. Database Blocks,

. Work Areas or Linkage Areas.

The descriptions of the latter category may involve specifying Data Structures and/or Data Elements.

It is preferable to define the WORK or LINKAGE fields on the screen provided for this purpose (-W). If the Program is a Macro-Structure (P.M.S.), the '-W' is generated in the calling Program, not the '-CD'.

NOTE: A Data Structure call in the -W screen does not allow for the creation of continuation lines (which limits the number of Segment selections to four Segments, for example).

Also, utilization, control breaks, and file matching cannot be specified on -W lines.

AUTOMATIC PROCESSING OPTIONS

The user identifies the data structures used in the program, providing their:

- Physical characteristics (external name, organization, access, blocking factor, etc.),
- File matching criteria, controlled by three different fields (for input data structures):
  - SORT KEY, which identifies the keys to match on, arranged hierarchically from the major-most key,
  - NUMBER OF CONTROL BREAKS, which specifies how many control breaks there are,
  - FILE MATCHING LEVEL NUMBER, which specifies the number of levels to match.
- The RECORD TYPE / USE WITHIN D.S.: Several description variants may be defined from the data structure descriptions contained in the VA Pac database.

  These variants are:
  - The format type used,
  - The selection of certain segments, taken from the various data structure descriptions in the library,
  - The selection of certain reserved data elements or groups of data elements,
  - The record description mode (redefined or not, repeated, etc.), and the COBOL level number,
  - The location of the generated description in the DATA DIVISION (this location can vary from one record to another),
  - The type of use of the data structure, controlling generation of certain specific procedures (table loading, validation, updating, etc.).

LIMITATIONS

There is no limit for the number of data structure calls per program. However, principal data structures, or data structures with control breaks or file matching must appear among the first 23. If not, file matching might not be carried out as desired and the updating of these principal data structures will not take place.

For I-, V-, or S-organization files, the number of call lines must not exceed 100.

The maximum number of times a single data structure can be called is limited to 500, for all the programs that are generated in one run.

FILE RETRIEVAL

It is generated according to the file matching and control break criteria indicated on the -CD line.

To have an example of how it works and how the corresponding matching (XX-CFn), File Break (XX-IBn, XX-FBn), Total break (ITBn, FTBn), Update occurrence (XX-OCn) variables are managed, refer to the Chapter 'Example of generated program' at the end of the 'Batch Applications' manual.

COMPOSITE DATA STRUCTURES

It is possible at the Program level to build a Data Structure with Segments belonging to different Data Structures.

This is accomplished by assigning the same DATA STRUCTURE CODE IN THE PROGRAM to different Data Structures, and selecting the desired Segments from each.

The common part will be made of the code of the Data Structure called on the first line.

In order to call in a Program Data Structure two or more Segments which have the same two-character SEGMENT CODE or the same LAST CHARACTER OF THE REPORT CODE, but are extracted from different Data Structures in the Library, it is necessary to change the code of one of them in the Program, in the SELECTION field.

```
                    PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
DATA STRUCTURES USED IN PROGRAM : 1  VRPREP VENDOR RATING PREPARATION

2 3  4    5  6      7 9 11      13 14  16   18 19     21 22        23  25  27
                     8 10 12         15  17      20                  24  26
A DP CO : DL EXTERN OARFU BLOCK T  B M U RE SE L UNIT C  SELECTION F E R L PL
  CO    : CO PMSCO  SSFOU     0 R      D                            I     1
        :    STAT.FLD: 28          ACC. KEY: 29          RECTYPEL 30
  OI    : OI PMSPOF VSFID     0 R  1   C                            I     1
        :    STAT.FLD:             ACC. KEY:             RECTYPEL
  SO    : CO SORT   SSFTU     0 R      D                            I     1
        :    STAT.FLD:             ACC. KEY:             RECTYPEL
  WO    : CO WORK   WSFOU     0 R      D                            I     1
        :    STAT.FLD:             ACC. KEY:             RECTYPEL
        :
        :    STAT.FLD:             ACC. KEY:             RECTYPEL
        :
        :    STAT.FLD:             ACC. KEY:             RECTYPEL
        :
        :    STAT.FLD:             ACC. KEY:             RECTYPEL
        :
        :    STAT.FLD:             ACC. KEY:             RECTYPEL

O: C1 CH: -CD
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
1	6		PROGRAM CODE (REQUIRED)
			Code identifying the program in the library.
2	1		ACTION CODE
		'C'	Creation of the line
		'M'	Modification of the line
		'D' or 'A'	Deletion of the line
		'T'	Transfer of the line
		'B'	Beginning of multiple deletion
		'G'	Multiple transfer
		'?'	Request for HELP documentation
		'E' or '-'	Inhibit implicit update
		'X'	Implicit update without upper/lowercase processing
3	2		DATA STRUCTURE CODE IN THE PROGRAM (REQUIRED)
			This code establishes the sequence in which the Data Structure will be processed in the Program.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			The first character must be alphabetic but the second one can be numeric or alphabetic.
			It is recommended to keep the same DATA STRUCTURE CODE IN THE PROGRAM and IN THE LIBRARY when the Data Structure described in the Library is used only once in the Program.
4	2	ALPHA.	Continuation of D.S. Description
		blank	First line of a Data Structure description. This line must contain all information defining the input-output characteristics, all technical characteristics and the description of the Data Structure.
			Two-letter code indicating a continuation line.
			The continuation lines are used to select the records of the different Data Structures in the Library and to request their description in a specified position.
5	2		DATA STRUCTURE CODE
			This code is made up of two alphanumeric characters. This is a logical code internal to the Database and therefore independent of the names used in Database Blocks and Programs.
6	6		EXTERNAL NAME OF THE FILE
			(Default option: DATA STRUCTURE CODE IN THE PROGRAM.)
			(NOTE: In this discussion, the term 'COBOL Variant' = the value in the TYPE OF COBOL TO GENERATE field)
			FOR the 'Y' ORGANIZATION:
			This field must contain the code of the COBOL COPY clause which represents the communication area of the Pacbench C/S Application Component which accesses the Logical View. For more details, refer to the 'Pacbench C/S Applications - Business Logic' Manual.
			FOR SQL ORGANIZATIONS:
			This field must contain the VisualAge Pacbase code of the SQL block.
			For explanations, refer to the 'Structured Code'
			manual, chapter 'Modifying the Procedure Division', subchapter 'Procedural Code Screen (-P)', and to the 'SQL Databases' manual, chapter 'SQL Accesses', Sub-chapter 'Customized SQL Accesses'.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			FOR ALL THE OTHER ORGANIZATIONS:
			IBM OS/390 (variant X): DDNAME in 1 to 6 positions.
			COBOL II IBM VS2 (Variant X): The ASSIGN clause (for sequential files, 'S' organization) with SYSnnn as external name is generated in the following form:
			SYSnnn-UT-....-S-SYSnnn
			IBM DOS (COBOL Variant 1), three forms:
			.SYSnnn Symbolic unit name.
			.xxxnnn Specifies at the same time the symbolic unit name and the external name of the Data Structure.
			.xxxxxx External name. The symbolic unit is generated with SYSnnn, nnn being incremented by one for each Data Structure starting with SYS010.
			BULL Gcos7 (COBOL Variant 4):
			.INTERNAL-FILE-NAME in 1 to 6 position.
			BULL Gcos8 (COBOL Variant 5):
			.File code (2 characters). UNISYS A Series (COBOL Variant 8):
			.nnppp numeric, generate AREA nn, AREASIZE pppp.
			TANDEM (Variant F): external name in 1 to 6 positions.
			DEC/VMS (COBOL Variant I): external name in 1 to 6 positions.
			PHYSICAL CHARACTERISTICS OF FILE
7	1		ORGANIZATION
		'S'	Sequential (Default value).
		'I'	Indexed sequential
		'V'	VSAM (IBM), UFAS (BULL), etc.
			Generates the STATUS KEY IS clause and the corresponding field is declared in the STATUS FIELD: VSAM FILE INDICATOR field.
			The file is considered sequential if the name of the key in the record is absent; it is considered indexed if the key name is entered.
		'W'	File descriptions are generated in WORKING-STORAGE before the constant 'WSS-BEGIN'.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			A Data Structure thus described will be used like a work area or processed through a function of a generalized management system (Database in particular).
		'L'	Identical to 'W' except that the user may choose the description location (See CODE FOR COBOL PLACEMENT).
		'X'	Data Structure used as a comment, not used for generation.
		'G'	Table description.
			Generates the communication area with the access module.
		'Y'	Call of the COPY clause which corresponds to the communication area between the client and the server (Pacbench C/S Business Components only).
			For details, refer to the 'Pacbench C/S Applications - Business Logic' Manual.
			DATABASES
			The values of the following codes are reserved for Database Descriptions when the Database Description function is not used. These values are taken into account by application programs.
		'D'	Reserved for the Description of Segments or records of the different Databases, IMS (DL/1), IDS II, (according to the TYPE OF COBOL TO GENERATE selected), in the generation of DBD, SYSGEN, schemas or application Programs (according to the TYPE AND STRUCTURE OF PROGRAM selected).
		'B'	Reserved for the description of records for an IDMS Database in the sub-schemas or application programs.
		'A'	Reserved for an ADABAS file description in the definition programs or usage programs of the Database.
		'T'	Reserved for the description of 'TOTAL' files in the definition programs or the usage programs of the Database.
		'Q'	Reserved for the description of SQL/DS, DB2/2 or DB2/6000 Databases (IBM), or
			ALLBASE/SQL Databases (HP3000), or
			DB2/2 or DB2/600 Databases (MICROFOCUS).
		'2'	Generation-Description of a DB2 or VAX/SQL Segment. Only physical accesses are not generated. The structure of variable indicators corresponding to the columns of the DB2 or VAX/SQL table is always generated.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'C'	Reserved for the description of an INTEREL RDBC, RFM Database Structure.
		'O'	Reserved for the description of an ORACLE (< V6) Database Structure.
		'P'	Reserved for the description of an ORACLE (V6 and V7) Database Structure.
		'R'	Reserved for the description of an RDMS Database Structure.
		'4'	Reserved for the description of a DB2/400 Database Structure.
		'N'	Reserved for the description of a NONSTOP SQL Database Structure.
		'M'	Reserved for the description of a DATACOM DB Database Structure.
		'9'	Reserved for the description of an INFORMIX, SYBASE, INGRES/SQL, and SQL SERVER Database Structure.
			The use of the System with the different DBMS's is documented in specific 'Database Description' manuals.
8	1		Access mode
		'S'	Sequential (default option).
		'R'	Random - Direct (indexed sequential organization only).
		'D'	Dynamic (VSAM files only - 'V' organization)
9	1		RECORDING MODE
		'C'	For 'P'-type organizations (Oracle V6 and V7) and '9'-type organizations (Sybase): Automatic generation of CONNECT AT Database, DECLARE Database and access SQL AT Database.
		'F'	Fixed (default option).
			At generation time, the lengths of the different records are aligned with the length of the longest record.
		'V'	Variable.
		'U'	Undefined.
		'S'	Spanned (Reserved for IBM MVS and DOS variants).
10	1		FILE TYPE - INPUT / OUTPUT
		'I'	Input file - Default option with the following values of USAGE OF DATA STRUCTURE: 'C', 'T', 'X', 'M', 'N' 'P'. This value is prohibited with all other USAGEs.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'O'	Output file - Default option with the following values of USAGE OF DATA STRUCTURE: 'D', 'S', 'R', 'E', 'I' and 'J'. This value is prohibited for all other USAGEs.
		'E'	Output file. Generation of an OPEN EXTEND clause
		'T'	Sort (on Input or Output, depending on the USAGE OF DATA STRUCTURE value).
		'R'	Input-Output (direct access Data Structures only).
11	1		UNIT TYPE
		'U'	Magnetic storage with sequential access.
			Default value.
		'D'	Magnetic memory with selective access.
			Direct access device.
		'R'	Slow peripherals (Card punch reader, printer).
			This parameter is important for the TYPEs OF COBOL TO GENERATE variant for which the "ASSIGN" clause, the FD level or the WRITE statements depend on the UNIT TYPE.
12	5	NUMER.	BLOCK SIZE SPACES AND ZEROES ARE EQUIVALENT
			PURE NUMERIC FIELD
			(Note: In this discussion the term 'COBOL Variant' = the value in the TYPE OF COBOL TO GENERATE field)
		0	Default value.
		2	The blocking factor can be zero in the following cases:
			. IBM OS (COBOL variant 0) except for indexed organization files.
			. IBM MVS. The BLOCK CONTAINS clause is generated for a VSAM file only if the library is in COBOL II.
			The corresponding COBOL clause (BLOCK CONTAINS) is not generated in the following cases:
			.sort file,
			.disk Data Structure (file stored on a disk) if no number is mentioned,
			.file with UNIT TYPE = 'R' in IBM DOS (COBOL variant 1)
			.Block 0 for UNISYS A Series (COBOL Variant 8) and AS 400 (COBOL Variant O).
			.Block 0 for IBM VSE COBOL II and file with UNIT TYPE = 'N'.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
13	1		BLOCK SIZE UNIT TYPE
		'R'	Records (default value).
		'C'	Characters.
		'N'	The BLOCK CONTAINS clause is not generated.
14	1	NUMER.	NUMBER OF CONTROL BREAKS
			(BATCH SYSTEMS DEVELOPMENT Function) All spaces are replaced with zeroes.
			For sequentially accessed, sorted files: Enter the number of Elements (elementary or group) on which there is to be control break processing for the Data Structure.
		'0'	Default.
		'1 to 9'	1 to 9 levels, according to the number of Elements to be used for control break processing. These Elements are identified as the SORT KEYs for this Data Structure.
			When there is control break processing on one or more Data Structures, two indicators keep track of the status of the records being processed:
			Note: The term 'nth key Data Element' includes all key Data Elements up to and including the nth level.
			.dd-IBn = '1': the nth key Data Element of the current record of Data Structure dd contains a new value,
			.dd-FBn = '1': the nth key Data Element of the current record of Data Structure dd contains the last occurrence of the present value.
			When these files are synchronized with others, (see FILE MATCHING LEVEL NUMBER) the control breaks are kept synchronized via two additional switches:
			.ITBn = '1': a new value in the nth key Data Element has been detected. This signals beginning processing on all synchronized D.S's.
			.FTBn = '1': the present value of the nth key Data is occurring for the last time. This signals end processing for the records in this iteration for all synchronized D.S's.
			For output files (USAGE OF DATA STRUCTURE value 'D'):
			A non-zero value will create a duplicate file layout to be generated in the WORKING-STORAGE area identifiable by a prefix of '1-'.
			Note however a preferable procedure to accomplish this is via the Work Areas (-W) Screen.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
15	1	NUMER.	FILE MATCHING LEVEL NUMBER
			BLANKS REPLACED BY ZEROES.
			For sequentially accessed files:
			Used to establish the synchronization of two or more files.
		'0'	Default.
		'1 to 9'	Enter the number of Elements (Elementary or Group) on which file matching is to be synchronized for this Data Structure. This number identifies the number of the key fields (identified in the SORT KEY/ field) that are involved in the synchronization.
			For an automatic file matching, the following conditions must be met:
			. The Data Structure control break level must be equal to the file matching level - 1, except for a transaction Data Structure, whose control break level must be equal or superior to the file matching level.
			. The Data Element(s) which constitute(s) the sort keys of a Data Structure must be sorted in ascending order.
			. The Data Element(s) which constitute(s) the sort keys of a Data Structure must have the same length for the same level.
			. These Data Elements must have a display format (if they are numeric, they must be whole numbers and unsigned).
			Switches generated to control the file matching are:
			.dd-CFn: which indicates whether a file should be processed or bypassed in this iteration, ('1' = process, '0' = bypass).
			.dd-OCn: which indicates the status of processing on a record of a principal file (USAGE OF DATA STRUCTURE = 'P').
			For sequentially accessed files:
			'1' = WRITE to the principal file
			'0' = do not WRITE.
			For direct access files:
			'1' = CREATE or REWRITE
			'0' = DELETE
16	1		USAGE OF DATA STRUCTURE

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			This code defines the role of the Data Structure in the Program and determines the generated functions.
		'C'	Consult
			Any input file (Data Structure).
		'D'	Direct
			Any output file (default).
		'T'	Table
			A file to be fully stored in memory. The table is generated according to the number of repetitions indicated on each Segment Definition. (See OCCURRENCES OF SEGMENT IN TABLE).
			The maximum number of selected Segments per D.S. = 50.
		'X'	Table
			A file to be partially stored in memory. Only Data Elements other than FILLER are loaded.
			Elementary Data Elements other than FILLER are limited to 10 (in addition to the RECORD TYPE ELEMENT) for the '00' Segment and to 29 for each specific non-00 Segment.
		'S'	Selected
			Output file extracted from another file.
			It differs from USAGE value 'D' since the generated description in the output area is not detailed. For Data Elements with an 'OCCURS DEPENDING ON' clause, the USAGE OF DATA STRUCTURE must be 'D'.
			The following values are specific to the Batch Systems Development function:
		'P'	Principal
			Input file, likely to be updated (by a transaction file - usage value 'M' or 'N').
		'R'	Result
			Updated principal file in sequential access mode. (When the Data Structure contains an 'OCCURS DEPENDING ON' clause, the output/result D.S must be declared as 'D').
		'M'	Transactions to be validated:
			Input file to be validated which may update other file(s). The generated functions range from 30 to 76.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			Note: Only one 'M' or 'N' Data Structure is allowed per Program.
		'N'	Transactions not to be validated:
			Input file which can update other files.
			The generated functions are: 30, 33, 39, 70 to 76.
			Note: Only one 'M' or 'N' D.S. is allowed per Program.
		'E'	Transaction file with errors detected:
			Output transaction file containing a field identifying records with errors. The system will generate the field(s) to track the erroneous Elements, erroneous Segments and user defined errors using the reserved Data Elements ENPR, GRPR and ERUT. (The option is selected in the RESERVED ERROR CODES IN TRANS. FILE field). Selected or not, the descriptions of these Elements are generated (using the Data Elements DE-ERR and ER-PRR).
			These descriptions precede the descriptions of the Elements.
		'I'	Direct printing (or by SYSOUT in IBM MVS)
			At the generation level, the lines with STRUCTURE NUMBER value of '00' will be ignored.
		'J'	Indirect printing to be processed by a spool Program.
			Fields required for identifying the lines, line skips, etc. are defined in Report STRUCTURE NUMBER value 00.
17	2		RESULTING FILE DATA STRUCTURE CODE
			With USAGE OF DATA STRUCTURE value 'P', indicate the DATA STRUCTURE CODE IN THE PROGRAM of the resulting output D.S.
			For an output type USAGE OF DATA STRUCTURE (value 'R' or 'D'), indicate the DATA STRUCTURE CODE IN THE PROGRAM of the input principal D.S.
18	2		SOURCE OR ERROR DATA STRUCTURE CODE
			For a transaction file (USAGE OF DATA STRUCTURE = 'M' or 'N'), enter the DATA STRUCTURE CODE IN THE PROGRAM of the transaction file containing the error fields (USAGE OF DATA STRUCTURE = 'E') if one has been called.
			For a transaction file with the error field (USAGE OF DATA STRUCTURE'), enter the DATA STRUCTURE CODE IN THE PROGRAM of the corresponding transaction file (USAGE OF DATA STRUCTURE = 'M' or 'N').

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			For a selected file (USAGE OF DATA STRUCTURE = 'S'), enter the DATA STRUCTURE CODE IN THE PROGRAM of the input source with the corresponding Data Structure code of the selected file on the line where the source file is being called.
19	1		TRANSACTION CONTROL BREAK LEVEL
			ALL SPACES REPLACED BY ZEROS.
			Default option: NUMBER OF CONTROL BREAKS
			In a transaction file, enter the position within the SORT KEY/ of the ACTION CODE ELEMENT. For example, if the SORT KEY/ value is ABCDE and the ACTION CODE ELEMENT is 'D', enter '4' here.
			This element is the minor-most key of the sort key and the one used to differentiate one type of transaction from another of the same principal file. Duplicates are detected if any key elements below this one are found to match.
20	4		PHYSICAL UNIT TYPE
			NOTE: The term 'COBOL Variant' = the value in the TYPE OF COBOL TO GENERATE field) generates the following in the SELECT clause of some COBOL variants:
			IBM DOS (COBOL Variant 1):
			Enter the model type (examples: 2314, 3330, 2400).
			MICROFOCUS, COBOL II, IBM VISUAL SET (COBOL Variant 3)
		EXT	Generation of the EXTERNAL clause at the file FD level
		LS	Generation of the LINE SEQUENTIAL clause
		EXLS	Generation of the LINE SEQUENTIAL clause and of the EXTERNAL clause at the file FD level
			ACU COBOL (COBOL Variant Q) :
		LS	Generation of the LINE SEQUENTIAL clause
			Gcos7 (COBOL Variant 4):
		'SSF'	Option WITH SSF in the SELECT clause
		'OUT'	Option -SYSOUT suffix after the filename in the SELECT clause (WITH SSF is generated).
			Gcos8 ASCII (COBOL Variant 5):
		'PT'	Printer.
		'CR'	Card reader.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'SSF'	ORGANIZATION IS GFRC SEQUENTIAL SSF CODE SET IS IS GBCD.
		'IBM'	ORGANIZATION IS IBM-OS SEQUENTIAL.
		'xxx'	WITH xxx.
		'...V'	A 'V' in the 4th position generates the clause 'VALUE OF FILE-ID is 3-FF00-IDENT' (FF is the program Data Structure code being called).
			The field 3-FF00-IDENT must be defined in -W by the user.
			BURROUGHS large system (COBOL Variant 8) UNISYS A Series:
		DK or	
		'blank'	Disk.
		'DKS'	Sort Disk (with T opening).
		'DKM'	Merge Disk (with T opening).
		'RD'	Reader.
		'PT'	Printer.
		'PO'	File.
		'TP'	Tape.
			For the 2-character codes, a third character can specify a particular final disposition:
		'..P'	Purge.
		'..R'	Release.
		'..L'	Lock.
		'..S'	Save.
		'...V'	A 'V' in the 4th position generates the clause 'VALUE OF D.S. NAME IS 3-FF00-IDENT'.
			UNISYS 2200 (COBOL Variant U):
		'CR'	Card reader.
		'CP'	Card punch.
		'UN'	Uniservo.
		'TP'	Tape.
		'PN'	Printer with external name. If the COMPLEMENTARY PHYSICAL UNIT TAPE field contains input, the RECORDING clause is also generated.
		'PT'	Printer without external name.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'PF'	Printer with external name and:
			VALUE OF PRINTER-FORMS 3-FF00-FORMS
			LINAGE IS 3-FF00-LINES
			TOP IS 3-FF00-TOP
			BOTTOM IS 3-FF00-BOTTOM
			These 4 data-names are to be declared in Work Areas (-W) lines with their appropriate values.
			AS 400 (COBOL Variant O):
		DB	Database.
		RD	Reader.
		CP	Card Punch.
		PT	Printer.
		TP	Tape.
		DK or	
		'blank'	Disk.
21	1		COMPLEMENTARY PHYSICAL UNIT TYPE
			NOTE: The term 'COBOL Variant' = the value in the TYPE OF COBOL TO GENERATE field.
			IBM DOS (COBOL Variant 1):
		'R'	Reader.
		'P'	Punch.
			BULL Gcos8 (COBOL Variant 5):
		'S'	EBCDIC Set code.
		'C'	ASCII Set code.
			UNISYS 2200 (variant U):
		'S'	Recording followed by lock mode.
			BULL Gcos7 (COBOL Variant 4) and Gcos8 (COBOL Variant 6)
		'O'	If the value 'O' is entered in this field, the OPTIONAL option is not generated.
			Otherwise, the OPTIONAL option is generated by default.
			DEC VAX VMS (COBOL Variant I)
		'A'	File opening with option ALLOWING ALL and sequential reading with option REGARDLESS.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			IBM MVS :
		'F'	OPTIONAL parameter generated in the SELECT clause of a VSAM file.
22	9		SELECTION
			This field has three mutually exclusive uses:
			1. Composition of the sort key
			This is the group of Data Elements making up the sort key for control break processing. They are identified by the value entered in the KEY INDICATOR FOR ACCESS OR SORT field on the Segment Call of Elements (-CE) screen.
			The order of sorting these key Data Elements may be entered here using the values assigned on the Call of Elements (-CE) screen in the desired order of major to minor - left to right. If no explicit entry is made here, Elements coded with value 1 to 9 will be taken as the default.
			The Data specifying the sort order must be entered on first line of the Data Structure call. (That is on the line where the CONTINUATION OF D.S. DESCRIPTION field remains blank.)
			Note: for transaction files, include the ACTION CODE and RECORD TYPE ELEMENTs as a part of the key. The order in which these Elements are sorted will determine the sequence in which the transactions update the principal file, and the policy for duplicate record detection.
			2. Selection of Segments in a Data Structure
			Rather than having all of the Segments belonging to a Data Structure described, the user may select the ones that are needed, thus avoiding unnecessary description lines and wasted work area space. This may be significant for tables (USAGE OF DATA STRUCTURE = 'T').
			This is done by entering an '*' in the first column of this field followed by a maximum of 4 SEGMENT CODES, in addition to the common part. The Segments may come from different D.S.'s, but in this case, it is better to call these Segments into another Segment.
			When the user wishes to re-create the file matching key and select records, he/she must indicate the file matching on the first Segment Call line, and the selected records on continuation lines.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			When Segments come from different D.S.'s Descriptions, the common part of the first D.S. called is considered to be the resulting file common part. The other D.S.'s must not have a common part.
			3. Report selection: To select a particular Report, the third character in the Report code must be entered in the field. To select all Reports with the same prefix, you must leave the field blank.
			Generally, continuation lines are created if more than four Segments or nine Reports are selected.
			It is possible to rename a SEGMENT CODE or LAST CHARACTER OF REPORT CODE: one line per Segment or Report to be renamed is created. Enter the LAST CHARACTER OF REPORT CODE as known in the Library, followed by the desired code for the Program separated an ″=″ sign.
			Follow the same procedure to rename the SEGMENT CODE, but precede the old Segment code with an asterisk.
			EXAMPLE:
			1=2 Rename report code 1 report code 2
			*01=02 Rename segment code 01 segment code 02.
23	1		NON-PRINTING DATA STRUCTURE FORMAT
			This option is reserved for Data Structures with a USAGE OF DATA STRUCTURE other than 'I' or 'J'.
		'E'	Input format. (Default option with USAGE OF D.S. = 'M', 'N' or 'E').
		'I'	Internal format (Default with USAGE OF D.S. NOT= 'M', 'N' or 'E').
		'S'	Output format.
			Note: the Elements making up the Segments must not exceed 999 characters.
24	1		RESERVED ERROR CODES IN TRANS. FILE
			Indicates if reserved Data Elements (ENPR, GRPR, ERUT) contained in the Data Structure Description are to be described.
		blank	The Description is not generated.
		'V'	The Descriptions are generated for all of these Data Elements.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'W'	Same as 'V', but the Data Element ENPR represents the error vector. (Reserved for USAGE OF D.S. = 'M', 'N' or 'E'.)
		'E'	Only the 'ENPR' and 'GRPR' Descriptions are generated.
		'U'	Only the 'ERUT' Description is generated.
			In a transaction file (USAGE OF D.S.= 'M', 'N' or E'), these Data Elements must appear at the beginning of the Description and are used to carry results of validations to the update.
			.ENPR: n+1 positions for values 'V' or 'E' and m+1 positions for value 'W', where:
			n = number of elementary Data Elements in the Data Structure description.
			m = greatest number of elementary Elements in the file : that is, those in the common part Segment plus the largest non-00 Segment. The extra position is the identification error.
			It initializes the DE-ERR vector.
			.GRPR: 1 position per record + 1 for group error.
			It initializes the SE-ERR vector.
			When these Elements are used in a file other than a transaction-type file, the placement and format is at the option of the user.
		1..9,0	With the Pactables function, it specifies the number of sub-schemas desired. Refer to the 'Pactables' Reference manual.
			With an SQL utilization file, it specifies the number of the sub-schemas desired (selection of a Column in a Table).
25	1		RECORD TYPE / USE WITHIN D.S.
			This option is used to select the type of record description to be used in the COBOL Program to allow different uses of the Segment Description stored in the Library.
		blank	Redefined records (Default option). No VALUE clause is generated.
		'1'	A record set without initial values or repetitions of records. These records are presented with the Segment common part followed by the different specific parts.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			If the Data Structure Description appears in the COBOL FILE SECTION, the LEVEL NUMBER (COBOL) OF THE RECORD must be 2. With this value, the specific Segments are described without redefines, at the COBOL 02 level. Several Segment Descriptions are grouped together under the same I/O area.
		'2'	A record set with the specific initial values of the Data Element of the Segment as defined on the Call of Elements or Data Element Description screen. These values may also default to blank or zero depending on the format.
			This type of description cannot be used for a Data Structure having a number of repetitions in the common part Definition. (Use ORGANIZATION = 'W' or 'L').
			Initial values are also generated for the occursed fields if the 'Generated language' of the Library is set to 'D' (COBOL II, 85, or 370).
		'3'	A record set which incorporates the number of repetitions specified in OCCURRENCES OF SEGMENT IN TABLE on the Segment Definition Screen. No VALUE clause will be generated.
			If the description of the Data Structure appears in the COBOL FILE SECTION, the LEVEL NUMBER (COBOL) OF THE RECORD must be '2'.
		'4'	A record set which incorporates the number of repetitions specified in the OCCURRENCES OF SEGMENT IN TABLE on the Segment Definition Screen.
			The associated LEVEL NUMBER (COBOL) OF THE RECORD must be '3'.
			Comment specific to the OLSD function: For a description type of '4' and a COBOL 03 level, the index is not generated.
			A COBOL 02 level is used to access the table made up of repetitions of the same record (ddssT).
			A COBOL 01 level is used to group the whole Data Structure together - common or specific parts, whether repeated or not.
			A group level field that incorporates all occurrences is generated.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			For Data Structures that do not have a value specified for the OCCURRENCES OF SEGMENT IN TABLE, use ORGANIZATION = 'W' with USAGE OF Data Structure = 'T'.
26	1		LEVEL NUMBER (COBOL) OF THE RECORD
			This option, used in conjunction with the RECORD TYPE /USE WITHIN D.S. field, defines the COBOL level number for the descriptions of Data Structures, Segments and Elements.
			In the following descriptions, the term 'D.S. Area' is meant as the area 'dd00' (possibly 1-dd00, 2-dd00).
		'1'	COBOL 01 level for D.S. Area and Segments. (Default value).
			If the Data Structure Description appears in the COBOL FILE SECTION, the Segments must be redefined.
			If a Data Structure has no common part with a non-redefined Description, the D.S. Area will only appear when the RECORD TYPE / USE WITHIN D.S. = blank.
		'2'	COBOL 01 level for D.S. Area and Segments at 02 level.
			If the RECORD TYPE / USE WITHIN D.S. = blank, both the DS Area and the Segments will be described at the 02 level. (To define the 01 level, use ORGANIZATION = 'L' and Work Areas (-W) lines.)
		'3'	Reserved for D.S. with an ORGANIZATION = 'W' or 'L'.
			COBOL 02 level for the D.S. Area and Segments at 03 level when associated with RECORD TYPE / USE WITHIN D.S. = 1, 2, or 3.
			01 level for the D.S. Area and Segments at 03 level when associated with RECORD TYPE / USE WITHIN D.S.= 4.
			03 level for both the D.S. Area and the Segments when associated with RECORD TYPE / USE WITHIN D.S. = blank.
		'4'	Reserved for Data Structures with an 'L' ORGANIZATION and USAGE OF DATA STRUCTURE = 'D'. The 01 level is to be defined via the Work Areas Screen (-W).
			COBOL 02 level for group Data Elements or elementary Elements that are not part of a group.
			Elementary Elements that are part of a group appear. The D.S. Area and Segment levels disappear.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'5'	Reserved for Data Structures in ORGANIZATION 'L' or 'W' and with a USAGE OF DATA STRUCTURE = 'D'.
			COBOL 01 level for group Data Elements or elementary Elements that are not part of a group.
			Elementary Elements that are part of a group appear. The D.S. Area and Segment levels disappear.
		'6'	Reserved for Data Structures with an 'L' ORGANIZATION and USAGE OF DATA STRUCTURE = 'D'. The 01 level is to be defined via the Work Areas Screen (-W).
			COBOL 02 level for group Data Elements or elementary Elements that are not part of a group.
			Elementary Elements that are part of a group disappear as well as D.S. Area and Segment levels.
			For standard OLSD Screens only.
		'7'	Reserved for Data Structures in ORGANIZATION 'L' or 'W' and with a USAGE OF DATA STRUCTURE = 'D'.
			COBOL 01 level for group Data Elements or elementary Elements that are not part of a group.
			Elementary Elements that are part of a group disappear as well as D.S. Area and Segment levels.
			For standard OLSD Screens only.
27	2		CODE FOR COBOL PLACEMENT
			PSEUDO-NUMERIC FIELD, blanks replaced by zeros.
			This field concerns only the principal Description of a D.S. (ddss) and not the Descriptions preceded by a prefix (1-ddss or 2-ddss).
			This field is used to obtain a Description of a D.S. in a particular area (COMMUNICATION area with DBMS's or the LINKAGE SECTION which the user must define by a Work Areas (-W) line), or at the beginning of the WORKING-STORAGE SECTION.
			This field is reserved for D.S.'s with an 'L', 'D' or 'W' ORGANIZATION, in order to place the I/O area in WORKING STORAGE.
			To have a Data Structure described in WORKING-STORAGE it is preferable to use the Work Areas (-W) lines.
		'00'	The Description of the D.S. is inserted after all the Work Areas (-W) lines. (Default value).

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		alphabet.	The Description of the D.S. is inserted after all the Work Areas (-W) lines whose 5-digit line number begins with this value.
			The Description and Work Areas (-W) lines are found at the beginning of the generated Program WORKING-STORAGE SECTION. These lines appear both before Data Structures with ORGANIZATION = 'W' and before those whose DATA STRUCTURE CODE IN THE PROGRAM is greater than this alphabetic code.
			(Do not use this field with a Data Structure whose ORGANIZATION = 'W'.)
		alphanum.	The Description of the D.S. is inserted after all the Work Areas (-W) lines whose 5-digit line number begins with this value. The Work Areas (-W) lines and the Description can be found in the generated Program, at the end of the WORKING-STORAGE SECTION among the user areas.
			The location is indicated on the first line of the D.S. call (CONTINUATION OF DS DESCRIPTION field = blank), and is repeated (by default) on all of its continuation lines.
			However, it is possible to attribute different locations to each record description of D.S. in a Program. This is done by entering several call lines for this D.S., specifying a record selection and a location for each one.
			Therefore, the Data Structure must have an unpacked description, whether implicit or explicit.
			WARNING: with ORACLE, you must use numeric values so that the DECLARE SECTION will be correctly generated (with data fields and indicators included in it).
28	10		STATUS FIELD - FILE INDICATOR
			(Note: In this discussion, the term 'COBOL Variant' = the value in the TYPE OF COBOL TO GENERATE field)
			Enter the DATA STRUCTURE, SEGMENT and DATA ELEMENT CODEs in the following format:
			ddsseeeeee
			(Recommendation: ss = 00).
			This field is used in one of three ways:
			For VSAM files
			.The FILE STATUS IS clause is generated using 1-ddss-eeeeee (declared as a two byte field).
			For hardware other than Gcos8 BCD and non-VSAM files

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			.The NOMINAL, SYMBOLIC or ACTUAL KEY depending on the COBOL Variant.
			The user must define the corresponding work area: 1-ddss-eeeeee.
			The positioning of this key as well as the read of the D.S. must be programmed by using Procedural Code (-P).
			For Gcos8 BCD (COBOL Variant 6)
			.Identification of the Data Structure.
			.The corresponding 'VALUE OF' clause will be generated only if it's filled in.
			.The return-code area of the input-output operations
			.The corresponding 'FILE STATUS IS' clause will be generated only if it's filled in.
29	6		Indexed Data Structure Access Key
			Required for indexed Data Structures: Enter the DATA ELEMENT CODE of the access key Element.
30	6		CODE OF RECORD TYPE ELEMENT
			Enter the code of the Data Element whose values define different record types of a Data Structure.
			Note: Must be in the common part (00 Segment).
			This code can also be specified on the Segment Definition Screen for the 00 Segment in the CODE OF RECORD TYPE ELEMENT field, and is then used as a default value at generation level.

Generation Options (-GO)

You can enter the following options on the Program Generation Options (-GO).

Each of these options must be entered on a separate line, starting from column 1.

MODIFYING THE DATE TRANSFORMATION FUNCTION

Date transformation is managed by default in the F9520 function. However you can choose another function if you enter DATPRO=ffss, where ffss is the chosen function-subfunction.

BREAKING DOWN GENERATED DATES

You can request the breakdown of generated dates into elementary fields by entering BREAKDATE=YES (or inhibit it by entering BREAKDATE=NO if it has been set to YES at the Library level).

This breakdown will be effective:

- For Programs: on the elementary Data Elements of the Segments called in the Call of Data Structures (-CD) and in the Work Areas (-W), on 'F'-type lines.
- For Dialogues/Screens: on the elementary Data Elements of the Segments called in the Dialogue Complement (-O), in the Screen Calls of Segments (-CS) and in the Work Areas (-W), on 'F'-type lines.
- on Data Elements called in the Work Areas (-W), on 'I', 'E' or 'S'-type lines.

If you indicate the BREAKDATE=YES option, the Data Elements defined with a date format will be generated as elementary fields which correspond to the year, month and day and a separator (if it is included in the date format).

Example of a date defined with an M-type format (MM/DD/YYYY):

```
10   ffnn-date.
 11  ffnn-date-MMX.
  12 ffnn-date-MM    PICTURE 99.
 11  ffnn-date-S1   PICTURE X.
 11  ffnn-date-DDX.
  12 ffnn-date-DD    PICTURE 99.
 11  ffnn-date-S2   PICTURE X.
 11  ffnn-date-YYX.
  12 ffnn-date-YY    PICTURE 9(4).
```

Notes: A date will be broken down only if the generated COBOL level of the date field is lower than or equal to 47.

If a VALUE has been entered, it will be generated in the group field.

Any additional information (such as a VALUE) must be entered on the same line as the Data Element call on 'I', 'E' or 'S'-type lines on the Work Areas (-W). If a continuation line has been specified, the date will not be broken down.

The Data Elements called in SQL Segments cannot be broken down, except if these SQL Segments are DB2 Segments and if the DESCR=ALL option has been entered on the DB2 Block Generation Options (-GO). Since host variables cannot be group fields, the elementary fields will be generated under a redefined group field in the following way:

```
ffnn-date-BRK REDEFINES ffnn-date.
```

MODIFYING THE FORMAT OF GENERATED INDEXES

The format of generated indexes is entered as parameter INDIC=. It it is not present, the format then will depend on the type of code to generate.

Example :

```
LIBRARY   GENERAL DOC.            EXP    LIBRARY EXAMPLE
A LIN : T COMMENT                               LIB
  100 : O INDIC=COMPUTATIONAL-3
```

On-line access commands

```
LIST OF PROGRAMS
CHOICE             SCREEN                                UPD
------             ------                                ---
LCPaaaaaa          List of Programs by code               NO
                   (starting with Program 'aaaaaa').
LNPaaaaaa          List of Programs by name               NO
                   (starting with program 'aaaaaa').
LTPnPaaaaaa        List of Programs of type 'n'           NO
                   (starting with program 'aaaaaa').
LEPeeeeeeee        List of Programs by external name      NO
                   (starting with external name 'eeeeeeee').
DESCRIPTION OF PROGRAM 'aaaaaa'
CHOICE             SCREEN                                UPD
------             ------                                ---
Paaaaaa            Definition of Program 'aaaaaa'.       YES
PaaaaaaGCbbb       Comments for Program 'aaaaaa'         YES
                   (starting with line 'bbb').
PaaaaaaGObbb       Generation option of Program 'aaaaaa' YES
                   (starting with line 'bbb').
PaaaaaaXVbbbbbb    X-references of Program 'aaaaaa' to    NO
                   Documents (starting with Document
                   'bbbbbb').
PaaaaaaATbbbbbb    Text assigned to Program 'aaaaaa'      NO
                   (starting with text 'bbbbbb').
PaaaaaaX           X-references of Program 'aaaaaa'.      NO
PaaaaaaXPbbbbbb    X-references of Program 'aaaaaa' to    NO
                   programs (starting with Program 'bbbbbb')
PaaaaaaXObbbbbb    X-references of Program 'aaaaaa' to    NO
                   screens (starting with Screen 'bbbbbb').
PaaaaaaXQrrrrrr    List of occurrences linked to Program  NO
                   'aaaaaa' through User Relationship
                   'rrrrrr'.
PaaaaaaCR          Occurrences linked to Program         YES
                   'aaaaaa' through User Relationship
```

PaaaaaaCDbb	Call of Data Structures of Program 'aaaaaa' (starting with Data Structure 'bb').	YES
PaaaaaaCPbbbbbb	Call of Parameterized Macro-Structure of Program 'aaaaaa' (starting with P.M.S. 'bbbbbb').	YES
PaaaaaaBbbccddd	Beginning Insertions Modifications of Program 'aaaaaa' (starting with section 'bb', paragraph 'cc', line 'ddd').	YES
PaaaaaaWbbccc	Description of Work Areas of Program 'aaaaaa' (starting with Work Area 'bb' line 'ccc').	YES
PaaaaaaPfusfnnn	Description of Procedural Code of Program 'aaaaaa' (starting with function 'fu', sub-function 'sf', line number 'nnn').	YES
PaaaaaaPGfusfnnn	View of Procedures Generated of Program 'aaaaaa' (starting with function 'fu', sub-function 'sf', line number 'nnn'), with display of generated procedure titles.	YES
Paaaaaa9bbbbbb	Description of Pure COBOL Source Code of Program 'aaaaaa' (starting with -9 line 'bbbbbb').	YES
PaaaaaaTCfusf	View of Titles and Conditions of automatic and specific procedures of Program 'aaaaaa' (starting with function 'fu', sub-function 'sf').	YES
PaaaaaaTCfusf<nn or Paaaaaa<nnTCfusf	View of Titles and Conditions of automatic and specific procedures of Program 'aaaaaa' up to level 'nn' (starting with function 'fu', sub-function 'sf').	YES
PaaaaaaTOfusf	View of Titles Only of automatic and specific procedures of Program 'aaaaaa' (starting with function 'fu', sub-function 'sf').	NO
PaaaaaaTOfusf<nn or Paaaaaa<nnTOfusf	View of Titles Only of automatic and specific procedures of Program 'aaaaaa' up to level 'nn' (starting with function 'fu', sub-function 'sf').	NO

NOTE: After the first choice of type 'Paaaaaa', 'Paaaaaa' can be replaced with '-'.

All notations between parentheses are optional.

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
LIST OF PROGRAMS BY CODE

PROGRA MEMBER NAME OF PROGRAM OR MODULE      V N FR BA PROGR.ID TNT TYPE LIBR
AAAB   ----   ADABAS MACRO STRUCTURES ----   N                  PB  M    *CEN
AAAB10 AAAB10 ADABAS general access          N    L  L AAAB10   PB  M    *CEN
AAAB20 AAAB20 ADABAS on line structure       N    L  L AAAB20   PB  M    *CEN
AAAB30 AAAB30 ADABAS STANDARD FILE DESCRIPT. N    L  L AAAB30   PB  M    *CEN
AAAD   ----   IMS-DL1  MACRO STRUCTURES ---- N                  PB  M    *CEN
AAADEM AAADEM IMS error processing in monit. N    L  L AAADEM   PB  M    *CEN
AAADL2 AAADL2 Display list on 2 levels       N    L  L AAADL2   PB  M    *CEN
AAADL3 AAADL3 Display list on 3 levels       N    L  L AAADL3   PB  M    *CEN
AAADMS AAADMS MONITOR SWITCHING              N    L  L AAADMS   PB  P    *CEN
AAADSA AAADSA Definition of standard SSA     N    L  L AAADSA   PB  M    *CEN
AAADSO AAADSO SORT OF A DATA BASE AS INPUT   N    L  L AAADSO   PB  M    *CEN
AAADSW AAADSW On line program switch         N    L  L AAADSW   PB  M    *CEN
AAADUP AAADUP Update segment without key     N    L  L AAADUP   PB  M    *CEN
AAAD2S AAAD2S Display 2 segments in list     N    L  L AAAD2S   PB  M    *CEN
AAAD30 AAAD30 DL1 batch program structure    N    L  L AAAD30   PB  M    *CEN
AAAD31 AAAD31 DL1 BMP program structure      N    L  L AAAD31   PB  M    *CEN
AAAD40 AAAD40 IMS standard call              N    L  L AAAD40   PB  M    *CEN
AAAD50 AAAD50 IMS STANDARD I-O CALL          N    L  L AAAD50   PB  M    *CEN

O: C1 CH: LCP
```

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
 PROGRAM   GENERAL DOCUMENTATION AAPR20 Display the file counters

 A LIN : T DESCRIPTION                                                  LIB
 . 010 :       THIS MACRO STRUCTURE IS USED TO DISPLAY THE NUMBER OF     *CEN
 . 020 :   RECORDS READ OR WRITTEN FOR A FLAT FILE.                      *CEN
 . 030 :                                                                 *CEN
 . 040 :   PARAMETERS :  $1 -> SEQUENCE NUMBER                          *CEN
 . 050 :                 $2 -> FILE CODE (4 CHAR.)                      *CEN
       :
       :
       :
       :
       :
       :
       :
       :
       :
       :
       :
       :

 O: C1 CH: Paapr20GC
```

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
PROGRAMS CROSS-REFERENCES                AAPR20 DISPLAY PGM BEGIN. AND  END

A T PG/SC  LN C : COMMENTS OR PARAMETER VALUES                      D E
  C        10   :  NO PARAMETERS TO DEFINE.
  P JIPED1      :
  P JIPED2      :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :

O: C1 CH: Paapr20XP
```

```
                  PURCHASING MANAGEMENT SYSTEM        SG000008.LILI.CIV.1583
PROGRAM CROSS-REFERENCES                AVJIA1 Validation and Update

A T PG/SC  LN C : COMMENTS OR PARAMETER VALUES                    D E
  O JIE020      : 020/A/
  O JIE050      : 050/A/
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :
                :

O: C1 CH: Pavjia1XO
```

Generation and/or printing

Programs can be generated and printed by entering certain commands, either on-line, on the Generation and Print Commands (GP) screen (used for documentation and generation requests), or in batch mode (see the 'Developer's Procedures' manual).

