PERCON V2.9A

Edition January 2007

Comments… Suggestions… Corrections…

The User Documentation Department would like to know your opinion on this manual. Your feedback helps us to optimize our documentation to suit your individual needs.

Feel free to send us your comments by e-mail to
manuals@fujitsu-siemens.com

Certified documentation according to DIN EN ISO 9001:2000

To ensure a consistently high quality standard and user-friendliness, this documentation was created to meet the regulations of a quality management system which complies with the requirements of the standard DIN EN ISO 9001:2000.

cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de

Copyright and Trademarks

Contents

1 Preface

This manual describes the functions and mode of operation of the software product PERCON.

1.1 Brief product description

PERCON (PERipheral CONverter) is a program for the transfer and conversion of data to any volume.

PERCON V2.9A can be run under BS2000/OSD-BC V6.0 or higher.

PERCON offers the following functions for interactive and batch operation:

- copying of files; modification of file attributes
- editing of file contents for output to SYSLST, SYSOUT or cataloged files
- duplication of tapes / tape cartridges
- selection of records / blocks
- restructuring of records
- editing of files or tape contents
- formation of groups
- invocation as a subprogram
- integration of user routines for record and label processing or for error recovery.
- support of Unicode

1.2 Target group

This manual is intended for BS2000 users who would like to convert their data with the aid of PERCON.

The user should be familiar with the operating system BS2000, in particular with its most important commands. Please refer to the manuals “Commands” [9], for more detailed information.

The user should also be familiar with the BS2000 command language SDF (System Dialog Facility), since the user interface, screen layout, and user prompting by PERCON are defined by means of SDF (see the “Introductory Guide to the SDF Dialog Interface” [6]).

1.3 Summary of contents

The manual comprises the following chapters:

- **Preface**

  This chapter contains an overview of the functions offered by PERCON.

- **Functions**

  This chapter describes the individual functions and the statements used to implement them.

- **Working with PERCON**

  This chapter describes how PERCON is started and terminated, how statements are entered, and how files are assigned.

- **Statements**

  Detailed descriptions of the statements are given in alphabetical order.

- **PERCON as a subprogram**

  This chapter contains the information necessary for calling PERCON from user programs.

- **User interfaces**

  This chapter describes how control of the PERCON run can be transferred to user modules.

- **Sample applications**

  This chapter contains selected examples of frequent PERCON applications.

- **Messages**

  This chapter lists all the PERCON messages, accompanied by the appropriate explanation and recommended course of action.

You will find various indexes at the back of the manual that will make working with this manual easier.

README file

Information on functional changes and additions to the current product version described in this manual can be found in the product-specific README file. You will find the README file on your BS2000 computer under the file name `SYSRME.PERCON.029.E`. The user ID under which the README file is cataloged can be obtained from your system support. You can also determine the file name with IMON by entering the following command:

```
/SHOW-INSTALLATION-PATH INSTALLATION-UNIT=PERCON, LOGICAL-ID=SYSRME.E
```

You can view the README file using the `/SHOW-FILE` command or an editor, and print it out on a standard printer using the following command:

```
/PRINT-DOCUMENT $userid.SYSRME.PERCON.029.E,
                LINE-SPACING=*BY-EBCDIC-CONTROL
```

1.4 Changes since the last version of the manual

This new version is PERCON V2.9A, and the following principal changes have been made since the last edition for PERCON V2.7A:

– PERCON V2.9A can run in BS2000/OSD-BC V6.0 and higher.

– Support of Unicode for files on disk and tape (cataloged files). Conversion can take place from a non-Unicode format to a Unicode format or vice versa. When Unicode variant UTF-16 is used, normalization can take place in addition to conversion. The ASSIGN-OUTPUT-FILE and SET-RECORD-MAPPING statements have been supplemented to support Unicode.

– Inclusion of a new example which illustrates conversion of a file from EDF03IRV to UTF-16.

– System administrators and applications which run in the privileged processor state TPR can work with tape blocks of more than 32 Kbytes. To permit this, the BUFFER-LENGTH operand of the ASSIGN-INPUT-TAPE statement has been supplemented.

– 6 rather than 5 characters are now used to output the page and line counter values.

– New installation files (suffix 029)

– PCROOT is no longer contained as an object module (OM). The linkage editor BINDER [5] must consequently be used to link PERCON as a subroutine. TSOSLNK should no longer be used. Since PCROOT is no longer available as an OM, the call `/EXEC (percsdf,$tsos.syslnk.percon.029)` no longer results in PERCON being loaded.

Incompatibilities

Older applications that invoke PERCON as a subprogram, and therefore have an integrated starter module from PERCON V2.5A or earlier, must be linked in again with the PERCON V2.9 starter module.

1.5 Notational conventions

The following notational conventions are used in this manual:

- The metasyntax of the PERCON statements is in SDF command format. This is described from page 69.
- References to publications are shown in the text by abbreviated titles and square brackets [ ]. The full title of each publication to which reference is made is listed under "Related publications" at the back of the manual.
- User inputs in the sample runs are shown in **boldface**.
- The following pictogram is used to emphasize important information:

ℹ	Indicates particularly important information.

2 Functions

All PERCON functions involve data transfer. A distinction is made between three basic functions and five subsidiary functions.

The three **basic functions** comprise:

- copying files
- editing tapes
- duplicating tapes.

These functions are distinguished by the access method for the input and output data.

Basic function	Access to	
	Input data	**Output data**
Copying files	Logical	Logical
Editing tapes	Physical	Logical
Duplicating tapes	Physical	Physical

In the case of **logical access**, files are accessed via their file name. The units to be processed are records; they are processed in their logical order.

In the case of **physical access**, tapes and tape cartridges are accessed via their volume serial number (VSN). The units to be processed are blocks; they are processed in their physical order.

Information which applies to both tapes and tape cartridges is described below solely with reference to tapes; unless otherwise specified, it may be assumed that the information is also valid for tape cartridges.

The basic functions may be complemented by five **subsidiary functions**; these five functions comprise:

– selecting records
– formatting records
– editing records and blocks
– forming groups of records
– positioning tapes.

The following combinations of basic functions and subsidiary functions are allowed:

	Basic functions		
Subsidiary function	**Copying files**	**Editing tapes**	**Duplicating tapes**
Selecting	+	+	-
Formatting	+	-	-
Editing	+	+	-
Grouping	+	-	-
Positioning	-	+	-

+	permissible combination
-	illegal combination

2.1 Basic functions

2.1.1 Copying files

The copying of files involves the transfer of one or more input files to one or more output files. Several input files are processed consecutively in the order in which they are assigned; records are transferred simultaneously to several output files. **Input files** can be files on disk or tape (cataloged files) or on SYSDTA.
Output files can be files on disk or tape (cataloged files) on SYSOUT or SYSLST.

The files are assigned using the statements ASSIGN-INPUT-FILE and ASSIGN-OUTPUT-FILE. If only one input or output file is to be assigned, this can be done by means of a ADD-FILE-LINK command using the link name PCIN or PCOUT.

PERCON is capable of processing files with a capacity in excess of 32 GBytes (without having to make changes at the user interface).

2.1.2 Editing tape contents

This function involves a tape being read block by block and one record for each block being transferred to one or more output files. SYSOUT and SYSLST are also permitted as output files.

The input tape must be assigned by means of the ASSIGN-INPUT-TAPE statement. The output file is assigned via an ASSIGN-OUTPUT-FILE statement and/or a ADD-FILE-LINK command; both assignment options are also possible if there is more than one output file.

The input tape can be positioned by means of a CHANGE-INPUT-TAPEPOSITION statement. Data block transfer is initiated by means of a START-TAPE-PROCESSING statement. Positioning and transfer can be repeated as often as required.

Editing is terminated by means of the START-CONVERSION or END statement.

This function also enables non-BS2000 or damaged tapes to be processed.

2.1.3 Duplicating tapes

Duplicating tapes involves the block-by-block transfer of an input tape. Multifile/multivolume sets (MF/MV sets, see the manual “Introductory Guide to DMS” [1]) can also be transferred.

The tapes must be assigned via their VSNs using the ASSIGN-INPUT-TAPE and ASSIGN-OUTPUT-TAPE statements.

A tape is read as far as the double tape mark which delimits the tape.

If not enough output tapes are specified for the output data when duplicating tapes, PERCON issues the message PER0084 and requests the operator to mount another output tape. Further tapes are requested until the duplication operation has been completed. If the operator ignores any of these requests, PERCON outputs the message PER0069 and aborts the conversion step, with the result that the copy will be incomplete.

Tapes or MF/MV sets with standard or nonstandard labels or without labels can be processed.

The following applies to MF/MV sets with standard labels:
If the input and output tapes have different sizes or recording densities, tape marks and labels which serve only to continue a file on the next tape are not transferred to the output set. As a result the input and output tapes do not have the same number of labels. If, on the other hand, input and output tapes have the same size and recording density, all standard labels are transferred to the MF/MV output set (including those required for continuing a file). The result is an MF/MV set with standard labels.

The following applies to MF/MV sets without standard labels:
PERCON transfers the contents of each input tape to an output tape without restructuring the data, thus creating another MF/MV set without standard labels.

Notes

– FDDRL save tape sets cannot be copied unless PERCON uses the user interface provided by FDDRL. Individual FDDRL tapes (not sets) can, however, be copied without the user interface (see example on page 227).

– The copying function provided by ARCHIVE must be used instead of PERCON for copying ARCHIVE save tape sets (see the “ARCHIVE” manual [10]). Since HSMS uses ARCHIVE, this also applies for tapes created using HSMS.

– In the case of physical access to tapes, i.e. when editing and duplicating, labels are also read and written like data blocks. The precondition for this is that the user has the relevant access rights (e.g. with `/MODIFY-USER-ATTRIBUTES PROTECTION-ATTRIBUTE=*PAR(TAPE-ACCESS=*BYPASS-LABEL)`).

2.2 Subsidiary functions

2.2.1 Selecting records or blocks

Records (in the case of file copying) or blocks (in the case of tape editing) can be selected for further processing. The selection is made by specifying a logical expression in the SELECT-INPUT-RECORDS statement; the following sequence of operations is required:

- Compare individual fields of a record or block with a selected literal (constant).
- Compare an internal PERCON block counter, byte counter or record counter (see page 78) with a selected literal.
- Check the data format of a field (numeric, alphabetic).
- Check whether the records have been sorted in ascending or descending order.
- Check the characters contained in a field against a table of permissible characters.

The following comparison operators may be used:

- equal to
- less than
- less than or equal to
- greater than
- greater than or equal to
- not equal to

Logical expressions may be generated on the basis of several such comparisons. Two logical operators may be used:

- AND
- OR

The expression is processed from left to right, “AND” having priority over “OR”. The input record/block is not transferred to the output file specified via OUTPUT-LINK-NAME unless the comparison yields the value “true”.

The order in which expressions are compared can be influenced by the use of parentheses.

Records or blocks which do not match any of the selection criteria can be transferred to so-called “residual files”.

2.2.2 Formatting records

When files are copied, the output records can be formatted in accordance with the user's wishes. The format is determined by specifications in the SET-RECORD-MAPPING statement relating to:

– input record fields
– literals (constants)
– keywords provided by PERCON.

The output area can be prefilled either with the input record or a fill character. If ISAM output files already exist, the records of these can also be used to prefill the output area.

In the output record a string can be represented in alphanumeric, hexadecimal or binary form, and a number in packed, zoned or edited form.

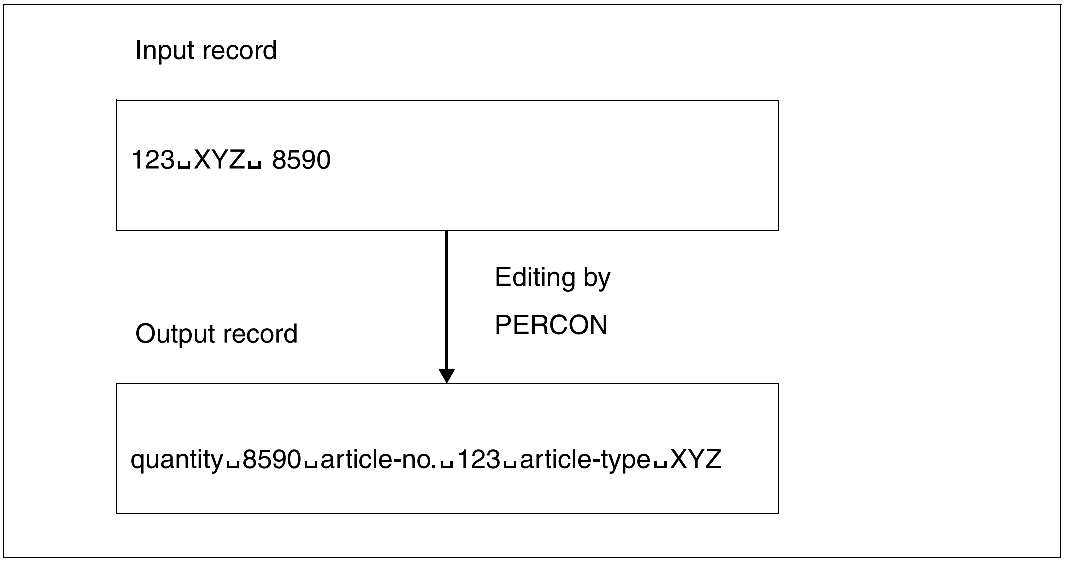


Figure 1: Format of an output record

When formatting an output record it may be necessary to modify the file attributes. For example, when inserting literals in a fixed-length record the record length must be changed.

If the formatted record is then edited, pay attention when defining the output position to whether an additional record length field is to be added to the data section of the unedited record (see section entitled “Editing records and blocks” on the next page).

A mandatory minimum length can be defined if the output records to be formatted are of variable length.

2.2.3 Editing records and blocks

When copying files or editing tapes, the page-justified records to be output are edited with the aid of the SET-PAGE-LAYOUT statement. The output records have a variable record format. Output is to SYSOUT, SYSLST or to a cataloged file. The unedited output records have a variable record format in a SAM file.

The unedited output records are used as the basis for editing; these records have the following characters by default:

– 32768 bytes maximum length

– undefined record format.

The default setting for unedited records can be modified using the following parameters in the ADD-FILE-LINK statement:

– RECORD-SIZE and/or BUFFER-LENGTH is used to define the record length. If both operands are specified, the lesser of the two values is assumed as the maximum record length.

– RECORD-FORMAT is used to define the format of the unedited output record. If RECORD-FORMAT=VARIABLE, the unedited record is provided with an additional record length field which is included in editing as a data section.

Note

If the output file has been defined with FILE-ATTRIBUTES = INPUT-FILE and if the first input file has the record format VARIABLE, the unedited record is still provided with the additional record length field in the data section.

If output to a cataloged file is required, the specifications in the TFT entry have priority over those in the ASSIGN-OUTPUT-FILE statement. Specifications in the catalog entry are not interpreted.

The output record counter (RECORD-COUNTER) counts the number of unedited records.

When an empty file is edited, a page which contains the header line is generated. If no header line is required, an empty page is generated.

Formatting the output via page and line editing

The print format can be defined on either a page or line basis by means of the SET-PAGE-LAYOUT statement. The following specifications are possible:

– The header line can be formatted (HEADER-LINE operand).

– Line areas to which the output records are to be written can be defined (OUTPUT-AREA operand).

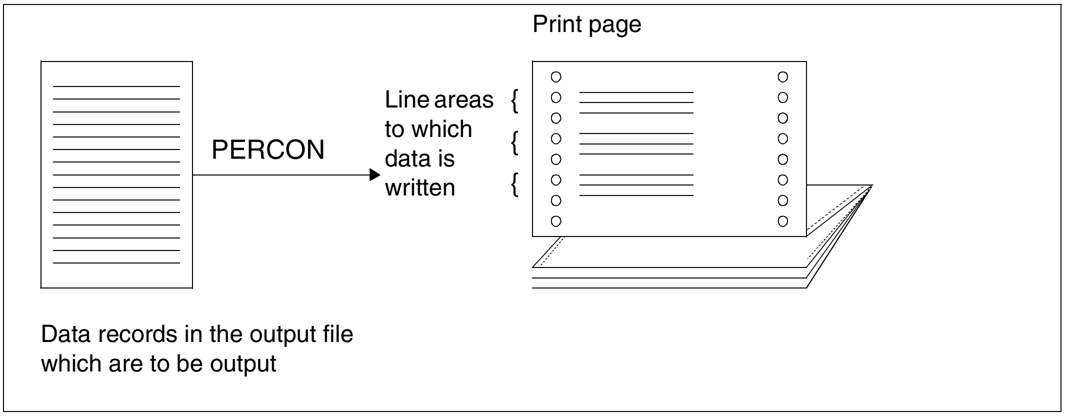

Print page
PERCON
Line areas to which data is written
Data records in the output file which are to be output

Figure 2: Formatting line areas

– Column areas to be written to in each line can be defined. Each of these areas is delimited by a space (COLUMN-SIZE operand).

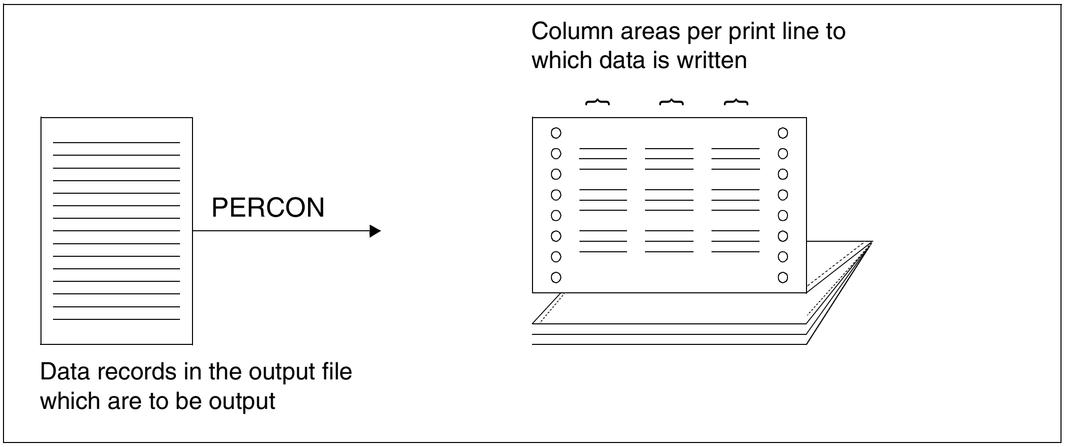

Column areas per print line to which data is written
PERCON
Data records in the output file which are to be output

Figure 3: Formatting column areas

- The number of characters of the unedited output record in each output line can be defined (LINE-SIZE operand).
- Output of lines with identical contents can be suppressed (SUPPRESS-EQUAL-LINES operand).
- On each page lines can be defined in which text defined by the user is to be inserted (user lines). No output records are written to these lines (USER-LINE operand).
- Data can be represented in hexadecimal format, character format or both (OUTPUT-FORMAT operand).
- The number of empty lines following an output line can be defined (SPACING operand).

The maximum length of an edited output record is the product of the leader and the operands COLUMN-SIZE, LINE-SIZE and OUTPUT-FORMAT. The length must not exceed 204 bytes.

Standard page formatting

In the case of output to SYSLST or SYSOUT as well as when a SET-GROUP-ATTRIBUTES, statement is specified, page editing is performed even if no SET-PAGE-LAYOUT statement was specified.
Standard page formatting has the following form:

for file copying

Page formatting	Output medium SYSLST or a cataloged file	SYSOUT
Page header (HEADER-LINE)	Page counter	None
Data area (OUTPUT-AREA)	Line 5 through 66	All lines
Data format (OUTPUT-FORMAT)	Character	Character
Line length (LINE-SIZE)	136 characters	64 characters
Group length (COLUMN-SIZE)	136 characters	16 characters
Suppression of identical continuation lines (SUPPRESS-EQUAL-LINES)	YES	YES
Additional information (as leader on the line)	None	9 characters Record position (BYTCNT)

for tape editing

Page formatting	Output medium SYSLST	SYSOUT
Page header (HEADER-LINE)	Date, time of day page counter	None
Data area (OUTPUT-AREA)	Line 5 through 66	All lines
Data format (OUTPUT-FORMAT)	Character and hexadecimal	Character and hexadecimal
Line length (LINE-SIZE)	100 characters	64 characters
Group length (COLUMN-SIZE)	20 characters	16 characters
Suppression of identical continuation lines (SUPPRESS-EQUAL-LINES)	YES	YES
Additional information (as leader on the line)	21 characters Tape mark counter (TMCNT) Block counter (BLKCNT) Record counter (BYTCNT)	9 characters Record position (BYTCNT)

2.2.4 Grouping records

Groups of records are formed by combining the records in a file in accordance with a common criterion, the grouping criterion.

In order to do this, it is necessary to specify in the SET-GROUP-ATTRIBUTES statement the fields in the input record which constitute the grouping criterion (BC, EF and IK in the example). Defined groups can be output together with data from this group and information specified by the user.

When a group is formed, editing of records (SET-PAGE-LAYOUT) is implicitly specified.

Forming a group with a group break

One or more fields of the input record are defined as the grouping criterion. Consecutive records with the same grouping criterion form one group. If all the records with the same grouping criterion are to be identified as a group, the file must be sorted according to this grouping criterion (e.g. by means of the software product SORT; see the “SORT” manual [4]).

A group break occurs whenever two consecutive files with different grouping criteria are detected when reading a file.

PERCON provides routines which are executed whenever a group break occurs. These routines, “group leader” and “group trailer”, are executed before and after a group break, respectively.

The following actions can be performed in the event of a group break:

– editing of group break lines from fields of the input record, from keywords and literals

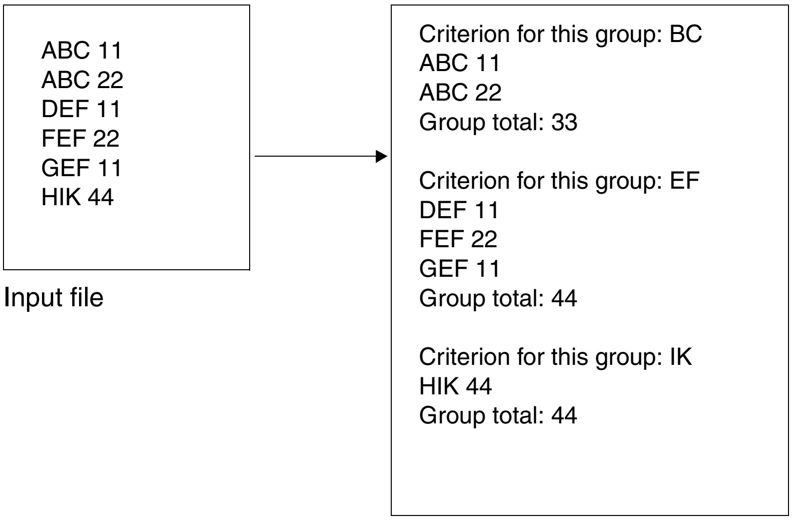
– formatting of group break lines with the aid of user modules

– line feed by 1 to 15 lines

– page feed by one page

– output of totals of specified record fields of the relevant group.

Example



Input file

Output file

Group definitions can be nested, i.e. several overlapping group criteria can be defined for a single file.

This gives rise to the concept of group levels. Up to 8 group levels are possible. Priorities are assigned by numbering the levels from 1 to 8, level 1 being the highest level. If a group break occurs on a higher priority level, it also takes place on all lower levels, even if the criterion for a group break is not satisfied in these groups. An example of a group break is given on page 209.

Each group level of a file is assigned a group counter (see page 80).

2.2.5 Positioning tapes

When editing tapes, it is possible to position the input tape before transferring data blocks. The CHANGE-INPUT-TAPEPOSITION statement can be used to position (either forwards or backwards):

- a specific number of blocks
- a specific number of tape marks
- to beginning-of-tape
- to end-of-tape
- to beyond end-of-tape (system administrator only)
- to the next double tape mark. Positioning is implemented immediately.

3 Working with PERCON

This chapter describes how the PERCON run is started, terminated and interrupted. It also illustrates how PERCON works in conversion steps, the sequence in which PERCON statements are entered, and how the input and output are assigned. Furthermore, the chapter also describes how cataloged files are processed, how PERCON behaves in the XS environment, and how extended character sets are used in PERCON.

3.1 Starting the PERCON run

The command /START-PERCON or /PERCON calls PERCON as an autonomous program. For the syntax description see page 69.

Application areas: **FILE, UTILITIES**

START-PERCON / **PERCON**
VERSION = ***STD** / <product-version> / <product-version without-corr> / <product-version without-man>
,**CPU-LIM**IT = ***JOB-REST** / <integer 1..32767>
,**MONJV** = ***NONE** / <filename 1..54 without-gen-vers>
,**PROG**RAM-**MODE** = 24 / ***ANY**

VERSION = *STD / / <product-version> / <product-version without-corr> / <product-version without-man>
Specifies the PERCON version to be called. The version can be called with various details as indicated below:

nn.nann or n.nann	(e.g. 02.9A00)	fully qualified version specification
nn.na or n.na	(e.g. 02.9A)	specification without correction status
nn.n or n.n	(e.g. 02.9)	specification without correction status and without release status

i n is a numeric, a is an alpha character.
The version specification can be prefixed by the letter V and enclosed in single quotes.

Omission of the VERSION operand or specification of the operand value *STD means that no particular version is requested. If the version specification does not supply sufficient details, thus referring to more than one installed PERCON version, the correct version is selected using the criteria listed on page 189. Execution of the start command is aborted and error message PER0104 is output if no version can be found that matches the version specification.

As an alternative, the command /SELECT-PRODUCT-VERSION can be used to request a particular PERCON version. This command is entered prior to calling PERCON (prior to calling the main program if PERCON is called as a subprogram). If the subsequent PERCON start command is entered without explicit version specification, the PERCON version specified with the /SELECT-PRODUCT-VERSION command will be started.

CPU-LIMIT = *JOB-REST / <integer 1..32767>
Specifies the maximum CPU time which a PERCON run may consume. If this limit is exceeded, either the system informs the user (interactive mode) or the PERCON run is terminated (batch mode).

CPU-LIMIT = *JOB-REST
If the operand CPU-LIMIT=*STD was specified in the SET-LOGON-PARAMETERS command, there is no time limit for the program.
If the operand CPU-LIMIT=t was specified in the SET-LOGON-PARAMETERS command, the value defined at system generation is taken as the time limit for the PERCON run.

MONJV = *NONE / <filename 1..54 without-gen-vers>
Specifies the name of the job variable that is to monitor the PERCON run. The job variable must already have been cataloged. This is only available to users with the software product JV (see the “Job Variables” manual [7]).

During the PERCON run the system sets the following values for the job variables:

Value	Meaning/reason for setting the value
$R	PERCON is executing.
$T	PERCON was terminated normally.
$A	PERCON was aborted.

MONJV = *NONE
No job variable is defined.

PROGRAM-MODE =
Defines the location in the address space where PERCON is loaded.

PROGRAM-MODE = <u>24</u>
The entire load unit is loaded below the 16-Mb boundary. The program is executed in 24-bit addressing mode. External references are interpreted as 24-bit addresses.

PROGRAM-MODE = *ANY
The modules of the load unit can be located either below or above the 16-Mb boundary.

3.2 Terminating the PERCON run

In order to terminate the PERCON run the END statement must be specified.

3.3 Command return code

When called as the main program, PERCON sets a command RC which indicates whether all PERCON statements have been executed without error and/or whether PERCON has terminated normally or abnormally. In the event of a warning or command abortion, the command RC is supplied with the relevant message code.

No command RC is set if PERCON is called as a subprogram.

The following events trigger a command RC:

1. normal termination: `SC2=X'00' / SC1=X'00' / MC=C'CMD0001'`
2. normal termination with warning: `SC2=X'02' / SC1=X'00' / MC=C'PERxxxx'`
3. abnormal termination after abortion of at least one conversion step:
   `SC2=X'00' / SC1=X'01' / MC=C'PERxxxx'`

   If there is no PER message (e.g. after syntactical errors in a mandatory sequence of statements, the following is set: `MC=C'CMD0202'`.
4. abnormal termination following an error that prevents dynamic linking of the prelinked PERCON module (and thus the PERCON run):
   `SC2=X'00' / SC1=X'40' / MC=C'PERxxxx'`

If more than one error situation occurs within the same PERCON run, the most serious error is reported. This implies, for instance, that the command RC assigned to a warning will not be output if a conversion step has been aborted. If several errors of the same weight occur, the first error is reported.

The maincode is determined by the event that has occurred rather than by the type of message output selected. It is therefore irrelevant for the maincode whether the message is actually output or suppressed in accordance with the settings in the MODIFY-PERCON-OPTIONS statement.

PERxxxx	SC2	SC1	Maincode	Meaning
0000 0029 0030 0070	-	-	-	No RC
0001	-	-	-	The program abortion is described in another PERCON message
	00	01	CMD0202	Other
0031	-	-	-	A warning is output by another PERCON message
	00	00	CMD0001	Other
0009 0012 0014 0016 0024 0040 0045 0051 0052 0054 0059 0065 0071 0084 0086 0092 0093 0094 0097 0109 0113 0117 0118	02	00	PERxxxx	Warnings
0021	02	00	PER0021	Response HN or SN received
	00	01	PER0021	Response HA or SA received
0022	00	01	PER0022	– for output files – for input files with TERMINATION = abnormal
	02	00	PER0022	for input files with TERMINATION = normal
0063	02	00	PER0063	Response I or S received
	00	01	PER0063	Response H received
0005 0006 0007 0008 0020 0025 0026 0027	-	-	-	with statement input via terminal
0028 0034 0043 0046 0047 0056 0062	00	01	PERxxxx	with mandatory sequence of statements
0010 0033 0049 0050 0055 0058 0067 0072	-	-	-	with statement input via terminal
0088 0089 0090 0091	02	00	PERxxxx	with fixed sequence of statements
0013 0066 0080 0087	-	-	-	with statement input via terminal
	02	00	PERxxxx	ISP statements, mandatory sequence of statements
	00	01	PERxxxx	SDF statements, mandatory sequence of statements

PERxxxx	**SC2**	**SC1**	**Maincode**	**Meaning**
0002 0004 0011 0015 0017 0018 0019 0023 0032 0035 0036 0037 0038 0039 0041 0042 0044 0048 0053 0057 0060 0061 0064 0068 0069 0073 0074 0075 0076 0077 0078 0079 0081 0082 0083 0085 0096 0098 0099 0100 0101 0102 0103 0112 0114 0115 0116	00	01	PERxxxx	Conversion step aborted
0095 0104 0106 0107 0110 0111	00	64	PERxxxx	PERCON aborted
0105	00	64	CMD0205	PERCON aborted with TERM UNITS=STEP
0003	00	64	PER0003	Insufficient main memory during dynamic linking of prelinked module
	00	01	PER0003	Conversion step aborted due to insufficient main memory during PERCON run

3.4 Interrupting the PERCON run

The PERCON run can be interrupted as follows:

- interactive mode: by means of the interrupt key at the terminal (K2 key) and by entering the command /INFORM-PROGRAM
- in ENTER mode by entering /INFORM-PROGRAM from the console.

After the /INFORM-PROGRAM command has been entered, the message PER0070 is output. Further execution is controlled by entering one of the following operation codes in response.

Operation code	**Meaning**
DISP	The current processing status is output (record/block counter).
CONT	Processing is continued.
START	The current conversion step is aborted and the next conversion step started.
FIN	Reading from the current input file is terminated and continued from the subsequent input file, if there is one.
TERM	The PERCON run is terminated.

If DISP is entered, the same message issued after the /INFORM-PROGRAM command is also output after the information on the current processing status. Further execution again depends on the subsequent entry of an operation code.

Note

The entries DISP, CONT, START, FIN and TERM must not be abbreviated.

3.5 Conversion step

PERCON works in conversion steps. A conversion step is a section of the PERCON run beginning with the specifications for the transfer operation and ending with a completion message from PERCON. Any number of conversion steps can be started in the same PERCON run.

Either the START-CONVERSION statement or the END statement can be used to indicate completion of the entries for a conversion step.
The RESET-INPUT statement resets all previously made entries and formally terminates the conversion step.

Further conversion steps are possible after either the START-CONVERSION or the RESET-INPUT statement, but the END statement terminates the PERCON run.

In the case of file copying, any number of input/output files can be assigned per conversion step. All the statements belonging to the same conversion step are collected and executed in one transfer operation.

In the case of tape editing, only one input tape but several output files can be assigned per conversion step. Every statement in a conversion step is executed immediately, thus enabling the results of a statement to be used in the following statements.

In the case of tape duplication, only one input tape or MF/MV set, but several output tapes or MF/MV sets can be assigned per conversion step. Output is directed to all output tapes simultaneously.

Example

```
/START-PERCON
//ASSIGN-INPUT-FILE...
//ASSIGN-OUTPUT-FILE...
//SELECT-INPUT-RECORDS...
//START-CONVERSION

PER0030 .....
PER0030 ..... ——————————————————————————————————————————————————————  (1)

//ASSIGN-INPUT-FILE
//ASSIGN-OUTPUT-FILE...
//SET-PAGE-LAYOUT...
//END

PER0030 .....
PER0030 ..... ——————————————————————————————————————————————————————  (2)

PER0031 PERCON TERMINATED NORMALLY  ——————————————————————————————————  (3)
```

(1) End of 1st conversion step

(2) End of 2nd conversion step

(3) End of the PERCON run

If a logical error occurs during processing of a conversion step:

- control branches to the next STEP or END statement in procedures and batch jobs;
- the subsequent statements for the next conversion step are interpreted in interactive mode.

An errored conversion step results in abnormal termination (TERM MODE= *ABNORMAL) of the entire PERCON run (branch to /SET-JOB-STEP), even if all subsequent conversion steps were terminated without errors.

3.6 Sequence of statements

First of all the input and output media such as files and tapes must be assigned, so that subsequent statements involving these files and tapes can be executed (statements ASSIGN-INPUT-FILE, ASSIGN-OUTPUT-FILE, ASSIGN-INPUT-TAPE, ASSIGN-OUTPUT-TAPE).

To duplicate a tape, the ASSIGN-INPUT-TAPE statement must precede the ASSIGN-OUTPUT-TAPE statement.

The START-TAPE-PROCESSING statement can be specified only following an ASSIGN-OUTPUT-TAPE statement.

In guided dialog, the statements ASSIGN-OUTPUT-TAPE, CHANGE-INPUT-TAPEPOSITION and START-TAPE-PROCESSING are not available until **after** entry of the ASSIGN-INPUT-TAPE statement.

The CHANGE-INPUT-TAPEPOSITION statement can be specified only following the ASSIGN-INPUT-TAPE statement, but it can precede or follow the ASSIGN-OUTPUT-FILE statement.

The following statements can refer to one, selected or all previously specified output files of a conversion step: START-TAPE-PROCESSING, SELECT-INPUT-RECORDS, SET-RECORD-MAPPING, SET-PAGE-LAYOUT and SET-GROUP-ATTRIBUTES. If a statement is to refer only to one or to selected output files, the associated link names of these output files must be specified in this statement.

If one of the specified statements is specified without a link name, it refers to all already specified ASSIGN-OUTPUT-FILE statements. Any ASSIGN-OUTPUT-FILE statements that were issued later are not affected.

With the exception of the MODIFY-PERCON-OPTIONS statement, all statements are valid for one conversion step only. They must be entered in full, as subsequent additions are not possible.
The MODIFY-PERCON-OPTIONS statement remains valid until it is redefined or until the PERCON run is terminated.

TFT entries generated by means of a /ADD-FILE-LINK command are retained throughout the PERCON run unless the release of the file link names is requested explicitly (REMOVE-FILE-LINK operand in the ASSIGN-INPUT-FILE and ASSIGN-OUTPUT-FILE statements).

Example

```
/CREATE-FILE FILE-NAME=OUT1
/ADD-FILE-LINK LINK-NAME=OUT1,-
/              FILE-NAME=OUTFIL,-
/              ACCESS-METHOD=SAM ————————————————————————————————————————  (1)
/CREATE-FILE FILE-NAME=OUT2
/ADD-FILE-LINK LINK-NAME=OUT2,-
/              FILE-NAME=TESTFIL,-
/              ACCESS-METHOD=SAM ————————————————————————————————————————  (2)
/START-PERCON ———————————————————————————————————————————————————————————  (3)
//ASSIGN-INPUT-FILE FILE=DISK-FILE(NAME=INFIL),-
//                  LINK-NAME=IN1 ————————————————————————————————————————  (4)
//ASSIGN-OUTPUT-FILE LINK-NAME=OUT1 ————————————————————————————————————  (5)
//ASSIGN-OUTPUT-FILE LINK-NAME=OUT2 ————————————————————————————————————  (6)
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=OUT1,-
//                     CONDITION=((15,1)=ALPHA) —————————————————————————  (7)
//START-CONVERSION ——————————————————————————————————————————————————————  (8)
```

(1) The output file OUTFIL is assigned with the link name OUT1 and its file attributes.

(2) The output file TESTFIL is assigned with the link name OUT2 and its file attributes.

(3) PERCON is called.

(4) The file INFIL is assigned as the input file with the link name IN1.

(5) The output file OUTFIL is linked to PERCON via the link name OUT1.

(6) The output file TESTFIL is linked to PERCON via the link name OUT2.

(7) The SELECT-INPUT-RECORDS statement refers to the output file OUTFIL for record selection via the link name OUT1.

(8) The transfer operation is started.

3.7 I/O assignment

The following section describes how disk input and output files and tape input and output files are assigned. The BS2000 commands used for this purpose are described in “Commands” [9].

Priority of assignments

The specification of file attributes for processing purposes using the ADD-FILE-LINK command has priority over specifications in ASSIGN-INPUT-FILE and ASSIGN-OUTPUT-FILE statements and catalog entries. Different rules may apply when storing files in the catalog (e.g. in the case of FOREIGN tapes, the specifications in the catalog entry may be taken over from HDR2). In the case of input files already cataloged, make sure that the specifications in the ADD-FILE-LINK command do not conflict with the catalog entries or the ASSIGN-INPUT-FILE specifications. If this is the case, it is always the ADD-FILE-LINK statements which are interpreted.

The specification of file attributes via ASSIGN-INPUT-FILE and ASSIGN-OUTPUT-FILE statements has priority over catalog entries. Here, too, in the case of input files already cataloged, it is necessary to make sure that the specifications in the ASSIGN-INPUT-FILE statements and the catalog entries do not conflict with each other, because this would otherwise lead to errors.

File attributes which have been assigned neither via the ADD-FILE-LINK command nor via ASSIGN-INPUT-FILE or ASSIGN-OUTPUT-FILE statements are provided with the catalog entries, in so far as there are any.

To summarize, the priority of the assignments can be represented by the following hierarchy:

– ADD-FILE-LINK command

– PERCON control statements

– catalog entry

– PERCON default values

Input file

The following BS2000 commands and PERCON statements are of importance with regard to the assignment of input files:

1. IMPORT-FILE command

   This command must be specified if the file to be processed resides on a private volume and has not yet been cataloged.

2. ADD-FILE-LINK command

   This command must be specified when standard processing operands (e.g. OPEN-MODE=INPUT) are to be modified. Conflicting specifications for the file attributes in the catalog entry result in a file open error. Only one file can be assigned by means of the standard link name PCIN.

3. ASSIGN-INPUT-FILE statement

   This statement must be specified if the standard link name PCIN is not to be used. If there is more than one input file, each one must be assigned by a separate ASSIGN-INPUT-FILE statement and a different link name must be used for each. The files are processed one after the other in the order in which they were specified. The connection to a specified ADD-FILE-LINK command is established by the link name.

If the name of a cataloged file is specified both in the ADD-FILE-LINK command and in the ASSIGN-INPUT-FILE statement, the file name in the ADD-FILE-LINK command has priority. If the file names do not match, message PER0012 is issued.

If input is via SYSDTA or DISKETTE, only the entries in the ASSIGN-INPUT-FILE statement are required.

Output file

The following BS2000 commands and PERCON statements are of importance for assigning output files on disk or tape:

1. CREATE-FILE command

   This command must be specified if the file does not exist.

2. ADD-FILE-LINK command

   This command must be specified if the file attributes
   - of an existing file, which are used for processing by PERCON, are to deviate from the entries in the catalog;
   - of a new file, which have not been specified yet, are to deviate from the PERCON default values.

3. ASSIGN-OUTPUT-FILE statement

   This statement must be specified if the standard link name PCOUT is not to be used. If there is more than one file, each one must be assigned via a separate ASSIGN-OUTPUT-FILE statement and a different link name must be used for each one. Data is written to the different files simultaneously. The connection to a specified ADD-FILE-LINK command is established via the link name.

If the name of a cataloged file is specified both in the ADD-FILE-LINK command and in the ASSIGN-OUTPUT-FILE statement, the file name in the ADD-FILE-LINK command has priority. If the file names do not math, message PER0012 is issued.

If output is via SYSLST or SYSOUT, only the specifications in the ASSIGN-OUTPUT-FILE statement are required.

Input tape

In the case of tape editing, one tape can be assigned; in the case of tape duplication, several tapes can be assigned. For this purpose the VSN must be specified in the ASSIGN-INPUT-TAPE statement. The maximum length of the input blocks is specified in the BUFFER-LENGTH operand. If any block exceeds this length, it is shortened to the length specified in the BUFFER-LENGTH operand.

Output tape

In the case of tape duplication, several output tapes can be assigned. For this purpose the volume serial numbers must be specified in the ASSIGN-OUTPUT-TAPE statement.

3.8 Actions at the OPENC exit

If a file cannot be opened because it was not closed correctly during previous processing (e.g. due to a system crash), PERCON branches to the EXLST exit OPENC. This exit provides the VERIF macro which can be used to unlock the file and restore consistency.

The contents of an ISAM file may be modified as a result of a restore. With the other access methods, the file is merely unlocked, possibly with updating of HIGHEST-USED-PAGE (LAST-PAGE-POINTER). For this reason, PERCON actions at the OPENC exit differ depending on the access method used.

The following actions at the OPENC exit are programmed:

– Non-ISAM files:
  Since file contents are not modified in this case, restore using the VERIF macro is always attempted. The user is informed of the attempt via message PER0092. If an error occurs during the attempted restore, message PER0016 is additionally output.

– ISAM files in interactive mode:
  Message PER0093 prompts the user to specify whether a restore is to be attempted or not. A positive response (YES) results in a restore attempt using the VERIF macro, a negative response (NO) results in quitting the OPENC exit. If an error occurs during VERIF processing, message PER0016 is additionally output.

– ISAM files, non-interactive mode:
  Message PER0094 informs the user that a restore is necessary. Since the user cannot be prompted to confirm, no restore takes place and PERCON quits the OPENC exit.

If a file is restored without error message, PERCON quits the OPENC exit and attempts to open the file again.

If no restore is performed or an error occurs during restore, input files are handled in accordance with the value of the OPEN-ERROR operand of the ASSIGN-INPUT-FILE statement. If the error concerns an output file, the conversion step is aborted.

3.9 Processing cataloged files

ℹ PERCON is capable of processing files with a capacity in excess of 32 GBytes (without having to make changes at the user interface).

3.9.1 File attributes

All the file attributes described below can be assigned via the ADD-FILE-LINK command (see “Commands” [9]).

If no specifications are made and the file attribute cannot be ascertained by reference to the catalog entry or tape labels, the default value defined by PERCON takes effect (see also “Priority of assignments” on page 37.

ACCESS-METHOD

Access method for the file. PERCON can use the access methods SAM, ISAM, UPAM and BTAM.

Default value:

Input files are read using the access method specified in the catalog entry.

The following applies to output files:

File copying in disk files:	ACCESS-METHOD=*ISAM
File copying in tape files:	ACCESS-METHOD=*SAM
Tape editing:	ACCESS-METHOD=*SAM

Format-edited files are always SAM files.

If the output file is to be an ISAM file, the records to be output must be sorted in ascending order of key. This may be the record counter, which can be inserted in the output record by means of a SET-RECORD-MAPPING statement.

BLOCK-CONTROL-INFO

This new file attribute defines the block structure. It is the result of eliminating the PAM key (see page 46) and is relevant for the access methods SAM, ISAM and PAM.

A distinction is made between the following attributes:

PAMKEY:	16 bytes of management information (PAM key) are assigned to each 2-Kbyte block; these 16 bytes do not form part of the 2-Kbyte block.
WITHIN-DATA:	Each logical block contains 12 bytes of system information. Basically speaking, system information comprises the data which is contained in the PAM key when BLOCK-CONTROL-INFO=*PAMKEY is specified.
NO:	The file contains no system information.

The file attribute BLOCK-CONTROL-INFO is part of the catalog entry. New files are given this attribute on the basis of a default specification which depends on the particular version of BS2000, the volume, the class 2 option BLKCTRL and the file access method.

BUFFER-LENGTH

Block size of the file. Files consist of standard blocks (BUFFER-LENGTH=*STD). Nonstandard blocks (BUFFER-LENGTH=*NSTD) can also be specified, but only for tape files.

The block size influences the maximum record length. If the record to be output is too long, the record will be truncated.

For format-edited files, the entry `BUFFER-LENGTH=*STD(SIZE=n)` always applies. The value of SIZE depends on the pubset format:
`SIZE=1` with 2K-formatted pubsets,
`SIZE=2` with 4K-formatted pubsets.
For newly created output files, this value applies by default.

The primary size of a file should always be at least twice the BUFFER-LENGTH value, whereas the secondary allocation value should be at least the BUFFER-LENGTH value.

RECORD-FORMAT

Record format of the files. PERCON supports the record formats FIXED, VARIABLE and UNDEFINED.

Default value:	`RECORD-FORMAT=*VARIABLE`

Format-edited files always use the variable record format.

RECORD-SIZE

Record length. With RECORD-FORMAT=*FIXED, the length of the records to be output is specified; with RECORD-FORMAT=*VARIABLE, the maximum length of the records is specified. If an output record is longer than specified in the RECORD-SIZE operand, the record is truncated. If RECORD-FORMAT=*FIXED is specified and the output record is shorter than specified in the RECORD-SIZE operand, the record is padded with space characters (X'40').

If no record length or record length 0 is specified or if the specified length exceeds the maximum value determined for BUFFER-LENGTH (see the table below), this maximum value is assumed for the length of the output record.

ACCESS-METHOD	**BLOCK-CONTROL-INFO**	**RECORD-FORMAT**	**Maximum value**
SAM without any format editing	PAMKEY	VARIABLE FIXED UNDEFINED	BUFFER-LENGTH - 4 BUFFER-LENGTH BUFFER-LENGTH 1)
	WITHIN-DATA-BLOCK	VARIABLE FIXED UNDEFINED	BUFFER-LENGTH -16 BUFFER-LENGTH -16 BUFFER-LENGTH -16 1)
	NO 2)	VARIABLE FIXED UNDEFINED	BUFFER-LENGTH - 4 BUFFER-LENGTH BUFFER-LENGTH 1)
ISAM	PAMKEY	VARIABLE FIXED	BUFFER-LENGTH BUFFER-LENGTH - 4
	WITHIN-DATA-BLOCK	VARIABLE FIXED	BUFFER-LENGTH BUFFER-LENGTH - 4
PAM 1)	PAMKEY		BUFFER-LENGTH 2048 3)
	WITHIN-DATA-BLOCK		BUFFER-LENGTH - 12
	NO		BUFFER-LENGTH
BTAM		VARIABLE FIXED UNDEFINED	BUFFER-LENGTH 1) BUFFER-LENGTH BUFFER-LENGTH 1)

1) The RECORD-SIZE specification is not used to determine the maximum value.

2) Applies to tape files only.

3) Applies only when the current input file uses ACCESS-METHOD=*UPAM and BLOCK-CONTROL-INFO=*PAMKEY and when no output file specifies BLOCK-CONTROL-INFO=*WITHIN-DATA-BLOCK or *NO.

KEY-POSITION

Position of the ISAM key in the output record (relevant for ISAM files only).

Default values:

RECORD-FORMAT=*FIXED: `KEY-POSITION=1`
RECORD-FORMAT=*VARIABLE: `KEY-POSITION=5`

KEY-LENGTH

Length of the ISAM key in the record (relevant for ISAM files only).

Default value: `KEY-LENGTH=8`

OPEN-MODE

Open mode of the file. The *EXTEND value can be used to add the records to be output to an existing SAM or BTAM file. In the case of SAM files, the records read in can be written back to the input file if both the input file and the output file were opened with OPEN-MODE=*UPDATE. In this case the current output record has the same length as the current input record. If necessary, it is truncated or padded with blanks until the lengths are equal. With ISAM files, the output records can be inserted at the appropriate place by virtue of their key fields, provided OPEN-MODE=*OUTIN (for new files) or OPEN-MODE=*INOUT (for existing files) was specified.

Default values:

For input:		`OPEN-MODE=*INPUT`
For output:	PAM files:	`OPEN-MODE=*OUTIN`
	in all other cases	`OPEN-MODE=*OUTPUT`

3.9.1.1 4K pubsets

4K pubsets consist of disks with a 4-Kbyte block structure. The block size of the file must be a multiple of 4K, i.e. BUFFER-LENGTH=*STD(SIZE=n), where n is an even number. In addition, the block structure must not be BLOCK-CONTROL-INFO=*PAMKEY.

In the case of output files, the default value for the BUFFER-LENGTH is dependent on the pubset on which the file is created:

– 2K-formatted pubsets: `BUFFER-LENGTH=*STD(SIZE=1)`;

– 4K-formatted pubsets: `BUFFER-LENGTH=*STD(SIZE=2)`.