These commands are listed below:

- LCP

List of all Programs by code.

C1: without keywords,

C2: with keywords.

- LNP

List of all Programs by name.

- LEP

List of all Programs by external name.

- LKP

List of Programs by keywords. The user may limit the keywords to explicit or implicit only. The keywords are specified on a continuation line (see the The 'Character Mode User Interface' guide).

- LTP

List of all Programs by type.

- DCP

Description information for the Program whose code is entered in the ENTITY CODE field; if no code has been entered, the Description information for all Programs will be provided.

C1: without assigned text,

C2: with the assigned text.

- DSP

Description information for the reversed Program whose code is entered in the ENTITY CODE field.

- GCP

Generation and description of a Program whose code must be indicated.

- GSP

Generation and description of the reversed Program (with SC lines).

- FLP

Specify the flow of the programs. The user may specificy the environment (PEI), control card options, and parameters (as needed).

C1 option only.

- FSP

Specify the flow of the reversed Programs.

Chapter 3. Segments

Definition

A Segment is defined by its code and name.

The Segment code is made of the Data Structure code and a number.

Depending on future needs, it is also possible to specify:

- the number of occurrences of the Segment (used in the activity calculation of the PACMODEL function),
- the maximum number of items of the table, if the Segment describes a table item.

STANDARD FILES

A standard file may have several types of records.

Nevertheless, the sort criteria and keys must be on all the records. This 'common part' is described once in the Segment number '00'.

The specific part of each record is described in a Segment number 'nn'.

In generated programs, a record description will be made of the concatenation of the '00' and the appropriate 'nn' segment descriptions.

A data element used to identify the specific record type has to be defined on the common part : the CODE OF RECORD TYPE.

This data element code is specified on the definition line of segment number '00'; the appropriate value is coded on the definition line of the specific part segment.

For a file that has only one type of record, a unique '00' segment is described.

TRANSACTION FILE (BATCH SYSTEMS DEVELOPMENT FUNCTION)

A transaction file is made of records that update a 'permanent' file.

A data element belonging to the common part of the file is used to identify the type of update being done (Creation, Modification, Deletion, or other cases). It is called the ACTION CODE.

This Data Element code and values are indicated on the Definition line of the '00' Segment, respectively in the 'CODE OF ACTION CODE' and 'VALUES OF TRANSACTION CODE' fields.

When each specific part Segment is defined, the rules concerning its presence or absence with each type of update are specified in the corresponding fields.

PREREQUISITE

The data structure must have been previously defined.

ASSOCIATED LINES

- Comments (-GC).

  These lines are used for documentation purposes.
- 'Generation Elements' (-GG).

  These lines are used to customize SQL accesses.
- 'Error Messages - Help' (-GE) where you specify error messages and on-line help on the Segment.
- 'Generation Options' (-GO) for the uppercase-lowercase management in customized SQL accesses.

```
                PURCHASING MANAGEMENT SYSTEM        SG000008.LILI.CIV.1583

                            1 2
SEGMENT DEFINITION.......: PR00

NAME.....................: COMPLETE PRODUCT RECORD  3

OCCUR. OF SEGMENT IN TABLE: 4
EST. NUMBER OF INSTANCES..: 5



CODE OF RECORD TYPE ELEM..: 6
CODE OF ACTION CODE ELEM..: 7
VALUES OF TRANSACTION CODE: CR:  8    MO:  9    DE: 10
                            M4: 11    M5: 12    M6: 13


EXPLICIT KEYWORDS..:  14


SESSION NUMBER.....: 0059       LIBRARY......: CIV    LOCK :


O: C1 CH: Spr00                     ACTION:
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			DATA STRUCTURE / SEGMENT CODE
1	2		DATA STRUCTURE CODE (REQUIRED)
			This code is made up of two alphanumeric characters. This is a logical code internal to the Database and therefore independent of the names used in Database Blocks and Programs.
2	2		Segment number (REQUIRED)
			The first character must be numeric and the second either numeric or alphabetic. However the second character can be alphabetic only if the first character is other than zero.
		00	For standard files:
			Used to indicate the common part of records in a file, located at the beginning of each record (Default).
			The control break sort keys, the record type and the keys of indexed files are contained in this Segment.
			A file does not necessarily have a common part.
			Records on files with only one type of record should be coded as a '00' Segment.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			With the Pactables function, this value is not allowed.
		01-99	Designates a specific Segment. The common part Data Elements are automatically concatenated with each specific part Segment. Although a data element may not be used twice in the same Segment, it may be used in both the common part and in one or more specific Segments (except data structures used as Tables).
3	36		SEGMENT NAME (REQUIRED IN CREAT)
			This name must be as explicit as possible because it is used in the automatic building of keywords, Words used here become implicit keywords (subject to limitations specified in the Character-Mode User Interface Guide, chapter 'Search for Instances', subchapter 'Searching by Keywords').
4	4	NUMER.	Occurrences of segment in table
			PURE NUMERIC FIELD
			BATCH SYSTEMS DEVELOPMENT:
			This is the amount of space reserved for a Segment in memory (USAGE OF DATA STRUCTURE 'T' or 'X', or RECORD TYPE = 3, or 4.
			For tables (USAGE OF DATA STRUCTURE 'T' or 'X'), the default value at generation time is 100.
			Pactables:
			This field is strictly for documentation purposes.
			PACBENCH C/S:
			The value entered in this field indicates the repetitive read or update capacity of the server which calls the Logical View. This capacity is expressed by a maximum number of repetitions. The Logical View can then be used as a repeated structure.
			NOTE: The use of a Logical View in a card layout does not exclude its use in a row layout. It is therefore strongly recommended to systematically fill in this field. Moreover, the entered value must be high enough to limit the exchanges between the client and the server.
5	9	NUMER.	Estimated number of instances
			PURE NUMERIC FIELD
			For the Batch Systems Development function, this field is used to specify the estimated number of occurrences for a segment in a database or in a standard file.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			For the METHODOLOGY function, this field is used for activity calculation on the record or set using the Segment (on-line only).
			For the DBD function, this field is used to specify the application number of an entity in a SOCRATE/CLIO Block.
6	10		Code/value of record elm. - table id
			For the Batch Systems Development function:
			CODE OF RECORD TYPE ELEM for the '00' segment:
			Enter the code of the data element used to identify the type of record (left-justified, six characters maximum).
			VALUE OF RECORD TYPE ELEM for the non-00 segments:
			Enter the value to differentiate the individual segments from one another.
			This information is required every time a variable1 file is used in a Segment.
			DL/1, SQL:
			Enter the external name of the segment or object 1 to 8 characters, between quotes).
			For Pactables table segments:
			Enter the END USER TABLE ID on 6 characters.
7	6		Code of action code element
			In the BATCH SYSTEMS DEVELOPMENT FUNCTION:
			Enter the DATA ELEMENT CODE for the element used to identify the transaction type. The System will generate validation logic appropriate for creation, modification, deletion and implicit action codes, as well as user-defined transaction types. Six values are associated with this code. Validation and updates are automatic for these six values:
			. transaction 1 creation, . transaction 2 modification, . transaction 3 deletion, . transaction 4 modification . transaction 5 modification, . transaction 6 modification.
			If there is no ACTION CODE ELEMENT, this field remains blank, and the transaction type is a modification. In this case, presence specifications for the segment are entered in the MOD-4 : ACTN CODE VALUE / SEG PRES. field, and for the elements, in the MOD-4 field on the Call of Elements (-CE) screen.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			The CODE OF ACTION CODE ELEMENT and the values must be entered on only one segment of the data structure, preferably on the common part '00'.
8	5		CREATE : ACTN CODE VALUE / SEG PRES.
			(Specific to the Batch Systems Development function).
			ACTION CODE VALUE:
			On the '00' segment, enter the value that stands for "create" for this file: Example: 'ADD'. Note: for alphabetic characters use quotes.
			SEGMENT PRESENCE:
			On the non-00 segments, enter the presence specifications for the individual segment.
		'O'	Obligatory: the segment must be present on a "create"
		'I'	Invalid: the segment must not be present on a "create"
		'F'	Optional (default).
9	5		MODIFY : ACTN CODE VALUE / SEG PRES.
			(Specific to the Batch Systems Development function).
			ACTION CODE VALUE:
			On the '00' segment, enter the value that stands for "modify" for this file: Example: 'CHG'. Note: for alphabetic characters use quotes.
			SEGMENT PRESENCE:
			On the non-00 segments, enter the presence specifications for the individual segment.
		'O'	Obligatory: the segment must be present on a "modify"
		'I'	Invalid: the segment must not be present on a "mofify"
		'F'	Optional (default)
10	5		DELETE : ACTN CODE VALUE / SEG PRES.
			(Specific to the Batch Systems Development function).
			ACTION CODE VALUE:
			On the '00' segment, enter the value that stands for "delete" for this file: Example: 'DEL'. Note: for alphabetic characters use quotes.
			SEGMENT PRESENCE:
			On the non-00 segments, enter the presence specifications for the individual segment.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'O'	Obligatory: the segment must be present on a "delete"
		'I'	Invalid: the segment must not be present on a "delete"
		'F'	Optional (default).
11	5		MOD-4 : ACTN CODE VALUE / SEG PRES.
			(Specific to the Batch Systems Development function).
			ACTION CODE VALUE:
			On the '00' segment, enter the value that stands for implicit action codes - (creates or modifications). Note: for alphabetic characters use quotes.
			SEGMENT PRESENCE:
			On the non-00 segments, enter the presence specifications for the individual segment.
		'O'	Obligatory: the segment must be present.
		'I'	Invalid: the segment must not be present.
		'F'	Optional (default).
12	5		MOD-5 : ACTN CODE VALUE / SEG PRES.
			(Specific to the Batch Systems Development function).
			ACTION CODE VALUE:
			On the '00' segment, enter the value that stands for this user-defined action. Note: for alphabetic characters use quotes.
			SEGMENT PRESENCE:
			On the non-00 segments, enter the presence specifications for the individual segment.
		'O'	Obligatory: the segment must be present.
		'I'	Invalid: the segment must not be present.
		'F'	Optional (default).
13	5		MOD-6 : ACTN CODE VALUE / SEG PRES.
			(Specific to the Batch Systems Development function).
			ACTION CODE VALUE:
			On the '00' segment, enter the value that stands for this user-defined action. Note: for alphabetic characters use quotes.
			SEGMENT PRESENCE:

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			On the non-00 segments, enter the presence specifications for the individual segment.
		'O'	Obligatory: the segment must be present.
		'I'	Invalid: the segment must not be present.
		'F'	Optional (default)
14	55		EXPLICIT KEYWORDS
			This field allows you to enter additional (explicit) keywords. By default, keywords are generated from the instance's name (implicit keywords).
			Keywords must be separated by at least one space. Keywords have a maximum length of 13 characters which must be alphanumeric. However, '=' and '*' are reserved for special usage and are therefore ignored in keywords.
			Keywords are not case-sensitive: uppercase and lower-case letters are equivalent.
			NOTE: Accented and special characters can be declared as equivalent to an internal value in order to optimize the search of instances by keywords (Administrator workbench, 'Window' menu, 'Parameters browser' choice, in 'Special Characters' tab).
			A maximum of ten explicit keywords can be assigned to one entity. For more details, refer to the 'Character Mode User Interface' guide, chapter 'Search for Instances', subchapter 'Searching by Keywords'.

Call of Elements screen (-CE)

A Segment is described by listing (calling) the Data Elements it contains. This is done by the -CE screen.

Additional information may be coded, according to the future use of the Segment (validation and update for transaction files, keys for database Segments, Pactables information..).

It is highly recommended to dedicate a Segment to only one type of future use.

OPERATION CODE

```
C1: default value (Update).
C2: display of the internal format of the Data Elements.
```

```
        display of Elements of a called "data aggregate"
        (see below).
        display of  names of Elements defined at the
        Segment level.
    C3: display of the input format of each Data Element
        called in the Segment.
```

GENERAL CHARACTERISTICS

A Segment is described by an ordered sequence of Data Elements. This sequence may include group Data Elements, or repetitions of elementary or group Data Elements.

Redefinitions are possible within a Segment.

For files and databases, access and control break sort keys are indicated. Initial values can be defined for work areas.

A segment is described by Data Elements defined in the Specifications Dictionary. As a result, the clear name of the Data Element, its formats and USAGE clauses are channeled down to the Segment level.

It is not possible to modify those characteristics at the Segment level.

It is possible to use Data Element codes which are not defined in the Specifications Dictionary, only when they do not have a real functional meaning (group Elements, fillers, error tables, etc.) In this case, a name and/or a format are required.

It is also possible to describe a Segment containing different aggregates of previously defined data, such as Segments or entities described with the PACMODEL function (Objects and Relationships).

It is not possible to modify the description of the called entity at the Segment level.

The same Data Element code, used in more than one place in a Segment, will provoke generation of identical data names.

PREREQUISITE

The Segment and the Data Elements (except some technical Data Elements which can be defined in the Segment description lines) must have been previously defined.

ASSOCIATED SCREENS

There are additional screens associated (via the LINE NUMBER) with each of the entities called onto the Segment Call of Elements (-CE) screen:

- the S....CEnnnGC screen for comments on the line,
- the S....CEnnnGG screen for additional information about the generation of Database Blocks,
- the S....CEnnnGE screen for additional documentation concerning error messages (Batch Systems Development function).

GROUP ELEMENTS

A Group Element is identified in the list by the number of elementary Data Elements it contains. These Elements are listed after the group element.

A group may include other groups. All elementary Elements are then counted to define the group.

If a dictionary Data Element is used as a group, its length is recalculated (sum of the lengths of the elementary data elements), regardless of its dictionary format.

REDEFINITION

Redefinition is possible within a Segment (generating the COBOL 'REDEFINES' clause). The following is entered in the UPDATE TARGET field:

```
.'R*'   in the UPDATE TARGET / FIRST PART,
. Blank in the rest of the UPDATE TARGET field.
```

The Data Element containing this option redefines the Data Element of the same COBOL level which precedes it in the Segment description. (See UPDATE TARGET / FIRST PART.)

If a Data Element which redefines another Data Element is contained in a group, it is considered to be an elementary Data Element. It must be taken into account in the calculation of the number of Data Elements contained in a group (except for DL1 database Segments).

NOTE: When Data Elements are redefined, the system does not take their respective lengths into account. This is the user's responsibility.

In the calculation of address length (Segment Level, Address and Length Description (-LAL)), the redefined Data Element length is used for the address calculation.

DATA AGGREGATES

Segments, Model Objects and Relationships (PACMODEL) are also called "data aggregates". They may be called into other segments.

The data aggregate code is indicated instead of the data element code in the list, and it is specified as a special group (see NO. OF ELEMENTARY ELEMENTS IN A GROUP). It may be occurred (See OCCURRENCES (COBOL 'Occurs' clause)).

The description (list of elements) will be included, but it cannot be modified at this level.

NOTE: On the -CE screen, the list of Data Elements of a called aggregate is only viewed in O: C2. When a Segment description is printed (DCS), only the SEGMENT CODE will appear. The expanded view of the Segment may be seen on the Segment Level, Address and Length (-LAL) screen.

LIMITATION

Called Segments may also contain segments. This 'nesting' may occur up to three times.

EXAMPLE:

	ELEM.	**GR**	**01 level: Segment BL00**
	ELEM.		01 level: Segment BL00
S BL00 CE	DELCO1		05 level: Delco1
	CL10	**	Segment CL10
S CL10 CE	DELCO2		10 level: Delco2
	DL20	**	Segment DL20
S DL20 CE	DELCO3		15 level: Delco3
	DELCO4		Delco4
			Segment AA30
S AA30 CE	DELCO5	**	20 level: Delco5

DATABASES SEGMENT DESCRIPTION

- Existing DL/1 segments

  DL/1 Segments defined prior to the installation of the System may have used Data Element codes that are eight characters in length. This does not conform to the System standards.

In that case, it is possible to define the Elements in the Dictionary to ensure future management in the System, and associate them with the old codes, to maintain compatibility with the existing applications.

- SQL external names

  SQL Data element codes are used also by the end-user, so they must be significant. In some cases, a Data Element must be given a code other than its System code.

In these cases, the two codes can be managed as follows:

On the Segment Call of Elements (-CE) screen, enter:

- The data element code in the DATA ELEMENT CODE field,
- 'A*' in the UPDATE TARGET / FIRST PART field,
- The former code (up to 8 characters) in the UPDATE TARGET / SECOND and LAST PARTs.

For DL/1, the 'old' code will be not only used in the Database Block description, but also in generated SSAs for on-line or batch programs.

TRANSACTION FILES

For each data element, there is a presence, class and value validation, with automatic reference to the values and intervals defined on the data element itself. Updates to be executed are also indicated.

NOTE:: Several principal data structures can be updated from one transaction data structure. The update processing will only be generated in a program if:

- The transaction data structure has a USAGE OF D.S. value of 'M' or 'N',
- The principal data structure has a USAGE OF D.S. value of 'P'.

For transaction data structures used to update principal data structures:

- Each transaction d.s. can update 10 principal d.s.'s.
- A "record pair" is one transaction d.s. and one principal data structure.
- Each record pair generates a sub-function.

EXAMPLE:

Using 'PD' and 'QD' as Principal data structures, and 'MD' and 'ND' as transaction data structures:

- If 'PD' is updated by 'MD' and 'QD' is updated by 'ND', two sub-functions will be generated.
- If 'ND' also updates 'PD', a third sub-function will be be generated.

There is a limit of 99 sub-functions per program and 200 for all programs, for each transaction Data Structure.

```
                  PURCHASING MANAGEMENT SYSTEM        SG000008.LILI.CIV.1583
                           1 2
SEGMENT CALL OF ELEMENTS PR00 COMPLETE PRODUCT RECORD

3 4     5      7         8 9   10 11  14 17  20 22        23        26
                                     12 15  18 21            24
                                      13 16  19               25
A LIN : ELEM.  INT.FORM. U OCC GR K CMD456 CONT VALUE/SFC UPD/TRGET DOC LIBR
  000 : PRDKEY                  1 U                                   0059
  010 : VENUMB                    B                                   0059
  020 : PR01                   **                                     0059
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      : NAME    : 6
*** END ***
O: C1 CH: -CE
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			DATA STRUCTURE / SEGMENT CODE
1	2		DATA STRUCTURE CODE (REQUIRED)
			This code is made up of two alphanumeric characters. This is a logical code internal to the Database and therefore independent of the names used in Database Blocks and Programs.
2	2		Segment number (REQUIRED)
			The first character must be numeric and the second either numeric or alphabetic. However the second character can be alphabetic only if the first character is other than zero.
		00	For standard files:
			Used to indicate the common part of records in a file, located at the beginning of each record (Default).
			The control break sort keys, the record type and the keys of indexed files are contained in this Segment.
			A file does not necessarily have a common part.
			Records on files with only one type of record should be coded as a '00' Segment.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			With the Pactables function, this value is not allowed.
		01-99	Designates a specific Segment. The common part Data Elements are automatically concatenated with each specific part Segment. Although a data element may not be used twice in the same Segment, it may be used in both the common part and in one or more specific Segments (except data structures used as Tables).
3	1		ACTION CODE (REQUIRED)
		'C'	Creation of the line
		'M'	Modification of the line
		'D' or 'A'	Deletion of the line
		'T'	Transfer of the line
		'B'	Beginning of multiple deletion
		'G'	Multiple transfer
		'?'	Request for HELP documentation
		'E' or '-'	Inhibit implicit update
		'X'	Implicit update without upper/lowercase processing
4	3		Line number
			Numeric.
			It is advisable to begin with line number '100' and then number in intervals of 20. This facilitates subsequent line insertions, as necessary.
5	6		DATA ELEMENT CODE
			ELEMENTARY DATA ELEMENT DEFINED IN THE DICTIONARY
			The Data Element automatically assumes the characteristics defined at the Specifications Dictionary level.
			If the Data Element is used as a group, its format depends on the characteristics of the elementary Elements that make up the group.
			If the group is used as a key (sort or access key), the composite format of the elementary Elements must be compatible with the format specified for the group.
			DATA ELEMENT NOT DEFINED IN THE DICTIONARY
			The name and/or format of undefined Data Elements must be indicated at the segment level.
			RESERVED DATA ELEMENT CODES

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		SUITE	Prohibited. This code is reserved for the System for program generation.
		FILLER	Data Element that is used for the alignment of fields.
			OPTIONS OF THE BATCH SYSTEMS DEVELOPMENT FUNCTION
			These codes (when used) precede other entries made in this field, in the sequence described below.
		ENPR	Used to store Element error verifications in a transaction file. The length is n + 1 where n = either the total number of elementary Elements in the file, or the number of elementary Elements in the '00' Segment added to the largest non-00 Segment. ("Largest" here means the most elementary Elements.) This depends upon the value entered in the RESERVED ERROR CODES IN TRANS FILE field on the Call of Data Structures (-CD) screen.
		GRPR	Used to store Segment error verifications. Its length is n + 1 where n = the number of records.
		ERUT	Used to store error verifications for users.
			Normally, these last three Data Elements are used in transaction files for error verification fields. When used in other types of files as "optional" Data Elements, they may be used as group fields whose generation may be invoked or suppressed according to the option selected in the RESERVED ERROR CODES IN TRANS. FILE field. (Note: this will affect the elementary Elements within the group as well.)
			CALLING DATA AGGREGATES
			A SEGMENT CODE or a Model Entity code (Relationship or Object in the METHODOLOGY function) can be entered in this field. The called data aggregate will be interpreted as if the individual Elements that make it up had been entered.
			The NO. OF ELEMENTARY ELEMENTS IN GROUP field is used to identify data aggregate calls.
			Enter the code at the location the elements are to be included in the Segment description.
			In O:C2, the level of 'nesting' is displayed in the Action Code (up to four levels).
			The number of authorized nesting levels varies according to the type of generator. Up to 4 nesting levels are authorized for data generation and PAF use.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			CONTINUATION LINES
			It is possible to create continuation lines. This may be necessary if there are many validations on a Data Element. In this case, leave the DATA ELEMENT CODE field blank, and use a LINE NUMBER value that sequentially follows that of the line where the Data Element code was entered.
6	18		NAME OF DATA ELEMENT
			It is not required for a Data Element which is not defined in the Data Dictionary.
			However, it is optional for a data aggregate or a FILLER.
			NOTE: For on-line entry of Data Elements that are not declared in the Dictionary, this field cannot be used to input more than one Data Element at a time. There is actually only one available field on this screen, whether for input or for display.
			To define an Element at the Segment level :
			- Enter the Element code (and possibly the format) on the -CE, line nnn,
			- On the 'name' line, repeat the line number (nnn), and indicate the name (18 characters maximum),
			- Use the C2 option to view the name and format.
			NOTE: If several undefined Data Elements have been defined in the Dictionary, only the name of the first Data Element will be displayed if the Choice 'CH:S.....CE' is used.
			To view the name of the Data Element CODEL, on line 130, for example, use the choice 'O: C2 CH: Sssss-CE130'. This will display the Data Elements called in the Segment 'ssss' from the line 130 on.
7	10		Data element internal format
			It is required only in the following cases :
			- For an elementary Data Element not defined in the Dictionary (COBOL format),
			- For a group Data Element that is or belongs to a key; its length must be the sum of the lengths of its elementary Data Elements,
			- For a FILLER-type field.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			It is the internal format; input and output formats will be the same (but with usage Display). It is defined as on a Data Element Definition screen.
8	1		INTERNAL USE
			For Data Elements not defined in the Specifications Dictionary when the INTERNAL FORMAT OF DATA ELEMENT field has been given a value, enter the appropriate USAGE (default : 'D' for DISPLAY).
			For valid values, see the USAGE field on the Data Element Definition Screen.
9	3		OCCURRENCES (COBOL "OCCURS" CLAUSE)
			PURE NUMERIC FIELD
			This field represents the 'OCCURS' clause at an elementary Data Element level, or at a group level (Maximum of 3 levels).
			It can be changed into an 'OCCURS DEPENDING ON' clause by entering '**' in the UPDATE TARGET field, followed by the counter's Segment and Data Element codes.
			The COBOL restrictions on the OCCURS clause apply.
10	2		No. of elementary elements in group
			PSEUDO NUMERIC FIELD
		'1 to 99'	For group Data Elements, enter the number of elementary Elements that belong to the group (A Segment call is considered as an elementary Data Element).
			Groups may contain up to 99 elementary Elements. Group Elements may contain embedded groups however the total number of elementary Elements cannot exceed 99. (The group Data Element codes are not counted). The maximum number of levels of 'nesting' is 9.
			This field is also used to identify the entity called in the DATA ELEMENT CODE field as Methodology entities or previously defined Segments.
		'*M' '**'	Call of an Object or a Relationship. Call of a Segment.
		'**'	SQL DBD function: Call of a Segment into a view.
11	1		Access or sort key
			This field identifies all data elements that might be used as control break sort keys, or as access keys to a file, a database or a Pactables table.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			Note: It is highly recommended to dedicate a Segment to only one type of use.
			Each data element that may belong to a sort key must be referenced by a unique alphabetic or numeric character. It is recommended to reference the indicators by a series (1, 2, 3 ...).
			The actual sort sequence will be chosen at the program level (on the Call of Data Structures (-CD) screen) by sequencing the characters in the appropriate order.
			Reminder:
			The format of key group data elements must have been entered in the Dictionary or at the segment level.
			PACTABLES:
		'U'	References the access key for a VisualAge Pacbase table. This value must be indicated on the group data element if it is a group key.
		'S'	Indicates that the data element belongs to at least one sub-system.
			DL1 DBD (See the DL/1 DATABASE DESCRIPTION Reference Manual)
		'U'	References a unique key for an DL/1 database.
		'M'	References a multiple key for an DL/1 database.
		1 to 9	Secondary index
			All other values designate a search field.
			DBD AS400 physical file (See the corresponding DBD Reference Manual)
		0 to 9	AS400 physical file key.
			Relational databases (See the corresponding DBD Reference Manual)
		'V'	Variable length column
		'Blank'	Fixed length column
		'W'	For DB2 SQL, SQL/DS and ORACLE, generation of a variable length column (VARCHAR).
		'L'	For DB2 SQL, SQL/DS and ORACLE, generation of a LONG VARCHAR.
			NOTE: Sort keys are not allowed on data elements redefining other data elements (see VALIDATION and UPDATE FIELDS, below).

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			DATA ELEMENT PRESENCE VALIDATION
12	1		CREATE : ELEMENT PRESENCE
		'O'	Required. Generation of a level 'E' (transaction refused) in standard error messages.
		'P'	Required. Generation of a level 'C' (data element refused) in standard error messages.
		'F'	Optional (default value).
		'I'	Not allowed.
			Relational Databases (Refer to the corresponding DBD Reference manual)
			It indicates the presence of a Column in a Table.
13	1		MODIFY : ELEMENT PRESENCE
		'O'	Required. Generation of a level 'E' (transaction refused) in standard error messages.
		'P'	Required. Generation of a level 'C' (data element refused) in standard error messages.
		'F'	Optional (default value).
		'I'	Not allowed.
14	1		DELETE : ELEMENT PRESENCE
		'O'	Required. Generation of a level 'E' (transaction refused) in standard error messages.
		'P'	Required. Generation of a level 'C' (data element refused) in standard error messages.
		'F'	Optional (default value).
		'I'	Not allowed.
15	1		MOD-4 : ELEMENT PRESENCE
		'O'	Required. Generation of a level 'E' (transaction refused) in standard error messages.
		'P'	Required. Generation of a level 'C' (data element refused) in standard error messages.
		'F'	Optional (default value).
		'I'	Not allowed
			Note: for segments without action code elements, enter element presence specifications.
16	1		MOD-5 : ELEMENT PRESENCE

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'O'	Required. Generation of a level 'E' (transaction refused) in standard error messages.
		'P'	Required. Generation of a level 'C' (data element refused) in standard error messages.
		'F'	Optional (default value).
		'I'	Not allowed
17	1		MOD-6 : ELEMENT PRESENCE
		'O'	Required. Generation of a level 'E' (transaction refused) in standard error messages.
		'P'	Required. Generation of a level 'C' (data element refused) in standard error messages.
		'F'	Optional (default value).
		'I'	Not allowed.
			DATA ELEMENT CONTENTS VALIDATION
18	1		CLASS (ALPHA / NUMERIC)
			Must appear on the first line for the data element. Validate the data element contents:
		'A'	Alpha or spaces are valid.
		'L'	Alpha Lowercase.
		'U'	Alpha Uppercase.
		'9'	Numeric values only.
		'B'	Numeric with leading spaces to be replaced by zeroes.
		'Z'	Numeric or spaces, the spaces are replaced by zeroes.
			'B' and 'Z' type validations are possible for any data element with a 'display' format (unpacked).
		BLANK	No class validations on the contents.
19	1		OPERATORS (AND / OR)
		' '	Must not appear on the first line for a data element.
		'E'	AND,
		'O'	OR.
20	1		Negation (NOT)
		'N'	Negation ('NOT' is generated).
		blank	No negation.
21	1		TYPE : VALIDATION, UPDATE, VALUES

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			This field has several different uses. More than one entry may be needed to assign all the validation conditions, update conditions and values that apply to a data element. In this case, enter the desired values on as many lines as needed, immediately following the original line used to call the element.
			1. Definition of the type of validation
			A. Contents Validation:
		'='	Equal to the value entered in the VALUES/SUB-FUNCTION CODE field.
		'>'	Greater than the value entered (as above).
		'<'	Less than the value entered (as above).
		'T'	Must be in the table indicated in the UPDATE TARGET field. Content validations entered following a 'T' type validation are not executed.
		'E'	Must have one of the values defined on the Description screen (-D) for this data element.
			B. Validation by PERFORM:
		'P'	Validation by PERFORM of a sub-function defined by the user. There may be only one validation by PERFORM per data element called in a segment.
			The following operations are executed:
			.transfer of the data element into the COBOL work area named in the UPDATE TARGET field. The naming of the work area on the appropriate line is the responsibility of the user.
			.PERFORM the sub-function entered (left-justified) in the VALUES / SUB-FUNCTION CODE field.
			This sub-function may check and modify (as needed) the data element.
			The result of the validation is indicated in the error indicator (DEL-ER), which is automatically generated.
			.This result is automatically transferred to the error table (DE-ERR) in the location that corresponds to the element being processed.
			.transfer of data from the work area to the initial data element, thereby incorporating any modifications made as a result of the performed function.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			This option is recommended for date validation, with possible inversion. In this case, the date must be defined as an elementary data element.
			In the description of a data element in a transaction, a ″Validation by PERFORM″ can be executed before or after a ″Content Validation″.
			If it appears before, it is executed only if the data is present with no error.
			If it appears after, it is executed only if there is a content error. The value for the corresponding location in the DE-ERR table then becomes the responsibility of the user.
			2. Definition of the type of update:
		blank	Direct update of the data element in the UPDATE TARGET field, contingent upon valid presence of the data element. This type of update can also be used with with ’contents Validations″ other than ’T’.
		’+’	Update by addition, contingent upon valid presence.
		’-’	Update by subtraction, contingent upon valid presence.
		’M’	Update by unconditional substitution (MOVE). Updating is done regardless of the validation result. This type of update can be used with group data elements.
			3. Definition of an initial value
		’V’	Initial value: generates a value using the literal entered in the VALUES / SUB-FUNCTION CODE field.
			It is the default value defined on the element description if the VALUES / SUB-FUNCTION CODE field is not used and if the element description has a D-type line (see the corresponding Chapter and Subchapter in the Specifications Dictionary manual).
			The RECORD TYPE / USE WITHIN D.S. field on the Call of Data Structures (-CD) screen must allow for the generation of VALUES clauses.
		’W’	Same as ’V’, but the literal can be continued into the UPDATE TARGET field. The two fields together would be considered as one.
			4. Special usages:
			DL/1 GROUP KEY DATA ELEMENTS

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'M'	To indicate a group key data element associated with the code entered (after 'A*') in the UPDATE TARGET. See "DL/1 SEGMENT DESCRIPTIONS" in Chapter "SEGMENTS" Subchapter "CALL OF ELEMENTS (-CE)".
			PACTABLES FUNCTION
		'S'	This indicates that the data element belongs to one or or more sub-schemas. The sub-schemas are entered in the VALUES / SUB-FUNCTION CODE field.
			If the data element belongs to a group element, you must enter a sub-schema number on the group element line.
			SQL RELATIONAL DBD FUNCTION
			The VALUE / SUBFUNCTION CODE field is used to indicate the sub-schema(s) a Column belongs to.
22	10		VALUES / SUB-FUNCTION CODE
			The input made in this field depends on the value of the TYPE : VALIDATION, UPDATE, VALUES field:
			Numeric or alphanumeric literal, name of manually positioned work area or sub-function code (left- justified), called by PERFORM in a data element validation.
			With '=', '>' or '<', enter the value to be compared.
			With 'P' enter the sub-function code to be performed. This code must be left-justified. (For more information, see Subchapter "DATA ELEM. CONTENTS VALIDATION (F45)".
			With '+', '-' or 'M' enter the value to be added, subtracted, or moved.
			With 'V' enter the literal to use as the initial value
			With 'W' enter the first part of the literal (which extends into the next field).
			With 'S' (PACTABLES and SQL DBD functions), enter the letter 'O' in the position in this field that corresponds to the sub-schemas to which the element belongs:
			Example:
			CONT VALUE/SFC DELCO S O OOO
			In this example, the data element 'DELCO' belongs to sub-schemas 1,3,4 and 5.
			UPDATE TARGET
			This field has several different usages:

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			1. To identify the target of the update;
			2. To identify the counter field defining a variable number of repetitions;
			3. To cause the redefinition of a data element within a segment;
			4. To identify the external name of a DL/1 search or key field;
			5. As a continuation of a literal.
23	2		UPDATE TARGET / FIRST PART
			DATA STRUCTURE CODE IN THE PROGRAM of a permanent file (USAGE OF D.S.= 'P' on the Call of Data Structures screen) to be updated, or of a table data structure with TYPE : VALIDATION, UPDATE, VALUES = 'T'.
			The data structure code for the target of an update.
			It can also be the WORKING data structure code for the data element communication area in a 'PERFORM' (TYPE : VALIDATION, UPDATE, VALUES = 'P').
		'**'	Associated with a repetitions number, in order to generate a variable number of OCCURs, using a counter contained in an element. This counter is referenced by the segment and data element codes which are indicated in the UPDATE TARGET / SECOND and LAST PARTs.
			Generation of an OCCURS DEPENDING ON clause. Transfers of the counter between input, WORKING and output areas are carried out automatically by VA Pac if this counter belongs to the common part.
		'R*'	To redefine a data element within a segment. The data element named in the DATA ELEMENT CODE field will refine the first data element that precedes it which is generated at the same COBOL level.
			Example:
			ELEM. GR GRPFLD 2 ELEM1 ELEM2 R* <--- or NEWVAL R* <---
			If 'R*' is entered opposite ELEM2, ELEM2 will redefine ELEM1. If 'R*' is entered opposite NEWVAL, NEWVAL will redefine GRPFLD.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'A*'	To identify the external name of a DL/1 key or search field. The external name (8 characters) is entered in the UPDATE TARGET / SECOND and LAST PARTs, and applies to the data element entered in the DATA ELEMENT CODE field on this line.
			SQL Relational Databases (Refer to the corresponding DBD Manual)
			.UPD/TRGET:
			The relational label of a Column can be identified in this field; the value 'A*' must be left flushed and followed by the external name of the Column.
			On the complementary screen displaying the origin of the columns of each view (-DBE), this field contains both the segment and the data element of the original Table.
24	2		UPDATE TARGET / SECOND PART
			SEGMENT CODE (default).
			When applicable:
			Enter the continuation of a literal.
			Enter the SEGMENT CODE.
			Enter the first two characters of the DL/1 external name.
25	6		UPDATE TARGET / LAST PART
			(Default Option: Data Element code)
			The default option also works for a modification.
26	1		DOCUMENTATION INDICATOR
			This field is used in on-line mode only. It is a read-only field.
		'*'	A Comment, a Generation Element or an Error Message has been assigned to the element called on this line.
			Access to line nnn: -CEnnn, or -Dxnnn for a Database Block (with x = C, H or R depending on the Block type)
			To access the Comment, Generation Element or Error Message assigned to the called element, enter the access to line nnn followed (without blank) by GC (for Comment), GG (for Generation Element) or GE (for Error Message).

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
DESCRIPTION OF SEGMENT : PR00 COMPLETE PRODUCT RECORD

A LIN LEVEL      ELEM.  OCC   INT. FOR.  U LGTH   ADD    INP. FOR. LGTH  ADD
  000 10         PRDKEY                           1
  010  11        VENUMB       X(5)       D    5     1    X(5)         5
  020 10         PR01          ----> SEGMENT PRODUCT INFORMATION
1 100  11        PRNUMB       X(10)      D   10     6    X(10)       10
1 110  11        PRDESC       X(30)      D   30    16    X(30)       30    1
1 120  11        PRPRIC       9(6)V99    3    5    46    9(6)V99      8    4
1 130  11        PRDTIM       999        3    2    51    999          3    5
1 140  11        PRMEAS       XX         D    2    53    XX           2    5

*** END ***
O: C1 CH: Spr00LAL
```

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
DESCRIPTION OF SEGMENT : PR00 COMPLETE PRODUCT RECORD

A ELEM.  NAME                          INP. FOR.  INT. FOR.  U OCC GR K LIBR
  PRDKEY PRODUCT KEY                                                1 U 0059
  VENUMB VENDOR NUMBER                 X(5)       X(5)       D        B 0059
  PR01                                                              **   0059
1 PRNUMB PRODUCT NUMBER                X(10)      X(10)      D        A 0059
1 PRDESC PRODUCT DESCRIPTION           X(30)      X(30)      D          0059
1 PRPRIC PRODUCT PRICE                 9(6)V99    9(6)V99    3          0059
1 PRDTIM ESTIMATED DELIVERY TIME       999        999        3          0059
1 PRMEAS UNIT OF MEASURE               XX         XX         D          0059

*** END ***
O: C1 CH: -DED
```

```
|                  PURCHASING MANAGEMENT SYSTEM        SG000008.LILI.CIV.1583 |
| DESCRIPTION OF SEGMENT : PR00 COMPLETE PRODUCT RECORD                       |
|                                                                             |
|                                    PR00                           TOTAL     |
|                                                                             |
| NUMBER OF DATA ELEMENTS......:       8                              8       |
| NUMBER OF ELEMENTARY FIELDS..:       6                              6       |
|                                                                             |
| INPUT LENGTH.................:      58                             58       |
| INTERNAL LENGTH..............:      54                             54       |
| OUTPUT LENGTH................:      54                             54       |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
|                                                                             |
| *** END ***                                                                 |
| O: C1 CH: -STA                                                              |
```

On-line access commands

```
LIST OF SEGMENTS

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

CHOICE              SCREEN                              UPD
------              ------                              ---

LCSaaaa             List of Segments by code             NO
                    (starting with Segment 'aaaa').

LNSaaaa             List of Segments by name             NO
                    (starting with Segment 'aaaa').

DESCRIPTION OF SEGMENT 'aaaa'
-----------------------------

CHOICE              SCREEN                              UPD
------              ------                              ---

Saaaa               Definition of Segment 'aaaa'.       YES

SaaaaCR             Instances linked to Segment         YES
                    'aaaa' via User Relations.

SaaaaGCbbb          Comments on Segment 'aaaa'          YES
                    (starting with line number 'bbb').

SaaaaGEbbb          Error messages on Segment 'aaaa'    YES
                    (starting with line number 'bbb').

SaaaaGGbbb          Generation Elements for Segment     YES
                    'aaaa'(starting with line number 'bbb').
```

```
SaaaaGObbb          Generation option for Segment 'aaaa' YES
                    (starting with line number 'bbb').

SaaaaATbbbbbb       Text assigned to Segment 'aaaa'       NO
                    (starting with text 'bbbbbb').

SaaaaLSPbbbb        List of Parent Segments for Segment   NO
                    'aaaa' (starting with Parent Segment
                    'bbbb').

SaaaaLSCbbbb        List of Child Segments for Segment    NO
                    'aaaa' (starting with Child Segment
                    'bbbb').

SaaaaX              X-references of Segment 'aaaa'.       NO

SaaaaXSbbbb         X-references of Segment 'aaaa' to     NO
                    segments (starting with Segment 'bbbb').

SaaaaXBbbbbbb       X-references of Segment 'aaaa' to     NO
                    Blocks (starting with Block 'bbbbbb').

SaaaaXQbbbbbb       Occurrences linked to Segment         NO
                    'aaaa' through User Relations
                    (starting with Relation 'bbbbbb').

SaaaaXVbbbbbb       X-references of Segment 'aaaa' to     NO
                    Documents (starting with Document
                    'bbbbbb').

SaaaaXPbbbbbb       X-references of Segment 'aaaa' to     NO
                    programs  (starting with program
                    'bbbbbb').

SaaaaXPbbbbbbCPcccccc X-references of Segment 'aaaa' to   NO
                    Call of P.M.S. (-CP) of Program
                    'bbbbbb' starting with Macro-Structure
                    'cccccc').

SaaaaXPbbbbbbWccddd X-references of Segment 'aaaa' to     NO
                    Work Areas (-W) of Program 'bbbbbb'
                    (starting with Work Area 'cc', line
                    number 'ddd').

SaaaaXObbbbbb       X-references of Segment 'aaaa' to     NO
                    Screens (starting with Screen 'bbbbbb').

SaaaaXObbbbbbCPcccccc
                    X-references of Segment 'aaaa' to     NO
                    Call of P.M.S.(-CP) of Screen 'bbbbbb'
                    (starting with Macro-Structure
                    'cccccc').

SaaaaXObbbbbbWccnnn X-references of Segment 'aaaa' to     NO
                    Work Areas (-W) of Screen 'bbbbbb'
                    (starting with Work Area 'cc', line
                    number 'nnn').

SaaaaSSbn           Definition of the sub-schemas or     YES
                    sub-systems of Segment 'aaaa' in
                    the Pactables function (starting with
                    sub-schema 'n' with 'b' = 's', or
                    sub-system 'n' with 'b' = 'y'.
```

SaaaaCEbbb	Call of Elements/Attributes of Segment 'aaaa'(starting with line number 'bbb').	YES
SaaaaCEbbbGCccc	Comments on the Element/Attribute called on line 'bbb' of Segment 'aaaa' (starting with Comments line number 'ccc").	YES
SaaaaCEbbbGEccc	Error message on the Elem/Attribute called on line 'bbb' of Segment 'aaaa' (starting with line number 'ccc').	YES
SaaaaCEbbbGGccc	Generation Elements on the Element/ Attribute called on line 'bbb' of Segment 'aaaa' (starting with line number 'ccc').	YES
SaaaaDBEbbb	SQL view source for view 'aaaa' (starting with line 'bbb').	YES
SaaaaLALbbb	Level, address and length of Segment 'aaaa' (starting with line 'bbb').	NO
SaaaaDEDbbb	Data Element details of Segment 'aaaa' (starting with line 'bbb').	NO
	If this choice is used in C2 option, the relational label replaces that of the Data Element.	
SaaaaCNbbbbbb	List of constraints of Segment 'aaaa' integrity (from the block 'bbbbbb')	NO
SaaaaSTA	Statistics on Segment 'aaaa'.	NO
SaaaaACT	Activity calculation on Segment 'aaaa'.	NO

NOTE: After the first choice of type 'Saaaa', 'Saaaa' can be replaced with '-'.