3.9.1.2 NK4-capable ISAM

A NK4 ISAM file has the following file attributes:

```
BUFFER-LENGTH=*STD(SIZE=n)                where n is an even number
BLOCK-CONTROL-INFO=*WITHIN-DATA-4K-BLOCK
```

Both file attributes can be specified by means of the ADD-FILE-LINK command or FILE macro.

A NK4 ISAM file is created under the following conditions:

```
Pubset:               4K
BUFFER-LENGTH:        not specified /
                      *STD(SIZE=n)
                      where n is an even number
BLOCK-CONTROL-INFO:   not specified /
                      *WITHIN-DATA-BLOCK /
                      *WITHIN-DATA-4K-BLOCK
```

If FILE-ATTRIBUTES=*INPUT-FILE is specified in the ASSIGN-OUTPUT-FILE statement, the block size and BLOCK-CONTROL-INFO attributes of the input file are transferred to the output.

If the input file is an ISAM file and specifies BLOCK-CONTROL-INFO=*WITHIN-DATA-2K-BLOCK, it is entered with BLOCK-CONTROL-INFO=*WITHIN-DATA-BLOCK. This means that the file can be set up with WITHIN-DATA-2K-BLOCK or WITHIN-DATA-4K-BLOCK, depending on the pubset on which it is created.

3.9.2 PAM key elimination

BS2000/OSD supports FBA disks (fixed block architecture) with a fixed block size of 2 Kb, 4 Kb, and so forth. These fixed block sizes prevent simple accommodation of the PAM keys, which will therefore be done away with (PAM key elimination).

The system information previously contained in the PAM key will then either be accommodated in a data block within a special block control field or it will be omitted. The block control field occupies the first 12 bytes of the data block and also contains the coded field identification used for checking for gaps. This block control field reduces the maximum usable record length of the data block.

Depending on the file access method defined in the ADD-FILE-LINK command, new file formats come into use as a result of PAM key elimination, which is detected by means of the BLOCK-CONTROL-INFO parameter. Further information on these new file formats can be found in the “Systems Support” manual [3].

The following sections describe a number of special considerations for processing PAM files, which arise in the event of PAM key elimination.

SAM, ISAM and BTAM files are not affected by the elimination of the PAM key.

3.9.2.1 PAM input files

– Processing of empty blocks (EMPTY-BLOCK operand)

  The EMPTY-BLOCK operand in the ASSIGN-INPUT-FILE statement enables the user to control whether an empty block is skipped or whether an empty block (before the HIGHEST-USED-PAGE (LAST-PAGE-POINTER) in a file) is to be processed as an input record with the contents binary zero (X’00’).

– Block structure (BLOCK-CONTROL-INFO operand)

  BLOCK-CONTROL-INFO=*PAMKEY

  Each logical block forms an input record with the length specified by BUFFER-LENGTH. For the gap test, the coded file identification (CFID) given in the PAM key of the first PAM block of the logical block is compared with the one entered in the catalog.
  If they are not identical, the logical block is processed in accordance with the value for EMPTY-BLOCK.

  If ACCESS-METHOD=*PAM and BLOCK-CONTROL-INFO=*PAMKEY have been specified for the output file, the user data areas in the PAM keys are passed to the output file, but they do not form part of the record.

  Note

  If BLOCK-CONTROL-INFO=*PAMKEY has been specified for the PAM input file and no output file has the specification BLOCK-CONTROL-INFO=*WITHIN-DATA or *NO, each PAM block forms an input record with a length of 2048 bytes. The coded file identification given in the PAM key of each PAM block is used for the gap test. If a comparison yields discrepancies, the PAM block is processed in accordance with the EMPTY-BLOCK value specified in the ASSIGN-INPUT-FILE statement.

  BLOCK-CONTROL-INFO=*WITHIN-DATA

  Each logical block forms an input record with the length BUFFER-LENGTH minus 12. The block control field is not part of the input record. For the gap test, the CFID in the block control field of the logical block is used.
  If a comparison yields discrepancies, the logical block is processed in accordance with the value specified for the EMPTY-BLOCK operand.

  BLOCK-CONTROL-INFO=*NO

  Each logical block forms an input record with the length specified by BUFFER-LENGTH. Since no gap test can be performed, all logical blocks are processed. Blocks at the end of this file which do not belong to it (block number greater than HIGHEST-USED-PAGE (LAST-PAGE-POINTER)) are not transferred.

3.9.2.2 PAM output files

The way in which PERCON handles PAM output files depends primarily on the BLOCK-CONTROL-INFO file attribute.

BLOCK-CONTROL-INFO=*PAMKEY

Each output record has the size of a logical block. The user data areas of the PAM keys are transferred if ACCESS-METHOD=*PAM has been specified for the current input file, otherwise they are overwritten with binary 0.

Note

If the current input file is a PAM file with BLOCK-CONTROL-INFO=*PAMKEY and no output file has the specification BLOCK-CONTROL-INFO=*WITHIN-DATA or *NO, each output record has the same size as a PAM block (fixed length of 2048 bytes). The blocking factor n specifies the number of PAM blocks to be chained to form a logical block.

BLOCK-CONTROL-INFO=*WITHIN-DATA

Each output record is entered at start of block plus 12, resulting in an output record with a length of BUFFER-LENGTH minus 12.

BLOCK-CONTROL-INFO=*NO

Each output record has the same size as a logical block (i.e. the fixed length of BUFFER-LENGTH).

3.9.2.3 Examples of copying PAM files

Example 1

A PAM file with BLOCK-CONTROL-INFO=*PAMKEY is to be converted to a PAM file with the same file format. In the input file, 2 PAM blocks form a logical block; in the output file, 4 PAM blocks are to form a logical block.

In the course of this conversion, the user data in the PAM keys is also transferred, but it does not form part of the record and therefore cannot be changed.

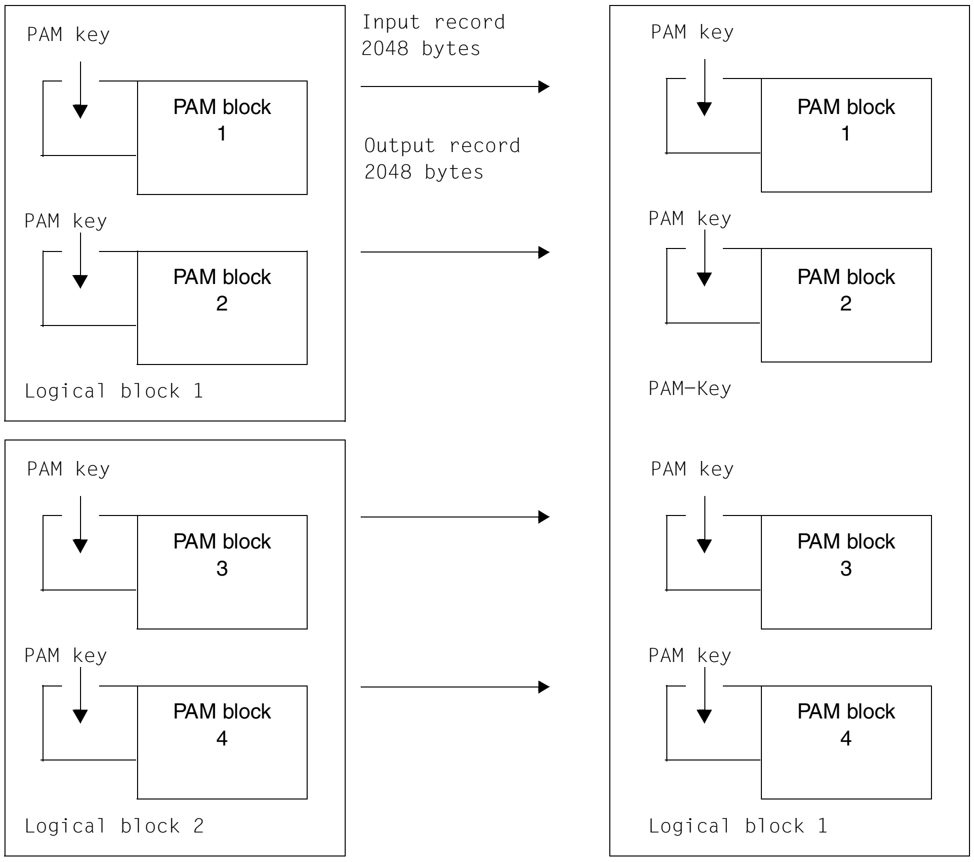
The following file attributes are of importance for this conversion:

Input file:

```
ACCESS-METHOD=*PAM
BUFFER-LENGTH=*STD(SIZE=2)
BLOCK-CONTROL-INFO=*PAMKEY
```

Output file:

```
ACCESS-METHOD=*PAM
BUFFER-LENGTH=*STD(SIZE=4)
BLOCK-CONTROL-INFO=*PAMKEY
```



Example 2

A PAM file without a PAM key is to be converted to a PAM file with a PAM key. In the input file, two standard blocks form one logical block; in the output file, four PAM blocks are to form one logical block.

The size of the input record is calculated as follows:

```
n x standard size - block control field = 2 x 2048 - 12 = 4084 bytes.
```

The block control field is not part of the input record.

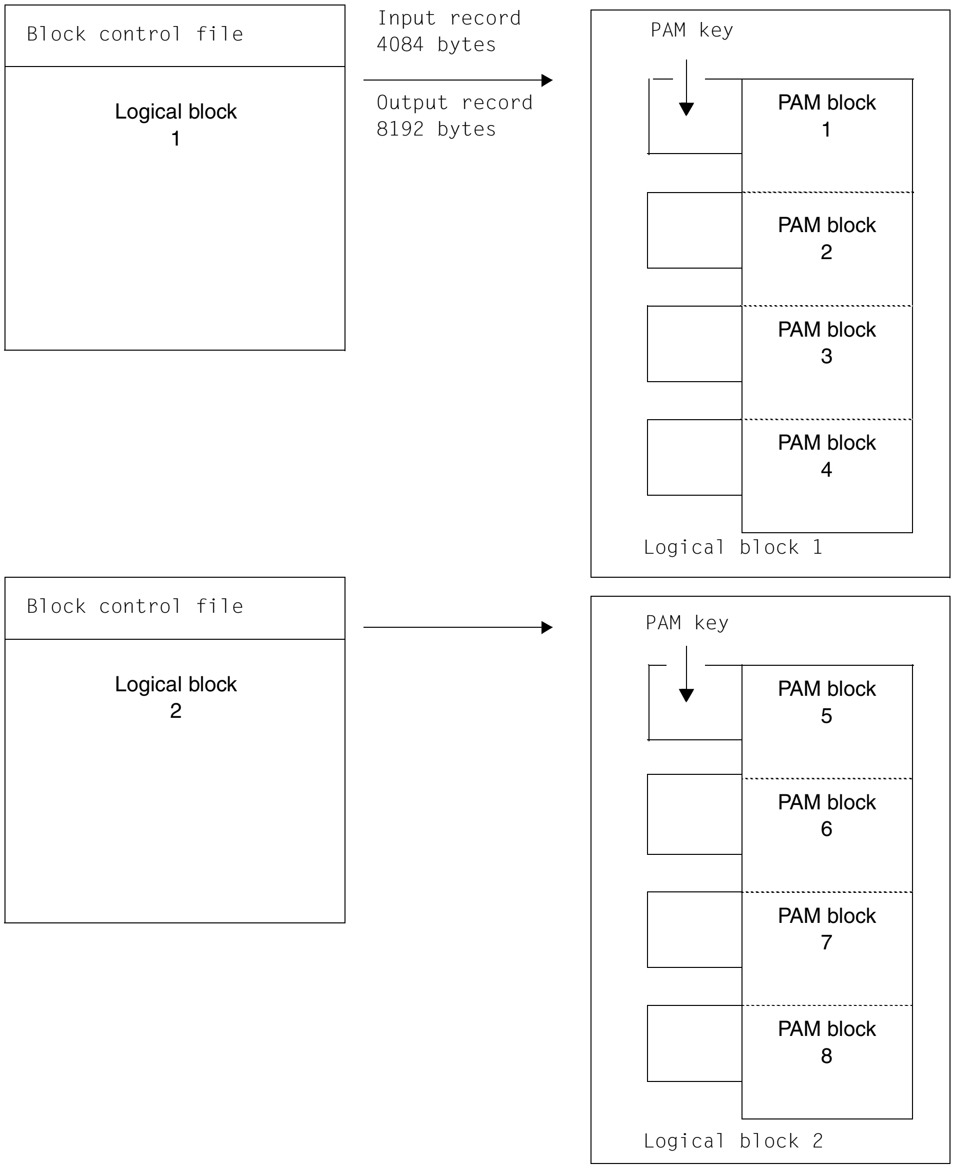
The size of the output record is the product of:

```
m x standard size = 4 x 2048 = 8192 bytes
```

During this conversion, the user data areas in the PAM keys are overwritten with the value X’00’.

The following file attributes are of importance for this conversion:

Input file:	Output file:
`ACCESS-METHOD=*PAM`	`ACCESS-METHOD=*PAM`
`BUFFER-LENGTH=*STD(SIZE=2)`	`BUFFER-LENGTH=*STD(SIZE=4)`
`BLOCK-CONTROL-INFO=*WITHIN-DATA`	`BLOCK-CONTROL-INFO=*PAMKEY`



Example 3

A PAM file with BLOCK-CONTROL-INFO=*PAMKEY is to be converted to a PAM output file with BLOCK-CONTROL-INFO=*NO and without a PAM key. In the input file, two PAM blocks form one logical block; in the output file, the size of a logical block must be 4096 bytes.

In the course of this conversion, the user data in the PAM keys is not included in the transfer.

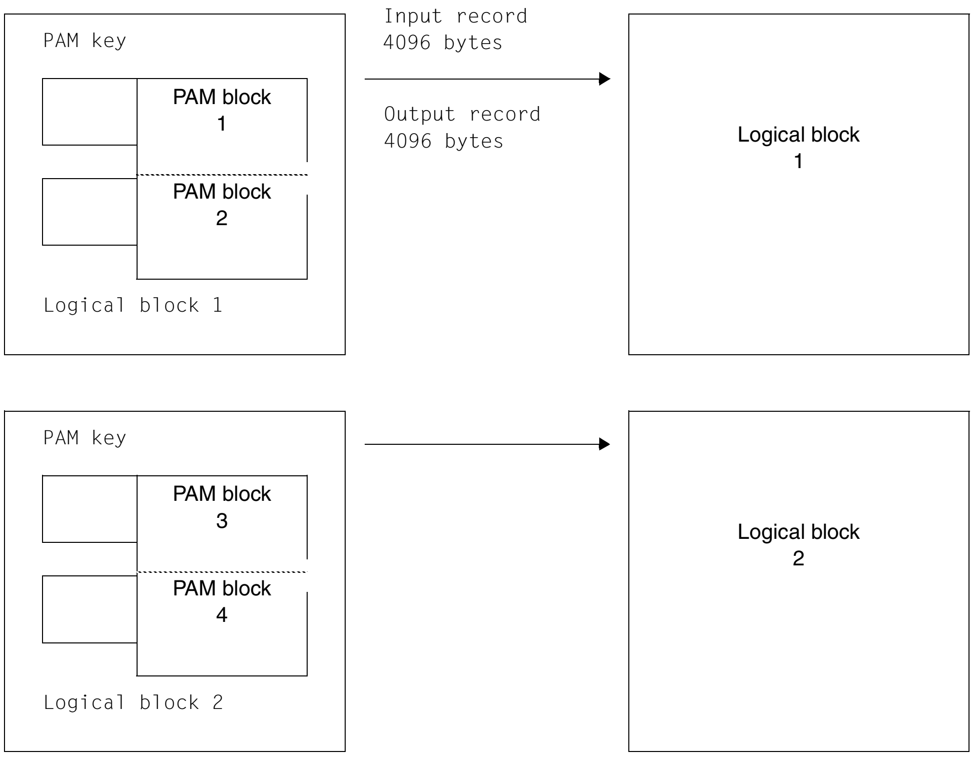
The following file attributes are of importance for this conversion:

Input file:

```
ACCESS-METHOD=*PAM
BUFFER-LENGTH=*STD(SIZE=2)
BLOCK-CONTROL-INFO=*PAMKEY
```

Output file:

```
ACCESS-METHOD=*PAM
BUFFER-LENGTH=*STD(SIZE=2)
BLOCK-CONTROL-INFO=*NO
```



Example 4

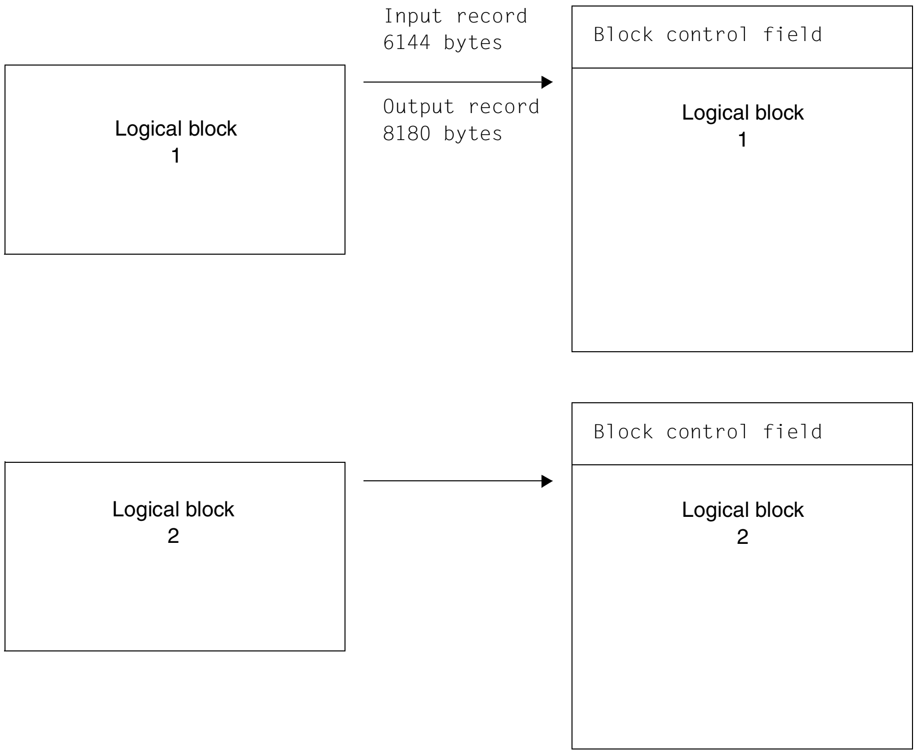
A PAM file without a PAM key is to be converted to a PAM output file of the same sort. The input and output files, however, have different block sizes and block structures, as shown below:

Input file:	Output file:
`ACCESS-METHOD=*PAM`	`ACCESS-METHOD=*PAM`
`BUFFER-LENGTH=*STD(SIZE=3)`	`BUFFER-LENGTH=*STD(SIZE=4)`
`BLOCK-CONTROL-INFO=*NO`	`BLOCK-CONTROL-INFO=*WITHIN-DATA`

The size of an output record is calculated as follows:

```
   n x standard size - block control field = 4 x 2048 -12 = 8180 bytes.
```

The output record is written to the logical block, behind the block control field.



3.9.3 Duplicating secondary keys

When converting NK-ISAM files, PERCON will not automatically copy secondary keys that may exist in the input files to the output files. If required, however, PERCON can obtain the secondary keys of input files and reconstruct them in the output files after conversion. Since PERCON usually processes more than one input file and more than one output file, the user has to specify whether this function is to be activated and for which files.

The reconstruction of the secondary keys for the output files is performed as the last action before completing a conversion step, provided no event has occurred up to this point that would require an abnormal termination, i.e. all output files must have been created without error.

In this case, PERCON determines the defined name, position and length values as well as the DUPKEY attribute of all secondary keys of the specified input file, subsequently using these values to reconstruct all secondary keys for the output file.

If any of the secondary keys to be duplicated already exists in the output file and the values defined for its position, length and DUPKEY attribute match those in the input file, processing is continued without further action.

If an error occurs during secondary-key duplication, one or more of the messages PER0099 thru PER0103 will be output and the conversion step will be aborted.

If the input file is a NK-ISAM file without secondary keys, however, processing is continued normally after issuing a PER0097 warning.

The following error situations may occur when duplicating secondary keys:

1. The input or output file is not an NK-ISAM file.
2. The input file has no secondary keys (results in PER0097 warning).
3. The output file contains duplicate primary keys or logical flags or value flags.
4. The output file already contains a secondary key to be duplicated, but the values for its position, length and DUPKEY attribute differ from those in the input file.
5. Duplicating the secondary keys would cause the maximum number of 30 secondary keys to be exceeded.
6. The secondary key has been defined with DUPKEY=*NO, but identical secondary key values occur in various records of the output file.
7. At least one record of the output file is too short to accommodate the complete secondary key.
8. Other unexpected system errors.

The ALTERNATE-INDEX operand is ignored for the Edit basic function.

3.10 PERCON in the XS environment (31-bit addressing)

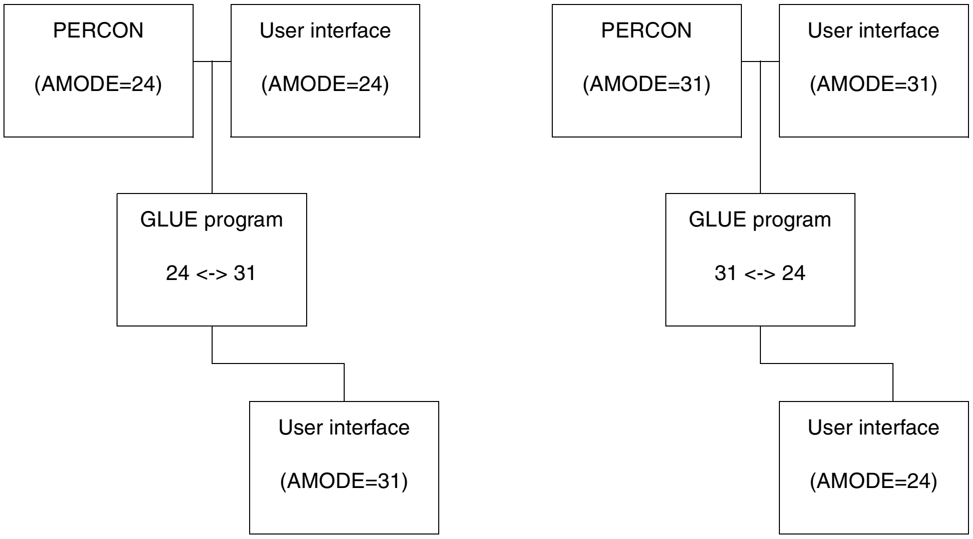
PERCON is executable in both 24- and 31-bit addressing modes. 31-bit addressing should be used if virtual memory requirements are correspondingly high (> 16 Mbytes) and PERCON is to be called as a subprogram or user modules are to be connected.

When called autonomously PERCON executes in 24-bit addressing mode unless otherwise specified. If 31-bit addressing mode is to be activated, PERCON must be started with `/START-PERCON PROGRAM-MODE=ANY`.

If PERCON is called as a subprogram and if the linking loader module PCROOT (LLM) linked to the main program (see page 171) has to change addressing mode (for instance, because PERCON has been loaded into the upper address space as a subprogram), the old addressing mode is reinstated when the user returns to the main program.

The following points must be borne in mind with regard to user interfaces:

1. PERCON does not change the addressing mode when calling a user interface.
   If PERCON and the user interface have different addressing modes:
   - the user interface must be adapted to have the same addressing mode as PERCON, or
   - if this is not possible, a “GLUE program” must be inserted in order to standardize the addressing mode.



2. Irrespective of the addressing mode, PERCON uses the 31-bit system interface (PARMOD=31). For this reason a 31-bit file control block (FCB) is offered for label processing in the case of user interfaces.

3.11 Use of extended character sets in PERCON

Each computer (host) and terminal operates using a set of letters, numbers and characters from which words and other elementary components of a language are built up, namely the **character set**.

These character sets can be extended to offer country-specific character representations, such as umlauts (German) or accents (French), at one and the same time within one character set. Unicode is the “fully-developed” form of an extended character set and includes all known text characters in the world in a single character set (see the section “Use of UNICODE character sets in PERCON” on page 61).

A **coded character set (CCS)** is the unambiguous representation of the characters of a character set in binary form. The contents of a coded character set and its rules (for example, the sorting sequence and conversion rules) are laid down in international standards.

Example:	The “ä” character is represented in the coded character set EBCDIC.DF.03-DRV (German Reference Version) by the byte X’FB’, in EBCDIC.DF.04-1 by X’43’.

Each coded character set, or code for short, is defined by its unambiguous name (**coded character set name, CCSN**).

Example:	The code EBCDIC.DF.03-IRV (International Reference Version) has the name “EDF03IRV”.

A list of the existing codes can be found in the appendix of the “XHCS” manual [2].

Extended codes supplement the existing

7-bit codes	EBCDIC.DF.03 for hosts and ISO646 for terminals with roughly 90 characters used

with

8-bit codes	EBCDIC.DF.04-x for hosts and ISO8859-x for terminals with roughly 190 characters used

It is possible to use several codes in parallel.

Requirements

The software product **XHCS** (e**X**tended **H**ost **C**ode **S**upport, subsystem XHCS-SYS) is required for generating extended codes in the host and for transferring data between the host and terminal. The “XHCS” manual [2] contains a detailed description of the principles and functions of XHCS and a list of the code tables and names of standard codes.

“8-bit terminals” must serve as the hardware for input/output of extended character sets to terminals. The 8-bit capability of terminals is tested using the software component VTSU.

3.11.1 PERCON-specific application of extended character sets

In order to be able to correctly interpret the data transferred to interfaces (statements, input records, output records), the user must know which CSS they belong to.

If it is not possible to associate a CCSN with the transfer values of an interface, internal processing is conducted using the mode active before XHCS was introduced.

When editing a file, the table of printable characters of the output CCS (if available) is used. Nonprintable characters are replaced by space characters. If the file does not have an output CSS, the characters X’00’ to X’3F’ are replaced by space characters as before.

PERCON does not support ISO-CCS.

File-oriented input

PERCON respects the CCS of the input source.

The CCSN must not change within an input source assigned via ASSIGN-INPUT-FILE. When reading in via SYSDTA, this must be taken into account when there is a change of assignment.

Files on private disks likewise have no CCSN, because there is no space available for one in their catalog entry.

Volume-oriented input

The blocks of the tape are made available without CCS.

The tape code is defined using the CODE operand of the ASSIGN-INPUT-TAPE statement:

CODE=*EBCDIC:
No code conversion takes place during reading.

CODE=*ISO7:
The tape is generated in ISO6461-IRV. All the bytes of a block are recoded to the EBCDI code during reading.

CODE=*OWN:
The tape code is defined using a conversion table. All the bytes of a block are recoded during reading.

File-oriented output

– Cataloged files
  FILE-ATTRIBUTES=*STD
  The CCSN recorded in the catalog entry is used. If there is not yet a catalog entry, it is now generated by PERCON using the CCSN of the first input of the conversion step.

  FILE-ATTRIBUTES=*INPUT-FILE
  The file is assigned the CCSN of the first input of the conversion step.

– SYSOUT
  If SYSOUT is assigned to the terminal, the CCSN of the first input of the conversion step is used as the output CCSN.

  If SYSOUT is assigned to a cataloged file, the CCSN of this file is used.

– SYSLST
  The CCSN of the SYSLST file is used.

– No CCSN can be stored in the catalog entry for files on private disks.

Volume-oriented output

No CCS is assigned to the blocks of the tape.

The tape code is defined using the CODE operand of the ASSIGN-OUTPUT-TAPE statement.

CODE=*EBCDIC:
No code conversion takes place during writing.

CODE=*ISO7:
The tape is generated in ISO6461-IRV. All the bytes of a block are recoded during writing.

CODE=*OWN:
The tape code is defined using a conversion table. All the bytes of a block are recoded during writing.

3.11.2 Processing functions

During processing, PERCON takes into account the statement CCS, the input CCS and the output CCS. If all the CCSs have the same name, there are no special rules to follow. If different names are assigned, the following rules apply:

Input CCS not same as output CCS

If the input CCS is not the same as the output CCS, the following points must be borne in mind:

The prerequisite for a conversion is that the input CCS and the output CCS are compatible.

If no SET-RECORD-MAPPING statement is specified, the characters of the input record are converted from the input CCS to characters of the output CCS during transfer to the output area, as long as the input file has a CCS which is not EDF03IRV.

If FILLER=*INPUT is specified in the SET-RECORD-MAPPING statement, the output record is prefilled with the input record, irrespective of its CCS.

The transfer of input record fields with INPUT-FORMAT=*CHARACTER is dependent on the OUTPUT-FORMAT:

– OUTPUT-FORMAT=*CHARACTER / *HEXADECIMAL / *BINARY
  The input field is converted to the output CCS and edited as necessary.

– OUTPUT-FORMAT=*TRANSLATION
  The bytes of the input fields are recoded in accordance with a conversion table and transferred to the output field. Input and output CCSNs are not taken into consideration. The table can take the form of a file or pairs of values. If the bytes are specified as c-strings via INPUT-CHARACTER / OUTPUT-CHARACTER, they are interpreted as characters of the statement CCS and entered accordingly into the conversion table.

– OUTPUT-FORMAT=*NO-TRANSLATION
  This format is used when the input field is not to be converted to the output CCS (e.g. with packed numbers).

– OUTPUT-FORMAT=*UNICODE-TRANSLATION
  The bytes of the input field are converted either from a non-Unicode format to a Unicode format or from a Unicode format to a non-Unicode format and transferred to the output field. When the Unicode variant UTF-16 is used, normalization can take place in addition to conversion.

Statement CCS not same as input CCS

If the statement CCS is not the same as the input CCS, the following points must be borne in mind:

If c-strings are used in the SELECT-INPUT-RECORDS statement, the statement CCS and input CCS must belong to the same code family and all the characters of the c-strings must be available in the input CCS.

Statement CCS not same as output CCS

If the statement CCS is not the same as the output CCS, the following points must be borne in mind:

If c-strings are used in the SET-GROUP-ATTRIBUTES, SET-RECORD-MAPPING or SET-PAGE-LAYOUT statement, the statement CCS and output CCS must belong to the same code family and all the characters of the c-strings must be available in the output CCS.

Input CCS(n) not same as input CCS(n+1)

If the CCS of the current input file is not the same as that of the previous input file of the conversion step, the following point must be borne in mind:

The group break is defined by comparing fields of consecutive input records, irrespective of the CCS.

When comparing with respect to ASCENDING or DESCENDING using the SELECT-INPUT-RECORDS statement, the input CCS is not taken into account.

3.12 Use of UNICODE character sets in PERCON

Unicode is a standardized alphanumeric character set and includes all known text characters in the world in a single character set. PERCON enables whole records or parts of records (see the SET-RECORD-MAPPING statement on page 151) which are not encoded in Unicode to be converted to or from a Unicode format. To permit this, the input and/or the output file must be assigned a Unicode format. The input and output files must both be SAM files. If a different access method (ISAM, BTAM or PAM) is used, the message PER0115 is issued and conversion is aborted. Automatic conversion takes place whenever the input and output files are assigned different Coded Character Set Names (CCSNs). The different CCSNs must be compatible for conversion to take place.

Note

In the case of an output file with the CCSN UTF-16, a check is made at the start of conversion to see whether the length of the output field is even. If the length of the output field is not even, conversion is aborted and the message PER0116 is issued.

As Unicode data in non-normalized form can exist, PERCON also offers an option for normalizing this data, i.e. conversion to composite character representation, see also the section “Normalization” on page 64.

In PERCON V2.9A, only files on disk or tape (catalogued files) can be used as input or output files with a Unicode CCS. Input files on SYSDTA and output files on SYSOUT or SYSLST are rejected with the error message PER0112. The SET-GROUP-ATTRIBUTES and SET-PAGE-LAYOUT statements cannot be used for output files with a Unicode CCSN. These statemetns are rejected with the error message PER0118.

PERCON supports the Unicode variants UTF-16, UTF-8 and UTFE which are offered by XHCS V2.0 (see XHCS (BS2000/OSD) [2]) (see the manual “Unicode in BS2000/OSD” [14]).

Converting files

The record length can change when conversion takes place from a non-Unicode format to a Unicode format or vice versa.
In the case of **variable-length records**, the length of the output record is automatically adjusted (split or shortened). This can differ from record to record. The user can specify a maximum record length for the output file. When this length is exceeded, the warning PER0009 is issued once, and the output records are truncated on the right and processed further.

In the case of **fixed-length records**. the user has the following options for adjusting the record length:

– The output record is assigned a variable record length (RECORD-FORMAT= *VARIABLE(...) in the ADD-FILE-LINK statement). The user can then specify a maximum record length “externally”. When this length is exceeded, the warning PER0009 is issued once, and the output record is truncated on the right and processed further.
  When RECORD-SIZE=0, the maximum value for the output record length specified in the table on page 43 is adapted. This value is generally high enough to accommodate the converted record completely.

– The output record is assigned a fixed record length (RECORD-FORMAT=*FIXED(...) in the ADD-FILE-LINK statement). The length is defined by the user. If the output field is too long, unused bytes are padded with Unicode the fill character when conversion to a Unicode format takes place. When conversion takes place to a non-Unicode format, the fill character in the output record’s code is used for padding purposes. In both cases the default is a blank. If the output record is not long enough for conversion, the warning PER0009 is issued once, and the output record is truncated on the right and processed further.
  When output takes place in Unicode format, the user can use the blank or the *NIL character as the Unicode filler (see UNICODE-FILLER in the ASSIGN-OUTPUT-FILE statement). However, if a sort is to follow, you are urgently recommended to use blanks for padding purposes as the *NIL character is ignored by SORT.

Note

In the case of record selection using SELECT-INPUT-RECORDS, characters in Unicode format cannot be specified as the selection criterion. The hexadecimal encoding of the Unicode characters can be used for the comparison operators equal to or not equal to. However, the comparison operators greater than or less than cannot be used in a meaningful way because they always relate to the characters’ hexadecimal encoding. They may not be confused with the actual sorting sequence of the characters.

Converting partial records

The PERCON statement SET-RECORD-MAPPING enables individual sections of a record to be converted and/or (if the Unicode variant UTF-16 is being used) normalized. To permit this, the OUTPUT-FORMAT operand must be assigned the value *UNICODE-TRANSLATION in the SET-RECORD-MAPPING statement. If this specification is missing, no conversion or normalization of the section concerned takes place.

When a non-Unicode format is converted to a Unicode format or vice versa, the length of the section to be output can change. This length change must be borne in mind when the length of the output field is specified (OUTPUT-LENGTH). If the output field is too long, the field will be padded with the Unicode fill character when conversion takes place to a Unicode format. When conversion is to a non-Unicode format, the fill character in the code of the output record is used for padding. In both cases the default is a blank. If the output record is too short, the warning PER0009 is issued once, and the field is truncated on the right and processed further.

Normalization

The encoding of a base character with a diacritic can vary in Unicode. A diacritic is an additional character (e.g. an accent) used to define how a letter is pronounced or stressed. Consequently several encodings can exist for one character in Unicode. The character “Ä”, for example, can also be written as a string consisting of “A” and “°”. Under certain circumstances this characteristic of Unicode can prove to be a hindrance for programming. To permit a uniform format to be assigned to the same characters with different encoding, PERCON offers the normalization function COMPOSED. COMPOSED combines a base character with the associated diacritic to form a single character. However, normalization can take place only if the input file and/or the output file is assigned the Unicode variant UTF-16.

The following format combinations are possible:

– The Unicode variant UTF-16 is only assigned to the input file.
  When normalization is requested, first normalization takes place and then conversion.
– The Unicode variant UTF-16 is only assigned to the output file.
  When normalization is requested, first conversion takes place and then normalization.
– The Unicode variant UTF-16 is assigned to both the input file and the output file.
  Conversion serves only for the purpose of normalization.
– The Unicode variant UTF-16 is assigned to neither the input file nor the output file.
  The requested normalization is ignored.

Note

Normalization does not take place automatically; it must always be requested by the user (see UNICODE-NORMALIZE in the ASSIGN-OUTPUT-FILE statement). The normalization procedure is very time-consuming, and the user should consequently only request it when it is really necessary.

3.13 PERCON and ACS

When using the subsystem ACS (Alias Catalog Service), the following should be observed:

1. If a user-created TFT entry exists, there is no way to verify whether the file name has been replaced or not. Therefore, ACS is interrupted (HOLD-ALIAS-SUBSTITUTION) if the file name from the TFT entry is used (e.g. SHOW-FILE-ATTRIBUTES).
2. If PERCON creates a TFT entry using the FILE macro, ACS continues to be in the state it had at the start of PERCON, i.e. file names can be replaced. In this case, no distinction is made between TFT entries created by the user and by PERCON.
3. If the user specifies a file name in an ASSIGN statement and a TFT entry already exists, he has to check whether the file name may have been replaced. Any file name comparison will make use of the file name specified in the statement irrespective of replacements, if any.
4. Changing the state of ACS (e.g. by loading, interrupting etc.) after creating TFT entries and before starting PERCON may result in ambiguities as to the file names actually used.

4 PERCON statements

This chapter describes the statements which can be entered during the PERCON run. The overview of the PERCON statements and of the SDF standard statements, the comparison of SDF and ISP statements, and the description of the syntax of the SDF user interface and of the formats, are followed by a description of the PERCON statements in alphabetical order.

4.1 Overview of PERCON statements

Statements can be entered in guided or unguided dialog, from procedure files or in batch mode. A detailed description of the input options can be found in the manual “Introductory Guide to the SDF Dialog Interface” [6]).

In guided dialog, only those statements which are legal at that particular moment are offered to the user. For example, START-TAPE-PROCESSING is offered only after ASSIGN-INPUT-TAPE (see section “Sequence of statements” on page 35).

Statement	**Meaning**
ASSIGN-INPUT-FILE	Assign an input file
ASSIGN-INPUT-TAPE	Assign an input tape
ASSIGN-OUTPUT-FILE	Assign an output file
ASSIGN-OUTPUT-TAPE	Assign an output tape
CHANGE-INPUT-TAPEPOSITION	Position the input tape
END	Start the conversion run, terminate PERCON
MODIFY-PERCON-OPTIONS	Control message output
RESET-INPUT	Reset statements
SELECT-INPUT-RECORDS	Define selection conditions
SET-GROUP-ATTRIBUTES	Define group break conditions
SET-PAGE-LAYOUT	Define the output format (page and line editing)
SET-RECORD-MAPPING	Define the output record format
START-CONVERSION	Start a conversion step
START-TAPE-PROCESSING	Control tape output

The SDF standard statements may be specified during the PERCON run. They are not described in the present manual. A description may be found in the manual “Introductory Guide to the SDF Dialog Interface” [6].

4.2 Comparison of SDF and ISP statements

The following table lists the statements in SDF format together with the corresponding statements in ISP format and a brief description of the most important differences.

SDF statement	ISP equivalent	Differences in function
ASSIGN-INPUT-FILE	FILIN	File attributes which are to differ from the default values: – must be assigned via /ADD-FILE-LINK command in SDF format; – can be assigned via /FILE command or a FILIN statement in ISP format.
ASSIGN-INPUT-TAPE	VOLIN	---
ASSIGN-OUTPUT-FILE	FILOUT	File attributes which are to differ from the default values: – must be assigned via /ADD-FILE-LINK command in SDF format; – can be assigned via /FILE command or a FILOUT statement in ISP format.
ASSIGN-OUTPUT-TAPE	VOLOUT	---
CHANGE-INPUT-TAPEPOSITION	POSIT	---
END	END/HALT	---
MODIFY-PERCON-OPTIONS	PARAM	---
RESET-INPUT	RESET	In SDF format this function is independent of the input medium.
SELECT-INPUT-RECORDS	SELECT	---
SET-GROUP-ATTRIBUTES	GROUP	---
SET-PAGE-LAYOUT	FORMAT	---
SET-RECORD-MAPPING	RECORD	---
START-CONVERSION	START	---
START-TAPE-PROCESSING	EDIT	---

4.3 Notational conventions

These notational conventions are based on SDF V4.5A.The syntax of the SDF command/statement language is explained in the following three tables.

Table 1: Metasyntax

The meanings of the special characters and the notation used to describe command and statement formats are explained in Table 1.

Table 2: Data types

Variable operand values are represented in SDF by data types. Each data type represents a specific set of values. The number of data types is limited to those described in Table 2.

The description of the data types is valid for the entire set of commands/statements. Therefore only deviations (if any) from the attributes described here are explained in the relevant operand descriptions.

Table 3: Suffixes for data types

Data type suffixes define additional rules for data type input. They may contain a length or interval specification, restrict the set of permitted values (suffix beginning with *without*), extend it (suffix beginning with *with*), or declare a particular specification as mandatory (suffix beginning with *mandatory*). The following short forms are used in this manual for data type suffixes:

generation	gen
lower-case	low

Metasyntax

Representation	Meaning	Examples
UPPERCASE LETTERS	Uppercase letters denote keywords. The keywords for constant operand values begin with *	**HELP-SDF** **SCREEN-STEPS** = ***NO**
UPPERCASE LETTERS in boldface	Uppercase letters printed in boldface denote guaranteed or suggested abbreviations of keywords.	**GUID**ANCE-**MODE** = ***Y**ES
=	The equals sign connects an operand name with the associated operand values.	**GUID**ANCE-**MODE** = ***NO**
< >	Angle brackets denote variables whose range of values is described by data types and suffixes (see table 2 and 3).	**SYNTAX-F**ILE = <filename 1..54>
Underscoring	Underscoring denotes the default value of an operand.	**GUID**ANCE-**MODE** = ***NO**
/	A slash serves to separate alternative operand values.	**NEXT-FIELD** = ***NO** / ***Y**ES
(…)	Parentheses denote operand values that initiate a structure.	,**UNGUID**ED-**DIA**LOG = ***Y**ES (...) / ***NO**
[ ]	Square brackets denote operand values which introduce a structure and are optional. The subsequent structure can be specified without the initiating operand value.	**SELECT** = [***BY-ATTR**IBUTES](...)
Indentation	Indentation indicates that the operand is dependent on a higher-ranking operand.	,**GUID**ED-**DIA**LOG = ***Y**ES (...) ***Y**ES(...) | **SCREEN-STEPS** = ***NO** / | ***Y**ES

Table 1: Metasyntax (part 1 of 2)

Representation	Meaning	Examples
|	A vertical bar identifies related operands within a structure. Its length marks the beginning and end of a structure. A structure may contain further structures. The number of vertical bars preceding an operand corresponds to the depth of the structure.	SUPPORT = *TAPE(...) *TAPE(...) | VOLUME = *ANY(...) | *ANY(...) | | ...
,	A comma precedes further operands at the same structure level.	GUIDANCE-MODE = *NO / *YES ,SDF-COMMANDS = *NO / *YES
list-poss(n):	The entry “list-poss” signifies that a list of operand values can be given at this point. If (n) is present, it means that the list must not have more than n elements. A list of more than one element must be enclosed in parentheses.	list-poss: *SAM / *ISAM list-poss(40): <structured-name 1..30> list-poss(256): *OMF / *SYSLST(...) / <filename 1..54>
Abbreviation:	The name that follows represents a guaranteed alias for the command or statement name.	**HELP-SDF** Abbreviation: **HPSDF**

Table 1: Metasyntax (part 2 of 2)

Data types

Data type	Character set	Special rules
alphanum-name	A…Z 0…9 $, #, @	
cat-id	A…Z 0…9	Not more than 4 characters; must not begin with the string PUB
command-rest	freely selectable	
composed-name	A…Z 0…9 $, #, @ hyphen period catalog ID	Alphanumeric string that can be split into multiple substrings by means of a period or hyphen. If a file name can also be specified, the string may begin with a catalog ID in the form :cat: (see data type filename).
c-string	EBCDIC character	Must be enclosed within single quotes; the letter C may be prefixed; any single quotes occurring within the string must be entered twice.
date	0…9 Structure identifier: hyphen	Input format: yyyy-mm-dd yyyy: year; optionally 2 or 4 digits mm: month dd: day
device	A…Z 0…9 hyphen	Character string, max. 8 characters in length, corresponding to a device available in the system. In interactive prompting, SDF displays the valid operand values. For notes on possible devices, see the relevant operand description.
fixed	+, - 0…9 period	Input format: [sign][digits].[digits] [sign]: + or - [digits]: 0...9 must contain at least one digit, but may contain up to 10 characters (0...9, period) apart from the sign.

Table 2: Data types (part 1 of 4)

Data type	**Character set**	**Special rules**
filename	A…Z 0…9 $, #, @ hyphen period	Input format: $\text{[:cat:][\$user.]}\left\{\begin{array}{l}\text{file}\\ \text{file(no)}\\ \text{group}\\ \\ \text{group}\left\{\begin{array}{l}\text{(*abs)}\\ \text{(+rel)}\\ \text{(-rel)}\end{array}\right\}\end{array}\right\}$:cat: optional entry of the catalog identifier; character set limited to A...Z and 0...9; maximum of 4 characters; must be enclosed in colons; default value is the catalog identifier assigned to the user ID, as specified in the user catalog. $user. optional entry of the user ID; character set is A…Z, 0…9, $, #, @; maximum of 8 characters; first character cannot be a digit; $ and period are mandatory; default value is the user' s own ID. $. (special case) system default ID file file or job variable name; may be split into a number of partial names using a period as a delimiter: $\text{name}_1\text{[.name}_2\text{[...]]}$ name_i does not contain a period and must not begin or end with a hyphen; file can have a max. length of 41 characters; it must not begin with a $ and must include at least one character from the range A...Z.

Table 2: Data types (part 2 of 4)

Data type	**Character set**	**Special rules**
filename (contd.)		#file (special case) @file (special case) # or @ used as the first character indicates temporary files or job variables, depending on system generation. file(no) tape file name no: version number; character set is A...Z, 0...9, $, #, @. Parentheses must be specified. group name of a file generation group (character set: as for “file”) $\text{group}\left\{\begin{array}{l}\text{(*abs)}\\\text{(+rel)}\\\text{(-rel)}\end{array}\right\}$ (*abs) absolute generation number (1-9999); * and parentheses must be specified. (+rel) (-rel) relative generation number (0-99); sign and parentheses must be specified.
integer	0…9, +, -	+ or -, if specified, must be the first character.
name	A…Z 0…9 $, #, @	Must not begin with 0...9.

Table 2: Data types (part 3 of 4)

Data type	**Character set**	**Special rules**
product-version	A…Z 0…9 period single quote	Input format: [[C]'][V][m]m.naso['] correction status release status where m, n, s and o each is a digit and a is a letter. The data type suffixes determine whether the release and/or correction status are mandatory or optional specifications (see without-corr, without-man, mandatory-man, and mandatory-corr suffixes in table 3). product-version may be enclosed within single quotes (possibly with a preceding C). The specification of the version may begin with the letter V.
structured-name	A…Z 0…9 $, #, @ hyphen	Alphanumeric string which may comprise a number of substrings separated by a hyphen. First character: A...Z or $, #, @
text	freely selectable	For the input format, see the relevant operand descriptions.
time	0…9 structure identifier: colon	Time-of-day entry: Input format: $\left\{\begin{array}{l}\text{hh:mm:ss}\\ \text{hh:mm}\\ \text{hh}\end{array}\right\}$ hh: hours mm: minutes ss: seconds Leading zeros may be omitted
vsn	a) A…Z 0…9 b) A…Z 0…9 $, #, @	a) Input format: pvsid.sequence-no max. 6 characters pvsid: 2-4 characters; PUB may not be entered sequence-no: 1-3 characters b) Max. 6 characters; PUB may be prefixed, but may not be followed by $, #, @.

Table 2: Data types (part 4 of 4)

Suffixes for data types

Suffix	Meaning
x..y	In the case of data type integer: interval specification x minimum value permitted for “integer”. x is an (optionally signed) integer. y maximum value permitted for “integer”. y is an (optionally signed) integer. In the case of other data types: length specification In the case of the data types catid, date, device, product-version, time, and vsn, the length specification is not displayed. x minimum length for the operand value; x is an integer. y maximum length for the operand value; y is an integer. x=y the length of the operand value must be precisely x.
with	Extends the specification options for a data type.
-low	Uppercase and lowercase letters are differentiated.
without	Restricts the specification options for a data type.
-cat	Specification of a catalog ID is not permitted.
-corr	Input format: [[C]'][V][n]n.na['] Specifications for the data type product-version must not include the correction status.
-gen	Specification of a file generation or file generation group is not permitted.
-man	Input format: [[C]'][V][n]n.n['] Specifications for the data type product-version must not include either release or correction status.
-vers	Specification of the version (see “file(no)”) is not permitted for tape files.

Table 3: Data type suffixes

Continuation lines

Statements can extend over a number of input lines. They are separated in accordance with the conventions of the BS2000 command language. A hyphen (-) is used as the separator. Statement lines can have a maximum length of 32763 characters.

Abbreviation options

Guaranteed abbreviations for all statements, operands and operand values are given in the syntax descriptions of the statements (see page 87ff). They are identified by means of boldface. However, even shorter specifications can be made (abbreviations within a structure as long as there is no possibility of confusion).
In order to avoid possible sources of confusion in the wake of functional extensions in future versions and to ensure readability for other users, abbreviations should not be used in procedures.

Response in the event of format input errors

Format input errors are understood to include both violations of the statement syntax and conflicting operands within a statement.

Conflicts between different statements are treated as logic errors (see page 33).

A format error has the following results:

– interactive mode: PERCON offers the option of requesting help menus referring to the statements and, if statements contain errors, the option of conducting a correction dialog via SDF (see the manual “Introductory Guide to the SDF Dialog Interface” [6]);

– batch mode: a branch is made to the next STEP or END statement.

Comments

Character strings which are enclosed in quotation marks are interpreted as comments and ignored.

4.4 Literals

PERCON can process constant values as comparison criteria or for insertion into output records. These values are known as literals.

Format of a literal for:

SET-GROUP-ATTRIBUTES SET-PAGE-LAYOUT SET-RECORD-MAPPING	**SELECT-INPUT-RECORDS**	**Meaning**
[C] 'c-string'		String of up to 256 uppercase or lowercase letters, digits or special characters. Any apostrophe within this string must be specified twice. Memory requirements: 1 byte per character.
X 'x-string'		String of up to 512 hexadecimal digits ('0',...,'F'). If an uneven number of digits is specified, the string is padded on the left with zeros. Memory requirements: 1 byte per pair of hexadecimal digits.
	P 'digit'	Decimal number of up to 16 digits (with optional sign). PERCON stores this number in packed format.
	Z 'digit'	Decimal number of up to 32 digits (with optional sign). PERCON stores this number in zoned format.
integer		Decimal number in the range: -2^{31} to $2^{31} - 1$

4.5 Keywords

PERCON stores various internal information which the user can access by means of keywords. These keywords can be inserted in output records or in print pages as required, or used as comparison criteria. PERCON provides this information multiply for both input and output files.

The counters supplied by PERCON are either incremented or modified, e.g. the record counter of the input file is incremented after a record is read, while the record counter of the output file is incremented after the record is written. The counters are initialized with 0. This must be taken into account when these keywords are referenced in statements. The following keywords are available:

Meaning	Keyword [1]	

	SET-GROUP-ATTRIBUTES
	SET-PAGE-LAYOUT
	SET-RECORD-MAPPING
		SELECT-INPUT-RECORDS

Record length	RECORD-LENGTH	RECLEN
Block counter	BLOCK-COUNTER	BLKCNT
Byte counter	BYTE-COUNTER	BYTCNT
Record counter	RECORD-COUNTER	RECCNT
Page counter	PAGE-COUNTER	Cannot be specified
Date	DATE	
Time	TIME	

[1] The format and length of a keyword depends on the PERCON statement under which the keyword is used. For further information, refer to the syntax description of the corresponding statement.

Editing of keywords with output length greater than or equal to 1

When editing keywords (with the exception of DATE and TIME) in the statements SET-RECORD-MAPPING and SET-GROUP-ATTRIBUTES, the minimum value for the OUTPUT-LENGTH operand is 1.

However, OUTPUT-LENGTH=1 in conjunction with output format DECIMAL or SIGNED-DECIMAL results in a semantical error (message PER0007); a minimum output length of 2 is required for these formats to accommodate the sign.

PERCON makes sure that no valid positions are truncated when editing a keyword (exception: SET-PAGE-LAYOUT). If the specified output length is insufficient and a keyword value to be edited does not fit in the output field, the conversion step is aborted and message PER0042 is output.

Note about the GROUP-COUNTER keyword

Counters are run while grouping input records. These counters are known as group counters (see SET-GROUP-ATTRIBUTES statement on page 141).

GROUP-COUNTER is available for group levels 1 to 8 of every output medium. Each counter is defined unambiguously by its link name and by a group level.

The counters are preset to 0 and their readings are changed under the following circumstances:

– Counters set to 1
  After the group leader has been written or when GROUP-HEADER=*NONE if the criterion for writing the group leader is given.

– Counters incremented by 1
  After an input record has been read if the criterion of a group break is not met.

4.6 Formats

A field is a section of the input/output record characterized by:
– its start position in the record
– its length
– its format.

The format specifies how the record in the input record is to be interpreted or how it is to be modified for the output record when it is transferred. Keywords and literals have an implicitly fixed format. PERCON can process the following formats:

CHARACTER

The field designated by “character” either already contains characters or is to be provided with characters. The term “characters” is understood to include letters, numbers and special characters.

Example

Contents of input field **Input format CHARACTER**	**Contents of output field** **Output format CHARACTER**
X'C1C2F1F2F3F4' (length: 6 bytes)	C’AB1234’ (length: 6 bytes)

HEXADECIMAL

Output format only. The input field is edited in hexadecimal format and output to the designated output field, i.e. each character in the input field is converted to the corresponding hexadecimal code and occupies 2 bytes in the output field.

Example

Contents of input field **Input format CHARACTER**	**Contents of output field** **Output format CHARACTER**
X'C1C2F1F2F3F4' (length: 6 bytes)	C'C1C2F1F2F3F4' (length: 12 bytes)

BINARY

Output format only. The input field is edited in binary format and output to the designated output field, i.e. each character of the input field is converted to binary code and occupies 8 bytes in the output field.

Example

Contents of input field **Input format CHARACTER**	**Contents of output field** **Output format BINARY**
X'C1C2' (length: 2 bytes)	C'1100000111000010' (length: 16 bytes)

ZONED-DECIMAL

The designated field contains an unpacked decimal number in EBCDI code or is to accept such a number. The sign accompanying the number is ignored.

Example

Contents of input field **Input format ZONED-DECIMAL**	**Contents of output field** **Output format ZONED-DECIMAL**
X'F1F2F3F4' (length: 4 bytes)	C'1234' (length: 4 bytes)

Contents of input field **Input format PACKED-DECIMAL**	**Contents of output field** **Output format ZONED-DECIMAL**
X'123F' (length: 2 bytes)	C'123' (length: 3 bytes)

SIGNED-DECIMAL

Output format only. The output field is to accept an unpacked decimal number in which the sign +, or –) is always contained in the first byte of the field.

This must be taken into account for the length entry of the field.

Example

Contents of input field **Input format ZONED-DECIMAL**	**Contents of output field** **Output format SIGNED-DECIMAL**
X'F1F2F3F4' (length: 4 bytes)	C'+1234' (length: 5 bytes)

Contents of input field **Input format PACKED-DECIMAL**	**Contents of output field** **Output format SIGNED-DECIMAL**
X'123F' (length: 2 bytes)	C'+123' (length: 4 bytes)

DECIMAL

Output format only. The output field is to contain an unpacked decimal number whose sign is printed only if the number has a negative value. If it has a positive value, a space character is printed instead of the sign. This must be taken into account for the length entry.

Example

Contents of input field **Input format ZONED-DECIMAL**	**Contents of output field** **Output format DECIMAL**
X'F1F2F3F4' (length: 4 bytes)	C'␣1234' (length: 5 bytes)

Contents of input field **Input format PACKED-DECIMAL**	**Contents of output field** **Output format DECIMAL**
X'123F' (length: 2 bytes)	C'␣123' (length: 4 bytes)

PACKED-DECIMAL

The designated field either contains a packed decimal number or is to accept one. The length of a packed number must not exceed 16 bytes.

Example

Contents of input field **Input format ZONED-DECIMAL**	**Contents of output field** **Output format PACKED-DECIMAL**
X'F1F2F3F4' (length: 4 bytes)	X’01234F' (length: 3 bytes)

Contents of input field **Input format PACKED-DECIMAL**	**Contents of output field** **Output format PACKED-DECIMAL**
X'123F' (length: 2 bytes)	C'123F' (length: 2 bytes)

ZONED-DECIMAL-LEFT

Output format; for keywords only. The output field is to contain an unpacked decimal number. The right-hand part of the output field is filled with space characters.

Example

Contents of	**Output format ZONED-DECIMAL-LEFT**
RECORD-COUNTER	C'82579␣␣␣' (length: 8 bytes)

TRANSLATION

This entry is only permissible within the field definition of the OUTPUT-FIELDS operand in the SET-RECORD-MAPPING statement. The individual characters are converted according to the code table.

NO-TRANSLATION

This specification is valid only within the field description of the OUTPUT-FIELDS operand in the SET-RECORD-MAPPING statement. NO-TRANSLATION must be used whenever the input field is not to be converted to the output character set, e.g. with packed numbers.

UNICODE-TRANSLATION

This specification is valid only within the field description of the OUTPUT-FORMAT operand in the SET-RECORD-MAPPING statement. The input file and/or output file must have a Unicode format assigned. This statement enables individual sections of a record to be converted and/or (if the Unicode variant UTF-16 is being used) normalized. If the OUTPUT-FORMAT=*UNICODE-TRANSLATION specification is missing, no conversion or normalization of the section concerned takes place.

When a non-Unicode format is converted to a Unicode format or vice versa, it must be borne in mind that the length of the section to be output can change (see page 63).

Formatted numbers with an editing mask

Output format: for numerical output fields only. In addition to the output formats ZONED-DECIMAL, SIGNED-DECIMAL and ZONED-DECIMAL-LEFT, it is also possible to specify an editing mask for numerical output fields. In such a mask, leading zeros can be suppressed, trailing positions displayed or numeric grouping carried out.

A mask is defined in the SET-GROUP-ATTRIBUTES or SET-RECORD-MAPPING statement using the operand OUTPUT-FORMAT=<c-string>. In addition, the OUTPUT-LENGTH operand must have the value *STD.

The following characters can be used in a mask:

```
Fill characters (fillers)  :  all printable characters

Numeric characters         :  digit selector ('Z')
                              significance starter ('N')

Insertion characters       :  characters not included under 'Z' and 'N'
```

– Mask structure

  The first character of the mask is interpreted as a fill character (filler). The following conditions apply to mask characters to the right of the filler: the number of numeric characters must be odd and no more than one significance starter may be present. If these conditions are not kept to, the mask is rejected as containing a syntax error. Insertion characters may be located between the numeric characters.

– Mask size

  If the mask contains more numeric characters than are present in the number to be edited, the number to be edited is filled to the left with zeros.

  If the number of digits in the number to be edited is greater than the number of numeric characters in the mask, the number to be edited is shortened (starting from the left). If relevant digits (≠ leading zeros) are cut off during this shortening procedure, the conversion step is immediately terminated abnormally (see message PER0042).

In the case of the previous output formats of PERCON, the following restriction still applies: the number of digit positions in the output field must be greater than or equal to that of the input field.

– Mask processing

  The mask is processed from left to right. All characters before the first numeric character (’Z’, ’N’) are replaced by the fill character.

  As processing continues, all characters (the insertion characters, too) continue to be replaced by the fill character until either a digit selector is replaced by a digit ≠ 0 or a significance starter occurs in the mask.

  Afterwards, the numeric characters are replaced by the corresponding digits, characters to be inserted remaining unchanged.

  Characters to be inserted after the last numeric character remain unchanged if the number to be edited is negative, otherwise they are replaced by the fill character.

Example 1

```
FIELD (INPUT-POSITION=...,-
       INPUT-LENGTH=5,-
       INPUT-FORMAT=*PACKED-DECIMAL,-
       OUTPUT-POSITION=...,-
       OUTPUT-LENGTH=*STD,-
       OUTPUT-FORMAT='*ZZNZ.ZZZ-')
```

The following conversion procedure is carried out:

```
   Field to be edited:      0 0 0 0 0 0 0 3 8 C

   Mask specified:          * Z Z N Z . Z Z Z -

   Output field:            * * * * 0 . 0 3 8 *
```

The packed number is positive; therefore, the minus sign after the number is replaced by the fill character. In the field to be edited, the leading byte (which contains zero) is cut off to adapt the number to the mask. If field elements not equal to zero were to be cut off, this would lead to abnormal termination accompanied by the error message PER0042.

Example 2

```
FIELD (INPUT-POSITION=...,-
       INPUT-LENGTH=4,-
       INPUT-FORMAT=*ZONED-DECIMAL,-
       OUTPUT-POSITION=...,-
       OUTPUT-LENGTH=*STD,-
       OUTPUT-FORMAT='xZ ZNZ,ZZZ -')
```

The following conversion procedure is carried out:

```
   Field to be edited:                  F 3 F 9 F 7 D 1

   Mask specified:            x Z   Z N Z , Z Z Z   -

   Output field:              x x x x x 3 , 9 7 1   -
```

Because the zoned number is negative, the space character and minus sign after the number are transferred to the output field.

4.7 Description of the statements

The PERCON statements are described alphabetically under the following headings:

- statement name and function
- description of statement function
- representation of statement format
- description of statement operands

ASSIGN-INPUT-FILE

This statement is only permissible for file copying. It assigns the input file. Furthermore it can also be used to define the response to errors and which user modules are to be used.

ASSIGN**-INPUT-F**ILE

FILE = ***DISK-F**ILE (...) / ***TAPE-F**ILE(...) / ***SYSDTA**(...)

  ***DISK-FILE** (...)

    | **NAME** = ***STD** / <filename 1..54>

    | ,**REM**OVE**-FILE-LINK** = ***STD** / ***NO** / ***Y**ES(...)

    | ***Y**ES(...)

    | | **RELE**ASE**-DEV**ICE = ***YES** / ***NO**

    | ,**OPEN-ERROR** = ***STD** / ***FIN**ISH**-INPUT**(...) / ***CONT**INUE**-NEXT-F**ILE(...) / ***DIALOG** / ***MOD**ULE(...)

    | ***FIN**ISH**-INPUT**(...)

    | | **TERM**INATION = ***ABNORM**AL / ***NORM**AL

    | ***CONT**INUE**-NEXT-F**ILE(...)

    | | **TERM**INATION = ***ABNORM**AL / ***NORM**AL

    | ***MOD**ULE(...)

    | | **NAME** = <name 1..8>

    | ,**PAR**ITY**-ERR**OR = ***TERM**INATE**-ABNORM**AL / ***IGNORE-ERROR** / ***CONT**INUE**-NEXT-BLOCK** / ***DIALOG** / ***MOD**ULE(...)

    | ***MOD**ULE(...)

    | | **NAME** = <name 1..8>

    | ,**LEN**GTH**-ERROR** = ***TERM**INATE**-ABNORM**AL / ***IGNORE-ERROR** / ***CONT**INUE**-NEXT-BLOCK** / ***DIALOG** / ***MOD**ULE(...)

    | ***MOD**ULE(...)

    | | **NAME** = <name 1..8>

    | ,**EMPTY-BLOCK** = ***STD** / ***TAKE** / ***SKIP**

continued ➟

```
*TAPE-FILE(...)
  | NAME = *STD / <filename 1..54>
  | ,REMOVE-FILE-LINK = *STD / *NO / *YES(...)
  |    *YES(...)
  |       | RELEASE-DEVICE = *YES / *NO
  |       | ,UNLOAD-RELEASED-TAPE = *NO / *YES
  | ,END-POSITION = *BEGIN-OF-TAPE / *BEGIN-OF-FILE / *END-OF-FILE / *UNLOAD-TAPE
  | ,CODE-TRANSLATION = *NO / *MODULE(...)
  |    *MODULE(...)
  |       | NAME = <name 1..8>
  | ,LABEL-EXIT = *NO / *MODULE(...) / *SYSOUT(...)
  |    *MODULE(...)
  |       | NAME = <name 1..8>
  |       | ,CONTROLLED-LABEL = *STD / list-poss(2000): *VOL-USER-LABEL /
  |       |          *HDR-USER-LABEL / *EOV-USER-LABEL / *END-USER-LABEL /
  |       |          *CLOSE-POSITION / *LABEL-ERROR
  |    *SYSOUT(...)
  |       | CONTROLLED-LABEL = *STD / list-poss(2000): *VOL-USER-LABEL /
  |       |          *HDR-USER-LABEL / *EOV-USER-LABEL / *END-USER-LABEL
  | ,OPEN-ERROR = *STD / *FINISH-INPUT(...) / *CONTINUE-NEXT-FILE(...) / *DIALOG /
  |                  *MODULE(...)
  |    *FINISH-INPUT(...)
  |       | TERMINATION = *ABNORMAL / *NORMAL
  |    *CONTINUE-NEXT-FILE(...)
  |       | TERMINATION = *ABNORMAL / *NORMAL
  |    *MODULE(...)
  |       | NAME = <name 1..8>
```

continued ➠

```
        ,PARITY-ERROR = *TERMINATE-ABNORMAL / *IGNORE-ERROR / *CONTINUE-NEXT-BLOCK /
                        *DIALOG / *MODULE(...)

          *MODULE(...)

            NAME = <name 1..8>

        ,LENGTH-ERROR = *TERMINATE-ABNORMAL / *IGNORE-ERROR / *CONTINUE-NEXT-BLOCK /
                        *DIALOG / *MODULE(...)

          *MODULE(...)

            NAME = <name 1..8>

        ,EMPTY-BLOCK = *STD / *TAKE / *SKIP

    *SYSDTA(...)

        RECORD-SIZE = 32767 / <integer 4..32767>

        ,INPUT-ERROR = *TERMINATE-ABNORMAL / *IGNORE-ERROR / *CONTINUE-NEXT-RECORD /
                       *DIALOG / *MODULE(...)

          *MODULE(...)

            NAME = <name 1..8>

,LINK-NAME = PCIN / <filename 1..8 without-gen>

,INPUT-EXIT = *NO / *MODULE(...)

    *MODULE(...)

        NAME = <name 1..8>
```

FILE =
FILE is used to select an input medium.
When FILE=*DISK-FILE or FILE=*TAPE-FILE is specified, supplementary specifications can be made with the aid of the ADD-FILE-LINK command. When an input file is assigned via a ADD-FILE-LINK command or via an ASSIGN-INPUT-FILE statement, an entry is created in the task file table (TFT) under the link name.

FILE = *DISK-FILE (...)
The input file is on disk.

NAME = *STD / <filename 1..54>
Name of the input file. This is not required if the file name has already been specified in a ADD-FILE-LINK command to which the ASSIGN-INPUT-FILE statement refers via the LINK-NAME operand. The file name in the ADD-FILE-LINK command has priority over this entry.

NAME = *STD
The default value is the file name which accompanies the link name defined in the ADD-FILE-LINK command. If no ADD-FILE-LINK command has been issued, the name declared by the LINK-NAME operand is used as the file name.

REMOVE-FILE-LINK =
Defines whether or not and how the TFT entry is to be deleted after file processing or if it is to be retained.

REMOVE-FILE-LINK = *STD
PERCON deletes the TFT entry only if PERCON itself has created it via an ASSIGN-INPUT-FILE statement. If the file has been assigned using a ADD-FILE-LINK command, the TFT entry is retained.

REMOVE-FILE-LINK = *NO
The TFT entry is retained, regardless of how it was created.

REMOVE-FILE-LINK = *YES(...)
The TFT entry is deleted, regardless of how it was created.

RELEASE-DEVICE = <u>*YES</u> / *NO
Used for management of the disk device after processing; for private disks only. *YES releases the disk device, with *NO the device remains assigned to this task.

OPEN-ERROR =
Controls PERCON’s response to open errors on the input file.

OPEN-ERROR = <u>*STD</u>

Interactive mode:	*DIALOG
Batch mode:	*FINISH-INPUT(TERMINATION=*ABNORMAL)

OPEN-ERROR = *FINISH-INPUT(...)
The input files which could be opened up until the input file was addressed are processed.

TERMINATION = *ABNORMAL / *NORMAL
The conversion step is terminated either abnormally or normally.

OPEN-ERROR = *CONTINUE-NEXT-FILE(...)
The next input file is processed.

TERMINATION = *ABNORMAL / *NORMAL
The conversion step is terminated either normally or abnormally.

OPEN-ERROR = *DIALOG
Only for input via the terminal. PERCON expects instructions from the terminal regarding further processing.
The following responses are possible.

HA:	The conversion step is terminated abnormally.
HN:	The conversion step is terminated normally.
SA:	The next input file is processed. The conversion step is terminated abnormally.
SN:	The next input file is processed. The conversion step is terminated normally.

If this operand value is specified in a batch job, the specification is ignored, a warning is output and the default value for batch mode goes into effect.

OPEN-ERROR = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for the recovery of open errors” on page 188).

PARITY-ERROR =
Controls PERCON’s response if a block of a SAM file cannot be read due to a parity error or if only part of a block can be read.

PARITY-ERROR = *TERMINATE-ABNORMAL
The input and output files are closed and the conversion step is terminated with an error.

PARITY-ERROR = *IGNORE-ERROR
The error is ignored. PERCON reacts as if the block in question could be read. The block is transferred to the output file with the error.

PARITY-ERROR = *CONTINUE-NEXT-BLOCK
The errored block is skipped.

PARITY-ERROR = *DIALOG
Only for input via the terminal. PERCON expects instructions from the terminal relating to further processing. The following responses are possible:

H:	The conversion step is terminated with an error.
I:	The error is ignored
S:	The block is skipped

If this operand value is specified in a batch job, the specification is ignored, a warning is output and the default value goes into effect.

PARITY-ERROR = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for the recovery of read/length errors” on page 187).

LENGTH-ERROR =
Controls PERCON’s response if a block is to be read whose length deviates from the total number of records contained therein.

LENGTH-ERROR = *TERMINATE-ABNORMAL
The input and output files are closed and the conversion step is terminated with an error.

LENGTH-ERROR = *IGNORE-ERROR
The error is ignored and the block in question is processed despite the error.

LENGTH-ERROR = *CONTINUE-NEXT-BLOCK
The errored block is skipped.

LENGTH-ERROR = *DIALOG
Only for input via the terminal. PERCON expects instructions from the terminal relating to further processing. The following responses are possible:

H:	The conversion step is terminated with an error.
I:	The error is ignored
S:	The block is skipped

If this operand value is specified in a batch job, the specification is ignored, a warning is output and the default value goes into effect.

LENGTH-ERROR = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for the recovery of read/length errors” on page 187).

EMPTY-BLOCK =
Specifies how empty blocks are to be processed (see “PAM input files” on page 47).

EMPTY-BLOCK = <u>*STD</u>
*SKIP when BLOCK-CONTROL-INFO = *PAMKEY,
*TAKE when BLOCK-CONTROL-INFO = *WITHIN-DATA.

EMPTY-BLOCK = *TAKE
Empty blocks are treated like input records.

EMPTY-BLOCK = *SKIP
Empty blocks are skipped.

FILE = *TAPE-FILE(...)
The input file is on tape.

NAME = <u>*STD</u> / <filename 1..54>
Specifies the name of the input file.
This is not required if the file name has already been specified in a ADD-FILE-LINK command to which the ASSIGN-INPUT-FILE statement refers via the LINK-NAME operand. The file name in the ADD-FILE-LINK command has priority over the one specified here.

NAME = <u>*STD</u>
The default value is the name associated with the link name specified in the ADD-FILE-LINK command. If no ADD-FILE-LINK command has been issued, the name declared in the LINK-NAME operand is used as the file name.

REMOVE-FILE-LINK =
Defines whether or not and how the TFT entry is to be deleted after processing or if it is to be retained.

RELEASE-DEVICE = <u>*STD</u>
PERCON deletes the TFT entry only if the entry has been created by PERCON itself by means of an ASSIGN-INPUT-FILE statement. If the file has been assigned using a ADD-FILE-LINK command, the TFT entry is retained.

REMOVE-FILE-LINK = *NO
The TFT entry is retained, regardless of how it was created.

REMOVE-FILE-LINK = *YES(...)
The TFT entry is deleted, regardless of how it was created.

> **RELEASE-DEVICE = <u>*YES</u> / *NO**
> Is used to manage the tape device after processing. The device is released when *YES is entered and remains assigned to this task when *NO is entered.
>
> **UNLOAD-RELEASED-TAPE = *NO / <u>*YES</u>**
> Is used for management of the tape after processing. The tape remains loaded when *NO is entered, and is unloaded when *YES is entered.

END-POSITION =
Specifies how the tape is to be positioned after the file has been processed.

END-POSITION = <u>*BEGIN-OF-TAPE</u>
The tape is wound back to the BOT marker.

END-POSITION = *BEGIN-OF-FILE
The tape is positioned to the beginning of the file that has just been processed.

END-POSITION = *END-OF-FILE
The position of the tape is not changed after processing.

END-POSITION = *UNLOAD-TAPE
The tape is rewound and unloaded.

CODE-TRANSLATION =
Specifies whether code conversion is to be performed for the input records.

CODE-TRANSLATION = <u>*NO</u>
No code conversion is to take place.

CODE-TRANSLATION = *MODULE(...)
The code conversion table resides in the module library assigned by means of SET-TASKLIB. Its contents are the 256 hexadecimal encryptions (X’00’ to X’FF’) of the characters to be converted to the input characters.

Conversion is based on the hexadecimal value of the input character. This value is added to the start address of the conversion table; the character at the resulting position of the conversion table replaces the input character.

NAME = <name 1..8>
Name of the user module or entry point in the user module containing the code conversion table.

LABEL-EXIT =
Defines user label processing.

LABEL-EXIT = *NO
User modules are not processed separately.

LABEL-EXIT = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for label processing” on page 180).

CONTROLLED-LABEL =
Control branches to the user module when user labels are specified.

CONTROLLED-LABEL = *STD
Control branches to the user module when the following four user labels are processed. VOL-USER-LABEL, HDR-USER-LABEL, EOV-USER-LABEL, END-USER-LABEL

CONTROLLED-LABEL = *VOL-USER-LABEL
Control branches to the user module when the user volume labels are processed.

CONTROLLED-LABEL = *HDR-USER-LABEL
Control branches to the user module when the user header labels are processed.

CONTROLLED LABEL = *EOV-USER-LABEL
Control branches to the user module when the user end-of-volume labels are processed for a tape swap.

CONTROLLED-LABEL = *END-USER-LABEL
Control branches to the user module when the user end labels are processed.

CONTROLLED-LABEL = *CLOSE-POSITION
Control branches to the user module to position the tape during CLOSE processing.

CONTROLLED-LABEL = *LABEL-ERROR
Control branches to the user module in the event of end-of-tape errors.

LABEL-EXIT = *SYSOUT(...)
PERCON logs the user labels to SYSOUT. This specification is only permitted for files with standard labels.

CONTROLLED-LABEL =
The specified user label groups are output to SYSOUT.

CONTROLLED-LABEL = *STD
The following user label groups are output to SYSOUT: VOL-USER-LABEL, HDR-USER-LABEL, EOV-USER-LABEL, END-USER-LABEL

CONTROLLED-LABEL = *VOL-USER-LABEL
User volume labels are output to SYSOUT.

CONTROLLED-LABEL = *HDR-USER-LABEL
User header labels are output to SYSOUT.

CONTROLLED-LABEL = *EOV-USER-LABEL
User end-of-volume labels are output to SYSOUT.

CONTROLLED-LABEL = *END-USER-LABEL
User end labels are output to SYSOUT.

OPEN-ERROR =
Controls PERCON's response to open errors in the input file.

OPEN-ERROR = *STD

Interactive mode:	*DIALOG
Batch mode:	*FINISH-INPUT(TERMINATION=*ABNORMAL)

OPEN-ERROR = *FINISH-INPUT(...)
The input files which could be opened up until the specified input file are processed.

TERMINATION = *ABNORMAL / *NORMAL
The conversion step is terminated either abnormally or normally.

OPEN-ERROR = *CONTINUE-NEXT-FILE(...)
The next input file is processed.

TERMINATION = *ABNORMAL / *NORMAL
The conversion step is terminated either abnormally or normally.

OPEN-ERROR = *DIALOG
Only for input from the terminal. PERCON expects instructions from the terminal relating to further processing. The following responses are possible.

HA:	The conversion step is terminated abnormally.
HN:	The conversion step is terminated normally.
SA:	The next input file is processed. The conversion step is terminated abnormally.
SN:	The next input file is processed. The conversion step is terminated normally.

If this operand value is specified in a batch job, the specification is ignored, a warning is issued and the default value for batch mode becomes effective.

OPEN-ERROR = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for the recovery of open errors” on page 188).

PARITY-ERROR =
Controls PERCON’s response if a block of a SAM file or BTAM file cannot be read due to a parity error or if only part of a block can be read.

PARITY-ERROR = *TERMINATE-ABNORMAL
The input and output files are closed and the conversion step is terminated with an error.

PARITY-ERROR = *IGNORE-ERROR
The error is ignored. PERCON reacts as if the block in question could be read. The block, including the error, is transferred to the output file.

PARITY-ERROR = *CONTINUE-NEXT-BLOCK
The errored block is skipped.

PARITY-ERROR = *DIALOG
Only for input via the terminal. PERCON expects instructions from the terminal relating to further processing.
The following responses are possible:

H:	The conversion step is terminated with an error.
I:	The error is ignored
S:	The block is skipped

If this operand value is specified in a batch job, the specification is ignored, a warning is issued and the default value goes into effect.

PARITY-ERROR = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for the recovery of read/length errors” on page 187).

LENGTH-ERROR =
Controls PERCON’s response if a block is to be read whose length deviates from the value defined in the BUFFER-LENGTH operand of the ADD-FILE-LINK command.

LENGTH-ERROR = *TERMINATE-ABNORMAL
The input and output files are closed and the conversion step is terminated with an error.

LENGTH-ERROR = *IGNORE-ERROR
The error is ignored and the block in question is processed despite the error.

LENGTH-ERROR = *CONTINUE-NEXT-BLOCK
The errored block is skipped.

LENGTH-ERROR = *DIALOG
Only for input from the terminal. PERCON expects instructions from the terminal relating to further processing.
The following responses are possible:

H:	The conversion step is terminated with an error.
I:	The error is ignored
S:	The block is skipped

If this operand value is specified in a batch job, the specification is ignored, a warning is issued and the default value goes into effect.

LENGTH-ERROR = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for the recovery of read/length errors” on page 187).

EMPTY-BLOCK =
Specifies how empty blocks are to be processed (see also “PAM input files” on page 47).

EMPTY-BLOCK = <u>*STD</u>
*SKIP when BLOCK-CONTROL-INFO = *PAMKEY,
*TAKE when BLOCK-CONTROL-INFO = *WITHIN-DATA.

EMPTY-BLOCK = *TAKE
Empty blocks are treated as input records.

EMPTY-BLOCK = *SKIP
Empty blocks are skipped.

FILE = *SYSDTA(...)
Input is from the SYSDTA system file. Input begins with the record read from SYSDTA after either the START-CONVERSION or END statement and terminates when the end-of-file condition is encountered. This happens when:

– a record begins with “/EOF”
– the end of a cataloged file is reached
– a BS2000 command is recognized in a procedure or a batch job.

Notes

- When the end-of-file condition is encountered when reading a file, SYSDTA is assigned to SYSCMD again.
- When the data is entered via terminal, the EOF condition is signaled by the following sequence:
  - interrupt key K2
  - /EOF
  - /RESUME-PROGRAM

RECORD-SIZE = 32767 / <integer 4..32767>
Specifies the maximum length of the records to be processed.

INPUT-ERROR =
Controls PERCON’s response when an input error occurs.

INPUT-ERROR = *TERMINATE-ABNORMAL
The input and output files are closed and the conversion step is terminated with an error.

INPUT-ERROR = *IGNORE-ERROR
The error is ignored. PERCON responds as if the record in question could be read. The record is transferred with the error.

INPUT-ERROR = *CONTINUE-NEXT-BLOCK
The errored record is skipped.

INPUT-ERROR = *DIALOG
Only for input via the terminal. PERCON expects instructions from the terminal relating to further processing.
The following responses are possible:

H:	The conversion step is terminated with an error.
I:	The error is ignored
S:	The record is skipped

If this operand value is specified in a batch job, the specification is ignored, a warning is issued and the default value goes into effect.

INPUT-ERROR = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for the recovery of read/length errors” on page 187).

LINK-NAME = PCIN / <filename 1..8 without-gen>
PERCON link name used for reference in keywords and the ADD-FILE-LINK command.

INPUT-EXIT =
Defines the name of the user module or entry point in the user module which processes the input records immediately after reading.

INPUT-EXIT = *NO
No processing by the user module

INPUT-EXIT = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for input” on page 182).

ASSIGN-INPUT-TAPE

This statement is permissible for tape editing or duplication only. It assigns the input tape or MF/MV set. Furthermore the response to errors and the user modules to be connected can be defined for error situations.

```
ASSIGN-INPUT-TAPE

 VOLUME = list-poss(100): <alphanum-name 1..6>
,LINK-NAME = PCIN / <filename 1..8 without-gen>
,END-POSITION = *BEGIN-OF-TAPE / *END-OF-TAPE / *UNLOAD-TAPE
,CODE = *EBCDIC / *ISO7 / *OWN(...)
   *OWN(...)
     |  NAME = <name 1..8>
,DEVICE-TYPE = T1600 / <structured-name 1..8>
,BUFFER-LENGTH = *STD / <integer 18..32768> / *MULTIPLE-OF-2K(..)
   *MULTIPLE-OF-2K(..)
     |  MULTIPLE = <1..128>
,PARITY-ERROR = *TERMINATE-ABNORMAL / *IGNORE-ERROR / *CONTINUE-NEXT-BLOCK / *DIALOG/
                *MODULE(...)
   *MODULE(...)
     |  NAME = <name 1..8>
,LENGTH-ERROR = *TERMINATE-ABNORMAL / *IGNORE-ERROR / *CONTINUE-NEXT-BLOCK /
                *DIALOG / *MODULE(...)
   *MODULE(...)
     |  NAME = <name 1..8>
```

VOLUME = list-poss(100): <alphanum-name 1..6>
Volume serial number of the input tape.
In the case of tape duplication, this operand can be used to specify the VSN of an input MF/MV set to be processed.

LINK-NAME = PCIN / <filename 1..8 without-gen>
Internal PERCON link name used for reference in keywords.

END-POSITION =
Specifies how the tape is to be positioned after processing.

END-POSITION = <u>*BEGIN-OF-TAPE</u>
The tape is wound back to BOT.

END-POSITION = *END-OF-TAPE
The position of the tape is not changed after processing.

END-POSITION = *UNLOAD-TAPE
The tape is wound back and unloaded.

CODE =
Specifies the code in which the tape is written. During code conversion it should be noted that all the data (including labels) on the tape is newly encoded.

CODE = <u>*EBCDIC</u>
The tape is written in EBCDI code and processed without code conversion.

CODE = *ISO7
The tape is written in ISO7 code. The input data is converted to EBCDI code.

CODE = *OWN(...)
The contents of the tape are to be converted using the user’s own code conversion table located in the module library assigned via SET-TASKLIB. It contains the 256 hexadecimal encryptions (X’00’ to X’FF’) of the characters to which the input characters are to be converted.
Conversion is based on the hexadecimal value of the input character. This value is added to the start address of the conversion table; the character at the resulting position of the conversion table replaces the input character.

NAME = <name 1..8>
Name of the user module or entry point in the user module containing the code conversion table.

DEVICE-TYPE =
Specifies the type of tape device.

DEVICE-TYPE = <u>T1600</u>
9-track tape device with 1600 bpi.

DEVICE-TYPE = <structured-name 1..8>
Explicit specification of the type of tape device.

BUFFER-LENGTH =
Maximum block length of the input tape or MF/MV set. If longer blocks occur, the action specified in the LENGTH-ERROR operand is executed.

BUFFER-LENGTH = *STD / <integer 18..32768>
For the non-privileged user.
Default value *STD: 32768 bytes

BUFFER-LENGTH = *STD / <integer 18..32768>/ *MULTIPLE-OF-2K(...)
Only for system administrators (TSOS or HSMSADM privilege required) and applications which run in the privileged processor state TPR.
Default value *STD: *MULTIPLE-OF-2K(128)

> **MULTIPLE = <1..128>**
> In PERCON V2.8A and higher, tape block sizes of more than 32 KB can be processed. A maximum tape block size of 256 KB is supported.

Note

> PERCON recognizes that large tape blocks are to be used from the setting for the tape block size in the file control block (FCB), the task file table (TFT) or the HDR1 label.
> If the der *MULTIPLE-OF-2K(...) operand is specified by non-privileged users, it is rejected with the OPEN return code DCA.

PARITY-ERROR =
Controls PERCON’s response if a block cannot be read due to a parity error, or if only a part of a block can be read.

PARITY-ERROR = *TERMINATE-ABNORMAL
The conversion step is terminated with an error.

PARITY-ERROR = *IGNORE-ERROR
The error is ignored. PERCON responds as if the block in question could be read. The block is transferred to the output file/output tape with the error.

PARITY-ERROR = *CONTINUE-NEXT-BLOCK
The errored block is transferred.

PARITY-ERROR = *DIALOG
Only for input via the terminal.
PERCON expects instructions from the terminal relating to further processing.
The following responses are possible:

H:	The conversion step is terminated with an error.
I:	The error is ignored.
S:	The block is skipped.

If this operand value is specified in a batch job, the entry is ignored, a warning is issued, and the default value goes into effect.

PARITY-ERROR = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for the recovery of read/length errors” on page 187).

LENGTH-ERROR =
Controls the response of PERCON in the event of a length error while reading a block. Length errors can occur with certain processing modalities (e.g. RECF=F and BUFFER-LENGTH small enough) if the length of the block read exceeds that specified in the BUFFER-LENGTH operand.

LENGTH-ERROR = <u>*TERMINATE-ABNORMAL</u>
The conversion step is terminated with an error.

LENGTH-ERROR = *IGNORE-ERROR
The error is ignored and the block in question is transferred.

LENGTH-ERROR = *CONTINUE-NEXT-BLOCK
The errored block is skipped.

LENGTH-ERROR = *DIALOG
Only for input via the terminal.
PERCON expects instructions from the terminal relating to further processing.
The following responses are possible:

H:	The conversion step is terminated with an error.
I:	The error is ignored.
S:	The block is skipped.

If this operand value is specified in a batch job, the specification is ignored, a warning is issued and the default value goes into effect.

LENGTH-ERROR = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for the recovery of read/length errors” on page 187).

ASSIGN-OUTPUT-FILE

This statement is only permissible for file copying or tape editing. It assigns an output file. Furthermore, the response to errors and the subsequent user modules to be connected can be defined for error situations.

```
ASSIGN-OUTPUT-FILE

FILE = *DISK-FILE (...) / *TAPE-FILE(...) / *SYSLST(...) / *SYSOUT(...)

  *DISK-FILE (...)

    |  NAME = *STD / <filename 1..54>
    |
    | ,REMOVE-FILE-LINK = *STD / *NO / *YES(...)
    |
    |    *YES(...)
    |
    |    |  RELEASE-DEVICE = *YES / *NO
    |
    | ,FILE-ATTRIBUTES = *STD / *INPUT-FILE
    |
    | ,OVERWRITE = *YES / *NO
    |
    | ,ALTERNATE-INDEX = *NONE / *FROM-INPUT-FILE(...)
    |
    |    *FROM-INPUT-FILE(...)
    |
    |    |  LINK-NAME = PCIN / <filename 1..8 without-gen>
    | ,UNICODE-NORMALIZE = *NO / *COMPOSED
    |
    | ,UNICODE-FILLER = *BLANK / *NIL

  *TAPE-FILE(...)

    |  NAME = *STD / <filename 1..54>
    |
    | ,REMOVE-FILE-LINK = *STD / *NO / *YES(...)
    |
    |    *YES(...)
    |
    |    |  RELEASE-DEVICE = *YES / *NO
    |    |
    |    | ,UNLOAD-RELEASED-TAPE = *NO / *YES
    |
    | ,FILE-ATTRIBUTES = *STD / *INPUT-FILE
    |
    | ,OVERWRITE = *YES / *NO
    |
    | ,END-POSITION = *BEGIN-OF-TAPE / *BEGIN-OF-FILE / *END-OF-FILE / *UNLOAD-TAPE
    | ,CODE-TRANSLATION = *NO / *MODULE(...)
    |
    |    *MODULE(...)
    |
    |    |  NAME = <name 1..8>
```

continued ➠

```
      | ,LABEL-EXIT = *NO / *MODULE(...)
      |    *MODULE(...)
      |       | NAME = <name 1..8>
      |       | ,CONTROLLED-LABEL = *STD / list-poss(2000): *VOL-USER-LABEL /
      |       |         *HDR-USER-LABEL / *EOV-USER-LABEL / *END-USER-LABEL /
      |       |         *CLOSE-POSITION / *LABEL-ERROR
      | ,UNICODE-NORMALIZE = *NO / *COMPOSED
      | ,UNICODE-FILLER = *BLANK / *NIL
   *SYSLST(...)
      | RECORD-FORMAT = *UNDEFINED / *VARIABLE(...) / *FIXED(...)
      |    *VARIABLE(...)
      |       | RECORD-SIZE = 32768 / <integer 4..32768>
      |    *FIXED(...)
      |       | RECORD-SIZE = 32768 / <integer 1..32768>
   *SYSOUT(...)
      | RECORD-FORMAT = *UNDEFINED / *VARIABLE(...) / *FIXED(...)
      |    *VARIABLE(...)
      |       | RECORD-SIZE = 32768 / <integer 4..32768>
      |    *FIXED(...)
      |       | RECORD-SIZE = 32768 / <integer 1..32768>
,LINK-NAME = PCOUT / <filename 1..8 without-gen>
,OUTPUT-EXIT = *NO / *MODULE(...)
   *MODULE(...)
      | NAME = <name 1..8>
```

FILE =
The FILE command is used to select an output medium.
When FILE=*DISK-FILE or *TAPE-FILE is specified, supplementary specifications can be made via the ADD-FILE-LINK command. When assigning an output file via a ADD-FILE-LINK command or an ASSIGN-OUTPUT-FILE statement, an entry is made in the task file table (TFT) under the link name.

FILE = *DISK-FILE (...)
The output file is on disk.

NAME = *STD / <filename 1..54>
Specifies the name of the output file.This is not required if the file name has already been specified in a ADD-FILE-LINK command to which the ASSIGN-OUTPUT-FILE statement refers via the LINK-NAME operand. The file name in the ADD-FILE-LINK command has priority over the one specified here.

NAME = *STD
The default value is the file name associated with the link name defined in the ADD-FILE-LINK command. If no ADD-FILE-LINK command has been issued, the name declared by means of the LINK-NAME operand is used as the file name.

REMOVE-FILE-LINK =
Defines whether or not and how the TFT entry is to be deleted after file processing or retained.

REMOVE-FILE-LINK = *STD
PERCON deletes the TFT entry only if PERCON itself has created it using an ASSIGN-OUTPUT-FILE statement. If the file has been assigned using a ADD-FILE-LINK command, the TFT entry is retained.

REMOVE-FILE-LINK = *NO
The TFT entry is retained, regardless of how it was created.

REMOVE-FILE-LINK = *YES(...)
The TFT entry is deleted, regardless of how it was created.

> **RELEASE-DEVICE = *YES / *NO**
> Used for management of the disk device after processing.
> For private disks only.
> *YES releases the disk device, *NO specifies that the disk remains assigned to this task.

FILE-ATTRIBUTES =
Defines file attributes for an output file to be newly created which were not specified by means of a ADD-FILE-LINK command.

FILE-ATTRIBUTES = *STD
PERCON assumes the following default values for the file attributes:

```
ACCESS-METHOD = *ISAM
BLOCK-CONTROL-INFO = default value
BUFFER-LENGTH = *STD(SIZE = n)
      where n = 1 for 2K-formatted pubset
      where n = 2 for 4K-formatted pubset
RECORD-FORMAT = *VARIABLE
KEY-POSITION = 5
KEY-LENGTH = 8
```

FILE-ATTRIBUTES = *INPUT-FILE
The file attributes `ACCESS-METHOD, BLOCK-CONTROL-INFO`, `BUFFER-LENGTH, RECORD-FORMAT, RECORD-SIZE, KEY-LENGTH` and `KEY-POSITION` are taken from the values set for processing of the first input file.

If the first input file is located on SYSDTA, the default values described under *STD are used with one exception: SIZE=2 is always set for BUFFER-LENGTH.

If the first input file has a variable record format and if editing is requested for the output file, make sure that the unedited output record is provided with an additional record length field in the data section.

If the input file is an ISAM file and specifies BLOCK-CONTROL-INFO=*WITHIN-DATA-2K-BLOCK, it is entered with BLOCK-CONTROL-INFO=*WITHIN-DATA-BLOCK. This means that the file can be set up with WITHIN-DATA-2K-BLOCK or WITHIN-DATA-4K-BLOCK, depending on the pubset on which it is created.

OVERWRITE =
Defines for cases where the ISAM output file already exists how PERCON is to handle input records with keys already present in the output file.

OVERWRITE = *<u>YES</u>
Records with identical key can be processed.
Depending on the specification for DUPLICATE-KEY, the following applies:
If DUPLICATE-KEY=*NO is specified, the record with identical key is overwritten. If more than one record in the output file have the same key, the first of these is overwritten.
If DUPLICATE-KEY=*YES, the record is inserted behind the records with identical key.

OVERWRITE = *NO
Records with identical key cannot be processed.
Only records with new keys can be inserted. This applies irrespective of the specification for DUPLICATE-KEY. The conversion step is aborted and message PER0098 is output if PERCON detects an input record with the same key as a record in the output file. This also applies if the input file contains more than one record with identical key (the first record is inserted, the second causes PERCON to abort).

ALTERNATE-INDEX =
Defines whether the secondary keys of an input file are to be duplicated for the output file or not.

ALTERNATE-INDEX = <u>*NONE</u>
No secondary keys are to be duplicated for the output file.

ALTERNATE-INDEX = *FROM-INPUT-FILE(...)
Secondary keys are to be duplicated for the output file.

LINK-NAME = <u>PCIN</u> / <filename 1..8 without-gen>
PERCON link name of the input file whose secondary keys are to be duplicated.

UNICODE-NORMALIZE =
Defines whether or not the characters are to be normalized. They can be normalized only if the input file and/or output file is assigned the Unicode variant UTF-16. If neither the input file nor the output file is assigned the Unicode variant UTF-16, the specification is ignored.

UNICODE-NORMALIZE = *NO
No normalization takes place.

UNICODE-NORMALIZE = *COMPOSED
Normalization takes place in the composite character representation of the COMPOSED format.

UNICODE-FILLER =
Defines the fill character for Unicode.

UNICODE-FILLER = *BLANK
The blank in the code of the output record is used as the fill character.

UNICODE-FILLER = *NIL
The character X’00’ or X’0000’ is used as the fill character.

FILE = *TAPE-FILE(...)
Output is to a tape file.

NAME = *STD / <filename 1..54>
Specifies the name of the output file.
This is not required if the file name has already been specified in a ADD-FILE-LINK command to which the ASSIGN-OUTPUT-FILE statement refers via the LINK-NAME operand. The file name in the ADD-FILE-LINK command has priority over the one specified here.

NAME = *STD
The default value is the file name associated with the link name specified in the ADD-FILE-LINK command. If no ADD-FILE-LINK command has been issued, the name specified in the LINK-NAME operand is used as the file name.

REMOVE-FILE-LINK =
Defines whether or not and how the TFT entry is to be deleted after file processing or if it is to be retained.

REMOVE-FILE-LINK = *STD
PERCON deletes the TFT entry only if PERCON itself has created it via the ASSIGN-OUTPUT-FILE statement. If the file was assigned using a ADD-FILE-LINK command, the TFT entry is retained.

REMOVE-FILE-LINK = *NO
The TFT entry is retained, regardless of how it was created.

REMOVE-FILE-LINK = *YES(...)
The TFT entry is deleted, regardless of how it was created.

RELEASE-DEVICE = <u>*YES</u> / *NO
Used to manage the reservation of tape device types after processing. If set to *YES the device is released. If set to *NO, the tape remains assigned to this task.

UNLOAD-RELEASED-TAPE = *NO / *<u>YES</u>
Used for management of the tape after processing. The tape remains loaded if *NO is entered. The tape is unloaded if *YES is entered.

FILE-ATTRIBUTES =
Defines file attributes for an output file to be newly created which were not specified via a ADD-FILE-LINK command.

FILE-ATTRIBUTES = <u>*STD</u>
PERCON uses the following default values for the file attributes:

```
ACCESS-METHOD = *SAM
BLOCK-CONTROL-INFO = *PAMKEY
BUFFER-LENGTH = *STD(SIZE = 1)
RECORD-FORMAT = *VARIABLE
```

FILE-ATTRIBUTES = *INPUT-FILE
The file attributes `ACCESS-METHOD`, `BLOCK-CONTROL-INFO`, `BUFFER-LENGTH`, `RECORD-FORMAT` and `RECORD-SIZE` are taken from the values set for processing of the first input file.

If the first input file is located on SYSDTA, the default values described under *STD are used.

If the first input file has a variable record format and if editing is requested for the output file, make sure that the unedited output record is provided with an additional record length field in the data section.

OVERWRITE =
Defines whether an existing output file can be overwritten or merely extended.

OVERWRITE = <u>*YES</u>
The output file can be overwritten sequentially.

OVERWRITE = *NO
The output file cannot be overwritten, but only extended.

END-POSITION =
Specifies how the tape is to be positioned after the file has been processed.

END-POSITION = <u>*BEGIN-OF-TAPE</u>
The tape is rewound to BOT.

END-POSITION = *BEGIN-OF-FILE
Specifies how the tape is to be positioned after the file has been processed.

END-POSITION = *END-OF-FILE
The position of the tape is not changed after processing.

END-POSITION = *UNLOAD-TAPE
The tape is rewound and unloaded.

CODE-TRANSLATION =
Specifies whether code conversion is to be performed for the output records.

CODE-TRANSLATION = <u>*NO</u>
No code conversion takes place.