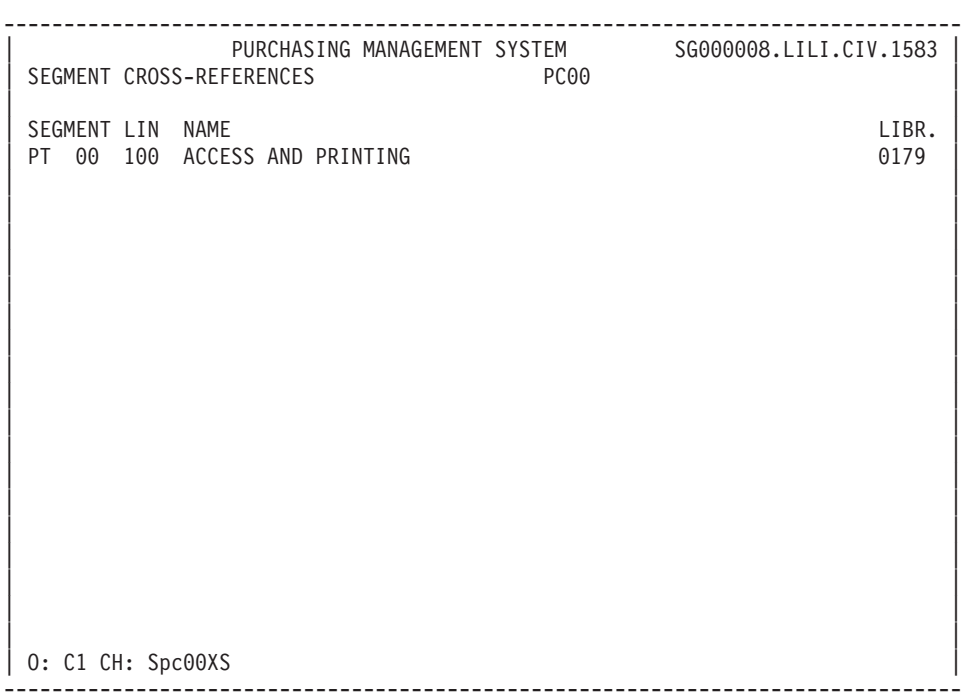
All notations between parentheses are optional.

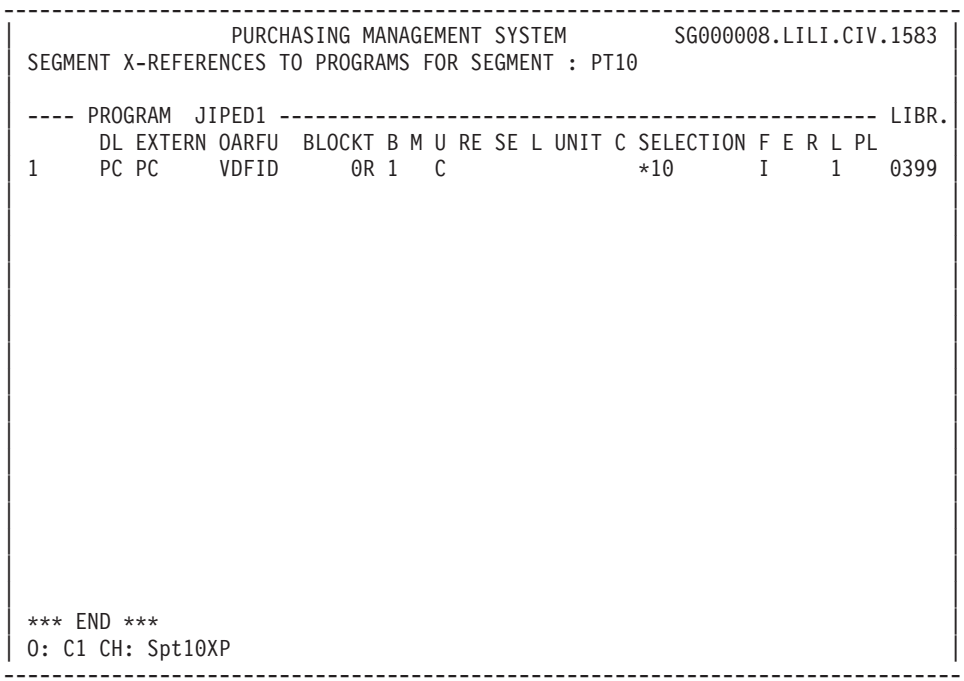
```
                  PURCHASING MANAGEMENT SYSTEM        SG000008.LILI.CIV.1583
LIST OF SEGMENTS BY CODE

CODE    NAME OF THE SEGMENT OR D.S.          TYPE OF THE D.S.          LIBR
CO      ORDER PREPARATION                    Z DATA STRUCTURE          0059
CO00    ORDER ITEM                                                     0059
LE      PACBASE ERROR MESSAGES               Z DATA STRUCTURE          *CEN
LE00    PACBASE ERROR MESSAGES                                         *CEN
OI      PURCHASE ORDER INFORMATION           Z DATA STRUCTURE          0059
OI00    PURCHASE ORDER KEYS                                            0059
OI10    BASIC ORDER DATA                                               0059
OI20    ORDER LINE ITEM DATA                                           0059
PR      PRODUCT FILE                         Z DATA STRUCTURE          0059
PR00    COMPLETE PRODUCT RECORD                                        0059
PR01    PRODUCT INFORMATION                                            0059
TT      TABLE DESCRIPTION                    G TABLES                  0093
TT20    AREA CODES                                                     0093
VE      VENDOR FILE                          Z DATA STRUCTURE          0059
VE00    VENDOR INFORMATION                                             0059
XO      Structure for On-line guide          Z DATA STRUCTURE          *CEN
XO01    Password                                                       *CEN
XO02    Root segment                                                   *CEN

O: C1 CH: LCS
```

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
SEGMENT LIST OF PARENT SEGMENTS FOR SEGMENT : PC10

PRNT BLOCK  LIN SET    MODEL   OCC. NAME  OF REL./COMMENTS             LIBR.
CO00 SPCH01 100 H01001 STCOUN     0 STATE/COUNTY                       *JIA




O: C1 CH: Spc10LSP
```

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
SEGMENT LIST OF CHILD SEGMENTS FOR SEGMENT :  CO00

CHLD BLOCK  LIN SET    MODEL   OCC. NAME OF REL./COMMENTS              LIBR.
PC10 SPCH01 100 H01001 STCOUN     0 STATE/COUNTY                       *JIA




O: C1 CH: Sco00LSC
```

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
SEGMENT CROSS-REFERENCES                    PC00

SEGMENT LIN  NAME                                                      LIBR.
PT  00  100  ACCESS AND PRINTING                                       0179 
```




PURCHASING MANAGEMENT SYSTEM SG000008.LILI.CIV.1583
SEGMENT X-REFERENCES TO PROGRAMS FOR SEGMENT : PT10
---- PROGRAM JIPED1 ---------- LIBR.
DL EXTERN OARFU BLOCKT B M U RE SE L UNIT C SELECTION F E R L PL
1 PC PC VDFID 0R 1 C *10 I 1 0399
*** END ***
O: C1 CH: Spt10XP

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
SEGMENT : TT20        CITY CODES
SUB-SCHEMAS

A T N : NAME                           ENT.                              LIBR.
  S 1 : CITY INFORMATION                          SUB-SCHEMA   1         0289
  S 2 : AREA CODE                                 SUB-SCHEMA   2         0289
  Y 1 : DISTRIBUTING OFFICES           0500       SUB-SYSTEM   1         0289
  Y 2 : LOCATIONS                      1500       SUB-SYSTEM   2         0289
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :

O: C1 CH: Stt20SS
```

```
                  PURCHASING MANAGEMENT SYSTEM        SG000008.LILI.CIV.1583
ACTIVITY                     AG00    test for segment common part

TEXT   PA LIN DESCRIPTION OF THE ACTIVITY                                LIBR.
TEST01 AG 000 activity calculation                                       0377
          060 TEST                          3*N                 N=  357 0377
          080 TEST                          N+1                 N=  357 0377
          100 TEST                          N/2                 N=  357 0377
          120 TEST                          12                          0377
FREQUENCY      1  SUB-TOTAL -->       1619

                      TOTAL -->       1619









*** END ***
O: C1 CH: Sag00ACT
```

Generation and/or printing

Lists and description reports on data structures may be obtained by entering certain commands on the Generation and Print Commands (GP) screen.

LISTS

LCS: List of Segments sequenced by code.

C1 OPTION: Without explicit keywords,

C2 OPTION: With explicit keywords.

LKS: List of Segments sequenced by keyword.

After typing LKS, a selection field (SEL:) enables the user to choose implicit ('L') or explicit ('M') keywords, or both (' '). Keywords are entered on a continuation line or

DESCRIPTION

DCS : Segment description On the GP screen, enter the Data Structure code in the ENTITY field. The segment selection is made by listing the 2-characters numbers (00,10,20..) on the continuation line. To get the continuation line, put an '*' in the 'S' field.

The format of the Elements may be selected. After typing 'DCS', a FORMAT: field appears.

The valid values are :

.I = internal,

.E = input,

.S = output.

.R = internal format but if there is a relational

name, it replaces the Data Element label.

Regardless of the selected Library code, the print option for this entity can only be '1' or '2' (C1, U1, etc., C2, U2, etc.).

Option '1' generates the printing of:

- The definition line of the data structure:
  Associated keywords and general comments lines,
  Cross-references to programs and screens,
  The list of segments belonging to the data structure,
- The definition line of each segment:
  Associated keywords and comments lines,
  Cross-references to all other entities,
- Description lines of each segment:
  The list of sub-schemas and sub-systems (Pactables only)
  The call of elements (including the comments),
  The statistics of the segment (number of elementary elements and record length).

NOTE: For table segments, see the Pactables Reference Manual.

Option '2' provides the same listings as above, but adds a listing of the texts assigned to the data structure and the segment.

Chapter 4. Reports

Definition screen (R)

The Report Definition screen is accessed by entering in the CHOICE field:

CH : Rddd

where 'ddd' is replaced by the code of the report code.

GENERAL CHARACTERISTICS

When used in a program, the user may opt to:

- Print all the reports with the same prefix,
- Print only selected reports.

  For more details, read Chapter 'Program', Sub-chapter 'Data Structures Call', 'Report Selection' part.

A report cannot be generated by itself. The report is included in a batch program on the Data Structure call screen.

This causes an F8x edit function to be generated, where x is the Report code.

```
                  PURCHASING MANAGEMENT SYSTEM        SG000008.LILI.CIV.1583
                         1 2
 REPORT CODE...........: EO1

 NAME.................: VENDOR ACTIVITY  3

 COMMENTS.............: 4

 NATURE...............: E REPORT  5
 PRINTER TYPE.........: L         6

 LINE LENGTH..........: 132       7
 FORMAT FOR TOTALS     : INTEGER........: 11   8
                         DECIMAL PLACES.: 07   9



 EXPLICIT KEYWORDS..:  10


 UPDATED BY.........:             ON :           AT :   :   :  LIB :
 SESSION NUMBER.....: 0059        LIBRARY......: CIV    LOCK :

 O: C1 CH: Reo1                      ACTION:
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			Report code
			The Report code consists in three numeric or alphabetic characters.
1	30		NAME OF REPORT (REQUIRED IN CREAT)
			Do not begin by 'Report of....'.
			This name must be as explicit as possible. It is used for the automatic creation of keywords, as detailed in Chapter "Search for Instances" in the Character Mode User Interface Guide.
2	36		REPORT COMMENTS
			For documentary purposes only: Enter comments.
3	1		NATURE CODE
			This code is for documentary purposes. It identifies the nature of the report and is used to restrict listings of reports to those of the specified nature: (CH: LTRnRddr where n = NATURE CODE).
		'E'	Report,

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'K'	Indicates a screen layout: a report can be used as a way to paint a screen layout prior to implementation.
		'L'	Table,
		'I'	Indicates a report that is a form, to be subsequently filled in.
4	1		REPORT PRINTER TYPE
			This field contents cannot be blank.
		'L'	Default option: standard line printing.
		'P'	Layout of a report to be printed on a 3800 printer, with character set codes specified in the Report Layout lines (in column labeled 'C').
			NOTE: These character sets are not taken into account when the Report occurrence is used as a Volume Print Layout.
		'S'	Layout of a report to be printed on a 3800 printer, without definition of character sets. For a Report used with Microfocus, this value generate a skip character.
5	3		LINE LENGTH (MAXIMUM)
			PURE NUMERIC FIELD
			This value identifies the length of the longest report constant line which is taken in account at generation.
			Default option: 132.
		'1 to 264'	The length indicated here will be the one considered at generation time for the calculation of the WORKING-STORAGE length for report descriptions.
			Note: The actual length of the report to be printed is determined from the value entered on the Report Description (-D) Screen Top. Example: You may want a report containing technical comments in columns 81 to 132 but truncate the display in the report for the users to the 80th column. This can be accomplished by using the 132 default here, and entering 80 as the value of the LINE LENGTH (MAXIMUM) field on the Report Description screen.
			FORMAT FOR TOTALS
			Internal accumulators, (counters) are generated by PACBASE when the report contains data elements that are to be totaled.
			The default value is 9(11)V9(7).

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			The total number of digits must remain within the limit allowed by the compiler (this is not verified by VA Pac).
6	2		NO. OF DIGITS LEFT OF THE DECIMAL
			PURE NUMERIC FIELD
		'>00'	Default option: 11.
7	2		NO. OF DIGITS RIGHT OF THE DECIMAL
			PURE NUMERIC FIELD
			Default option: 7.
8	55		EXPLICIT KEYWORDS
			This field allows you to enter additional (explicit) keywords. By default, keywords are generated from the instance's name (implicit keywords).
			Keywords must be separated by at least one space. Keywords have a maximum length of 13 characters which must be alphanumeric. However, '=' and '*' are reserved for special usage and are therefore ignored in keywords.
			Keywords are not case-sensitive: uppercase and lower-case letters are equivalent.
			NOTE: Accented and special characters can be declared as equivalent to an internal value in order to optimize the search of instances by keywords (Administrator workbench, 'Window' menu, 'Parameters browser' choice, in 'Special Characters' tab).
			A maximum of ten explicit keywords can be assigned to one entity. For more details, refer to the 'Character Mode User Interface' guide, chapter 'Search for Instances', subchapter 'Searching by Keywords'.

Layout screen (-L)

The purpose of the Layout (-L) screen is to describe a page of the end Report; all significant lines are described at least once. It is then possible :

- To present it to the end-user for discussion,
- To directly define all the constant elements (Title, labels..) of the Report.

The layout is normally produced during the functional analysis phase.

The screen contains the following fields:

- an identifier line which specifies the REPORT CODE, name and line length.

- a LINE NUMBER used to sequence the lines of the layout.
- a CONSTANT PART NUMBER, used to identify the different titles, labels, column headings... that appear on the Report.
- the LINE SKIP BEFORE PRINTING, which is used in prototyping.
- a CHARACTER SET OPTION field (which will only appear on the screen if the REPORT PRINTER TYPE = 'P').
- a LAYOUT LINE, which shows the column numbers. As a suggestion, left-justifying the Report will enable easier referencing.

The Report lines cannot contain the litteral delimiter in use on site (single (') or double (") quote).

As you are drawing the Report layout, you assign a CONSTANT PART NUMBER to the lines containing literals which are to appear on the printed Report. These numbers must start with '01' and increase consecutively. The variable fields on these lines (if any) which will receive input when the Report is generated, will overlay the portion of the layout line, as specified on the Report Description (-D) screen.

ACCESS TO THE DIFFERENT PARTS OF THE LAYOUT

The Layout screen has a maximum of 264 columns. Thus, to access the different parts of the layout screen (scrolling right or left, up or down), enter the following input in the CHOICE field:

CH: RddeLnnCppp

which will display the Layout from Line 'nn' and Column 'ppp'.

Use the following commands to view specific parts of the layout:

- '<': shift to the left; for example enter "<20" to shift 20 columns to the left. Default shift is 66 columns.
- '>': shift to the right; for example enter ">20" to shift 20 columns to the right. Default shift is 66 columns.
- '=n': positioning on column n.
- '=': repositioning on column 001.

CONSTANT TABLES

The Report Layout (-L) screen is also used to describe the constant tables, internal to programs, even if they are not used for a printed report.

To describe such tables, the user has to:

- define a report for each table, specifying the table position length,

- no STRUCTURE NUMBER or CATEGORY value is entered,
- constants must be described on lines assigned CONSTANT PART NUMBERs, entered in the appropriate sequence,
- call the data structure into programs via the Call of Data Structures (-CD) screen using an ORGANIZATION of 'W', and selecting the tables needed as you would any report.

No functions will be generated for reports without structures and categories.

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
REPORT LAYOUT :       1 EO2 VENDOR ACTIVITY                       LENGTH= 132

2 3  4  5    6
A LN CP S             1    1    2    2    3    3    4    4    5    5    6
             1...5....0....5....0....5....0....5....0....5....0....5....0....
  03  1 *    Date: 10/11/88          Q U A R T E R L Y   V E N D O R   A C T
  06  2 2          Activity of vendor: CALIBRATION ENGINEERING, INC.
  09  3 2    ------------------------------------------------------------------
  12  4       | PRODUCT     | PRODUCT DESCRIPTION            | PRODUCT    | QUANT
  15  5       | NUMBER      |                                | PRICE      | RECEI
  18          ------------------------------------------------------------------
  21  6       | X362-1A441  | MASS SPECTROMETER              |    456.78  | 12318
  24          |             |                                |            |
  27    9     |             |                                |            |
  30          ------------------------------------------------------------------
  33  7 2                                                          Total amoun
  36  8 2

O: C1 CH: -L
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
1	3		Report code (REQUIRED)
			The Report code consists in three numeric or alphabetic characters.
2	1		ACTION CODE
		'C'	Creation of the line
		'M'	Modification of the line
		'D' or 'A'	Deletion of the line
		'T'	Transfer of the line
		'B'	Beginning of multiple deletion
		'G'	Multiple transfer
		'?'	Request for HELP documentation
		'E' or '-'	Inhibit implicit update
		'X'	Implicit update without upper/lowercase processing
3	2		LINE NUMBER (REQUIRED)
			PURE NUMERIC FIELD

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'00-99'	It is advisable to leave gaps in the numbering sequence to allow for future line insertions as necessary.
4	2	NUMER.	LINE LABEL NUMBER (REQUIRED)
			PURE NUMERIC FIELD
			This value identifies lines that contains labels to be printed in the actual Report. The label number is to be indicated on the first part of the line, and it is automatically set to the next parts.
			BATCH SYSTEMS DEVELOPMENT
		'blank'	Lines without constant parts.
		'01-99'	Lines with constant parts.
			Lines with constants are stored in a table. This number is the subscript. Therefore, begin with '01' and number the lines consecutively. ('00' is not valid).
			In Batch mode, this value need not be repeated for lines that are described using more than one part.
			A constant line cannot be deleted unless it is the last one of the report. To delete a line, either renumber the lines, or delete the line and renumber the lines, or delete the line and renumber the last constant line with the deleted line value. Note that the Description (-D) screen field must also be updated to reflect the change.
			CONSTANT PART NUMBERS are not necessarily in the same sequence as Line Numbers.
			The value entered here can only be used once per layout.
			P.D.M. EXTENSION
			The Line Label Number identifies the Layout component. In some cases, it may be necessary to create several lines of the same label number.
			NOTE: ALL print windows must have a minimum length of 30 characters.
			1. DOCUMENT PRINT LAYOUT:
			For detailed information, refer to Paragraph Volume Layout Description Principles.
		1	Line for setting parameters' values.
		10	Line for page header or footer.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		70	No.0 Print Window (required). Default Print Window used in relation to A-, G-, H-, and all S-type Volume Description lines.
			NOTE: This default value may be modified by the $DL parameter and at the Volume Description line level, in the W-labeled field. Print Windows specified in Text lines necessarily override these defaults.
			Enter between the ″$″ delimiters the number of repetitive lines per page (default=48).
			It is recommended that the No.0 Print Window's length be at least 78 characters.
			The No.0 Print Window's framing characteristics also apply to: . Section Titles generated with $VT=nm and printed in the section's title-page. . Title lines generated with $GT=1 using the GV or GA print option, and printed in the call's first page.
		71 to 79	Print window No.1 to print window No.9.
			NOTE: Line labels are not necessarily entered in increasing order. A 10-labeled line which describes a page footer must be entered after the 7n-labeled lines.
			2. SPECIFIC LAYOUT:
			Line labels must be entered in increasing order.
			*** TITLE LAYOUT
		20	Title-page header
		25	. When used for titles printed in title-pages: - Print window - Framing characteristics
			. If used for the Table of Contents and Index titles when printed in their title-page: - Print window only
			Framing characteristics are those specified in the 35-labeled line for the Table of Contents, in the 55-labeled line for the Index.
			. This line also includes the number of lines in a title-page (header and footer lines excluded), followed by the number of the line where the title is printed.
			Default: number specified in the 70-labeled line of the Volume Print Layout.
			. Also used for title-page blank lines to specify framing characteristics.
		29	Title-page footer

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			*** TABLE OF CONTENTS LAYOUT
		30	Table of Contents header
		35	Number of lines in a Table of Contents page (header and footer lines excluded).
			Also used for Table of Contents blank lines & Table of Contents title line when printed in its title- page (See also 25-labeled line) to specify framing characteristics. This line is required.
		39	Table of Contents footer
		40	Title for the Table of Contents
		41 to 49	Print windows for (sub)entries in the Table of Contents and framing characteristics
			*** INDEX LAYOUT
		50	Index header
		55	Number of lines in an Index page (header and footer lines excluded). Also used for Index blank lines & Index title line when printed in its title-page (See also 25-labeled line) to specify framing characteristics. This line is required.
		59	Index footer
		60	Index title
		61	Print window & framing characteristics for Index Entries
		62	Print window & framing characteristics for Index Comments
		63	Print window & framing characteristics for Index references, i.e. Index lines where page numbers are printed.
			*** GENERATED TITLES LAYOUT
		71 to 79	Print windows for Level-1 to Level-9:
			. Generated Section Titles printed in sections' first pages ($VT=nm),
			. Generated Titles printed in calls' first pages ($GT=1, GV or GA print option).
5	1		PAGE BREAK - LINE SKIP
			BATCH SYSTEMS DEVELOPMENT

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			The value entered in this field is used for paging and line spacing when generating the Report Layout, i.e. with the DCR Generation-Print request, or an R...L call in a Document Description. Paging and line spacing for the actual printed Report is specified in the SKP-labeled fields in the Report Description (-D).
		' *'	Page break.
			NOTE: A page break is automatically generated on the first line of a Volume Layout.
		'1 to 9'	Line spacing from single spacing (1) to 9x spacing (9) (1 is the default value).
		' 0'	Overprinting. This value is reserved for type 3800 layouts. It is interpreted as single line spacing in the Layout Description with formatting.
			P.D.M. EXTENSION
			Page break or line skip associated with the Print Window unless specified otherwise in the called Text Description.
		' *'	Page break.
		'1 to 9'	Line spacing from single spacing (1) to 9x spacing (9) (1 is the default value).
6	132		PRINT LITERAL/DOCUMENT PRINT LAYOUT
			Simple or double quotes are replaced by blank characters in this field IF the same quote is the delimiter chosen on the Library Definition. This replacement prevents COBOL compilation errors due to the presence of this delimiter in the 'values'.

Call of Elements screen (-CE)

The purpose of this screen is to describe the data elements of each Report.

This is achieved by listing the data elements and identifying their position on the layout line, the source of the data and under what conditions the data is to be moved into the data element.

Lines that contain the same data elements using the same formats and locations may be described as the same structure even if the print condition differs. For example, when totals are to be printed at different control break levels, only one structure is needed. When a single data element is to be filled with different data, depending upon the condition, increment the LINE

NUMBER value within the structure. The STARTING ADDRESS (COLUMN NUMBER) remains the same, and the various conditions may be entered.

OPERATION CODE

C1: default value.

C2: displays the output format of the data element, and the BLANK WHEN ZERO specification.

```
                PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
REPORT CALL OF ELEMENTS1 EO1 VENDOR ACTIVITY

2 3  4      5   6   7 8 9 10  12    13  14
                          11
A ST ELEM   L : STA C O W SOURCE    FLD CONDITION                      LIBR.
  01 XDAT8  0 :   7   I * DATOR                                        0059
  01 XPAGE  0 :  90   M 5 EO001PC                                      0059
- -- ------ - : --- - - - ------------ ------------------------------ -----
  02 VENAME 0 :  27   M   VE00VENAME                                   0059
- -- ------ - : --- - - - ------------ ------------------------------ -----
  03 PRNUMB 0 :   3   M   CO00PRNUMB    CATX = 'CA'                    0059
  03 PRDESC 0 :  16   M   PR00PRDESC    CATX = 'CA'                    0059
  03 PRPRIC 0 :  48   M   PR00PRPRIC    CATX = 'CA'                    0059
  03 ITQREC 0 :  60   M   CO00ITQREC    CATX = 'CA'                    0059
  03 6LIB10 A :  71   M * 'MILLIMETERS' 1-PR00-PRMEAS = 'MM'           0059
  03 6LIB10 B :  71 *                   AND CATX = 'CA'                0059
  03 6LIB10 C :  71   M * 'GRAMS'       1-PR00-PRMEAS = 'GR'           0059
  03 6LIB10 D :  71 *                   AND CATX = 'CA'                0059
  03 6LIB10 E :  71   M * 'CENTIMETERS' 1-PR00-PRMEAS = 'CM'           0059
  03 6LIB10 F :  71 *                   AND CATX = 'CA'                0059
  03 6LIB10 G :  71   M * 'METERS'      1-PR00-PRMEAS = 'ME'           0059

O: C1 CH: -CE
```

```
                  PURCHASING MANAGEMENT SYSTEM        SG000008.LILI.CIV.1583
REPORT CALL OF ELEMENTS1 EO1 VENDOR ACTIVITY

2 3  4      5   6   7 8 9 10  12    13  15                          16
                              11
A ST ELEM   L : STA C O W SOURCE    FLD PICTURE                    : Z  LIBR.
  01 XDAT8  0 :   7   I * DATOR         X(8)                       :    0059
  01 XPAGE  0 :  90   M 5 EO001PC       ZZ9                        :    0059
- -- ------ - : --- - - - ------------ ---------------------------:--- -----
  02 VENAME 0 :  27   M   VE00VENAME    X(25)                      :    0059
- -- ------ - : --- - - - ------------ ---------------------------:--- -----
  03 PRNUMB 0 :   3   M   CO00PRNUMB    X(12)                      :    0059
  03 PRDESC 0 :  16   M   PR00PRDESC    X(20)                      :    0059
  03 PRPRIC 0 :  48   M   PR00PRPRIC    ZZ9,99                     :    0059
  03 ITQREC 0 :  60   M   CO00ITQREC    ZZZZ9                      : Z  0059
  03 6LIB10 A :  71   M * 'MILLIMETERS' X(15)                      :    0059
  03 6LIB10 B :  71 *                                             :    0059
  03 6LIB10 C :  71   M * 'GRAMS'       X(15)                      :    0059
  03 6LIB10 D :  71 *                                             :    0059
  03 6LIB10 E :  71   M * 'CENTIMETERS' X(15)                      :    0059
  03 6LIB10 F :  71 *                                             :    0059
  03 6LIB10 G :  71   M * 'METERS'      X(15)                      :    0059

O: C2 CH: -CE
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
1	3		Report code (REQUIRED)
			The Report code consists in three numeric or alphabetic characters.
2	1		ACTION CODE
		'C'	Creation of the line
		'M'	Modification of the line
		'D' or 'A'	Deletion of the line
		'T'	Transfer of the line
		'B'	Beginning of multiple deletion
		'G'	Multiple transfer
		'?'	Request for HELP documentation
		'E' or '-'	Inhibit implicit update
		'X'	Implicit update without upper/lowercase processing
3	2	NUMER.	Structure number (REQUIRED)
			PURE NUMERIC FIELD

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'01 to 98'	The structure number sequence must start from 01 (or 00) and contain no gaps.
			This value becomes a subscript for a table containing all the structures.
			Each structure listed on the Report Description (-D) screen must have at least one corresponding line on the Report Call of Elements (-CE) screen. For structures that come from other reports, (see TYPE OF LINE IN REPORT on the Report Description (-D) screen), the elements belonging to the structure are listed on the Call of Elements (-CE) of the report that describes the detail line, as does the STRUCTURE NUMBER value. For example, DDR is a report with a detail line to be used in report DDS. This detail line is located in Structure 06 of DDR. The data elements for this structure are entered on the Call of Elements (-CE) of DDR. STRUCTURE NUMBER = '06' does not appear on DDS's Call of Elements screen. Note: In our example, there would have to be a structure '01' to '05' to avoid gaps.
			Note on deletion of structures:
			When a structure, other than the last one, is no longer required, either a dummy structure must be maintained or the last structure renumbered with the value of the one not needed. The Layout (-L) and Call of Elements screens may need to be updated to reflect the change.
			A structure cannot be deleted globally. It must be done data element by data element.
		' 00'	This value is used to identify fields required for user-defined spooling. (See USAGE OF D.S. = 'J' on the Call of Data Structures (-CD) screen, and also, "DIRECT PRINT /APPLIC. SPOOLING RTN." Subchapter.)
			The data elements belonging to this structure are positioned relative to the beginning of the record, and not to the beginning of the line, as is true of all other structures.
			The two data elements 'LSKP' or 'SAUT' for a French generator and 'LIGNE' are reserved. LSKP is a pointer to the SKIP field which controls line skips. LIGNE controls the placement and alignment of the layout line.
			At generation, structure '00' is taken into consideration only if the USAGE OF DATA STRUCTURE = 'J'.
4	6		DATA ELEMENT CODE (REQUIRED)

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			Enter the mnemonic code which references the Data Element independently of any Data Structure, Report or Screen to which the Data Element might belong.
			There is no need to include a Report, Screen or Segment code in the Data Element code since the System does it automatically.
			This code consists of alphabetic or numeric characters only.
			Some Data Element codes are reserved by the System for use in Data Structures, Reports or Screens and cannot be defined in the Specifications Dictionary:
		'SUITE'	Prohibited. This code is reserved for the System for program generation.
		'FILLER'	Data Element that is used for the alignment of fields.
			Options of the BSD Function:
			Error Verification fields on transaction files:
		'ENPR' 'GRPR' 'ERUT'	Used for Data Element error verification. Used for Segment error verification. Used for user defined errors.
			For more information see DATA ELEMENT CODE on the Segment Call of Elements.
			For Reports:
		'LIGNE'	Reserved for the placement and alignment of the layout line. It is used only for a '00' structure.
		'LSKP'	Reserved usage only in the '00' Report Structure. See STRUCTURE NUMBER on the Report Call of Elements.
		'SAUT'	Reserved usage. This code is the counterpart of LSKP and used with the French version of the System.
			Options of the OLSD and Pacbench C/S (TUI Client) Functions:
		'ERMSG'	Data Element for the placement of the error message.
		'LIERR'	Reserved usage. This code is the counterpart of ERMSG and used with the French version of the System.
		'PFKEY'	Used to represent the programmable function keys.
		'*PASWD'	(IMS only): Used for passwords on a specific screen.
			For more information see DATA ELEMENT CODE OR SCREEN CODE TO CALL on the Call of Elements.
5	1		CONTINUATION LINE NUMBER

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			BLANKS REPLACED BY ZEROS.
			Alphabetic or numeric character.
		blank or 0	Default value.
			Enter a value when more than one line is needed to describe a data element. This may occur when the condition is longer than the field allows, or when different values fill in the data element according to the conditions.
			The maximum number of lines per data element within a structure is 36.
6	3		STARTING ADDRESS (COLUMN NUMBER)
			PURE NUMERIC FIELD
			Enter the column number, in which the data element field begins. (Required in creation).
			This value is to be specified on the first line that concerns the data element - that is, not on a continuation line.
7	1		CONTINUATION OF CONDITION OR SOURCE
			The source or the condition of a data element may take more than one line to describe.
		'blank'	Indicates the first line.
		'*'	Indicates continuation lines.
8	1		OPERATION ON SOURCE FIELD
		'blank'	This value is used on a continuation line. (The CONTINUATION OF CONDITION OR SOURCE field contains an asterisk ('*')).
			NOTE: There must be at least as many continuation lines as there are lines needed to complete the condition.
		'M'	Move (default option if the SOURCE FIELD area contains an entry).
		'+' '-' '*' '/'	Add. Subtract. Multiply. Divide.
			NOTE: With these four values, generation of a COMPUTE. On the first line, the user must enter a '+' or 'R' value in order to indicate the beginning of a calculation.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			The division of a report is performed in the following way: Enter '+' in the Operation on Source field followed by the code of the Data Element to be divided. On a continuation line ('*' in the Continuation of Source field) enter '/' in the Operation on Source field followed by the 'divider' Data Element. The procedure is the same for a multiplication, except that '/' must be replaced by '*'.
		'R'	Provide a rounded result on the calculation. This value must be entered as the first operation line for the data element concerned (within the structure).
		'U'	Transfer of data via user-specified procedures. Only the description of the corresponding 6- Data Element is generated. A U-type line may be used: . as a complementary line to an S-type line (transfer of data after a table search), . as a continuation line if the number of source continuation lines is inferior to the number of condition continuation lines.
		'0'	Loading of the century from a DAT-CTY field initialized to '19', it can be modified.
		'1'	Loading of the century to '19' if the year is lower than the value in the DAT-CTYT field ('61' by default), loading to '20' in the other case.
		'2'	Loading of the century field to '20' if the year is lower than the value in the DAT-CTYT field ('61' by default), to '19' in the other case.
		'D'	Print a date in extended format: XX/XX/XX. The target data element must be 8 characters long, and the source, 6 characters.
		'I'	Same as with the 'D' value, except that a machine date is used and is formatted as follows: MM/DD/YY.
		'C'	A date of the form XXYYZZZZ becomes XX/YY/ZZZZ
		'E'	A date of the form XXYYZZZZ becomes YY/XX/ZZZZ Be sure that the sending field is 8 characters long and the receiving field is 10 characters long.
		'T'	Data element to be totaled, and the total printed.
			When the TYPE OF LINE IN REPORT on the Report Description (-D) screen = '*' or 'T':
			The value indicated in the SOURCE FIELD will be added to the value in the DATA ELEMENT CODE field and moved into the latter data element.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			When the TYPE OF LINE IN REPORT on the Report Description (-D) screen = '0' to '9':
			The value indicated in the SOURCE FIELD will be accumulated in either the "Intermediate Totals Accumulator" (Trst-eeeeee(n)), or in a "Grand Totals Accumulator" (Grst-eeeeee). The desired total will be moved into the data element when the appropriate break level is attained, and when the conditions are true. The total will be printed. (See Note below.)
			A set of internal accumulators is associated with each data element to be totaled. The calculation of the sum is made each time through the processing loop.
			If a data element is only printed under certain conditions, these conditions will also apply to the totaling. The total itself will only be printed on a line designated for totaling.
			The maximum number of data elements to be totaled is 99 per program.
			The conditions concerning all other data elements are entered, making sure that the data element is a part of the appropriate Report Category (CATEGORY OF REPORT field on the Report Description screen) by using the VA Pac-generated indicator 'CATX'.
			NOTE: When a basic totaling structure is defined in a report, the proper loading and moving is generated if the data element to be totaled has 'T' entered on the line containing the first occurrence of the data element within the structure.
			Example: The following is correct:
			NN 071 O QTTIT T DDSSQTTIT NN 071 1 QTTIT M * ZERO Condition
			while the next two lines do not generate the total:
			NN 071 0 QTTIT M * ZERO NN 071 1 QTTIT T DDSSQTTIT Condition
		'S'	Transfer of data after table search.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			Coding this operation takes two lines: On the first line, enter 'S' and specify the search argument in the SOURCE FIELD. On the second line, (a continuation line), enter 'U' and specify the data element to be matched. Table search can only be performed from a non-repetitive field which has been defined in the standard way (ddss-delco or x-ddss-delco). If the search is successful, the target data element will receive data from the table data element with the same name.
			SOURCE FIELD
9	1		WORKING-STORAGE PREFIX OF SOURCE
			Indicates the WORKING-STORAGE prefix area the source data element comes from.
		'*'	Indicates that the source does not have a standard PACBASE structure. The 13 characters that follow will contain the expression (data name, literal, etc.) to be integrated into the generated source language.
			The following values are used to indicate that the source data element has a standard structure; the value entered replaces the 'w' in w-ddss-eeeeee.
			The values below may be used for areas other than the ones mentioned in the description.
		'blank'	This is the read area of a file, as generated in the FILE SECTION.
		'1'	Normally used for the processing area for files with control breaks, and tables.
		'2'	This is the update area of principal files.
		'5'	These are lines directly related to the report itself like record counter fields, line count fields, etc.
		'6'	This value is used for the output area.
			Other numeric and alphabetic values may also be used for user-defined prefixes.
10	2		SOURCE FIELD - FIRST PART
			For sources that are data elements:
			Enter the DATA STRUCTURE CODE IN THE PROGRAM of the data structure containing the source data element.
			For sources that are literals:
			Enter the beginning of the literal (starting with a quote).

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			Note: For literals longer than 11 characters, you must use the Work Areas (-W) screen and define a specific VALUE clause.
11	2		SOURCE FIELD - SECOND PART
			For sources that are data elements:
			Enter the SEGMENT CODE of the segment containing the source data element.
			For sources that are literals:
			Enter a continuation of the literal. If the literal value ends in this field, enter the close quote.
12	6		SOURCE FIELD - THIRD PART
			For sources that are data elements:
			Enter the DATA ELEMENT CODE of the source data element (default if the WORKING-STORAGE PREFIX OF SOURCE value is not '*', and if the SOURCE FIELD is not blank).
			For sources that are literals:
			Enter a continuation of the literal. If the literal value ends in this field, enter the close quote.
13	3		SOURCE FIELD - LAST PART
			FALSE NUMERIC FIELD
			For sources that are data elements:
			This field is used to identify indexes.
		'blank'	No index
		001 to 999	Number of repetitions (OCCURS)
		'nnn'	User defined index name
		'I**'	The standard look-up index for tables (USAGE OF DATA STRUCTURE = 'T' or 'X' or a Work Areas table): The index is generated in the form IddssR, where ddss = DATA STRUCTURE and SEGMENT CODEs.
		'*cc'	'*' is the fixed code and 'cc' is the category code.
			It is the standard index for repetitive category cc. The index is generated in the form Jddrcc, where ddr = REPORT CODE cc = CATEGORY CODE (repetitive category).
			For sources that are literals: Where relevant, enter the continuation of the literal. Enter the close quote character to end the literal.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
14	32		CONDITION
			This field is used to indicate the conditions under which the source should be transferred to the target. The condition may take several consecutive lines. This is indicated by an asterisk ('*') in the CONTINUATION OF CONDITION OR SOURCE field.
			Format of entry:
			For IF conditions, use COBOL format but omit the 'IF'.
			For ANDs, ORs etc., use COBOL format.
			Note: The period (full stop) is generated automatically and therefore should not be entered by the user.
15	14		PICTURE : OUTPUT FORMAT
			This field is viewed with OPERATION field value C2: O: C2 CH: -CE
			For data elements defined to the Specifications Dictionary, this field cannot be modified. It displays the OUTPUT FORMAT as defined on the Data Element Definition Screen.
			For data elements not defined to the Specifications Dictionary, this field is used to specify the output format of the element, using COBOL syntax. This can be modified.
16	1		GENERATION CLAUSE BLANK WHEN ZERO
			This field is viewed with OPERATION field value C2: O: C2 CH: -CE
			For data elements defined in the Specifications Dictionary, this field cannot be modified. It displays the BLANK WHEN ZERO CLAUSE option as entered on the Data Element Definition screen.
			For data elements not defined in the Specifications Dictionary, this field may be used to cause the generation of the BLANK WHEN ZERO clause.
		'Z'	Generate the BLANK WHEN ZERO clause.

Description screen (-D)

The Report Description screen has a two-fold purpose:

- To define the general characteristics of a report: the number of characters per line and lines per page, segment type overlay, print condition, etc.,

- To position the report lines: lines are grouped into categories to be printed under the same condition. Each line is composed of a constant, a structure, a skip character and additional elements.

The general characteristics are entered using the description Screen Top, sometimes referred to as the 'E-line'. The screen layout for this part of the screen, along with a detailed description of the fields follows.

A screen layout for the Description Screen Body appears subsequently with the details concerning these fields.

Description screen top

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
REPORT DESCRIPTION:     1 EO1 VENDOR ACTIVITY

A: 2 LINE LENGTH: 3  132 LI PAGE: 4 60 CAT TBL INST: 5   WR OPT: 6 SECTION: 7
     COMMENTS....: 8             CONDITIONS 9 CO-CF2 = 1

A CA LIN T TLI ST CP SKP FUSF COMMENTS      CONDITIONS
  BA 100        1 01 01*      HEADING       ITB1 = 1
  BA 120        2 02 02                     OR 5-EO00-1LC NOT < 5-EO00-1LCM
  BA 140          03 03
  BA 160          04 01
  BA 180          05 01
  BA 200          03 01
- -- --- - --- -- -- -- --- ---- ------------ --------------------------------
  CA 100 *      3 06 01  96BA CURRENT LINE
- -- --- - --- -- -- -- --- ---- ------------ --------------------------------
  DA 100          03 01       FRAME CLOSING FTB1 = 1
  DA 120                                    OR 5-EO00-1LC NOT < 5-EO00-1LCM
- -- --- - --- -- -- -- --- ---- ------------ --------------------------------
  EA 100 1      3 07 02       TOTAL         FTB1 = 1
  EA 120        4 08 01
- -- --- - --- -- -- -- --- ---- ------------ --------------------------------

O: C1 CH: -D
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
1	3		Report code (REQUIRED)
			The Report code consists in three numeric or alphabetic characters.
2	1		ACTION CODE (REQUIRED)
			The different ACTION CODE values are listed in the Character Mode User Interface Guide for on-line mode and for those used in batch mode, see ″OPTIONS SPECIFIC TO BATCH MODE″ or ″GENERATION AND/OR PRINTING″ Subchapters.
		C	NOTE: An explicit CREATE action code value must be entered when the report is first being created.
3	3		LINE LENGTH (MAXIMUM)
			PURE NUMERIC FIELD
			Default option: 132. This code indicates the line length.
4	2		LINES PER PAGE
			PURE NUMERIC FIELD
			Default option: 60.
5	4		NO. OF INSTANCES IN CATEGORY TABLE

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			PURE NUMERIC FIELD
			Enter the number of positions to allocate to store the different categories in the report (at generation).
		'100'	Default.
		'0000'	Rather than using the category table to control the organization of printing the categories, the categories are printed directly.
			Note: If the number of positions is higher than 1000, the table is not generated.
6	1		WRITE OPTION : BEFORE OR AFTER
		'Blank'	Print options are generated according to the hardware variant indicated at the library level.
			Example: 'WRITE AFTER' for GCOS7 (variant 4). 'WRITE BEFORE' for GCOS8 (variant 5).
			In the case of conversion libraries, the print options are automatically reformulated according to the library variant.
		'N'	Prohibits any automatic reformulation of the print option, in a conversion library.
		'*'	Generation of 'WRITE BEFORE' statement.
7	2		SECTION PRIORITY
			This field is used with hardware requiring program segmentation due to small memory capacity. For information, consult a COBOL manual.
			Generates a segment type overlay between print functions in a program. It should only be used if input data structures to print programs are sorted by report code and if the COBOL variant is ANSI. Priorities less than 50 generate an overlay only in association with the 'SEGMENT LIMIT' clause, to be inserted in the ENVIRONMENT DIVISION.
8	13		COMMENTS
			The comment entered on the screen top refers to the whole report. Comments entered on the screen body normally refer to the individual lines.
9	35		CONDITIONS OF REPORT EXECUTION
			On the screen top - (the "E-line"):
			Enter conditions relevant for report execution.
			On the screen body:

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			Enter conditions concerning the execution of the Category of Report.
			Format of entry:
			Use the COBOL format to enter conditions but do not enter 'IF', nor GO TO, and do not enter any period.

Description screen body

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
REPORT DESCRIPTION:     1 EO1 VENDOR ACTIVITY

A:   LINE LENGTH:    132 LI PAGE:   60 CAT TBL INST:      WR OPT:   SECTION:
     COMMENTS....:               CONDITIONS   CO-CF2 = 1

2 3  4   5 6   7  8  9 10 11  12            13
A CA LIN T TLI ST CP SKP FUSF COMMENTS      CONDITIONS
  BA 100        1 01 01*      HEADING       ITB1 = 1
  BA 120        2 02 02                     OR 5-EO00-1LC NOT < 5-EO00-1LCM
  BA 140          03 03
  BA 160          04 01
  BA 180          05 01
  BA 200          03 01
- -- --- - --- -- -- --- ---- ------------- --------------------------------
  CA 100 *      3 06 01  96BA CURRENT LINE
- -- --- - --- -- -- --- ---- ------------- --------------------------------
  DA 100          03 01       FRAME CLOSING FTB1 = 1
  DA 120                                    OR 5-EO00-1LC NOT < 5-EO00-1LCM
- -- --- - --- -- -- --- ---- ------------- --------------------------------
  EA 100 1      3 07 02       TOTAL         FTB1 = 1
  EA 120        4 08 01
- -- --- - --- -- -- --- ---- ------------- --------------------------------

O: C1 CH: -D
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
1	3		Report code (REQUIRED)
			The Report code consists in three numeric or alphabetic characters.
2	1		ACTION CODE
		'C'	Creation of the line
		'M'	Modification of the line
		'D' or 'A'	Deletion of the line
		'T'	Transfer of the line
		'B'	Beginning of multiple deletion
		'G'	Multiple transfer
		'?'	Request for HELP documentation
		'E' or '-'	Inhibit implicit update
		'X'	Implicit update without upper/lowercase processing
3	2	ALPHA.	CATEGORY OF REPORT
			(maximum of 39 lines per category.)

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'AB to ZY'	The value entered here is used to differentiate categories from one another. Report lines are grouped together according to the conditions under which they will be printed (totaled, etc...).
			Leaving gaps in the category sequence will facilitate future modifications.
			Categories containing a detail line with elements to be totaled - (TYPE OF LINE = '*' or 'T'):
			.can only contain one detail line,
			.cannot contain a total line,
			.cannot be repetitive,
			.can contain other ordinary lines.
			Categories used for the lines containing the totals - (TYPE OF LINE = '0' to '9'):
			.can contain several total lines,
			.cannot have a detail line,
			.cannot be repetitive,
			.can contain other ordinary lines.
		'ZZ'	Prohibited.
		'AA'	Not recommended.
4	3		Line number
			Numeric.
			It is advisable to begin with line number '100' and then number in intervals of 20. This facilitates subsequent line insertions, as necessary.
5	1		TYPE OF LINE IN REPORT
			This field is used to identify the type of category.
			To designate a Header, repetitive area, or Footer:
		'A'	This value applies to repetitive categories only. This indicates the first line of a top-of-page category (header). Headers are automatically printed at the top of each page of a report. They are also printed when the repetitive category lines exceed the number of lines per page allowed for the report, causing a new page to be printed.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'I'	Indicates the first line of a category printed several times (repetitive category). This value causes the generation of a subscript which controls the number of repetitions. This number may be fixed or variable.
			For a fixed number:
			.enter a number in the TOTALING LINE INDICATOR field
			For a variable number:
			.enter a three-character code in the TOTALING LINE INDICATOR field. (The code was defined on the Work Areas (-W) screen for use as the subscript field. Procedural code is used to move in the values.) OR .use the standard PACBASE index (Jddrcc), generated for the category: Note: ddr = REPORT CODE, cc = CATEGORY OF REPORT (repetitive) See SOURCE FIELD - LAST PART on the Report Call of Elements (-CE) screen, with value '*cc'.
		'Z'	This value applies to repetitive categories only. This indicates the first line of an end-of-page category (footer). Footers are automatically printed when the repetitive category lines exceed the number of lines per page allowed for that report.
			To identify detail lines with fields to accumulate:
		'*'	This indicates a detail line containing fields whose values are to be accumulated for totaling. The lines will be printed in the report. Note: The data elements to total are identified on the Report Call of Elements screen by entering 'T' in OPERATION ON SOURCE FIELD. All elements are conditioned by report category. (See Subchapter "CALL OF DATA ELEMENTS (-CE)".)
			A category containing a detail line: . can contain only one detail line, . cannot contain a total line, . cannot be iterative, . can include other ordinary lines.
			The logic for data elements to be totaled is generated only if the conditions specified for the '*' line category are met.
		'T'	Same as '*', but the category containing this line is not to be printed.
			Note: For information concerning other lines that may or may not be included with lines of this type, see CATEGORY OF REPORT.
			One program may use several reports. There can only be 12 '*' and 'T' type lines (combined) per program.
			To identify lines displaying accumulated totals:

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'0'	Indicates a line for Grand Totals. Note: Grand Totals may only be requested if there is at least one Total at a control break level. At least one control break has to be specified for a file on the -CD screen.
		'1 to 9'	Indicates a line for totaling at the control break level corresponding to this value.
			A category containing a total line: . may contain several of them, . cannot contain a detail line, . cannot be iterative, . can include other ordinary lines.
			See CATEGORY OF REPORT for information on other lines that may or may not be included in a category with totaling-type lines.
			NOTE: A detail line may be defined in a different report. For example, a summary report based on accumulations from other reports may be needed. This can be done using the following technique: The STRUCTURE NUMBER assigned to the detail line of the other report is not used on the summary report's Call of Elements screen, and on its Description (-D) screen, the TYPE OF LINE value is entered and the TOTALING LINE INDICATOR will be comprised of the LAST CHARACTER OF REPORT CODE of the report containing the detail line, followed by its STRUCTURE NUMBER. Only the totaled data elements will be printed, at the designated control break level.
6	3		TOTALLING LINE INDICATOR
			On a line that has fields being totaled (TYPE OF LINE values '0' to '9'), which has a detail line described in a different report, enter the following:
			.first character: LAST CHARACTER OF REPORT CODE of the report containing the description,
			.2nd and 3rd characters: STRUCTURE NUMBER.
			On the first line of a repetitive category (TYPE OF LINE = 'I'), this value causes the generation of a subscript which controls the number of repetitions. This number may be fixed or variable.
			For a fixed number:
			.enter an absolute number value.
			For a variable number:

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
		'blank'	.enter the three character code defined on the Work Areas (-W) screen for use as the subscript field. (The values are determined via Procedural Code.) OR .use the standard PACBASE index (Jddrcc), generated for the category.
7	2		STRUCTURE OF THE LINE FOR PRINTING
			PURE NUMERIC FIELD
			It is the variable part of the line, called 'structure'. Enter here the number of the chosen structure (from '01' to '98') which must have been defined on the 'Call of elements' screen (-CE).
8	2		CONSTANT PART NUMBER
			FALSE NUMERIC FIELD
			The constant part is defined on the Report Layout (-L) screen. Enter here its corresponding number, also defined on the Layout.
			SKIP
9	2		LINE SKIP
			PURE NUMERIC FIELD
			This line skip is taken into account at the report generation.
			(default option: 01).
			Enter the number of lines to skip, or an absolute line number.
		'0'	Overprinting
10	1		LINE SKIP TYPE
		'blank'	Skips the number of lines indicated in the field. (Default option).
		' *'	Absolute line number, when indicated on the first line of a category (except for the heading category).
			Ex: if you indicated *70, a category is printed after line '70'.
11	4		FUNCTION SUB-FUNCTION PRIOR TO PRINT
			Enter the code of the function (and sub-function) to be performed before the processing of the STRUCTURE NUMBER indicated on this line, and before the WRITE.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			Note: The same STRUCTURE NUMBER may be used in several categories. In this case, the PERFORM will take place each time through the processing loop for that structure. It is not necessary to enter the (sub)function code on the first category that uses that structure. A function must not be mentioned more than once for the same structure.
			In cases where several functions are to be performed with the same structure, the execution sequence may be problematic.
			For lines without a STRUCTURE NUMBER specified, the function will be performed once only, preceding the completion of processing of the structures, (F8199), and just prior to the WRITE.
			This function is performed according to the positioning of the associated structure and thus to the type or condition of the category in which the structure is called.
12	13		COMMENTS
			The comment entered on the screen top refers to the whole report. Comments entered on the screen body normally refer to the individual lines.
13	35		CONDITIONS OF REPORT EXECUTION
			On the screen top - (the ″E-line″):
			Enter conditions relevant for report execution.
			On the screen body:
			Enter conditions concerning the execution of the Category of Report.
			Format of entry:
			Use the COBOL format to enter conditions but do not enter 'IF', nor GO TO, and do not enter any period.

Direct print / application spooling routines

GENERAL INFORMATION

For the purpose of this discussion, the term 'direct print' applies to those automatic spooling programs that are transparent to the user. Reference to 'application spooling routines' are those where the user specifies the spooling, for instance, in order to sort reports after they are produced.

The user identifies which type of report it is via the USAGE OF DATA STRUCTURE value for the report data structure on the Call of Data Structures (-CD) screen of the program.

DIRECT PRINT REPORTS: USAGE OF DATA STRUCTURE = 'I'

The generated WRITE statements take the line SKIP values entered on the Report Description (-D) screen into account.

Some hardware permits the output of files using the direct print option (usage = 'I') to be sent to devices other than printers. The first position of each record is therefore reserved for the 'skip' character, and automatically translated by the compiler in WRITE commands. A utility program then transfers it to the printer.

APPLICATION SPOOLING ROUTINES: USAGE OF DATA STRUCTURE = 'J'

Spooling consists of storing the print file lines on an intermediate tape or disk file. The stored file is retrieved by a program executing a print job, with the spooled file as input.

For certain operating systems, the spooling program is written according to specific criteria and may use external parameters. Each record image of the stored file (on an intermediate tape or disk) contains information that will not be printed: information used to control line skips, sort criteria, and the output line.

WRITE commands in a spooled report do not check for line SKIP field values. The PACBASE data element 'LSKP' acts like a pointer to this value. 'LIGNE' is a group field into which the sorted output is moved.

These fields are included by using STRUCTURE NUMBER = '00', in which sort criteria, like the REPORT CODE, may be entered (major-to-minor sequence).

USE OF 'LSKP' DATA ELEMENT:

If the 'LSKP' element is not used, a 'WRITE' statement is generated.

Entering 'LSKP' in the '00' STRUCTURE generates a 'WRITE AFTER LSKP' statement.

If 'LSKP' is the first element of the 00 STRUCTURE, the first character of the file is automatically filled with the corresponding ASA skip value, if this operating system specification is available.

If the 'LSKP' is not entered as the first element, it is necessary to enter the skip value in this field.

Data elements of a '00' structure are referenced in relation to the beginning of the record. They are listed on the Report Call of Elements (-CE) screen exactly as the data Elements of all the other structures are.

Reports that are spooled are described exactly as reports printed directly, with respect to the Layout, Description and Call of Elements, except for the inclusion of a '00' structure as described above.

Spooling is transparent at the program level. Therefore the user may change the USAGE OF DATA STRUCTURE value to send the output directly to the printer. This may be convenient for testing purposes. The '00' structure will not be used with usage = 'I'. At implementation, the only modification to make is to change the usage back to 'J'.

On-line access commands

```
LIST OF REPORTS
CHOICE             SCREEN                                UPD
------             ------                                ---
LCRaaa             List of Reports by code (starting      NO
                   with Report 'aaa').
LNRaaa             List of Reports by name (starting      NO
                   with Report 'aaa').
LTRbRaaa           List of Reports by type 'b'(starting   NO
                   with Report 'aaa').
DESCRIPTION OF REPORT 'aaa'
CHOICE             SCREEN                                UPD
------             ------                                ---
Raaa               Definition of Report 'aaa'.           YES
RaaaGCbbb          Comments of Report 'aaa'.             YES
                   (starting with line 'bbb').
RaaaCRbbbbbb       Occurrences linked to Report 'aaa'
                   through User Relationship 'bbbbbb'.
RaaaATbbbbbb       Text assigned to Report 'aaa'          NO
                   (starting with text 'bbbbbb').
RaaaX              X-references of Report 'aaa'.          NO
RaaaXVbbbbbb       X-references of Report 'aaa' to        NO
                   Documents (starting with Document
                   'bbbbbb').
RaaaXPbbbbbb       X-references of Report 'aaa' to        NO
                   programs ( starting with program
                   'bbbbbb').
```

```
RaaaXQbbbbbb       List of occurrences linked to Report   NO
                   'aaa' through User Relationship
                   'bbbbbb'.
RaaaLbbCccc        Layout of Report 'aaa'                 YES
                   (starting with line 'bb', column 'ccc').
RaaaDbbccc         Description of Report 'aaa'            YES
                   (starting with category 'bb',
                   line 'ccc').
RaaaCEbbccc        Call of Data Elements in Report 'aaa' YES
                   (starting with Structure 'bb',
                   position 'ccc').
```

NOTE: After the first choice of type 'Raaa', 'Raaa' can be replaced with '-'.

All notations between parentheses are optional.

```
                  PURCHASING MANAGEMENT SYSTEM        SG000008.LILI.CIV.1583
LIST OF REPORTS BY CODE

CODE  NAME AND COMPLEMENT                  T TYPE          LGT  TOTAL  LIBR
EO 1  VENDOR ACTIVITY                      E REPORT        132  11 07  0059
EO 2  VENDOR LIST                          E REPORT        132  11 07  0059
XE R  CONTROL REPORT                       E REPORT        132  11 07  *CEN
XO C  Comment of data element              E REPORT        132  11 07  *CEN
XO D  Dialogue                             E REPORT        132  11 07  *CEN
XO E  Complement of the Dialogue           E REPORT        132  11 07  *CEN
XO K  List of data elements (c1)           E REPORT        132  11 07  *CEN
XO L  List of data elements (c2)           E REPORT        132  11 07  *CEN
XO M  Macro structures called              E REPORT        132  11 07  *CEN
XO P  Structured code                      E REPORT        132  11 07  *CEN
XO S  Screen definition                    E REPORT        132  11 07  *CEN
XO 2  Segments used                        E REPORT        132  11 07  *CEN
XO 7  Lines '7'                            E REPORT        132  11 07  *CEN
XY A  TRANSACTION REPORT                   E REPORT        132  11 07  *CEN
XY B  PRODUCTION REPORT                    E REPORT        132  11 07  *CEN
XY C  TRANSACTION SELECTION CARDs          E REPORT        132  11 07  *CEN


*** END ***
O: C1 CH: LCR
```

```
                  PURCHASING MANAGEMENT SYSTEM         SG000008.LILI.CIV.1583
REPORT    GENERAL DOCUMENTATION ED1    FOLLOW-UP AND STATISTICS

A LIN : T DESCRIPTION                                                    LIB
  100 :   THIS EDITION SHOULD BE EXECUTED EVERY NIGHT.
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :

O: C1 CH: -GC
```

Generation and/or printing

With COMMAND FOR PRINT REQUEST = 'DCR':

The ENTITY CODE is optional. When selecting a report or Reports, enter the prefix of the ENTITY CODE, and the LAST CHARACTER OF REPORT CODE(s) in the continuation area (beginning in column 31 in batch mode).

Whatever the library selection code happens to be, the output option for a report can be only '1' or '2' (C1, U1,..., C2, U2...).

The '1' option generates the printing of:

.the definition line of reports:

- associated keywords and general documentation lines,
- cross-references to programs,

.description lines of reports:

- report layouts,
- report descriptions (general characteristics and list of categories),
- report call of elements.

The '2' option provides the same listings as above, but adds a listing of the data structure assigned text and the report assigned text.

With COMMAND FOR PRINT REQUEST = 'LCR':

A list of reports in report code sequence is provided.

With COMMAND FOR PRINT REQUEST = 'LKR':

A list of reports in keyword sequence is provided. The user may restrict the listing by specifying the keyword type:

Explicit only = 'M'; Implicit only = 'L', entered in column 30 (batch mode). The keyword to search on may be specified, by entering it in the continuation area (column 31 in batch mode).

With COMMAND FOR PRINT REQUEST = 'LTR':

A list of reports in report type sequence is provided.

Chapter 5. Error messages

Introduction

The System manages error messages that will be used to inform you of input errors detected by application programs.

Error messages can be created as needed, or generated upon request, to update the sequential error message file. This file will be used to create application error message files. They can be indexed files or databases, depending on the hardware in use.

The generation is performed by the GPRT procedure, using the GED generation/print command. It is possible, in this command, to specify the generation language (EN or FR) of the error messages, which is by default the language assigned to you.

It generates the error messages for the Screens specified in the GED command inside the PAC7GL file. Error messages of other Screens found in the PAC7LG file are copied in the PAC7GL file and not modified.

GENERAL INFORMATION

There are two different types of error messages for batch: those that are generated automatically, and those that are user-defined.

Standard error messages will appear for errors detected in processing of transactions according to the DATA ELEMENT PRESENCE and CONTENTS specifications entered on the Segment Call of Elements (-CE) screen. These messages may be modified by the user, and/or supplemented with text.

User-defined error messages may be used with other validations. They are defined in a program using Procedural Code lines, and then attached to the transaction data structure to which they apply. Any program with appropriate messages may be associated with the transaction, however since the maximum number of programs that can be associated is two, it is advisable (perhaps) to design a program or two whose only function is to contain these messages.

The Error Message File must be generated and the sequential file loaded into the program. Backout issues may also need to be addressed.

AUTOMATIC ERROR MESSAGES

An error message record is automatically generated for each control coded in the Segment description lines. It consists of two parts which follow one after the other:

- A message corresponding to the error type and therefore to the type of control being performed. These standard messages are stored in a VA Pac file, but they can be modified on-site by the Database Administrator).

  Example:

  'INVALID ABSENCE OF THE DATA ELEMENT'
- The data element clear name in the dictionary.

  Example:

  'ORDER NUMBER'

  Concatenating the two gives the following result:

  'INVALID ABSENCE OF THE DATA ELEMENT ORDER NUMBER'

REPLACEMENT OF AUTOMATIC MESSAGES

Automatic messages can be replaced by specific messages such as:

'THE ORDER NUMBER IS REQUIRED'

These messages are indicated on 'S' type lines assigned to data element call lines in the Segments (SddssCEnnnGE, where nnn is the Data element call line number).

EXPLICIT ERROR MESSAGES

Controls coded on Data element calls in Data Structures are the only ones that cause error messages to be automatically generated. For all types of errors detected by other controls, automatic or otherwise, error messages must be defined explicitly with the 'E' operator on structured language description lines (-P).

(See Subchapter ″Procedural Code Screen″ in the Chapter ″Modifying the Procedure Division″ of the Manual Structured Code.)

DOCUMENTATION MESSAGES

Besides error messages, it is possible to generate documentation messages of the same format. These documentation messages consist of the following:

- Description lines of the Data elements called in the Segments.
- Text lines called in the -GE screen assigned to the Data Element call lines.
- lines of 'D' type assigned to the Data element call lines (SddssCEnnnGE).

Replacing automatic messages and defining documentation labels are not possible with the generation of VA Pac Version 6 type error messages.

ERROR MESSAGE EDIT EXAMPLE

```
ERR G ! LIST OF ERROR MESSAGES
------!-------------------------------------------------
      !
      ! NUMBER OF DELIVERIES
      ! --------------------
      ! Text or comment lines associated with the data
      ! element.
      !
      ! Data element description lines.
      ! 0     :  Before creating the 1st delivery.
      ! 1 to 9:  Each time a delivery is created, its value
      !          is incremented by 1.
      !
 2  E !   INVALID ABSENCE OF THE DATA ELEMENT NUMBER OF
      !   DELIVERIES
      !
 4  E !   NON-NUMERIC CLASS DATA ELEMENT NUMBER OF
      !   DELIVERIES
      !   Text or comment lines associated with type 4 Data
      !   element errors
      !
 5  E !   INVALID VALUE FOR DATA ELEMENT NUMBER OF
      !   DELIVERIES
```

Coding of error messages

CODING OF ERROR MESSAGES

Automatic error messages are built in two parts. The first part is a description of the type of error. The second part is the clear name of the erroneous data element. The first part may be modified on-site by the Data Administrator. Additionally, the error message can be customized to suit the specific data element it concerns by entering the message on the Data Structure Definition line, using the LINE NUMBER value to attach the message to the appropriate element.

The TYPE OF LINE value determines whether the contents of the COMMENT field override a message or supplement it.

To override a message, enter 'S' for TYPE OF LINE, and code the COMMENT field as follows:

Column 1: ERROR TYPE (2, 3, 4 or 5)

Column 2: blank

Column 3: ERROR GRAVITY (E, C or W)

Column 4: blank

Column 5: enter the message beginning here.

Example: To replace the automatically generated message for an erroneous value of the data element called on line 120:

```
------------------------------------------------------------
!      LIN : T DESCRIPTION                                 !
!      010 : S 5 E THIS VENDOR IS SUSPENDED                !
!                                                          !
!O: C1 CH: -ce120ge                                        !
------------------------------------------------------------
```

SUPPLEMENTING AUTOMATIC ERROR MESSAGES

To supplement the error report with extra documentation, enter 'D' for the TYPE OF LINE, and code the COMMENT field as follows:

```
Column 1: 0 = place this information before Data
              Element Description (-D) lines,
          1 = place this information after Data
              Element Description (-D) lines,
          2 to 5 = place the documentation after
                   the corresponding error message
Column 2: blank
Column 3: blank = a documentary message
          T     = the call of a text
Column 4: blank
Column 5: Begin the documentary message or
          Enter the text & paragraph code being called.
          Two asterisks (**) for the paragraph code is
          a permitted value, it will call all the para-
          graphs of the text.
```

EXAMPLE: To precede all error messages for the data element called on line 230 with a text:

```
------------------------------------------------------------
!      LIN : T DESCRIPTION                                 !
!      010 : D 0 T TEXTCDPP                                !
!                                                          !
!O: C1 CH: -ce230ge                                        !
------------------------------------------------------------
```

PROVIDING ADDITIONAL ERROR MESSAGES

The only error messages that are automatically generated are for errors detected according to the data element validation specifications entered on the Segment Call of Elements (-CE) screen. All other types of messages must be explicitly defined.

Since only two programs containing error messages can be associated with the transaction data structure concerned, it may be convenient to define separate programs just to contain these messages.

DEFINING USER ERROR MESSAGES

User error messages are defined in Structured Code on the Procedural Code (-P) screen, using the 'E' OPERATOR. The OPERAND field is coded as described below.

```
Column 1: A User Error Code character.
          Note: Avoid values 0 to 5 inclusive,
                as they have pre-defined meanings.
          Recommendation: Use '6', since this is
          the value used in standard macros.

Column 2 to 4: Enter a unique identifying number
               for this message.

Column 5: Error gravity.

Column 6: Begin your error message
```

In the CONDITION field, the message may be continued.

Example:

```
-------------------------------------------------------------
!LIN OPE OPERANDS                        LVTY CONDITION    !
!    N   USER ERRORS                     10BL              !
! 10 E   6001 ZIPCODE DOES NOT CORRESPOND    TO STATE      !
! 20 E   6002 FIRST CLASS SMOKING SECTION    IS FULL       !
!                                                          !
!O: C1 CH: Perrpg1 P00ut                                   !
-------------------------------------------------------------
```

ASSOCIATING THE USER ERROR MESSAGE WITH THE ERROR

This is normally accomplished using the User Error Table (UT-UPR(n)), which is generated with the error variable, 'ERUT'. Error messages are stored positionally according to the error number (example 001, then 002). In order to specify which error message is desired, use Procedural Code: Move '1' into UT-UPR(n), where n = the error number of the message.

ASSOCIATING ERROR MESSAGE PROGRAM(S) WITH THE TRANSACTION

On the Data Structure Definition screen of the transaction data structure, enter the error program's PROGRAM CODE in the COMPLEMENT field as follows:

Column 1 : blank

Column 2 : E

Column 3 to 8 : first program with error messages

Column 9 to 14 : second program with error messages.

GENERATING THE ERROR MESSAGE FILE

In order to include error messages in a program, the error message file must be generated. This is accomplished by using the 'GED' COMMAND FOR PRINT REQUEST, with the data structure being the transaction data structure code.

Using the C2 print option, a report similar to the one below will be produced.

```
-------------------------------------------------------
!ERR G ! ERROR MESSAGE LIST                            !
!------!-----------------------------------------------!
!      !                                               !
!      ! NUMBER OF DELIVERIES                          !
!      ! --------------------                          !
!      ! Text or general documentation lines associated    !
!      ! with the data element from SddssCEnnnG, TYPE OF   !
!      ! LINE = 'D' and COMMENT first column = '0'.        !
!      !                                               !
!      ! Data element description lines: EeeeeeeD.         !
!      ! 0    .before first delivery                    !
!      ! 1  9 .with each delivery, the value is incremented!
!      !       by one.                                 !
! 2  E ! INVALID ABSENCE OF DATA ELEM. NUMBER OF DELIVERIES!
! 4  E ! NON NUMERIC CLASS DATA ELEM. NUMBER OF DELIVERIES !
!      ! Text or general documentation lines associated    !
!      ! with error type 4: SddssCEnnnG, TYPE OF LINE = 'D'!
!      ! and COMMENT first column = '4'.                 !
! 5  E ! ERRONEOUS VALUE FOR DATA ELE. NUMBER OF DELIVERIES!
-------------------------------------------------------
```

NOTE: Loading of the sequentila error file and addressing backout issues may be accomplished by calling in Parameterized Macro Structures.

Description of error message file

The System generates an error message file. The records generated for this file are described on the following pages.

Examples of error message file records:

```
------------------------------------------------------------
! AP6AMB00  0035000EERRONEOUS VALUE FOR DATA ELEMENT DELAY !
!                                                          !
! GCCHJIE0100054000ENON-NUMERIC CLASS DATA ELEMENT ACTION  !
!                                                          !
! LU1IDO000116  002 009                                    !
------------------------------------------------------------
```

Decoding the first example:

```
LIBRARY CODE : AP6
ENTITY TYPE  : A (Segment)
ENTITY CODE  : MB00
ERROR NUMBER : 003 (rank - location on the list of elements
                         of the segment)
ERROR TYPE   : 5 (erroneous value)
LINE NUMBER  : 000
ERROR GRAVITY: E
ERROR MESSAGE: ERRONEOUS VALUE ...........
```

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
1	3		LIBRARY CODE
		lll	This code identifies a Library. The Library code is assigned at the time a Library is created and cannot be modified.
			Special characters are not allowed in a Library code but any alphabetic or numeric character can be used.
		***	This value is forbidden to define a library. It must be used only to select all the Libraries when viewing the Database.
2	1		ENTITY TYPE
			Used to specify the type of entity.
		'A'	For Data Structures or Segments (BSD error messages).
		'H'	For Screens (OLSD error messages).
		'I'	Record reserved for internal use by the OLSD function. It is used by the Help function to indicate the position of a field on a Screen, using a line / column formula.
3	6		ENTITY CODE
4	3		ERROR NUMBER
			For automatically generated error messages:
			It is the data element position (or sequence number) in the segment or screen.
			For user-defined error messages:
			This is the unique error code entered on a Procedural Code (-P) screen with OPERATOR = 'E'. This value is entered in columns 2 to 5 of the OPERAND field.

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
5	1		ERROR TYPE
			The following values are used by the system to flag erroneous conditions as specified in the validation fields on the Segment or Screen Call of Elements (-CE) screens for data elements:
		'2'	. Invalid absence.
		'3'	. Invalid presence.
		'4'	. Erroneous class.
		'5'	. Erroneous value.
			Other error types can be defined by the user, for non-standard validations. They must be inserted via procedural Code (-P) in validation and update programs.
			Documentary messages assigned to data elements are identified by the following values:
		'0'	Documentation placed prior to Data Element Description information.
		'1'	Documentation placed after Data Element Description information.
6	3		LINE NUMBER
			This number is managed by the system.
		'000'	Error messages
		'001-999'	Documentary messages
			NOTE: For an ENTITY TYPE 'I' record, this number is managed by the system and contains the LINE NUMBER of the erroneous field on the Screen.
7	1		ERROR GRAVITY
			The value of this zone may be controlled by the user in order to restrict transaction rejections. For example: 'W' = Warning. Transaction accepted. 'C' = Caution, error. The data element is corrected, or its update is refused (the rest of the transaction is accepted. 'E' = Error. This error is not corrected. The transaction is rejected.
			Standard PACBASE does not check the value of this field, and rejects all erroneous transactions.
8	30		ERROR MESSAGE FIRST PART
			For automatic error messages, this part of the message remains constant and is used to indicate the type of error:
			2: INVALID ABSENCE OF DATA ELEMENT,

NUM	LEN	CLASS VALUE	DESCRIPTION OF FIELDS AND FILLING MODE
			3: INVALID PRESENCE OF DATA ELEMENT,
			4: CLASS OF DATA ELEMENT NOT NUMERIC/ ALPHABETIC,
			5: ERRONEOUS VALUE FOR DATA ELEMENT.
			For explicit error messages, this is the first part of the error message as entered in the OPERAND field on the Procedural Code (-P) screen.
			For ENTITY TYPE = 'I' records, the value in this field identifies the column of the erroneous field.
9	36		ERROR MESSAGE 2ND PART
			For automatic error messages, this is the clear name of the erroneous data element as defined on the Data Element Definition screen, or on the Segment Call of Elements (-CE) screen.
			For explicit error messages, this is the part of the message entered in the CONDITION field of the Procedural Code (-P) screen.

Generation and/or printing

GED: Generate the error messages defined for a data structure and for each segment. (A language code (EN or FR) can be entered to generate the error messages in a language other than the language assigned to the user).

C1: Error messages defined for the data structure and for each segment.

C2: Error messages generated through option 1 plus documentary help messsages.

LED: List the error messages defined for the data structure and for each segment.

This command is accessible in option 1 only.

This list only includes messages that have already been generated.

NOTE: If a segment suffix is entered on the continuation line of a GED or LED command, error messages are generated/ printed for this segment only.

Chapter 6. Example of generated program

Introduction

The purpose of this chapter is to present a program designed in the System, as it is generated in COBOL.

The objective of this program is to demonstrate a wide variety of options, not a model for ″good programming″.

In this chapter, the user will find the following:

- coding of the data names,
- different types of data structure descriptions,
- a complete glossary of variables, counters and indexes,
- the description of all the standard functions with their generation condition.

Highlights of various screen images used in the generated example are entered below:

Transaction file Definition screen:

```
-------------------------------------------------------------
!DATA STRUCTURE DEFINITION       MV                         !
!NAME.............: TRANSACTION FILE                        !
!COMPLEMENT.......:                                        !
!TYPE.............: Z DATA STRUCTURE                        !
!                                                           !
!O: C1  CH: d mv                                           !
-------------------------------------------------------------
```

Transaction Segment (common part segment) Definition screen:

```
-------------------------------------------------------------
!SEGMENT DEFINITION........: MV00                          !
!NAME.....................: TRANSACTION SEGMENT            !
!OCCUR. OF SEGMENT IN TABLE:                               !
!EST. NUMBER OF INSTANCES..:                               !
!CODE OF RECORD TYPE ELEM..: NUCAR                         !
!CODE OF ACTION CODE ELEM..: CODMV                         !
!VALUES OF TRANSACTION CODE: CR: 'C' MO: 'M' DE: 'S'       !
!                            M4: 'D' M5: 'E' M6: 'F'       !
!                                                          !
!O: C1  CH: s mv00                                         !
-------------------------------------------------------------
```

Transaction Segment (common part) Call of Elements screen:

```
-------------------------------------------------------------
!SEGMENT CALL OF ELEMENTS MV00 TRANSACTION SEGMENT          !
!ELEM.  .... U OCC GR K CMD456 CONT VALUE/SFC UPD/TRGET     !
!ENPR              1                                        !
!EPR            10                                         !
!GRPR              1                                        !
!GPR             2                                         !
!ERUT              1                                        !
!UPR            10                                         !
!NOCL              3             M           LV00NOCL      !
!NOCL11              A OOOOOO                               !
!NOCL12              B OOOOOO                               !
!NOCL2               C OOOOOO 9                             !
!NUORD               D OOOOOO 9                             !
!                               N< '1'                      !
!                              EN> '8'                      !
!                              O = '9'                      !
!CODMV                E                                     !
!NUCAR                F                                     !
!                                                           !
!O: C1 CH: s mv00 ce                                        !
-------------------------------------------------------------
```

Transaction Segment (specific part) Definition screen:

```
-------------------------------------------------------------
!SEGMENT DEFINITION........: MV01                           !
!NAME.....................: TRANSACTION SEGMENT            !
!OCCUR. OF SEGMENT IN TABLE:                                !
!EST. NUMBER OF INSTANCES..:                                !
!VALUE OF RECORD TYPE ELEM.: 'A'                            !
!CODE OF ACTION CODE ELEM..:                                !
!PRESENCE.................: CR: O   MO: I   DE: I          !
!                           M4:     M5:     M6:            !
!                                                           !
!O: C1  CH: s mv01                                          !
-------------------------------------------------------------
```

Transaction Segment (specific part) Call of Elements screen:

```
-------------------------------------------------------------
!SEGMENT CALL OF ELEMENTS MV01 TRANSACTION SEGMENT          !
!ELEM.  INT.FORM. U.... CMD456 CONT VALUE/SFC UPD/TRGET     !
!NOMCL                  O      A                            !
!ADRES                  O                                   !
!NUDEP                  O         T           TD01NUDEP    !
!FILLER X(6)      D                                         !
!                                                           !
!                                                           !
!O: C1 CH: s mv01 ce                                        !
-------------------------------------------------------------
```

Transaction Segment (specific part) Definition screen:

```
-------------------------------------------------------------
!SEGMENT DEFINITION........: MV02                           !
!NAME.....................: TRANSACTION SEGMENT            !
!OCCUR. OF SEGMENT IN TABLE:                                !
!EST. NUMBER OF INSTANCES..:                                !
```

```
!VALUE OF RECORD TYPE ELEM.: 'B'                              !
!CODE OF ACTION CODE ELEM..:                                  !
!PRESENCE..................: CR: O   MO:     DE: I           !
!                            M4: O   M5: O   M6: O           !
!                                                            !
!O: C1  CH: s mv02                                           !
--------------------------------------------------------------
```

Transaction Segment (specific part) Call of Elements screen:

```
--------------------------------------------------------------
!SEGMENT CALL OF ELEMENTS MV02 TRANSACTION SEGMENT            !
!ELEM.  INT.FORM. U.... CMD456 CONT VALUE/SFC UPD/TRGET       !
!MREEL9                                                      !
!MREEL9                                          R*          !
!DALI                                                        !
!FILLER X(62)     D                                          !
!                                                            !
!                                                            !
!O: C1 CH: s mv02 ce                                         !
--------------------------------------------------------------

--------------------------------------------------------------
!REPORT DEFINITION.....: ED1                                 !
!NAME..................: TEST FOR BATCH MANUAL               !
!COMMENTS..............:                                     !
!NATURE................: E REPORT                           !
!PRINTER TYPE..........: P                                   !
!LINE LENGTH...........: 045                                 !
!FORMAT FOR TOTALS     : INTEGER........: 11                  !
!                        DECIMAL PLACES.: 07                 !
!                                                            !
!O: C1  CH: r ed1                                            !
--------------------------------------------------------------
```

Report Layout for Report 1:

```
--------------------------------------------------------------
!REPORT LAYOUT :   ED1 TEST FOR BATCH MANUAL   LENGTH= 045 !
!LN CP S C          1    1    2    2    3    3    4    4   !
!         1...5....0....5....0....5....0....5....0....5...!
!01  1 * 1                 UPDATE REPORT        XXXXXXX    !
!10      0                                                   !
!20  2   1 NUMBER OF VALID TRANSACTIONS       :  495       !
!30  3   2 NUMBER OF INVALID TRANSACTIONS     :   55       !
!40  4 2 0 NUMBER OF TRANSACTIONS             :  550       !
!50  5   0 PERCENTAGE OF INVALID TRANSACTIONS :   10,00    !
!60  6 2 0 NUMBER OF FILE RECORDS             :            !
!70  7   0                                                   !
!80  8 4 0                                CD  :  100       !
!90  9 3 0 ********************************************   !
!                                                            !
!O: C1 CH: r ed1 l                                           !
--------------------------------------------------------------
```

Report Call of Elements for Report 1:

```
------------------------------------------------------------
!REPORT CALL OF ELEMENTS ED1 TEST FOR BATCH MANUAL         !
!ST ELEM   L : STA C O W SOURCE    FLD CONDITION           !
!00 LSKP   0 :   1   M * LSKP                              !
!00 PAGE   0 :   3   M 5 LI001CP                           !
!00 NULIG  0 :   6                                         !
!00 LIGNE  0 :   9                                         !
!01 ACCEP  0 :  39   M   WA04ACCEP                         !
!02 REFUS  0 :  39   M   WA04REFUS                         !
!03 TOTAL  0 :  39   R   WA04ACCEP                         !
!03 TOTAL  1 :  39 * +   WA04REFUS                         !
!04 POURC  0 :  39   M * ZERO                              !
!04 POURC  1 :  39   R * 100           WA04-ACCEP > 0 OR...!
!04 POURC  2 :  39 * *   WA04REFUS                         !
!04 POURC  3 :  39 * /   (WA04ACCEP                        !
!04 POURC  4 :  39 * +   WA04REFUS)                        !
!05 NOFICH 0 :  32   M   WC02NOFICH*DD                     !
!05 CPTENR 0 :  38   M   WC03CPTENR*DD                     !
!06 ZLIB03 0 :   1                                         !
!                                                          !
!O: C1 CE: r ed1 ce                                        !
------------------------------------------------------------
```

Report Description for Report 1:

```
------------------------------------------------------------
!REPORT DESCRIPTION :    ED1 TEST FOR BATCH MANUAL         !
!LINE LENGTH: 045 LI PAGE: 60 CAT TBL INST: 0000 ..SECT. 00!
!COMMENTS...:             CONDITIONS FT = ALL '1'          !
!CA LIN T TLI ST CP SKP FUSF COMMENTS  CONDITIONS          !
!BC 100          01 01* 91BC                               !
!BC 110        1 02 02                                     !
!BC 120        2 03 02                                     !
!BC 130        3 04 02                                     !
!BC 140        4 05 02                                     !
!BC 150          06 02                                     !
!BC 160          07 01                                     !
!DD 100 I 012  5 08 01                                     !
!EE 100          09 01                                     !
!                                                          !
!O: C1 CH: r ed1 d                                         !
------------------------------------------------------------
```

Report Call of Elements for Report 3:

```
------------------------------------------------------------
!REPORT CALL OF ELEMENTS ED3 TEST FOR BATCH MANUAL         !
!ST ELEM   L : STA C O W SOURCE    FLD CONDITION           !
!01 DATEM  0 :  46   M * DAT8C                             !
!01 PAGE   0 :  76   M 5 ED003PC                           !
!02 NOCL   0 :  10   M 2 CL00NOCL                          !
!02 NOMCL  0 :  17   M 2 CL00NOMCL                         !
!03 FILLER 0 :  38   M * 'DELIVERY'                        !
!03 JED3FA 0 :  48   M * JED3FA                            !
!03 DATE   0 :  53   M 2 LV00DALI  *FA                     !
!03 QULI   0 :  75   M 2 LV00QULI  *FA                     !
!04 4      0 :  35   M 1 LI004     J05                     !
!04 NOCL11 0 :  56   M 2 CL00NOCL11    J05 < 4            !
```

```
!04 NOCL12 0 :  57   M 2 CL00NOCL12    J05 = 2 OR J05 = 3  !
!04 NOCL2  0 :  59   M 2 CL00NOCL2     J05 = 3             !
!04 QUCO   0 :  64   T 2 CD00QUCO                          !
!04 QTLI   0 :  76   T 2 LV00QTLI                          !
!04 SOLDE  0 :  88   R 2 CD00QUCO      J05 = 3             !
!04 SOLDE  2 :  88 * - 2 LV00QTLI                          !
!04 SOLDE  3 :  88   R   T304QUCO  J05 J05 NOT = 3         !
!04 SOLDE  4 :  88 * -   T304QTLI  J05                     !
!                                                          !
!O: C1 CH: r ed3 ce                                        !
------------------------------------------------------------
```

The main characteristics of the Program Call of Data Structures (-CD) screen used for the generated program are illustrated below:

```
------------------------------------------------------------
! DP DL ! OARFU ! B M ! U ! RE SE ! L ! SELECT. ! F E R L  !
! CD CD ! SSFIU ! 2 3 ! P ! DC    !   ! ABC     ! I     1  !
! CL CL ! SSFIU ! 2 3 ! P ! LC SE !   ! ABC     ! I     1  !
! DC CD ! SSFOU !     ! R ! CD    !   !         ! I     1  !
! ED ED ! SSFOU !     ! I !       !   ! 3       ! I     1  !
! EN MV ! SSFIU !     ! C !       !   !         ! I     1  !
! GL GR ! SSFIU !   2 ! C !       !   ! AB      ! I     1  !
! LC CL ! SSFOU !     ! R ! CL    !   !         ! I     1  !
! LI ED ! SSFOU !     ! J !       !   ! 1       ! I     1  !
! LV LV ! SSVIU ! 2 3 ! P ! VL    !   ! ABC     ! I     1  !
! MO MO ! VSFID !     ! T ! CD    !   !         ! I     1  !
!   STAT.FLD: MO00STATUS  ACC. KEY: MOIS    RECTYPEL:      !
! MV MV ! SSFTU ! 6 3 ! M !    VM ! 5 ! ABCDEF  ! I     1  !
! SE CL ! SSFOU !     ! S !    CL !   !         ! I     1  !
! TD TD ! SSFIU !     ! X !       !   ! *0102   ! I     1  !
! VL LV ! SSFOU !     ! D ! LV    !   !         ! I     1  !
! VM MV ! SSFOU !     ! E !    MV !   !         ! I W   1  !
! WA WG ! WSFOU !     ! D !       !   ! *04     ! I   2 2  !
! WR WR ! WSFOU !     ! D !       !   ! *02     ! I   2 2  !
!                                                          !
!O: C1 CH: p pjjps1 cd                                     !
------------------------------------------------------------
```

Identification division

The user may modify the IDENTIFICATION DIVISION of the generated program, via the Beginning Insertions (-B) screen.

(See the STRUCTURED CODE Reference Manual).

```
   IDENTIFICATION DIVISION.
   PROGRAM-ID.  PJJPS1.                                    PJJPS1
   AUTHOR.         VALIDATION/UPDATE.                      PJJPS1
   DATE-COMPILED.   07/03/04.                              PJJPS1
```

Environment division

The ENVIRONMENT DIVISION is adapted to the appropriate COBOL variant according to the TYPE OF COBOL TO GENERATE option.

(IBM MVS is used for the sample program).

In general:

-three types of file organization are accepted:

.sequential,

.indexed,

.'VSAM', for IBM MVS and DOS variants.

-three types of access methods are accepted:

.sequential access,

.dynamic (for VSAM organization only),

.direct access.

In the latter case, the generated NOMINAL KEY (or SYMBOLIC KEY) is always in the form 1-ddss-eeeeee where dd, ss, and eeeeee have been defined by the user on the Program Call of Data Structures (-CD). In fact, this key normally appears in a transaction file work area. If not, it is up to the user to define and control it.

NOTE

The user can modify this part of the program via the Beginning Insertions (-B) screen.

(See the STRUCTURED CODE Reference Manual.)

```
ENVIRONMENT DIVISION.                                          PJJPS1
CONFIGURATION SECTION.                                         PJJPS1
SOURCE-COMPUTER. IBM-370.                                      PJJPS1
OBJECT-COMPUTER. IBM-370.                                      PJJPS1
SPECIAL-NAMES.                                                 PJJPS1
     C01 IS LSKPP                                              PJJPS1
     CSP IS LSKP0.                                             PJJPS1
INPUT-OUTPUT SECTION.                                          PJJPS1
FILE-CONTROL.                                                  PJJPS1
     SELECT     CD-FILE       ASSIGN    UT-S-CD.              PJJPS1
     SELECT     CL-FILE       ASSIGN    UT-S-CL.              PJJPS1
```

```
SELECT     DC-FILE       ASSIGN    UT-S-DC.                PJJPS1
SELECT     ED-FILE       ASSIGN    UT-S-ED.                PJJPS1
SELECT     EN-FILE       ASSIGN    UT-S-EN.                PJJPS1
SELECT     GL-FILE       ASSIGN    UT-S-GL.                PJJPS1
SELECT     LC-FILE       ASSIGN    UT-S-LC.                PJJPS1
SELECT     LI-FILE       ASSIGN    UT-S-LI.                PJJPS1
SELECT     LV-FILE       ASSIGN    UT-S-LV.                PJJPS1
SELECT          MO-FILE     ASSIGN  TO  ENT01             PJJPS1
ORGANIZATION INDEXED                                       PJJPS1
FILE STATUS IS                   1-MO00-STATUS             PJJPS1
RECORD KEY  IS                     MO00-MOIS.              PJJPS1
SELECT     MV-FILE       ASSIGN    UT-S-MV.                PJJPS1
SELECT     SE-FILE       ASSIGN    UT-S-SE.                PJJPS1
SELECT     TD-FILE       ASSIGN    UT-S-TD.                PJJPS1
SELECT     VL-FILE       ASSIGN    UT-S-VL.                PJJPS1
SELECT     VM-FILE       ASSIGN    UT-S-VM.                PJJPS1
```

Data division : File section

The user cannot modify this part of the program in any way, except via the actual description of the data structures.

The FILE SECTION

All the data structures of a program with an ORGANIZATION S, I, or V, appear in the FILE SECTION. They are described according to their USAGE OF DATA STRUCTURE, their NUMBER OF CONTROL BREAKS and FILE TYPE.

Each record described appears in the form ddss where:

.dd = DATA STRUCTURE CODE IN THE PROGRAM

.ss = SEGMENT CODE.

Each data element appears in the form ddss-eeeeee with its format, or if defined as a group data element, is sub-defined in the Segment Call of Elements (-CE) screen.

Data structures without REDEFINES have only one COBOL record dd00, which includes the common and specific parts described in the PACBASE library.

Input data structures without control breaks or for which a description was requested, input-output data structures and direct output data structures (USAGE OF D.S. = 'D') are described fully in the FILE SECTION.