CODE-TRANSLATION = *MODULE(...)
The code conversion table is in the module library assigned via SET-TASKLIB. Its contents are the 256 hexadecimal encryptions (X’00’ to X’FF’) of the characters to which the output characters are to be converted.
Conversion is based on the hexadecimal value of the output character. This value is added to the start address of the conversion table; the character at the resulting position of the conversion table replaces the output character.

> **NAME = <name 1..8>**
> Name of the user module or entry point in the user module containing the code conversion table.

LABEL-EXIT =
Defines user label processing.

LABEL-EXIT = <u>*NO</u>
User labels are not processed separately.

LABEL-EXIT = *MODULE(...)

> **NAME = <name 1..8>**
> Name of the user module or entry point in the user module which handles further processing (see “Interface for label processing” on page 180).
>
> **CONTROLLED-LABEL =**
> Specifies that control is to branch to the user module at the specified points.
>
> **CONTROLLED-LABEL = <u>*STD</u>**
> Control branches to the user module when the following four user labels are processed:
> VOL-USER-LABEL, HDR-USER-LABEL, EOV-USER-LABEL, END-USER-LABEL
>
> **CONTROLLED-LABEL = *VOL-USER-LABEL**
> Control branches to the user module when the user volume labels are processed.

CONTROLLED-LABEL = *HDR-USER-LABEL
Control branches to the user module when the user header labels are processed.

CONTROLLED-LABEL = *EOV-USER-LABEL
Control branches to the user module when the user end-of-volume labels are processed for a tape swap.

CONTROLLED-LABEL = *END-USER-LABEL
Control branches to the user module when the user end labels are processed.

CONTROLLED-LABEL = *CLOSE-POSITION
Control branches to the user module in order to position the tape for CLOSE processing.

CONTROLLED-LABEL = *LABEL-ERROR
Control branches to the user module in the event of errored end-of-tape labels.

UNICODE-NORMALIZE =
Defines whether or not the characters are to be normalized. They can be normalized only if the input file and/or output file is assigned the Unicode variant UTF-16. If neither the input file nor the output file is assigned the Unicode variant UTF-16, the specification is ignored.

UNICODE-NORMALIZE = <u>*NO</u>
No normalization takes place.

UNICODE-NORMALIZE = *COMPOSED
Normalization takes place in the composite character representation of the COMPOSED format.

UNICODE-FILLER =
Defines the fill character for Unicode.

UNICODE-FILLER = <u>*BLANK</u>
The blank in the code of the output record is used as the fill character.

UNICODE-FILLER = *NIL
The character X’00’ or X’0000’ is used as the fill character.

FILE = *SYSLST(...)
Output is to system file SYSLST in the prescribed standard format; this format can be modified by means of the SET-PAGE-LAYOUT statement.

RECORD-FORMAT =
Defines the record format of the unedited output record.

RECORD-FORMAT = <u>*UNDEFINED</u>
Defines records of undefined length.

RECORD-FORMAT = *VARIABLE(...)
Defines records of variable length.

RECORD-SIZE = <u>32768</u> / <integer 4..32768>
Maximum length of the unedited output records.

RECORD-FORMAT = *FIXED(...)
Defines records of fixed length.

RECORD-SIZE = <u>32768</u> / <integer 1..32768>
Length of the unedited output records.

FILE = *SYSOUT(...)
Output is to system file SYSOUT in the prescribed standard format; this format can be modified by means of the SET-PAGE-LAYOUT statement.

RECORD-FORMAT =
Defines the record format of the unedited output record.

RECORD-FORMAT = <u>*UNDEFINED</u>
Defines records of undefined length.

RECORD-FORMAT = *VARIABLE(...)
Defines records of variable length.

RECORD-SIZE = <u>32768</u> / <integer 4..32768>
Maximum length of the unedited output record.

RECORD-FORMAT = *FIXED(...)
Defines records of fixed length.

RECORD-SIZE = <u>32768</u> / <integer 1..32768>
Length of the unedited output records.

LINK-NAME = <u>PCOUT</u> / <filename 1..8 without-gen>
Specifies the PERCON link name for reference in keywords and in the ADD-FILE-LINK command.

OUTPUT-EXIT =
Name of the user module or entry point in the user module which processes the output records immediately before output. It may only be specified when copying a file.

OUTPUT-EXIT = <u>*NO</u>
No processing by a user module takes place.

OUTPUT-EXIT = *MODULE(...)

NAME = <name 1..8>
Name of the user module or entry point in the user module which handles further processing (see “Interface for output” on page 184).

ASSIGN-OUTPUT-TAPE

This statement is only permissible in the case of tape editing. It assigns the output tape and/or tapes. Furthermore the response to errors and the user modules to be connected can be defined for error situations.

ASSIGN**-OUT**PUT**-TAPE**
VOLUME = list-poss(2000): <alphanum-name 1..6>
,**LINK**-NAME = **PCOUT** / <filename 1..8 without-gen>
,**END-POS**ITION = ***B**EGIN**-OF-T**APE / ***UNLOAD-TAPE**
,**CODE** = ***EBCDIC** / ***ISO7** / ***OWN**(...)
***OWN**(...)
| **NAME** = <name 1..8>
,**DEV**ICE**-TYPE** = ***TAPE** / <structured-name 1..8>
,**REN**AME-**VOL**UME = ***NO** / list-poss: <alphanum-name 1..6>

VOLUME = list-poss: <alphanum-name 1..6>
Specifies the volume serial number(s) of the output tape(s).

LINK-NAME = PCOUT / <filename 1..8 without-gen>
Specifies the internal PERCON link name.

END-POSITION =
Specifies how the tape is to be positioned after processing.

END-POSITION = *BEGIN-OF-TAPE
The tape is rewound to BOT.

END-POSITION = *UNLOAD-TAPE
The tape is rewound and unloaded.

CODE =
Specifies the code in which the tape is written. For code conversion it should be noted that all data (including labels) on the tape is recoded.

CODE = *EBCDIC
The tape must be written in EBCDI code and it is processed without code conversion.

CODE = *ISO7
The tape must be written in IS07 code. The output data is converted to ISO7 code.

CODE = *OWN(...)
The contents of the tape are to be converted with the aid of a user-defined code conversion table contained in the module library assigned via SET-TASKLIB. Its contents are the 256 hexadecimal encryptions (X’00’ to X’FF’) of the characters to which the output characters are to be converted.
Conversion is based on the hexadecimal value of the output character. This value is added to the start address of the conversion table; the character at the resulting position of the conversion table replaces the output character.

NAME = <name 1..8>
Name of the user module or entry point in the user module containing the code conversion table.

DEVICE-TYPE =
Specifies the type of tape device.

DEVICE-TYPE = *TAPE
Tape device with the largest recording density.

DEVICE-TYPE = <structured-name 1..8>
Explicit specification of the type of tape device, e. g. TAPE-C4.

RENAME-VOLUME =
Specifies the new volume serial number(s) of the output tape(s). This operand is relevant only if there are standard labels on the input tape or MF/MV set.

i RENAME-VOLUME cannot be used in conjunction with VTS (Virtual Tape Server).

RENAME-VOLUME = *NO
The VSNs of the tapes specified using the VOLUME operand are retained.

RENAME-VOLUME = list-poss: <alphanum-name 1..6>
Restricted to the system administrator (TSOS privilege required).
Defines the new VSN(s) of an MF/MV set.

CHANGE-INPUT-TAPEPOSITION

This statement is only permissible in conjunction with tape editing. It is used to position the input tape prior to editing output data. The input tape must have been assigned using the ASSIGN-INPUT-TAPE statement.

The CHANGE-INPUT-TAPEPOSITION statement can be specified only following the ASSIGN-INPUT-TAPE statement, but it can precede or follow the ASSIGN-OUTPUT-FILE statement.

```
CHANGE-INPUT-TAPEPOSITION
-----------------------------------------------------------------------------------------
DIRECTION = *BACKWARD (...) / *FORWARD(...)

  *BACKWARD (...)
    |  DESTINATION = *BEGIN-OF-TAPE / *DOUBLE-TAPE-MARK / *BLOCKS(...) / *TAPE-MARKS(...)
    |    *BLOCKS(...)
    |      |  BLOCKS = <integer 1..2147483647>
    |    *TAPE-MARKS(...)
    |      |  TAPE-MARKS = <integer 1..2147483647>
  *FORWARD(...)
    |  DESTINATION = *DOUBLE-TAPE-MARK / *PAST-END-OF-TAPE / *END-OF-TAPE / *BLOCKS(...)
    |                / *TAPE-MARKS(...)
    |    *BLOCKS(...)
    |      |  BLOCKS = <integer 1..2147483647>
    |    *TAPE-MARKS(...)
    |      |  TAPE-MARKS = <integer 1..2147483647>
```

DIRECTION =
Specifies the direction in which the tape is to be positioned.

DIRECTION = *BACKWARD(...)
The input tape is to be positioned backward, i.e. towards the beginning of the tape.

DESTINATION =
Specifies the desired position of the tape.

DESTINATION = *BEGIN-OF-TAPE
The tape is positioned to the beginning of the tape.

DESTINATION = *DOUBLE-TAPE-MARK
The tape is positioned to the next double tape mark.

DESTINATION = *BLOCKS(...)

BLOCKS = <integer 1..2147483647>
Specifies the number of blocks by which the tape is to be repositioned.

DESTINATION = *TAPE-MARKS(...)

TAPE-MARKS = <integer 1..2147483647>
Number of tape marks by which the tape is to be repositioned.

i When reading blocks after positioning the tape backward (i.e. towards the beginning of the tape), it is not always possible to determine the current block count correctly. If this occurs, the tape is positioned backward starting at 99.999.999.

DIRECTION = *FORWARD(...)
The tape is positioned forwards, i.e. towards the end of the tape.

DESTINATION =
Specifies the desired position of the tape.

DESTINATION = *DOUBLE-TAPE-MARK
The tape is positioned to the next double tape mark.

DESTINATION = *PAST-END-OF-TAPE
Use of this operand is restricted to the system administrator.
The tape is positioned behind the logical end-of-tape. If there is no double tape mark on the tape, the tape is spooled out. This operand cannot be used for tape cartridges.

DESTINATION = *END-OF-TAPE
The tape is positioned to the logical end-of-tape.

DESTINATION = *BLOCKS(...)

BLOCKS = <integer 1..2147483647>
Specifies the number of blocks by which the tape is to be positioned towards the end of the tape.

DESTINATION = *TAPE-MARKS(...)

TAPE-MARKS = <integer 1..2147483647>
Specifies the number of tape marks by which the tape is to be positioned towards the end of the tape.

END

The END statement terminates the entry of statements to PERCON.

In the case of file copying and tape duplication, the END statement is used to start the transfer operation and to terminate PERCON.

In the case of tape editing, additional entries are required via the START-TAPE-PROCESSING statement in order to start the transfer operation. In this mode the END statement serves simply to terminate PERCON.

If only the END statement has been specified, a standard conversion run is performed. If no ADD-FILE-LINK commands have preceded this, i.e. no input or output files have been defined, PERCON uses the files with the link name PCIN or PCOUT.

END

The END statement does not have any operands.

MODIFY-PERCON-OPTIONS

This statement controls the scope of messages which PERCON outputs to SYSOUT and SYSLST.

MODIFY-**PER**CON-**OPT**IONS
SYSOUT-LOGGING = <u>***UNCHA**NGED</u> / ***ALL** / ***ERRORS** / ***NO**   ,**SYSLST-LOG**GING = <u>***UNCHA**NGED</u> / ***ALL** / ***ERRORS** / ***NO**

SYSOUT-LOGGING =
Defines the extent of the messages output to SYSOUT.
Default values:

Interactive and batch modes:	SYSOUT-LOGGING = *ALL
Subprogram:	SYSOUT-LOGGING = *NO

SYSOUT-LOGGING = <u>*UNCHANGED</u>
The setting of SYSOUT-LOGGING is not changed.

SYSOUT-LOGGING = *ALL
PERCON outputs all messages to SYSOUT.

SYSOUT-LOGGING = *ERRORS
Except for the final report on the number of records/blocks processed, PERCON outputs all messages to SYSOUT. (The final report contains a PER0029 or PER0030 message for each file or volume of the conversion step.)

SYSOUT-LOGGING = *NO
PERCON outputs no messages to SYSOUT. In the case of logic errors, no interactive guidance is given.

SYSLST-LOGGING =
Defines the extent of the log output to SYSLST.
Default values:

Interactive mode and subprogram:	SYSLST-LOGGING = *NO
Batch mode:	SYSLST-LOGGING = *ALL

SYSLST-LOGGING = <u>*UNCHANGED</u>
The setting of SYSLST-LOGGING is not changed.

SYSLST-LOGGING = *ALL
Except for the start message, PERCON outputs all messages to SYSLST.

SYSLST-LOGGING = *ERRORS
Except for the start message and the final report on the number of records/blocks processed, PERCON outputs all messages to SYSLST.

SYSLST-LOGGING = *NO
PERCON outputs no messages to SYSLST.

RESET-INPUT

This statement is used to reset all specifications made for the current conversion step; thereafter new statements for the next conversion step can be entered.

RESET-INPUT

The RESET-INPUT statement does not have any operands.

SELECT-INPUT-RECORDS

In the case of file copying, this statement serves to select records from an input file on the basis of certain criteria, and transfer them to one or more output files. In the case of tape editing, blocks from the input tape are selected. The positioning specifications then refer to the beginning of the block, and not to the beginning of the record.

Input files which do not match any of the selection criteria can be transferred to special output files. These are referred to as "residual files" in the following description. They are defined by means of the REMAINING-RECORDS keyword of the CONDITION operand.

It is possible to define a separate selection criterion for each output file using the SELECT-INPUT-RECORDS statement.

If several SELECT-INPUT-RECORDS statements in a conversion step refer to the same link name, only the last statement specified is executed for this link name.

SELECT-**INPUT-REC**ORDS
OUTPUT-**LINK-NAME** = ***STD** / list-poss(2000): <filename 1..8 without-gen> ,**COND**ITION = ***REMAIN**ING-**REC**ORDS / <text 7..1800 with-low>

OUTPUT-LINK-NAME =
Link names of the output files to which this statement is to refer.

OUTPUT-LINK-NAME = *STD
This statement refers to all output files specified up to now.

OUTPUT-LINK-NAME = list-poss(2000): <filename 1..8 without-gen>
If the statement is to refer only to a few output files, the link names of these files must be specified.

CONDITION =
This operand serves to define the selection criteria (conditions) or the residual files.

CONDITION = *REMAINING-RECORDS
The output files specified in the OUTPUT-LINK-NAME operand are to be used as residual files. All input records that do not match any of the selection criteria are written to these residual files.
It is possible to define more than one residual file in any one conversion step.

Note

The statements SET-RECORD-MAPPING, SET-GROUP-ATTRIBUTES and SET-PAGE-LAYOUT as well as user interfaces can be used for residual files without any restrictions.

If a user interface for input determines that a record is not to be transferred, this decision applies to all output files including the residual files.

If a record matches the selection criteria but is not transferred as a result of a decision made in the user interface for output, it is still considered to be selected and is therefore not written to a residual file.

In the START-TAPE-PROCESSING statement, the OUTPUT-LINK-NAME operand determines the defined output files to which the statement is to refer. This means that different output files can be referred to each time this statement is called. The specification *STD applies to all output files specified before with the exception of the residual files. If any of the link names of output files specified explicitly with this operand refers to a residual file, the statement will be rejected with message PER0062. During processing of the START-TAPE-PROCESSING statement, a block is only written to a residual file if it cannot be transferred to any of the output files specified in the OUTPUT-LINK-NAME operand, i.e. if the block does not match any of the selection criteria for any of these output files.

Defining a residual file is not useful unless a SELECT-INPUT-RECORDS statement that defines a condition takes effect for all “normal” output files. Otherwise there will be no records to be written to a residual file.

An example of how to make use of a residual file is given on page 199.

CONDITION = <text 7..1800 with-low>
This operand defines the condition under which an input record is selected for output. The record will be transferred if the condition is satisfied (i.e. when “true” applies). The condition can consist of one or more comparisons linked by logical AND or OR. AND has priority over OR. This priority rule can be changed through the use of parentheses, thus improving readability and shortening the program runtime, since individual comparisons are not processed several times. The condition is processed from left to right.

The runtime can be reduced by deliberately putting the comparisons into a specific sequence (in particular, comparisons which for the most part yield the same result):

e.g.: a OR b OR c
The comparison satisfied most frequently must be on the left.

a AND b AND c
The comparison satisfied most frequently must be on the right.

Restriction on parenthesis nesting: 1800 bytes for CONDITION.

Several comparisons linked by AND in a relationship result in the value “true” if all comparisons are “true”.

Several relationships linked by OR or comparisons which do not belong to any relationship result in the value “true” if at least one relationship or one comparison is “true”.

In the case of two comparisons the following applies:

1st comparison	Link	2nd comparison	Result
true	AND	true	true
true	AND	false	false
false	AND	true	false
false	AND	false	false
true	OR	true	true
true	OR	false	true
false	OR	true	true
false	OR	false	false

The syntax used to specify the condition in the CONDITION operand is as follows:

```
condition  : = (comparison [AND/OR comparison] ...)

comparison : = (position[,length]) EQ/NE N[UMERIC]/AL[PHA]
           or  (position[,length]) EQ/NE
                                   AS[CENDING]/D[ESCENDING]/M[ODULE](modname)
           or  (position[,length[,C]]) op <c-string 1..256>/<x-string 1..512>
           or  (position,length,Z/P) op Z‘digit‘/P‘digit‘
           or  RECCNT(linkname) op Z‘digit‘/P‘digit‘
           or  BYTCNT(linkname) op Z‘digit‘/P‘digit‘
           or  BLKCNT(linkname) op Z‘digit‘/P‘digit‘
           or  RECLEN(linkname) op Z‘digit‘/P‘digit‘
           or  condition

position   : = <integer 1..32768>

length     : = <integer 1..256>
           or  RECLEN (record length), where 1 <= RECLEN <= 256 must apply
           or  1 <= RECLEN-reduction <= 256
               If the condition is not fulfilled, the conversion step is
               aborted and message PER0042 is output.

op         : = EQ/NE/GT/LT/GE/LE

digit      : = same as in “literals”, see page 78
```

The specifications for the comparison operators are equivalents and have the following meanings:

Special characters	Combinations of letters	Meaning
=	EQ	equal to
>	GT	greater than
<	LT	less than
>=	GE	greater than or equal to
<=	LE	less than or equal to
<>	NE	not equal to

Record selection

Records are selected as the result of one of the following operations:

a) Checking whether the character format of a file is numeric or alphabetic

The specified field is checked to ensure that it has the required character format.

NUMERIC	PERCON checks whether the specified field contains digits only.
ALPHA	PERCON checks whether the specified field contains only uppercase letters, lowercase letters, German umlauts and blanks.

Examples

```
COND=((5,3) EQ NUMERIC)
COND=((20,4) EQ NUMERIC AND (25,5) EQ ALPHA)
```

b) Checking whether the fields are sorted in ascending or descending order

PERCON checks whether the specified field has been sorted in ascending or descending order, i.e. whether it has a higher or lower value than the field of the previous record. If the SELECT-INPUT-RECORDS condition prevents a record from being transferred to the output file, the field in this record cannot be used as the comparison field for comparison with the field in the next record.

ASCENDING	PERCON checks for sorting in ascending order
DESCENDING	PERCON checks for sorting in descending order

c) Checking the validity of the characters in a field by reference to a user table

MODULE	PERCON checks whether the specified field contains only characters authorized by the user.
modname	“modname” specifies the name of a user module (1st CSECT name) or of an area (ENTRY name) in a user module comprising 256 characters X’00’ or X’FF’. PERCON adds the value of each character from the field

to the start address of the user module or user area. If the resulting position contains X’00’, the character is permitted; if it contains X’FF’, the character is not permitted. If one byte in the field fails to satisfy the condition, the comparison is not satisfied.

d) Comparing a field with a literal (constant)

The field is compared with the specified literal. Special rules apply in the event of differing formats and lengths (see the table below).

e) Comparing a keyword with a literal (constant)

The current value of “keyword” (see page 79) is compared with the specified literal (see the table below).

Example

Two input files (IN1 and IN2) are to be converted to one output file. All records of the first input file, and all records of the second input file, from the fifth one onwards, are to be transferred. The condition for this is as follows:

```
COND=(RECCNT(IN1) GT Z'0' AND RECCNT(IN2) EQ Z'0' OR RECCNT(IN2) GE Z'5')
```

Examples of parenthesis nesting:

The first example presents a comparison of two fields.

If the first field contains an unpacked number > 100 **or** < 50 **and** the second field contains the character string ’1.1’ **or** ’2.2’ at the same time, the record is transferred.

```
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=02,-
//                     CONDITION=(-
//     (   (10,5,Z) GT Z'100' -
//     OR  (10,5,Z) LT Z'50' -
//     ) -
// AND (   (20,3,C) EQ C'1.1' -
//     OR  (20,3,C) EQ C'2.2' -
// )                          )
```

The second example illustrates the possibilities of parenthesis nesting. The comparisons are numbered from V1 to V12.

A record is output only if:

1. V1, V2 **and** V3 are true **and**
   at least one of the comparisons V4, V5 or V6 is true **and** V7 is true.
2. V1, V2 **and** V3 are true **and**
   at least one of the comparisons V4, V5 or V6 is true **and** V8 and V9 are true **and**
   at least one of the comparisons V10, V11 **or** V12 is true.

```
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=03,-
//                     CONDITION=(-
//               RECCNT(IN1) LE Z'5000' -                          V1
// AND           RECCNT(03) LT Z'300' -                            V2
// AND           (28,5) EQ ASCENDING -                             V3
// AND (         (6,1,C) EQ 'P' -                                  V4
//       OR      (6,2,C) EQ 'ST' -                                 V5
//       OR      (6,3,C) EQ 'MOD' -                                V6
//       ) -
// AND (         (20,5) EQ NUMERIC -                               V7
//     OR        RECLEN(IN1) GT Z'120' -                           V8
//        AND    (35,2) EQ MODULE(ABISK) -                         V9
//        AND (  (37,3,P) LT Z'230' -                              V10
//            OR (37,3,P) EQ Z'241' -                              V11
//            OR (37,3,P) GT Z'260' -                              V12
//            ) -
//     )                        )
```

Conversion rules in the event of differing character formats and unequal field length

The CONDITION operand specifies operands which are to be compared with one another. If two operands specified for comparison have different character formats or are of different lengths, the following conditions apply:

Literal		**C,X**	**Z,P**	**Comments**
Field	Keyword			
C		+	-	If “field” and “literal” are of different lengths, the shorter of the two is filled to the right with blanks for the comparison. If part or all of “field” lies outside a variable-length input record, the condition is considered fulfilled for “<>”, but not fulfilled for all other comparison operators. The comparison is based on the EBCDIC code.
P,Z	RECCNT, BYTCNT, BLKCNT, RECLEN	-	+	same as in “literals”, see page 78

+ valid comparison

- invalid comparison

SET-GROUP-ATTRIBUTES

This statement is permitted for file copying only. It serves to form groups by combining input records which have the same grouping criterion. The formation of groups results in implicit format editing. Both group header and group trailer lines can be formatted. User lines are lines containing text written by the user himself. This text can consist of input record fields, keywords and literals. The user lines are referred to as group header or group trailer lines, depending on whether they precede a group or follow it.

If several SET-GROUP-ATTRIBUTES statements of the same group level refer to the same link name, only the last statement specified is executed for this link name in the group level specified.

SET-GROUP-**ATTR**IBUTES

OUTPUT-**LINK-NAME** = ***STD** / list-poss(2000): <filename 1..8 without-gen>

,**GR**OUP-**LEV**EL = **1** / <integer 1..8>

,**GR**OUP-**CONTR**OL = ***ALL** / list-poss(2000): ***FIELD**(...)

    ***FIELD**(...)

        POSITION = <integer 1..32767>

        ,**LEN**GTH = **1** / <integer 1..256>

,**GR**OUP-**HEAD**ER = ***NONE** / list-poss(2000): ***FIELD**(...) / ***REC**ORD-**LEN**GTH(...) / ***BYTE-COUNT**ER(...) / ***REC**ORD-**COUNT**ER(...) / ***PAGE-COUNT**ER(...) / ***DATE**(...) / ***TIME**(...) / <c-string 1..204 with-low>(...) / <x-string 1..408>(...) / <integer -2147483648..2147483647>(...) / ***NEW-PAGE** / ***SPACING**(...) / ***MOD**ULE(...)

    ***FIELD**(...)

        INPUT-POSITION = <integer 1..32768>

        ,**INPUT-LEN**GTH = **1** / <integer 1..204>

        ,**INP**UT-**FORM**AT = ***CHAR**ACTER / ***ZONED-DECIMAL** / ***PACKED-DECIMAL**

        ,**OUT**PUT-**POS**ITION = <integer 1..204>

        ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..204>

        ,**OUT**PUT-**FORM**AT = ***STD** / ***CHAR**ACTER / ***HEX**ADECIMAL / ***BIN**ARY / <c-string 2..204 with-low> / ***ZONED-DECIMAL** / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

continued ➠

```
*RECORD-LENGTH(...)
  | LINK-NAME = <filename 1..8 without-gen>
  | ,OUTPUT-POSITION = <integer 1..203>
  | ,OUTPUT-LENGTH = *STD / <integer 1..204>
  | ,OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..204 with-low> / *ZONED-DECIMAL /
  |         *SIGNED-DECIMAL / *DECIMAL
*BYTE-COUNTER(...)
  | LINK-NAME = <filename 1..8 without-gen>
  | ,OUTPUT-POSITION = <integer 1..203>
  | ,OUTPUT-LENGTH = *STD / <integer 1..204>
  | ,OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..204 with-low> / *ZONED-DECIMAL /
  |         *SIGNED-DECIMAL / *DECIMAL
*RECORD-COUNTER(...)
  | LINK-NAME = <filename 1..8 without-gen>
  | ,OUTPUT-POSITION = <integer 1..203>
  | ,OUTPUT-LENGTH = *STD / <integer 1..204>
  | ,OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..204 with-low> / *ZONED-DECIMAL /
  |         *SIGNED-DECIMAL / *DECIMAL
*PAGE-COUNTER(...)
  | LINK-NAME = <filename 1..8 without-gen>
  | ,OUTPUT-POSITION = <integer 1..203>
  | ,OUTPUT-LENGTH = *STD / <integer 1..204>
  | ,OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..204 with-low> / *ZONED-DECIMAL /
  |         *SIGNED-DECIMAL / *DECIMAL
*DATE(...)
  | OUTPUT-POSITION = <integer 1..197>
  | ,OUTPUT-LENGTH = *STD / <integer 8..204>
  | ,OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
  | ,CENTURY = *NO / *YES
```

continued ➠

***TIME**(...)

  OUTPUT-**POS**ITION = <integer 1..197>

  ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 8..204>

  ,**OUT**PUT-**FORM**AT = ***CHAR**ACTER / ***HEX**ADECIMAL / ***BIN**ARY

<c-string with-low>(...)

  OUTPUT-**POS**ITION = <integer 1..204>

  ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..204>

  ,**OUT**PUT-**FORM**AT = ***CHAR**ACTER / ***HEX**ADECIMAL / ***BIN**ARY

<x-string>(...)

  OUTPUT-**POS**ITION = <integer 1..204>

  ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..204>

  ,**OUT**PUT-**FORM**AT = ***CHAR**ACTER / ***HEX**ADECIMAL / ***BIN**ARY

<integer>(...)

  OUTPUT-**POS**ITION = <integer 1..204>

  ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..204>

  ,**OUT**PUT-**FORM**AT = ***ZONED-DECIMAL** / <c-string 2..204 with-low> / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

***SPACING**(...)

  LINES = **1** / <integer 1..15>

***MOD**ULE(...)

  NAME = <name 1..8>

continued ➠

```
,GROUP-TRAILER = *NONE / list-poss(2000): *FIELD(...) / *SUM-FIELD(...) / *GROUP-COUNTER(...) /
      *RECORD-LENGTH(...) / *BYTE-COUNTER(...) / *RECORD-COUNTER(...) / *PAGE-COUNTER(...) /
      *DATE(...) / *TIME(...) / <c-string 1..204 with-low>(...) / <x-string 1..408>(...) /
      <integer -2147483648..2147483647>(...) / *NEW-PAGE / *SPACING(...) / *MODULE(...)

   *FIELD(...)
      |  INPUT-POSITION = <integer 1..32768>
      | ,INPUT-LENGTH = 1 / <integer 1..204>
      | ,INPUT-FORMAT = *CHARACTER / *ZONED-DECIMAL / *PACKED-DECIMAL
      | ,OUTPUT-POSITION = <integer 1..204>
      | ,OUTPUT-LENGTH = *STD / <integer 1..204>
      | ,OUTPUT-FORMAT = *STD / *CHARACTER / *HEXADECIMAL / *BINARY /
      |       <c-string 2..204 with-low> / *ZONED-DECIMAL / *SIGNED-DECIMAL /
      |       *DECIMAL
   *SUM-FIELD(...)
      |  INPUT-POSITION = <integer 1..32768>
      | ,INPUT-LENGTH = 1 / <integer 1..16>
      | ,INPUT-FORMAT = *ZONED-DECIMAL / *PACKED-DECIMAL
      | ,OUTPUT-POSITION = <integer 1..204>
      | ,OUTPUT-LENGTH = *STD / <integer 1..204>
      | ,OUTPUT-FORMAT = *ZONED-DECIMAL / <c-string 2..204 with-low> / *SIGNED-DECIMAL /
      |       *DECIMAL
   *GROUP-COUNTER(...)
      |  LINK-NAME = <filename 1..8 without-gen>
      | ,GROUP-LEVEL = <integer 1..8>
      | ,OUTPUT-POSITION = <integer 1..203>
      | ,OUTPUT-LENGTH = *STD / <integer 1..204>
      | ,OUTPUT-FORMAT = C ' ZZZ' / <c-string 2..204 with-low> / *ZONED-DECIMAL-LEFT
```

continued ➠

***REC**ORD-**LEN**GTH(...)

LINK-NAME = <filename 1..8 without-gen>

,**OUT**PUT-**POS**ITION = <integer 1..203>

,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..204>

,**OUT**PUT-FORMAT = ***ZON**ED-**DEC**IMAL-**LEFT** / <c-string 2..204 with-low> / ***ZONED-DECIMAL** / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

***BYTE-COUNT**ER(...)

LINK-NAME = <filename 1..8 without-gen>

,**OUT**PUT-**POS**ITION = <integer 1..203>

,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..204>

,**OUT**PUT-FORMAT = ***ZON**ED-**DEC**IMAL-**LEFT** / <c-string 2..204 with-low> / ***ZONED-DECIMAL** / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

***REC**ORD-**COUNT**ER(...)

LINK-NAME = <filename 1..8 without-gen>

,**OUT**PUT-**POS**ITION = <integer 1..203>

,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..204>

,**OUT**PUT-FORMAT = ***ZON**ED-**DEC**IMAL-**LEFT** / <c-string 2..204 with-low> / ***ZONED-DECIMAL** / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

***PAGE-COUNT**ER(...)

LINK-NAME = <filename 1..8 without-gen>

,**OUT**PUT-**POS**ITION = <integer 1..203>

,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..204>

,**OUT**PUT-FORMAT = ***ZON**ED-**DEC**IMAL-**LEFT** / <c-string 2..204 with-low> / ***ZONED-DECIMAL** / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

***DATE**(...)

OUTPUT-**POS**ITION = <integer 1..197>

,**OUT**PUT-**LEN**GTH = ***STD** / <integer 8..204>

,**OUT**PUT-FORMAT = ***CHAR**ACTER / ***HEX**ADECIMAL / ***BIN**ARY

,**CENTURY** = ***NO** / ***Y**ES

continued ➠

```
*TIME(...)
   |  OUTPUT-POSITION = <integer 1..197>
   | ,OUTPUT-LENGTH = *STD / <integer 8..204>
   | ,OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
<c-string>(...)
   |  OUTPUT-POSITION = <integer 1..204>
   | ,OUTPUT-LENGTH = *STD / <integer 1..204>
   | ,OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
<x-string>(...)
   |  OUTPUT-POSITION = <integer 1..204>
   | ,OUTPUT-LENGTH = *STD / <integer 1..204>
   | ,OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
<integer>(...)
   |  OUTPUT-POSITION = <integer 1..204>
   | ,OUTPUT-LENGTH = *STD / <integer 1..204>
   | ,OUTPUT-FORMAT = *ZONED-DECIMAL / <c-string 2..204 with-low> / *SIGNED-DECIMAL /
   |         *DECIMAL
*SPACING(...)
   |  LINES = 1 / <integer 1..15>
*MODULE(...)
   |  NAME = <name 1..8>
```

OUTPUT-LINK-NAME =
Link names of the output files to which this statement is to refer.

OUTPUT-LINK-NAME = *STD
The statement references all output files specified up to now.

OUTPUT-LINK-NAME = list-poss(2000): <filename 1..8 without-gen>
If the statement is to reference only certain output files, the link names of these files must be specified.

GROUP-LEVEL = 1 / <integer 1..8>
Specifies the group level for which this SET-GROUP-ATTRIBUTES statement is valid. Up to 8 nested group levels are possible. Specifying group levels assigns priorities to the group criteria. A separate SET-GROUP-ATTRIBUTES statement must be specified for each group level. In the case of a group break on one level, groups with a lower GROUP-LEVEL are also subjected to a group break (1: highest level, 8: lowest level)

GROUP-CONTROL =
Specifies the grouping criterion.

GROUP-CONTROL = *ALL
The entire input file is interpreted as one group. This specification is only possible in the highest group level (GROUP-LEVEL=1).

GROUP-CONTROL = *FIELD(...)
A field in the input record is specified as the grouping criterion.

POSITION = <integer 1..32767>
Specifies the position where the comparison field begins, relative to the beginning of the input record.

LENGTH = 1 / <integer 1..256>
Specifies the length of the comparison field, limited by the record length of the input record.

GROUP-HEADER =
Defines user lines which form the group header. These lines are output before the first line of the corresponding group when there is a group break.

GROUP-HEADER = *NONE
No user lines are output.

GROUP-HEADER = *FIELD(...)
Transfers a field of the input record (input field) to a field in the user line (output field).

INPUT-POSITION = <integer 1..32768>
Specifies the position where the input field begins, relative to the beginning of the input record.

INPUT-LENGTH = 1 / <integer 1..204>
Specifies the length of the input field, limited by the record length of the input record.

INPUT-FORMAT = *CHARACTER / *ZONED-DECIMAL / *PACKED-DECIMAL
Format of the input field in the input record (see page 80ff).
The input record field which is to be edited using a mask must not be longer than 203 bytes for INPUT-FORMAT=*ZONED-DECIMAL and no longer than 102 bytes for INPUT-FORMAT=*PACKED-DECIMAL.

OUTPUT-POSITION = <integer 1..204>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..204>
Specifies the output field.

The table below shows the default values:

OUTPUT-FORMAT	**INPUT-FORMAT**		
	CHARACTER	**ZONED-DECIMAL**	**PACKED-DECIMAL**
CHARACTER	INPUT-LENGTH		
HEXADECIMAL	2 * INPUT-LENGTH		
BINARY	8 * INPUT-LENGTH		
<c-string> (mask)		Mask length	Mask length
SIGNED-DECIMAL		INPUT-LENGTH + 1	2*INPUT-LENGTH
DECIMAL		INPUT-LENGTH + 1	2*INPUT-LENGTH
ZONED-DECIMAL		INPUT-LENGTH	2*INPUT-LENGTH-1

OUTPUT-FORMAT = *STD / *CHARACTER / *HEXADECIMAL / *BINARY / <c-string 2..204 with-low> / *ZONED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

Only the following combinations are permissible:

OUTPUT-FORMAT	**INPUT-FORMAT**		
	CHARACTER	**ZONED-DECIMAL**	**PACKED-DECIMAL**
CHARACTER	+		
HEXADECIMAL	+		
BINARY	+		
<c-string> (mask)		+	+
SIGNED-DECIMAL		+	+
DECIMAL		+	+
ZONED-DECIMAL		+	+

The table below shows the default values:

INPUT-FORMAT	**OUTPUT-FORMAT**
CHARACTER	CHARACTER
PACKED-DECIMAL	ZONED-DECIMAL
ZONED-DECIMAL	ZONED-DECIMAL

GROUP-HEADER = *RECORD-LENGTH(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file whose current record length is to be used in the user line.

OUTPUT-POSITION = <integer 1..203>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH =*STD / <integer 1..204>
Length of the output field.

The table below shows the default values:

OUTPUT-FORMAT	OUTPUT-LENGTH
ZONED-DECIMAL-LEFT	8 characters
ZONED-DECIMAL	8 characters
SIGNED-DECIMAL	9 characters
DECIMAL	9 characters

OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..204 with-low> / *ZONED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field, (see page 80ff).

GROUP-HEADER = *BYTE-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file whose current byte counter is to be used in the user line.

OUTPUT-POSITION = <integer 1..203>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..204>
Length of the output field.

The table below shows the default values:

OUTPUT-FORMAT	OUTPUT-LENGTH
ZONED-DECIMAL-LEFT	10 characters
ZONED-DECIMAL	10 characters
SIGNED-DECIMAL	11 characters
DECIMAL	11 characters

OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..204 with-low> / *ZONED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

GROUP-HEADER = *RECORD-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file whose current record counter value is to be used in the user line.

OUTPUT-POSITION = <integer 1..203>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..204>
Specifies the length of the output field.

The table below shows the default values:

OUTPUT-FORMAT	**OUTPUT-LENGTH**
ZONED-DECIMAL-LEFT	8 characters
ZONED-DECIMAL	8 characters
SIGNED-DECIMAL	9 characters
DECIMAL	9 characters

OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..204 with-low> / *ZONED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

GROUP-HEADER = *PAGE-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the output file whose current page counter value is to be used in the user line.

OUTPUT-POSITION = <integer 1..203>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..204>
Specifies the length of the output field.

The table below shows the default values:

OUTPUT-FORMAT	**OUTPUT-LENGTH**
ZONED-DECIMAL-LEFT	6 characters
ZONED-DECIMAL	6 characters
SIGNED-DECIMAL	7 characters
DECIMAL	7 characters

OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..204 with-low> / *ZONED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

GROUP-HEADER = *DATE(...)
Specifies the start date of the PERCON conversion step as an 8-digit string in the form yy-mm-dd.

OUTPUT-POSITION = <integer 1..197>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 8..204>
Specifies the length of the output field.

The table below shows the default values:

OUTPUT-FORMAT	**OUTPUT-LENGTH**
CHARACTER	8 characters
HEXADECIMAL	16 characters
BINARY	64 characters

OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
Specifies the format of the output field (see page 80ff).

CENTURY = *NO / *YES
If CENTURY=*YES is specified, the date is output as a 10
digit character string, with the century specification in the format yyyy-mm-dd; this reduces the maximum value for OUTPUT-POSITION by 2 to 195 and the minimum value for OUTPUT-LENGTH is increased by 2 to 10.

GROUP-HEADER = *TIME(...)
Specifies the start time of the PERCON conversion step as an 8-digit string in the form hh:mm:ss.

OUTPUT-POSITION = <integer 1..197>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 8..204>
Specifies the length of the output field.

The table below shows the default values:

OUTPUT-FORMAT	**OUTPUT-LENGTH**
CHARACTER	8 characters
HEXADECIMAL	16 characters
BINARY	64 characters

OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
Specifies the format of the output field (see page 80ff).

GROUP-HEADER =<c-string 1..204 with-low>(...)

OUTPUT-POSITION = <integer 1..204>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..204>
Specifies the length of the output field.

The table below shows the default values:

OUTPUT-FORMAT	**OUTPUT-LENGTH**
CHARACTER	Length of the c-string
HEXADECIMAL	2 * Length of the c-string
BINARY	8 * Length of the c-string

OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
Specifies the format of the output field (see page 80ff).

GROUP-HEADER =<x-string 1..408>(...)

OUTPUT-POSITION = <integer 1..204>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..204>
Specifies the length of the output field.

The table below shows the default values:

OUTPUT-FORMAT	**OUTPUT-LENGTH**
CHARACTER	0,5 * Length of the x-string
HEXADECIMAL	Length of the x-string
BINARY	4 * Length of the x-string

OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
Specifies the format of the output field (see page 80ff).

GROUP-HEADER =<integer -2147483648..2147483647>(...)

OUTPUT-POSITION = <integer 1..204>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..204>
Specifies the length of the output field. The default value is the minimum length required to fit the output number in the specified format.

OUTPUT-FORMAT = *ZONED-DECIMAL / <c-string 2..204 with-low> / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

GROUP-HEADER = *NEW-PAGE
Causes a feed to the next page and output of the previously formatted user line. After execution, a further user line can be generated.

GROUP-HEADER = *SPACING(...)
Causes output of the previously formatted user line and a feed of from 1 to 15 lines. After execution of the operand, a further user line can be generated.

LINES = 1 / <integer 1..15>
Specifies the number of line feeds.

GROUP-HEADER = *MODULE(...)

NAME = <name 1..8>
Editing of a group header user line takes place in a user module. The name of the user module or the entry point in the user module must be specified. A user line which has been created but not yet printed is made available to the user routine (see “Interface for group processing” on page 185).

GROUP-TRAILER = *NONE / list-poss(2000): *FIELD(...) / *SUM-FIELD(...) / *GROUP-COUNTER(...) / *RECORD-LENGTH / *BYTE-COUNTER(...) / *RECORD-COUNTER(...) / *PAGE-COUNTER(...) / *DATE(...) / *TIME(...) / <c-string 1..204 with-low>(...) / <x-string 1..408>(...) / <integer -2147483648..2147483647>(...) / *NEW-PAGE / *SPACING(...) / *MODULE(...)
Defines user lines which form a group trailer. These lines are output after the last line of the corresponding group when there is a group break.

When GROUP-TRAILER = *FIELD, the FIELD information is taken from the last record of the group.

See GROUP-HEADER (page 133ff) for a description of the operands (except for SUM-FIELD(...) and GROUP-COUNTER(...).

GROUP-TRAILER = *SUM-FIELD(...)
The values in the numbers field of all records of the last groups processed are totaled. This sum is transferred to the SUM-FIELD of the user line.The SUM-FIELD must be sufficiently long. Several fields can be totaled.

INPUT-POSITION = <integer 1..32768>
Specifies the position of the numbers field, relative to the beginning of the input record.

INPUT-LENGTH = 1 / <integer 1..16>
Specifies the length of the numbers field.

INPUT-FORMAT = *ZONED-DECIMAL / *PACKED-DECIMAL
Specifies the format of the numbers field (see page 80ff).

OUTPUT-POSITION = <integer 1..204>
Specifies the position where the sum field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..204>
Specifies the length of the sum field.

The default value is the minimum length required for accommodating the number to be output in the format specified. When outputting with a mask, *STD is the mask length. The following values are used for *STD in the conversions listed below:

OUTPUT-FORMAT	INPUT-FORMAT	
	ZONED-DECIMAL	**PACKED-DECIMAL**
SIGNED-DECIMAL	INPUT-LENGTH + 1	2 * INPUT-LENGTH
DECIMAL	INPUT-LENGTH + 1	2 * INPUT-LENGTH
ZONED-DECIMAL		2 * INPUT-LENGTH-1

In all other cases, INPUT-LENGTH is assumed.

OUTPUT-FORMAT = *ZONED-DECIMAL / <c-string 2..204 with-low> / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the sum field (see page 80ff).

GROUP-TRAILER = *GROUP-COUNTER(...)
Specifies counters per group. GROUP-COUNTER is available for group levels 1 to 8 of each output medium. Each counter is unambiguously defined by its file link name and by a group level.

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file from which a group counter is to be used in the user line.

GROUP-LEVEL = <integer 1..8>
Specifies which group counter of the input/output file labelled LINK-NAME is to be used.

OUTPUT-POSITION = <integer 1..203>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 4..204>
Specifies the length of the output field. The standard (default) value is either 4 when OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT or the mask length.

OUTPUT-FORMAT = C' ZZZ' / <c-string 2..204 with-low> / *ZONED-DECIMAL-LEFT
Specifies the format of the output field (see page 80ff).

SET-PAGE-LAYOUT

This statement defines the page layout of the output file.

The SET-PAGE-LAYOUT statement can be omitted if FILE=*SYSOUT or FILE=*SYSLST has been specified in the ASSIGN-OUTPUT-FILE statement or if a SET-GROUP-ATTRIBUTES statement has been specified. In this case, processing is carried out in accordance with the default values (see page 21).

If several SET-PAGE-LAYOUT statements in a conversion step refer to the same link name, only the last statement specified is executed for this link name.

The following lines are not taken into account when editing an output for SYSOUT:

– header lines (irrespective of the specification in the HEADER-LINE operand)
– blank lines
– user lines (irrespective of the specification in the USER-LINE operand).

Any control characters X´00´ thru X´3F´ contained in an output record to be edited will be replaced. This can be suppressed by means of the REPLACE-CONTROL-CHAR operand, for instance if printer control characters are inserted by the user.

SET-PAGE-LAYOUT

```
 OUTPUT-LINK-NAME = *STD / list-poss(2000): <filename 1..8 without-gen>
,HEADER-LINE = *STD / *NONE / *PAGE-COUNTER / *TITLE
,OUTPUT-AREA = list-poss(2000): *LINE-RANGE (...)
    *LINE-RANGE (...)
       |   FIRST-LINE = 5 / <integer 1..112>
       |  ,LAST-LINE = 66 / <integer 1..112>
,SPACING = 0 / <integer 0..3>
,OUTPUT-FORMAT = *STD / *CHARACTER / *HEXADECIMAL / *BOTH
,LINE-SIZE = *STD / <integer 1..204>
,COLUMN-SIZE = *LINE-SIZE / <integer 1..204>
,SUPPRESS-EQUAL-LINES = *YES / *NO
,REPLACE-CONTROL-CHAR = *YES / *NO
```

continued ➠

,**USER-LINE** = ***NONE** / list-poss(2000): ***REC**ORD-**LEN**GTH(...) / ***BLOCK-COUNT**ER(...) /
***BYTE-COUNT**ER(...) / ***REC**ORD-**COUNT**ER(...) / ***PAGE-COUNT**ER(...) / ***DATE**(...) / ***TIME**(...) /
<c-string 1..204 with-low>(...) / <x-string 1..408>(...)

***REC**ORD-**LEN**GTH(...)

LINK-NAME = <filename 1..8 without-gen>

,**OUT**PUT-**LINE** = <integer 1..112>

,**OUT**PUT-**COL**UMN = <integer 1..197>

,**OUT**PUT-**LENGTH** = ***STD** / <integer 1..32>

***BLOCK-COUNT**ER(...)

LINK-NAME = <filename 1..8 without-gen>

,**OUT**PUT-**LINE** = <integer 1..112>

,**OUT**PUT-**COL**UMN = <integer 1..197>

,**OUT**PUT-**LENGTH** = ***STD** / <integer 1..32>

***BYTE-COUNT**ER(...)

LINK-NAME = <filename 1..8 without-gen>

,**OUT**PUT-**LINE** = <integer 1..112>

,**OUT**PUT-**COL**UMN = <integer 1..195>

,**OUT**PUT-**LENGTH** = ***STD** / <integer 1..32>

***REC**ORD-**COUNT**ER(...)

LINK-NAME = <filename 1..8 without-gen>

,**OUT**PUT-**LINE** = <integer 1..112>

,**OUT**PUT-**COL**UMN = <integer 1..197>

,**OUT**PUT-**LENGTH** = ***STD** / <integer 1..32>

***PAGE-COUNT**ER(...)

LINK-NAME = <filename 1..8 without-gen>

,**OUT**PUT-**LINE** = <integer 1..112>

,**OUT**PUT-**COL**UMN = <integer 1..199>

,**OUT**PUT-**LENGTH** = ***STD** / <integer 1..32>

***DATE**(...)

OUTPUT-**LINE** = <integer 1..112>

,**OUT**PUT-**COL**UMN = <integer 1..197>

,**CENTURY** = ***NO** / ***Y**ES

continued ➠

```
*TIME(...)
  | OUTPUT-LINE = <integer 1..112>
  | ,OUTPUT-COLUMN = <integer 1..197>
<c-string>(...)
  | OUTPUT-LINE = <integer 1..112>
  | ,OUTPUT-COLUMN = <integer 1..204>
<x-string>(...)
  | OUTPUT-LINE = <integer 1..112>
  | ,OUTPUT-COLUMN = <integer 1..204>
```

OUTPUT-LINK-NAME =
Link names of the output files to which this statement is to refer.

OUTPUT-LINK-NAME = *STD
This statement refers to all output files specified up to now.

OUTPUT-LINK-NAME = list-poss(2000): <filename 1..8 without-gen>
If the statement is to reference only certain output files, the link names of these files must be specified.

HEADER-LINE =
Specifies which header line is output for a page.

HEADER-LINE = *STD
File copying: HEADER-LINE = *PAGE-COUNTER
Tape editing: HEADER-LINE = *TITLE

HEADER-LINE = *NONE
PERCON does not output a header line.

HEADER-LINE = *PAGE-COUNTER
The text `PAGE xxxxxx` is inserted in line 1, column 119, of each page. `xxxxxx` is the current status of the page counter.

HEADER-LINE = *TITLE
PERCON outputs a standard header line in the first line.

```
1            36         41              86          91          119
PERCON       DATE       yyyy-mm-dd      TIME        hh:mm:ss    PAGE xxxxxx
```

OUTPUT-AREA =
This operand uses FIRST-LINE and LAST-LINE to define the start and end of a line range per print page to which PERCON outputs the output records. Several such lines can be defined per print page.

This operand can be used to overwrite the header line.

OUTPUT-AREA = *LINE-RANGE(...)
Defines the line range.

FIRST-LINE = 5 / <integer 1..112>
Specifies the start line for data output.

LAST-LINE = 66 / <integer 1..112>
Specifies the end line for data output.

SPACING = 0 / <integer 0..3>
Specifies the number of blank lines output after every print line.

OUTPUT-FORMAT =
Specifies the output format of the data to be output.

OUTPUT-FORMAT = *STD
File copying: OUTPUT-FORMAT = *CHARACTER
Tape editing: OUTPUT-FORMAT = *BOTH

OUTPUT-FORMAT =*CHARACTER
Outputs data in the form of characters. Nonprintable characters are only represented as blanks if REPLACE-CONTROL-CHAR=*YES applies and none of the tables of printable characters supplied by XHCS is available.

OUTPUT-FORMAT = *HEXADECIMAL
Outputs data in hexadecimal format.

OUTPUT-FORMAT = *BOTH
Data is output in hexadecimal format. Printable characters are additionally output in a second line above the corresponding hexadecimal codes. Nonprintable characters are only represented as blanks if REPLACE-CONTROL-CHAR=*YES applies and none of the tables of printable characters supplied by XHCS is available. Therefore 2 data lines are occupied by each output record (or each part of it if it is longer than the line, see the LINE-SIZE operand).

LINE-SIZE = *STD / <integer 1..204>
Number of characters in the record to be edited that are to be output per line. This enables an output record to be distributed over several lines. The length of an output line is determined not only by the LINE-SIZE operand, but also by the COLUMN-SIZE and OUTPUT-FORMAT operands.
If the records to be edited are variable in format, the record length field is also part of the data to be output.
In addition, a leader is inserted at the start of each line; its length depends on the selected output medium and the selected function, as shown in the following table.

Function	Output file	
	SYSLST, cataloged file	**SYSOUT**
File copying	No leader	9 characters (record position) (BYTCNT)
Tape editing	21 characters (tape mark counter) (TMC) (block counter) (BLKCNT) (record position) (BYTCNT)	9 characters (record position) (BYTCNT)

COLUMN-SIZE = *LINE-SIZE / <integer 1..204>
Defines the number of characters of the output record which are to belong to one group and are to be separated from the next group by means of blanks.

COLUMN-SIZE = *LINE-SIZE
There is no subdivision into groups.

SUPPRESS-EQUAL-LINES =
Influences the printing of lines of a record with identical contents. The last line of an edited record is always printed out, irrespective of the assignment of the SUPPRESS-EQUAL-LINES parameter.

SUPPRESS-EQUAL-LINES = *YES
If two or more identical lines occur consecutively, the repeat line is printed only once.
The message

```
'xxxxxx IDENTICAL LINES SUPPRESSED',
```

appears in the next line, where “xxxxxx” is the number of lines with identical contents.

SUPPRESS-EQUAL-LINES = *NO
All lines are output, even duplicated ones.

REPLACE-CONTROL-CHAR =
Determines how printer control characters are to be handled.

REPLACE-CONTROL-CHAR = <u>*YES</u>
Any control characters X´00´ thru X´3F´ in the output records are replaced by blanks if either the XHCS is not active, or XHCS is active but the output file has no CCSN.
If XHCS is active and the output file has a CCSN, the XHCS table of printable characters for this CCS is used for the replacement.

REPLACE-CONTROL-CHAR = *NO
The output records contain characters that are <u>not</u> to be replaced.
No check is performed as to whether there are any nonprintable characters. No character replacement is performed.

USER-LINE =
Outputs keywords and literals at specific positions on the page. The header line and line range defined by the OUTPUT-AREA operand may be overwritten.

USER-LINE = <u>*NONE</u>
No user lines are output.

USER-LINE = *RECORD-LENGTH(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file or of the input tape whose current record length is to be used in the user line.
This is represented as an 8-digit number.

OUTPUT-LINE = <integer 1..112>
Specifies the number of the line to which the record length is written.

OUTPUT-COLUMN = <integer 1..197>
Specifies the position where the output file begins, relative to the beginning of the user line.

OUTPUT-LENGTH = <u>*STD</u> / <integer 1..32>
Length of the output field. The default value *STD is 8.

USER-LINE = *BLOCK-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input tape whose current block counter value is to be used in the user line. This is represented as an 8-digit decimal number.

i If the contents of this keyword are longer than the output field, the digits on the extreme left will be truncated without any warning.

OUTPUT-LINE = <integer 1..112>
Specifies the number of the line to which the value of the block counter is written.

OUTPUT-COLUMN = <integer 1..197>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..32>
Length of the output field. The default value *STD is 8.

USER-LINE = *BYTE-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file or of the input tape whose current byte counter value is to be used in the user line.This is represented as a 10-digit decimal number.

i If the contents of this keyword are longer than the output field, the digits on the extreme left will be truncated without any warning.

OUTPUT-LINE = <integer 1..112>
Specifies the number of the line to which the value of the byte counter is written.

OUTPUT-COLUMN = <integer 1..195>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..32>
Length of the output field. The default value *STD is 10.

USER-LINE = *RECORD-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file whose current record value is to be written in the user line.
This is represented as an 8-digit decimal number.

i If the contents of this keyword are longer than the output field, the digits on the extreme left will be truncated without any warning.

OUTPUT-LINE = <integer 1..112>
Specifies the number of the line to which the value of the record counter is written.

OUTPUT-COLUMN = <integer 1..197>
Specifies the position where the output field begins, relative to the beginning of the user line.

OUTPUT-LENGTH = *STD / <integer 1..32>
Length of the output field. The default value *STD is 8.

USER-LINE = *PAGE-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the output file whose current page counter value is to be used in the user line.
This is represented as a 6-digit decimal number.

OUTPUT-LINE = <integer 1..112>
Specifies the number of the line to which the page number is written.

OUTPUT-COLUMN = <integer 1..199>
Specifies the number of the line to which the page number is written.

OUTPUT-LENGTH = *STD / <integer 1..32>
Length of the output field. The default value *STD is 6.

USER-LINE = *DATE(...)
Specifies the start date of the PERCON conversion step as an 8-digit string in the form yy-mm-dd.

OUTPUT-LINE = <integer 1..112>
Specifies the number of the line to which the date is written.

OUTPUT-COLUMN = <integer 1..197>
Specifies the position where the output field begins, relative to the beginning of the user line.

CENTURY = <u>*NO</u> / *YES
If CENTURY=*YES is specified, the date is output as a 10-digit character string with the century specified in the format yyyy-mm-dd. This reduces the maximum value for OUTPUT-COLUMN by 2.

USER-LINE = *TIME(...)
Specifies the start time of the PERCON conversion step as an 8-digit string in the form hh:mm:ss.

OUTPUT-LINE = <integer 1..112>
Specifies the number of the line to which the time is written.

OUTPUT-COLUMN = <integer 1..197>
Specifies the position where the output field begins, relative to the beginning of the user line.

USER-LINE =<c-string 1..204 with-low>

OUTPUT-LINE = <integer 1..112>
Specifies the number of the line to which the character constant is written.

OUTPUT-COLUMN = <integer 1..204>
Specifies the position where the output field begins, relative to the beginning of the user line.

USER-LINE =<x-string 1..408>(...)

OUTPUT-LINE = <integer 1..112>
Specifies the number of the line to which the character constant is written.

OUTPUT-COLUMN = <integer 1..204>
Specifies the position where the output field begins, relative to the beginning of the user line.

SET-RECORD-MAPPING

This statement is only permissible in conjunction with file copying. It can be used to format the output record relative to the input record. The output record can be made up of input record fields, literals and keywords.

Overwriting the output record

An existing output record can be overwritten either with the input record or with a record made up of the input record data, literals and keywords. With the second option, the output area is prefilled with either the input record or a fill character.

Updating the output record

It is possible to update individual fields in an existing output record (keyword *OUTPUT of the FILLER operand). The output area is prefilled with an output file record in this case. The following limiting conditions have to be observed:

1. The input record must be unequivocally associated with the corresponding output record. Since no sorting conditions apply to the input file, random access to the output file must be possible. This is why only existing ISAM files may be used as output files. Output records are accessed via a key, which may be a secondary key. The output file is opened in INOUT mode. If the output file is not an ISAM file, the conversion step is aborted and message PER0053 is output.
2. There are no restrictions as to the file type of the input file. However, each input record must have a field at identical position containing the key that is used to establish the corresponding output record.
3. The key field in the input file must have the same length as the key in the output file.
4. The minimum length of each input record must be sufficient to accommodate the complete key field. If the key field is longer than the input record, the conversion step is aborted and message PER0044 is output.
5. If the input file is an ISAM file, the key field used to determine the appropriate output record need not be an ISAM-key field.
6. If the output file contains more than one record with identical value in the referenced key, the first of these records is used for prefilling the output area.
7. The record length of the new output record may be modified. Note, however, that when writing back a record whose length has been increased, the position of the record within a sequence of records with identical key may change (ISAM feature). It can therefore not be excluded that a different record will be processed by a subsequent access using this key.

8. The operand values specified in the SET-RECORD-MAPPING statement must prevent all modification of the key field of the primary key in the output record (ISAM requirement). If this requirement is not met, the conversion step is aborted an message PER0085 is output.
9. Warning PER0054 is output if an output file key cannot be found; the message contains an insert that indicates the key. Processing is continued with the next input record. A maximum of ten occurrences of this message is possible per link name.
10. If the length of the record formatted in accordance with the SET-RECORD-MAPPING-statement exceeds the length of the record used for prefilling the output area, any areas not occupied by either the prefilling record or described by the OUTPUT-FIELDS operand will be filled with blanks.

The following steps are normally executed when processing an input record:

1. An input record is read and the key for the corresponding output record is determined.
2. The corresponding output record is read using the key, and the output area is prefilled with this record.
3. The complete output record is formatted in accordance with the specifications in the SET-RECORD-MAPPING statement.
4. The output record is written back to the output file.

Defining the minimum record length

When formatting an output record, its length is determined by the highest record position specified in the OUTPUT-FIELDS operand of the SET-RECORD- MAPPING statement unless a fixed record length has been specified. The MIN-RECORD-LENGTH operand can be used to specify a minimum length for the data; this can either be a fixed length or it can be made dependent on the length of the corresponding input record. If the output area is prefilled with the input record or the existing output record and the length of the formatted record is less than the defined minimum length, the record is either padded with the filler or with blanks up to the minimum length.

The SET-RECORD-MAPPING statement is optional. If it is omitted, the input record is transferred as the output record, without any changes.

Note

Any conversion to an output CCS is not regarded as being a change in this sense.

If several SET-RECORD-MAPPING statements in a conversion step refer to the same link name, only the last statement specified is executed for this link name.

SET-RECORD-**MAP**PING

OUTPUT-**LINK-NAME** = ***STD** / list-poss(2000): <filename 1..8 without-gen>

,**FILL**ER = **C ' '** / <c-string 1..1 with-low> / <x-string 1..2> / ***INPUT** / ***OUTPUT**(...)

***OUTPUT**(...)

| **KEY-NAME** = ***PRIM**ARY / <name 1..8>

| ,**KEY-VALUE** = ***BY-INPUT-REC**ORD (...)

| ***BY-INPUT-REC**ORD(...)

| | **KEY-POS**ITION = ***STD** / <integer 1..32768>

,**OUT**PUT-**FIELDS** = ***COMPL**ETE-**REC**ORD / list-poss(2000): ***FIELD**(...) / ***GR**OUP-**COUNT**ER(...) / ***REC**ORD-**LEN**GTH(...) / ***BYTE-COUNT**ER(...) / ***REC**ORD-**COUNT**ER(...) / ***PAGE-COUNT**ER(...) / ***DATE**(...) / ***TIME**(...) / <c-string 1..256 with-low>(...) / <x-string 1..512>(...) / <integer -2147483648..2147483647>(...)

***FIELD**(...)

| **INPUT-POS**ITION = <integer 1..32768>

| ,**INPUT-LEN**GTH = **1** / <integer 1..32767> / ***REC**ORD-**LEN**GTH(...)

| ***REC**ORD-**LEN**GTH(...)

| | **RED**UCTION = **0** / <integer 0..32767>

| ,**IN**PUT-**FORM**AT = ***CHAR**ACTER / ***ZONED-DECIMAL** / ***PACKED-DECIMAL**

| ,**OUT**PUT-**POS**ITION = <integer 1..32768>

| ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..32767> / ***REC**ORD-**LEN**GTH(...)

| ***REC**ORD-**LEN**GTH(...)

| | **RED**UCTION = **0** / <integer 0..32767>

| ,**OUT**PUT-FORMAT = ***INP**UT-**FORM**AT / ***CHAR**ACTER / ***HEX**ADECIMAL / ***BIN**ARY / ***TRANS**LATION / ***NO-TRANS**LATION / ***ZONED-DECIMAL** / ***PACKED-DECIMAL** / <c-string 2..256 with-low> / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL / ***UNI**CODE-**TRANS**LATION

continued ➠

***GR**OUP-**COUNT**ER(...)

| **LINK**-NAME = <filename 1..8 without-gen>
| ,**GR**OUP-**LEV**EL = <integer 1..8>
| ,**OUT**PUT-**POS**ITION = <integer 1..32767>
| ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..32767>
| ,**OUT**PUT-FORMAT = **C ' ZZZ'** / <c-string 2..256 with-low> / ***ZON**ED-**DEC**IMAL-**LEFT**

***REC**ORD-**LEN**GTH(...)

| **LINK**-NAME = <filename 1..8 without-gen>
| ,**OUT**PUT-**POS**ITION = <integer 1..32767>
| ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..32767>
| ,**OUT**PUT-FORMAT = ***ZON**ED-**DEC**IMAL-**LEFT** / <c-string 2..256 with-low> / ***PACKED-DECIMAL** /
| ***ZONED-DECIMAL** / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

***BYTE-COUNT**ER(...)

| **LINK**-NAME = <filename 1..8 without-gen>
| ,**OUT**PUT-**POS**ITION = <integer 1..32767>
| ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..32767>
| ,**OUT**PUT-FORMAT = ***ZON**ED-**DEC**IMAL-**LEFT** / <c-string 2..256 with-low> / ***PACKED-DECIMAL** /
| ***ZONED-DECIMAL** / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

***REC**ORD-**COUNT**ER(...)

| **LINK**-NAME = <filename 1..8 without-gen>
| ,**OUT**PUT-**POS**ITION = <integer 1..32767>
| ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..32767>
| ,**OUT**PUT-FORMAT = ***ZON**ED-**DEC**IMAL-**LEFT** / <c-string 2..256 with-low> / ***PACKED-DECIMAL** /
| ***ZONED-DECIMAL** / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

***PAGE-COUNT**ER(...)

| **LINK**-NAME = <filename 1..8 without-gen>
| ,**OUT**PUT-**POS**ITION = <integer 1..32767>
| ,**OUT**PUT-**LEN**GTH = ***STD** / <integer 1..32767>
| ,**OUT**PUT-FORMAT = ***ZON**ED-**DEC**IMAL-**LEFT** / <c-string 2..256 with-low> / ***PACKED-DECIMAL** /
| ***ZONED-DECIMAL** / ***SIGN**ED-**DEC**IMAL / ***DEC**IMAL

continued ➠

```
   *DATE(...)
     |  OUTPUT-POSITION = <integer 1..32761>
     | ,OUTPUT-LENGTH = *STD / <integer 8..32767>
     | ,OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
     | ,CENTURY = *NO / *YES
   *TIME(...)
     |  OUTPUT-POSITION = <integer 1..32761>
     | ,OUTPUT-LENGTH = *STD / <integer 8..32767>
     | ,OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
   <c-string>(...)
     |  OUTPUT-POSITION = <integer 1..32768>
     | ,OUTPUT-LENGTH = *STD / <integer 1..32767>
     | ,OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
   <x-string>(...)
     |  OUTPUT-POSITION = <integer 1..32768>
     | ,OUTPUT-LENGTH = *STD / <integer 1..32767>
     | ,OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
   <integer>(...)
     |  OUTPUT-POSITION = <integer 1..32768>
     | ,OUTPUT-LENGTH = *STD / <integer 1..32767>
     | ,OUTPUT-FORMAT = *ZONED-DECIMAL / <c-string 2..256 with-low> / *PACKED-DECIMAL /
     |        *SIGNED-DECIMAL / *DECIMAL
,CODE-TRANSLATION = *NONE / *TABLE(...) / list-poss(2000): *REPLACE-CHARACTER(...)
   *TABLE(...)
     |  FILE-NAME = <filename 1..54>
   *REPLACE-CHARACTER(...)
     |  INPUT-CHARACTER = <c-string 1..1 with-low> / <x-string 1..2>
     | ,OUTPUT-CHARACTER = <c-string 1..1 with-low> / <x-string 1..2>
,MIN-RECORD-LENGTH = *NONE / *BY-INPUT-RECORD(...) / <integer 1..32768>
   *BY-INPUT-RECORD(...)
     |  ADDITION = 0 / <integer -32767..32767>
```

OUTPUT-LINK-NAME =
Specifies link names of the output files to which this statement is to refer.

OUTPUT-LINK-NAME = *STD
This statement refers to all output files specified up to now.

OUTPUT-LINK-NAME = list-poss(2000): <filename 1..8 without-gen>
If the statement is to refer only to certain output files, the link names of these files must be specified.

FILLER =
Specifies a filler character for the output record areas not defined via the OUTPUT-FIELDS operand.

FILLER = C' '
The filler character is a blank.

FILLER = <c-string 1..1 with-low>
The filler character is in character format.

FILLER = <x-string 1..2>
The filler character is in hexadecimal format.

FILLER = *INPUT
The output record already contains the input record. If the output record is longer than the input record, the remainder of the output record is filled with blanks.

FILLER = *OUTPUT (...)
An output record is to be used to prefill the output area.

KEY-NAME =
Specifies the output file key to be used to determine the output record.

KEY-NAME = *PRIMARY
The primary key is to be used.

KEY-NAME = <name 1..8>
The name of the secondary key that is to be used is specified.
If the specified secondary key does not exist, the conversion step is aborted and a message is output.

KEY-VALUE = *BY-INPUT-RECORD (...)
The key used to access the output file is contained in the input record. The length of the key is assumed to be identical with the length of the output file key specified with KEY-NAME.

KEY-POSITION = *STD
If the format of the input records is variable, KEY-POSITION=5 applies, otherwise KEY-POSITION=1.

KEY-POSITION = <integer 1.. 32768>
Specifies the start position of the key field in the input record.

If the FILLER=*OUTPUT operand of the SET-RECORD-MAPPING statement is specified for an output file, the statements SET-GROUP-ATTRIBUTES and SET-PAGE-LAYOUT are not permitted for this output file. If this restriction is not observed, the user is informed of the violation by means of message PER0056.

If the OVERWRITE=*NO operand is specified in the ASSIGN-OUTPUT-FILE statement and FILLER=*OUTPUT is defined for an output file, PERCON recognizes the incompatibility and rejects it with message PER0056.

Specifying FILLER=*OUTPUT in the SET-RECORD-MAPPING statement is incompatible with specifying OUTPUT-FIELDS=*COMPLETE-RECORD; this will be rejected with message PER0087.

OUTPUT-FIELDS =
Constructs the output record from parts of the input record, keywords and literals. In the event of different length and format specifications for the input and output fields, PERCON makes appropriate conversions.

When RECORD-FORMAT = *VARIABLE is specified for output files:

– In the SET-RECORD-MAPPING statement, the first 4 characters of the output record must not be addressed during data transfer, since they occupy the record length field (values are supplied by PERCON).
– The output record length is determined by the last field in the output record that is specified in the OUTPUT-FIELDS operand or by the MIN-RECORD-LENGTH operand. If the OUTPUT-FIELDS operand is omitted, either the length of the input record or the specification for MIN-RECORD-LENGTH is assumed.

OUTPUT-FIELDS = *COMPLETE-RECORD
The input record is transferred to the output file. If the FILLER operand is not equal to INPUT, all characters of the input record are replaced by fillers.

OUTPUT-FIELDS = *FIELD(...)
Transfers a field in the input record (input field) to a field of the output record (output field). The specified record field must be within the input record.

INPUT-POSITION = <integer 1..32768>
Specifies the position where the input field begins, relative to the beginning of the input record.

INPUT-LENGTH = 1 / <integer 1..32767> / *RECORD-LENGTH(...)
Specifies the length of the input field.

INPUT-LENGTH = *RECORD-LENGTH(...)
Defines the record length of the current input record (for variable-format records, without the record length field). This operand can only be specified in conjunction with INPUT-FORMAT = *CHARACTER.

REDUCTION = 0 / <integer 0..32767>
Specifies by how many characters the current input record length is to be shortened.

INPUT-FORMAT = *CHARACTER / *ZONED-DECIMAL / *PACKED-DECIMAL
Specifies the format of the input field (see page 80ff). The input record field which is to be edited by means of a mask must not be longer than 255 bytes when INPUT-FORMAT=*ZONED-DECIMAL and no longer than 128 bytes when INPUT-FORMAT=*PACKED-DECIMAL.

OUTPUT-POSITION = <integer 1..32768>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 1..32767> / *RECORD-LENGTH(...)
Specifies the length of the output field.

OUTPUT-LENGTH = *STD
INPUT-LENGTH is assumed by default.
The following combinations represent exceptions:

	INPUT-FORMAT		
OUTPUT-FORMAT	**CHARACTER INPUT-LENGTH ≠ *RECORD-LENGTH**	**ZONED-DECIMAL**	**PACKED-DECIMAL**
HEXADECIMAL	2 * INPUT-LENGTH		
BINARY	8 * INPUT-LENGTH		
SIGNED-DECIMAL		INPUT-LENGTH + 1	2 * INPUT-LENGTH
DECIMAL		INPUT-LENGTH + 1	2* INPUT-LENGTH
ZONED-DECIMAL		INPUT-LENGTH	2 * INPUT-LENGTH -1
PACKED-DECIMAL		INPUT-LENGTH/2+1	
<c-string> (Mask)		Mask length	Mask length

If the input file and/or the output file is assigned a Unicode format and the OUTPUT-FORMAT operand has the value *UNICODE-TRANSLATION, the default value for the output length is determined as follows when conversion and/or normalization takes place.

Input format	**Output format**	**Output length**
Not Unicode	Unicode is UTF-16	INPUT-LENGTH*2
Not Unicode	Unicode is not UTF-16	INPUT-LENGTH*3
Unicode is UTF-16	Unicode is UTF-16	INPUT-LENGTH*1
Unicode is UTF-16	Unicode is not UTF-16	INPUT-LENGTH*2
Unicode is UTF-16	Not Unicode	INPUT-LENGTH:2 (rounded up)
Unicode is not UTF-16	Unicode is UTF-16	INPUT-LENGTH*2
Unicode is not UTF-16	Unicode is not UTF-16 (CCSN not identical)	INPUT-LENGTH*2
Unicode is not UTF-16	Unicode is not UTF-16 (CCSN identical)	INPUT-LENGTH*1
Unicode is not UTF-16	Not Unicode	INPUT-LENGTH*1

Note

When output takes place in Unicode format UTF-16, the maximum value for the length of the output field is 32766. When output takes place in a Unicode format other than UTF-16, the maximum value for the output field is 32767.

OUTPUT-LENGTH = *RECORD-LENGTH(...)
Defines the record length. This operand can only be specified in conjunction with INPUT-FORMAT = *CHARACTER.

REDUCTION = 0 / <integer 0..32767>
Specifies by how many characters the current input record length is to be shortened.

OUTPUT-FORMAT = *INPUT-FORMAT / *CHARACTER / *HEXADECIMAL / *BINARY /*TRANSLATION / *NO-TRANSLATION / *ZONED-DECIMAL / *PACKED-DECIMAL /<c-string 2..256 with-low> / *SIGNED-DECIMAL / *DECIMAL / *UNICODE-TRANSLATION

*NO-TRANSLATION must be used when the input field is not to be converted to the output character set (e.g. with packed numbers). Specifies the format of the rest of the output field (see page 80ff).

Only the following combinations are permissible:

OUTPUT-FORMAT	**INPUT-FORMAT**		
	CHARACTER	**ZONED-DECIMAL**	**PACKED-DECIMAL**
INPUT-FORMAT	+	+	+
CHARACTER	+		
HEXADECIMAL	+		
BINARY	+		
TRANSLATION	+		
NO-TRANSLATION	+	+	+
ZONED-DECIMAL	+	+	+
PACKED-DECIMAL		+	+
<c-string>		+	+
SIGNED-DECIMAL		+	+
UNICODE-TRANSLATION	+		

Decimal conversion of field contents

By combining the operands INPUT-FORMAT=*CHARACTER and OUTPUT-FORMAT=*ZONED-DECIMAL, a binary field value can be converted to a decimal value. The following restrictions apply to the conversion of binary input field values to decimal output field records:

1. The maximum length of the input field is four characters (INPUT-LENGTH operand). Specification of a larger value is regarded as a syntactical error and rejected with message PER0005.
2. The input field is assumed to be unsigned.
3. The length of the output field (OUTPUT-LENGTH operand) must be sufficient to accommodate all valid positions. Otherwise the conversion step will be aborted and message PER0042 is output.
4. If OUTPUT-LENGTH=*STD is specified, a value of 10 is assumed.
5. If INPUT-LENGTH=*RECORD-LENGTH is specified, observance of the maximum input length of four bytes cannot be checked until the record is processed. Any violation of the restriction results in the conversion step being aborted and message PER0042 being output.

Example

A file contains a field with the length of 2 and start position 12. The binary value of this field is to be converted to decimal format and written to the output record with a length of 6 starting at position 8.

Input field contents: X'02C3'

Operands of the SET-RECORD-MAPPING statement:

```
OUTPUT-FIELDS=FIELD( -
            INPUT-POSITION =12,INPUT-LENGTH =2, -
            OUTPUT-POSITION= 8,OUTPUT-LENGTH=6, -
            INPUT-FORMAT =*CHARACTER, -
            OUTPUT-FORMAT=*ZONED-DECIMAL)
```

Contents of the output field starting at position 8: `000707`

OUTPUT-FIELDS = *GROUP-COUNTER(...)
Specifies counters per group. GROUP-COUNTER is available for group levels 1 to 8 of each output medium. Each counter is unambiguously defined by its link name and by a group level.

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file from which a group counter is to be used in the user line.

GROUP-LEVEL = <integer 1..8>
Specifies which group counter of the input/output file labelled LINK-NAME is to be used.

OUTPUT-POSITION = <integer 1..32767>
Specifies the start position of the output field, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 1..32767>
Specifies the length of the output field. The standard (default) value is either 4 when OUTPUT-FORMAT = *ZONED-DECIMAL or the mask length.

OUTPUT-FORMAT = C' ZZZ' / <c-string 2..256 with-low> / *ZONED-DECIMAL-LEFT
Specifies the format of the output field (see page 80ff).

OUTPUT-FIELDS = *RECORD-LENGTH(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file whose current record length is to be used in the output record.

OUTPUT-POSITION = <integer 1..32767>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 1..32767>
Specifies the length of the output field.

OUTPUT-LENGTH = *STD

OUTPUT-FORMAT	**OUTPUT-LENGTH**
ZONED-DECIMAL-LEFT	8 characters
ZONED-DECIMAL	8 characters
SIGNED-DECIMAL	9 characters
DECIMAL	9 characters
PACKED-DECIMA	5 characters
<c-string> (mask)	Mask length

OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..256 with-low> / *PACKED-DECIMAL / *ZONED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

OUTPUT-FIELDS = *BYTE-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file whose current byte counter value is to be used in the output record.

OUTPUT-POSITION = <integer 1..32767>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 1..32767>
Specifies the length of the output field.

OUTPUT-LENGTH = *STD

OUTPUT-FORMAT	**OUTPUT-LENGTH**
ZONED-DECIMAL-LEFT	10 characters
ZONED-DECIMAL	10 characters
SIGNED-DECIMAL	11 characters
DECIMAL	11 characters
PACKED-DECIMAL	6 characters
<c-string> (mask)	Mask length

OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..256 with-low> / *PACKED-DECIMAL / *ZONED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

OUTPUT-FIELDS = *RECORD-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the input/output file whose current record counter length is to be used in the output record.

OUTPUT-POSITION = <integer 1..32767>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 1..32767>
Specifies the length of the output field.

OUTPUT-LENGTH = *STD

OUTPUT-FORMAT	**OUTPUT-LENGTH**
ZONED-DECIMAL-LEFT	8 characters
ZONED-DECIMAL	8 characters
SIGNED-DECIMAL	9 characters
DECIMAL	9 characters
PACKED-DECIMAL	5 characters
<c-string> (mask)	Mask length

OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..256 with-low> / *PACKED-DECIMAL / *ZONED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

OUTPUT-FIELDS = *PAGE-COUNTER(...)

LINK-NAME = <filename 1..8 without-gen>
Specifies the link name of the output file whose current record counter length is to be used in the output record.

OUTPUT-POSITION = <integer 1..32767>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 1..32767>
Specifies the length of the output field.

OUTPUT-LENGTH = *STD

OUTPUT-FORMAT	OUTPUT-LENGTH
ZONED-DECIMAL-LEFT	6 characters
ZONED-DECIMAL	6 characters
SIGNED-DECIMAL	7 characters
DECIMAL	7 characters
PACKED-DECIMAL	4 characters
<c-string> (mask)	Mask length

OUTPUT-FORMAT = *ZONED-DECIMAL-LEFT / <c-string 2..256 with-low> / *PACKED-DECIMAL / *ZONED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

OUTPUT-FIELDS = *DATE(...)
Specifies the start date of the PERCON conversion step as an 8-digit string in the form yy-mm-dd.

OUTPUT-POSITION = <integer 1..32761>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 8..32767>
Specifies the length of the output field.

OUTPUT-LENGTH = _*STD_

	CENTURY = *NO	CENTURY = *YES
OUTPUT-FORMAT	**OUTPUT-LENGTH**	**OUTPUT-LENGTH**
CHARACTER	8 characters	10 characters
HEXADECIMAL	16 characters	20 characters
BINARY	64 characters	80 characters

OUTPUT-FORMAT = _*CHARACTER_ / *HEXADECIMAL / *BINARY
Specifies the format of the output field (see page 80ff).

CENTURY = _*NO_ / *YES
If CENTURY=*YES is specified, the date is output as a 10-digit character string, with the century specification in the format yyyy-mm-dd. This reduces the maximum value for OUTPUT-POSITION by 2 to 32759 and the minimum value for OUTPUT-LENGTH is increased by 2 to 10.

OUTPUT-FIELDS = *TIME(...)
Specifies the start time of the PERCON conversion step as an 8-digit string in the form hh:mm:ss.

OUTPUT-POSITION = <integer 1..32761>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = _*STD_ / <integer 8..32767>
Specifies the length of the output field.

OUTPUT-LENGTH = _*STD_

OUTPUT-FORMAT	**OUTPUT-LENGTH**
CHARACTER	8 characters
HEXADECIMAL	16 characters
BINARY	64 characters

OUTPUT-FORMAT = _*CHARACTER_ / *HEXADECIMAL / *BINARY
Specifies the format of the output field (see page 80ff).

OUTPUT-FIELDS =<c-string 1..256 with-low>(...)

OUTPUT-POSITION = <integer 1..32768>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 1..32767>
Specifies the length of the output field.

OUTPUT-LENGTH = *STD

OUTPUT-FORMAT	**OUTPUT-LENGTH**
CHARACTER	Length of the c-string
HEXADECIMAL	2 * Length of the c-string
BINARY	8 * Length of the c-string

OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
Specifies the format of the output field (see page 80ff).

OUTPUT-FIELDS =<x-string 1..512>(...)

OUTPUT-POSITION = <integer 1..32768>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 1..32767>
Specifies the length of the output field.

OUTPUT-LENGTH = *STD

OUTPUT-FORMAT	**OUTPUT-LENGTH**
CHARACTER	0,5 * Length of the x-string
HEXADECIMAL	Length of the x-string
BINARY	4 * Length of the x-string

OUTPUT-FORMAT = *CHARACTER / *HEXADECIMAL / *BINARY
Specifies the format of the output field (see page 80ff).

OUTPUT-FIELDS =<integer -2147483648..2147483647>(...)

OUTPUT-POSITION = <integer 1..32768>
Specifies the position where the output field begins, relative to the beginning of the output record.

OUTPUT-LENGTH = *STD / <integer 1..32767>
Specifies the length of the output field.

OUTPUT-LENGTH = *STD
The default value is the minimum length required to represent the number to be output in the specified format; the mask length with <c-string>.

OUTPUT-FORMAT = <u>*ZONED-DECIMAL</u> / <c-string 2..256 with-low> / *PACKED-DECIMAL / *SIGNED-DECIMAL / *DECIMAL
Specifies the format of the output field (see page 80ff).

CODE-TRANSLATION =
Mandatory if the structure operand OUTPUT-FIELDS = *FIELD was used in the OUTPUT-FORMAT = *TRANSLATION operand.

CODE-TRANSLATION = <u>*NONE</u>
No code conversion takes places, even if OUTPUT-FORMAT = *TRANSLATION has been specified.

CODE-TRANSLATION = *TABLE(...)
Specifies the file containing the code conversion table.

FILE-NAME = <filename 1..54>
This file contains 16 records, each with the hexadecimal encryption of 16 characters. The corresponding characters of the input field are converted into these characters and transferred to the output field. Assignment is via the hexadecimal value contained in the code conversion table. The file can be either a SAM or ISAM file. In the case of ISAM files, the ISAM record key must be at the beginning of the record and must have a length of 8 bytes (position 1 for fixed-length records, position 5 for variable-length records).

CODE-TRANSLATION = *REPLACE-CHARACTER(...)
Replaces character by character.

INPUT-CHARACTER = <c-string 1..1> / <x-string 1..2>
Character to be recoded.

OUTPUT-CHARACTER = <c-string 1..1> / <x-string 1..2>
Replacement character for the character to be recoded.

MIN-RECORD-LENGTH =
Defines the minimum length for the output record.
With output files with variable record format, PERCON additionally evaluates the record length field.

MIN-RECORD-LENGTH = <u>*NONE</u>
No minimum output record length is defined.

MIN-RECORD-LENGTH = <integer 1..32768>
A fixed minimum output record length is defined.

MIN-RECORD-LENGTH = *BY-INPUT-RECORD(...)
The record length of the current input record is used, without record length field if the input records have a variable record format.

ADDITION = <integer -32767..32767>
Specifies the number of characters by which the current input record length is to be modified.

Notes

– If the specified minimum length exceeds the permissible maximum length of an output record, the maximum value is taken as the minimum length. No minimum length is observed for a particular record in the event of a negative or zero result.

– The MIN-RECORD-LENGTH operand has no meaning for fixed-length output records. It will be evaluated during the syntax check but ignored during processing.

– When editing records with the SET-PAGE-LAYOUT statement or during default editing, the specification of a minimum length causes any blanks preceding the end of the record to be retained. This means that records will not be truncated during editing in this case.

START-CONVERSION

When copying files and duplicating tapes, the START-CONVERSION statement is used to start the transfer operation and to terminate the conversion step.

When editing tapes, additional specifications must be made with the aid of the START-TAPE-PROCESSING statement to start transfer operations; the START-CONVERSION statement in this mode is only used to terminate the conversion step.

In contrast to the END statement, this does not terminate the PERCON run. A further conversion step can be performed immediately.

START-CONVERSION

The START-CONVERSION statement does not have any operands.

START-TAPE-PROCESSING

This statement is only permissible in conjunction with tape editing for transfer from a tape to a file. It controls the extent of output and starts the transfer operation.

The number of blocks processed is not reported until the START-CONVERSION or END statement has been entered.

Either SYSLST or SYSOUT (see the ASSIGN-OUTPUT-FILE statement, page 105, FILE=*SYSLST or FILE=*SYSOUT operand) or a cataloged file must be assigned as the output medium.

Output can be to one of the above-mentioned output media or simultaneously to more than one output medium, or even different ones.

```
START-TAPE-PROCESSING

 OUTPUT-LINK-NAME = *STD / list-poss(2000): <filename 1..8 without-gen>
,INPUT-RANGE = *NONE / *BLOCKS(...) / *TAPE-MARKS(...)
    *BLOCKS(...)
      |   BLOCKS = <integer 1..2147483647>
    *TAPE-MARKS(...)
      |   TAPE-MARKS = <integer 1..2147483647>
,TERMINATION = *END-OF-TAPE / *DOUBLE-TAPE-MARK
,END-POSITION = *LEAVE-POSITION / *START-POSITION
```

OUTPUT-LINK-NAME =
Link names of the output files to which this statement is to refer.

OUTPUT-LINK-NAME = *STD
This statement refers to all output files specified up to now with the exception of the “residual files”.

OUTPUT-LINK-NAME = list-poss: <filename 1..8 without-gen>
If the statement is to refer only to certain output files, the link names of these files must be specified. Specification of the link name of a residual file is not permissible and will be rejected as an error (message PER0062).

INPUT-RANGE =
Reads the input tape, starting at the current position. The blocks which have been read are transferred to the assigned output medium.

INPUT-RANGE = *NONE
Either the operand value specified for TERMINATION or the appropriate default value applies.

INPUT-RANGE = *BLOCKS(...)

BLOCKS = <integer 1..2147483647>
Specifies the number of blocks to be processed.

INPUT-RANGE = *TAPE-MARKS(...)

TAPE-MARKS = <integer 1..2147483647>
All blocks read up until the number specified by TAPE-MARKS is reached are processed.

TERMINATION =
This operand can be used to impose further restrictions on the processing of tapes.

TERMINATION = *END-OF-TAPE
Processing terminates no later than when the logical end of the tape is reached. Dummy files are recognized if a standard HDR3 label precedes the double tape mark.

TERMINATION = *DOUBLE-TAPE-MARK
Reading in is terminated at the latest when a double tape mark is reached.

END-POSITION =
Controls the positioning of the tape after processing.

END-POSITION = *LEAVE-POSITION
The input tape is not positioned after processing.

END-POSITION = *START-POSITION
The input tape is positioned after processing.

5 Calling PERCON as a subprogram

PERCON can be called as a subprogram. The following entry points are offered:

- PERCONU
  A small object module (PCROOT) is linked to the main program. This object module dynamically loads the prelinked module PCNSR9 from object module library $.SYSLNK.PERCON.029.
- PCNSR9 (=PCNSR7,PCNSR)
  The prelinked module is linked permanently to the calling program.

The entry point PCNSR9 should only be used if it is necessary to link in PERCON in its entirety.

The statements can be either read from SYSDTA, or transferred from the main program.

Note

PERCON operates on the basis of the SPL runtime system. Here a stack is created with the name PCNSRSTK. This name is entered into the linkage editor and loader structure and therefore cannot be used by main programs.

Assignment of object module library $.SYSLNK.PERCON.029

External references are resolved after calling the binder BINDER via INCLUDE-MODULES.

For further information needed for the linkage of a program see the “BINDER” manual [5].

5.1 UP call of PERCON

PERCON files can be stored under any user ID and be assigned different names. As a result, the expansion of the external reference PERCONU may give rise to problems when linking with BINDER or when using the dynamic binder-loader DBL (see “BLSSERV” manual [12]).

To enable the DBL to locate the PERCON starter module addressed via PERCONU, the name of the associated library must be made known, taking into consideration the object type of the main program (LLM). The object type must also be noted when calling the program.

For this purpose, it is recommended that you use an S procedure created using SDF-P, containing an S variable OBJTYP which defines the object type of the main program to be linked in.

The following command supplies the LIB-NAME variable with the name of the current PERCON library:

```
/SET-VARIABLE LIB-NAME = INSTALLATION-PATH -
/             (LOGICAL-ID        = 'SYSLNK' -
/             ,INSTALLATION-UNIT = 'PERCON'-
/             ,VERSION           = '*STD'-
/             ,DEFAULT-PATH-NAME = '$.SYSLNK.PERCON.029')
```

The following command uses the name determined in this way:

```
/IF " &OBJTYP = 'LLM' "
/ ADD-FILE-LINK FILE-NAME=&VAR,LINK-NAME=BLSLIB00
/ START-EXECUTABLE-PROGRAM ...
/END-IF
```

The example below shows part of a procedure in which the main program `<prog>` is started following successful compilation. The object created during the compilation run is stored in the library `<lib>` in the form of an LLM.

```
<compile main program>

/SET-VARIABLE OBJTYP = 'LLM'

/SET-VARIABLE LIB-NAME = '$.SYSLNK.PERCON.029'
/IF (SDF-P-VERSION >= 'V02.0A00')
/    LIB-NAME  = INSTALLATION-PATH                 -
/                (LOGICAL-ID        = 'SYSLNK'     -
/                ,INSTALLATION-UNIT = 'PERCON'     -
/                ,VERSION           = '*STD'       -
/                ,DEFAULT-PATH-NAME = '&LIB-NAME')
/END-IF

/IF " &OBJTYP = 'LLM' "
/ ADD-FILE-LINK FILE-NAME=&LIB-NAME,LINK-NAME=BLSLIB00
/ START-EXECUTABLE-PROGRAM -
/    FROM-FILE=*LIBRARY-ELEMENT(LIBRARY=<lib>,ELEMENT-OR-SYMBOL=<prog>), -
/    DBL-PARAMETERS=*PAR(RESOLUTION=*PAR(ALTERNATE-LIBRARIES=BLSLIB##))
/END-IF
```

S procedures that perform static linking using BINDER or TSOSLNK have to be created with a similar structure. This also applies where the prelinked PERCON module is to be linked to the main program via entry point PCNSR9.

If PERCONU is used, the PERCON starter module, the linking loader module PCROOT (LLM), is linked to the main program. The UP call does not necessarily cause this starter module to branch to the prelinked module of the PERCON version it belongs to. It has been designed to invoke the PERCON version which the user determines via entries in the parameter area at execution time (see “Format of the address list” on page 175),

For operating systems still running applications with starter modules PERCON V2.5A or lower, see page 190.

5.2 Register conventions

The following register conventions must be observed when calling PERCON as a subprogram:

Register 1:	address of the address list.
Register 13:	address of a save area comprising 18 words; this area must be made available by the main program. This area is used by PERCON as a save area, e.g. for the registers of the calling program.
Register 14:	return address for the main program.
Register 15:	address of the entry point PERCONU or PCNSR9.

Note

PERCON carries out its own STXIT call. As a result, any STXIT management block created in the main program (specified assignment “STXIT event class - STXIT routine”) is modified and supplemented. The STXIT routines defined in the main program are no longer activated (exception: PERCON does not change the RTIMER assignment).

5.3 Format of the address list

The address list indicated by the address in register 1 consists of at least 8 bytes.

If only the parameter area is to be transferred, the address list has a length of 8 bytes:

Byte	Length	Meaning
0 to 3	4	Address constant indicating the parameter area
4 to 7	4	X'80000000' as end criterion for the address list

If an area is provided for return information, the address list has a length of at least 12 bytes:

Byte	Length	Meaning
0 to 3	4	Address constant indicating the parameter area
4 to 7	4	Address constant indicating the area for the return information
8 to 1		X'80000000' as end criterion for the address list

If statements are submitted by the main program in memory and are to be processed with a coded character set name, the address list has a length of at least 16 bytes.

Byte	Length	Meaning
0 to 3	4	Address constant indicating the parameter area
4 to 7	4	Address constant indicating the area for the return information (or zero)
8 to 11	4	Address constant indicating the CCSN area
12 to 15		X'80000000' as end criterion for the address list

If a version area is to be transferred, the address list has a length of 20 bytes.

Byte	Length	Meaning
0 to 3	4	Address constant indicating the parameter area
4 to 7	4	Address constant indicating the area for the return information (or zero)
8 to 11	4	Address constant indicating the CCSN area
12 to 15	4	Address constant indicating the version area
16 to 20	4	X'80000000' as end criterion for the address list

Format of the parameter area

Specification of the parameter area is mandatory, since it contains the identifier for the SDF options.

Byte	Length	Meaning
0 to 3	4	Identifier for the type of statement transfer: X'00000080': The statements are read from SYSDTA. X'00000084': The statements are transferred from the main program.
4 to n	Unlimited	The statements are stored in this area in the form of variable-length records. The individual records must be aligned on a halfword boundary.

Format of the area for return information

Byte	Length	Meaning
0 to 3	4	In the event of an error, the DMS message code of the last DMS message issued; otherwise the contents remain unchanged.
4 to 15	12	In this area all the messages issued during program execution are registered by setting the appropriate bits. The bit nnnn within the 12 bytes, counted from the left, corresponds to the message PERnnnn, where nnnn is a number between 0 and 95. The area is not deleted.

Format of the CCSN area

Byte	Length	Meaning
0 to 7	8	Statement CCSN

If the statements in the memory are not assigned a coded character set name, they are processed without any consideration being taken of a CCS. All statements must belong to the same CCS.

Format of the version area

Byte	length	Meaning
0-3 0-6	4 7	*STD or version in format [m]m.n[a[so]], e.g. 02.9A00

The version can be entered in abbreviated form `[m]m.n`. In this case, any missing positions have to be replaced by blanks. If more than one version match the version specification, the correct version is selected according to the criteria listed on page 189. An incorrect version specification results in abnormal termination and output of an error message.

6 User interfaces

In the statements ASSIGN-INPUT-FILE, ASSIGN-OUTPUT-FILE, ASSIGN-INPUT-TAPE and SET-GROUP-ATTRIBUTES the user has the option of branching to user modules. Control of the PERCON run is thus transferred to the user for a limited period of time.

PERCON provides the following user interfaces:

– interface for label processing
  (statements: ASSIGN-INPUT-FILE, ASSIGN-OUTPUT-FILE)
– input interface (statement ASSIGN-INPUT-FILE)
– output interface (statement ASSIGN-OUTPUT-FILE)
– interface for group processing (statement SET-GROUP-ATTRIBUTES)
– interface for the recovery of read/length errors
  (statements: ASSIGN-INPUT-FILE, ASSIGN-INPUT-TAPE)
– interface for the recovery of open errors (statement ASSIGN-INPUT-FILE)

The connection between PERCON and the address of the user module is established by means of either a module name (1st CSECT name) or a name within a module (ENTRY name).

The user module must be contained in a library which has previously been assigned using the following command:

```
/SET-TASKLIB LIBRARY=<library>
```

or

```
/ADD-FILE-LINK FILE-NAME=<library>,LINK-NAME=BLSLIB00
```

6.1 Register conventions

If the control of PERCON is transferred to a user module, the contents of registers 1, 13, 14 and 15 have the following meanings:

Register 1:	Contains the address of a sequence of 4-byte address constants indicating parameter areas. The number of parameter areas varies depending on the interface. The end of the address list is indicated by a constant in the form X’80000000’.
Register 13:	Using this register PERCON transfers the address of an 18-word save area to the user module. In this area the PERCON registers are to be saved by means of the SAVE macro, which must be issued at the beginning of the user routine. By means of the RETRN macro (see the “Executive Macros” manual [8]) control can be returned to PERCON. With the help of the save area the RETRN macro can also reconstruct the contents of the PERCON registers. For this reason register 13 must not be overwritten by the user or must be reconstructed before RETRN is called.
Register 14:	Return address.
Register 15:	Branch address of the user routine.

6.2 Common parameter area for the interfaces

If PERCON branches to a user module, register 1 contains the address of a sequence of address constants which have a different meaning for each individual exit. The first entry of this address sequence, however, always indicates the beginning of a parameter area with the following structure, which is identical for all interfaces:

Byte	**Length**	**Meaning**
0 to 1	2	Length of the parameter area (including the length field). The parameter area is aligned on a word boundary.
3	1	Additional information on the type of interface.
4	1	Return code which is transferred to PERCON by the user module.
5	1	Reserved.
6 to 13	8	Link name of the current file.
14 to 21	8	CCSN of the transferred record/block.

The CCSN of the record/block is offered as a transfer parameter with all user interfaces.

Any data transfer at a user interface is then linked to the CCSN specified.

– Input user interface, read error user interface, length error user interface
  Record/block is offered in the input CCS.

– Output user interface, group break user interface
  Record is offered in the output CCS.

The validity of return codes supplied by user routines is checked. All return codes described in the manual are valid. If an invalid return code is detected, the conversion step is aborted and message PER0048 is output.

6.3 Description of the interfaces

The user interfaces described below have the following identifiers:

Interface for ...	Identifier
Label processing	X'00'
Input	X'04'
Output	X'08'
Group processing	X'0C'
Recovery of read/length errors	X'10'
Recovery of open errors	X'14'

6.3.1 Interface for label processing

If a user exit is specified in the LABEL-EXIT operand of the ASSIGN-INPUT-FILE or ASSIGN-OUTPUT-FILE statement, PERCON branches to the specified user module after reading or before writing the labels.

Register 1 indicates the following address list:

```
A(parameter area)
A(FCB)
X'800000000'
```

A description of the structure of the parameter area is provided on page 179.

The following values are entered in the parameter area:

Identifier of the interface: X'00'

Additional information	**Meaning**
X'00'	OPENV exit for UVL labels
X'04'	LABGN exit for UHL labels
X'08'	LABEOV exit for UTL labels with tape swaps
X'0C'	LABEND exit for UTL labels
X'10'	LABERR exit for errored end-of-tape labels
X'14'	CLOSPOS exit for positioning the tape during CLOSE processing

A description of the individual exits is provided in the “DMS Macros” manual [11] under the EXLST macro.