Input data structures with control breaks and for which a description was requested are only described partially. Only the common part appears in

detail. The other data elements are regrouped into the PACBASE group data element 'SUITE' in the format dd00-SUITE.

For output data structures linked to input data structures and for print data structures (USAGE OF D.S. = 'I' or 'J'), details of data elements do not appear here.

The description of an output transaction file (USAGE = 'E') depends on the value in the RESERVED ERROR CODES IN TRANS. FILE field on the Call of Data Structures (-CD) screen for the description of error tables.

If the descriptions of the reserved data elements are requested, the formats etc. will come from the specifications entered for them on the Segment Call of Elements screen. If not, the descriptions are generated as follows:

```
   dd00-ENPR             PICTURE X(n)
   dd00-GRPR             PICTURE X(m)
where:
   n = number of data elements in transaction d.s. + 1,
   m = number of record types in transaction d.s. + 1.
```

In any case, all other data elements in the data structure are grouped under:

```
   dd00-SUITE            PICTURE X(p)
where:
   p = length of the longest record in the transaction d.s.
```

Transaction data structures (USAGE OF D.S.= 'M' or 'N') that select descriptions of the reserved error codes, have two additional group levels within the dd00 level.

dd00V, for the description of reserved data elements,

dd00E, for the record image.

```
   DATA DIVISION.                                                PJJPS1
   FILE SECTION.                                                 PJJPS1
   FD                CD-FILE                                     PJJPS1
        BLOCK             00000 RECORDS                        PJJPS1
        DATA RECORD                                            PJJPS1
                     CD00                                      PJJPS1
             LABEL RECORD STANDARD.                            PJJPS1
   01                CD00.                                     PJJPS1
        10           CD00-NOCL.                                PJJPS1
        11           CD00-NOCL11 PICTURE  X.                   PJJPS1
        11           CD00-NOCL12 PICTURE  XX.                  PJJPS1
        11           CD00-NOCL2  PICTURE  XX.                  PJJPS1
        10           CD00-QUCO   PICTURE  S9(5)V99             PJJPS1
                     COMPUTATIONAL-3.                          PJJPS1
   FD                CL-FILE                                   PJJPS1
```

```
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                    CL00                                       PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                  CL00.                                      PJJPS1
     10             CL00-KEYCI.                                PJJPS1
     11             CL00-NOCL.                                 PJJPS1
     12             CL00-NOCL11 PICTURE  X.                    PJJPS1
     12             CL00-NOCL12 PICTURE  XX.                   PJJPS1
     12             CL00-NOCL2  PICTURE  XX.                   PJJPS1
     10             CL00-NOMCL  PICTURE  X(20).                PJJPS1
     10             CL00-ADRES  PICTURE  X(43).                PJJPS1
     10             CL00-NUDEP  PICTURE  XXX.                  PJJPS1
     10             CL00-LIDEP  PICTURE  X(24).                PJJPS1
     10             CL00-NUREG  PICTURE  XXX.                  PJJPS1
     10             CL00-LIREG  PICTURE  X(24).                PJJPS1
FD                  DC-FILE                                    PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                    DC00                                       PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                  DC00.                                      PJJPS1
     10        FILLER          PICTURE  X(00166).              PJJPS1
FD                  ED-FILE                                    PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                    ED00                                       PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                  ED00.                                      PJJPS1
     10        FILLER          PICTURE  X(097).                PJJPS1
FD                  EN-FILE                                    PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                    EN00                                       PJJPS1
                    EN01                                       PJJPS1
                    EN02                                       PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                  EN00.                                      PJJPS1
   05               EN00-00.                                   PJJPS1
     10             EN00-NOCL.                                 PJJPS1
     11             EN00-NOCL11 PICTURE  X.                    PJJPS1
     11             EN00-NOCL12 PICTURE  XX.                   PJJPS1
     11             EN00-NOCL2  PICTURE  XX.                   PJJPS1
     10             EN00-NUORD  PICTURE  X.                    PJJPS1
     10             EN00-CODMV  PICTURE  X.                    PJJPS1
     10             EN00-NUCAR  PICTURE  X.                    PJJPS1
   05               EN00-SUITE.                                PJJPS1
     15        FILLER          PICTURE  X(00072).              PJJPS1
01                  EN01.                                      PJJPS1
     10        FILLER          PICTURE  X(00008).              PJJPS1
     10             EN01-NOMCL  PICTURE  X(20).                PJJPS1
     10             EN01-ADRES  PICTURE  X(43).                PJJPS1
     10             EN01-NUDEP  PICTURE  XXX.                  PJJPS1
     10             EN01-FILLER PICTURE  X(6).                 PJJPS1
01                  EN02.                                      PJJPS1
```

```
     10       FILLER         PICTURE  X(00008).                PJJPS1
     10            EN02-MREEL9 PICTURE  S9(5)V99               PJJPS1
                   COMPUTATIONAL-3.                            PJJPS1
     10            EN02-DALI   PICTURE  X(6).                  PJJPS1
     10       FILLER         PICTURE  X(00062).                PJJPS1
FD                 GL-FILE                                     PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                   GL00                                        PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                 GL00.                                       PJJPS1
     10            GL00-NOCL11 PICTURE  X.                     PJJPS1
     10            GL00-NOCL12 PICTURE  XX.                    PJJPS1
FD                 LC-FILE                                     PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                   LC00                                        PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                 LC00.                                       PJJPS1
     10       FILLER         PICTURE  X(00122).                PJJPS1
FD                 LI-FILE                                     PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                   LI00                                        PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                 LI00.                                       PJJPS1
     10       FILLER         PICTURE  X(056).                  PJJPS1
FD                 LV-FILE                                     PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     RECORDING  V                                              PJJPS1
     DATA RECORD                                               PJJPS1
                   LV00                                        PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                 LV00.                                       PJJPS1
     10            LV00-NOCL.                                  PJJPS1
     11            LV00-NOCL11 PICTURE  X.                     PJJPS1
     11            LV00-NOCL12 PICTURE  XX.                    PJJPS1
     11            LV00-NOCL2  PICTURE  XX.                    PJJPS1
     10            LV00-NBLIV  PICTURE  9.                     PJJPS1
     10            LV00-QTLI   PICTURE  S9(5)V99               PJJPS1
                   COMPUTATIONAL-3.                            PJJPS1
     10            LV00-GROUPE                                 PJJPS1
                   OCCURS       009     TIMES                  PJJPS1
                   DEPENDING  ON        LV00-NBLIV.            PJJPS1
     11            LV00-QULI   PICTURE  S9(5)V99               PJJPS1
                   COMPUTATIONAL-3.                            PJJPS1
     11            LV00-DALI   PICTURE  X(6).                  PJJPS1
FD                 MO-FILE                                     PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                   MO00                                        PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                 MO00.                                       PJJPS1
     10            MO00-ANNUL  PICTURE  X.                     PJJPS1
     10            MO00-MOIS   PICTURE  99.                    PJJPS1
```

```
     10            MO00-LMOIS  PICTURE  X(9).                  PJJPS1
     10            MO00-FILLER PICTURE  X(68).                 PJJPS1
SD                 MV-FILE                                     PJJPS1
     DATA RECORD                                               PJJPS1
                   MV00.                                       PJJPS1
01                 MV00.                                       PJJPS1
   05              MV00-00.                                    PJJPS1
     10            MV00-NOCL.                                  PJJPS1
     11            MV00-NOCL11 PICTURE  X.                     PJJPS1
     11            MV00-NOCL12 PICTURE  XX.                    PJJPS1
     11            MV00-NOCL2  PICTURE  XX.                    PJJPS1
     10            MV00-NUORD  PICTURE  X.                     PJJPS1
     10            MV00-CODMV  PICTURE  X.                     PJJPS1
     10            MV00-NUCAR  PICTURE  X.                     PJJPS1
   05              MV00-SUITE.                                 PJJPS1
     15       FILLER         PICTURE  X(00072).                PJJPS1
FD                 SE-FILE                                     PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                   SE00                                        PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                 SE00.                                       PJJPS1
     10       FILLER         PICTURE  X(00122).                PJJPS1
FD                 TD-FILE                                     PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                   TD00                                        PJJPS1
                   TD01                                        PJJPS1
                   TD02                                        PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                 TD00.                                       PJJPS1
   05              TD00-00.                                    PJJPS1
     10            TD00-NOTAB  PICTURE  X.                     PJJPS1
   05              TD00-SUITE.                                 PJJPS1
     15       FILLER         PICTURE  X(00030).                PJJPS1
01                 TD01.                                       PJJPS1
     10       FILLER         PICTURE  X(00001).                PJJPS1
     10            TD01-NUDEP  PICTURE  XXX.                   PJJPS1
     10            TD01-LIDEP  PICTURE  X(24).                 PJJPS1
     10            TD01-NUREG  PICTURE  XXX.                   PJJPS1
01                 TD02.                                       PJJPS1
     10       FILLER         PICTURE  X(00001).                PJJPS1
     10            TD02-NUREG  PICTURE  XXX.                   PJJPS1
     10            TD02-LIREG  PICTURE  X(24).                 PJJPS1
     10       FILLER         PICTURE  X(00003).                PJJPS1
FD                 VL-FILE                                     PJJPS1
     BLOCK              00000 RECORDS                          PJJPS1
     DATA RECORD                                               PJJPS1
                   VL00                                        PJJPS1
          LABEL RECORD STANDARD.                               PJJPS1
01                 VL00.                                       PJJPS1
     10            VL00-NOCL.                                  PJJPS1
     11            VL00-NOCL11 PICTURE  X.                     PJJPS1
     11            VL00-NOCL12 PICTURE  XX.                    PJJPS1
     11            VL00-NOCL2  PICTURE  XX.                    PJJPS1
```

```
     10            VL00-NBLIV  PICTURE  9.                      PJJPS1
     10            VL00-QTLI   PICTURE  S9(5)V99                PJJPS1
                   COMPUTATIONAL-3.                             PJJPS1
     10            VL00-GROUPE                                  PJJPS1
                   OCCURS       009     TIMES                   PJJPS1
                   DEPENDING  ON        VL00-NBLIV.             PJJPS1
     11            VL00-QULI   PICTURE  S9(5)V99                PJJPS1
                   COMPUTATIONAL-3.                             PJJPS1
     11            VL00-DALI   PICTURE  X(6).                   PJJPS1
FD                 VM-FILE                                      PJJPS1
     BLOCK              00000 RECORDS                           PJJPS1
     DATA RECORD                                                PJJPS1
                   VM00                                         PJJPS1
          LABEL RECORD STANDARD.                                PJJPS1
01                 VM00.                                        PJJPS1
     10            VM00-ENPR.                                   PJJPS1
     11            VM00-EPR    PICTURE  X                       PJJPS1
                   OCCURS       010     TIMES.                  PJJPS1
     10            VM00-GRPR.                                   PJJPS1
     11            VM00-GPR    PICTURE  X                       PJJPS1
                   OCCURS       002     TIMES.                  PJJPS1
     10            VM00-ERUT.                                   PJJPS1
     11            VM00-UPR    PICTURE  X                       PJJPS1
                   OCCURS       010     TIMES.                  PJJPS1
     10            VM00-SUITE.                                  PJJPS1
           15  FILLER        PICTURE  X(00080).                  PJJPS1
```

Beginning of Working Storage

Data structures with ORGANIZATION = 'W', or ORGANIZATION = 'L' or 'D' with an alphabetic CODE FOR COBOL PLACEMENT will be generated at the beginning of the WORKING-STORAGE SECTION.

For data structures with ORGANIZATION = 'W' or 'L', all description types are possible here. Furthermore, complementary levels may be inserted, either between data structures, or between segments in the same data structure, via the Work Areas (-W) screen.

WSS-BEGIN will be generated in every program, after these descriptions.

The constant 'BLANC' is only generated when Data Structure Usage is 'M' or 'N'.

The variable 'IK' is always generated.

The PACBASE-CONSTANTS level includes:

- the session number (NUGNA),
- the Library code (APPLI),

- the generation date (DATGN) (MM/DD/YY if user language = 'E', or DD/MM/YY otherwise),
- the Program code in library (PROGR),
- the user code (CODUTI),
- the generation time (TIMGN),
- the COBOL Program-ID (PROGE),
- the Database code (COBASE),
- the generation date with the century (DATGNC) (MM/DD/CCYY if user language = 'E', or DD/MM/CCYY otherwise),
- the generator version (RELEAS),
- the generator date (DATGE),
- the skeleton date (DATSQ).

These constants are always generated.

The 'DATCE' variable includes the CENTUR field (containing the value of the century), and a blank date area (DATOR) in which the user can store the processing date in a year-month-day format (DATOA-DATOM-DATOJ).

Note: in COBOL II and COBOL 85, if you use the date operator ADT or ADC, and if the year is less than '61', the CENTUR field is automatically set to '20'.

Fields to handle date rotations, slashes, century etc. are DAT6, DAT8, DAT8E, DAT6C and DAT8C.

The 'DATSEP' variable contains the separator used in the dates. You can modify its default value (/) by giving another value to the DATSEP Data Element in the -P lines.

```
WORKING-STORAGE SECTION.                                        PJJPS1
01              WA00.                                           PJJPS1
     02         WA04.                                           PJJPS1
     10         WA04-REFUS  PICTURE  S9(3)                      PJJPS1
                VALUE               ZERO                        PJJPS1
                COMPUTATIONAL-3.                                PJJPS1
     10         WA04-ACCEP  PICTURE  S9(3)                      PJJPS1
                VALUE               ZERO                        PJJPS1
                COMPUTATIONAL-3.                                PJJPS1
     10         WA04-INTER  PICTURE  S9(3)                      PJJPS1
                VALUE               ZERO                        PJJPS1
                COMPUTATIONAL-3.                                PJJPS1
     10         WA04-DECLA  PICTURE  S9(8)                      PJJPS1
                VALUE               ZERO                        PJJPS1
                COMPUTATIONAL.                                  PJJPS1
01              WR00.                                           PJJPS1
     10         WR00-DAT1.                                      PJJPS1
     11         WR00-DAT11  PICTURE  XX                         PJJPS1
                VALUE               SPACE.                      PJJPS1
```

```
           11            WR00-DAT12  PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           11            WR00-DAT13  PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           10            WR00-DAT113 PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           10            WR00-AMJ.                                    PJJPS1
           11            WR00-AMJA.                                   PJJPS1
           12            WR00-AMJA9  PICTURE  99                      PJJPS1
                         VALUE                ZERO.                   PJJPS1
           11            WR00-AMJM   PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           11            WR00-AMJJ   PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           10            WR00-BIS    PICTURE  9                       PJJPS1
                         VALUE                ZERO.                   PJJPS1
           10            WR00-DHORDI.                                 PJJPS1
           11            WR00-DORDI.                                  PJJPS1
           12            WR00-DORDIA PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           12            WR00-DORDIM PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           12            WR00-DORDIJ PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           11            WR00-HORDI  PICTURE  9(6)                    PJJPS1
                         VALUE                ZERO.                   PJJPS1
           10            WR00-DORDE.                                  PJJPS1
           11            WR00-DORDEJ PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           11            WR00-SLASH1 PICTURE  X                       PJJPS1
                         VALUE                SPACE.                  PJJPS1
           11            WR00-DORDEM PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
           11            WR00-SLASH2 PICTURE  X                       PJJPS1
                         VALUE                SPACE.                  PJJPS1
           11            WR00-DORDEA PICTURE  XX                      PJJPS1
                         VALUE                SPACE.                  PJJPS1
       01   WSS-BEGIN.                                                PJJPS1
            05   FILLER PICTURE X(7) VALUE 'WORKING'.                 PJJPS1
            05   BLANC  PICTURE X    VALUE SPACE.                     PJJPS1
            05   IK     PICTURE X.                                    PJJPS1
       01  PACBASE-CONSTANTS.                                         PJJPS1
           05  FILLER PICTURE X(50)  VALUE                            PJJPS1
              '0630 TOA03/10/06PJJPS1PDXC   10:41:21PJJPS1  NDOC      PJJPS1
           03/10/20063.5 V03O2/01/200602/09/2006'.
       01  CONSTANTS-PACBASE REDEFINES PACBASE-CONSTANTS              PJJPS1
           05  NUGNA   PICTURE X(5).                                  PJJPS1
           05  APPLI   PICTURE X(3).                                  PJJPS1
           05  DATGN   PICTURE X(8).                                  PJJPS1
           05  PROGR   PICTURE X(6).                                  PJJPS1
           05  CODUTI  PICTURE X(8).                                  PJJPS1
           05  TIMGN   PICTURE X(8).                                  PJJPS1
           05  PROGE   PICTURE X(8).                                  PJJPS1
           05  COBASE  PICTURE X(4).                                  PJJPS1
           05  DATGNC  PICTURE X(10).
```

```
    05  RELEAS  PICTURE X(7).
    05  DATGE   PICTURE X(10).
    05  DATSQ   PICTURE X(10).
01  DATCE.                                                  PJJPS1
  05  CENTUR   PICTURE XX   VALUE '20'.                     PJJPS1
  05  DATOR.                                                PJJPS1
    10  DATOA  PICTURE XX.                                  PJJPS1
    10  DATOM  PICTURE XX.                                  PJJPS1
    10  DATOJ  PICTURE XX.                                  PJJPS1
01  DAT6.                                                   PJJPS1
     10 DAT61   PICTURE XX.                                 PJJPS1
     10 DAT62   PICTURE XX.                                 PJJPS1
     10 DAT63   PICTURE XX.                                 PJJPS1
01  DAT8.                                                   PJJPS1
     10 DAT81   PICTURE XX.                                 PJJPS1
     10 DAT8S1  PICTURE X.                                  PJJPS1
     10 DAT82   PICTURE XX.                                 PJJPS1
     10 DAT8S2  PICTURE X.                                  PJJPS1
     10 DAT83   PICTURE XX.                                 PJJPS1
01  DAT8E    REDEFINES    DAT8.                            PJJPS1
     10 DAT81E  PICTURE X(4).                               PJJPS1
     10 DAT82E  PICTURE XX.                                 PJJPS1
     10 DAT83E  PICTURE XX.                                 PJJPS1
01  DAT6C.                                                  PJJPS1
    10  DAT61C  PICTURE XX.                                 PJJPS1
    10  DAT62C  PICTURE XX.                                 PJJPS1
     10 DAT63C  PICTURE X(4).                               PJJPS1
01  DAT8C.                                                  PJJPS1
     10 DAT81C  PICTURE XX.                                 PJJPS1
     10 FILLER  PICTURE X   VALUE '/'.                      PJJPS1
     10 DAT82C  PICTURE XX.                                 PJJPS1
     10 FILLER  PICTURE X   VALUE '/'.                      PJJPS1
     10 DAT83C  PICTURE X(4).                               PJJPS1
01  DATSEP  PICTURE X  VALUE '/'.                           PJJPS1
```

Variables and indexes

According to specifications provided by the user for the application program, PACBASE will generate the appropriate variables, indexes, etc.

CONDITIONAL VARIABLES

FTB: Final total control breaks.

- Group field for all FTBn's.

FTBn: Final total control break at level n.

- Used to indicate the status of processing. The value of this flag changes when the value of the nth key data element, (or of a key subordinate to the nth key) does not match the corresponding data element in the next record read.

- Generated if the program contains at least one input data structure for which a control break level has been requested.
- 1 = key of level n is being processed for the last time.
- 0 = (above is) not true

ITB: Initial total control breaks.

- Group field for all ITBn's.

ITBn: Initial total control break at level n.

- The first record at level n is being processed. By moving in the value of the FTBn flag, the iteration following a "last-record-detected" status identifies a new control break level.
- Generated with FTBn.
- 1 = key at level n is being processed for the first time.
- 0 = (above is) not true

dd-FB: Final control breaks on data structure dd.

- Group field for all dd-FBn's.

dd-FBn: Final control break on data structure dd at level n.

- The last record, at level n, on data structure dd, is ready for processing.
- Generated if the control break level given for D.S. dd is greater than or equal to n and if the key data element at level n has been declared in the data structure description.
- 1 = last record on dd at level n is being processed
- 0 = (above is) not true

dd-IB: Initial control breaks on data structure dd.

- Group-level field for all dd-IBn's.
- Generated with dd-FB.

dd-IBn: Initial control break on data structure dd, level n.

- The first record, at level n, on data structure dd, is ready for processing.
- Generated with dd-FBn.
- 1 = first record on dd, level n is being processed
- 0 = (above is) not true

dd-CF: Configuration indicator on data structure dd.

- Group field for dd-CFn's.
- Generated if file matching was requested for the dd file.

dd-CFn: Configuration on data structure dd at level n.

- At level n, the input record of data structure dd is to be processed in this program cycle.
- Generated if the file matching level specified for data structure dd is greater than or equal to n and if there is an nth key named for this data structure on the Segment Call of Elements screen.
- 1 = Yes - there is a record at level n to be processed this iteration
- 0 = (above is) not true

dd-OC: Occurrence variables for data structure dd.

- Group field for all dd-OCn's.
- Generated if file matching was requested for the principal file (USAGE OF D. S. = 'P').
- Provides information concerning the state of the update area (2-dd00).

dd-OCn: Occurrence on data structure dd at level n.

- A record of data structure dd, with key at level n, is being processed in this program cycle.
- Generated for principal data structures whose file matching level is greater than or equal to n and if there is an nth key named for this data structure on the Call of Data Structures screen.
- 1 = record in the update area (2-area) should exist on the output file: WRITE, REWRITE or CREATE.
- 0 = record in the update area should not be written on the output file: do not WRITE, or, DELETE.

FT: End-of-Processing indicator for all files.

- Used to indicate processing has been completed for all files when FT = ALL '1'.

dd-FT: End-of-Processing indicator for data structure dd.

- Used to indicate when processing for all the records of this data structure has been completed.
- Generated for every sequential data structure with a USAGE OF D.S. = 'C', 'M', 'N', 'P', and for every data structure with a USAGE of 'T' or 'X' and an ORGANIZATION = 'W' or 'L'.
- 1 = all records in data structure dd have been processed (including the last one).
- 0 = (above is) not true

dd-FI: End-of-File indicator on data structure dd.

- Used to indicate that all records of data structure dd have been read.

- Generated for all input data structures for which control breaks have been specified.
- 1 = all records in data structure dd have been read.
- 0 = (above is) not true

FBL: Minor-most final control break level detected in this run. This variable keeps track of the current level of break being processed this iteration.

- Generated if at least one control break level has been specified for any input data structure.

IBL: Minor-most initial control break level detected in this run. This variable keeps track of the current level of break being processed this iteration.

- Generated if at least one control break level has been specified for any input data structure.

INDEXES

Used for validation processing: I01 to I51.

I01: Stores the rank of the record type, according to the value of the record type number.

= 1 if only one record type.

I02: Stores the rank of the action type, according to its value (example: C = 1, M = 2, D = 3, etc.)

= 4 if no action type specified.

I03: Considering the aggregate of data elements within the transaction, stores a pointer (rank) to the first element of the specific part segment of the record being processed. This index is not generated when the transaction file consists of only one record type.

I04: Considering the aggregate of data elements within the transaction, stores a pointer (rank) to the last data element of the specific part segment being processed. This index is not generated when the transaction file consists of only one record type.

I06: Working index.

I50: Stores the rank of the last data element of the common part. This index is always generated. It is initialized by a VALUE clause.

I51: Stores the number of record types. This index is always generated. It is initialized by a VALUE clause.

Used for loading and consulting tables:

IddssM: Contains the value of the maximum number of entries specified by the user.

IddssL: Contains the value of the number of entries actually loaded from segment ss in data structure dd. This number cannot exceed the maximum specified above.

IddssR: Varying from 1 to IddssL, used for all look-ups on the table loaded from data structure dd, segment ss. Once the table is loaded, this index is initialized to zero if there is no overflow, or to the number of records read if an overflow has occurred.

These three indexes are generated for all records of:

1. data structures defined as tables, or
2. data structures with a non-redefined description with OCCURs, where there is a maximum number of records specified, or
3. if a table (W-ddss) was declared in the user Work Areas (-W) screen.

Used for print processing:

J00: Look-up index for the category table, CAT-TAB.

J01: Look-up index for the three dimensional table (containing the structure and constant part numbers, and line/page skip character), called ST-TA.

Jddrcc: Index associated with repetitive category cc for report r of data structure dd.

Contains the rank of the category (cc) being printed, at the time the structures are being loaded.

J05, J06, J07: Accumulator indexes.

Accumulators are always indexed, except at the grand totaling level.

The value in the index = the totaling level being processed.

Source data elements are added into the accumulators at the lowest level when the condition for printing the category has been satisfied.

When a final control break is detected, accumulators at each level (J07) are added into the accumulators at the next highest level (J06). This process is carried out for all accumulators, at a level less than or equal to the highest control break level detected in the iteration.

```
01   CONDITIONAL-VARIABLES.                                          PJJPS1
     05             FTB.                                             PJJPS1
       10           FTB1     PICTURE X VALUE '1'.                    PJJPS1
       10           FTB2     PICTURE X VALUE '1'.                    PJJPS1
       10           FTB3     PICTURE X VALUE '1'.                    PJJPS1
       10           FTB4     PICTURE X VALUE '1'.                    PJJPS1
       10           FTB5     PICTURE X VALUE '1'.                    PJJPS1
       10           FTB6     PICTURE X VALUE '1'.                    PJJPS1
     05           FBL        PICTURE 9 VALUE 1.                      PJJPS1
     05           IBL        PICTURE 9 VALUE ZERO.                   PJJPS1
     05             ITB.                                             PJJPS1
       10           ITB1     PICTURE X VALUE '1'.                    PJJPS1
       10           ITB2     PICTURE X VALUE '1'.                    PJJPS1
       10           ITB3     PICTURE X VALUE '1'.                    PJJPS1
       10           ITB4     PICTURE X VALUE '1'.                    PJJPS1
       10           ITB5     PICTURE X VALUE '1'.                    PJJPS1
       10           ITB6     PICTURE X VALUE '1'.                    PJJPS1
     05           CD-FB.                                             PJJPS1
       10         CD-FB1     PICTURE X VALUE '1'.                    PJJPS1
       10         CD-FB2     PICTURE X VALUE '1'.                    PJJPS1
     05           CL-FB.                                             PJJPS1
       10         CL-FB1     PICTURE X VALUE '1'.                    PJJPS1
       10         CL-FB2     PICTURE X VALUE '1'.                    PJJPS1
     05           LV-FB.                                             PJJPS1
       10         LV-FB1     PICTURE X VALUE '1'.                    PJJPS1
       10         LV-FB2     PICTURE X VALUE '1'.                    PJJPS1
     05           MV-FB.                                             PJJPS1
       10         MV-FB1     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-FB2     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-FB3     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-FB4     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-FB5     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-FB6     PICTURE X VALUE '1'.                    PJJPS1
     05           CD-IB.                                             PJJPS1
       10         CD-IB1     PICTURE X VALUE '1'.                    PJJPS1
       10         CD-IB2     PICTURE X VALUE '1'.                    PJJPS1
     05           CL-IB.                                             PJJPS1
       10         CL-IB1     PICTURE X VALUE '1'.                    PJJPS1
       10         CL-IB2     PICTURE X VALUE '1'.                    PJJPS1
     05           LV-IB.                                             PJJPS1
       10         LV-IB1     PICTURE X VALUE '1'.                    PJJPS1
       10         LV-IB2     PICTURE X VALUE '1'.                    PJJPS1
     05           MV-IB.                                             PJJPS1

       10         MV-IB1     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-IB2     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-IB3     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-IB4     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-IB5     PICTURE X VALUE '1'.                    PJJPS1
       10         MV-IB6     PICTURE X VALUE '1'.                    PJJPS1
```

```
      05             VCF.                                          PJJPS1
        10          CD-CF.                                         PJJPS1
          15        CD-CF1     PICTURE X VALUE '1'.                PJJPS1
          15        CD-CF2     PICTURE X VALUE '1'.                PJJPS1
          15        CD-CF3     PICTURE X VALUE '1'.                PJJPS1
        10          CL-CF.                                         PJJPS1
          15        CL-CF1     PICTURE X VALUE '1'.                PJJPS1
          15        CL-CF2     PICTURE X VALUE '1'.                PJJPS1
          15        CL-CF3     PICTURE X VALUE '1'.                PJJPS1
        10          GL-CF.                                         PJJPS1
          15        GL-CF1     PICTURE X VALUE '1'.                PJJPS1
          15        GL-CF2     PICTURE X VALUE '1'.                PJJPS1
        10          LV-CF.                                         PJJPS1
          15        LV-CF1     PICTURE X VALUE '1'.                PJJPS1
          15        LV-CF2     PICTURE X VALUE '1'.                PJJPS1
          15        LV-CF3     PICTURE X VALUE '1'.                PJJPS1
        10          MV-CF.                                         PJJPS1
          15        MV-CF1     PICTURE X VALUE '1'.                PJJPS1
          15        MV-CF2     PICTURE X VALUE '1'.                PJJPS1
          15        MV-CF3     PICTURE X VALUE '1'.                PJJPS1
      05            CD-OC.                                         PJJPS1
        10          CD-OC1     PICTURE X VALUE '0'.                PJJPS1
        10          CD-OC2     PICTURE X VALUE '0'.                PJJPS1
        10          CD-OC3     PICTURE X VALUE '0'.                PJJPS1
      05            CL-OC.                                         PJJPS1
        10          CL-OC1     PICTURE X VALUE '0'.                PJJPS1
        10          CL-OC2     PICTURE X VALUE '0'.                PJJPS1
        10          CL-OC3     PICTURE X VALUE '0'.                PJJPS1
      05            LV-OC.                                         PJJPS1
        10          LV-OC1     PICTURE X VALUE '0'.                PJJPS1
        10          LV-OC2     PICTURE X VALUE '0'.                PJJPS1
        10          LV-OC3     PICTURE X VALUE '0'.                PJJPS1
      05               FT.                                         PJJPS1
        10          CD-FT      PICTURE X VALUE '0'.                PJJPS1
        10          CL-FT      PICTURE X VALUE '0'.                PJJPS1
        10          EN-FT      PICTURE X VALUE '0'.                PJJPS1
        10          GL-FT      PICTURE X VALUE '0'.                PJJPS1
        10          LV-FT      PICTURE X VALUE '0'.                PJJPS1
        10          MV-FT      PICTURE X VALUE '0'.                PJJPS1
      05               FI.                                         PJJPS1
        10          CD-FI      PICTURE X VALUE '0'.                PJJPS1
        10          CL-FI      PICTURE X VALUE '0'.                PJJPS1
        10          LV-FI      PICTURE X VALUE '0'.                PJJPS1
        10          MV-FI      PICTURE X VALUE '0'.                PJJPS1
01    INDICES  COMPUTATIONAL  SYNC.                                 PJJPS1
      05        I01       PICTURE S9(4) VALUE +1.                   PJJPS1
      05        I02       PICTURE S9(4) VALUE +4.                   PJJPS1
      05        I03       PICTURE S9(4) VALUE  ZERO.                PJJPS1
      05        I04       PICTURE S9(4) VALUE  ZERO.                PJJPS1
      05        I50       PICTURE S9(4) VALUE +006.                 PJJPS1
      05        I06       PICTURE S9(4) VALUE  ZERO.                PJJPS1
      05        I51       PICTURE S9(4) VALUE +002.                 PJJPS1
      05        J00       PICTURE S9(4) VALUE +1.                   PJJPS1
      05        J01       PICTURE S9(4) VALUE +1.                   PJJPS1
      05        J05       PICTURE S9(4) VALUE +0.                   PJJPS1
```

```
05       J06       PICTURE S9(4) VALUE +0.                PJJPS1
05       J07       PICTURE S9(4) VALUE +0.                PJJPS1
05       JLI1DD    PICTURE S9(4) VALUE  ZERO.             PJJPS1
05       JLI1DDM   PICTURE S9(4) VALUE  ZERO.             PJJPS1
05       JED3FA    PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          IMO00L PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          IMO00R PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          IMO00M PICTURE S9(4) VALUE +0012.             PJJPS1
05          ITD01L PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          ITD01R PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          ITD01M PICTURE S9(4) VALUE +0103.             PJJPS1
05          ITD02L PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          ITD02R PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          ITD02M PICTURE S9(4) VALUE +0016.             PJJPS1
05          IWC02L PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          IWC02R PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          IWC02M PICTURE S9(4) VALUE +0011.             PJJPS1
05          IWC03L PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          IWC03R PICTURE S9(4) VALUE  ZERO.             PJJPS1
05          IWC03M PICTURE S9(4) VALUE +0011.             PJJPS1
```

Key, validation, print areas

KEY STORAGE AREAS: CONF-CALCULATION-AREA

IND: .Stores the major-most key level of all input data structures to be matched.

.Generated only if there are at least two input data structures to be matched.

ddIND: .Stores the current value of the key of the record on data structure dd.

.Generated only for an input data structure with file matching.

RECORD COUNTERS: FILE-COUNTERS

5-dd00-RECCNT Record counter for data structure dd.

.This counter is generated for each data structure whose USAGE OF D.S. is not 'T' or 'X'.

.Incremented with each READ or WRITE of the d.s.

VALIDATION PROCESSING (WORK AREAS AND VARIABLES)

DE-TAB: .Stores DATA ELEMENT PRESENCE VALIDATION specifica_ tions for each transaction file data element.

.Generated only if the program has a transaction file to be validated.

DE-ERR: .Stores the presence status of each data element of the transaction being processed.

Each elementary data element (eeeeee), other than FILLER, ENPR, GRPR, ERUT and their sub-elements, is provided with a status field within the table. This field is named ER-ss-eeeeee (ss = SEGMENT CODE).

The values vary at different points in the processing cycle:

0 = data element absent,

1 = data element present,

2 = invalid absence of data element,

3 = invalid presence of data element,

4 = erroneous class,

5 = invalid content.

DE-TTE: .Stores the presence validation (optional, required or not allowed) to be done on the data element being processed.

.Generated only if the program has a transaction file to be validated.

ID-ER: .The last field in the table is ID-ER and is used for storing the record identification status:

0 = record type and action code are valid values,

5 = error detected on record type,

6 = error detected on action code.

DEL-ER: .Stores the presence status of the data element being processed.

.Generated only if the program contains a transaction file (to be validated or not).

DE-ERR: .Used only to carry out transfers between DE-ERR and a data structure (USAGE OF D.S. = 'M', 'N' or 'E') with a reduced error array (RESERVED ERROR CODES IN TRANS. FILE = 'W').

ER-ID: .Will receive ID-ER.

ER-PRR: .Generated if a reduced error table has been requested on at least one of the D.S. (transaction file with or without errors detected).

ER-PR0: .Will receive the error status of each data element belonging to the common part of the data structure.

ER-PRM: .Will receive the error status of each data element belonging to the specific part segment being processed.

SE-TAB: .Stores the theoretical absence or presence of each record type of the transaction file for the various action codes specified. (See SEGMENT PRESENCE on the Segment Definition screen).

.Generated only if the program contains a transaction file to be validated.

SE-ERR: .Stores the presence status of each transaction file record type.

.Generated if the program contains a transaction file (to be validated or not).

Each record type is provided with a status field within this table. This field is named SE-ER(I01).

The values vary at different points in the processing cycle:

0 = record absent,

1 = record present,

2 = invalid absence of record,

3 = invalid presence of record,

7 = duplicate record,

8 = invalid creation,

9 = invalid modification or deletion.

TR-ER: .The last field in the table is named TR-ER and is used for storing errors detected.

1 = no error detected.

SE-ERE: .Stores the presence status of the record being pro_ cessed.

.Generated if the program contains a transaction file (to be validated or not).

GR-ER: .Stores information concerning errors detected on a group of transactions which update a record, of at least one principal data structure.

.Generated only if the program updates one or more data structure.

UT-ERUT: .Stores the user's errors. If the program contains a transaction file, (USAGE OF D.S. = 'M', 'N' or 'E') with the user error table 'ERUT', the description generated will be as specified on the Call of Data Structures (-CD) screen, using sub-elements named UT-eeeeee.

TABLES USED FOR REPORTS

CAT-TAB: .Category table: stores all categories to be printed in this iteration.

.Generated only if categories have been defined for at least one report without direct printing, in the program.

ST-TA: .Table storing the structure number, constant part number, and page/line skip for the category to be printed.

.Generated only if categories have been defined for at least one report without direct printing, in the program.

1. .Table containing constants for report r.

STORE AREAS FOR PRINT PROCEDURES

TS-r-cc: .Definition of the contents of category cc of report r.

.Generated only for reports with categories not printed directly.

ABS-r-cc: .Variable indicating if category cc of report r begins after a page skip.

.Generated only for reports with categories not printed directly.

1. .Number of lines necessary for printing category cc of report r.

These areas are generated only if categories have been defined for the report.

ACCUMULATORS

rst-CPT OCCURS n.

Group level of the accumulators associated with structure st in report r. n is the lowest accumulation level for this structure appearing in the report definition (default 1).

Trst-eeeeee(n)

Accumulator at level n, for data element eeeeee of structure st in report r.

Grst-eeeeee

Grand total accumulator, for data element eeeeee of structure st in report r. Appears if the structure is used in a category with grand totaling (TYPE OF LINE IN REPORT = '0').

PRINT VARIABLES AND COUNTERS

```
ST-SLS       .A table subdivided into:
   STX       -STRUCTURE NUMBER (redefined by ST9),
   J02       -CONSTANT PART NUMBER,
   LSKP      -SKIP to be executed before writing a line,
   NUPOL     -CHAR. SET OPTION : SPECIAL PRINTER
CATX         .Stores the CATEGORY OF REPORT being printed.
5-dd00-rPC   .Page counter for report r of data structure dd.
5-dd00-rLC   .Line counter for report r of data structure dd,
              incremented at category table load time and
              indicating the line number of the last line of
              the category just printed. Initialized at 99 by
              value.
5-dd00-rLC1 .Line counter for report r of data structure dd,
              incremented at each output line and indicating
              the line number of the last written line.
5-dd00-rLCM .Counter for maximum number of lines per page.
5-dd00-rRC   .Counter for number of lines written for the
              report. Incremented after writing.
5-dd00-rTP   .Top of page indicator for report r of D.S. dd.
```

All these variables are generated for report r, of D.S. dd, for which structures have been defined.

STORE AREAS FOR PRINT PROCEDURES

These areas are generated only if categories have been defined for a Report with no line printing.

TS-r-cc: definition of the contents of category cc of Report r.

ABS-r-cc: variable indicating if category cc of Report r begins after a page skip.

Generated only for Reports with categories not printed directly.

1. Number of lines necessary for printing category cc of Report r.

ACCUMULATORS

rst-CPT OCCURS n.

Group level of the accumulators associated with structure ss of Report r. N is the lowest accumulation level for this structure appearing in the Report definition (default 1).

Trst-eeeee (n)

Accumulator at level n associated with Data Element eeeeee of structure st in Report r.