In the case of nonstandard labels, the user reads/writes his or her labels using the access method BTAM and returns control to PERCON without a return code.

In the case of standard labels, register 0 indicates the beginning of the user label; the user checks/generates the label himself and returns control to PERCON using the return code X’F1’ (if user label processing is to be terminated) or X’F2’ (if label processing by PERCON is to be continued).

For files with standard labels a branch is made to the LABERR exit if an error occurred during end-of-tape processing. The error code is contained in the ID1ECB field of the FCB.

The user can position the tape during CLOSE processing by means of the CLOSPOS exit. After positioning the tape using BTAM macros the user returns control to PERCON without a return code. (For a more detailed description of these exits, see the “DMS Macros” manual [11].)

Control can be returned to PERCON by means of the following return codes:

Return code	**Meaning**
X'00'	Execute volume swapping as if the correct EOV/EOF labels had been read.
X'01'	Terminate task with an error.
X'02'	Perform end-of-file processing.
X'F1'	Terminate user label processing.
X'F2'	Continue user label processing by PERCON.

6.3.2 Interface for input

If the operand value *MODULE(NAME=modulename) is specified in the INPUT-EXIT operand of the ASSIGN-INPUT-FILE statement, PERCON branches to the user module every time a record from this file has been read.

Register 1 indicates the following address list:

```
A(parameter area)
A(current input record)
A(user input record)
X'80000000'
```

A description of the structure of the parameter area is provided on page 179.

The following values are entered in the parameter area:

Identifier of the interface: X'04'

Additional information	Meaning
X'00'	The input record is transferred to the user module.
X'04'	End of the input file. No record is transferred; in this case only the return codes X'0C' thru X'18' are relevant. A return code X’00’ specified by the user is handled by PERCON in the same way as X’10’.

The user transfers information to PERCON via the return code of the parameter area.

Return code	**Meaning**
X'00'	Transfer record. PERCON transfers the record which has been made available. This record may have been changed by the user. The first 4 bytes (i.e. the record length field) of variable-length records must not be overwritten.
X'04'	Replace record. The record which is stored at the address of the user input record is transferred by PERCON as the input record. The address field is set by the user. In the case of variable-length records PERCON expects the record length field in the first 4 bytes. This return code is not permissible for OPEN=UPDATE. When RECORD-FORMAT=U, the length of the replaced record is expected to be that of the transferred record.
X'08'	Do not transfer record. Processing of this record is not continued. PERCON reads the next record in the input file.
X'0C'	Insert record. The user can insert a record in front of the record read by PERCON. The address of the record to be inserted must be the third entry in the address list. When RECORD-FORMAT=U, the length of the record to be inserted is expected to be that of the transferred record. Once the inserted record has been processed by PERCON, PERCON branches back to the user module and makes the input record read prior to the insertion available again. This return code is not permissible for OPEN=UPDATE.
X'10'	Do not branch back to user module. PERCON does not branch back to the user exit for this input file. The record made available is transferred in the same way as with return code X'00'.
X'14'	Terminate reading of input file. PERCON closes this input file and opens the next input file of this conversion step, if available. The record made available is no longer transferred.
X'18'	Terminate conversion step immediately with an error.

6.3.3 Interface for output

If the operand value *MODULE(NAME=modulename) is specified in the OUTPUT-EXIT operand of the ASSIGN-OUTPUT-FILE statement, PERCON branches directly (i.e. before the record is output) to the user module “modulename”.

Register 1 indicates the following address list:

```
A(parameter area)
A(current output record)
X'80000000'
```

A description of the structure of the parameter area is provided on page 179.

The following values are entered in the parameter area:

Identifier of the interface: X'08'

The user transfers information to PERCON via the return code of the parameter area.

Return code	**Meaning**
X'00'	Transfer record. The contents of the output record can be modified by the user. The record length cannot be modified. The first 4 bytes of variable-length records must not be overwritten.
X'08'	Do not transfer record. The record is not transferred to the current output file.
X'10'	Do not branch back to the user module. PERCON does not branch back to the user exit for this output file. The current record, however, is transferred.
X'14'	Terminate writing to current output file. PERCON no longer outputs records to the current output file; even the last record to be transferred to the user module is rejected.
X'18'	Terminate conversion step immediately with an error. The current record is not transferred.

6.3.4 Interface for group processing

If the structure operand value *MODULE(NAME=modulename) is specified in the GROUP-HEADER or GROUP-TRAILER operand of the SET-GROUP-ATTRIBUTES statement, PERCON branches to the user module specified by “modulename” in the event of a group break for this level. In this routine the user can generate a group leader or group trailer line and a printer feed control character. If the line which had previously been edited by PERCON has not yet been output (using *SPACING or *NEW-PAGE), this line is made available to the user. If the header or trailer line had not been set before the user module was called, it contains space characters (X’40’).

Register 1 indicates the following address list:

```
A(parameter area)
A(print line)
X'80000000'
```

A description of the structure of the parameter area is provided on page 179.

The following values are entered in the parameter area:

Identifier of the interface: X'0C'

Additional Information	Meaning
X'00'	Group leader
X'04'	Group trailer

The user transfers information to PERCON via the return code of the parameter area.

Return code	Meaning
X'00'	Print out print line.
X'04'	Do not branch back to user module. PERCON does not branch back to the user exit for this output file.
X'08'	Terminate output to this printer file. PERCON initiates the trailer routines for all group levels which have been opened and terminates output to this printer file.

The print line transferred by PERCON comprises 205 characters, the first of which is a printer feed control character while the remaining 204 are the data characters of the header or trailer line. If the printer control character is set by the user module, it must be transferred as an EBCDIC control character, regardless of the format specified in the PRINT-CONTROL operand of the ADD-FILE-LINK statement. The meanings of the individual bits of the EBCDIC control characters are given below.

The maximum length of a group break line is the product of the line leader (see the SET-PAGE-LAYOUT statement, page 142) and the COLUMN-SIZE, LINE-SIZE and OUTPUT-FORMAT operands of the SET-PAGE-LAYOUT statement. Data extending beyond this length is not transferred.

EBCDIC control characters

The individual bytes have the following meanings:

Line feed:

Before	**After**	**printing**
Byte	**Byte**	**Number of lines**
X'40'	X'00' *	No line feed
X'41'	X'01'	1 line
X'42'	X'02'	2 lines
X'43'	X'03'	3 lines
.	.	.
.	.	.
.	.	.
X'4F'	X'0F'	15 lines

* Control byte X’00’ can suppress the line feed once only.

This overview does not take account of the line which is advanced automatically.

Channel feed:

Before	After	printing
Byte	**Byte**	**Feed to channel**
X'C0'	X'80'	Not permitted
X'C1'	X'81'	1
X'C2'	X'82'	2
X'C3'	X'83'	3
.	.	.
.	.	.
.	.	.
X'C8'	X'88'	8
X'CA'	X'8A'	10
X'CB'	X'8B'	11

6.3.5 Interface for the recovery of read/length errors

If *MODULE(NAME=modulename) is specified for any of the operands PARITY-ERROR, LENGTH-ERROR or INPUT-ERROR in the ASSIGN-INPUT-FILE or ASSIGN-OUTPUT-FILE statement, PERCON branches to the user module “modulename” whenever a read or length error occurs.

Register 1 indicates the following address list:

```
A(parameter area)
A(errored block)
X'80000000'
```

A description of the structure of the parameter area is provided on page 179.

The following values are entered in the parameter area:

Identifier of the interface: X'10'

Additional information	Meaning
X'00'	Length error
X'04'	Read error

The user transfers information to PERCON via the return code of the parameter area.

Return code	Meaning
X'00'	Ignore error. PERCON informs DMS that the error is to be ignored. The errored block is made available by DMS and is processed as if it had been read correctly. Processing is continued normally.
X'04'	Skip block. PERCON informs DMS that the errored block is to be skipped. The next block is made available by DMS and processing is continued normally.
X'08'	Terminate processing of file. Processing of the input file is terminated. All output files are closed. The conversion step is terminated with an error.

6.3.6 Interface for the recovery of open errors

If the operand OPEN-ERROR=*MODULE(NAME=modulename) is specified in the ASSIGN-INPUT-FILE statement, PERCON branches to the user module “modulename” whenever an open error occurs in an input file.

Register 1 indicates the following address list:

```
A(parameter area)
X'80000000'
```

A description of the structure of the parameter area is provided on page 179.

The following values are entered in the parameter area:

Identifier of the interface: X'14'

The user transfers information to PERCON via the return code of the parameter area.

Return code	Meaning
X'00'	The conversion step is terminated with errors.
X'04'	The conversion step is terminated normally.
X'08'	The next input file is processed. The conversion step is terminated with errors.
X'0C'	The next input file is processed. The conversion step is terminated normally.

7 Installation

PERCON is installed by means of the installation monitor IMON. This means that it is not bound by any fixed file names or IDs.

PERCON can also be used as a subsystem with the options for exchangeability (EXCHANGE) and coexistence (COEXISTENCE) available in PERCON V2.6A or later.

Installation information on all software products is stored in the IMON software configuration inventory, and can be retrieved using IMON commands.

The following information is required to determine the location of a product file:

1. the product name (PERCON)
2. the logical file ID (e.g. SYSLNK)
3. the product version (029)

If a fully qualified version specification is submitted with the VERSION parameter of the start command or in the parameter list of the UP call, that version is activated (provided the version has been installed properly). If no version is specified (i.e. if *STD applies) or if the version specification is not detailed enough, more than one version may match the specification. In this case, the command server or PERCON obtains all versions matching the specification during UP call processing. The final selection is made according to the following priorities:

1. The version which was specified with the /SELECT-PRODUCT-VERSION command.
2. The highest PERCON version installed with IMON

If the IMON-GPN subsystem is not active at the PERCON start, the start is aborted and message PER0106 is output.

PERCON version V2.9A can be run under the BS2000 operating system version V5.0 or higher. The overview given below shows the components of PERCON that are to be installed:

file name	**logical ID**	**meaning**
PERCON	SYSPRG	Program file (compatibility)
SYSLNK.PERCON.029	SYSLNK	Object module library
SYSREP.PERCON.029	SYSREP	Rep file
SYSNRF.PERCON.029	SYSNRF	BLS rep processing
SYSRMS.PERCON.029	SYSRMS	RMS selectable unit
SYSSDF.PERCON.029	SYSSDF	SDF syntax file
SYSMES.PERCON.029	SYSMES	Message file
SYSSSC.PERCON.029	SYSSSC	subsystem catalogue
SYSSSC.PERCON.029.LOW	SYSSSC.LOW	subsystem catalogue (below 16 Mb)
SYSSII.PERCON.029	SYSSII	Structural and installation information file
SYSFGM.PERCON.029.D	SYSFGM.D	Release notes (german)
SYSFGM.PERCON.029.E	SYSFGM.E	Release notes (english)

Additional information:

1. If applications are run under BS2000/OSD-BC V6.0 or higher that call PERCON as a subprogram with a starter module of a version up to and including PERCON V2.5A linked in, it should be borne in mind that the main module PCNSR9 cannot be loaded dynamically by PERCON. The relevant application must be linked in again using the PERCON V2.9A starter module.
2. In SYSSSC files without the suffix LOW, the MEMORY-CLASS parameter has been set to a value that ensures that the subsystem is loaded in the higher address space (i.e. > 16 Mbytes). The PERCON subsystem, however, is loaded in the lower address space (i.e. < 16 Mbytes) if the SYSSSC files with suffix LOW are used.

8 Sample applications

The examples given in the next sections involve the use of the following two input files:

PERS.DPT.1 and PERS.DPT.2

```
/SHOW-FILE-ATTRIBUTES FILE-NAME=PERS.DPT.1,INFORMATION=*ALL-ATTRIBUTES
00000003 :catid:$userid.PERS.DPT.1
 ------------------------------- HISTORY      -------------------------------
 CRE-DATE   = 2001-07-11  ACC-DATE   = 2001-07-11  CHANG-DATE = 2001-07-11
 CRE-TIME   =   10:40:16  ACC-TIME   =   14:10:02  CHANG-TIME =   10:40:16
 ACC-COUNT  = 8           S-ALLO-NUM = 0
 ------------------------------- SECURITY     -------------------------------
 READ-PASS  = NONE        WRITE-PASS = NONE        EXEC-PASS  = NONE
 USER-ACC   = OWNER-ONLY  ACCESS     = WRITE       ACL        = NO
 AUDIT      = NONE        DESTROY    = NO          EXPIR-DATE = 2001-07-11
 SP-REL-LOCK= NO                                   EXPIR-TIME =   00:00:00
 ------------------------------- BACKUP       -------------------------------
 BACK-CLASS = A           SAVED-PAG  = COMPL-FILE  VERSION    = 1
 MIGRATE    = ALLOWED
 ------------------------------- ORGANIZATION -------------------------------
 FILE-STRUC = SAM         BUF-LEN    = STD(1)      BLK-CONTR  = PAMKEY
 IO(USAGE)  = READ-WRITE  IO(PERF)   = STD         DISK-WRITE = IMMEDIATE
 REC-FORM   = (V,N)       REC-SIZE   = 0
 ------------------------------- ALLOCATION   -------------------------------
 SUPPORT    = PUB         S-ALLOC    = 3           HIGH-US-PA = 1
 EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE
     1       PUBF00       D348F
 NUM-OF-EXT = 1
:catid:    PUBLIC:     1 FILE  RES=       3  FREE=       2  REL=      0 PAGES
```

Contents of the file PERS.DPT.1

```
BELL         JOHN       MANCHESTER    BOLSOVER STREET 4      DPT1
BOTHAM       NORMAN     MANCHESTER    TOWER AVENUE 10        DPT2
FINN         SUSANNA    NORWICH       ROSE STREET 11         DPT2
GREENE       WALTER     BURY          SINCLAIR CRESCENT 7    DPT1
KING         MONICA     FALMOUTH      INMAN SQUARE 61        DPT3
LAKER        ERICA      MANCHESTER    BANK DRIVE 8           DPT1
PRICE        ALFRED     MANCHESTER    THAMES ROAD 4          DPT1
WILSON       RICHARD    MANCHESTER    ACACIA AVENUE 24       DPT3
```

```
/SHOW-FILE-ATTRIBUTES FILE-NAME=PERS.DPT.2,INFORMATION=*ALL-ATTRIBUTES
0000003 :catid:$userid.PERS.DPT.2
 ------------------------------ HISTORY      ------------------------------
 CRE-DATE   = 2001-07-11  ACC-DATE   = 2001-07-11  CHANG-DATE = 2001-07-11
 CRE-TIME   =   10:40:19  ACC-TIME   =   14:10:03  CHANG-TIME =   10:40:19
 ACC-COUNT  = 8           S-ALLO-NUM = 0
 ------------------------------ SECURITY     ------------------------------
 READ-PASS  = NONE        WRITE-PASS = NONE        EXEC-PASS  = NONE
 USER-ACC   = OWNER-ONLY  ACCESS     = WRITE       ACL        = NO
 AUDIT      = NONE        DESTROY    = NO          EXPIR-DATE = 2001-07-11
 SP-REL-LOCK= NO                                   EXPIR-TIME =   00:00:00
 ------------------------------ BACKUP       ------------------------------
 BACK-CLASS = A           SAVED-PAG  = COMPL-FILE  VERSION    = 1
 MIGRATE    = ALLOWED
 ------------------------------ ORGANIZATION ------------------------------
 FILE-STRUC = SAM         BUF-LEN    = STD(1)      BLK-CONTR  = PAMKEY
 IO(USAGE)  = READ-WRITE  IO(PERF)   = STD         DISK-WRITE = IMMEDIATE
 REC-FORM   = (V,N)       REC-SIZE   = 0
 ------------------------------ ALLOCATION   ------------------------------
 SUPPORT    = PUB         S-ALLOC    = 3           HIGH-US-PA = 1
 EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE
     1       PUBF00       D348F
 NUM-OF-EXT = 1
:catid:    PUBLIC:     1 FILE  RES=       3  FREE=       2  REL=      0 PAGES
```

Contents of the file PERS.DPT.2

```
BATES        FRANK      HULL          WOOD STREET 29         DPT4
HARRISON     MAURICE    MANCHESTER    SINCLAIR STREET 149    DPT4
ROYCE        ROBERT     MANCHESTER    LION'S GATE 74         DPT4
SIMPSON      ELLEN      KINGS LYNN    WILLOUGHBY STREET 8    DPT4
```

Meaning of the columns in PERS.DPT.1 and PERS.DPT.2

```
Column
5.......     17.......  29.......     44......               69.......
Last name    1st name   Town          Street                 Department
```

8.1 Conversion of two SAM files into one ISAM file

Input:

SAM file PERS.DPT.1 on disk
SAM file PERS.DPT.2 on disk

Output:

ISAM file PERS.DPT,RECORD-FORMAT=*FIXED,RECORD-SIZE=100 on disk
SAM file PERS.TA,RECORD-FORMAT=*VARIABLE on tape

Tracer listing:

```
/CREATE-FILE FILE-NAME=PERS.DPT ————————————————————————————————————   (1)
/ADD-FILE-LINK FILE-NAME=PERS.DPT,-
/               RECORD-FORMAT=*FIXED,RECORD-SIZE=100,-
/               SUPPORT=*DISK,-
/               LINK-NAME=OUT1,-
/               ACCESS-METHOD=*ISAM
/CREATE-FILE FILE-NAME=PERS.TA,- ———————————————————————————————————   (2)
/            SUPPORT=*TAPE(VOLUME=BD0002,DEVICE-TYPE=T6250)
/ADD-FILE-LINK FILE-NAME=PERS.TA,-
/              SUPPORT=*TAPE,-
/              LINK-NAME=OUT2,-
/              ACCESS-METHOD=*SAM
/START-PERCON ——————————————————————————————————————————————————————   (3)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-FILE FILE=*DISK-FILE(NAME=PERS.DPT.1),- ———————————————   (4)
//                  LINK-NAME=IN1
//ASSIGN-INPUT-FILE FILE=*DISK-FILE(NAME=PERS.DPT.2),- ———————————————   (5)
//                  LINK-NAME=IN2
```

(1) The disk file PERS.DPT is assigned with its various attributes.

(2) The tape file PERS.TA is assigned with its various attributes.

(3) PERCON is called.

(4) The input file PERS.DPT.1 is assigned with the link name IN1.

(5) The input file PERS.DPT.2 is assigned with the link name IN2.

```
//ASSIGN-OUTPUT-FILE FILE=*DISK-FILE(NAME=PERS.DPT),- ————————————————— (6)
//                   LINK-NAME=OUT1
//ASSIGN-OUTPUT-FILE FILE=*TAPE-FILE(NAME=PERS.TA),- —————————————————— (7)
//                   LINK-NAME=OUT2
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=OUT1,- —————————————————————————— (8)
//                   OUTPUT-FIELDS=(*RECORD-COUNTER(LINK-NAME=OUT1,-
//                                                  OUTPUT-POSITION=1,-
//                                                  OUTPUT-LENGTH=8,-
//                                                  OUTPUT-FORMAT=*ZONED-DECIMAL),-
//                                        *FIELD(INPUT-POSITION=5,-
//                                                  INPUT-LENGTH=68,-
//                                                  OUTPUT-POSITION=14,-
//                                                  OUTPUT-LENGTH=68))
//END ————————————————————————————————————————————————————————————————— (9)
%  DMS0DE3 TAPE WITH VSN BD0002 FOR FILE :catid:$userid.PERS.TA IS ——— (10)
           MOUNTED ON DEVICE AF
%  DMS0DE7 SAM FILE CLOSED: FILE NAME=:catid:$userid.PERS.TA,
           LINKNAME=OUT2, BLOCK COUNT=000001 ————————————————————————— (11)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=IN1 (FILE=PERS.DPT.1): 8 (12)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=IN2 (FILE=PERS.DPT.2): 4
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=OUT1 (FILE=PERS.DPT): 12
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=OUT2 (FILE=PERS.TA): 12
%  PER0031 PERCON TERMINATED NORMALLY ———————————————————————————————— (13)
/PRINT-FILE FILE-NAME=PERS.DPT
%  SCP0810 SPOOLOUT OF FILE ':catid:$userid.PERS.DPT' ACCEPTED: TSN: '2203',
           PNAME: 'TINE'
```

(6) The output file PERS.DPT is assigned with the link name OUT1.

(7) The output file PERS.TA is assigned with the link name OUT2.

(8) The values of the record counter are transferred to the output file PERS.DPT.
The record counter is transferred to the output record in zoned format with a length of 8 bytes, starting at byte 1.
In this way the ISAM key is created for the ISAM file PERS.DPT. 68 bytes of the input record are written to the output record, starting from byte position 14.

(9) The END statement starts the transfer operation and terminates PERCON.

(10) DMS message: the tape for the file PERS.DPT has been mounted.

(11) DMS message: the SAM file PERS.TA has been closed.

(12) PERCON messages: the number of records transferred per file is output. The files are listed with their link and file names.

(13) PERCON was terminated normally.

Printout of the file PERS.DPT

```
00000000   BELL        JOHN       MANCHESTER   BOLSOVER STREET 4      DPT1
00000001   BOTHAM      NORMAN     MANCHESTER   TOWER AVENUE 10        DPT2
00000002   FINN        SUSANNA    NORWICH      ROSE STREET 11         DPT2
00000003   GREENE      WALTER     BURY         SINCLAIR CRESCENT 7    DPT1
00000004   KING        MONICA     FALMOUTH     INMAN SQUARE 61        DPT3
00000005   LAKER       ERICA      MANCHESTER   BANK DRIVE 8           DPT1
00000006   PRICE       ALFRED     MANCHESTER   THAMES ROAD 4          DPT1
00000007   WILSON      RICHARD    MANCHESTER   ACACIA AVENUE 24       DPT3
00000008   BATES       FRANK      HULL         WOOD STREET 29         DPT4
00000009   HARRISON    MAURICE    MANCHESTER   SINCLAIR STREET 149    DPT4
00000010   ROYCE       ROBERT     MANCHESTER   LION'S GATE 74         DPT4
00000011   SIMPSON     ELLEN      KINGS LYNN   WILLOUGHBY STREET 8    DPT4
```

8.2 Conversion of a noncataloged tape file to disk

Input:

SAM file PERS.TA on tape

Output:

- ISAM file PERS.TAB
  The ISAM record key is created with the record counter.
  The output records are restructured.
- Output to SYSLST
  All the records containing C'MANCHESTER', starting from column 29, are selected for printer output.

Tracer listing:

```
/EXPORT-FILE FILE=*NAME(FILE-NAME=PERS.TA) ──────────────────────────── (1)
/IMPORT-FILE SUPPORT=*TAPE(FILE-NAME=PERS.TA,-
/                          VOLUME=BD0002,DEVICE-TYPE=T6250)
/ADD-FILE-LINK FILE-NAME=PERS.TA,-
/              SUPPORT=*TAPE,-
/              LINK-NAME=IN
/CREATE-FILE FILE-NAME=PERS.TAB ──────────────────────────────────────── (2)
/ADD-FILE-LINK FILE-NAME=PERS.TAB,-
/              RECORD-FORMAT=*FIXED,RECORD-SIZE=82,-
/              SUPPORT=*DISK,-
/              LINK-NAME=OUT1
/START-PERCON ─────────────────────────────────────────────────────────── (3)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-FILE FILE=*TAPE-FILE(NAME=PERS.TA),- ─────────────────── (4)
//                  LINK-NAME=IN
//ASSIGN-OUTPUT-FILE FILE=*DISK-FILE(NAME=PERS.TAB),- ───────────────── (5)
//                   LINK-NAME=OUT1
```

(1) The noncataloged tape file PERS.TA with its attributes is assigned.

(2) The output file PERS.TAB with its attributes is assigned.

(3) PERCON is called.

(4) The noncataloged tape file PERS.TA is assigned.

(5) The output file PERS.TAB is assigned with the link name OUT1.

```
//ASSIGN-OUTPUT-FILE FILE=*SYSLST,- ―――――――――――――――――――――――――――――――――――   (6)
//                   LINK-NAME=OUT2
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=OUT1,- ――――――――――――――――――――――――――   (7)
//                   OUTPUT-FIELDS=(*RECORD-COUNTER(LINK-NAME=OUT1,-
//                                                  OUTPUT-POSITION=1,-
//                                                  OUTPUT-LENGTH=8,-
//                                                  OUTPUT-FORMAT=*ZONED-DECIMAL),-
//                                  '|'(OUTPUT-POSITION=10),-
//                                  *FIELD(INPUT-POSITION=5,-
//                                        INPUT-LENGTH=10,-
//                                        OUTPUT-POSITION=12),-
//                                  '|'(OUTPUT-POSITION=24),-
//                                  *FIELD(INPUT-POSITION=17,-
//                                        INPUT-LENGTH=10,-
//                                        OUTPUT-POSITION=26),-
//                                  '|'(OUTPUT-POSITION=37),-
//                                  *FIELD(INPUT-POSITION=29,-
//                                        INPUT-LENGTH=10,-
//                                        OUTPUT-POSITION=39),-
//                                  '|'(OUTPUT-POSITION=51),-
//                                  *FIELD(INPUT-POSITION=44,-
//                                        INPUT-LENTGH=20,-
//                                        OUTPUT-POSITION=53),-
//                                  '|'(OUTPUT-POSITION=75),-
//                                  *FIELD(INPUT-POSITION=69,-
//                                        INPUT-LENGTH=4,-
//                                        OUTPUT-POSITION=77),-
//                                  '|'(OUTPUT-POSITION=82))
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=OUT2,- ――――――――――――――――――――――――   (8)
//                     CONDITION=((29,10)=C'MANCHESTER')
//END ―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――   (9)
```

(6) The printer output file is assigned with the link name OUT2.

(7) The SET-RECORD-MAPPING statement transfers the record counter and selected fields of the input record to the output record.

(8) The SELECT-INPUT-RECORDS statement selects all the records containing C’MANCHESTER’ as of column 29.
The records are transferred to the print file with the link name OUT2.

(9) The END statement starts the transfer operation and terminates PERCON.

```
%  DMSODE3 TAPE WITH VSN BD0002 FOR FILE :catid:$userid.PERS.TA IS ──── (10)
           MOUNTED ON DEVICE AF
%  DMSODE7 SAM FILE CLOSED: FILE NAME=:catid:$userid.PERS.TA, LINKNAME=IN,
           BLOCK COUNT=000001 ──────────────────────────────────────── (11)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=IN (FILE=PERS.TA):    12 (12)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=OUT1 (FILE=PERS.TAB):   12
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=OUT2  :         7
%  PER0031 PERCON TERMINATED NORMALLY ──────────────────────────────── (13)
/PRINT-FILE FILE-NAME=PERS.TAB
%  SCP0810 SPOOLOUT OF FILE ':catid:$userid.PERS.TAB' ACCEPTED: TSN: '2205',
           PNAME: 'TINE'
```

(10) DMS message: the tape with the volume serial number BD0002, which contains the input file, has been mounted.

(11) DMS message: the SAM file PERS.TA has been closed.

(12) PERCON messages: the number of records transferred per file is output. The files are listed with their link and file names.

(13) PERCON was terminated normally.

Printout of the file PERS.TAB

```
00000000 | BELL      | JOHN    | MANCHESTER  | BOLSOVER STREET 4    | DPT1 |
00000001 | BOTHAM    | NORMAN  | MANCHESTER  | TOWER AVENUE 10      | DPT2 |
00000002 | FINN      | SUSANNA | NORWICH     | ROSE STREET 11       | DPT2 |
00000003 | GREENE    | WALTER  | BURY        | SINCLAIR CRESCENT 7  | DPT1 |
00000004 | KING      | MONICA  | FALMOUTH    | INMAN SQUARE 61      | DPT3 |
00000005 | LAKER     | ERICA   | MANCHESTER  | BANK DRIVE 8         | DPT1 |
00000006 | PRICE     | ALFRED  | MANCHESTER  | THAMES ROAD 4        | DPT1 |
00000007 | WILSON    | RICHARD | MANCHESTER  | ACACIA AVENUE 24     | DPT3 |
00000008 | BATES     | FRANK   | HULL        | WOOD STREET 29       | DPT4 |
00000009 | HARRISON  | MAURICE | MANCHESTER  | SINCLAIR STREET 149  | DPT4 |
00000010 | ROYCE     | ROBERT  | MANCHESTER  | LION'S GATE 74       | DPT4 |
00000011 | SIMPSON   | ELLEN   | KINGS LYNN  | WILLOUGHBY STREET 8  | DPT4 |
```

Printout of the selected records

```
 BELL         JOHN       MANCHESTER    BOLSOVER STREET 4      DPT1
 BOTHAM       NORMAN     MANCHESTER    TOWER AVENUE 10        DPT2
 LAKER        ERICA      MANCHESTER    BANK DRIVE 8           DPT1
 PRICE        ALFRED     MANCHESTER    THAMES ROAD 4          DPT1
 WILSON       RICHARD    MANCHESTER    ACACIA AVENUE 24       DPT3
 HARRISON     MAURICE    MANCHESTER    SINCLAIR STREET 149    DPT4
 ROYCE        ROBERT     MANCHESTER    LION'S GATE 74         DPT4
```

8.3 Transferring records to a residual file

All records of a file that contain “MANCHESTER” as the town name are to be transferred to the file PERS.MANCHESTER, those with “NORWICH” as the town entry are to be transferred to the file PERS.NORWICH. The remaining records are to be transferred to the file REST-OF-THE-WORLD. The name of the input file is PERS.DPT.1.

Tracer listing:

```
/ADD-FILE-LINK FILE-NAME=PERS.MANCHESTER,SUPPORT=*DISK,- ──────────── (1)
/              LINK-NAME=OUT1,ACCESS-METHOD=*SAM
/ADD-FILE-LINK FILE-NAME=PERS.NORWICH,SUPPORT=*DISK,-
/              LINK-NAME=OUT2,ACCESS-METHOD=*SAM
/ADD-FILE-LINK FILE-NAME=REST-OF-THE-WORLD,SUPPORT=*DISK,-
/              LINK-NAME=OUT3,ACCESS-METHOD=*SAM
/START-PERCON ─────────────────────────────────────────────────────── (2)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-FILE  FILE=*DISK-FILE(NAME=PERS.DPT.1),- ───────────── (3)
//                   LINK-NAME=IN1
//ASSIGN-OUTPUT-FILE FILE=*DISK-FILE(NAME=PERS.MANCHESTER),- ──────── (4)
//                   LINK-NAME=OUT1
//ASSIGN-OUTPUT-FILE FILE=*DISK-FILE(NAME=PERS.NORWICH),- ─────────── (5)
//                   LINK-NAME=OUT2
//ASSIGN-OUTPUT-FILE FILE=*DISK-FILE(NAME=REST-OF-THE-WORLD),- ────── (6)
//                   LINK-NAME=OUT3
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=OUT1,- ────────────────────── (7)
//                     CONDITION=((29,10)='MANCHESTER')
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=OUT2,- ────────────────────── (8)
//                     CONDITION=((29,7)='NORWICH')
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=OUT3,- ────────────────────── (9)
//                     CONDITION=*REMAINING-RECORDS
//END ─────────────────────────────────────────────────────────────── (10)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='IN1' ──────────────── (11)
          (FILE=:catid:$userid.PERS.DPT.1):         8
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='OUT1'
          (FILE=:catid:$userid.PERS.MANCHESTER):         5
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='OUT2'
          (FILE=:catid:$userid.PERS.NORWICH):         1
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='OUT3'
          (FILE=:catid:$userid.REST-OF-THE-WORLD):         2
%  PER0031 PERCON TERMINATED NORMALLY ──────────────────────────────── (12)
```

(1) The /ADD-FILE-LINK command defines the three output files including their file attributes.

(2) PERCON is called.

(3) The input file PERS.DPT.1 is assigned with the link name IN1.

(4) The output file PERS.MANCHESTER is assigned with the link name OUT1.

(5) The output file PERS.NORWICH is assigned with the link name OUT2.

(6) The output file REST-OF-THE-WORLD is assigned with the link name OUT3.

(7) The SELECT-INPUT-RECORDS statement selects all records containing the entry C'MANCHESTER' starting at column 29. The records are transferred to the output file with the link name OUT1.

(8) The SELECT-INPUT-RECORDS statement selects all records containing the entry C'NORWICH' starting at column 29. The records are transferred to the output file with the link name OUT2.

(9) The SELECT-INPUT-RECORDS statement defines the file with the link name OUT3 as residual file i.e. the file to which all input records are to be transferred that do not contain either of the entries C'MANCHESTER' or C'NORWICH' starting at column 29.

(10) The END statement starts the transfer operation and terminates PERCON. Any records not transferred to either of the output files with the link names OUT1 and OUT2 are transferred to the output file with the link name OUT3.

(11) PERCON messages:
The number of records transferred per file is output.

(12) PERCON was terminated normally.

Printout of the file PERS.MANCHESTER

```
BELL        JOHN        MANCHESTER      BOLSOVER STREET 4       DPT1
BOTHAM      NORMAN      MANCHESTER      TOWER AVENUE 10         DPT2
LAKER       ERICA       MANCHESTER      BANK DRIVE 8            DPT1
PRICE       ALFRED      MANCHESTER      THAMES ROAD 4           DPT1
WILSON      RICHARD     MANCHESTER      ACACIA AVENUE 24        DPT3
```

Printout of the file PERS.NORWICH

```
FINN        SUSANNA     NORWICH         ROSE STEET 11           DPT2
```

Printout of the file REST-OF-THE-WORLD

```
GREENE      WALTER      BURY            SINCLAIR CRESCENT 7     DPT1
KING        MONICA      FALMOUTH        INMAN SQUARE 61         DPT3
```

8.4 Updating fields in an existing output record

The ISAM file PERSONNEL already exists and is available as output file; it has the following attributes:

```
RECORD-FORMAT = FIXED    RECORD-SIZE = 60
KEY-POSITION  = 1        KEY-LENGTH  = 4
```

Contents of the file PERSONNEL prior to the PERCON run:

```
Column
1-4        5-17       18-29     30-34  35-39    41-60

Pers.no.   Last name  1st name  Dpt.   Salary   Town

0005       MILLER     HOWARD    A      06500    MANCHESTER
0008       BROWN      KAREN     B1     04800    FULBOURN
0012       INGRAM     OLIVER    A23    03450    UTTOXETER
0015       ALBURY     IRENE     B12    02880    MANCHESTER
0023       BERGER     ALBERT    A1     04250    NORWICH
0036       BOTHA      BARBARA   K2     04300    WORTHING
```

The file is to be updated, incorporating modifications in the salary and town columns. The update data is to be taken from the PERS.UPDATE file. A 3-character identifier is to be used to indicate the field to be updated. The personnel number (key field) is used to link input and output records:

Structure of the update records (variable record format):

```
Column
5-7    8-11       12 - max. 31

Id.    Pers.no.   Update data (depending on identifier)

TWN    0015       READING            Change of town
SAL    0008       05300              Change of salary
TWN    0011       SHEFFIELD          Change of town, but pers.no.
                                     not found
```

PERCON statements:

```
/ADD-FILE-LINK LINK-NAME=PCOUT1,FILE-NAME=PERSONNEL,- ─────────────────── (1)
/              OPEN-MODE=*INOUT
/ADD-FILE-LINK LINK-NAME=PCOUT2,FILE-NAME=PERSONNEL,-
/              OPEN-MODE=*INOUT
/START-PERCON ─────────────────────────────────────────────────────────── (2)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-FILE FILE=*DISK-FILE(NAME=PERS.UPDATE) ─────────────────── (3)
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT1 ──────────────────────────────────── (4)
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT2
//SELECT-INPUT-RECORDS CONDITION=((5,3)='SAL'),- ──────────────────────── (5)
//                     OUTPUT-LINK-NAME=PCOUT1
//SELECT-INPUT-RECORDS CONDITION=((5,3)='TWN'),- ──────────────────────── (6)
//                     OUTPUT-LINK-NAME=PCOUT2
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT1,- ────────────────────────── (7)
//    FILLER=*OUTPUT(KEY-NAME=*PRIMARY,-
//                   KEY-VALUE=*BY-INPUT-RECORD(KEY-POSITION=8)),-
//    OUTPUT-FIELDS=*FIELD(INPUT-POSITION=12,INPUT-LENGTH=5,-
//                         INPUT-FORMAT=*ZONED-DECIMAL,-
//                         OUTPUT-POSITION=35)
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT2,- ────────────────────────── (8)
//    FILLER=*OUTPUT(KEY-NAME=*PRIMARY,-
//                   KEY-VALUE=*BY-INPUT-RECORD(KEY-POSITION=8)),-
//    OUTPUT-FIELDS=*FIELD(INPUT-POSITION=12,-
//                         INPUT-LENGTH=*RECORD-LENGTH(REDUCTION=7),-
//                         OUTPUT-POSITION=41,OUTPUT-LENGTH=20)
//END ─────────────────────────────────────────────────────────────────── (9)
```

(1) If ISAM files are used, several ADD-FILE-LINK commands with OPEN-MODE=*INOUT and different link names can be issued for one and the same output file. These link names serve to establish links with the various PERCON statements.

Note

Specifying FILLER=*OUTPUT in the SET-RECORD-MAPPING statement ensures that the output files concerned are always opened with OPEN-MODE=*INOUT; the corresponding specification in /ADD-FILE-LINK is therefore optional.

(2) PERCON is started.

(3) The input file PERS.UPDATE, which contains the update data, is assigned.

(4) The output file is assigned several times using different link names.

(5) This statement selects those records from the input file which are to be used to update the “Salary” field via the identifier SAL. The link name PCOUT1 establishes the link with the SET-RECORD-MAPPING statement which modifies the field “Salary”.

(6) This statement selects those records from the input file which are to be used to update the “Town” field via the identifier TWN. The link name PCOUT2 establishes the link with the SET-RECORD-MAPPING statement which modifies the field “Town”.

(7) This statement causes a record from the output file to be used to prefill the output area. The record is selected using the field “Pers.no.” contained in the update record. The update data itself is to be transferred to the “Salary” field in the output record.

(8) This statement causes a record from the output file to be used to prefill the output area. The record is selected using the field “Pers.no.” contained in the update record. The update data itself is to be transferred to the “Town” field in the output record.

(9) The END statement starts the transfer operation and terminates PERCON.

Contents of the file PERSONNEL after a PERCON run using the above statements:

Two records have been updated, the ones with personnel numbers 0008 and 0015.

```
Column
1-4         5-17        18-29      30-34  35-39    41-60

Pers.no.    Last name   1st name   Dpt.   Salary   Town

0005        MILLER      HOWARD     A      06500    MANCHESTER
0008        BROWN       KAREN      B1     05300    FULBOURN ─────────────── (10)
0012        INGRAM      OLIVER     A23    03450    UTTOXETER
0015        ALBURY      IRENE      B12    02880    READING ──────────────── (11)
0023        BERGER      ALBERT     A1     04250    NORWICH
0036        BOTHA       BARBARA    K2     04300    WORTHING
```

(10) In this record, the “Salary” field has been updated.

(11) In this record, the “Town” field has been updated.

The file PERSONNEL does not contain any record with personnel number 0011. When the relevant update record is processed, message PER0054 is output.

8.5 Output of a file to tape with simultaneous code conversion

Input: ISAM file PERS.TAB

Printout of the input file PERS.TAB

```
00000000 | BELL      | JOHN     | MANCHESTER  | BOLSOVER STREET 4     | DPT1 |
00000001 | BOTHAM    | NORMAN   | MANCHESTER  | TOWER AVENUE 10       | DPT2 |
00000002 | FINN      | SUSANNA  | NORWICH     | ROSE STREET 11        | DPT2 |
00000003 | GREENE    | WALTER   | BURY        | SINCLAIR CRESCENT 7   | DPT1 |
00000004 | KING      | MONICA   | FALMOUTH    | INMAN SQUARE 61       | DPT3 |
00000005 | LAKER     | ERICA    | MANCHESTER  | BANK DRIVE 8          | DPT1 |
00000006 | PRICE     | ALFRED   | MANCHESTER  | THAMES ROAD 4         | DPT1 |
00000007 | WILSON    | RICHARD  | MANCHESTER  | ACACIA AVENUE 24      | DPT3 |
00000008 | BATES     | FRANK    | HULL        | WOOD STREET 29        | DPT4 |
00000009 | HARRISON  | MAURICE  | MANCHESTER  | SINCLAIR STREET 149   | DPT4 |
00000010 | ROYCE     | ROBERT   | MANCHESTER  | LION'S GATE 74        | DPT4 |
00000011 | SIMPSON   | ELLEN    | KINGS LYNN  | WILLOUGHBY STREET 8   | DPT4 |
```

Output: SAM file PERS.CON with converted code

Conversion table

This table is generated as an Assembler source program using definitions of constants and compiled using the assembler.

```
TRANS    START
TRANS    AMODE ANY
TRANS    RMODE ANY
         DC    X'000102030405060708090A0B0C0D0E0F'
         DC    X'101112131415161718191A1B1C1D1E1F'
         DC    X'202122232425262728292A2B2C2D2E2F'
         DC    X'303132333435363738393A3B3C3D3E3F'
         DC    X'404142434445464748494A4B4C4D4E5C'
         DC    X'505152535455565758595A5B4F5D5E5F'
         DC    X'606162636465666768696A6B6C6D6E6F'
         DC    X'707172737475767778797A7B7C7D7E7F'
         DC    X'808182838485868788898A8B8C8D8E8F'
         DC    X'909192939495969798999A9B9C9D9E9F'
         DC    X'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'
         DC    X'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'
         DC    X'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'
         DC    X'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'
         DC    X'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'
         DC    X'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'
         END   TRANS
```

The object module TRANS is the translation of the table. It is entered in an object module library (TRANS.LIB) created with LMS. PERCON accesses object module TRANS as a member of the TRANS.LIB library and uses TRANS as a code conversion table. Prior to the PERCON call, library TRANS.LIB must therefore have been specified (assigned) as TASKLIB for the operating system.

Tracer listing:

```
/SET-TASKLIB LIBRARY=TRANS.LIB ---------------------------------------------  (1)
/CREATE-FILE FILE-NAME=PERS.CON,- ------------------------------------------  (2)
/            SUPPORT=*TAPE(VOLUME=BD0002,DEVICE-TYPE=T6250)
/ADD-FILE-LINK FILE-NAME=PERS.CON,-
/              LINK-NAME=OUT,-
/              SUPPORT=*TAPE,-
/              ACCESS-METHOD=*SAM,-
/              BUFFER-LENGTH=2048
/START-PERCON --------------------------------------------------------------  (3)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-FILE FILE=*DISK-FILE(NAME=PERS.TAB) --------------------  (4)
//ASSIGN-OUTPUT-FILE FILE=*TAPE-FILE(NAME=PERS.CON,- -----------------  (5)
//                               CODE-TRANSLATION=*MODULE(NAME=TRANS)),-
//                   LINK-NAME=OUT
//END -----------------------------------------------------------------  (6)
%  DMS0DE3 TAPE WITH VSN BD0002 FOR FILE :catid:$userid.PERS.TAB IS ---  (7)
           MOUNTED ON DEVICE AF
%  DMS0DE7 SAM FILE CLOSED: FILE NAME=:catid:$userid.PERS.CON,
           LINKNAME=OUT, BLOCK COUNT=000001 ---------------------------  (8)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=PCIN (FILE=PERS.TAB):   12 (9)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=OUT (FILE=PERS.CON):   12
%  PER0031 PERCON TERMINATED NORMALLY ---------------------------------  (10)
```

(1) The library TRANS.LIB ia assigned as TASKLIB. This library contains the code conversion table TRANS as an object module.

(2) The output file PERS.CON is assigned with its file attributes.

(3) PERCON is called.

(4) The input file PERS.TAB is assigned.

(5) The output file PERS.CON is assigned. CODE-TRANSLATION= *MODULE(NAME=TRANS) means that the code conversion table to be used to write the output file is located in object module TRANS (module name).

(6) The END statement starts the transfer operation and terminates PERCON.

(7) DMS message: the tape has been mounted for the output file PERS.CON.

(8) PERCON messages: the number of records transferred per file is output. The files are listed with their link and file names.

(9) DMS message: the SAM file PERS.CON has been closed.

(10) PERCON was terminated normally.

Printout of the output file PERS.CON

```
00000000 * BELL      * JOHN     * MANCHESTER  * BOLSOVER STREET 4     * DPT1 *
00000001 * BOTHAM    * NORMAN   * MANCHESTER  * TOWER AVENUE 10       * DPT2 *
00000002 * FINN      * SUSANNA  * NORWICH     * ROSE STREET 11        * DPT2 *
00000003 * GREENE    * WALTER   * BURY        * SINCLAIR CRESCENT 7   * DPT1 *
00000004 * KING      * MONICA   * FALMOUTH    * INMAN SQUARE 61       * DPT3 *
00000005 * LAKER     * ERICA    * MANCHESTER  * BANK DRIVE 8          * DPT1 *
00000006 * PRICE     * ALFRED   * MANCHESTER  * THAMES ROAD 4         * DPT1 *
00000007 * WILSON    * RICHARD  * MANCHESTER  * ACACIA AVENUE 24      * DPT3 *
00000008 * BATES     * FRANK    * HULL        * WOOD STREET 29        * DPT4 *
00000009 * HARRISON  * MAURICE  * MANCHESTER  * SINCLAIR STREET 149   * DPT4 *
00000010 * ROYCE     * ROBERT   * MANCHESTER  * LION'S GATE 74        * DPT4 *
00000011 * SIMPSON   * ELLEN    * KINGS LYNN  * WILLOUGHBY STREET 8   * DPT4 *
```

8.6 Sorting an ISAM file

Input:

SAM file PERS.FIL

Output:

ISAM file PERS.LIS

Printout of the input file PERS.FIL

```
FINN        SUSANNA     NORWICH        ROSE STREET 11            DPT2
BELL        JOHN        MANCHESTER     BOLSOVER STREET 4         DPT1
GREENE      WALTER      BURY           SINCLAIR CRESCENT 7       DPT1
LAKER       ERICA       MANCHESTER     BANK DRIVE 8              DPT1
WILSON      RICHARD     MANCHESTER     ACACIA AVENUE 24          DPT3
BOTHAM      NORMAN      MANCHESTER     TOWER AVENUE 10           DPT2
PRICE       ALFRED      MANCHESTER     THAMES ROAD 4             DPT1
KING        MONICA      FALMOUTH       INMAN SQUARE 61           DPT3
```

Tracer listing:

```
/CREATE-FILE FILE-NAME=PERS.LIS ——————————————————————————————————————   (1)
/ADD-FILE-LINK LINK-NAME=PCOUT,-
/          FILE-NAME=PERS.LIS,-
/          ACCESS-METHOD=*ISAM,-
/          SUPPORT=*DISK(ISAM-ATTRIBUTES=(KEY-POSITION=5,KEY-LENGTH=25)),-
/          OPEN-MODE=*OUTIN
/START-PERCON ————————————————————————————————————————————————————————   (2)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-FILE FILE=*DISK-FILE(NAME=PERS.FIL) ————————————————————   (3)
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT,- ————————————————————————————————   (4)
//                   FILE=*DISK-FILE(OVERWRITE=*YES)
//END ————————————————————————————————————————————————————————————————   (5)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=PCIN (FILE=PERS.FIL): 8   (6)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=PCOUT (FILE=PERS.LIS): 8
%  PER0031 PERCON TERMINATED NORMALLY ————————————————————————————————   (7)
```

(1) The output file PERS.LIS is created via a CREATE-FILE command and assigned with its file attributes by means of a ADD-FILE-LINK command.

(2) PERCON is called.

(3) The input file PERS.FIL is assigned.

(4) The output file PERS.LIS is assigned with its file attributes.

(5) The END statement starts the transfer operation and terminates PERCON.

(6) PERCON messages: the number of records transferred per file is output. The files are listed with their link and file names.

(7) PERCON was terminated normally.

Printout of the output file PERS.LIS

```
BELL         JOHN       MANCHESTER    BOLSOVER STREET 4       DPT1
BOTHAM       NORMAN     MANCHESTER    TOWER AVENUE 10         DPT2
FINN         SUSANNA    NORWICH       ROSE STREET 11          DPT2
GREENE       WALTER     BURY          SINCLAIR CRESCENT 7     DPT1
KING         MONICA     FALMOUTH      INMAN SQUARE 61         DPT3
LAKER        ERICA      MANCHESTER    BANK DRIVE 8            DPT1
PRICE        ALFRED     MANCHESTER    THAMES ROAD 4           DPT1
WILSON       RICHARD    MANCHESTER    ACACIA AVENUE 24        DPT3
```

8.7 Formation of groups

Input:

SAM file PERS.ACCNT

Output:

SAM file PERS.REPORT

Printout of the input file PERS.ACCNT

```
2002 JANUARY   $ 83000
2002 JANUARY   $  4000
2002 JANUARY   $ 17600
2002 JANUARY   $ 12110
2002 FEBRUARY  $ 16900
2002 FEBRUARY  $ 43000
2002 FEBRUARY  $ 78000
2002 APRIL     $ 93500
2002 APRIL     $ 26000
2002 JULY      $ 11450
2002 JULY      $ 98000
2002 JULY      $ 40500
2002 JULY      $ 13000
2002 SEPTEMBER $ 32500
2002 SEPTEMBER $ 72500
2002 DECEMBER  $ 21500
2002 DECEMBER  $ 73000
2002 DECEMBER  $ 33000
2002 DECEMBER  $ 43000
2003 JANUARY   $ 81000
2003 JANUARY   $ 92500
2003 JANUARY   $ 33300
2003 FEBRUARY  $ 99000
2003 FEBRUARY  $ 93000
2003 MARCH     $ 14000
2003 MARCH     $ 86700
2003 MARCH     $ 36400
2003 MAY       $ 23000
2003 MAY       $ 23000
2003 MAY       $ 65500
2003 JUNE      $ 93000
2003 JUNE      $  8400
2003 OCTOBER   $ 95000
2003 OCTOBER   $  7100
2003 DECEMBER  $ 89300
2003 DECEMBER  $  3000
2003 DECEMBER  $ 43000
```

Tracer listing:

```
/ADD-FILE-LINK FILE-NAME=PERS.ACCNT,- ---------------------------------  (1)
/              LINK-NAME=PCIN
/CREATE-FILE FILE-NAME=PERS.REPORT -------------------------------------  (2)
/ADD-FILE-LINK FILE-NAME=PERS.REPORT,-
/              ACCESS-METHOD=*SAM,-
/              LINK-NAME=PCOUT
/START-PERCON ----------------------------------------------------------  (3)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//SET-GROUP-ATTRIBUTES GROUP-LEVEL=1,- ---------------------------------  (4)
//                     GROUP-CONTROL=*FIELD(POSITION=5,-
//                                          LENGTH=4),-
//                     GROUP-HEADER=(*SPACING(LINES=3),-
//                                   'ANNUAL REPORT'(OUTPUT-POSITION=2),-
//                                   *FIELD(INPUT-POSITION=5,-
//                                          INPUT-LENGTH=4,-
//                                          INPUT-FORMAT=*CHARACTER,-
//                                          OUTPUT-POSITION=16),-
//                                   *SPACING(LINES=1)),-
//                     GROUP-TRAILER=(*SPACING(LINES=2),-
//                                   'YEARLY TOTAL FOR'(OUTPUT-POSITION=2),-
//                                   *FIELD(INPUT-POSITION=5,-
//                                          INPUT-LENGTH=4,-
//                                          INPUT-FORMAT=*CHARACTER,-
//                                          OUTPUT-POSITION=18),-
//                                   ':  $ '(OUTPUT-POSITION=22),-
//                                   *SUM-FIELD(INPUT-POSITION=23,-
//                                         INPUT-LENGTH=5,-
//                                         INPUT-FORMAT=*ZONED-DECIMAL,-
//                                         OUTPUT-POSITION=27,-
//                                         OUTPUT-LENGTH=6,-
//                                         OUTPUT-FORMAT=*ZONED-DECIMAL))
```

(1) The input file PERS.ACCNT is assigned with the standard link name PCIN by means of an ADD-FILE-LINK command.

(2) The output file PERS.REPORT is created via a CREATE-FILE-COMMAND and assigned with the standard link name PCOUT by means of an ADD-FILE-LINK command.

(3) PERCON is called.

(4) This SET-GROUP-ATTRIBUTES statement defines group level 1, the year as the grouping criterion, and the group header and trailer for this group.

```
//SET-GROUP-ATTRIBUTES GROUP-LEVEL=2,- ——————————————————————————————————————  (5)
//                     GROUP-CONTROL=*FIELD(POSITION=10,-
//                                          LENGTH=9),-
//                     GROUP-HEADER=(*SPACING(LINES=1),-
//                                   'MONTH:'(OUTPUT-POSITION=4),-
//                                   *FIELD(INPUT-POSITION=10,-
//                                          INPUT-LENGTH=9,-
//                                          INPUT-FORMAT=*CHARACTER,-
//                                          OUTPUT-POSITION=11),-
//                                   *SPACING(LINES=1)),-
//                     GROUP-TRAILER=(*SPACING(LINES=1),-
//                                   'MONTHLY TOTAL FOR'(OUTPUT-POSITION=2),-
//                                    *FIELD(INPUT-POSITION=10,-
//                                           INPUT-LENGTH=9,-
//                                           INPUT-FORMAT=*CHARACTER,-
//                                           OUTPUT-POSITION=18),-
//                                    ':  $ '(OUTPUT-POSITION=27),-
//                                    *SUM-FIELD(INPUT-POSITION=23,-
//                                           INPUT-LENGTH=5,-
//                                           INPUT-FORMAT=*ZONED-DECIMAL,-
//                                           OUTPUT-POSITION=32,-
//                                           OUTPUT-LENGTH=6,-
//                                           OUTPUT-FORMAT=*ZONED-DECIMAL))
//END —————————————————————————————————————————————————————————————————————————  (6)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=PCIN (FILE=PERS.ACCNT):   37 (7)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=PCOUT(FILE=PERS.REPORT): 37
%  PER0031 PERCON TERMINATED NORMALLY ———————————————————————————————————————  (8)
```

(5) This SET-GROUP-ATTRIBUTES statement defines group level 2, the month as the grouping criterion, and the group header and trailer for this group.

(6) The END statement starts the transfer operation and terminates PERCON.

(7) PERCON messages: the number of records transferred per file is output. The files are listed with their link and file names.

(8) PERCON was terminated normally.

Printout of the output file PERS.REPORT

```
                                                     PAGE     1
  ANNUAL REPORT 2002

    MONTH: JANUARY
 2002 JANUARY   $  3000
 2002 JANUARY   $  400
 2002 JANUARY   $ 17600
 2002 JANUARY   $ 12110
  MONTHLY TOTAL FOR JANUARY:   $ 116710

    MONTH: FEBRUARY
 2002 FEBRUARY  $ 16900
 2002 FEBRUARY  $ 43000
 2002 FEBRUARY  $ 78000

  MONTHLY TOTAL FOR FEBRUARY:  $ 137900

    MONTH: APRIL
 2002 APRIL     $ 93500
 2002 APRIL     $ 26000

  MONTHLY TOTAL FOR APRIL  :   $ 119500

    MONTH: JULY
 2002 JULY      $ 11450
 2002 JULY      $ 98000
 2002 JULY      $ 40500
 2002 JULY      $ 13000

  MONTHLY TOTAL FOR JULY   :   $ 162950

    MONTH: SEPTEMBER
 2002 SEPTEMBER $ 32500
 2002 SEPTEMBER $ 72500

  MONTHLY TOTAL FOR SEPTEMBER: $ 105000

    MONTH: DECEMBER
 2002 DECEMBER  $ 21500
 2002 DECEMBER  $ 73000
 2002 DECEMBER  $ 33000
 2002 DECEMBER  $ 43000

  MONTHLY TOTAL FOR DECEMBER : $ 170500

  YEARLY TOTAL FOR 2002:  $ 812560
  ANNUAL REPORT 2003

    MONTH: JANUARY
 2003 JANUARY   $ 81000
 2003 JANUARY   $ 92500
 2003 JANUARY   $ 33300

  MONTHLY TOTAL FOR JANUARY:   $ 206800
```

```
                                                                   PAGE 2
     MONTH: FEBRUARY
  2003 FEBRUARY  $ 99000
  2003 FEBRUARY  $ 93000

   MONTHLY TOTAL FOR FEBRUARY : $ 192000

     MONTH: MARCH
  2003 MARCH     $ 14000
  2003 MARCH     $ 86700
  2003 MARCH     $ 36400

   MONTHLY TOTAL FOR MARCH  :   $ 137100

     MONAT: MAY
  2003 MAY       $ 23000
  2003 MAY       $ 23000
  2003 MAY       $ 65500

   MONTHLY TOTAL FOR MAY    :   $ 111500

     MONTH: JUNE
  2003 JUNE      $ 93000
  2003 JUNE      $  8400

   MONTHLY TOTAL FOR JUNE   :   $ 101400

     MONTH: OCTOBER
  2003 OCTOBER   $ 95000
  2003 OCTOBER   $  7100

   MONTHLY TOTAL FOR OCTOBER:   $ 102100

     MONTH: DECEMBER
  2003 DECEMBER  $ 89300
  2003 DECEMBER  $  3000
  2003 DECEMBER  $ 43000

   MONTHLY TOTAL FOR DECEMBER: $ 135300

   YEARLY TOTAL FOR 2003: $ 986200
```

8.8 Output of a defined area of a tape

Input:

Tape with the volume serial number BD0002

Output:

– Output of the first block of the tape to SYSOUT (in character mode)

– Output of the first block to SYSLST (in hexadecimal mode)

Tracer listing:

```
/START-PERCON ─────────────────────────────────────────────────────────  (1)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-TAPE VOLUME=BD0002 ──────────────────────────────────────  (2)
%  DMSODE3 TAPE WITH VSN BD0002 FOR FILE :catid:$userid.PERCON.
           TPWORK.D.2076.VS0001 IS MOUNTED ON DEVICE AF
//ASSIGN-OUTPUT-FILE FILE=*SYSOUT,- ────────────────────────────────────  (3)
//                   LINK-NAME=OUT1
//ASSIGN-OUTPUT-FILE FILE=*SYSLST,- ────────────────────────────────────  (4)
//                   LINK-NAME=OUT2
//SET-PAGE-LAYOUT OUTPUT-LINK-NAME=OUT1,- ──────────────────────────────  (5)
//                OUTPUT-FORMAT=*CHARACTER,-
//                LINE-SIZE=65
//CHANGE-INPUT-TAPEPOSITION DIRECTION=*FORWARD(DESTINATION=*TAPE-MARKS-   (6)
//                                            (TAPE-MARKS=1))
```

(1) PERCON is called.

(2) The input tape with the VSN BD0002 is assigned.

(3) SYSOUT is assigned as the output medium with the link name OUT1.

(4) SYSLST is assigned as the output medium with the link name OUT2.

(5) Character mode and the number of characters to be edited per line for output to SYSOUT are defined.

(6) This statement means that labels are skipped.

```
//START-TAPE-PROCESSING INPUT-RANGE=*BLOCKS(BLOCKS=1) ——————————————————  (7)
         TMCNT: 001       BLOCK: 00000001 ——————————————————————————————  (8)
(00000)  1036008600000000 * BELL        * JOHN       * MANCHESTER  * BOLSO
(00065)  VER STREET 4     * DPT1 *008600000001 * BOTHAM      * NORMAN
(00130)  * MANCHESTER  * TOWER AVENUE 10       * DPT2 *008600000002 * FINN
(00195)          * SUSANNA    * NORWICH     * ROSE STREET 11        * DPT2
(0260)   *008600000003 * GREENE      * WALTER     * BURY        * SINCLAI
(00325)  R CRESCENT 7   * DPT1 *008600000004 * KING        * MONICA     *
(00390)  FALMOUTH    * INMAN SQUARE 61       * DPT3 *008600000005 * LAKER
(00455)        * ERICA      * MANCHESTER  * BANK DRIVE 8          * DPT1 *
(00520)  008600000006 * PRICE       * ALFRED     * MANCHESTER  * THAMES RO
(00585)  AD 4         * DPT1 *008600000007 * WILSON      * RICHARD    * MA
(00650)  NCHESTER  * ACACIA AVENUE 24      * DPT3 *008600000008 * BATES
(00715)      * FRANK      * HULL        * WOOD STREET 29        * DPT4 *00
(00780)  8600000009 * HARRISON    * MAURICE    * MANCHESTER  * SINCLAIR ST
(00845)  REET 49    * DPT4 *008600000010 * ROYCE       * ROBERT     * MANC
(00910)  HESTER  * LION'S GATE 74        * DPT4 *008600000011 * SIMPSON
(00975)    * ELLEN      * KINGS LYNN * WILLOUGHBY STREET 8   * DPT4 *
//END ——————————————————————————————————————————————————————————————————  (9)
%  PER0029 NUMBER OF PROCESSED BLOCKS FOR LINK=PCIN:   8(INCL.  7 LABELS) (10)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=OUT1:  8(INCL.  7 LABELS)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=OUT2:  8(INCL.  7 LABELS)
%  PER0031 PERCON TERMINATED NORMALLY ————————————————————————————————— (11)
```

(7) A block of the input tape is output to the assigned output media.

(8) Output to SYSOUT:

The current values of the tape mark counter TMCNT and the block counter are output prior to the data. The current values of the byte counter BYTCNT precede the data lines.

(9) The END statement terminates PERCON.

(10) PERCON messages specifying the number of blocks transferred for each input/output medium.

(11) PERCON is terminated normally.

Output to printer

```
PERCON           DATE 2000-10-11 TIME 13:45:32                                                                      PAGE     1
TMC BLKCNT   BYTCNT
                      1 0 3 6 0 0 8 6 0 0  0 0 0 0 0 0   *   B  E L L                  *   J O H N                *   M A N C
001 00000001 (00000)  F1F0F3F6F0F0F8F6F0F0 F0F0F0F0F0F0405C40C2 C5D3D340404040404040 405C40D1D6C8D5404040 404040405C40D4C1D5C3
                      H E S T E R     *    B O L S O V E R   S  T R E E T   4            *   D P T     *  0 0 8 6 0 0 0 0 0 0
             (00050)  C8C5E2E3C5D940405C40 C2D6D3E2D6E5C5D940E2 E3D9C5C5E340F4404040 40405C40C4D7E340405C F0F0F8F6F0F0F0F0F0F0
                      0 1   *   B O T H A  M             *   N  O R M A N            *   M A N C H E S T  E R     *   T O W E
             (00100)  F0F1405C40C2D6E3C8C1 D44040404040405C40D5 D6D9D4C1D54040404040 5C40D4C1D5C3C8C5E2E3 C5D940405C40E3D6E6C5
                      R   A V E N U E   1  0               *    D P T     * 0 0 8 6  0 0 0 0 0 0 0 2   *    F I N N
             (00150)  D940C1E5C5D5E4C540F1 F0404040404040405C40 C4D7E340405CF0F0F8F6 F0F0F0F0F0F0F0F2405C 40C6C9D5D54040404040
                            *   S U S A N  N A         *   N O  R W I C H            *   R O S E   S T R  E E T   1 1
             (00200)  4040405C40E2E4E2C1D5 D5C1404040405C40D5D6 D9E6C9C3C84040404040 5C40D9D6E2C540E2E3D9 C5C5E340F1F140404040
                              *   D P T      * 0 0 8 6 0 0 0 0  0 0 0 3   *   G R E  E N E             *    W A L T E R
             (00250)  404040405C40C4D7E340 405CF0F0F8F6F0F0F0F0 F0F0F0F3405C40C7D9C5 C5D5C54040404040405C 40E6C1D3E3C5D9404040
                          *   B U R Y                  *   S I  N C L A I R   C R E  S C E N T   7        *   D P T     * 0 0
             (00300)  40405C40C2E4D9E84040 4040404040405C40E2C9 D5C3D3C1C9D940C3D9C5 E2C3C5D5E340F7404040 5C40C4D7E340405CF0F0
                      8 6 0 0 0 0 0 0 0 4    *   K I N G                  *   M O N  I C A           *    F A L M O U T H
             (00350)  F8F6F0F0F0F0F0F0F0F4 405C40D2C9D5C7404040 40404040405C40D4D6D5 C9C3C140404040405C40 C6C1D3D4D6E4E3C84040
                          *   I N M A N    S Q U A R E   6 1                *   D P  T     * 0 0 8 6 0 0  0 0 0 0 0 5   *   L
             (00400)  40405C40C9D5D4C1D540 E2D8E4C1D9C540F6F140 4040404040405C40C4D7 E340405CF0F0F8F6F0F0 F0F0F0F0F0F5405C40D3
                      A K E R                *   E R I C A              *   M A N C  H E S T E R     *    B A N K   D R I V E
             (00450)  C1D2C5D9404040404040 405C40C5D9C9C3C14040 404040405C40D4C1D5C3 C8C5E2E3C5D940405C40 C2C1D5D240C4D9C9E5C5
                        8                      *   D P T     *  0 0 8 6 0 0 0 0 0 0  0 6   *   P R I C E                *   A
             (00500)  40F84040404040404040 40405C40C4D7E340405C F0F0F8F6F0F0F0F0F0F0 F0F6405C40D7D9C9C3C5 404040404040405C40C1
                      L F R E D            *   M A N C H E S T  E R     *   T H A M  E S   R O A D   4                    *
             (00550)  D3C6D9C5C44040404040 5C40D4C1D5C3C8C5E2E3 C5D940405C40E3C8C1D4 C5E240D9D6C1C440F440 40404040404040405C40
                      D P T     * 0 0 8 6  0 0 0 0 0 0 0 7   *    W I L S O N              *   R I C H A  R D         *   M A
             (00600)  C4D7E340405CF0F0F8F6 F0F0F0F0F0F0F0F7405C 40E6C9D3E2D6D5404040 4040405C40D9C9C3C8C1 D9C4404040405C40D4C1
                      N C H E S T E R      *   A C A C I A   A  V E N U E   2 4              *   D P T      * 0 0 8 6 0 0 0 0
             (00650)  D5C3C8C5E2E3C5D94040 5C40C1C3C1C3C9C140C1 E5C5D5E4C540F2F44040 404040405C40C4D7E340 405CF0F0F8F6F0F0F0F0
                      0 0 0 8   *   B A T  E S               *    F R A N K              *   H U L L                  *   W O
             (00700)  F0F0F0F8405C40C2C1E3 C5E2404040404040405C 40C6D9C1D5D240404040 40405C40C8E4D3D34040 4040404040405C40E6D6
                      O D   S T R E E T    2 9                  *   D P T     * 0 0  8 6 0 0 0 0 0 0 0 9    *   H A R R I S O
             (00750)  D6C440E2E3D9C5C5E340 F2F94040404040404040 5C40C4D7E340405CF0F0 F8F6F0F0F0F0F0F0F0F9 405C40C8C1D9D9C9E2D6
                      N         *   M A U  R I C E         *    M A N C H E S T E R      *   S I N C L A  I R   S T R E E T
             (00800)  D5404040405C40D4C1E4 D9C9C3C5404040405C40 D4C1D5C3C8C5E2E3C5D9 40405C40E2C9D5C3D3C1 C9D940E2E3D9C5C5E340
                      1 4 9       *   D P  T     * 0 0 8 6 0 0  0 0 0 0 1 0   *   R  O Y C E                *   R O B E R T
             (00850)  F1F4F94040405C40C4D7 E340405CF0F0F8F6F0F0 F0F0F0F0F1F0405C40D9 D6E8C3C5404040404040 405C40D9D6C2C5D9E340
                              *   M A N C  H E S T E R     *    L I O N ' S   G A T  E   7 4                  *   D P T     *
             (00900)  404040405C40D4C1D5C3 C8C5E2E3C5D940405C40 D3C9D6D57DE240C7C1E3 C540F7F4404040404040 40405C40C4D7E340405C
                      0 0 8 6 0 0 0 0 0 0  1 1   *   S I M P S  O N           *   E  L L E N              *   K I N G S   L Y
             (00950)  F0F0F8F6F0F0F0F0F0F0 F1F1405C40E2C9D4D7E2 D6D540404040405C40C5 D3D3C5D5404040404040 5C40D2C9D5C7E240D3E8
                      N     *   W I L L  O U G H B Y   S T R  E E T   8       *    D P T     *
             (01000)  D540405C40E6C9D3D3 D6E4C7C8C2E840E2E3D9 C5C5E340F84040405C40 C4D7E340405C
```

8.9 Duplicating a tape

Input:

Tape with the volume serial number BD0002

Output:

Tape with the volume serial number C5414A
Tape with the volume serial number C5426A

Tracer listing:

```
/START-PERCON ——————————————————————————————————————————————————————————  (1)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-TAPE VOLUME=BD0002 ————————————————————————————————————————  (2)
%  DMS0DE3 TAPE WITH VSN BD0002 FOR FILE :catid:$userid.PERCON.
           TPWORK.D.2076.VS0001IS MOUNTED ON DEVICE AF
//ASSIGN-OUTPUT-TAPE VOLUME=C5414A,- ————————————————————————————————————  (3)
//                   LINK-NAME=OUT1
%  DMS0DE3 TAPE WITH VSN C5414A FOR FILE :catid:$userid.PERCON.
           TPWORK.D.2076.VS0002 IS MOUNTED ON DEVICE AG
//ASSIGN-OUTPUT-TAPE VOLUME=C5426A,- ————————————————————————————————————  (4)
//                   LINK-NAME=OUT2
%  DMS0DE3 TAPE WITH VSN C5426A FOR FILE :catid:$userid.PERCON.
           TPWORK.D.2076.VS0003 IS MOUNTED ON DEVICE AH
//END ———————————————————————————————————————————————————————————————————  (5)
%  PER0029 NUMBER OF PROCESSED BLOCKS FOR LINK=PCIN:  8 (INCL.  7 LABELS) (6)
%  PER0029 NUMBER OF PROCESSED BLOCKS FOR LINK=OUT1:  8 (INCL.  7 LABELS)
%  PER0029 NUMBER OF PROCESSED BLOCKS FOR LINK=OUT2:  8 (INCL.  7 LABELS)
%  PER0031 PERCON TERMINATED NORMALLY ————————————————————————————————————  (7)
```

(1) PERCON is called.

(2) The input tape with the VSN BD0002 is assigned.

(3) The output tape with the VSN C5414A and the link name OUT1 is assigned.

(4) The output tape with the VSN C5426A and the link name OUT2 is assigned.

(5) The END statement starts the transfer operation and terminates PERCON.

(6) PERCON messages: the number of blocks transferred per tape is output.

(7) PERCON was terminated normally.

8.10 Example with a COBOL main program

PERCON is to be called as a subprogram by a COBOL main program.
The statements are supplied in the program.

Source listing

```
↓ column 7
 IDENTIFICATION DIVISION.
 PROGRAM-ID.  PERCONCO.
*        EXAMPLE: CALLING PERCON FROM A COBOL-PROGRAM.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SPECIAL-NAMES.
     TERMINAL IS SCREEN.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01 PARAM. ------------------------------------------------------------ (1)
   02 CHECK-INFORMATION         PIC S9(5) COMP SYNC VALUE 132. ------- (2)
   02 STATEMENT-1 SYNC.
     03 LENGTH-OF-REC           PIC S99 COMP VALUE 45. --------------- (3)
     03 FILLER                  PIC S99 COMP VALUE 0.
     03 CONTENTS-OF-REC         PIC X(41) VALUE ---------------------- (4)
                                "MODIFY-PERCON-OPTIONS SYSOUT-LOG
-                               "GING=*ALL".
   02 STATEMENT-2 SYNC.
     03 LENGTH-OF-REC           PIC S99 COMP VALUE 51.
     03 FILLER                  PIC S99 COMP VALUE 0.
     03 CONTENTS-OF-REC         PIC X(47) VALUE
                                "ASSIGN-INPUT-FILE FILE=*DISK-FIL
-                               "E(NAME=PERS.TAB)".
   02 STATEMENT-3 SYNC.
     03 LENGTH-OF-REC           PIC S99 COMP VALUE 40.
     03 FILLER                  PIC S99 COMP VALUE 0.
     03 CONTENTS-OF-REC         PIC X(36) VALUE
                                "ASSIGN-OUTPUT-FILE LINK-NAME=OUT
-                               "PUT".
```

(1) Symbolic address of the parameter area.

(2) Byte 4 contains the identifier indicating the way in which the statements are transferred. X’84’: the PERCON statements are transferred by the main program in the form of variable-length records.

(3) The record length field of the statement is defined by means of LENGTH-OF-REC and FILLER.

(4) The field for the record contents of the statement is defined by means of CONTENTS-OF-REC.

```
     02 STATEMENT-4 SYNC.
       03 LENGTH-OF-REC          PIC S99 COMP VALUE 7.
       03 FILLER                 PIC S99 COMP VALUE 0.
       03 CONTENTS-OF-REC        PIC X(3) VALUE "END".
   01 RETCODE. ----------------------------------------------------------  (5)
     02 DMS-CODE                 PIC S9(5) COMP SYNC. -----------------  (6)
     02 MESSAGE-ID -------------------------------------------------------  (7)
        03 BYTE-1-2              PIC 9(4) COMP.
        03 BYTEAN-1-2            REDEFINES BYTE-1-2.
           04 BYTEAN-1           PIC X.
           04 FILLER             PIC X.
        03 BYTES-REST            PIC X(10).
   01 RESERVED-FIELDS
     02 FILLER                   PIC S9(5) COMP.
     02 FILLER                   PIC S9(5) COMP.
   77 AUXILIARY-FI               PIC 9(5).
   77 BIT-2                      PIC 9.
   77 FOURTEEN-RIGHT             PIC 9(5) VALUE 16384.
   PROCEDURE DIVISION.
   CALL-PERCON SECTION.
   AR-1.
       CALL "PERCONU" USING PARAM RETCODE RESERVED. ---------------------  (8)
       SUBTRACT 32768 FROM BYTE-1-2.
       IF BYTEAN-1 NOT < SPACE
       MOVE 1 TO BIT-2
       ELSE MOVE ZERO TO BIT-2.
       IF BIT-2 = 0
          DISPLAY "PERCON RUN SUCCESSFUL" UPON SCREEN.
       ELSE
          DISPLAY "PERCON RUN SUCCESSFUL" UPON SCREEN.
       STOP RUN.
```

(5) Symbolic address of the area for return information.

(6) These four bytes are reserved for the last DMS message.

(7) This area is reserved for the PERCON messages.

(8) PERCON is called. PERCONU is the entry point. The following data is transferred:

- the data group PARAM, consisting of a field containing the code for the call level, followed by the control statements;
- the data group RETCODE, consisting of 4 fields which can accommodate the return code;
- the data group RESERVED-FIELDS, consisting of 2 fields for subsequent extensions.

Compiling, linking and calling the program (tracer listing)

```
/DELETE-SYSTEM-FILE FILE-NAME=*OMF
/START-COBOL2000-COMPILER SOURCE=COB.TEST ──────────────────────────────  (9)
%  BLS0500 PROGRAM 'COBOL2000-BC', VERSION '01.4A02' OF '2006-06-16' LOADED
%    CBL9000 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006
           ALL RIGHTS RESERVED
%  CBL9017 COMPILATION INITIATED, VERSION IS  V01.4A02
%  CBL9095 SAVLST FILE :cat:$user.OPTLST.COBOL.PERCONCO CREATED AND CLOSED
%  CBL9095 SAVLST FILE :cat:$user.SRCLST.COBOL.PERCONCO CREATED AND CLOSED
%  CBL9095 SAVLST FILE :cat:$user.ERRFIL.COBOL.PERCONCO CREATED AND CLOSED
%  CBL9097 COMPILATION COMPLETED WITHOUT ERRORS
%  CBL9004 COMPILATION OF PERCONCO USED  0.2295 CPU SECONDS
/START-BINDER ───────────────────────────────────────────────────────── (10)
//START-LLM-CREATION INTERNAL-NAME=COB.PROG ─────────────────────────── (11)
//INCLUDE-MODULES LIBRARY=*OMF,ELEMENT=*ALL
//RESOLVE-BY-AUTOLINK LIBRARY=$.SYSLNK.PERCON.029 ───────────────────── (12)
//RESOLVE-BY-AUTOLINK LIBRARY=$.SYSLNK.CRTE
//SAVE-LLM LIBRARY=COB.PROG
%  BND3101 SOME EXTERNAL REFERENCES UNRESOLVED
%  BND3102 SOME WEAK EXTERNS UNRESOLVED
%  BND1501 LLM FORMAT: '1'
//END
%  BND1101 BINDER NORMALLY TERMINATED. SEVERITY CLASS: 'UNRESOLVED EXTERNAL'
```

(9) The COBOL compiler is called. The SOURCE parameter assigns the file COB.TEST as the input file.

(10) The BINDER is called.

(11) The name of the program is defined.

(12) The object module library SYSLNK.PERCON.029 and the runtime system SYSLNK.CRTE are assigned.

```
/CREATE-FILE FILE-NAME=TEST
/ADD-FILE-LINK LINK-NAME=OUTPUT,- ------------------------------------ (13)
/              FILE-NAME=TEST,-
/              ACCESS-METHOD=*ISAM,-
/              SUPPORT=*DISK(ISAM-ATTRIBUTES=(KEY-POSITION=5,KEY-LENGTH=8)),-
/              RECORD-FORMAT=*VARIABLE
/START-EXECUTABLE-PROGRAM FROM-FILE=*LIBRARY-ELEMENT(LIBRARY=COB.PROG,-
/                                   ELEMENT-OR-SYMBOL=COBPROG1) ------- (14)
ASSIGN-INPUT-FILE FILE=DISK-FILE(NAME=PERS.TAB) ---------------------- (15)
ASSIGN-OUTPUT-FILE LINK-NAME=OUTPUT
END
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=PCIN (FILE=PERS.TAB): 12 (16)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=OUTPUT (FILE=TEST):  12
%  PER0031 PERCON TERMINATED NORMALLY
PERCON RUN SUCCESSFUL ------------------------------------------------ (17)
```

(13) The output file TEST is assigned with the link name OUTPUT and its file attributes.

(14) The program which has just been linked is loaded and started.

(15) Subsequent statements are made available by the COBOL program.

(16) Messages from PERCON as a subprogram.

(17) Output from the main program.

8.11 Example with an Assembler main program

PERCON is to be called as a subprogram by an Assembler main program.

The statements are read from SYSDTA.

Source listing

```
PERUP    START
PERUP    AMODE ANY
PERUP    RMODE ANY
         BALR  8,0
         USING *,8
         LA    1,ATAB                 ADDRESS OF PARAMETER LIST ──────  (1)
         LA    13,SAV                 ADDRESS OF SAVE AREA ───────────  (2)
         L     15,=V(PERCONU) ────────────────────────────────────────  (3)
         BALR  14,15                  BRANCH TO SUBPROGRAM ───────────  (4)
         TERM
         SPACE 3
SAV      DS    20F                    SAVE AREA ──────────────────────  (5)
         SPACE
ATAB     DC    A(PARAM)               PARAMETER LIST ─────────────────  (6)
         DC    A(RETCODE) ────────────────────────────────────────────  (7)
         DC    X'80000000' ───────────────────────────────────────────  (8)
         SPACE
RETCODE  DC    F'0' ──────────────────────────────────────────────────  (9)
         DC    12X'00' ───────────────────────────────────────────────  (10)
         SPACE
PARAM    DC    X'00000080'            STATEMENTS FROM SYSDTA ─────────  (11)
         END   PERUP
```

(1) Register 1 is loaded with the symbolic address ATAB of the address list.

(2) Register 13 is loaded with the symbolic address SAV of the save area.

(3) Register 15 is loaded with a V-type constant specifying the address of the entry point PERCONU.

(4) Branch to the subprogram; register 14 is loaded with the return address.

(5) 20 words are reserved for the save area.

Definition of the address list:

(6) Address constant PARAM indicating the parameter area

(7) Address constant RETCODE indicating the area for return information

(8) End criterion for the address list

Area for return information:

(9) This area is reserved for the message code of the last DMS message issued.

(10) All PERCON messages from the range PER0000 thru PER0095 issued during program execution are registered in this area.

(11) Parameter area
Byte 4 contains X’00’; this means that the statements are read from SYSDTA in SDF format.

Compiling, linking and calling the program (tracer listing)

```
/DELETE-SYSTEM-FILE FILE-NAME=*OMF
/START-ASSEMBH
%  BLS0500 PROGRAM 'ASSEMBH', VERSION '1.2C00' OF '2002-03-06' LOADED
%  BLS0552 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2002.
           ALL RIGHTS RESERVED
%  BLS0519 PROGRAM 'ASSEMBH' LOADED
%  ASS6010 V01.2C00 OF BS2000 ASSTRAN  READY
//COMPILE SOURCE=ASS.TEST,MACRO-LIBRARY=MACROLIB
%  ASS6011 ASSEMBLY TIME: 143 MSEC
%  ASS6018 0 FLAGS, 0 PRIVILEGED FLAGS, 0 MNOTES
%  ASS6019 HIGHEST ERROR-WEIGHT: NO ERRORS
/START-BINDER
//START-LLM-CREATION INTERNAL-NAME=ASSPROG1
//INCLUDE-MODULES LIBRARY=*OMF,ELEMENT=*ALL
//RESOLVE-BY-AUTOLINK LIBRARY=$.SYSLNK.PERCON.029 ———————————————————— (12)
//SAVE-LLM LIBRARY=ASS.PROG ——————————————————————————————————————————— (13)
%  BND1501 LLM FORMAT: '1'
//END
%  BND1101 BINDER NORMALLY TERMINATED. SEVERITY CLASS: 'OK'
/ADD-FILE-LINK FILE-NAME=PERS.FAL,- ———————————————————————————————————— (14)
/              ACCESS-METHOD=*SAM,-
/              LINK-NAME=OUT
/START-EXECUTABLE-PROGRAM FROM-FILE=*LIBRARY-ELEMENT(LIBRARY=COB.PROG,-
/                                   ELEMENT-OR-SYMBOL=COBPROG1)—————— (15)
//MODIFY-PERCON-OPTIONS SYSOUT-LOGGING=*ALL ——————————————————————————— (16)
//ASSIGN-INPUT-FILE FILE=*DISK-FILE(NAME=PERS.DPT) ———————————————————— (17)
//ASSIGN-OUTPUT-FILE FILE=*DISK-FILE(NAME=PERS.FAL),- ————————————————— (18)
//                   LINK-NAME=OUT
//SELECT-INPUT-RECORDS CONDITION=((38,8)='FALMOUTH') —————————————————— (19)
//END ————————————————————————————————————————————————————————————————— (20)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=PCIN (FILE=PERS.DPT):   12
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK=OUT (FILE=PERS.FAL):     1
%  PER0031 PERCON TERMINATED NORMALLY ————————————————————————————————— (21)
```

(12) The object module library OMLPERCON is assigned for linkage.

(13) The program ASSPROG1 which has been assembled and linked is written to the library ASS.PROG.

(14) The output file PERS.FAL is assigned with the link name OUT and its file attributes.

(15) The program ASSPROG1 contained in the library ASS.PROG is called.

(16) The branch to the subprogram has been made.

Statements are expected.

The MODIFY-PERCON-OPTIONS statement controls the output of the messages to SYSOUT. All the messages are output in full to SYSOUT.

(17) The input file PERS.DPT is assigned.

(18) The output file PERS.FAL is linked to PERCON via the link name OUT.

(19) Any records containing C’FALMOUTH’ starting in column 38 are transferred to the output file.

(20) The END statement starts the transfer operation and terminates PERCON.

(21) PERCON is terminated normally.

Control then branches from the subprogram back to the main program.

8.12 Example with formatted numbers and counters per group

The data entered via SYSDTA is edited and output in the LIST file.

```
/CREATE-FILE FILE-NAME=LIST
/ADD-FILE-LINK LINK-NAME=PCOUT,FILE-NAME=LIST,ACCESS-METHOD=SAM ───────  (1)
/START-PERCON ───────────────────────────────────────────────────────────  (2)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-FILE FILE=*SYSDTA ───────────────────────────────────────  (3)
//ASSIGN-OUTPUT-FILE FILE=*DISK-FILE(NAME=LIST) ────────────────────────  (4)
//SET-RECORD-MAPPING OUTPUT-FIELDS=(*GROUP-COUNTER(LINK-NAME=PCOUT,- ──  (5)
//                                                 GROUP-LEVEL=1,-
//                                                 OUTPUT-POSITION=10),-
//                                 '/'(OUTPUT-POSITION=16),-
//                                 *FIELD(INPUT-POSITION=5,-
//                                        INPUT-LENGTH=10,-
//                                        OUTPUT-POSITION=22),-
//                                 '/'(OUTPUT-POSITION=35),-
//                                 *FIELD(INPUT-POSITION=15,-
//                                        INPUT-LENGTH=3,-
//                                        INPUT-FORMAT=*ZONED-DECIMAL,-
//                                        OUTPUT-POSITION=40,-
//                                        OUTPUT-FORMAT=' ZNZ'))
//SET-GROUP-ATTRIBUTES GROUP-TRAILER=(*SPACING(LINES=4),- ────────────  (6)
//                                 'Items:'(OUTPUT-POSITION=10),-
//                                 *GROUP-COUNTER(LINK-NAME=PCOUT,-
//                                                 GROUP-LEVEL=1,-
//                                                 OUTPUT-POSITION=25,-
//                                                 OUTPUT-FORMAT=' ZZZNZ'),-
//                                 *SPACING,-
//                                 'Total number:'(OUTPUT-POSITION=10),-
//                                 *SUM-FIELD(INPUT-POSITION=15,-
//                                            INPUT-LENGTH=3,-
//                                            INPUT-FORMAT=*ZONED-DECIMAL,-
//                                            OUTPUT-POSITION=25,-
//                                            OUTPUT-FORMAT=' ZZZNZ'))
//SET-PAGE-LAYOUT HEADER-LINE=*NONE ────────────────────────────────────  (7)
//END ──────────────────────────────────────────────────────────────────  (8)
*Suit     014 ──────────────────────────────────────────────────────────  (9)
*Pants    053
*Shirt    162
*Blouse   064
*Shoes    136
*/EOF
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK 'PCIN'  :           5
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK 'PCOUT' (FILE=LIST): 5
%  PER0031 PERCON TERMINATED NORMALLY ──────────────────────────────── (10)
```

(1) The output file LIST is assigned with the link name PCOUT.

(2) PERCON is called.

(3) Input is to be made via SYSDTA.

(4) The output file LIST is assigned.

(5) The group counter and certain fields of the input record are transferred to the output record by means of the SET-RECORD-MAPPING statement.

(6) The SET-GROUP-ATTRIBUTES statement defines the format and contents of the group trailer lines.

(7) The SET-PAGE-LAYOUT statement specifies that no header line is to be output.

(8) The END statement starts the transfer procedure and terminates PERCON.

(9) Input of the data via SYSDTA.

(10) PERCON is terminated normally.

Contents of the output file LIST:

The position scale is not contained in the file and refers only to the data, i.e. record length field and feed control characters are not taken into account.

```
    5    10   15   20   25   30   35   40   45   50   (Position scale)
----|----|----|----|----|----|----|----|----|----|

             1   /   Suit          /       14
             2   /   Pants         /       53
             3   /   Shirt         /      162
             4   /   Blouse        /       64
             5   /   Shoes         /      136


          Items       :        5
          Total number:      429
```

8.13 Special applications

8.13.1 Copying an FDDRL save tape

When copying FDDRL save tape sets, the user interfaces for label processing offered by FDDRL must be used.

```
/SET-TASKLIB LIBRARY=SYSLNK.FDDRL.160
/IMPORT-FILE SUPPORT=*TAPE(VOLUME=(IN1,IN2),DEVICE-TYPE=TAPE-C4,-
/                          FILE-NAME=FDDRL.V16.0A)
/ADD-FILE-LINK LINK-NAME=PCIN,-
/              FILE-NAME=FDDRL.V16.0A,-
/              ACCESS-METHOD=*BTAM
/CREATE-FILE FILE-NAME=FDDRL.V16.0A.COPY,-
/            SUPPORT=*TAPE(VOLUME=(OUT1,OUT2),DEVICE-TYPE=TAPE-C4)
/ADD-FILE-LINK LINK-NAME=PCOUT,-
/              FILE-NAME=FDDRL.V16.0A.COPY
/START-PERCON
//ASSIGN-INPUT-FILE FILE=*TAPE-FILE(NAME=FDDRL.V16.0A,-
//                        LABEL-EXIT=*MODULE(NAME=USERIN,-
//                        CONTROLLED-LABEL=(*HDR,*EOV,*END)))
//ASSIGN-OUTPUT-FILE FILE=*TAPE-FILE(NAME=FDDRL.V16.0A.COPY,-
//                         LABEL-EXIT=*MODULE(NAME=USEROUT,-
//                         CONTROLLED-LABEL=(*HDR,*EOV,*END)),-
//                         FILE-ATTRIBUTES=*INPUT-FILE)
//END
```

Individual FDDRL save tapes can be copied without these interface routines. This is subject to the condition that no more than one output volume is produced per input tape.

8.13.2 Copying a SLED file to tape

The following procedure is designed for K, NK2 and NK4 disks.

```
/IMPORT-FILE SUPPORT=*TAPE(VOLUME=IN,DEVICE-TYPE=T6250,-
/                          FILE-NAME=SLEDFILE)
/ADD-FILE-LINK LINK-NAME=PCIN,-
/              FILE-NAME=SLEDFILE,-
/              ACCESS-METHOD=*SAM,-
/              BLOCK-CONTROL-INFO=*BY-CATALOG,-
/              BUFFER-LENGTH=*BY-CATALOG,-
/              RECORD-FORMAT=*FIXED,RECORD-SIZE=4096
/CREATE-FILE FILE-NAME=SLEDTEST,-
/            SUPPORT=*PUBLIC-DISK(SPACE=*RELATIVE(PRIMARY-ALLOCATION=2000,-
/                                                 SECONDARY-ALLOCATION=500))
/ADD-FILE-LINK LINK-NAME=PCOUT,-
/              FILE-NAME=SLEDTEST,-
/              ACCESS-METHOD=*UPAM
/              BUFFER-LENGTH=*STD(SIZE=2)
/START-PERCON
//END
```

Further information about how to use SLEDFILEs in conjunction with other disk formats can be found in the “Introductory Guide to Systems Support” [3].

8.13.3 Selecting and updating records

Records can be selected and updated using the SET-RECORD-MAPPING statement.

In the case of output files, several ADD-FILE-LINK commands can be issued in the same open mode and with different link names. These link names are then used to assign the PERCON statements. In this way records are selected with SELECT-INPUT-RECORDS and updated with SET-RECORD-MAPPING. All other records remain unchanged unless they are updated in some other manner. There are certain differences when input and output file are identical or when a new output file has to be created.

Input file identical to output file

In the case of SAM or ISAM files, several ADD-FILE-LINK commands with OPEN-MODE=*UPDATE or OPEN-MODE=*INOUT and with different link names can be issued for the same file. When the input file and the output file are identical, the number of records in this file remains unchanged. If records are selected via several SELECT-INPUT-RECORDS statements and updated via several SET-RECORD-MAPPING statements, the records are overwritten as often as required. Only the last update version is retained.

In the case of **SAM files** opened with *OPEN-MODE=*UPDATE, the record length is not changed. Records which have been lengthened are shortened to their original length; records that have been truncated are extended to their original length by means of the filler character of the SET-RECORD-MAPPING statement.

```
/ADD-FILE-LINK FILE-NAME=FILE,-
/              LINK-NAME=PCIN1,-
/              OPEN-MODE=*UPDATE
/ADD-FILE-LINK FILE-NAME=FILE,-
/              LINK-NAME=PCOUT1,-
/              OPEN-MODE=*UPDATE
/ADD-FILE-LINK FILE-NAME=FILE,-
/              LINK-NAME=PCOUT2,-
/              OPEN-MODE=*UPDATE
/ADD-FILE-LINK FILE-NAME=FILE,-
/              LINK-NAME=PCOUT3,-
/              OPEN-MODE=*UPDATE
/START-PERCON
//ASSIGN-INPUT-FILE LINK-NAME=PCIN1
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT1
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=PCOUT1,-
//                     CONDITION=((5,1)='A')
```

```
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT1,-
//                   FILLER=*INPUT,-
//                   OUTPUT-FIELDS=C'X'(OUTPUT-POSITION=5) ------------  (1)
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT2
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=PCOUT2,-
//                     CONDITION=((5,1)='B')
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT2,-
//                   FILLER=*INPUT,-
//                   OUTPUT-FIELDS=C'Y'(OUTPUT-POSITION=5) ------------  (2)
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT3
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=PCOUT3,-
//                     CONDITION=((6,1)<='2')
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT3,-
//                   FILLER=*INPUT,-
//                   OUTPUT-FIELDS=C'9'(OUTPUT-POSITION=6) ------------  (3)
//END
```

Old file:	New file:
A0 *	A9
B1 *	B9
B2 *	B9
B3	Y3
A4	X4
C5	C5
B6	Y6

* Because of (3), updates resulting from (1) and (2) are not taken into account.

(1) In column 5, ’A’ is replaced by ’X’.

(2) In column 5, ’B’ is replaced by ’Y’.

(3) In column 6, all characters ≤ ’2’ are replaced by ’9’.

In the case of **ISAM files**, loops may occur inadvertently

– if the record key is changed

– if DUPLICATE-KEY=*YES is specified

– if the record which is output precedes the record which has been read in, since the write position defines the next read position.

Input file not identical to output file

With ISAM or UPAM files, several ADD-FILE-LINK commands with OPEN-MODE=*OUTPUT or OPEN-MODE=*OUTIN and with different link names can be issued for the same output file.

The number of output records can be reduced if no input records have been selected by means of the SELECT-INPUT-RECORDS statement. By the same token, it can be increased if input records have been selected using more than one SELECT-INPUT-RECORDS statement.

With ISAM files, if DUPLICATE-KEY=*YES has been specified, an input record can be split up into more than one output record.

```
/ADD-FILE-LINK FILE-NAME=INPUTFILE,-
/              LINK-NAME=PCIN1,-
/              OPEN-MODE=*INPUT
/CREATE-FILE FILE-NAME=OUTPUTFILE
/ADD-FILE-LINK FILE-NAME=OUTPUTFILE,-
/              LINK-NAME=PCOUT1,-
/              OPEN-MODE=*OUTPUT,-
/              ACCESS-METHOD=*ISAM(DUPLICATE-KEY=*YES)
/              ACCESS-METHOD=*ISAM,-
/              SUPPORT=*DISK(ISAM-ATTRIBUTES=(DUPLICATE-KEY=*YES))
/ADD-FILE-LINK FILE-NAME=OUTPUTFILE,-
/              LINK-NAME=PCOUT2,-
/              OPEN-MODE=*OUTPUT,-
/              ACCESS-METHOD=*ISAM(DUPLICATE-KEY=*YES)
/              SUPPORT=*DISK(ISAM-ATTRIBUTES=(DUPLICATE-KEY=*YES))
/ADD-FILE-LINK FILE-NAME=OUTPUTFILE,-
/              LINK-NAME=PCOUT3,-
/              OPEN-MODE=*OUTPUT,-
/              ACCESS-METHOD=*ISAM
/              SUPPORT=*DISK(ISAM-ATTRIBUTES=(DUPLICATE-KEY=*YES))

/START-PERCON
//ASSIGN-INPUT-FILE LINK-NAME=PCIN1
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT1,-
//                   FILE=*DISK-FILE(FILE-ATTRIBUTES=INPUT-FILE)
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=PCOUT1,-
//                     CONDITION=((13,1)='A')
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT1,-
//                   FILLER=*INPUT,-
//                   OUTPUT-FIELDS=C'X'(OUTPUT-POSITION=13)
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT2,-
//                   FILE=*DISK-FILE(FILE-ATTRIBUTES=INPUT-FILE)
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=PCOUT2,-
//                     CONDITION=((13,1)='B')
```

```
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT2,-
//                   FILLER=*INPUT,-
//                   OUTPUT-FIELDS=C'Y'(OUTPUT-POSITION=13)
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT3,-
//                   FILE=DISK-FILE(FILE-ATTRIBUTES=*INPUT-FILE)
//SELECT-INPUT-RECORDS OUTPUT-LINK-NAME=PCOUT3,-
//                     CONDITION=((14,1)<='2')
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT3,-
//                   FILLER=*INPUT,-
//                   OUTPUT-FIELDS=C'9'(OUTPUT-POSITION=14)
//END
```

Input file:		Output file:
00000001A0	(1)	00000001X0
00000002B1	(1)	00000001A9
00000003B2	(1)	00000002Y1
00000004B3		00000002B9
00000005A4		00000003Y2
00000006C5	(2)	00000003B9
00000007B6		00000004Y3
		00000005X4
		00000007Y6

(1) The records are selected by the first and third SELECT-INPUT-RECORDS statement or by the second and third such statements, for which reason they appear twice in the output file.

(2) This record is not selected by a SELECT-INPUT-RECORDS statement and therefore does not appear in the output file.