Grst-eeeeee

Grand total accumulator associated with Data Element eeeeee of structure st in Report r. Appears if the structure is used in a category with grand totaling (type of line in Report = '0').

```
01   CONF-CALCULATION-AREA.                                    PJJPS1
    05              IND.                                       PJJPS1
     16            TIND3.                                      PJJPS1
     17            TIND2.                                      PJJPS1
     18            TIND1.                                      PJJPS1
     19             IND1     PICTURE X(001).                   PJJPS1
     18             IND2     PICTURE X(002).                   PJJPS1
     17             IND3     PICTURE X(002).                   PJJPS1
    05           CDIND.                                       PJJPS1
     10          CDIND1.                                      PJJPS1
      15       CD-IN-NOCL11  PICTURE   X.                     PJJPS1
     10          CDIND2.                                      PJJPS1
      15       CD-IN-NOCL12  PICTURE   XX.                    PJJPS1
     10          CDIND3.                                      PJJPS1
      15       CD-IN-NOCL2   PICTURE   XX.                    PJJPS1
    05           CLIND.                                       PJJPS1
     10          CLIND1.                                      PJJPS1
      15       CL-IN-NOCL11  PICTURE   X.                     PJJPS1
     10          CLIND2.                                      PJJPS1
      15       CL-IN-NOCL12  PICTURE   XX.                    PJJPS1
     10          CLIND3.                                      PJJPS1
      15       CL-IN-NOCL2   PICTURE   XX.                    PJJPS1
    05           GLIND.                                       PJJPS1
     10          GLIND1.                                      PJJPS1
      15       GL-IN-NOCL11  PICTURE   X.                     PJJPS1
     10          GLIND2.                                      PJJPS1
```

```
      15        GL-IN-NOCL12  PICTURE   XX.                      PJJPS1
    05             LVIND.                                        PJJPS1
     10            LVIND1.                                       PJJPS1
      15        LV-IN-NOCL11  PICTURE   X.                       PJJPS1
     10            LVIND2.                                       PJJPS1
      15        LV-IN-NOCL12  PICTURE   XX.                      PJJPS1
     10            LVIND3.                                       PJJPS1
      15        LV-IN-NOCL2   PICTURE   XX.                      PJJPS1
    05             MVIND.                                        PJJPS1
     10            MVIND1.                                       PJJPS1
      15        MV-IN-NOCL11  PICTURE   X.                       PJJPS1
     10            MVIND2.                                       PJJPS1
      15        MV-IN-NOCL12  PICTURE   XX.                      PJJPS1
     10            MVIND3.                                       PJJPS1
      15        MV-IN-NOCL2   PICTURE   XX.                      PJJPS1
01   FILE-COUNTERS            COMPUTATIONAL-3.                  PJJPS1
     05       5-CD00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-CL00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-DC00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-EN00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-GL00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-LC00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-LV00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-MV00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-SE00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-VL00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-VM00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-WA00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
     05       5-WR00-RECCNT PICTURE S9(9) VALUE ZERO.           PJJPS1
01  STATUS-AREA.                                               PJJPS1
     05          1-MO00-STATUS PICTURE XX VALUE ZERO.           PJJPS1
01   VALIDATION-VARIABLES.                                     PJJPS1
     05         DE-TAB.                                        PJJPS1
       10    EN-00NOCL11 PICTURE X(6) VALUE 'OOOOOO'.         PJJPS1
       10    EN-00NOCL12 PICTURE X(6) VALUE 'OOOOOO'.         PJJPS1
       10    EN-00NOCL2  PICTURE X(6) VALUE 'OOOOOO'.         PJJPS1
       10    EN-00NUORD  PICTURE X(6) VALUE 'OOOOOO'.         PJJPS1
       10    EN-00CODMV  PICTURE X(6) VALUE 'FFFFFF'.         PJJPS1
       10    EN-00NUCAR  PICTURE X(6) VALUE 'FFFFFF'.         PJJPS1
       10    EN-01NOMCL  PICTURE X(6) VALUE 'OFFFFF'.         PJJPS1
       10    EN-01ADRES  PICTURE X(6) VALUE 'OFFFFF'.         PJJPS1
       10    EN-01NUDEP  PICTURE X(6) VALUE 'OFFFFF'.         PJJPS1
       10    EN-02MREEL9 PICTURE X(6) VALUE 'FFFFFF'.         PJJPS1
       10    EN-02DALI   PICTURE X(6) VALUE 'FFFFFF'.         PJJPS1
     05         DE-T    REDEFINES  DE-TAB.                    PJJPS1
       10       DE-TTT  OCCURS  011.                          PJJPS1
         15     DE-TT   OCCURS     6  PICTURE X.              PJJPS1
     05         DE-ERR.                                       PJJPS1
       10       DE-ER   OCCURS  011   PICTURE X.              PJJPS1
       10       ID-ER   PICTURE X       VALUE ZERO.           PJJPS1
     05         DE-E    REDEFINES  DE-ERR.                    PJJPS1
     07           ER-00.                                      PJJPS1
     10           ER-00-NOCL.                                 PJJPS1
     11           ER-00-NOCL11 PICTURE X.                     PJJPS1
     11           ER-00-NOCL12 PICTURE X.                     PJJPS1
```

```
     11           ER-00-NOCL2  PICTURE X.                          PJJPS1
     10           ER-00-NUORD  PICTURE X.                          PJJPS1
     10           ER-00-CODMV  PICTURE X.                          PJJPS1
     10           ER-00-NUCAR  PICTURE X.                          PJJPS1
     07           ER-01.                                           PJJPS1
     10           ER-01-NOMCL  PICTURE X.                          PJJPS1
     10           ER-01-ADRES  PICTURE X.                          PJJPS1
     10           ER-01-NUDEP  PICTURE X.                          PJJPS1
     07           ER-02.                                           PJJPS1
     10           ER-02-MREEL9 PICTURE X.                          PJJPS1
     10           ER-02-DALI   PICTURE X.                          PJJPS1
     07         FILLER  PICTURE X.                                 PJJPS1
     05         DEL-ER  PICTURE X.                                 PJJPS1
     05         DE-TTE  PICTURE X.                                 PJJPS1
     05        ER-PRR.                                             PJJPS1
       10      ER-ID  PICTURE X       VALUE ZERO.                 PJJPS1
       10      ER-PR0 PICTURE X(006).                             PJJPS1
       10      ER-PRM.                                            PJJPS1
         15    ER-PR    OCCURS  003   PICTURE X.                  PJJPS1
     05         SE-TAB.                                           PJJPS1
       10       FILLER  PICTURE X(6) VALUE  'OIIFFF'.            PJJPS1
       10       FILLER  PICTURE X(6) VALUE  'OFIOOO'.            PJJPS1
     05         SE-T    REDEFINES  SE-TAB.                        PJJPS1
       10       SE-TTT  OCCURS    002.                            PJJPS1
         15     SE-TT   OCCURS     6  PICTURE X.                  PJJPS1
     05         SE-ERR.                                           PJJPS1
       10       SE-ER   OCCURS    002 PICTURE X.                  PJJPS1
       10       TR-ER   PICTURE X     VALUE '1'.                  PJJPS1
     05         SEG-ER  PICTURE X.                                PJJPS1
     05         GR-ER   PICTURE X    VALUE ZERO.                  PJJPS1
     05         LE-FIENR PICTURE X(4) VALUE 'MV00'.               PJJPS1
     05              UT-ERUT.                                     PJJPS1
     11              UT-UPR    PICTURE  X                         PJJPS1
                     OCCURS     010.                              PJJPS1
01   CAT-TAB.                                                     PJJPS1
     05  FILLER         PICTURE X(100) VALUE SPACES.              PJJPS1
     05  FILLER         PICTURE X(100) VALUE SPACES.              PJJPS1
01   CAT-TAB-R  REDEFINES CAT-TAB.                                PJJPS1
     05  CAT    PICTURE XX       OCCURS 0100.                     PJJPS1
01   ST-TA.                                                       PJJPS1
     05  ST-ABS      PICTURE X  VALUE SPACE.                     PJJPS1
     05  ST-T.                                                   PJJPS1
     07  ST-TT    OCCURS 40.                                     PJJPS1
       10  ST-ST     PICTURE XX.                                 PJJPS1
       10  ST-LI     PICTURE 99.                                 PJJPS1
       10  ST-SA     PICTURE 99.                                 PJJPS1
01  CONTENT-OF-CATEGORIES.                                       PJJPS1
    05          TS-3-DA.                                         PJJPS1
    10         ABS-3-DA PICTURE X VALUE  '*'.                    PJJPS1
     10  FILLER    PICTURE       X(30) VALUE                     PJJPS1
    '010101000201000302000401000301'.                            PJJPS1

    05          TS-3-EA.                                         PJJPS1
    10         ABS-3-EA PICTURE X VALUE  ' '.                    PJJPS1
     10  FILLER    PICTURE       X(12) VALUE                     PJJPS1
```

```
    '000501020501'.                                                PJJPS1
    05          TS-3-FA.                                           PJJPS1
    10         ABS-3-FA PICTURE X VALUE  ' '.                      PJJPS1
     10  FILLER    PICTURE       X(06) VALUE                       PJJPS1
    '030501'.                                                      PJJPS1
    05          TS-3-GA.                                           PJJPS1
    10         ABS-3-GA PICTURE X VALUE  ' '.                      PJJPS1
     10  FILLER    PICTURE       X(12) VALUE                       PJJPS1
    '000501040501'.                                                PJJPS1
    05          TS-3-HA.                                           PJJPS1
    10         ABS-3-HA PICTURE X VALUE  ' '.                      PJJPS1
     10  FILLER    PICTURE       X(06) VALUE                       PJJPS1
    '040501'.                                                      PJJPS1
    05          TS-3-IA.                                           PJJPS1
    10         ABS-3-IA PICTURE X VALUE  ' '.                      PJJPS1
     10  FILLER    PICTURE       X(06) VALUE                       PJJPS1
    '040501'.                                                      PJJPS1
    05          TS-3-IL.                                           PJJPS1
    10         ABS-3-IL PICTURE X VALUE  ' '.                      PJJPS1
     10  FILLER    PICTURE       X(12) VALUE                       PJJPS1
    '000501000301'.                                                PJJPS1
    05          TS-3-JA.                                           PJJPS1
    10         ABS-3-JA PICTURE X VALUE  ' '.                      PJJPS1
     10  FILLER    PICTURE       X(06) VALUE                       PJJPS1
    '040002'.                                                      PJJPS1
01  SIZE-OF-CATEGORIES        COMPUTATIONAL-3.                    PJJPS1
     05          1-BC-NL PICTURE S99 VALUE +11.                    PJJPS1
     05          1-DD-NL PICTURE S99 VALUE +01.                    PJJPS1
     05          1-EE-NL PICTURE S99 VALUE +01.                    PJJPS1
     05          3-DA-NL PICTURE S99 VALUE +05.                    PJJPS1
     05          3-EA-NL PICTURE S99 VALUE +02.                    PJJPS1
     05          3-FA-NL PICTURE S99 VALUE +01.                    PJJPS1
     05          3-GA-NL PICTURE S99 VALUE +02.                    PJJPS1
     05          3-HA-NL PICTURE S99 VALUE +01.                    PJJPS1
     05          3-IA-NL PICTURE S99 VALUE +01.                    PJJPS1
     05          3-IL-NL PICTURE S99 VALUE +02.                    PJJPS1
     05          3-JA-NL PICTURE S99 VALUE +02.                    PJJPS1
01   TOTALLING-AREA           COMPUTATIONAL-3.                    PJJPS1
     05        304-CPT  OCCURS  2.                                 PJJPS1
       10     T304-QUCO      PICTURE S9(07).                      PJJPS1
       10     T304-QTLI      PICTURE S9(07).                      PJJPS1
     05       G304-QUCO      PICTURE S9(07)        VALUE ZERO.    PJJPS1
     05       G304-QTLI      PICTURE S9(07)        VALUE ZERO.    PJJPS1
01  PRINT-COUNTERS-AND-VARIABLES.                                 PJJPS1
     05         COUNTERS      COMPUTATIONAL-3.                    PJJPS1
     10       5-ED00-3LCM PICTURE S999 VALUE +60.                 PJJPS1
     10       5-ED00-3RC  PICTURE S9(9) VALUE ZERO.               PJJPS1
     10       5-ED00-3LC  PICTURE S999 VALUE +60.                 PJJPS1
     10       5-ED00-3LC1 PICTURE S999 VALUE +60.                 PJJPS1
     10       5-ED00-3PC  PICTURE S9(7) VALUE ZERO.               PJJPS1
     10       5-LI00-1LCM PICTURE S999 VALUE +60.                 PJJPS1
     10       5-LI00-1RC  PICTURE S9(9) VALUE ZERO.               PJJPS1
     10       5-LI00-1LC  PICTURE S999 VALUE +60.                 PJJPS1
     10       5-LI00-1LC1 PICTURE S999 VALUE +60.                 PJJPS1
     10       5-LI00-1PC  PICTURE S9(7) VALUE ZERO.               PJJPS1
```

```
     05       5-LI00-1TP  PICTURE X    VALUE '1'.                PJJPS1
     05       5-ED00-3TP  PICTURE X    VALUE '1'.                PJJPS1
     05         ST-SLS.                                           PJJPS1
     10         STX      PICTURE XX.                              PJJPS1
     10         ST9  REDEFINES STX PICTURE 99.                    PJJPS1
     10         J02      PICTURE 99.                              PJJPS1
     10         LSKP     PICTURE 99.                              PJJPS1
     10         NUPOL    PICTURE X.                               PJJPS1
     05         CATX     PICTURE XX  VALUE SPACE.                 PJJPS1
01  REPORT-CONSTANTS.                                             PJJPS1
     05           1-LAB.                                          PJJPS1
         10       1-LAB01.                                        PJJPS1
             15  FILLER  PICTURE X(44) VALUE                      PJJPS1
    '       UPDATE REPORT        XXXXXXX         '.               PJJPS1
             15  FILLER  PICTURE X(01) VALUE                      PJJPS1
    ' '.                                                          PJJPS1
         10       1-LAB02.                                        PJJPS1
             15  FILLER  PICTURE X(44) VALUE                      PJJPS1
    'NUMBER OF VALID TRANSACTIONS       :  495   '.               PJJPS1
             15  FILLER  PICTURE X(01) VALUE                      PJJPS1
    ' '.                                                          PJJPS1
         10       1-LAB03.                                        PJJPS1
             15  FILLER  PICTURE X(44) VALUE                      PJJPS1
    'NUMBER OF INVALID TRANSACTIONS     :   55   '.               PJJPS1
             15  FILLER  PICTURE X(01) VALUE                      PJJPS1
    ' '.                                                          PJJPS1
         10       1-LAB04.                                        PJJPS1
             15  FILLER  PICTURE X(44) VALUE                      PJJPS1
    'NUMBER OF TRANSACTIONS             :  550   '.               PJJPS1
             15  FILLER  PICTURE X(01) VALUE                      PJJPS1
    ' '.                                                          PJJPS1
         10       1-LAB05.                                        PJJPS1
             15  FILLER  PICTURE X(44) VALUE                      PJJPS1
    'PERCENTAGE OF INVALID TRANSACTIONS :   10,00'.               PJJPS1
             15  FILLER  PICTURE X(01) VALUE                      PJJPS1
    ' '.                                                          PJJPS1
         10       1-LAB06.                                        PJJPS1
             15  FILLER  PICTURE X(44) VALUE                      PJJPS1
    'NUMBER OF FILE RECORDS             :        '.               PJJPS1
             15  FILLER  PICTURE X(01) VALUE                      PJJPS1
    ' '.                                                          PJJPS1
         10       1-LAB07.                                        PJJPS1
             15  FILLER  PICTURE X(44) VALUE                      PJJPS1
    '                                           '.               PJJPS1
             15  FILLER  PICTURE X(01) VALUE                      PJJPS1
    ' '.                                                          PJJPS1
         10       1-LAB08.                                        PJJPS1
             15  FILLER  PICTURE X(44) VALUE                      PJJPS1
    '                              CD  :  100   '.                PJJPS1
             15  FILLER  PICTURE X(01) VALUE                      PJJPS1
    ' '.                                                          PJJPS1
         10       1-LAB09.                                        PJJPS1
             15  FILLER  PICTURE X(44) VALUE                      PJJPS1
    '********************************************'.               PJJPS1
             15  FILLER  PICTURE X(01) VALUE                      PJJPS1
```

```
'*'.                                                                    PJJPS1
 05           1-LAB-R REDEFINES 1-LAB.                                  PJJPS1
      10   1-LI00-1 OCCURS  009.                                        PJJPS1
       15  FILLER        PICTURE  X(00045).                            PJJPS1
 05           3-LAB.                                                    PJJPS1
     10       3-LAB01.                                                  PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'        ORDER  AND  DELIVERY  REPORT  AT  07'.                         PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'/10/1986                 PAGE 123          '.                         PJJPS1
         15  FILLER  PICTURE X(08) VALUE                               PJJPS1
'        '.                                                            PJJPS1
     10       3-LAB02.                                                  PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'  ****************************************'.                          PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'*************                                '.                         PJJPS1
         15  FILLER  PICTURE X(08) VALUE                               PJJPS1
'        '.                                                            PJJPS1
     10       3-LAB03.                                                  PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'****************************************'.                            PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'****************************************'.                            PJJPS1
         15  FILLER  PICTURE X(08) VALUE                               PJJPS1
'********'.                                                            PJJPS1
     10       3-LAB04.                                                  PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'* CUSTOM *        NAME          *           '.                         PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'                * ORDERED  *DELIVERED *  BA'.                         PJJPS1
         15  FILLER  PICTURE X(08) VALUE                               PJJPS1
'LANCE  *'.                                                            PJJPS1
     10       3-LAB05.                                                  PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'*        *                        *           '.                      PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'               *          *         *    '.                           PJJPS1
         15  FILLER  PICTURE X(08) VALUE                               PJJPS1
'       *'.                                                            PJJPS1
 05           3-LAB-R REDEFINES 3-LAB.                                  PJJPS1
      10   1-LI00-3 OCCURS  005.                                        PJJPS1
       15  FILLER        PICTURE  X(00096).                            PJJPS1
 05           4-LAB.                                                    PJJPS1
     10       4-LAB01.                                                  PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
'***BATCH TOTAL                             '.                         PJJPS1
         15  FILLER  PICTURE X(26) VALUE                               PJJPS1
'                        '.                                            PJJPS1
     10       4-LAB02.                                                  PJJPS1
         15  FILLER  PICTURE X(44) VALUE                               PJJPS1
' **SUBTOTAL                                '.                         PJJPS1
         15  FILLER  PICTURE X(26) VALUE                               PJJPS1
'                        '.                                            PJJPS1
     10       4-LAB03.                                                  PJJPS1
```

```
          15  FILLER  PICTURE X(44) VALUE                    PJJPS1
'  *CHECK TOTAL                              '.              PJJPS1
          15  FILLER  PICTURE X(26) VALUE                    PJJPS1
'                          '.                                PJJPS1
      10        4-LAB04.                                    PJJPS1
          15  FILLER  PICTURE X(44) VALUE                    PJJPS1
'   SUM TOTAL                                 '.            PJJPS1
          15  FILLER  PICTURE X(26) VALUE                    PJJPS1
'                          '.                                PJJPS1
 05           4-LAB-R REDEFINES 4-LAB.                       PJJPS1
       10   1-LI00-4 OCCURS  004.                            PJJPS1
        15  FILLER        PICTURE  X(00070).                 PJJPS1
```

Data structure work areas

All input data structures for which a control break level has been entered, will be described completely, in the WORKING STORAGE SECTION.

The common part is named in the form 1-dd00. The variable parts either redefine each other or are defined successively, depending upon the RECORD TYPE/USE WITHIN D.S. value.

They are named 1-ddss where:

dd = DATA STRUCTURE CODE IN THE PROGRAM,

ss = SEGMENT CODE.

Each data element is named in the form 1-dd00-eeeeee, with its format, or sub-defined if it is a group level field.

When the D.S. has redefined variable length segments, each definition is completed with a FILLER so that each segment is the same length (equal to the longest).

The '1-' area is loaded at the READ of each d.s., from the data last read. Thus the read area of a data structure with control breaks will only be used for calculating these control breaks. The segment being processed is always in the '1-' (work) area.

A '2-' area is set up for each input principal file (USAGE OF D.S. = 'P') in which a common part is declared, as well as variable parts, through successive redefinition, according to the RECORD TYPE / USE WITHIN D.S. entered. The data elements are described in detail as in a '1-' area. All updating is done in this area.

An area in the WORKING-STORAGE SECTION is set up for each table D.S. For each segment to be loaded, an area will be allocated in the form 1-ddss OCCURS n, where:

n = OCCURRENCES OF SEGMENT IN TABLE.

If the D.S. has been defined with a USAGE of 'T', all data elements will be declared and loaded. If the USAGE is 'X', only data elements other than FILLER and the record type will appear. All elementary data elements at the 01 level, and all elementary or group data elements at the 02 level will be loaded.

The data element descriptions are the same as for the '1-' work areas for D.S.'s with control breaks, except for data elements of the common part which are described in each specific part segment.

For each print D.S., an area called 6-dd00 is set up, where dd is the DATA STRUCTURE CODE IN THE PROGRAM. All the lines of the different reports will be moved into this area before being written. This area is subdivided at level 05 by successive redefinitions for each report appearing in the print data structure. At the 10 level, the data elements common to all printed lines appear, as well as the different report structures. The names appear in the form 6-ddrst where:

```
dd = DATA STRUCTURE CODE IN THE PROGRAM,
r  = LAST CHARACTER OF REPORT CODE,
st = STRUCTURE NUMBER.
```

The structure descriptions are redefinitions of each other. The descriptions contain all the receiving data elements, plus FILLER's whose length is calculated by the generator. The data-names are in the form 6-ddrst-eeeeee, where:

```
eeeeee = DATA ELEMENT CODE in the Report Call of
         Elements (-CE) screen.
```

NOTE

The user can modify the contents of D.S work areas through data structure descriptions. However, their location in the the generated program cannot be modified.

THE USER WORK AREAS

Here, the user will find area or section names defined by Work Areas (-W) lines, where the CODE FOR COBOL PLACEMENT is numeric. If this code is alphabetic, the Work Areas (-W) lines are inserted at the beginning of WORKING-STORAGE.

The descriptions of some data structures with ORGANIZATION 'L' or 'D' are also located here.

There is a description among the user's areas generated for each d.s. with ORGANIZATION = 'L' or 'D' with an alphabetic CODE FOR COBOL PLACEMENT.

For these data structures, the user can request any possible description type in this area.

Moreover, using the level number and/or location, the D.S. description can appear under a level 01, or in a particular section (LINKAGE, IDS, ...) entered via the Work Areas (-W) screen.

NOTE

The user can modify the work areas, with respect to content and location, using the CODE FOR COBOL PLACEMENT and the LINE NUMBER of the Work Areas (-W) screen with data structures with an ORGANIZATION = 'L' or 'D'.

```
01           6-ED00.                                          PJJPS1
    05       6-ED00-3.                                        PJJPS1
     10      6-ED300-LSKP   PICTURE  X.                      PJJPS1
     10      6-ED300        PICTURE  X(096).                 PJJPS1
     10      6-ED301    REDEFINES   6-ED300.                 PJJPS1
       15    FILLER         PICTURE  X(045).                 PJJPS1
       15    6-ED301-DATEM  PICTURE X(10).                   PJJPS1
       15    FILLER         PICTURE  X(020).                 PJJPS1
       15    6-ED301-PAGE   PICTURE ZZ9.                     PJJPS1
       15    FILLER         PICTURE  X(018).                 PJJPS1
     10      6-ED302    REDEFINES   6-ED300.                 PJJPS1
       15    FILLER         PICTURE  X(009).                 PJJPS1
       15    6-ED302-NOCL   PICTURE X(5).                    PJJPS1
       15    FILLER         PICTURE  X(002).                 PJJPS1
       15    6-ED302-NOMCL  PICTURE X(20).                   PJJPS1
       15    FILLER         PICTURE  X(060).                 PJJPS1
     10      6-ED303    REDEFINES   6-ED300.                 PJJPS1
       15    FILLER         PICTURE  X(037).                 PJJPS1
       15    6-ED303-FILLER PICTURE X(9).                    PJJPS1
       15    FILLER         PICTURE  X(001).                 PJJPS1
       15    6-ED303-JED3FA PICTURE 9.                       PJJPS1
       15    FILLER         PICTURE  X(004).                 PJJPS1
       15    6-ED303-DATE   PICTURE X(6).                    PJJPS1
       15    FILLER         PICTURE  X(016).                 PJJPS1
       15    6-ED303-QULI   PICTURE Z(4)9,99.                PJJPS1
```

```
          15     FILLER         PICTURE  X(014).                  PJJPS1
        10       6-ED304    REDEFINES   6-ED300.                 PJJPS1
          15     FILLER         PICTURE  X(034).                  PJJPS1
          15     6-ED304-4      PICTURE X(20).                    PJJPS1
          15     FILLER         PICTURE  X(001).                  PJJPS1
          15     6-ED304-NOCL11 PICTURE X.                        PJJPS1
          15     6-ED304-NOCL12 PICTURE XX.                       PJJPS1
          15     6-ED304-NOCL2  PICTURE XX.                       PJJPS1
          15     FILLER         PICTURE  X(003).                  PJJPS1
          15     6-ED304-QUCO   PICTURE Z(4)9,99.                 PJJPS1
          15     FILLER         PICTURE  X(003).                  PJJPS1
          15     6-ED304-QTLI   PICTURE Z(4)9,99.                 PJJPS1
          15     FILLER         PICTURE  X(003).                  PJJPS1
          15     6-ED304-SOLDE  PICTURE -(5)9,99.                 PJJPS1
          15     FILLER         PICTURE  X(002).                  PJJPS1
01              6-LI00.                                          PJJPS1
      05        6-LI00-1.                                        PJJPS1
       10       6-LI100-ETAT   PICTURE X.                        PJJPS1
       10       6-LI100-LSKP   PICTURE 99.                       PJJPS1
       10       6-LI100-PAGE   PICTURE ZZ9.                      PJJPS1
       10       6-LI100-NULIG  PICTURE 9(3).                     PJJPS1
       10       6-LI100        PICTURE  X(045).                  PJJPS1
       10       6-LI101    REDEFINES   6-LI100.                 PJJPS1
         15     FILLER         PICTURE  X(038).                  PJJPS1
         15     6-LI101-ACCEP  PICTURE ZZ9.                      PJJPS1
         15     FILLER         PICTURE  X(004).                  PJJPS1
       10       6-LI102    REDEFINES   6-LI100.                 PJJPS1
         15     FILLER         PICTURE  X(038).                  PJJPS1
         15     6-LI102-REFUS  PICTURE ZZ9.                      PJJPS1
         15     FILLER         PICTURE  X(004).                  PJJPS1
       10       6-LI103    REDEFINES   6-LI100.                 PJJPS1
         15     FILLER         PICTURE  X(038).                  PJJPS1
         15     6-LI103-TOTAL  PICTURE ZZ9.                      PJJPS1
         15     FILLER         PICTURE  X(004).                  PJJPS1
       10       6-LI104    REDEFINES   6-LI100.                 PJJPS1
         15     FILLER         PICTURE  X(038).                  PJJPS1
         15     6-LI104-POURC  PICTURE ZZ9,99.                   PJJPS1
         15     FILLER         PICTURE  X(001).                  PJJPS1
       10       6-LI105    REDEFINES   6-LI100.                 PJJPS1
         15     FILLER         PICTURE  X(031).                  PJJPS1
         15     6-LI105-NOFICH PICTURE XX.                       PJJPS1
         15     FILLER         PICTURE  X(004).                  PJJPS1
         15     6-LI105-CPTENR PICTURE Z(3)9.                    PJJPS1
         15     FILLER         PICTURE  X(004).                  PJJPS1
       10       6-LI106    REDEFINES   6-LI100.                 PJJPS1
         15     6-LI106-ZLIB03 PICTURE 99999999999999.           PJJPS1
         15     FILLER         PICTURE  X(031).                  PJJPS1
01                 1-CD00.                                      PJJPS1
       10          1-CD00-NOCL.                                 PJJPS1
       11          1-CD00-NOCL11 PICTURE  X.                    PJJPS1
       11          1-CD00-NOCL12 PICTURE  XX.                   PJJPS1
       11          1-CD00-NOCL2  PICTURE  XX.                   PJJPS1
       10          1-CD00-QUCO   PICTURE  S9(5)V99              PJJPS1
                     COMPUTATIONAL-3.                           PJJPS1
01                 2-CD00.                                      PJJPS1
```

```
     10          2-CD00-NOCL.                                    PJJPS1
     11          2-CD00-NOCL11 PICTURE  X.                       PJJPS1
     11          2-CD00-NOCL12 PICTURE  XX.                      PJJPS1
     11          2-CD00-NOCL2  PICTURE  XX.                      PJJPS1
     10          2-CD00-QUCO   PICTURE  S9(5)V99                 PJJPS1
                   COMPUTATIONAL-3.                              PJJPS1
01               1-CL00.                                         PJJPS1
     10          1-CL00-KEYCI.                                   PJJPS1
     11          1-CL00-NOCL.                                    PJJPS1
     12          1-CL00-NOCL11 PICTURE  X.                       PJJPS1
     12          1-CL00-NOCL12 PICTURE  XX.                      PJJPS1
     12          1-CL00-NOCL2  PICTURE  XX.                      PJJPS1
     10          1-CL00-NOMCL  PICTURE  X(20).                   PJJPS1
     10          1-CL00-ADRES  PICTURE  X(43).                   PJJPS1
     10          1-CL00-NUDEP  PICTURE  XXX.                     PJJPS1
     10          1-CL00-LIDEP  PICTURE  X(24).                   PJJPS1
     10          1-CL00-NUREG  PICTURE  XXX.                     PJJPS1
     10          1-CL00-LIREG  PICTURE  X(24).                   PJJPS1
01               2-CL00.                                         PJJPS1
     10          2-CL00-KEYCI.                                   PJJPS1
     11          2-CL00-NOCL.                                    PJJPS1
     12          2-CL00-NOCL11 PICTURE  X.                       PJJPS1
     12          2-CL00-NOCL12 PICTURE  XX.                      PJJPS1
     12          2-CL00-NOCL2  PICTURE  XX.                      PJJPS1
     10          2-CL00-NOMCL  PICTURE  X(20).                   PJJPS1
     10          2-CL00-ADRES  PICTURE  X(43).                   PJJPS1
     10          2-CL00-NUDEP  PICTURE  XXX.                     PJJPS1
     10          2-CL00-LIDEP  PICTURE  X(24).                   PJJPS1
     10          2-CL00-NUREG  PICTURE  XXX.                     PJJPS1
     10          2-CL00-LIREG  PICTURE  X(24).                   PJJPS1
01               1-LV00.                                         PJJPS1
     10          1-LV00-NOCL.                                    PJJPS1
     11          1-LV00-NOCL11 PICTURE  X.                       PJJPS1
     11          1-LV00-NOCL12 PICTURE  XX.                      PJJPS1
     11          1-LV00-NOCL2  PICTURE  XX.                      PJJPS1
     10          1-LV00-NBLIV  PICTURE  9.                       PJJPS1
     10          1-LV00-QTLI   PICTURE  S9(5)V99                 PJJPS1
                   COMPUTATIONAL-3.                              PJJPS1
     10          1-LV00-GROUPE                                   PJJPS1
                   OCCURS       009                              PJJPS1
                   DEPENDING  ON      1-LV00-NBLIV.             PJJPS1
     11          1-LV00-QULI   PICTURE  S9(5)V99                 PJJPS1
                   COMPUTATIONAL-3.                              PJJPS1
     11          1-LV00-DALI   PICTURE  X(6).                    PJJPS1
01               2-LV00.                                         PJJPS1
     10          2-LV00-NOCL.                                    PJJPS1
     11          2-LV00-NOCL11 PICTURE  X.                       PJJPS1
     11          2-LV00-NOCL12 PICTURE  XX.                      PJJPS1
     11          2-LV00-NOCL2  PICTURE  XX.                      PJJPS1
     10          2-LV00-NBLIV  PICTURE  9.                       PJJPS1
     10          2-LV00-QTLI   PICTURE  S9(5)V99                 PJJPS1
                   COMPUTATIONAL-3.                              PJJPS1
     10          2-LV00-GROUPE                                   PJJPS1
                   OCCURS       009                              PJJPS1
                   DEPENDING  ON      2-LV00-NBLIV.             PJJPS1
```

```
     11          2-LV00-QULI   PICTURE  S9(5)V99                PJJPS1
                   COMPUTATIONAL-3.                              PJJPS1
     11          2-LV00-DALI   PICTURE  X(6).                    PJJPS1
01               1-MO-TABLE.                                     PJJPS1
    02           1-MO00T.                                        PJJPS1
     05          1-MO00 OCCURS               0012.               PJJPS1
     10          1-MO00-ANNUL  PICTURE  X.                       PJJPS1
     10          1-MO00-MOIS   PICTURE  99.                      PJJPS1
     10          1-MO00-LMOIS  PICTURE  X(9).                    PJJPS1
     10          1-MO00-FILLER PICTURE  X(68).                   PJJPS1
01               1-MV00.                                         PJJPS1
   05            1-MV00-00.                                      PJJPS1
     10          1-MV00-NOCL.                                    PJJPS1
     11          1-MV00-NOCL11 PICTURE  X.                       PJJPS1
     11          1-MV00-NOCL12 PICTURE  XX.                      PJJPS1
     11          1-MV00-NOCL2  PICTURE  XX.                      PJJPS1
     10          1-MV00-NUORD  PICTURE  X.                       PJJPS1
     10          1-MV00-CODMV  PICTURE  X.                       PJJPS1
     10          1-MV00-NUCAR  PICTURE  X.                       PJJPS1
   05            1-MV00-SUITE.                                   PJJPS1
     15       FILLER         PICTURE  X(00072).                  PJJPS1
01               1-MV01  REDEFINES    1-MV00.                    PJJPS1
     10       FILLER         PICTURE  X(00008).                  PJJPS1
     10          1-MV01-NOMCL  PICTURE  X(20).                   PJJPS1
     10          1-MV01-ADRES  PICTURE  X(43).                   PJJPS1
     10          1-MV01-NUDEP  PICTURE  XXX.                     PJJPS1
     10          1-MV01-FILLER PICTURE  X(6).                    PJJPS1
01               1-MV02  REDEFINES    1-MV00.                    PJJPS1
     10       FILLER         PICTURE  X(00008).                  PJJPS1
     10          1-MV02-MREEL9 PICTURE  9(5)V99.                 PJJPS1
     10          1-MV02-MREEL9X    REDEFINES                     PJJPS1
                 1-MV02-MREEL9  PICTURE X(007).                  PJJPS1
     10          1-MV02-DALI   PICTURE  X(6).                    PJJPS1
     10       FILLER         PICTURE  X(00059).                  PJJPS1
01               1-TD-TABLE.                                     PJJPS1
    02           1-TD01T.                                        PJJPS1
     05          1-TD01 OCCURS               0103.               PJJPS1
     10          1-TD01-NUDEP  PICTURE  XXX.                     PJJPS1
     10          1-TD01-LIDEP  PICTURE  X(24).                   PJJPS1
     10          1-TD01-NUREG  PICTURE  XXX.                     PJJPS1
    02           1-TD02T.                                        PJJPS1
     05          1-TD02 OCCURS               0016.               PJJPS1
     10          1-TD02-NUREG  PICTURE  XXX.                     PJJPS1
     10          1-TD02-LIREG  PICTURE  X(24).                   PJJPS1
01   USERS-AREAS       PICTURE X.                                PJJPS1
*SD: WB BIB: WG SEL: 01_____ FORM: I DESC: 2 NIV: 2 ORG: _ SS: _ 790020
01                 WB00.                                         PJJPS1
     02            WB01.                                         PJJPS1
     10            WB01-FILLER PICTURE  X(18)                    PJJPS1
                   VALUE      'CDCLDCENGLLCLVMVSE'.              PJJPS1
     10            WB01-FILLER PICTURE  X(4)                     PJJPS1
                   VALUE      'VLVM'.                            PJJPS1
     10            WB01-TABCPT PICTURE  X(44)                    PJJPS1
                   VALUE                 SPACE.                  PJJPS1
01                WB00-R REDEFINES WB00.                         791010
```

```
*SD: WC BIB: WG SEL: 0203____ FORM: I DESC: 3 NIV: 3 ORG: _ SS: _ 791020
      02            WC00.                                          PJJPS1
      03            WC02 OCCURS                0011.               PJJPS1
      10            WC02-NOFICH PICTURE  XX.                       PJJPS1
      03            WC03 OCCURS                0011.               PJJPS1
      10            WC03-CPTENR PICTURE  S9(7)                     PJJPS1
                    COMPUTATIONAL-3.                               PJJPS1
```

0A Declaratives

The F0A function contains one F0Aff function for each indexed file called in the -CD lines.

If the language generated is COBOL II ('D'), the function F0A90 contains the following clause:

'STOP RUN'

Otherwise, the clause is:

STOP 'INPUT-OUTPUT ERROR. CANCEL THE JOB'

```
   PROCEDURE DIVISION.                                             PJJPS1
   DECLARATIVES.                                                   PJJPS1
   SECMO SECTION.                                                  PJJPS1
       USE AFTER ERROR PROCEDURE ON   MO-FILE.                     PJJPS1
   F0AMO. DISPLAY  'STATUS : ENT01  = '  1-MO00-STATUS.            PJJPS1
   F0AMO-A. GO TO  F0A90.                                          PJJPS1
   F0AMO-FN. EXIT.                                                 PJJPS1
   F0A90.   STOP RUN.                                              PJJPS1
   F0A90-FN. EXIT.                                                 PJJPS1
   END DECLARATIVES.                                               PJJPS1
   SEC00 SECTION.                                                  P000
   N0DCA.    NOTE *APPEL DU TRI                        *.          P000
   F0DCA.                                                          P010
       SORT           MV-FILE                                      P020
       ON ASCENDING KEY                                            P110
       MV00-NOCL  MV00-NUORD                                       P120
       MV00-CODMV MV00-NUCAR                                       P500
       INPUT  PROCEDURE ENTREE                                     P510
       OUTPUT PROCEDURE SORTIE.                                    P900
       STOP RUN.                                                   P900
   F0DCA-FN. EXIT.                                                 P900
   ENTREE SECTION.                                                 P000
   N0F.      NOTE ************************************.           P000
                  *                                   *          P000
                  *PROCEDURE D'ENTREE                 *          P000
                  *                                   *          P000
                  ************************************.           P000
   F0F.           EXIT.                                           P000
   N0FBA.    NOTE *INITIALIZATION                      *.          P000
   F0FBA.                                                          P000
       OPEN INPUT     EN-FILE                                      P010
```

```
*PROCESSING DATE                                                    P080
     MOVE CURRENT-DATE TO DAT8                                      P100
     MOVE   DAT81 TO DATOM                                          P100
     MOVE   DAT82 TO DATOJ                                          P100
     MOVE   DAT83 TO DATOA                                          P100
     MOVE        DATCE                                             P110
     TO DAT8E DAT6C                                                 P110
     MOVE DAT81E TO DAT63C                                          P110
     MOVE DAT82E TO DAT61C  MOVE DAT83E TO DAT62C                   P110
     MOVE   DAT6C TO DATCE                                          P110
     MOVE        DATCE                                             P120
     TO DAT8E DAT6C                                                 P120
     MOVE DAT61C TO DAT81C  MOVE DAT62C TO DAT82C                   P120
     MOVE DAT63C TO DAT83C                                          P120
     MOVE   DAT8C TO  DAT8C.                                        P120
 F0FBA-FN. EXIT.                                                    P120
 N0FCA.    NOTE *TRAITEMENT FICHIER EN ENTREE       *.             P000
 F0FCA.    IF    EN-FT = 0                                        P000
           NEXT SENTENCE ELSE GO TO     F0FCA-FN.                   P000
     MOVE 0 TO IK                                                   P010
     READ           EN-FILE                                         P010
     AT END MOVE 1 TO IK.                                           P010
          IF    IK = 1                                            P020
     MOVE        1 TO EN-FT                                        P020
         GO TO     F0FCA-FN.                                      P030
     ADD         1 TO 5-EN00-RECCNT.                               P040
 N0FFF.    NOTE *DELIVERY DATE SELECTION            *.            P000
 F0FFF.    IF    EN00-NUCAR = 'B'                                  P000
           AND   EN02-DALI < DATOR                                 P020
           NEXT SENTENCE ELSE GO TO     F0FFF-FN.                  P020
         GO TO     F0FFF-FN.                                      P020
 N0FZA.    NOTE *ECRITURE                           *.            P000
 F0FZA.                                                           P000
     MOVE        EN00 TO MV00                                      P020
     MOVE 0 TO IK                                                  P030
     RELEASE     MV00.                                             P030
 F0FZA-FN. EXIT.                                                   P030
 F0FFF-FN. EXIT.                                                   P030
 F0FCA-900. GO TO F0FCA.                                           P030
 F0FCA-FN. EXIT.                                                   P030
 N0FZZ.    NOTE *FERMETURE                          *.            P000
 F0FZZ.                                                           P000
     CLOSE          EN-FILE.                                       P010
 F0FZZ-FN. EXIT.                                                   P010
 F0F-FN.   EXIT.                                                   P010
 SORTIE SECTION.                                                   P000
```

Initializations (F01)

Function F01 is always generated. Data structures defined as commentary (ORGANIZATION = 'X') are not described in this function. Data Structures described in WORKING-STORAGE or LINKAGE (ORGANIZATION = 'W' or 'L') are not described in F01, except those with USAGE = 'C', and control breaks. For these files, see the note below.

Primary purpose: Function F01 OPENs files, loads and CLOSEs table files.

Sub-functions: Each data structure is initialized in its own sub-function. The sub-function code is created using the DATA STRUCTURE CODE IN THE PROGRAM.

The sub-functions are generated in alphabetical order.

Each sub-function contains:

- the OPEN instruction for the data structure if its ORGANIZATION is 'S', 'I' or 'V', or 'W' or 'L' with control breaks.
- the prime READ instruction, for data structures with control break processing specified,
- the loading of the table files from the description in WORKING-STORAGE, if the ACCESS MODE is sequential, and if the USAGE OF DATA STRUCTURE = 'T' or 'X'. For these files, a CLOSE instruction is generated once the table is loaded.

NOTE

For input data structures (USAGE = 'C') described in WORKING STORAGE or LINKAGE (ORGANIZATION = 'W' or 'L'), with control breaks, an OPEN is generated followed by a PERFORM F95dd for the prime READ. It is the user's responsibility to code Sub-function F95dd, (normally using Procedural Code). This code may need to account for the end-of-processing and end-of-file indicators, as well as the OPEN and CLOSE of table files, etc.

```
N01.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *        INITIALIZATIONS             *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F01.      EXIT.                                                PJJPS1
N01CD.    NOTE *INITIALIZATION OF FILE  CD-FILE    *.          PJJPS1
F01CD.    OPEN INPUT                    CD-FILE.               PJJPS1
F01CD-10. READ     CD-FILE      AT END                         PJJPS1
          MOVE  1 TO                    CD-FI.                 PJJPS1
F01CD-FN. EXIT.                                                PJJPS1
N01CL.    NOTE *INITIALIZATION OF FILE  CL-FILE    *.          PJJPS1
F01CL.    OPEN INPUT                    CL-FILE.               PJJPS1
F01CL-10. READ     CL-FILE      AT END                         PJJPS1
          MOVE  1 TO                    CL-FI.                 PJJPS1
F01CL-FN. EXIT.                                                PJJPS1
N01DC.    NOTE *INITIALIZATION OF FILE  DC-FILE    *.          PJJPS1
F01DC.    OPEN OUTPUT                   DC-FILE.               PJJPS1
F01DC-FN. EXIT.                                                PJJPS1
N01ED.    NOTE *INITIALIZATION OF FILE  ED-FILE    *.          PJJPS1
F01ED.    OPEN OUTPUT                   ED-FILE.               PJJPS1
F01ED-FN. EXIT.                                                PJJPS1
N01GL.    NOTE *INITIALIZATION OF FILE  GL-FILE    *.          PJJPS1
```

```
F01GL.    OPEN INPUT                    GL-FILE.                PJJPS1
F01GL-FN. EXIT.                                                 PJJPS1
N01LC.    NOTE *INITIALIZATION OF FILE  LC-FILE    *.           PJJPS1
F01LC.    OPEN OUTPUT                   LC-FILE.                PJJPS1
F01LC-FN. EXIT.                                                 PJJPS1
N01LI.    NOTE *INITIALIZATION OF FILE  LI-FILE    *.           PJJPS1
F01LI.    OPEN OUTPUT                   LI-FILE.                PJJPS1
F01LI-FN. EXIT.                                                 PJJPS1
N01LV.    NOTE *INITIALIZATION OF FILE  LV-FILE    *.           PJJPS1
F01LV.    OPEN INPUT                    LV-FILE.                PJJPS1
F01LV-10. READ     LV-FILE      AT END                          PJJPS1
          MOVE  1 TO                    LV-FI.                  PJJPS1
F01LV-FN. EXIT.                                                 PJJPS1
N01MO.    NOTE *INITIALIZATION OF FILE  MO-FILE    *.           PJJPS1
F01MO.    OPEN INPUT                    MO-FILE.                PJJPS1
       IF        1-MO00-STATUS  NOT  = ZERO                     PJJPS1
       PERFORM    F0AMO                                        PJJPS1
       PERFORM    F0A90         THRU F0A90-FN.                 PJJPS1
F01MO-10. READ     MO-FILE      AT END                          PJJPS1
          GO TO   F01MO-20.                                     PJJPS1
         ADD 1 TO IMO00L      IF  IMO00L NOT > 0012            PJJPS1
          MOVE     MO00                                        PJJPS1
          TO     1-MO00          (IMO00L).                     PJJPS1
          GO TO   F01MO-10.                                     PJJPS1
F01MO-20.                                                       PJJPS1
          IF      IMO00L    >     IMO00M                      PJJPS1
          MOVE    IMO00L    TO    IMO00R                      PJJPS1
          MOVE    IMO00M    TO    IMO00L.                     PJJPS1
F01MO-99. CLOSE    MO-FILE.                                     PJJPS1
F01MO-FN. EXIT.                                                 PJJPS1
N01MV.    NOTE *INITIALIZATION OF FILE  MV-FILE    *.           PJJPS1
F01MV-10. RETURN   MV-FILE      AT END                          PJJPS1
          MOVE  1 TO                    MV-FI.                  PJJPS1
F01MV-FN. EXIT.                                                 PJJPS1
N01SE.    NOTE *INITIALIZATION OF FILE  SE-FILE    *.           PJJPS1
F01SE.    OPEN OUTPUT                   SE-FILE.                PJJPS1
F01SE-FN. EXIT.                                                 PJJPS1
N01TD.    NOTE *INITIALIZATION OF FILE  TD-FILE    *.           PJJPS1
F01TD.    OPEN INPUT                    TD-FILE.                PJJPS1
F01TD-10. READ     TD-FILE      AT END                          PJJPS1
          GO TO   F01TD-20.                                     PJJPS1
          IF       TD00-NOTAB  =     'D'                      PJJPS1
          NEXT SENTENCE     ELSE GO TO F01TD-01F.               PJJPS1
         ADD 1 TO ITD01L      IF  ITD01L NOT > 0103            PJJPS1
          MOVE     TD01-NUDEP   TO                            PJJPS1
                 1-TD01-NUDEP    (ITD01L)                       PJJPS1
          MOVE     TD01-LIDEP   TO                            PJJPS1
                 1-TD01-LIDEP    (ITD01L)                       PJJPS1
          MOVE     TD01-NUREG   TO                            PJJPS1
                 1-TD01-NUREG    (ITD01L)                       PJJPS1
          GO TO   F01TD-10.                                     PJJPS1
F01TD-01F.                                                      PJJPS1
          IF       TD00-NOTAB  =     'R'                      PJJPS1
          NEXT SENTENCE     ELSE GO TO F01TD-02F.               PJJPS1
         ADD 1 TO ITD02L      IF  ITD02L NOT > 0016            PJJPS1
```

```
          MOVE     TD02-NUREG   TO                              PJJPS1
                 1-TD02-NUREG    (ITD02L)                        PJJPS1
          MOVE     TD02-LIREG   TO                              PJJPS1
                 1-TD02-LIREG    (ITD02L)                        PJJPS1
          GO TO   F01TD-10.                                     PJJPS1
F01TD-02F.                                                      PJJPS1
          GO TO   F01TD-10.                                     PJJPS1
F01TD-20.                                                       PJJPS1
          IF      ITD01L    >     ITD01M                        PJJPS1
          MOVE    ITD01L    TO    ITD01R                        PJJPS1
          MOVE    ITD01M    TO    ITD01L.                       PJJPS1
          IF      ITD02L    >     ITD02M                        PJJPS1
          MOVE    ITD02L    TO    ITD02R                        PJJPS1
          MOVE    ITD02M    TO    ITD02L.                       PJJPS1
F01TD-99. CLOSE    TD-FILE.                                     PJJPS1
F01TD-FN. EXIT.                                                 PJJPS1
N01VL.    NOTE *INITIALIZATION OF FILE  VL-FILE    *.           PJJPS1
F01VL.    OPEN OUTPUT                   VL-FILE.               PJJPS1
F01VL-FN. EXIT.                                                 PJJPS1
N01VM.    NOTE *INITIALIZATION OF FILE  VM-FILE    *.           PJJPS1
F01VM.    OPEN OUTPUT                   VM-FILE.               PJJPS1
F01VM-FN. EXIT.                                                 PJJPS1
F01-FN.   EXIT.                                                 PJJPS1
```

Read sequential files with no control break (F05)

Function F05 is always generated, except in cases where the TYPE AND STRUCTURE OF PROGRAM selected does not generate the PROCEDURE DIVISION.

Primary purpose: Function F05 does the READ for all data structures without control breaks.

Special Note: Function F05 is the top of the iteration loop. Therefore it is important not to delete it, or if deleted, to insert the function number by other means.

Sub-functions: Each data structure without control breaks is given its own sub-function. The sub-function code is created using the DATA STRUCTURE CODE IN THE PROGRAM.

The data structures are read sequentially, (alphabetical order).

Each sub-function:

- contains the test giving access to the sub-function,
- contains the READ instruction,
- sets the end-of-processing indicator (dd-FT) AT END of READ,
- stores all data elements that make up the key for file matching, if a FILE MATCHING LEVEL NUMBER was entered (dd-IN-eeeeee),

- increments the record counter (5-dd00-RECCNT).

NOTE

For input data structures (USAGE = 'C') described in WORKING STORAGE or LINKAGE (ORGANIZATION = 'W' or 'L') without control breaks, the READ is generated as a PERFORM F95dd. It is the user's responsibility to code sub-function F95dd, normally using Procedural Code). This code may need to account for the end-of-processing and end-of-file indicators, as well as the OPEN and CLOSE of table files, etc.

```
*          NOTE *   BEGINNING OF PROGRAM ITERATION  *.           PJJPS1
 F05.           EXIT.                                             PJJPS1
 N05.       NOTE ************************************.           PJJPS1
                 *                                   *           PJJPS1
                 *READ SEQ.FILES NO CONTROL BREAK    *           PJJPS1
                 *                                   *           PJJPS1
                 ************************************.           PJJPS1
 N05GL.     NOTE *READ FILE              GL         *.           PJJPS1
 F05GL.     IF         FTB2   = '1'   AND GL-CF2 = '1'           PJJPS1
            NEXT SENTENCE ELSE GO TO     F05GL-FN.                PJJPS1
 F05GL-10. READ     GL-FILE      AT END                           PJJPS1
            MOVE  1 TO                    GL-FT                  PJJPS1
            MOVE HIGH-VALUE TO            GLIND                  PJJPS1
            GO TO   F05GL-FN.                                    PJJPS1
            MOVE     GL00-NOCL11 TO      GL-IN-NOCL11.           PJJPS1
            MOVE     GL00-NOCL12 TO      GL-IN-NOCL12.           PJJPS1
            ADD 1 TO 5-GL00-RECCNT.                              PJJPS1
 F05GL-FN. EXIT.                                                 PJJPS1
 F05-FN.   EXIT.                                                 PJJPS1
```

Read sequential files with control breaks (F10)

Function F10 is generated if there is at least one principal, consulted or transaction file (USAGE OF DATA STRUCTURE = 'P', 'C', 'M' or 'N') on which there is a control break.

Primary purpose: Function F10 MOVEs the prime read data from the read area to the work area, and then does a READ for next data in the read area.

Sub-functions: Each data structure with a control break is given its own sub-function. The sub-function code is created using the DATA STRUCTURE CODE IN THE PROGRAM.

The data structures are read sequentially, (alphabetical order).

Each sub-function:

- contains the test giving access to the subfunction, if a FILE MATCHING LEVEL NUMBER has been entered for the data structure,

- sets the initial control break variables (dd-IB),
- sets the end-of-processing indicator (dd-FT), if the end-of-file indicator (dd-FI) has been set,
- transfers 'OCCURS DEPENDING ON' counters, if they are in the common part ('00' segment) of the D.S.,
- transfers the read area data (dd00) to the work area (all file processing will be done in the work area),
- stores all data elements that make up the key for file matching if a FILE MATCHING LEVEL NUMBER was entered (dd-IN-eeeeee),
- increments the record counter (5-dd00-RECCNT),
- contains the READ instructions,
- sets end-of-file indicator (dd-FI), AT END.

NOTE

For data structures described in WORKING-STORAGE or LINKAGE, (ORGANIZATION = 'W' or 'L'), it is the user's responsibility to code the READ instruction. This is normally done by a PERFORM of sub-function F95dd, using Procedural Code. The code may need to account for the end-of-processing and end-of-file, as well as the OPEN and CLOSE of table files, etc.

```
N10.      NOTE ************************************.           PJJPS1
                *                                   *           PJJPS1
                *READ SEQ. FILES CONTROL BREAK      *           PJJPS1
                *                                   *           PJJPS1
                ************************************.           PJJPS1
F10.      EXIT.                                                 PJJPS1
N10CD.    NOTE *READ CONTROL BREAK      CD-FILE    *.           PJJPS1
F10CD.    IF         FTB3   = '1'   AND CD-CF3 = '1'           PJJPS1
          NEXT SENTENCE ELSE GO TO     F10CD-FN.                PJJPS1
F10CD-10. MOVE     CD-FB        TO      CD-IB.                 PJJPS1
          IF       CD-FI      = '1'                           PJJPS1
          MOVE HIGH-VALUE TO            CDIND                  PJJPS1
          MOVE 1 TO  CD-FT     GO TO F10CD-FN.                PJJPS1
          MOVE     CD00       TO      1-CD00.                 PJJPS1
          MOVE     CD00-NOCL11 TO      CD-IN-NOCL11            PJJPS1
          MOVE     CD00-NOCL12 TO      CD-IN-NOCL12            PJJPS1
          MOVE     CD00-NOCL2  TO      CD-IN-NOCL2             PJJPS1
          ADD 1 TO 5-CD00-RECCNT.                               PJJPS1
          READ     CD-FILE      AT END                         PJJPS1
          MOVE  1 TO                    CD-FI.                 PJJPS1
F10CD-FN. EXIT.                                                 PJJPS1
N10CL.    NOTE *READ CONTROL BREAK      CL-FILE    *.           PJJPS1
F10CL.    IF         FTB3   = '1'   AND CL-CF3 = '1'           PJJPS1
          NEXT SENTENCE ELSE GO TO     F10CL-FN.                PJJPS1
F10CL-10. MOVE     CL-FB        TO      CL-IB.                 PJJPS1
          IF       CL-FI      = '1'                           PJJPS1
          MOVE HIGH-VALUE TO            CLIND                  PJJPS1
          MOVE 1 TO  CL-FT     GO TO F10CL-FN.                PJJPS1
```

```
           MOVE     CL00       TO      1-CL00.                    PJJPS1
           MOVE     CL00-NOCL11 TO      CL-IN-NOCL11               PJJPS1
           MOVE     CL00-NOCL12 TO      CL-IN-NOCL12               PJJPS1
           MOVE     CL00-NOCL2  TO      CL-IN-NOCL2                PJJPS1
           ADD 1 TO 5-CL00-RECCNT.                                 PJJPS1
           READ     CL-FILE      AT END                             PJJPS1
           MOVE  1 TO                    CL-FI.                    PJJPS1
F10CL-FN.  EXIT.                                                   PJJPS1
N10LV.     NOTE *READ CONTROL BREAK      LV-FILE    *.             PJJPS1
F10LV.     IF         FTB3   = '1'   AND LV-CF3 = '1'              PJJPS1
           NEXT SENTENCE ELSE GO TO     F10LV-FN.                  PJJPS1
F10LV-10.  MOVE     LV-FB        TO      LV-IB.                    PJJPS1
           IF       LV-FI      = '1'                               PJJPS1
           MOVE HIGH-VALUE TO            LVIND                     PJJPS1
           MOVE 1 TO  LV-FT     GO TO F10LV-FN.                   PJJPS1
           MOVE     LV00-NBLIV  TO     1-LV00-NBLIV               PJJPS1
           MOVE     LV00       TO      1-LV00.                    PJJPS1
           MOVE     LV00-NOCL11 TO      LV-IN-NOCL11               PJJPS1
           MOVE     LV00-NOCL12 TO      LV-IN-NOCL12               PJJPS1
           MOVE     LV00-NOCL2  TO      LV-IN-NOCL2                PJJPS1
           ADD 1 TO 5-LV00-RECCNT.                                 PJJPS1
           READ     LV-FILE      AT END                             PJJPS1
           MOVE  1 TO                    LV-FI.                    PJJPS1
F10LV-FN.  EXIT.                                                   PJJPS1
N10MV.     NOTE *READ CONTROL BREAK      MV-FILE    *.             PJJPS1
F10MV.     IF       MV-CF3      = '1'                              PJJPS1
           NEXT SENTENCE ELSE GO TO     F10MV-FN.                  PJJPS1
F10MV-10.  MOVE     MV-FB        TO      MV-IB.                    PJJPS1
           IF       MV-FI      = '1'                               PJJPS1
           MOVE HIGH-VALUE TO            MVIND                     PJJPS1
           MOVE 1 TO  MV-FT     GO TO F10MV-FN.                   PJJPS1
           MOVE     MV00       TO      1-MV00.                    PJJPS1
           MOVE     MV00-NOCL11 TO      MV-IN-NOCL11               PJJPS1
           MOVE     MV00-NOCL12 TO      MV-IN-NOCL12               PJJPS1
           MOVE     MV00-NOCL2  TO      MV-IN-NOCL2                PJJPS1
           ADD 1 TO 5-MV00-RECCNT.                                 PJJPS1
           RETURN   MV-FILE      AT END                            PJJPS1
           MOVE  1 TO                    MV-FI.                    PJJPS1
F10MV-FN.  EXIT.                                                   PJJPS1
F10-FN.    EXIT.                                                   PJJPS1
```

End of run (F20)

Function F20 is always generated. The execution condition is that FT = ALL ’1’.

Primary purpose: Function F20 is used for closing files, and for the STOP RUN.

Sub-functions: Each data structure (other than those mentioned below) is given its own sub-function. The sub-function code is created using the DATA STRUCTURE CODE IN THE PROGRAM. A special Sub-function F2099 is generated for the STOP RUN instruction.

The data structures are closed sequentially according to their order on the Call of Data Structures (-CD) screen.

Each sub-function contains:

- the test giving access to the function,
- the CLOSE instruction for the data structure if its ORGANIZATION is S, I, or V, or W or L with control breaks.
- sub-function '99' contains the STOP RUN instruction if there is no sort data structure (FILE TYPE - INPUT / OUTPUT = 'T') in the program.

```
 N20.      NOTE ************************************.           PJJPS1
              *                                     *           PJJPS1
              *           END OF RUN                *           PJJPS1
              *                                     *           PJJPS1
              ************************************.           PJJPS1
 F20.      IF FT =            ALL '1'                          PJJPS1
           NEXT SENTENCE ELSE GO TO     F20-FN.                PJJPS1
 N20AA.    NOTE *END OF REPORTS                     *.         P000
 F20AA.                                                       P000
*UPDATE REPORT                                               P010
     PERFORM     F81 THRU F81-FN                            P100
*REPORT FOOTER                                               P150
     MOVE        5-ED00-3LCM TO 5-ED00-3LC                  P180
     PERFORM     F83IL THRU F83-FN.                         P200
 F20AA-FN. EXIT.                                             P200
 F20CD.    CLOSE    CD-FILE.                                 PJJPS1
 F20CD-FN. EXIT.                                             PJJPS1
 F20CL.    CLOSE    CL-FILE.                                 PJJPS1
 F20CL-FN. EXIT.                                             PJJPS1
 F20DC.    CLOSE    DC-FILE.                                 PJJPS1
 F20DC-FN. EXIT.                                             PJJPS1
 F20ED.    CLOSE    ED-FILE.                                 PJJPS1
 F20ED-FN. EXIT.                                             PJJPS1
 F20GL.    CLOSE    GL-FILE.                                 PJJPS1
 F20GL-FN. EXIT.                                             PJJPS1
 F20LC.    CLOSE    LC-FILE.                                 PJJPS1
 F20LC-FN. EXIT.                                             PJJPS1
 F20LI.    CLOSE    LI-FILE.                                 PJJPS1
 F20LI-FN. EXIT.                                             PJJPS1
 F20LV.    CLOSE    LV-FILE.                                 PJJPS1
 F20LV-FN. EXIT.                                             PJJPS1
 F20SE.    CLOSE    SE-FILE.                                 PJJPS1
 F20SE-FN. EXIT.                                             PJJPS1
 F20VL.    CLOSE    VL-FILE.                                 PJJPS1
 F20VL-FN. EXIT.                                             PJJPS1
 F20VM.    CLOSE    VM-FILE.                                 PJJPS1
 F20VM-FN. EXIT.                                             PJJPS1
 N2099.    NOTE *FIN PROGRAMME                      *.         P000
 F2099.                                                      P000
     GO TO F9999-FN.                                         P010
 F2099-FN. EXIT.                                             P010
 F20-FN.   EXIT.                                             P010
```

Calculate file control breaks (F22)

Function F22 is generated if there is at least one principal consulted or transaction file (USAGE OF DATA STRUCTURE = 'P', 'C', 'M' or 'N') on which there is a control break.

Primary purpose: Function F22 detects the next control break level by comparing key data in the work area to that in the read area.

Sub-functions: Each data structure with a control break is given its own sub-function. The sub-function code is created using the DATA STRUCTURE CODE IN THE PROGRAM.

The data structures are processed sequentially, in alphabetical order.

Each sub-function:

- Sets final control break variables (dd-FB) to zero,
- Calculates final control breaks, by comparing the values of the key fields in the read area to the corresponding values in the work area. This is done in the sequence of the data elements belonging to the SORT KEY field, from major to minor (1 to n) 'n' being the number entered for the NUMBER OF CONTROL BREAKS on the Call of Data Structures (-CD) screen,
- sets up the 'FTB' variable when the program does not contain file matching. In this case, FTB is used as dd-FB and has the same meaning,
- sets up the 'FBL' and 'IBL' variables, when the program does not contain file matching.

```
N22.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *CALCULATE FILE CONTROL BREAKS       *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F22.      EXIT.                                                PJJPS1
N22CD.    NOTE *CAL. CONTROL BREAK ON   CD-FILE    *.          PJJPS1
F22CD.    MOVE ZERO   TO                CD-FB.                 PJJPS1
          IF       CD-FI = '1' GO TO   F22CD-1.               PJJPS1
          IF       CD00-NOCL11 NOT =  1-CD00-NOCL11           PJJPS1
          GO  TO                       F22CD-1.              PJJPS1
          IF       CD00-NOCL12 NOT =  1-CD00-NOCL12           PJJPS1
          GO  TO                       F22CD-2.              PJJPS1
          GO TO   F22CD-FN.                                   PJJPS1
F22CD-1.  MOVE    1  TO                 CD-FB1.               PJJPS1
F22CD-2.  MOVE    1  TO                 CD-FB2.               PJJPS1
F22CD-FN. EXIT.                                               PJJPS1
N22CL.    NOTE *CAL. CONTROL BREAK ON   CL-FILE    *.          PJJPS1
F22CL.    MOVE ZERO   TO                CL-FB.                 PJJPS1
          IF       CL-FI = '1' GO TO   F22CL-1.               PJJPS1
          IF       CL00-NOCL11 NOT =  1-CL00-NOCL11           PJJPS1
          GO  TO                       F22CL-1.              PJJPS1
          IF       CL00-NOCL12 NOT =  1-CL00-NOCL12           PJJPS1
```

```
          GO  TO                       F22CL-2.                 PJJPS1
          GO TO   F22CL-FN.                                     PJJPS1
F22CL-1.  MOVE    1  TO                 CL-FB1.                  PJJPS1
F22CL-2.  MOVE    1  TO                 CL-FB2.                  PJJPS1
F22CL-FN. EXIT.                                                 PJJPS1
N22LV.    NOTE *CAL. CONTROL BREAK ON   LV-FILE    *.           PJJPS1
F22LV.    MOVE ZERO   TO                LV-FB.                   PJJPS1
          IF       LV-FI = '1' GO TO   F22LV-1.                 PJJPS1
          IF       LV00-NOCL11 NOT =  1-LV00-NOCL11             PJJPS1
          GO  TO                       F22LV-1.                 PJJPS1
          IF       LV00-NOCL12 NOT =  1-LV00-NOCL12             PJJPS1
          GO  TO                       F22LV-2.                 PJJPS1
          GO TO   F22LV-FN.                                     PJJPS1
F22LV-1.  MOVE    1  TO                 LV-FB1.                  PJJPS1
F22LV-2.  MOVE    1  TO                 LV-FB2.                  PJJPS1
F22LV-FN. EXIT.                                                 PJJPS1
N22MV.    NOTE *CAL. CONTROL BREAK ON   MV-FILE    *.           PJJPS1
F22MV.    MOVE ZERO   TO                MV-FB.                   PJJPS1
          IF       MV-FI = '1' GO TO   F22MV-1.                 PJJPS1
          IF       MV00-NOCL11 NOT =  1-MV00-NOCL11             PJJPS1
          GO  TO                       F22MV-1.                 PJJPS1
          IF       MV00-NOCL12 NOT =  1-MV00-NOCL12             PJJPS1
          GO  TO                       F22MV-2.                 PJJPS1
          IF       MV00-NOCL2  NOT =  1-MV00-NOCL2              PJJPS1
          GO  TO                       F22MV-3.                 PJJPS1
          IF       MV00-NUORD  NOT =  1-MV00-NUORD              PJJPS1
          GO  TO                       F22MV-4.                 PJJPS1
          IF       MV00-CODMV  NOT =  1-MV00-CODMV              PJJPS1
          GO  TO                       F22MV-5.                 PJJPS1
          IF       MV00-NUCAR  NOT =  1-MV00-NUCAR              PJJPS1
          GO  TO                       F22MV-6.                 PJJPS1
          GO TO   F22MV-FN.                                     PJJPS1
F22MV-1.  MOVE    1  TO                 MV-FB1.                  PJJPS1
F22MV-2.  MOVE    1  TO                 MV-FB2.                  PJJPS1
F22MV-3.  MOVE    1  TO                 MV-FB3.                  PJJPS1
F22MV-4.  MOVE    1  TO                 MV-FB4.                  PJJPS1
F22MV-5.  MOVE    1  TO                 MV-FB5.                  PJJPS1
F22MV-6.  MOVE    1  TO                 MV-FB6.                  PJJPS1
F22MV-FN. EXIT.                                                 PJJPS1
F22-FN.   EXIT.                                                 PJJPS1
```

File matching logic (F24)

Function F24 is generated if there is at least one input data structure on which there is file matching, or if there is one or more input(-output) principal data structure(s).

Primary purpose: Function F24 detects a new level of file matching. When the minor-most level has been attained, the work area is moved into the update area (1-dd00 --> 2-dd00).

Sub-functions: Each data structure with file matching is given its own sub-function. The sub-function code is created using the DATA STRUCTURE

CODE IN THE PROGRAM. In addition to those sub-functions, a numeric code is created based on the number of levels of file matching - one sub-function per level.

The sub-functions using the data structure code are generated in alphabetical order.

The alphabetic sub-functions will:

- set the Configuration Flag according to the current status of the file matching level (dd-CFn).

  The numeric sub-functions will:

- set the Occurrence Flag, once the file matching level processing has been completed (dd-OCn),
- at the minor-most level, for principal files, the work area is moved to the update area (1-dd00 --> 2-dd00).

```
N24.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *CAL.   CONFIGURATIONS   OCCURRENCES*           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F24.      MOVE  ZERO TO VCF  MOVE HIGH-VALUE TO IND.           PJJPS1
         IF TIND3 > CDIND      MOVE CDIND TO IND.              PJJPS1
         IF TIND3 > CLIND      MOVE CLIND TO IND.              PJJPS1
         IF TIND3 > LVIND      MOVE LVIND TO IND.              PJJPS1
         IF TIND3 > MVIND      MOVE MVIND TO IND.              PJJPS1
         IF TIND2 > GLIND      MOVE GLIND TO IND.              PJJPS1
F24CD.     IF       CDIND1     =          IND1                 PJJPS1
           MOVE    1  TO               CD-CF1.                PJJPS1
           IF       CDIND2     =          IND2                 PJJPS1
           MOVE     CD-CF1     TO       CD-CF2.                PJJPS1
           IF       CDIND3     =          IND3                 PJJPS1
           MOVE     CD-CF2     TO       CD-CF3.                PJJPS1
F24CD-FN. EXIT.                                               PJJPS1
F24CL.     IF       CLIND1     =          IND1                 PJJPS1
           MOVE    1  TO               CL-CF1.                PJJPS1
           IF       CLIND2     =          IND2                 PJJPS1
           MOVE     CL-CF1     TO       CL-CF2.                PJJPS1
           IF       CLIND3     =          IND3                 PJJPS1
           MOVE     CL-CF2     TO       CL-CF3.                PJJPS1
F24CL-FN. EXIT.                                               PJJPS1
F24GL.     IF       GLIND1     =          IND1                 PJJPS1
           MOVE    1  TO               GL-CF1.                PJJPS1
           IF       GLIND2     =          IND2                 PJJPS1
           MOVE     GL-CF1     TO       GL-CF2.                PJJPS1
F24GL-FN. EXIT.                                               PJJPS1
F24LV.     IF       LVIND1     =          IND1                 PJJPS1
           MOVE    1  TO               LV-CF1.                PJJPS1
           IF       LVIND2     =          IND2                 PJJPS1
           MOVE     LV-CF1     TO       LV-CF2.                PJJPS1
           IF       LVIND3     =          IND3                 PJJPS1
           MOVE     LV-CF2     TO       LV-CF3.                PJJPS1
```

```
F24LV-FN. EXIT.                                                         PJJPS1
F24MV.    IF       MVIND1     =           IND1                         PJJPS1
          MOVE    1  TO                 MV-CF1.                        PJJPS1
          IF       MVIND2     =           IND2                         PJJPS1
          MOVE     MV-CF1     TO        MV-CF2.                        PJJPS1
          IF       MVIND3     =           IND3                         PJJPS1
          MOVE     MV-CF2     TO        MV-CF3.                        PJJPS1
F24MV-FN. EXIT.                                                         PJJPS1
F2401.                                                                  PJJPS1
          IF         FTB1   = '1'                                       PJJPS1
          NEXT SENTENCE ELSE GO TO       F2401-FN.                      PJJPS1
          MOVE     CD-CF1     TO        CD-OC1.                        PJJPS1
          MOVE     CL-CF1     TO        CL-OC1.                        PJJPS1
          MOVE     LV-CF1     TO        LV-OC1.                        PJJPS1
F2401-FN. EXIT.                                                         PJJPS1
F2402.                                                                  PJJPS1
          IF         FTB2   = '1'                                       PJJPS1
          NEXT SENTENCE ELSE GO TO       F2402-FN.                      PJJPS1
          MOVE     CD-CF2     TO        CD-OC2.                        PJJPS1
          MOVE     CL-CF2     TO        CL-OC2.                        PJJPS1
          MOVE     LV-CF2     TO        LV-OC2.                        PJJPS1
F2402-FN. EXIT.                                                         PJJPS1
F2403.                                                                  PJJPS1
          IF         FTB3   = '1'                                       PJJPS1
          NEXT SENTENCE ELSE GO TO       F2403-FN.                      PJJPS1
          MOVE     CD-CF3     TO        CD-OC3.                        PJJPS1
          MOVE     CL-CF3     TO        CL-OC3.                        PJJPS1
          MOVE     LV-CF3     TO        LV-OC3.                        PJJPS1
          IF       CD-CF3    NOT = '1'                                 PJJPS1
          MOVE   SPACE          TO     2-CD00                         PJJPS1
          ELSE                                                          PJJPS1
          MOVE   1-CD00         TO     2-CD00.                        PJJPS1
          IF       CL-CF3    NOT = '1'                                 PJJPS1
          MOVE   SPACE          TO     2-CL00                         PJJPS1
          ELSE                                                          PJJPS1
          MOVE   1-CL00         TO     2-CL00.                        PJJPS1
          IF       LV-CF3    NOT = '1'                                 PJJPS1
          MOVE   SPACE          TO     2-LV00                         PJJPS1
          MOVE   ZERO           TO     2-LV00-NBLIV                   PJJPS1
          ELSE                                                          PJJPS1
          MOVE   1-LV00-NBLIV   TO     2-LV00-NBLIV                   PJJPS1
          MOVE   1-LV00         TO     2-LV00.                        PJJPS1
F2403-FN. EXIT.                                                         PJJPS1
F24-FN.   EXIT.                                                         PJJPS1
```

Total control break logic (F26)

Function F26 is generated if there is at least one principal, consulted or transaction file (USAGE OF DATA STRUCTURE = 'P', 'C', 'M' or 'N') with both control breaks and file matching.

Primary purpose: Function F26 detects when all processing on all files is complete, (the "total control break level"), and when the next READ on all files is ready to occur.

Sub-functions: none.

The Function will:

- set variables (ITB variables) indicating that a new cycle is about to begin on all files,
- set variables (FTB variables) to zero indicating that processing on the current set of data is ending,
- based on a series of tests (sequenced major to minor on the FILE MATCHING LEVEL NUMBER), calculate the level of total control breaks for the current iteration.

  This function cannot be altered in any way.

```
N26.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *CALCULATE TOTAL CONTROL BREAKS      *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F26.      MOVE FTB TO ITB. MOVE ZERO TO FTB.                    PJJPS1
          MOVE FBL TO IBL. MOVE ZERO TO FBL.                    PJJPS1
          IF      (CD-CF1   = '0' OR    CD-FB1 = '1'          PJJPS1
          AND      CD-CF3    = '1')                           PJJPS1
          IF      (CL-CF1   = '0' OR    CL-FB1 = '1'          PJJPS1
          AND      CL-CF3    = '1')                           PJJPS1
          IF      (LV-CF1   = '0' OR    LV-FB1 = '1'          PJJPS1
          AND      LV-CF3    = '1')                           PJJPS1
          IF      (MV-CF1   = '0' OR    MV-FB1 = '1'          PJJPS1
          AND      MV-CF3    = '1')                           PJJPS1
          MOVE          1   TO FBL GO TO F26-1.               PJJPS1
          IF      (CD-CF2   = '0' OR    CD-FB2 = '1'          PJJPS1
          AND      CD-CF3    = '1')                           PJJPS1
          IF      (CL-CF2   = '0' OR    CL-FB2 = '1'          PJJPS1
          AND      CL-CF3    = '1')                           PJJPS1
          IF      (LV-CF2   = '0' OR    LV-FB2 = '1'          PJJPS1
          AND      LV-CF3    = '1')                           PJJPS1
          IF      (MV-CF2   = '0' OR    MV-FB2 = '1'          PJJPS1
          AND      MV-CF3    = '1')                           PJJPS1
          MOVE          2   TO FBL GO TO F26-2.               PJJPS1
          IF       MV-CF3    = '0'  OR  MV-FB3 = '1'          PJJPS1
          MOVE          3   TO FBL GO TO F26-3.               PJJPS1
          IF       MV-CF3    = '0'  OR  MV-FB4 = '1'          PJJPS1
          MOVE          4   TO FBL GO TO F26-4.               PJJPS1
          IF       MV-CF3    = '0'  OR  MV-FB5 = '1'          PJJPS1
          MOVE          5   TO FBL GO TO F26-5.               PJJPS1
          IF       MV-CF3    = '0'  OR  MV-FB6 = '1'          PJJPS1
          MOVE          6   TO FBL GO TO F26-6.               PJJPS1
          GO TO     F26-FN.                                   PJJPS1
F26-1.    MOVE 1 TO  FTB1.                                    PJJPS1
F26-2.    MOVE 1 TO  FTB2.                                    PJJPS1
```

```
F26-3.    MOVE 1 TO  FTB3.                                        PJJPS1
F26-4.    MOVE 1 TO  FTB4.                                        PJJPS1
F26-5.    MOVE 1 TO  FTB5.                                        PJJPS1
F26-6.    MOVE 1 TO  FTB6.                                        PJJPS1
F26-FN.   EXIT.                                                   PJJPS1
```

Calculate validation variables (F30)

Function F30 is generated if there is an input transaction data structure (USAGE OF DATA STRUCTURE = 'M' or'N').

Primary purpose: Function F30 controls the initialization of the Error tables, as needed.

Sub-functions: none.

The Function contains:

.the test giving access to the function;

.the initialization of the error table fields:

A) For elements (DE-ERR and/or ER-PRR)

Source:

the error table from the transaction file with error fields detected (USAGE = 'E'), stored in PACBASE variable 'ENPR'.

Validation:

1. standard: direct initialization of DE-ERR,
2. reduced: initialization of ER-PRR and transfer into DE-ERR:
   ER-ID --> ID-ER
   ER-PR0 --> ER-00.

If the source is not as described above, the error table is initialized to zero;

B) For user-defined errors (UT-ERUT)

If ERUT is not a repeated data element:

1. using 'ERUT', if it is called into the transaction data structure (and selected in the RESERVED ERROR CODES IN TRANS. FILE field),
2. if not, initialized to zero;

C) For segments

For multi-record transaction processing, initialization of ″group″ variables:

According to the TRANSACTION CONTROL BREAK LEVEL indicator (dd-IBn), determine whether the transaction error table is being built, or if a new transaction cycle is beginning in this iteration:

1. If a new transaction cycle is beginning, set SE-ERR to zero,
2. If not, set SE-ERR from the error table contained on the record of the transaction file with error validations in the GRPR field;

For a new transaction cycle:

Initializing the ″group″ error variable (GR-ER): A new transaction cycle begins when all files match at the highest level (ITBn = ’1’ where n = highest FILE MATCHING LEVEL NUMBER).

This function cannot be altered in any way.

```
N30.      NOTE ************************************.          PJJPS1
               *                                    *          PJJPS1
               *  CALCULATE VALIDATION VARIABLES    *          PJJPS1
               *                                    *          PJJPS1
               ************************************.          PJJPS1
F30.      IF       MV-CF3   = '1'                             PJJPS1
          NEXT SENTENCE ELSE GO TO     F30-FN.                PJJPS1
          MOVE ZERO TO DE-ERR.                                PJJPS1
          MOVE ZERO TO UT-ERUT.                               PJJPS1
          IF       MV-IB5   = '1'                             PJJPS1
          MOVE ZERO TO SE-ERR MOVE 1 TO TR-ER.                PJJPS1
          IF         ITB3     = '1'                           PJJPS1
          MOVE 0 TO GR-ER.                                    PJJPS1
F30-FN.   EXIT.                                               PJJPS1
```

Identification validation (F33)

Function F33 is generated if the transaction d.s. contains an element to identify the record type or one for the action: (CODE / VALUE OF RECORD TYPE ELEMENT or CODE / VALUE OF ACTION CODE ELEMENT on the Segment Definition screen.)

Primary purpose: Function F33 checks to see if the value in the record type and action code fields is one of the values designated as valid. The presence of the segment is also detected.

```
Sub-functions: 'AA' for validation of the record type,
               'BB' for validation of the action code.
```

```
The Function contains:
```

- the test giving access to the function, if the minormost FILE MATCHING LEVEL NUMBER for the data Structure has been achieved;

- Sub-function F33AA: record type validation which:
  - assigns a rank to the record according to its type (i.e. the position of this record type in relation to all the records of the file) in index 'I01',
  - in the case of a reduced error validation initialized tialized by ENPR of the input D.S., transfer of ER-PRM into the part of DE-ERR corresponding to the record type (ER-NN),
  - sets the Identification Error indicator if the record type field does not contain one of the specified values (ID-ER = 5),
  - indicates record presence (via SE-ER (I01) = 1) if GRPR is not on the input data structure;
- Sub-function F33BB: Validation of the action, which:
  - assigns a rank to the action field value- (Create = 1; Modify = 2; Delete = 3; etc.), according to the value detected,
  - sets the Identification Error indicator if the action code field does not contain one of the specified values (ID-ER = 6).

```
N33.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *   IDENTIFICATION VALIDATION        *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F33.      IF       MV-CF3   = '1'                              PJJPS1
          NEXT SENTENCE ELSE GO TO     F33-FN.                 PJJPS1
F33AA.                                                         PJJPS1
          IF     1-MV00-NUCAR      = 'A'                      PJJPS1
          MOVE    'MV01'        TO   LE-FIENR                PJJPS1
          MOVE  001 TO I01    GO TO    F33AA-01.             PJJPS1
          IF     1-MV00-NUCAR      = 'B'                      PJJPS1
          MOVE    'MV02'        TO   LE-FIENR                PJJPS1
          MOVE  002 TO I01    GO TO    F33AA-01.             PJJPS1
          MOVE 5 TO ID-ER GO TO F33-FN.                        PJJPS1
F33AA-01. IF ID-ER = '0' MOVE 1 TO SE-ER (I01).                PJJPS1
F33AA-FN. EXIT.                                                PJJPS1
F33BB.                                                         PJJPS1
          IF     1-MV00-CODMV      = 'C'                      PJJPS1
          MOVE  1  TO  I02    GO TO    F33BB-FN.             PJJPS1
          IF     1-MV00-CODMV      = 'M'                      PJJPS1
          MOVE  2  TO  I02    GO TO    F33BB-FN.             PJJPS1
          IF     1-MV00-CODMV      = 'S'                      PJJPS1
          MOVE  3  TO  I02    GO TO    F33BB-FN.             PJJPS1
          IF     1-MV00-CODMV      = 'D'                      PJJPS1
          MOVE  4  TO  I02    GO TO    F33BB-FN.             PJJPS1
          IF     1-MV00-CODMV      = 'E'                      PJJPS1
          MOVE  5  TO  I02    GO TO    F33BB-FN.             PJJPS1
          IF     1-MV00-CODMV      = 'F'                      PJJPS1
          MOVE  6  TO  I02    GO TO    F33BB-FN.             PJJPS1
          MOVE 6 TO ID-ER.                                     PJJPS1
F33BB-FN. EXIT.                                                PJJPS1
F33-FN.   EXIT.                                                PJJPS1
```

Duplicate record validation (F36)

Function F36 is generated if the transaction file is to be validated in this program (USAGE OF DATA STRUCTURE = 'M'), if a control break has been specified, and also:

- either the record type element is part of the sort key and is the minor-most control break level,
- or the data structure has only one segment.

Primary purpose: Function F36 detects duplicate records.

Sub-functions: none.

The function contains:

- the test giving access to the function;
- the test to detect duplicate records, using dd-IBn and and dd-FBn, where n = the highest NUMBER OF CONTROL BREAKS (See also TRANSACTION CONTROL BREAK LEVEL);

  If a duplicate is detected,

- setting the Segment Error Indicator (SE-ER (I01) = 7).

This function cannot be altered in any way.

```
N36.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *    DUPLICATE RECORD VALIDATION     *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F36.      IF       MV-CF3   = '1'                              PJJPS1
          NEXT SENTENCE ELSE GO TO F36-FN.                     PJJPS1
          IF       MV-IB6   = '0' OR    MV-FB6 = '0'           PJJPS1
          MOVE  7  TO SE-ER (I01).                             PJJPS1
F36-FN.   EXIT.                                                PJJPS1
```

Presence of data elements (F39)

Function F39 is generated if there is a transaction data structure (USAGE OF DATA STRUCTURE = 'M' or 'N').

Primary purpose: Function F39 determines the status of each key data element, i.e., which are present and which are absent.

Sub-functions: Each different record type is given its own sub-function. The sub-function code is a number allocated by the system at generation time.

The function contains:

- the test giving access to the function:

There must be no identification error (i.e. ID-ER = 0) and if file matching has been specified, the record must be at the minor-most level of matching-(dd-CFn = 1 with n = FILE MATCHING LEVEL NUMBER);

- sub-functions which:
- test the record type value (according to values specified on the Segment Definition (S) screen),
- store pointers to the first and last data elements of the record in relation to the beginning of the record (in Index 'I03'),
- indicate the status of key data element presence using DE-ER(n) or ER-ss-eeeeee,

  The presence of a data element is detected by the fact that a value exists in the work area of the element. The test is done against blanks, zero or low-values, depending upon the option selected in the TYPE OF PRESENCE VALIDATION field on the Program Definition screen. This is only done for transactions without the error vector ENPR.

NOTE: The sub-functions are exclusive from one another.

```
N39.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *   PRESENCE OF DATA ELEMENTS        *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F39.      IF       MV-CF3   = '1'   AND  ID-ER = '0'           PJJPS1
          NEXT SENTENCE ELSE GO TO     F39-FN.                 PJJPS1
F3900.                                                         PJJPS1
          IF     1-MV00-NOCL11  NOT = BLANC                    PJJPS1
          MOVE  1  TO   ER-00-NOCL11.                          PJJPS1
          IF     1-MV00-NOCL12  NOT = BLANC                    PJJPS1
          MOVE  1  TO   ER-00-NOCL12.                          PJJPS1
          IF     1-MV00-NOCL2   NOT = BLANC                    PJJPS1
          MOVE  1  TO   ER-00-NOCL2.                           PJJPS1
          IF     1-MV00-NUORD   NOT = BLANC                    PJJPS1
          MOVE  1  TO   ER-00-NUORD.                           PJJPS1
          IF     1-MV00-CODMV   NOT = BLANC                    PJJPS1
          MOVE  1  TO   ER-00-CODMV.                           PJJPS1
          IF     1-MV00-NUCAR   NOT = BLANC                    PJJPS1
          MOVE  1  TO   ER-00-NUCAR.                           PJJPS1
F3900-FN. EXIT.                                                PJJPS1
F3901.                                                         PJJPS1
          IF     1-MV00-NUCAR      = 'A'                       PJJPS1
          NEXT SENTENCE ELSE GO TO     F3901-FN.               PJJPS1
          MOVE  007   TO          I03.                         PJJPS1
          IF     1-MV01-NOMCL   NOT = BLANC                    PJJPS1
          MOVE  1  TO   ER-01-NOMCL.                           PJJPS1
          IF     1-MV01-ADRES   NOT = BLANC                    PJJPS1
          MOVE  1  TO   ER-01-ADRES.                           PJJPS1
          IF     1-MV01-NUDEP   NOT = BLANC                    PJJPS1
          MOVE  1  TO   ER-01-NUDEP.                           PJJPS1
          MOVE  009   TO          I04.                         PJJPS1
          GO TO   F39-FN.                                      PJJPS1
```

```
F3901-FN. EXIT.                                                    PJJPS1
F3902.                                                             PJJPS1
          IF     1-MV00-NUCAR      = 'B'                         PJJPS1
          NEXT SENTENCE ELSE GO TO     F3902-FN.                   PJJPS1
          MOVE  010   TO          I03.                           PJJPS1
          IF     1-MV02-MREEL9X NOT = BLANC                      PJJPS1
          MOVE  1  TO   ER-02-MREEL9.                            PJJPS1
          IF     1-MV02-DALI    NOT = BLANC                      PJJPS1
          MOVE  1  TO   ER-02-DALI.                              PJJPS1
          MOVE  011   TO          I04.                           PJJPS1
          GO TO   F39-FN.                                        PJJPS1
F3902-FN. EXIT.                                                    PJJPS1
F39-FN.   EXIT.                                                    PJJPS1
```

Record structure validation (F42)

Function F42 is generated if the transaction d.s. is to be validated (USAGE OF DATA STRUCTURES = 'M').

```
Primary purpose: Function F42 evaluates whether the key data
                 elements are erroneously present or absent.

Sub-functions: '10' to validate data elements in the common
                    part segment,

               '20' to validate data elements in the
                specific part segments.
```

The function contains:

- the test giving access to the function:

  There must be no identification error (ID-ER = 0) and the record on the transaction file must participate in this iteration (dd-CFn = 1). The latter test is done only if file matching has been specified;
- Sub-function F4210, which checks whether a data element of the common part should be present or absent, according to the specifications entered on the segment Call of Elements (-CE) screen. If an error is detected, DEL-ER takes on the following values:

  2 = invalid absence,

  3 = invalid presence;
- Sub-function F4220, (if the file has more than one record type), which checks whether a data element of a specific part segment should be present or absent. If an error is detected, DEL-ER takes on the same values as mentioned above.

```
N42.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *    RECORD  STRUCTURE VALIDATION    *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F42.      IF       MV-CF3   = '1'   AND  ID-ER = '0'           PJJPS1
          NEXT SENTENCE ELSE GO TO     F42-FN.                 PJJPS1
F4210.    MOVE  1             TO  I06.                         PJJPS1
```

```
F4210-010. MOVE DE-TT (I06, I02) TO DE-TTE.                    PJJPS1
     IF  DE-TTE  =  'F'     GO TO F4210-090.                   PJJPS1
     MOVE  DE-ER  (I06)  TO  DEL-ER.                           PJJPS1
    IF DE-TTE = 'O' AND DEL-ER = '0' MOVE 2 TO DEL-ER.         PJJPS1
    IF DE-TTE = 'I' AND DEL-ER = '1' MOVE 3 TO DEL-ER.         PJJPS1
     MOVE  DEL-ER  TO  DE-ER  (I06).                           PJJPS1
F4210-090. IF I06 < I50 ADD 1 TO I06 GO TO F4210-010.          PJJPS1
F4210-FN. EXIT.                                                PJJPS1
F4220.    MOVE    I03         TO  I06.                         PJJPS1
F4220-010. MOVE DE-TT (I06, I02) TO DE-TTE.                    PJJPS1
     IF  DE-TTE  =  'F'     GO TO F4220-090.                   PJJPS1
     MOVE  DE-ER  (I06)  TO  DEL-ER.                           PJJPS1
    IF DE-TTE = 'O' AND DEL-ER = '0' MOVE 2 TO DEL-ER.         PJJPS1
    IF DE-TTE = 'I' AND DEL-ER = '1' MOVE 3 TO DEL-ER.         PJJPS1
     MOVE  DEL-ER  TO  DE-ER  (I06).                           PJJPS1
F4220-090. IF I06 < I04 ADD 1 TO I06 GO TO F4220-010.          PJJPS1
F4220-FN. EXIT.                                                PJJPS1
F42-FN.   EXIT.                                                PJJPS1
```

Data element contents validation (F45)

Function F45 is generated if the transaction d.s. is to be validated (USAGE OF DATA STRUCTURE = 'M').

Primary purpose: Function F45 checks the values in the key fields for valid class and contents.

Sub-functions: Each record type is given its own sub-function. The sub-function code is a number allocated by the system at generation time.

The function contains:

- the test giving access to the function:

  There must be no identification error (ID-ER = 0) and if file matching has been specified, the record on the transaction file participates in this iteration (dd-CFn = 1);
- The sub-functions are executed according to the value detected in the record type field. They are therefore exclusive from one another. If there are contents validations specified for data elements of the record type, (see DATA ELEMENT CONTENTS VALIDATIONS), each sub-function contains:
  - the test verifying the valid presence of this data element and its status of being error-free (ER-ss-eeeeee = 1),
  - class validation, if specified, can be:
    - purely numeric,
    - alphabetic with spaces,
    - numeric with spaces to the left,
    - numeric with spaces to the left or right,

Failure results in ER-ss-eeeeee = 4,

- contents validation, if specified, can:
  - check that the data element has (or does not have) some specified value(s),
  - check that the data element is within a given range(s),
  - check that the contents of data element are in a table accessed sequentially,
  - check that the contents correspond to a set of codes given on the Data Element Description (-D) screen,

    Failure results in ER-ss-eeeeeee = 4,
- if one of the types of validations specified for a data element is a PERFORM of a sub-function it is executed before or after the content validation depending upon the sequence in which it was entered on the Call of Elements (-CE) screen. (The sequence is determined by the LINE NUMBER value),

  If it precedes the class/contents validations, the PERFORM is executed only if the data element is present and still error free,

  If it follows the class/contents validations, the PERFORM is executed only if an error in the contents HAS been detected. This being the case the user must fill in the corresponding DE-ERR entity,

  The PERFORM statement is never executed, after a Table validation.

```
N45.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               * DATA ELEMENT  CONTENTS VALIDATION *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F45.      IF       MV-CF3   = '1'   AND  ID-ER = '0'           PJJPS1
          NEXT SENTENCE ELSE GO TO     F45-FN.                 PJJPS1
F4500.                                                         PJJPS1
       IF   ER-00-NOCL2   NOT  =  '1'                          PJJPS1
                            GO TO F4500-003.                   PJJPS1
       IF        1-MV00-NOCL2  NOT NUMERIC                     PJJPS1
       MOVE 4 TO  ER-00-NOCL2   GO  TO F4500-003.              PJJPS1
F4500-003.                                                     PJJPS1
       IF   ER-00-NUORD   NOT  =  '1'                          PJJPS1
                            GO TO F4500-004.                   PJJPS1
       IF        1-MV00-NUORD  NOT NUMERIC                     PJJPS1
       MOVE 4 TO  ER-00-NUORD   GO  TO F4500-004.              PJJPS1
       IF        1-MV00-NUORD   NOT  <   '1'                   PJJPS1
       AND       1-MV00-NUORD   NOT  >       '8'               PJJPS1
       OR        1-MV00-NUORD       =  '9'                    PJJPS1
       GO  TO     F4500-004.                                  PJJPS1
F4500-004C.                 MOVE 5 TO ER-00-NUORD.             PJJPS1
F4500-004.     EXIT.                                           PJJPS1
F4500-FN. EXIT.                                                PJJPS1
F4501.                                                         PJJPS1
          IF     1-MV00-NUCAR      = 'A'                      PJJPS1
          NEXT SENTENCE ELSE GO TO     F4501-FN.              PJJPS1
```

```
       IF   ER-01-NOMCL   NOT  =  '1'                          PJJPS1
                                  GO TO F4501-007.             PJJPS1
       IF        1-MV01-NOMCL  NOT ALPHABETIC                  PJJPS1
       MOVE 4 TO  ER-01-NOMCL   GO  TO F4501-007.              PJJPS1
F4501-007.                                                     PJJPS1
       IF   ER-01-NUDEP   NOT  =  '1'                          PJJPS1
                                  GO TO F4501-009.             PJJPS1
       MOVE 1  TO ITD01R.                                      PJJPS1
F4501-009A. IF    ITD01R     >    ITD01L                       PJJPS1
       MOVE 5  TO ER-01-NUDEP    GO TO F4501-009.             PJJPS1
       IF        1-TD01-NUDEP    (ITD01R) =                   PJJPS1
                 1-MV01-NUDEP    GO TO F4501-009.             PJJPS1
       ADD 1   TO ITD01R.        GO TO F4501-009A.            PJJPS1
F4501-009.                                                     PJJPS1
          GO TO   F45-FN.                                      PJJPS1
F4501-FN. EXIT.                                                PJJPS1
F45-FN.   EXIT.                                                PJJPS1
```

Record presence validation (F51)

Function F51 is generated if the transaction d.s. is to be validated in the program (USAGE OF DATA STRUCTURE = 'M'), and if it contains more than one record type.

```
Primary purpose: Function F51 detects an erroneous absence
                 or presence of a segment.

Sub-functions: '10' to detect invalid absence of a segment,

               '20' to detect invalid presence of a segment.

The function contains:
```

- the test giving access to the function:

  There must be no identification error (ID-ER = 0) and if file matching has been specified, the record on the transaction file participates in this iteration (dd-CFn = 1);
- Sub-function F5110 which verifies that the record is supposed to be present for this transaction (Segment Definition screen SEGMENT PRESENCE specifications), and if not, identifies the error: (SE-ER (I01) = 3);
- Sub-function F5120 is executed only when the minor- most TRANSACTION CONTROL BREAK LEVEL has been (dd-FBn = 1). This sub-function verifies that achieved all records needed for this transaction are present, and if not, flags the error for that particular record (SE-ER (I06) = 2 with I06 as the index specifying the record) and the transaction (TR-ER = 2).

```
N51.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *    RECORD PRESENCE VALIDATION      *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F51.       IF       MV-CF3   = '1'  AND ID-ER = '0'            PJJPS1
       NEXT SENTENCE ELSE        GO TO F51-FN.                 PJJPS1
```

```
F5110. IF SE-ER (I01) = '1'                                          PJJPS1
    AND SE-TT (I01, I02) = 'I' MOVE 3 TO SE-ER (I01).                PJJPS1
F5110-FN. EXIT.                                                      PJJPS1
F5120. IF          MV-FB5   = '1'                                    PJJPS1
       NEXT SENTENCE ELSE        GO TO F51-FN.                       PJJPS1
       MOVE 1 TO I06.                                                PJJPS1
F5120-010.                                                           PJJPS1
     IF SE-ER (I06) = '0' AND SE-TT (I06, I02) = 'O'                 PJJPS1
       MOVE 2 TO SE-ER (I06) MOVE 2 TO TR-ER.                        PJJPS1
       IF I06 < 002  ADD 1 TO I06 GO TO F5120-010.                   PJJPS1
F5120-FN. EXIT.                                                      PJJPS1
F51-FN.   EXIT.                                                      PJJPS1
```

Existence validation (F70)

Function F70 is generated if a transaction d.s. (USAGE OF DATA STRUCTURE = 'M' or 'N') contains data elements that update one or more Principal d.s.'s (USAGE = 'P') accessed in program.

Primary purpose: Function F70 evaluates the compatibility of the intended action with the status of segment presence or absence.

Sub-functions: Each principal data structure to be updated is given its own sub-function. The sub-function code is created using the DATA STRUCTURE CODE IN THE PROGRAM.

The function contains:

- the condition test giving access to the function:

  There must be no identification error (ID-ER = 0) and if file matching has been specified, the record on the transaction file participates in this iteration (dd-CFn = 1) and a new transaction cycle is beginning ginning (dd-IBn = 1 where n = the minor-most TRANSACTION CONTROL BREAK LEVEL specified);
- Each sub-function contains:
  - the test for erroneous existence on the principal file of a record to be created,
  - if detected, SE-ER (I01) = 8,
  - the test for erroneous absence on the principal file of a record to be deleted or modified,
  - if detected, SE-ER (I01) = 9.

```
N70.      NOTE ************************************.              PJJPS1
               *                                    *              PJJPS1
               *CORRESPONDENCE VALIDATION           *              PJJPS1
               *                                    *              PJJPS1
               ************************************.              PJJPS1
F70.      IF       MV-CF3   = '1'   AND  ID-ER = '0'              PJJPS1
          NEXT SENTENCE ELSE GO TO     F70-FN.                     PJJPS1
```

```
          IF       MV-IB5     = '1'                              PJJPS1
          NEXT SENTENCE ELSE GO TO F70-FN.                       PJJPS1
N70CD.    NOTE *CORRESPONDENCE VALID.  FILE  CD    *.           PJJPS1
F70CD.    IF I02      = 1 AND          CD-OC3 = '1'              PJJPS1
          MOVE  8  TO SE-ER      (I01).                          PJJPS1
          IF I02 NOT  = 1 AND          CD-OC3 = '0'              PJJPS1
          MOVE  9  TO SE-ER      (I01).                          PJJPS1
F70CD-FN. EXIT.                                                  PJJPS1
N70CL.    NOTE *CORRESPONDENCE VALID.  FILE  CL    *.           PJJPS1
F70CL.    IF I02      = 1 AND          CL-OC3 = '1'              PJJPS1
          MOVE  8  TO SE-ER      (I01).                          PJJPS1
          IF I02 NOT  = 1 AND          CL-OC3 = '0'              PJJPS1
          MOVE  9  TO SE-ER      (I01).                          PJJPS1
F70CL-FN. EXIT.                                                  PJJPS1
N70LV.    NOTE *CORRESPONDENCE VALID.  FILE  LV    *.           PJJPS1
F70LV.    IF I02      = 1 AND          LV-OC3 = '1'              PJJPS1
          MOVE  8  TO SE-ER      (I01).                          PJJPS1
          IF I02 NOT  = 1 AND          LV-OC3 = '0'              PJJPS1
          MOVE  9  TO SE-ER      (I01).                          PJJPS1
F70LV-FN. EXIT.                                                  PJJPS1
F70-FN.   EXIT.                                                  PJJPS1
```

Update (F73)

Function F73 is generated if a transaction d.s. has at least one data element that updates at least one data element of a Principal Data Structure in this program.

Primary purpose: Function F73 updates the principal file.

Note: A transaction record may be used to update more than one principal file, or conversely, a single principal file may be updated by more than one transaction record. Each occurrence of one transaction and one principal file shall be referred to as a ″record pair″.

Sub-functions: There is one sub-function for each Principal- Transaction record pair. The sub-function code is allocated by the system at generation time.

The function contains:

- the test giving access to the function:

  There must be no identification error (ID-ER = 0) and if file matching has been specified, the record on the transaction file participates in this iteration (dd-CFn = 1) and a new transaction cycle is beginning, (dd-IBn = 1, where n = the minor-most TRANSACTION CONTROL BREAK LEVEL specified);
- two types of sub-functions:

  1. Update the common part segment of the principal file:

The Occurrence variable at the minor-most control break level on the principal file (dd-OCn) is set to 1 or 0, depending upon whether a record is being created or deleted;

2. Update the specific part segments (non-'00'):

These sub-functions are conditioned by a test on the SEGMENT CODE of the record concerned;

- in both sub-function types, the update is carried out data element by data element, as specified on transaction file Call of Elements (-CE) screen (see TYPE: VALIDATION, UPDATE, VALUES):
  - with unconditional replacement of a data element in the principal file by the corresponding transaction file data element (MOVE),
  - with replacement, addition or subtraction conditioned by the fact that the transaction file data element is present and error-free.

```
N73.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *          UPDATE                    *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F73.      IF       MV-CF3   = '1'  AND ID-ER = '0'             PJJPS1
          NEXT SENTENCE ELSE GO TO     F73-FN.                 PJJPS1
          IF SE-ER (I01) NOT = '1' GO TO F73-FN.               PJJPS1
N7301.    NOTE *   UPDATE   OF  LV 00 BY  MV 00    *.          PJJPS1
F7301.                                                         PJJPS1
          IF      I02 = 3 MOVE 0 TO     LV-OC3                 PJJPS1
          GO TO                        F7301-FN.               PJJPS1
          IF      I02 = 1 MOVE 1 TO     LV-OC3.                PJJPS1
          MOVE   1-MV00-NOCL   TO     2-LV00-NOCL.             PJJPS1
F7301-FN. EXIT.                                                PJJPS1
F73-FN.   EXIT.                                                PJJPS1
```

Store errors and backout (F76)

Function F76 is generated if there is a transaction file in this program.

Primary purpose: Function F76 detects errors found in various validations and marks bad transactions (TR-ER), and/or bad group transactions (GR-ER). If an error has been detected, a backout procedure retrieves the initial state of the principal file.

Sub-functions: There is one sub-function generated for each Principal data structure (USAGE OF DATA STRUCTURE = 'P') to be updated. The sub-function code is created using the DATA STRUCTURE CODE IN THE PROGRAM of the Principal D.S.

The function contains:

- the condition test giving access to the function:

The record on the transaction d.s. must participate in this iteration (dd-CFn = 1).

- if there is an identification error, (ID-ER), mark the transaction (TR-ER),
- if there is an erroneous record, (SE-ER (I01)), mark the transaction (TR-ER),
- if there are any errors detected on data elements of a particular record, (DE-ER (I06)), mark the transaction (TR-ER = 4),
- if any user errors have been detected (UT-ERUT), mark the transaction (TR-ER). Note: this is true when the data element 'ERUT' has been called into a transaction d.s. (USAGE OF DATA STRUCTURE = 'M', 'N' or 'E') and that it does not have an OCCURS clause,
- if the transaction has been marked as bad, the group error indicator is also marked (GR-ER = 1),
- if no reserved data element was selected, (see RESERVED ERROR CODES IN TRANS. FILE field on the Call of Data Structures (-CD) screen), and if the program calls for an update report D.S., set up the output area, (see Function F90 for other conditions),
- Each sub-function contains:
  - the condition test for the file matching level, (FTBn = 1 with n = highest file matching level),
  - the condition test for the detection of an error on the transaction group (GR-ER = 1),

    If both conditions are true, the data structure is restored to its original state. This is done by the re-initialization of the Occurrence variable (dd-OCn) from the Configuration variable (dd-CFn) and if necessary, the transfer of the work area to the update area.

```
N76.      NOTE ************************************.           PJJPS1
                *                                   *           PJJPS1
                * STORE ERRORS, RETRIEVE INIT. STATE*           PJJPS1
                *                                   *           PJJPS1
                ************************************.           PJJPS1
F76.      IF       MV-CF3     = '1'                             PJJPS1
          NEXT SENTENCE ELSE GO TO     F76-FN.                  PJJPS1
N76-A.    NOTE *      STORE ERRORS                 *.           PJJPS1
F76-A.    IF ID-ER NOT = '0' MOVE ID-ER TO TR-ER                PJJPS1
          GO TO F76-C. MOVE SE-ER (I01) TO SEG-ER.              PJJPS1
          IF SEG-ER < '0' OR SEG-ER > '1'                       PJJPS1
          MOVE SEG-ER TO TR-ER GO TO F76-C.                     PJJPS1
          MOVE 1 TO I06.                                        PJJPS1
F76-B.    MOVE DE-ER (I06) TO DEL-ER.                           PJJPS1
     IF DEL-ER = '1'  OR DEL-ER = '0' GO TO F76-B1.             PJJPS1
          MOVE 4 TO TR-ER GO TO F76-C.                          PJJPS1
F76-B1.   IF I06 = I50 MOVE I03 TO I06 GO TO F76-B.             PJJPS1
          IF I06 < I04 ADD  1   TO I06 GO TO F76-B.             PJJPS1
F76-B2.   IF UT-ERUT NOT = ZERO MOVE 4 TO TR-ER.                PJJPS1
F76-C.    IF TR-ER NOT = '1' MOVE '1' TO GR-ER.                 PJJPS1
N76CD.    NOTE *RETRIEVE INITIAL STATE ON CD-FILE  *.           PJJPS1
F76CD.    IF         FTB3     = '1'                            PJJPS1
```

```
                AND  GR-ER    = '1'                                PJJPS1
           NEXT SENTENCE ELSE GO TO     F76CD-FN.                  PJJPS1
           MOVE     CD-CF3     TO        CD-OC3.                   PJJPS1
           IF       CD-CF3     = '1'                               PJJPS1
           MOVE   1-CD00       TO      2-CD00.                    PJJPS1
F76CD-FN.  EXIT.                                                   PJJPS1
N76CL.     NOTE *RETRIEVE INITIAL STATE ON CL-FILE  *.             PJJPS1
F76CL.     IF         FTB3     = '1'                               PJJPS1
                AND  GR-ER    = '1'                                PJJPS1
           NEXT SENTENCE ELSE GO TO     F76CL-FN.                  PJJPS1
           MOVE     CL-CF3     TO        CL-OC3.                   PJJPS1
           IF       CL-CF3     = '1'                               PJJPS1
           MOVE   1-CL00       TO      2-CL00.                    PJJPS1
F76CL-FN.  EXIT.                                                   PJJPS1
N76LV.     NOTE *RETRIEVE INITIAL STATE ON LV-FILE  *.             PJJPS1
F76LV.     IF         FTB3     = '1'                               PJJPS1
                AND  GR-ER    = '1'                                PJJPS1
           NEXT SENTENCE ELSE GO TO     F76LV-FN.                  PJJPS1
           MOVE     LV-CF3     TO        LV-OC3.                   PJJPS1
           IF       LV-CF3     = '1'                               PJJPS1
           MOVE   1-LV00-NBLIV  TO     2-LV00-NBLIV                PJJPS1
           MOVE   1-LV00       TO      2-LV00.                    PJJPS1
F76LV-FN.  EXIT.                                                   PJJPS1
F76-FN.    EXIT.                                                   PJJPS1
```

Report logic (F8r)

Function F8r is generated if there is a Print d.s. (USAGE OF DATA STRUCTURE = 'I' or 'J').

NOTE: The Function Code is created using the LAST CHARACTER OF REPORT CODE for the last character of the function code (replacing the 'r' of F8r).

Primary purpose: Function F8r controls the printing of reports. This includes moving the contents line to the output area, computing totals, moving the variable values, keeping track of the line counters, etc.

Sub-functions: One sub-function per Report Category to be printed, plus one sub-function per Report Structure is generated. The sub-function code is created using the alphabetic CATEGORY OF REPORT value, and the numeric STRUCTURE NUMBER values respectively.

The function contains:

- the condition for printing the report as defined by the user on the Report Description (-D) screen (Top);
- a sub-function per category, containing:
  - the condition for printing the category, as defined by the user on the body of the Report Description screen,

- the update of the line counter (5-dd00-1LC),
- depending upon the value entered in the NO. OF INSTANCES IN CATEGORY TABLE, either:
  1. loading the category code into the category table (CAT (J00)), or
  2. the direct printing of each line of the category (via a PERFORM of sub-function 'ZZ' - detailed explanation will follow),

If the category is repetitive (TYPE OF LINE IN REPORT = 'I'), its loading, or calling its lines to print, is done in a loop controlled by an index (Jddrcc). If a page overflow is detected when the table is being loaded, the top-of-page and end-of-page categories are automatically printed,

Since each iteration of the repetitive category loop causes an additional entry in the category table, the user must ensure that the total number of categories to be printed is less than (or equal to) the NO. OF INSTANCES IN CATEGORY TABLE (default = 100),

If there is totaling, the following paragraphs are generated:

- 090: puts zero in accumulators up to the highest initial control break level detected in this iteration (IBL),
- 150: loads the category if the condition is satisfied (generated if TYPE OF LINE IN REPORT = '*') and adds source data elements into the accumulators at the major-most level,
- 200 and 300: add accumulators of the major-most level to those at the next level, up to the minor-most final control break level detected in the iteration (FBL),
- Sub-function 'F8rZZ', which determines the next line to be printed and loads the information (STRUCTURE NUMBER, CONSTANT PART NUMBER, SKIP, etc.), necessary for printing this line;

  For direct printing, the loading is done for each line at the category level, and sub-function 'F8rZZ' begins by an unconditional skip to the end of function F8r;

  This Sub-function is the link for printing. Depending on the USAGE value, it contains:

  - Paragraph 005 which moves data on each category into the Structure table (ST-TA),
  - Paragraph 010 which:
    - resets the print line to spaces if necessary,
    - increments the page counter if necessary,
    - transfers the constants to be printed on the print line if necessary;
- Sub-function 'F8r00', if the report is to be printed by a spooling program (USAGE OF DATA STRUCTURE = 'J'), which contains:

- transfer of data to the common part segment,
- branch to the sub-function that prints the next structure;

• a sub-function per structure which contains:
- any 'PERFORM' commands the user has specified on the Report Description (-D) screen,
- incrementation of index Jddrcc, if the structure printed is the first of a repetitive category when the report is printed by category loading,
- the transfer of data to each data element in the structure,
- for structures containing totaling fields, the transfer of data is accomplished in three steps:
- non-totaled data elements,
- data elements to be totaled (where TYPE OF LINE IN REPORT = '*'),
- accumulator fields: (the CATEGORY OF REPORT being processed determines the level of accumulator to be moved);

• Sub-function 'F8r99' which contains:
- the WRITE commands for the report:

  For a direct print file (USAGE OF D. S. = 'I'), the commands vary according to the page/line skip characteristics,

  For a spooled file, there is only one WRITE command if the carriage control character is not the first element of the common part (00) structure. Otherwise, the commands vary as in the non-spooled file,

  If no category is defined, a simple WRITE statement is generated,
- incrementation of the counter of printed lines.

```
N81.      NOTE ************************************.           PJJPS1
               *                                    *           PJJPS1
               *PRINTING OF REPORT   1              *           PJJPS1
               *                                    *           PJJPS1
               ************************************.           PJJPS1
F81.                                                           PJJPS1
          IF    FT = ALL '1'                                   PJJPS1
          NEXT SENTENCE ELSE GO TO F81-FN.                     PJJPS1
N81BC.    NOTE *  LOADING    CATEGORY     BC       *.          PJJPS1
F81BC.                                                         PJJPS1
          MOVE                  01 TO 5-LI00-1LC             PJJPS1
          ADD      1-BC-NL TO         5-LI00-1LC             PJJPS1
          MOVE       'BC'   TO CATX.                         PJJPS1
          MOVE   '*'        TO ST-ABS                        PJJPS1
          MOVE   '0001011' TO ST-SLS.                        PJJPS1
          PERFORM   F81ZZ-010 THRU     F8199-FN.             PJJPS1
          MOVE   '0102021' TO ST-SLS.                        PJJPS1
          PERFORM   F81ZZ-010 THRU     F8199-FN.             PJJPS1
          MOVE   '0203022' TO ST-SLS.                        PJJPS1
          PERFORM   F81ZZ-010 THRU     F8199-FN.             PJJPS1
          MOVE   '030402 ' TO ST-SLS.                        PJJPS1
          PERFORM   F81ZZ-010 THRU     F8199-FN.             PJJPS1
          MOVE   '040502 ' TO ST-SLS.                        PJJPS1
```

```
          PERFORM   F81ZZ-010 THRU     F8199-FN.                PJJPS1
          MOVE   '000602 ' TO ST-SLS.                           PJJPS1
          PERFORM   F81ZZ-010 THRU     F8199-FN.                PJJPS1
          MOVE   '000701 ' TO ST-SLS.                           PJJPS1
          PERFORM   F81ZZ-010 THRU     F8199-FN.                PJJPS1
F81BC-FN. EXIT.                                                 PJJPS1
N81DD.    NOTE *  LOADING    CATEGORY     DD       *.          PJJPS1
F81DD.                                                          PJJPS1
          MOVE          012     TO    JLI1DD.                   PJJPS1
          MOVE JLI1DD           TO    JLI1DDM.                  PJJPS1
          MOVE 1                TO    JLI1DD.                   PJJPS1
F81DD-A.                                                        PJJPS1
          IF   JLI1DD > JLI1DDM  GO TO F81DD-FN.                PJJPS1
          ADD      1-DD-NL TO         5-LI00-1LC                PJJPS1
          MOVE      'DD'   TO CATX.                             PJJPS1
          MOVE   '050801 ' TO ST-SLS.                           PJJPS1
          PERFORM   F81ZZ-010 THRU     F8199-FN.                PJJPS1
          ADD  1         TO           JLI1DD                   PJJPS1
          GO    TO                     F81DD-A.                 PJJPS1
F81DD-FN. EXIT.                                                 PJJPS1
N81EE.    NOTE *  LOADING    CATEGORY     EE       *.          PJJPS1
F81EE.                                                          PJJPS1
          ADD      1-EE-NL TO         5-LI00-1LC                PJJPS1
          MOVE      'EE'   TO CATX.                             PJJPS1
          MOVE   '000901 ' TO ST-SLS.                           PJJPS1
          PERFORM   F81ZZ-010 THRU     F8199-FN.                PJJPS1
F81EE-FN. EXIT.                                                 PJJPS1
F81ZZ.    MOVE SPACE TO CATX.   GO TO  F81-FN.                  PJJPS1
F81ZZ-010.                                                      PJJPS1
         IF J02 = '00' MOVE SPACE TO 6-LI100 ELSE               PJJPS1
          MOVE 1-LI00-1 (J02)     TO 6-LI100.                   PJJPS1
          IF ST-ABS NOT = ' ' AND LSKP = '01'                   PJJPS1
          ADD         1            TO 5-LI00-1PC.               PJJPS1
F81ZZ-FN. EXIT.                                                 PJJPS1
N8100.    NOTE *    STRUCTURE 00  REPORT 1         *.          PJJPS1
F8100.                                                          PJJPS1
          PERFORM F91BC  THRU     F91BC-FN.                     PJJPS1
          MOVE  'B'             TO   6-LI100-ETAT.            PJJPS1
          MOVE  LSKP            TO   6-LI100-LSKP.            PJJPS1
          MOVE   5-LI00-1PC     TO   6-LI100-PAGE.            PJJPS1
          IF STX = '00'          GO TO F8199.                   PJJPS1
          GO TO                       F8101                     PJJPS1
                                      F8102                     PJJPS1
                                      F8103                     PJJPS1
                                      F8104                     PJJPS1
                                      F8105                     PJJPS1
                                      F8106                     PJJPS1
          DEPENDING ON ST9.                                     PJJPS1
F8100-FN. EXIT.                                                 PJJPS1
N8101.    NOTE *   PRINT    STRUCTURE   01         *.          PJJPS1
F8101.                                                          PJJPS1
          MOVE     WA04-ACCEP   TO   6-LI101-ACCEP.             PJJPS1
F8101-99. GO TO  F8199.                                         PJJPS1
F8101-FN. EXIT.                                                 PJJPS1
N8102.    NOTE *   PRINT    STRUCTURE   02         *.          PJJPS1
```

```
F8102.                                                          PJJPS1
         MOVE     WA04-REFUS   TO   6-LI102-REFUS.              PJJPS1
F8102-99. GO TO  F8199.                                         PJJPS1
F8102-FN. EXIT.                                                 PJJPS1
N8103.    NOTE *   PRINT    STRUCTURE   03         *.           PJJPS1
F8103.                                                          PJJPS1
      COMPUTE 6-LI103-TOTAL                     =               PJJPS1
                  WA04-ACCEP                                    PJJPS1
              +   WA04-REFUS.                                   PJJPS1
F8103-99. GO TO  F8199.                                         PJJPS1
F8103-FN. EXIT.                                                 PJJPS1
N8104.    NOTE *   PRINT    STRUCTURE   04         *.           PJJPS1
F8104.                                                          PJJPS1
         MOVE  ZERO            TO   6-LI104-POURC.              PJJPS1
         IF    WA04-ACCEP > 0 OR WA04-REFUS > 0                 PJJPS1
      COMPUTE 6-LI104-POURC          ROUNDED    =               PJJPS1
                100                                             PJJPS1
              *    WA04-REFUS                                   PJJPS1
              / (WA04-ACCEP                                     PJJPS1
              +    WA04-REFUS).                                 PJJPS1
F8104-99. GO TO  F8199.                                         PJJPS1
F8104-FN. EXIT.                                                 PJJPS1
N8105.    NOTE *   PRINT    STRUCTURE   05         *.           PJJPS1
F8105.                                                          PJJPS1
         MOVE     WC02-NOFICH       (JLI1DD)                   PJJPS1
                                TO   6-LI105-NOFICH.            PJJPS1
         MOVE     WC03-CPTENR       (JLI1DD)                   PJJPS1
                                TO   6-LI105-CPTENR.            PJJPS1
F8105-99. GO TO  F8199.                                         PJJPS1
F8105-FN. EXIT.                                                 PJJPS1
N8106.    NOTE *   PRINT    STRUCTURE   06         *.           PJJPS1
F8106.         EXIT.                                            PJJPS1
F8106-99. GO TO  F8199.                                         PJJPS1
F8106-FN. EXIT.                                                 PJJPS1
N8199.    NOTE *   WRITE  REPORT         1         *.           PJJPS1
F8199.    MOVE     6-LI00     TO   LI00.                       PJJPS1
         MOVE ' ' TO ST-ABS.                                    PJJPS1
         WRITE      LI00.                                       PJJPS1
F8199-20. ADD 1 TO 5-LI00-1RC.                                  PJJPS1
F8199-FN. EXIT.                                                 PJJPS1
F81-FN.   EXIT.                                                 PJJPS1
N83.      NOTE ************************************.            PJJPS1
              *                                    *            PJJPS1
              *PRINTING OF REPORT   3              *            PJJPS1
              *                                    *            PJJPS1
              ************************************.            PJJPS1
F83.                                                            PJJPS1
         IF    LV-OC3 = ZERO OR FTB3 = ZERO                     PJJPS1
         NEXT SENTENCE ELSE GO TO F83-FN.                       PJJPS1
N83DA.    NOTE *  LOADING    CATEGORY     DA        *.          PJJPS1
F83DA.                                                          PJJPS1
         IF    5-ED00-3LC + 2-LV00-NBLIV NOT <                  PJJPS1
               5-ED00-3LCM                                      PJJPS1
         MOVE                  01 TO 5-ED00-3LC                 PJJPS1
         ADD      3-DA-NL TO         5-ED00-3LC                PJJPS1
```

```
          MOVE      'DA' TO CAT (J00) ADD 1 TO J00.             PJJPS1
F83DA-FN. EXIT.                                                 PJJPS1
N83EA.    NOTE *  LOADING    CATEGORY     EA       *.          PJJPS1
F83EA.                                                          PJJPS1
          ADD      3-EA-NL TO         5-ED00-3LC                PJJPS1
          MOVE      'EA' TO CAT (J00) ADD 1 TO J00.            PJJPS1
F83EA-FN. EXIT.                                                 PJJPS1
N83FA.    NOTE *  LOADING    CATEGORY     FA       *.          PJJPS1
F83FA.         EXIT.                                            PJJPS1
F83FA-A.                                                        PJJPS1
          IF   JED3FA  = ZERO   GO  TO F83FA-FN.                PJJPS1
          IF  5-ED00-3LC  NOT  <      5-ED00-3LCM               PJJPS1
          PERFORM                      F83IL                    PJJPS1
          PERFORM                      F83DA.                   PJJPS1
          ADD      3-FA-NL TO         5-ED00-3LC                PJJPS1
          MOVE      'FA' TO CAT (J00) ADD 1 TO J00.            PJJPS1
          SUBTRACT   1  FROM          JED3FA                    PJJPS1
          GO    TO                     F83FA-A.                 PJJPS1
F83FA-FN. EXIT.                                                 PJJPS1
N83GA.    NOTE *  LOADING    CATEGORY     GA       *.          PJJPS1
F83GA.                                                          PJJPS1
          IF  IBL  =    ZERO                                    PJJPS1
          OR  IBL  >       2  GO TO    F83GA-100.              PJJPS1
          MOVE IBL TO J05.                                      PJJPS1
F83GA-090.                                                      PJJPS1
          MOVE  ZERO    TO         T304-QUCO   (J05).          PJJPS1
          MOVE  ZERO    TO         T304-QTLI   (J05).          PJJPS1
          ADD  1   TO   J05.                                    PJJPS1
          IF  J05 NOT >    2  GO TO    F83GA-090.               PJJPS1
F83GA-100. EXIT.                                                PJJPS1
F83GA-150.                                                      PJJPS1
          ADD    2-CD00-QUCO   TO  T304-QUCO   (2).           PJJPS1
          ADD    2-LV00-QTLI   TO  T304-QTLI   (2).           PJJPS1
          ADD      3-GA-NL TO         5-ED00-3LC                PJJPS1
          MOVE      'GA' TO CAT (J00) ADD 1 TO J00.            PJJPS1
F83GA-200.                                                      PJJPS1
          IF FBL = ZERO  GO TO         F83GA-FN.                PJJPS1
          MOVE             2  TO     J07.                      PJJPS1
F83GA-300. SUBTRACT 1 FROM J07 GIVING J06.                      PJJPS1
         IF J07 < FBL OR J07 = 1 GO TO F83GA-400.               PJJPS1
          ADD T304-QUCO   (J07) TO T304-QUCO   (J06).           PJJPS1
          ADD T304-QTLI   (J07) TO T304-QTLI   (J06).           PJJPS1
          SUBTRACT 1 FROM  J07 GO TO   F83GA-300.               PJJPS1
F83GA-400. EXIT.                                                PJJPS1
F83GA-500. IF FBL NOT = 1        GO TO F83GA-FN.                PJJPS1
          ADD T304-QUCO   (1)   TO G304-QUCO.                   PJJPS1
          ADD T304-QTLI   (1)   TO G304-QTLI.                   PJJPS1
F83GA-FN. EXIT.                                                 PJJPS1
N83HA.    NOTE *  LOADING    CATEGORY     HA       *.          PJJPS1
F83HA.                                                          PJJPS1
          IF    FTB2 = 1 AND LV-IB2 = 1                         PJJPS1
          ADD      3-HA-NL TO         5-ED00-3LC                PJJPS1
          MOVE      'HA' TO CAT (J00) ADD 1 TO J00.            PJJPS1
F83HA-FN. EXIT.                                                 PJJPS1
N83IA.    NOTE *  LOADING    CATEGORY     IA       *.          PJJPS1
```

```
F83IA.                                                          PJJPS1
          IF    FTB1 = 1 AND LV-CF1 = 1                         PJJPS1
          ADD      3-IA-NL TO         5-ED00-3LC                PJJPS1
          MOVE      'IA' TO CAT (J00) ADD 1 TO J00.             PJJPS1
F83IA-FN. EXIT.                                                 PJJPS1
N83IL.    NOTE *  LOADING    CATEGORY     IL       *.          PJJPS1
F83IL.                                                          PJJPS1
          IF    5-ED00-3LC NOT < 5-ED00-3LCM                    PJJPS1
          ADD      3-IL-NL TO         5-ED00-3LC                PJJPS1
          MOVE      'IL' TO CAT (J00) ADD 1 TO J00.             PJJPS1
F83IL-FN. EXIT.                                                 PJJPS1
N83JA.    NOTE *  LOADING    CATEGORY     JA       *.          PJJPS1
F83JA.                                                          PJJPS1
          IF    FT = ALL '1'                                    PJJPS1
          ADD      3-JA-NL TO         5-ED00-3LC                PJJPS1
          MOVE      'JA' TO CAT (J00) ADD 1 TO J00.             PJJPS1
F83JA-FN. EXIT.                                                 PJJPS1
F83ZZ.    MOVE 1 TO J00.                                        PJJPS1
F83ZZ-005. MOVE CAT (J00) TO CATX. IF CATX = '  '               PJJPS1
          MOVE 1 TO J00 MOVE SPACE TO CAT-TAB                   PJJPS1
          GO TO     F8399-FN.  MOVE 0 TO J01.                   PJJPS1
          IF CATX                 = 'DA'                        PJJPS1
          MOVE  TS-3-DA TO ST-TA GO TO F83ZZ-009.               PJJPS1
          IF CATX                 = 'EA'                        PJJPS1
          MOVE  TS-3-EA TO ST-TA GO TO F83ZZ-009.               PJJPS1
          IF CATX                 = 'FA'                        PJJPS1
          MOVE  TS-3-FA TO ST-TA GO TO F83ZZ-009.               PJJPS1
          IF CATX                 = 'GA'                        PJJPS1
          MOVE  TS-3-GA TO ST-TA GO TO F83ZZ-009.               PJJPS1
          IF CATX                 = 'HA'                        PJJPS1
          MOVE  TS-3-HA TO ST-TA GO TO F83ZZ-009.               PJJPS1
          IF CATX                 = 'IA'                        PJJPS1
          MOVE  TS-3-IA TO ST-TA GO TO F83ZZ-009.               PJJPS1
          IF CATX                 = 'IL'                        PJJPS1
          MOVE  TS-3-IL TO ST-TA GO TO F83ZZ-009.               PJJPS1
          IF CATX                 = 'JA'                        PJJPS1
          MOVE  TS-3-JA TO ST-TA GO TO F83ZZ-009.               PJJPS1
F83ZZ-009. ADD 1 TO J01.                                        PJJPS1
F83ZZ-010. MOVE ST-TT (J01) TO ST-SLS.                          PJJPS1
          IF  ST-SLS = SPACE                                    PJJPS1
          ADD 1 TO J00           GO TO F83ZZ-005.               PJJPS1
         IF J02 = '00' MOVE SPACE TO 6-ED300 ELSE               PJJPS1
          MOVE 1-LI00-3 (J02)     TO 6-ED300.                   PJJPS1
          IF ST-ABS NOT = ' ' AND LSKP = '01'                   PJJPS1
          ADD         1            TO 5-ED00-3PC.               PJJPS1
F83ZZ-FN. EXIT.                                                 PJJPS1
N8300.    NOTE *    STRUCTURE 00  REPORT 3         *.          PJJPS1
F8300.                                                          PJJPS1
          IF STX = '00'          GO TO F8399.                   PJJPS1
          GO TO                       F8301                     PJJPS1
                                      F8302                     PJJPS1
                                      F8303                     PJJPS1
                                      F8304                     PJJPS1
          DEPENDING ON ST9.                                     PJJPS1
F8300-FN. EXIT.                                                 PJJPS1
```

```
N8301.    NOTE *   PRINT    STRUCTURE   01         *.          PJJPS1
F8301.                                                         PJJPS1
          PERFORM F9101  THRU     F9101-FN.                    PJJPS1
          MOVE  DAT8C           TO   6-ED301-DATEM.          PJJPS1
          MOVE   5-ED00-3PC     TO   6-ED301-PAGE.           PJJPS1
F8301-99. GO TO  F8399.                                        PJJPS1
F8301-FN. EXIT.                                                PJJPS1
N8302.    NOTE *   PRINT    STRUCTURE   02         *.          PJJPS1
F8302.                                                         PJJPS1
          MOVE   2-CL00-NOCL    TO   6-ED302-NOCL.           PJJPS1
          MOVE   2-CL00-NOMCL   TO   6-ED302-NOMCL.          PJJPS1
F8302-99. GO TO  F8399.                                        PJJPS1
F8302-FN. EXIT.                                                PJJPS1
N8303.    NOTE *   PRINT    STRUCTURE   03         *.          PJJPS1
F8303.                                                         PJJPS1
          ADD  1  TO                  JED3FA.                  PJJPS1
          MOVE  'DELIVERY'      TO   6-ED303-FILLER.         PJJPS1
          MOVE  JED3FA          TO   6-ED303-JED3FA.         PJJPS1
          MOVE   2-LV00-DALI          (JED3FA)                 PJJPS1
                                TO   6-ED303-DATE.           PJJPS1
          MOVE   2-LV00-QULI          (JED3FA)                 PJJPS1
                                TO   6-ED303-QULI.           PJJPS1
F8303-99. GO TO  F8399.                                        PJJPS1
F8303-FN. EXIT.                                                PJJPS1
N8304.    NOTE *   PRINT    STRUCTURE   04         *.          PJJPS1
F8304.                                                         PJJPS1
          MOVE   1-LI00-4      (J05)                         PJJPS1
                                TO   6-ED304-4.              PJJPS1
          IF    J05 < 4                                      PJJPS1
          MOVE   2-CL00-NOCL11  TO   6-ED304-NOCL11.         PJJPS1
          IF    J05 = 2 OR J05 = 3                           PJJPS1
          MOVE   2-CL00-NOCL12  TO   6-ED304-NOCL12.         PJJPS1
          IF    J05 = 3                                      PJJPS1
          MOVE   2-CL00-NOCL2   TO   6-ED304-NOCL2.          PJJPS1
          IF    J05 = 3                                      PJJPS1
      COMPUTE 6-ED304-SOLDE                    =             PJJPS1
                2-CD00-QUCO                                  PJJPS1
              -  2-LV00-QTLI.                                PJJPS1
          IF    J05 NOT = 3                                  PJJPS1
      COMPUTE 6-ED304-SOLDE                    =             PJJPS1
                  T304-QUCO   (J05)                          PJJPS1
              -   T304-QTLI   (J05).                         PJJPS1
          IF CATX NOT =   'GA'  GO TO  F8304-TOT.            PJJPS1
          MOVE   2-CD00-QUCO    TO   6-ED304-QUCO.           PJJPS1
          MOVE   2-LV00-QTLI    TO   6-ED304-QTLI.           PJJPS1
          GO TO                       F8399.                 PJJPS1
F8304-TOT.                                                     PJJPS1
          IF CATX NOT  =  'IA'                                 PJJPS1
                                GO TO   F8304-IAF.           PJJPS1
         MOVE T304-QUCO      (1) TO  6-ED304-QUCO.           PJJPS1
         MOVE T304-QTLI      (1) TO  6-ED304-QTLI.           PJJPS1
                                GO TO  F8304-99.             PJJPS1
F8304-IAF.                                                     PJJPS1
          IF CATX NOT  =  'HA'                                 PJJPS1
                                GO TO   F8304-HAF.           PJJPS1
```

```
         MOVE T304-QUCO      (2) TO  6-ED304-QUCO.              PJJPS1
         MOVE T304-QTLI      (2) TO  6-ED304-QTLI.              PJJPS1
                               GO TO  F8304-99.                 PJJPS1
F8304-HAF.                                                      PJJPS1
          IF  CATX NOT =  'JA'                                  PJJPS1
                               GO TO  F8399.                    PJJPS1
         MOVE G304-QUCO          TO 6-ED304-QUCO.               PJJPS1
         MOVE G304-QTLI          TO 6-ED304-QTLI.               PJJPS1
F8304-99. GO TO  F8399.                                         PJJPS1
F8304-FN. EXIT.                                                 PJJPS1
N8399.    NOTE *   WRITE  REPORT         3         *.           PJJPS1
F8399.    MOVE     6-ED00     TO   ED00.                       PJJPS1
          IF ST-ABS = ' '        GO TO F8399-10.               PJJPS1
          MOVE ' ' TO ST-ABS.                                   PJJPS1
          IF LSKP = '01' MOVE 1    TO 5-ED00-3LC1               PJJPS1
          WRITE ED00    AFTER   ADVANCING LSKPP                 PJJPS1
          GO TO  F8399-20.                                      PJJPS1
          SUBTRACT 5-ED00-3LC1  FROM LSKP.                      PJJPS1
F8399-10. IF LSKP = '00'                                        PJJPS1
          WRITE ED00    AFTER   ADVANCING LSKP0 ELSE            PJJPS1
          WRITE ED00    AFTER   ADVANCING LSKP                  PJJPS1
          ADD  LSKP               TO 5-ED00-3LC1.               PJJPS1
F8399-20. ADD 1 TO 5-ED00-3RC.   GO TO F83ZZ-009.               PJJPS1
F8399-FN. EXIT.                                                 PJJPS1
F83-FN.   EXIT.                                                 PJJPS1
```

Write files (F90)

Function F90 is generated for all ouput sequential files with USAGE D, S, R, or E.

Primary purpose: Function F90 does the WRITE to the segment. Also, it unconditionally causes a loop back to Function F05.

Sub-functions: There is one sub-function per output d.s. (as described above). The sub-function code is created using the DATA STRUCTURE CODE IN THE PROGRAM.

This function contains:

- no execution conditions for the function;
- a sub-function per output file containing:
  - the test giving access to the sub-function write:

    For USAGE OF DATA STRUCTURE = 'D', 'S' or 'R':

    a) The highest file matching level is occuring,

    b) all control breaks have been processed,

    For USAGE OF DATA STRUCTURE = 'E':

    a) The highest file matching level is occuring;

- the transfer of 'OCCURS DEPENDING ON' counters if the file, linked to a principal file, contains the counter in the common part;
- transfer from the update area to the segment, (for USAGE = 'S', 'R' or 'D');
- the transfer of data into the reserved data elements (ENPR, GRPR, ERUT) from error tables, and into the element dd00-SUITE from the read area of the transaction file (for USAGE = 'E', if these elements are in the file, - see RESERVED ERROR CODES IN TRANS. FILE on the Call of Data Structures (-CD) screen);

NOTE:: If not selected, the transfer is done in Function F76;

- The WRITE command:

  For a variable length record, (RECORDING MODE = 'V'), there is one WRITE per record type, preceded by a test on record type;
- increment record counter;
- Paragraph F9099-ITER-FN, an unconditional GO TO F05.

By default, the date processing function is generated in F9520. However, if you have specific code lines in F9520, you may change the date processing function in order to keep your specific code lines. To do so, in an 'O'-type line of the Generation Options (-GO) screen of the Program, enter the DATPRO=ffss function ('ffss' being the new function-subfunction code).

Note: In the Library Generation Options screen, an Administrator can specify whether the F9520 function may be overridden by source code lines.

```
N90.      NOTE ************************************.          PJJPS1
               *                                    *          PJJPS1
               *          WRITE                     *          PJJPS1
               *                                    *          PJJPS1
               ************************************.          PJJPS1
F90.      EXIT.                                               PJJPS1
N90DC.    NOTE *     WRITE RECORDS ON   DC-FILE    *.         PJJPS1
F90DC.                                                        PJJPS1
          IF       CD-OC3   = '1'                             PJJPS1
                AND  FTB3   = '1'                             PJJPS1
          NEXT SENTENCE ELSE GO TO     F90DC-FN.              PJJPS1
          MOVE   2-CD00       TO          DC00.              PJJPS1
          WRITE    DC00.                                      PJJPS1
F90DC-99. ADD 1 TO 5-DC00-RECCNT.                             PJJPS1
F90DC-FN. EXIT.                                               PJJPS1
N90LC.    NOTE *     WRITE RECORDS ON   LC-FILE    *.         PJJPS1
F90LC.                                                        PJJPS1
          IF       CL-OC3   = '1'                             PJJPS1
                AND  FTB3   = '1'                             PJJPS1
          NEXT SENTENCE ELSE GO TO     F90LC-FN.              PJJPS1
          MOVE   2-CL00       TO          LC00.              PJJPS1
          WRITE    LC00.                                      PJJPS1
F90LC-99. ADD 1 TO 5-LC00-RECCNT.                             PJJPS1
```

```
F90LC-FN. EXIT.                                                    PJJPS1
N90SE.    NOTE *     WRITE RECORDS ON   SE-FILE    *.             PJJPS1
F90SE.                                                             PJJPS1
          IF       CL-OC3   = '1'                                  PJJPS1
                AND  FTB3   = '1'                                  PJJPS1
          NEXT SENTENCE ELSE GO TO     F90SE-FN.                   PJJPS1
          MOVE   2-CL00       TO          SE00.                  PJJPS1
          WRITE    SE00.                                         PJJPS1
F90SE-99. ADD 1 TO 5-SE00-RECCNT.                                  PJJPS1
F90SE-FN. EXIT.                                                    PJJPS1
N90VL.    NOTE *     WRITE RECORDS ON   VL-FILE    *.             PJJPS1
F90VL.                                                             PJJPS1
          IF       LV-OC3   = '1'                                  PJJPS1
                AND  FTB3   = '1'                                  PJJPS1
          NEXT SENTENCE ELSE GO TO     F90VL-FN.                   PJJPS1
          MOVE   2-LV00-NBLIV  TO       VL00-NBLIV                PJJPS1
          MOVE   2-LV00       TO          VL00.                  PJJPS1
          WRITE    VL00.                                         PJJPS1
F90VL-99. ADD 1 TO 5-VL00-RECCNT.                                  PJJPS1
F90VL-FN. EXIT.                                                    PJJPS1
N90VM.    NOTE *     WRITE RECORDS ON   VM-FILE    *.             PJJPS1
F90VM.                                                             PJJPS1
          IF       MV-CF3   = '1'                                  PJJPS1
          NEXT SENTENCE ELSE GO TO     F90VM-FN.                   PJJPS1
          MOVE    ID-ER         TO      ER-ID.                   PJJPS1
          MOVE    ER-00         TO      ER-PR0.                  PJJPS1
          IF      I01           = 001                            PJJPS1
          MOVE    ER-01         TO      ER-PRM.                  PJJPS1
          IF      I01           = 002                            PJJPS1
          MOVE    ER-02         TO      ER-PRM.                  PJJPS1
          MOVE    ER-PRR        TO      VM00-ENPR.               PJJPS1
          MOVE    SE-ERR        TO      VM00-GRPR.               PJJPS1
          MOVE    UT-ERUT       TO      VM00-ERUT.               PJJPS1
          MOVE   1-MV00         TO      VM00-SUITE.              PJJPS1
          WRITE    VM00.                                         PJJPS1
F90VM-99. ADD 1 TO 5-VM00-RECCNT.                                  PJJPS1
F90VM-FN. EXIT.                                                    PJJPS1
F90-FN.   EXIT.                                                    PJJPS1
F9099-ITER-FN.  GO TO F05.                                         PJJPS1
N91BC.    NOTE *LINE NUMBER IMPLEMENT             *.             P000
F91BC.                                                             P000
          IF    ST-ABS NOT = SPACE                               P001
          AND   LSKP = '01'                                      P120
    MOVE        ZERO TO 6-LI100-NULIG.                           P001
    ADD         1 TO 6-LI100-NULIG.                              P200
F91BC-FN. EXIT.                                                    P200
N9101.    NOTE *SAME PLAYER SHOOTS AGAIN          *.             P000
F9101.         EXIT.                                               P000
F9101-FN. EXIT.                                                    P000
N9999.    NOTE *RETOUR DU TRI                     *.             P000
F9999.         EXIT.                                               P000
F9999-FN. EXIT.                                                    P000
```

Part Number: DDBTC000352A - 7345

Printed in USA