Splitting a record

If a record is split up into three shorter records, a buffer file is required. Since this buffer file is an ISAM file, a pseudo key is added (for all records '1' in column 5). When the buffer file is further processed, this key is ignored.

```
/ADD-FILE-LINK FILE-NAME=INPUTFILE,-
/              LINK-NAME=PCIN1,-
/              OPEN-MODE=*INPUT
/CREATE-FILE FILE-NAME=BUFFERFILE
/ADD-FILE-LINK FILE-NAME=BUFFERFILE,-
/              LINK-NAME=PCOUT1,-
/              OPEN-MODE=*OUTPUT,-
/              ACCESS-METHOD=*ISAM,-
/              SUPPORT=*DISK(ISAM-ATTRIBUTES=(DUPLICATE-KEY=*YES,-
/                                             KEY-LENGTH=1,-
/                                             KEY-POSITION=5))
/ADD-FILE-LINK FILE-NAME=BUFFERFILE,-
/              LINK-NAME=PCOUT2,-
/              OPEN-MODE=*OUTPUT,-
/              ACCESS-METHOD=*ISAM,-
/              SUPPORT=*DISK(ISAM-ATTRIBUTES=(DUPLICATE-KEY=*YES,-
/                                             KEY-LENGTH=1,-
/                                             KEY-POSITION=5))
/ADD-FILE-LINK FILE-NAME=BUFFERFILE,-
/              LINK-NAME=PCOUT3,-
/              OPEN-MODE=*OUTPUT,-
/              ACCESS-METHOD=*ISAM,-
/              SUPPORT=*DISK(ISAM-ATTRIBUTES=(DUPLICATE-KEY=*YES,-
/                                             KEY-LENGTH=1,-
/                                             KEY-POSITION=5))
/ADD-FILE-LINK FILE-NAME=BUFFERFILE,-
/              LINK-NAME=PCIN2
/CREATE-FILE FILE-NAME=OUTPUTFILE
/ADD-FILE-LINK FILE-NAME=OUTPUTFILE,-
/              LINK-NAME=PCOUT4,-
/              ACCESS-METHOD=*SAM
/START-PERCON
//ASSIGN-INPUT-FILE LINK-NAME=PCIN1
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT1
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT1,-
//                   OUTPUT-FIELDS=(C'1'(OUTPUT-POSITION=5),-
//                                  *FIELD(INPUT-POSITION=5,-
//                                         INPUT-LENGTH=2,-
//                                         OUTPUT-POSITION=6))
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT2
```

```
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT2,-
//                   OUTPUT-FIELDS=(C'1'(OUTPUT-POSITION=5),-
//                                  *FIELD(INPUT-POSITION=7,-
//                                         INPUT-LENGTH=2,-
//                                         OUTPUT-POSITION=6))
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT3
//SET-RECORD-MAPPING OUTPUT-LINK-NAME=PCOUT3,-
//                   OUTPUT-FIELDS=(C'1'(OUTPUT-POSITION=5),-
//                                  *FIELD(INPUT=POSITION=9,-
//                                         INPUT-LENGTH=2,-
//                                         OUTPUT-POSITION=6))
//START-CONVERSION
//ASSIGN-INPUT-FILE  LINK-NAME=PCIN2
//ASSIGN-OUTPUT-FILE LINK-NAME=PCOUT4
//SET-RECORD-MAPPING OUTPUT-FIELDS=*FIELD(INPUT-POSITION=6,-
//                                        INPUT-LENGTH=2,-
//                                        OUTPUT-POSITION=5)
//END
```

Input file:	Buffer file:	Output file:
S1S2S3	1S1	S1
S4S5S6	1S2	S2
S7S8S9	1S3	S3
	1S4	S4
	1S5	S5
	1S6	S6
	1S7	S7
	1S8	S8
	1S9	S9

8.14 Working with MF/MV sets

The three disk files FILE.01, FILE.02 and FILE.03 are to be saved to tape. Since they do not all fit on a single tape, two output volumes are readied.

The two volumes are declared as a tape set. This serves to avoid multiple mounting and positioning of the save volumes.

```
/CREATE-TAPE-SET TAPE-SET-NAME=SET1,VOLUME=(TAPEKA,BD0150) ─────────── (1)
/CREATE-FILE FILE-NAME=FILE.01.TAPE,- ──────────────────────────────── (2)
/            SUPPORT=*TAPE(VOLUME=*NO,DEVICE-TYPE=T6250)
/ADD-FILE-LINK LINK-NAME=PCOUT1,FILE-NAME=FILE.01.TAPE,-
/            ACCESS-METHOD=*SAM,-
/            SUPPORT=*TAPE(-
/            VOLUME-LIST=*BY-TAPE-SET(TAPE-SET-NAME=SET1),-
/            FILE-SEQUENCE=1)
/CREATE-FILE FILE-NAME=FILE.02.TAPE,- ──────────────────────────────── (3)
/            SUPPORT=*TAPE(VOLUME=*NO,DEVICE-TYPE=T6250)
/ADD-FILE-LINK LINK-NAME=PCOUT2,FILE-NAME=FILE.02.TAPE,-
/            ACCESS-METHOD=*SAM,-
/            SUPPORT=*TAPE(-
/            VOLUME-LIST=*BY-TAPE-SET(TAPE-SET-NAME=SET1),-
/            FILE-SEQUENCE=*NEW)
/CREATE-FILE FILE-NAME=FILE.03.TAPE,-
/            SUPPORT=*TAPE(VOLUME=*NO,DEVICE-TYPE=T6250)
/ADD-FILE-LINK LINK-NAME=PCOUT3,FILE-NAME=FILE.03.TAPE,-
/            ACCESS-METHOD=*SAM,-
/            SUPPORT=*TAPE(-
/            VOLUME-LIST=*BY-TAPE-SET(TAPE-SET-NAME=SET1),-
/            FILE-SEQUENCE=*NEW)
/START-PERCON ───────────────────────────────────────────────────────── (4)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-FILE FILE=*DISK-FILE(NAME=FILE.01) ───────────────────── (5)
//ASSIGN-OUTPUT-FILE FILE=*TAPE-FILE,LINK-NAME=PCOUT1 ───────────────── (6)
//START-CONVERSION ──────────────────────────────────────────────────── (7)
%  DMS0DE3 TAPE WITH VSN 'TAPEKA' FOR FILE ':catid:$userid.FILE.01.TAPE'
           IS MOUNTED ON DEVICE 'T0'
%  DMS0DE7 SAM FILE CLOSED: FILE NAME=:catid:$userid.FILE.01.TAPE, LINK
           NAME=PCOUT1, BLOCK COUNT=00000001
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCIN'
           (FILE=:catid:$userid.FILE.01):         10
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCOUT1'
           (FILE=::catid:$userid.FILE.01.TAPE):         10
//ASSIGN-INPUT-FILE FILE=*DISK-FILE(NAME=FILE.02) ───────────────────── (8)
//ASSIGN-OUTPUT-FILE FILE=*TAPE-FILE,LINK-NAME=PCOUT2 ───────────────── (9)
```

```
//START-CONVERSION ───────────────────────────────────────────────────── (10)
%  DMSODE3 TAPE WITH VSN 'TAPEKA' FOR FILE ':catid:$userid.FILE.02.TAPE'
           IS MOUNTED ON DEVICE 'T0'
%  DMSODE3 TAPE WITH VSN 'BD0150' FOR FILE ':catid:$userid.FILE.02.TAPE'
           IS MOUNTED ON DEVICE 'T1' ──────────────────────────────────── (11)
%  DMSODE8 END OF TAPE FOR FILE ':catid:$userid.FILE.02.TAPE' WITH LINK
           NAME 'PCOUT2', BLOCK COUNT '000000' ON VOLUME WITH VSN 'TAPEKA
%  DMSODE8 END OF TAPE FOR FILE ':catid:$userid.FILE.02.TAPE' WITH LINK
           NAME 'PCOUT2', BLOCK COUNT '000060' ON VOLUME WITH VSN 'BD0150'
%  DMSODE7 SAM FILE CLOSED: FILE NAME=:catid:$userid.FILE.02.TAPE, LINK
           NAME=PCOUT2, BLOCK COUNT=00000060
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCIN'
           (FILE=:catid:$userid.FILE.02):         60
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCOUT2'
           (FILE=:catid:$userid.FILE.02.TAPE):         60
//ASSIGN-INPUT-FILE FILE=*DISK-FILE(NAME=FILE.03) ───────────────────── (12)
//ASSIGN-OUTPUT-FILE FILE=*TAPE-FILE,LINK-NAME=PCOUT3 ───────────────── (13)
//END ───────────────────────────────────────────────────────────────── (14)
%  DMSODE3 TAPE WITH VSN 'BD0150' FOR FILE ':catid:$userid.FILE.03.TAPE'
           IS MOUNTED ON DEVICE 'T1'
%  DMSODE7 SAM FILE CLOSED: FILE NAME=:catid:$userid.FILE.03.TAPE, LINK
           NAME=PCOUT3, BLOCK COUNT=00000001
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCIN'
           (FILE=:catid:$userid.FILE.03):         10
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCOUT3'
           (FILE=:catid:$userid.FILE.03.TAPE):         10
%  PER0031 PERCON TERMINATED NORMALLY
/DELETE-TAPE-SET TAPE-SET-NAME=SET1 ─────────────────────────────────── (15)
```

(1) The two volumes (TAPEKA and BD0150) are declared as a tape set.

(2) The commands CREATE-FILE and ADD-FILE-LINK define the first output file (FILE-SEQUENCE=1) to be written to tape. The two volumes are assigned via the TAPE-SET-NAME parameter.

(3) Further output files (FILE-SEQUENCE=*NEW) are defined in the same manner as (2).

(4) PERCON is started.

(5) The first input file is assigned.

(6) The first output file is assigned.

(7) Transfer is started. The complete input file fits on the first tape.

(8) Another input file is assigned.

(9) Another output file is assigned.

(10) Transfer is started.

(11) The input file does not fit completely on the first tape. The second tape is used for the remainder of the input file.

(12) Another input file is assigned.

(13) Another output file is assigned.

(14) Transfer is started. The complete input file fits on the second tape. Transfer is completed and PERCON is terminated.

(15) The tape set declaration is deleted.

Loading a file from a MF/MV set

The file FILE.02.TAPE, which was saved to two tapes, is to be loaded.

```
/DELETE-FILE FILE-NAME=FILE.02.TAPE ————————————————————————————————————  (1)
%  DMS0800 SPECIFIED FILE ':catid:$userid.FILE.02.TAPE' DELETED
/SET-JOB-STEP
/IMPORT-FILE SUPPORT=*TAPE(FILE-NAME=FILE.02.TAPE,- ————————————————————  (2)
/            VOLUME=(TAPEKA,BD0150),DEVICE-TYPE=T6250)
/ADD-FILE-LINK LINK-NAME=PCIN,FILE-NAME=FILE.02.TAPE,- —————————————————  (3)
/            SUPPORT=*TAPE(FILE-SEQUENCE=*UNKNOWN)
/ADD-FILE-LINK LINK-NAME=PCOUT,FILE-NAME=FILE.02,- —————————————————————  (4)
/            ACCESS-METHOD=*SAM,-
/            RECORD-FORMAT=*FIXED,RECORD-SIZE=2000
/START-PERCON ———————————————————————————————————————————————————————————  (5)
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-INPUT-FILE FILE=*TAPE-FILE ————————————————————————————————————  (6)
//END ———————————————————————————————————————————————————————————————————  (7)
%  DMS0DE3 TAPE WITH VSN 'TAPEKA' FOR FILE ':catid:$userid.FILE.02.TAPE'
           IS MOUNTED ON DEVICE 'T0'
%  DMS0DE3 TAPE WITH VSN 'BD0150' FOR FILE ':catid:$userid.FILE.02.TAPE'
           IS MOUNTED ON DEVICE 'T1'
%  DMS0DE8 END OF TAPE FOR FILE ':catid:$userid.FILE.02.TAPE' WITH LINK
           NAME 'PCIN', BLOCK COUNT '000000' ON VOLUME WITH VSN 'TAPEKA'
%  DMS0DE8 END OF TAPE FOR FILE ':catid:$userid.FILE.02.TAPE' WITH LINK
           NAME 'PCIN', BLOCK COUNT '000060' ON VOLUME WITH VSN 'BD0150'
%  DMS0DE7 SAM FILE CLOSED: FILE NAME=:catid:$userid.FILE.02.TAPE, LINK
           NAME=PCIN, BLOCK COUNT=00000060
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCIN'
           (FILE=:catid:$userid.FILE.02.TAPE):         60
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCOUT'
           (FILE=:catid:$userid.FILE.02):         60
%  PER0031 PERCON TERMINATED NORMALLY
```

(1) The catalog entry, if any, for the file to be loaded is deleted.

(2) The file to be loaded is imported from the volumes of the MF/MV set.

(3) The input file is assigned using the ADD-FILE-LINK command.

The parameter FILE-SEQUENCE=*UNKNOWN is specified since the position of the input file in the sequence of files contained on the volumes is not known.

(4) An output file is assigned on disk.

(5) PERCON is started.

(6) The input file is assigned:

Assigning the input file with ASSIGN-INPUT-FILE is necessary as FILE=*TAPE-FILE is not the default value.
The ASSIGN-OUTPUT-FILE statement can be omitted since the output file is defined unequivocally by the ADD-FILE-LINK command and the default settings of the ASSIGN statement.

(7) Transfer is started. The input file is loaded from two volumes and PERCON is terminated.

8.15 Conversion of a file from EDF03IRV to UTF-16 (Unicode)

The file PERS.DPT.1 is converted in various ways. The output files have the same file attributes. PERCON is notified of the file names by means of the TFT entry (PCIN/PCOUT). The output file is provided with the required CCSN (here UTF-16). From this, PERCON recognizes that conversion is to take place as the input file is not in the Unicode variant UTF-16.

Variant 1: Conversion of a complete record

Input:

SAM file PERS.DPT.1 with CCSN EDF03IRV

Output:

SAM file PERS.DPT.UFT16 with CCSN UTF-16

Tracer listing:

```
/ADD-FILE-LINK FILE-NAME=PERS.DPT.1,LINK-NAME=PCIN ———————————————————————  (1)
/CREATE-FILE FILE-NAME=PERS.DPT.UTF16
/ADD-FILE-LINK FILE-NAME=PERS.DPT.UTF16, - ———————————————————————————————  (2)
/          LINK-NAME=PCOUT,ACCESS-METHOD=*SAM
/MODIFY-FILE-ATTRIBUTES FILE-NAME=PERS.DPT.UTF16, - ——————————————————————  (3)
/          CODED-CHARACTER-SET=UTF16
/START-PERCON ————————————————————————————————————————————————————————————  (4)
%  BLS0523 ELEMENT 'PCROOT', VERSION '029', TYPE 'L' FROM LIBRARY
           ':BO5C:$TSOS.SYSLNK.PERCON.029' IN PROCESS
%  BLS0524 LLM 'PCROOT', VERSION '02.9A' OF '2006-06-06 15:57:33'LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006.
           ALL RIGHTS RESERVED
%  PER0000 PERCON STARTED, VERSION V02.9A00
//END ————————————————————————————————————————————————————————————————————  (5)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCIN' ———————————————————  (6)
           (FILE=:BO5C:$TSOS.PERS.DPT. 1):                   8
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCOUT'
           (FILE=:BO5C:$TSOS.PERS.DPT.UTF16):                   8
%  PER0031 PERCON TERMINATED NORMALLY ————————————————————————————————————  (7)
```

(1) The input file PERS.DPT.1 (CCSN: EDF03IRV or NONE) is assigned.

(2) The access method SAM (ACCESS=*SAM) is defined. This is mandatory for conversion.

(3) The output file PERS.DPT.UTF16 is assigned. The CCSN UTF-16 causes conversion to take place in the PERCON run.

(4) PERCON is called.

(5) The END statement starts the conversion operation and terminates PERCON.

(6) PERCON messages: The number of records transferred per file is output.

(7) PERCON was terminated normally.

Variant 2: Conversion and normalization of a complete record

Input:

SAM file PERS.DPT.1 with CCSN EDF03IRV

Output:

SAM file PERS.DPT.UFT16.NORM with CCSN UTF-16

Tracer listing:

```
/ADD-FILE-LINK FILE-NAME=PERS.DPT.1,LINK-NAME=PCIN ——————————————————————  (1)
/CREATE-FILE    FILE-NAME=PERS.DPT.UTF16.NORM
/ADD-FILE-LINK FILE-NAME=PERS.DPT.UTF16.NORM,  - ————————————————————————  (2)
/          LINK-NAME=PCOUT,ACCESS-METHOD=*SAM
/MODIFY-FILE-ATTRIBUTES FILE-NAME=PERS.DPT.UTF16.NORM,   - ——————————————  (3)
/          CODED-CHARACTER-SET=UTF16
/START-PERCON ———————————————————————————————————————————————————————————  (4)
%  BLS0523 ELEMENT 'PCROOT', VERSION '029', TYPE 'L' FROM LIBRARY
          ':B05C:$TSOS.SYSLNK.PERCON.029' IN PROCESS
%  BLS0524 LLM 'PCROOT', VERSION '02.9A' OF '2006-06-06 15:57:33'LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006.
          ALL RIGHTS RESERVED
%  PER0000 PERCON STARTED, VERSION V02.9A00
//ASSIGN-OUTPUT-FILE FILE=*DISK-FILE(UNICODE-NORMALIZE=*COMPOSED) ———————  (5)
//END ———————————————————————————————————————————————————————————————————  (6)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCIN' ——————————————————  (7)
          (FILE=:B05C:$TSOS.PERS.DPT. 1):           8
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCOUT'
          (FILE=:B05C:$TSOS.PERS.DPT.UTF16.NORM):      8
%  PER0031 PERCON TERMINATED NORMALLY ————————————————————————————————————  (8)
```

(1) The input file PERS.DPT.1 (CCSN: EDF03IRV or NONE) is assigned.

(2) The access method SAM (ACCESS=*SAM) is defined. This is mandatory for conversion.

(3) The output file PERS.DPT.UTF16.NORM is assigned. The CCSN UTF-16 causes conversion to take place in the PERCON run.

(4) PERCON is called.

(5) Normalization of the data is requested. In this case normalization is superfluous as all characters of the input file PERS.DPT.1 are available in normalized form. For performance reasons it is better to do without normalization in this example.

(6) The END statement starts the conversion operation and terminates PERCON.

(7) PERCON messages: The number of records transferred per file is output.

(8) PERCON was terminated normally.

Variant 3: Conversion of parts of a record

All the data is converted in this example.

Input:

SAM file PERS.DPT.1 with CCSN EDF03IRV

Output:

SAM file PERS.DPT.UFT16.PART with CCSN UTF-16

Tracer listing:

```
/ADD-FILE-LINK FILE-NAME=PERS.DPT.1,LINK-NAME=PCIN ——————————————————————  (1)
/CREATE-FILE    FILE-NAME=PERS.DPT.UTF16.PART
/ADD-FILE-LINK FILE-NAME=PERS.DPT.UTF16.PART,  - ————————————————————————  (2)
/          LINK-NAME=PCOUT,ACCESS-METHOD=*SAM
/MODIFY-FILE-ATTRIBUTES FILE-NAME=PERS.DPT.UTF16.PART,   - ——————————————  (3)
/          CODED-CHARACTER-SET=UTF16
/START-PERCON ———————————————————————————————————————————————————————————  (4)
%  BLS0523 ELEMENT 'PCROOT', VERSION '029', TYPE 'L' FROM LIBRARY
          ':B05C:$TSOS.SYSLNK.PERCON.029' IN PROCESS
%  BLS0524 LLM 'PCROOT', VERSION '02.9A' OF '2006-06-06 15:57:33'LOADED
%  BLS0551 COPYRIGHT (C) FUJITSU SIEMENS COMPUTERS GMBH 2006.
          ALL RIGHTS RESERVED
%  PER0000 PERCON STARTED, VERSION V02.9A00
//SET-RECORD-MAPPING OUTPUT-FIELDS=*FIELD(  - ———————————————————————————  (5)
//          INPUT-POSITION=05,INPUT-LENGTH=68,  -
//          OUTPUT-POSITION=5,OUTPUT-LENGTH=136,  -
//          OUTPUT-FORMAT=*UNICODE-TRANSLATION)
//END ———————————————————————————————————————————————————————————————————  (6)
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCIN' ——————————————————  (7)
          (FILE=:B05C:$TSOS.PERS.DPT. 1):          8
```

```
%  PER0030 NUMBER OF PROCESSED RECORDS FOR LINK='PCOUT'
          (FILE=:BO5C:$TSOS.PERS.DPT.UTF16.PART):       8
%  PER0031 PERCON TERMINATED NORMALLY  ——————————————————————————————  (8)
```

(1) The input file PERS.DPT.1 (CCSN: EDF03IRV or NONE) is assigned.

(2) The access method SAM (ACCESS=*SAM) is defined. This is mandatory for conversion.

(3) The output file PERS.DPT.UTF16.PART is assigned. The CCSN UTF-16 causes conversion to take place in the PERCON run.

(4) PERCON is called.

(5) Conversion of the record section from record position 5 through 72 (without record length field) is defined.

(6) The END statement starts the conversion operation and terminates PERCON.

(7) PERCON messages: The number of records transferred per file is output.

(8) PERCON was terminated normally.

Format of the file PERS.DPT.UTF16

The files PERS.DPT.UTF16.NORM and PERS.DPT.UTF16.PART have the appropriate format.

The first row is in printable format. The following rows contain the hexadecimal representation. Each character is separated by a blank here.

```
BELL        JOHN        MANCHESTER     BOLSOVER STREET 4        DPT 1

0042 0045 004C 004C 0020 0020 0020 0020 0020 0020 0020 0020 004A 004F 0048
004E 0020 0020 0020 0020 0020 0020 0020 0020 004D 0041 004E 0043 0048 0045
0053 0054 0045 0052 0020 0020 0020 0020 0020 0042 004F 004C 0053 004F 0056
0045 0052 0020 0053 0054 0052 0045 0045 0054 0020 0034 0020 0020 0020 0020
0020 0020 0020 0020 0044 0050 0054 0031

BOTHAM      NORMAN      MANCHESTER     TOWER AVENUE 10          DPT2

0042 004F 0054 0048 0041 004D 0020 0020 0020 0020 0020 0020 004E 004F 0052
004D 0041 004E 0020 0020 0020 0020 0020 0020 004D 0041 004E 0043 0048 0045
0053 0054 0045 0052 0020 0020 0020 0020 0020 0054 004F 0057 0045 0052 0020
0041 0056 0045 004E 0055 0045 0020 0031 0030 0020 0020 0020 0020 0020 0020
0020 0020 0020 0020 0044 0050 0054 0032

FINN        SUSANNA     NORWICH        ROSE STREET 11           DPT2

0046 0049 004E 004E 0020 0020 0020 0020 0020 0020 0020 0020 0053 0055 0053
0041 004E 004E 0041 0020 0020 0020 0020 0020 004E 004F 0052 0057 0049 0043
0048 0020 0020 0020 0020 0020 0020 0020 0020 0052 004F 0053 0045 0020 0053
0054 0052 0045 0045 0054 0020 0031 0031 0020 0020 0020 0020 0020 0020 0020
0020 0020 0020 0020 0044 0050 0054 0032
```

```
GREENE      WALTER      BURY           SINCLAIR CRESCENT 7      DPT1

0047 0052 0045 0045 004E 0045 0020 0020 0020 0020 0020 0020 0057 0041 004C
0054 0045 0052 0020 0020 0020 0020 0020 0020 0042 0055 0052 0059 0020 0020
0020 0020 0020 0020 0020 0020 0020 0020 0020 0053 0049 004E 0043 004C 0041
0049 0052 0020 0043 0052 0045 0053 0043 0045 004E 0054 0020 0037 0020 0020
0020 0020 0020 0020 0044 0050 0054 0031

KING        MONICA      FALMOUTH       INMAN SQUARE 61          DPT3

004B 0049 004E 0047 0020 0020 0020 0020 0020 0020 0020 0020 004D 004F 004E
0049 0043 0041 0020 0020 0020 0020 0020 0020 0046 0041 004C 004D 004F 0055
0054 0048 0020 0020 0020 0020 0020 0020 0020 0049 004E 004D 0041 004E 0020
0053 0051 0055 0041 0052 0045 0020 0036 0031 0020 0020 0020 0020 0020 0020
0020 0020 0020 0020 0044 0050 0054 0033

LAKER       ERICA       MANCHESTER     BANK DRIVE 8             DPT1

004C 0041 004B 0045 0052 0020 0020 0020 0020 0020 0020 0020 0045 0052 0049
0043 0041 0020 0020 0020 0020 0020 0020 0020 004D 0041 004E 0043 0048 0045
0053 0054 0045 0052 0020 0020 0020 0020 0020 0042 0041 004E 004B 0020 0044
0052 0049 0056 0045 0020 0038 0020 0020 0020 0020 0020 0020 0020 0020 0020
0020 0020 0020 0020 0044 0050 0054 0031

PRICE       ALFRED      MANCHESTER     THAMES ROAD 4            DPT1

0050 0052 0049 0043 0045 0020 0020 0020 0020 0020 0020 0020 0041 004C 0046
0052 0045 0044 0020 0020 0020 0020 0020 0020 004D 0041 004E 0043 0048 0045
0053 0054 0045 0052 0020 0020 0020 0020 0020 0054 0048 0041 004D 0045 0053
0020 0052 004F 0041 0044 0020 0034 0020 0020 0020 0020 0020 0020 0020 0020
0020 0020 0020 0020 0044 0050 0054 0031
```

```
WILSON       RICHARD      MANCHESTER   ACACIA AVENUE 24           DPT3
```

```
0057 0049 004C 0053 004F 004E 0020 0020 0020 0020 0020 0020 0052 0049 0043
0048 0041 0052 0044 0020 0020 0020 0020 0020 004D 0041 004E 0043 0048 0045
0053 0054 0045 0052 0020 0020 0020 0020 0020 0041 0043 0041 0043 0049 0041
0020 0041 0056 0045 004E 0055 0045 0020 0032 0034 0020 0020 0020 0020 0020
0020 0020 0020 0020 0044 0050 0054 0033
```

9 Messages

9.1 Message output to S variables

For a number of messages, message code and inserts (number and meaning) are guaranteed as invariable components of future PERCON and BS2000/OSD-BC versions. These messages are referred to as guaranteed messages.

Guaranteed messages of the PERCON product

The following types of PERCON messages are guaranteed messages:

– messages concerning PERCON start and termination

– messages indicating the number of records or blocks processed

– messages with inserts such as DMS error codes, file names etc.

– messages triggered by errors other than violations of the syntax or semantics of statements (e.g. records in a file that are too short)

The following is a list of guaranteed PERCON messages:

PER0000	PERCON started
PER0001	PERCON run aborted
PER0002	Error in a system macro
PER0003	Insufficient memory space
PER0004	File not on specified type of volume
PER0009	Output record length exceeds maximum length
PER0012	Inconsistency in file names
PER0014	Alternate (secondary) keys not transferred
PER0016	DMS error while repairing file
PER0017 - 19	XHCS problems
PER0021 - 23	OPEN/CLOSE problems
PER0028	Link error in user modules
PER0029 - 30	Number of processed blocks/records
PER0031	PERCON terminated normally
PER0035	ISAM key not ascending
PER0036	DMS error

PER0038	Abnormal termination due to parity or length error
PER0039	CCSN incompatible with SYSOUT
PER0041 - 42	Error in data record
PER0044	Data record too short
PER0045	Data records not in sequence
PER0048	Invalid return code from a user routine
PER0053 - 54	Output file incompatible with FILLER=*OUTPUT
PER0057	Device error during positioning
PER0069	Error during tape duplication
PER0092 - 94	File repairing using VERIF
PER0095	Prelinked PERCON module cannot be loaded
PER0098	ISAM key already exists
PER0099 - 103	Error during alternate (secondary) key transfer
PER0104 - 107	PERCON version not available
PER0109	CCSN for output file cannot be recorded
PER0110	Invalid BS2000 version
PER0111	BIND error during dynamical loading of prelinked module

When using the software product SDF-P (V2.0A or later), these guaranteed messages can be output in structured S variables. S variables provide direct access to specific message data even if the output layout of the messages is unknown. This permits further processing to be controlled in SDF-P procedures depending on the contents of this S variables.

For guaranteed messages the message attribute “Warranty” is documented by “ ◆ Warranty: Y” in the line after the message text.

How to proceed

An S variable that is to be supplied with messages must be built as a list of structures. Its name is optional; `varname` is used as its name in this section.
The user has to declare the S variable using the following command:

```
/DECLARE-VARIABLE NAME=varname(TYPE=*STRUCTURE(DEFINITION=*DYNAMIC)),-
/                 MULTIPLE-ELEMENTS=*LIST,SCOPE=*TASK
```

Note

Specifying SCOPE=*TASK is not required unless the S variable is to be valid also in procedures called after declaring the S variable.

Each structure (message) is to consist of the following elements:

Name of the structure element	Contents
varname(*LIST).MSG-TEXT	Complete message text
varname(*LIST).MSG-ID	Message code
varname(*LIST).I0	Insert 0
varname(*LIST).I1	Insert 1
:	:
varname(*LIST).I14	Insert 14
varname(*LIST).REPLY	Entered response

The number of inserts depends on the message; the same applies to presence or absence of a response.

Once the S variable has been declared, it is assigned the message stream by means of the following command:

```
/ASSIGN-STREAM STREAM-NAME=SYSMSG,TO=*VARIABLE(varname)
```

After this has been done, PERCON or a program which invokes PERCON as a subprogram can be started. During the PERCON/program run, messages are stored in the S variable.

After PERCON or program termination, the message stream to the S variable is stopped and reassigned to the default output destination for messages:

```
/ASSIGN-STREAM STREAM-NAME=SYSMSG,TO=*STD
```

The variable `varname` contains all guaranteed messages output between the two ASSIGN-STREAM commands. In particular, these include the BLS messages.

If PERCON is invoked as a subprogram, the S variable additionally contains all guaranteed messages output by the main program.

If the output of additional messages is to be suppressed, the two ASSIGN-STREAM commands can be executed via a CMD macro immediately before and after PERCON is called in the main program.

Messages are output to the S variables, whether message output to SYSOUT or SYSLST is suppressed during this time or not.

For a more detailed description of S variables please refer to the “SDF-P” manual [13].

Evaluating S variables

The contents of an S variable can be output using the SHOW-VARIABLE command. The following is an example of the contents of an S variable 'pervar' after termination of the PERCON run:

```
/show-variable variable-name=pervar
```

```
PERVAR(*LIST).MSG-TEXT = % BLS0517 MODULE 'PCROOT' LOADED
PERVAR(*LIST).MSG-ID = BLS0517
PERVAR(*LIST).I0 = PCROOT
PERVAR(*LIST).MSG-TEXT = % PER0000 PERCON STARTED, VERSION V02.9A00
PERVAR(*LIST).MSG-ID = PER0000
PERVAR(*LIST).I0 = V02.96A00
PERVAR(*LIST).MSG-TEXT = % PER0044 FIELD IN 'SET-RECORD-MAPPING' STATEMENT
EXCEEDS INPUT/OUTPUT RECORD END, LINK='PCOUT'
PERVAR(*LIST).MSG-ID = PER0044
PERVAR(*LIST).I0 = SET-RECORD-MAPPING
PERVAR(*LIST).I1 = PCOUT
PERVAR(*LIST).MSG-TEXT = % PER0030 NUMBER OF PROCESSED RECORDS FOR
LINK='PCIN' :         3
PERVAR(*LIST).MSG-ID = PER0030
PERVAR(*LIST).I0 = PCIN
PERVAR(*LIST).I1 =
PERVAR(*LIST).I2 =           3
PERVAR(*LIST).MSG-TEXT = %  PER0030 NUMBER OF PROCESSED RECORDS FOR
LINK='PCOUT' (FILE=:catid:$userid.XXX):          2
PERVAR(*LIST).MSG-ID = PER0030
PERVAR(*LIST).I0 = PCOUT
PERVAR(*LIST).I1 = (FILE=:catid:$userid.XXX)
PERVAR(*LIST).I2 =           2
PERVAR(*LIST).MSG-TEXT = %  PER0001 PERCON RUN ABORTED
PERVAR(*LIST).MSG-ID = PER0001
```

The individual structure elements of the variable can be addressed separately (e.g. `/show-variable variable-name=pervar#3.MSG-ID` obtains the message code of the third message).

The following example shows a section of an SDF-P procedure containing a FOR loop that stores guaranteed messages in S variables:

```
/DECLARE-VARIABLE NAME=PERVAR(TYPE=*STRUCTURE), - ——————————————————————  (1)
/                 MULTIPLE-ELEMENTS=*LIST
/DECLARE-VARIABLE NAME=PERSTR(TYPE=*STRUCTURE) —————————————————————————  (2)
/ASSIGN-STREAM STREAM-NAME=SYSMSG,TO=*VARIABLE(PERVAR) —————————————————  (3)
/START-PERCON ——————————————————————————————————————————————————————————  (4)
.
...   (PERCON statements etc.)
.
```

```
/ASSIGN-STREAM STREAM-NAME=SYSMSG,TO=*STD ─────────────────────────────  (5)
/FOR PERSTR= *LIST(PERVAR) ────────────────────────────────────────────  (6)
/    IF (PERSTR.MSG-ID = 'PER0001')
/        WRITE-TEXT '*-----------------------------------------*'
/        WRITE-TEXT '* PERCON ABORTED / RESPONSE REQUIRED *'
/        WRITE-TEXT '*-----------------------------------------*'
/    END-IF
/END-FOR
```

(1) The S variable PERVAR is declared as a list of structures.

(2) The structure variable PERSTR is declared for the subsequent FOR loop.

(3) Guaranteed messages are stored in PERVAR following this command.

(4) PERCON is started.

(5) Output of guaranteed messages to PERVAR is stopped by this command.

(6) A FOR loop searches the entire list of structures PERVAR looking for an occurrence of the message code PER0001 as MSG-ID. If it is found, information is output with WRITE-TEXT.

The error messages are output in 7-character (PERnnnn) form, together with an explanation and any actions which must be taken, either in English or in German. Guaranteed messages are marked by Warranty: Y.

The scope of the interactive messages can be controlled by means of specifications in the MODIFY-PERCON-OPTIONS statement.

9.2 PERCON messages

`PERCOPY` `Copyright (C) (&00) (&01). All rights reserved.`

`PERLOAD` `Program '(&00)', Version '(&01)' of '(&02)' loaded from file '(&03)'`

`PERSTRT` `Program '(&00)', Version '(&01)' of '(&02)' started from file '(&03)'`

`PER0000` `PERCON STARTED, VERSION (&00)`

◆ Warranty: Y

Meaning
(&00): PERCON version (nn.nann).

`PER0001` `PERCON RUN ABORTED`

◆ Warranty: Y

Meaning
The reason for the abortion is described:
- in the last error message if PERCON was called autonomously or
- in the return information if PERCON was called as a subroutine.
If PERCON was called autonomously in a procedure, the run will continue at the next /STEP or /SET-STEP command or at the end of the procedure.

Response
Repeat the PERCON run after removing the error.

`PER0002` `ERROR CODE X'(&01)' DURING '(&00)' MACRO`

◆ Warranty: Y

Meaning
Either a read or write error in a system macro causes abortion of the conversion step or the just activated PERCON interruption processing will be terminated after a SOLSIG/POSSIG call error.

Response
Consult the 'Executive Macros' manual for an explanation of the error type and repeat the run.

`PER0003` `INSUFFICIENT CLASS 6 MEMORY SPACE`

◆ Warranty: Y

Meaning
The conversion step terminates abnormally.

Response
Remove the cause of memory saturation and repeat the run.

```
PER0004    FILE NOT ON '(&01)', LINK='(&00)'
```
◆ Warranty: Y

Meaning
In the ASSIGN-INPUT-FILE or ASSIGN-OUTPUT-FILE statement the file specified
by FILE=DISK-FILE() was not a disk file or the file specified by FILE=TAPE-FILE() was not a tape file.
(&00): link name
(&01): DISK, TAPE.

Response
Correct the ASSIGN-INPUT-FILE or ASSIGN-OUTPUT-FILE statement or the /ADD-FILE-LINK (FILE) command and repeat the conversion step.

```
PER0005    INVALID FORMAT CONVERSION
```

Meaning
In the statement SET-RECORD-MAPPING or SET-GROUP-ATTRIBUTES an invalid combination of the structure operands INPUT-FORMAT and OUTPUT-FORMAT in the operand value FIELD was given
or an INPUT-LENGTH > 4 was given in combination with INPUT-FORMAT=CHARACTER
and OUTPUT-FORMAT=ZONED-DECIMAL.

Response
In interactive mode correct the contradictory operands,
in batch mode repeat the conversion step after correcting the statement.

```
PER0006    OPERAND VALUE 'RECORD-LENGTH' ONLY PERMISSIBLE FOR 'INPUT-FORMAT=CHARACTER'
```

Meaning
The structure operand value RECORD-LENGTH of the operand value FIELD of the
SET-RECORD-MAPPING statement is only allowed for INPUT-FORMAT=CHARACTER.

Response
In interactive mode correct the contradictory operands,
in batch mode repeat the conversion step after correction.

PER0007 STRUCTURE OPERAND 'OUTPUT-LENGTH' TOO SMALL

Meaning
In the OUTPUT-FIELDS operand of the SET-RECORD-MAPPING statement or in the GROUP-HEADER resp. GROUP-TRAILER operand of the statement SET-GROUP-ATTRIBUTES a field or keyword is used, whose structure operand OUTPUT-LENGTH is insufficient for output and format conversion.

Response
In interactive mode correct the invalid operand,
in batch mode repeat the conversion step after correction.

PER0008 OPERAND VALUE 'REPLACE-CHARACTER' SPECIFIED MORE THAN ONCE

Meaning
In the CODE-TRANSLATION operand of the SET-RECORD-MAPPING statement a character has been specified more than once as an INPUT-CHARACTER structure
operand.

Response
In interactive mode correct the invalid operand,
in batch mode repeat the conversion step after correction.

PER0009 WARNING: OUTPUT RECORD LENGTH > MAXIMUM RECORD LENGTH, LINK='(&00)'

- Warranty: Y

Meaning
Output records, which exceed the maximum permissible output record length, will be cut off on the right side. The message is only output for the first record that hits the above condition.
(&00): link name of the output file.

Response
Shorten records by using of SET-RECORD-MAPPING (RECORD) statement or enlarge record- and / or block size of the output file.

PER0010 NOT A PERCON STATEMENT

Meaning
The last statement read in cannot be analyzed.

Response
Correct the operation part.

`PER0011` `STATEMENT READ ERROR`

Meaning
An error has occurred when reading a statement.
Possible causes:
- a command was read instead of a statement or
- SDF is not available or
- PERCON statements are missing in the syntax file.

Response
Check the procedure; if necessary, consult the system administrator.

`PER0012` `WARNING: INCONSISTENCY IN FILE NAMES, LINK='(&00)'`

◆ Warranty: Y

Meaning
The file name in TFT-entry is different from the name in the ASSIGN-INPUT-FILE (FILIN) or ASSIGN-OUTPUT-FILE (FILOUT) statement or a filegeneration is used.
The file addressed by the name in the /ADD-FILE-LINK (FILE) command is used.
(&00): link name.

Response
Use file name only once or use equal names.

`PER0013` `SYNTACTICAL ERROR IN '(&00)' STATEMENT`

Meaning
The syntax of the given statement is incorrect. The protocol shows the incorrect section (operand or operand value).
(&00): name of the invalid statement.

Response
In interactive mode correct the invalid statement,
in batch mode repeat the conversion step after correcting the statement.

`PER0014` `WARNING: ALTERNATE KEYS NOT TRANSFERRED, LINK='(&00)'`

◆ Warranty: Y

Meaning
The input file has alternate keys in opposition to the output file.
For NK-ISAM files alternate keys can be created afterwards by the /CREATE-ALTERNATE-INDEX command, this is not possible for K-ISAM files.
(&00): link name.

Response
If necessary use /CREATE-ALTERNATE-INDEX command.

PER0015 `ILLEGAL COMBINATION OF INPUT, OUTPUT AND TRANSPOSE`

Meaning
A conversion can not be processed because input,
output and transpose parameter do not suit together.

Response
Check file characteristics and conversion statements.

PER0016 `WARNING: DMS ERROR '(&00)' WHILE REPAIRING FILE (LINK='(&01)'). FURTHER INFORMATION: /HELP-MSG DMS(&00)`

- Warranty: Y

Meaning
An error was detected when the file with link name (&01) was opened.
During the following repair by means of macro VERIF
a DMS error occurred in the OPENC exit.
Processing of the invalid file is continued.
For more detailed information about the DMS error code enter /HELP-MSG
or see the BS2000 manual 'System Messages'.
(&00): DMS error code
(&01): link name.

Response
Check if the output is complete.

PER0017 `C STRING OF STATEMENT '(&00)' NOT CONVERTIBLE, LINK='(&01)'`

- Warranty: Y

Meaning
When using C strings in the statement SELECT-INPUT-RECORDS, the
statement CCS and the input CCS must belong to the same code family and
all characters of the C strings must exist in the input CCS.
When using C strings in the SET-GROUP-ATTRIBUTES or SET-RECORD-MAPPING
or SET-PAGE-LAYOUT statement, the statement CCS and the output CCS
must belong to the same code family and all characters of the C strings
must exist in the ouput CCS.
(&00): possible invalid statement:
SELECT-INPUT-RECORDS,
SET-GROUP-ATTRIBUTES,
SET-RECORD-MAPPING,
SET-PAGE-LAYOUT,
(&01): link name of the output.

Response
Convert the statements to the input CCS or output CCS.

`PER0018` `EXTENDED HOST CODE TABLE NOT AVAILABLE, LINK='(&01)'`

- Warranty: Y

Meaning
PERCON cannot determine which characters of the input CCS are letters or digits, or which characters of the output CCS are printable.
(&01): link name.

Response
Contact the system administrator. Check the XHCS tables of the system.

`PER0019` `CONVERSION OF INPUT DATA INTO OUTPUT CCS NOT POSSIBLE, LINK='(&01)'`

- Warranty: Y

Meaning
The condition for a conversion is that the input CCS and the output CCS have the same name or that the input CCS and the output CCS belong to the same code family and the output CCS is a compatible CCS.
(&01): link name of the output file.

Response
Use another output CCS.

`PER0020` `ILLEGAL OVERLAPPING OF OUTPUT AREAS`

Meaning
The SET-PAGE-LAYOUT statement contains overlapping areas in the
OUTPUT-AREA
operand.

Response
Check the SET-PAGE-LAYOUT statement.

`PER0021` `OPEN ERROR ON INPUT FILE X'(&01)', LINK='(&00)'. REPLY (HA = CONV. TERM. ABNORMALLY; HN = CONV. TERM.; SA = SKIP INPUT FILE, PERCON ABORTED; SN = SKIP INPUT FILE, PERCON TERM.)`

- Warranty: Y

Meaning
When opening the inputfile with link name (&00) error (&01) occurred.
(&00): link name
(&01): DMS error code.

Response
Possible responses:
- HA: terminate conversion step, abnormal termination of PERCON or
- HN: terminate conversion step, normal termination of PERCON or
- SA: skip invalid input file, abnormal termination of PERCON or
- SN: skip invalid input file, normal termination of PERCON.

PER0022 `FILE OPEN ERROR X'(&01)', LINK='(&00)'. FURTHER INFORMATION: /HELP-MSG DMS(&01)`

- Warranty: Y

Meaning
When opening the file with link name (&00) error (&01) occurred.
If the invalid file is an input file, further processing by PERCON depends on the value set for the OPEN-ERROR (OPENER) operand of the ASSIGN-INPUT-FILE (FILIN) statement.
Open errors on output files lead to termination of the conversion step.
(&00): link name
(&01): DMS error code.

PER0023 `FILE CLOSE ERROR X'(&01)', LINK='(&00)'. FURTHER INFORMATION: /HELP-MSG DMS(&01)`

- Warranty: Y

Meaning
When closing the file with link name (&00) error X'(&01)' occurred.
The conversion step is terminated.
(&00): link name
(&01): DMS error code.

PER0024 `WARNING: INVALID STATEMENT IGNORED`

Meaning
The read statement is invalid.
The kind of error is described in the previous message.

PER0025 `INVALID USE OF A LINK NAME`

Meaning
A link name in the statement read in is either
- specified more than once or
- not assigned to a file or
- at variance with the BS2000 naming conventions.

Response
Check the link names used.

PER0026 `NO DATA LINES FOR PRINT EDITING`

Meaning
The output mask constructed via the SET-PAGE-LAYOUT (FORMAT) statement contains no lines which are free to accommodate data.

Response
Add data lines in the OUTPUT-AREA (OUTLINE) operand and repeat the run.

PER0027 `GROUP CONTROL MAY ONLY BE MISSING ON GROUP LEVEL 1`

Meaning
In the SET-GROUP-ATTRIBUTES (GROUP) statement no group control field was specified by the GROUP-CONTROL (CHANGE) operand for a group level higher than 1.

PER0028 `USER MODULE '(&00)' WITH LINK ERROR '(&01)'`
- Warranty: Y

Meaning
Error (&01) has occurred when dynamically loading
user module (&00) via LINK.
(&00): name respectively entry point of user routine
(&01): returncode provided by macro LINK.

Response
Check whether the module library has been assigned by /SET-TASKLIB (/SYSFILE TASKLIB) and whether the module exists in the assigned library.

PER0029 `NUMBER OF PROCESSED BLOCKS FOR LINK='(&00)':(&01) (&02)`
- Warranty: Y

Meaning
Number of handled blocks for one tape/tape set. If the tape/tape set contains labels the number of handled labels is also shown.
(&00): link name
(&01): number of blocks
(&02): (INCL. n LABELS), n: number of labels.

PER0030 `NUMBER OF PROCESSED RECORDS FOR LINK='(&00)' (&01):(&02)`
- Warranty: Y

Meaning
At the end of the conversion step for the named files the file name is also
displayed.
(&00): link name
(&01): file name
(&02): number of records.

PER0031 `PERCON TERMINATED NORMALLY`
- Warranty: Y

PER0032 `CONVERSION STEP STARTED IN SPITE OF INVALID STATEMENT`

Meaning
In a procedure or a batch job a conversion step was started despite the presence of an invalid statement.
The location of the error is shown by the last message.

Response
Correct the invalid statement and repeat the PERCON run.

PER0033 `LINK NAME '(&00)' USED MORE THAN ONCE. LAST STATEMENT IS VALID`

Meaning
Two FILIN or FILOUT statements were specified for the same link name.

Response
If necessary, repeat the first statement with a different link name.

PER0034 `VOLUME IDENTIFICATION MISSING, LINK='(&00)'`

Meaning
The VOLUME operand is missing in the VOLIN or VOLOUT statement or in the corresponding /ADD-FILE-LINK (FILE) command.
(&00): link name.

Response
Specify the VOLUME operand and repeat the run.

PER0035 `ISAM KEY NOT ASCENDING, LINK='(&00)'`
- Warranty: Y

Meaning
During output to an ISAM file it becomes clear, that the ISAM keys are not in ascending order.
(&00): link name.

Response
Use a SAM output file or arrange the records by key with the aid of the SORT utility routine.

PER0036 `DMS ERROR CODE X'(&01)', LINK='(&00)'. FURTHER INFORMATION: /HELP-MSG DMS(&01)`
- Warranty: Y

Meaning
A DMS error was detected during file processing.
The conversion step is terminated.
(&00): link name
(&01): DMS error code.

PER0037 `SWITCHOVER THE CHARACTER SET NOT PERMITTED`

Meaning
All statements must belong to the same CCS.
All records of an input file must belong to the same CCS.

Response
Use only one character set.

PER0038 `FILE CLOSED BECAUSE OF (&01) ERROR, LINK='(&00)'`
- Warranty: Y

Meaning
When reading the file with link name (&00) a (&01) error occurred.
The conversion step is terminated.
(&00): link name
(&01): PARITY, WLRERR.

Response
For informationen to this error see DMS-Manual.
The reaction to a given error can be controled by the
operands PARITY-ERROR, LENGTH-ERROR resp. INPUT-ERROR of
the ASSIGN-INPUT-FILE resp. ASSIGN-INPUT-TAPE statements.

PER0039 `CCS '(&00)' INCOMPATIBLE WITH SYSOUT, LINK='(&01)'`
- Warranty: Y

Meaning
It is not possible to do an output to SYSOUT with the CCS.
(&00): name of coded character set
(&01): link name.

Response
Check the used CCS.

PER0040 `WARNING: ILLEGAL DIALOG REQUEST. DEFAULT VALUE ASSUMED IN BATCH MODE`

Meaning
The operand value DIALOG (USER(CONV)) is only allowed in interactive mode.
In batch mode PERCON assumes the default value and continues with
the conversion step.

`PER0041    INVALID FIELD FORMAT '(&00)' AT RECORD (&01), LINK='(&02)'`

- Warranty: Y

Meaning
The processed field does not have the format specified in the
SET-RECORD-MAPPING (RECORD) or SET-GROUP-ATTRIBUTES (GROUP) statement.
(&00): PACKED-DECIMAL (P), ZONED-DECIMAL (Z)
(&01): record counter
(&02): link name.

`PER0042    ILLEGAL LENGTH COMBINATION IN '(&00)' STATEMENT, LINK='(&01)'`

- Warranty: Y

Meaning
The relevant field lengths in the specified statement do not match
or the mask length is not suitable to data length.
(&00): possible invalid statement:
SELECT-INPUT-RECORDS (SELECT)
SET-RECORD-MAPPING (RECORD)
SET-GROUP-ATTRIBUTES (GROUP)
(&01): link name.

`PER0043    LINE LENGTH ((&00)) > MAXIMUM LENGTH (204), LINK='(&01)'`

Meaning
The length of the output line produced by the SET-PAGE-LAYOUT (FORMAT)
statement exceeds the maximum length (204 characters).
(&00): invalid length
(&01): link name.

Response
Check the operands LINE-SIZE (LLENGTH), COLUMN-SIZE (LGROUP)
and OUTPUT-FORMAT(MODE).
Regard the length of the record header.

`PER0044    FIELD IN '(&00)' STATEMENT EXCEEDS INPUT/OUTPUT RECORD END, LINK='(&01)'`

- Warranty: Y

Meaning
In the SELECT-INPUT-RECORDS (SELECT) or SET-RECORD-MAPPING (RECORD)
statement a field is used that lies totally or partly outside the
input/output record.
(&00): invalid statement
(&01): link name.

```
PER0045     RECORD '(&00)' NOT IN SEQUENCE, LINK='(&01)'
```

◆ Warranty: Y

Meaning
The number of the first record to violate the order specified in the SELECT-INPUT-RECORDS (SELECT) statement is output.
No message is output for subsequent records that are out of sequence.
(&00): record counter
(&01): link name.

```
PER0046     ILLEGAL 'VOLIN'/'VOLOUT' STATEMENT DURING FILE PROCESSING
```

Meaning
A previously issued FILIN statement has initiated a file-oriented conversion step.
In interactive mode the VOLIN/VOLOUT statement is ignored,
in batch mode the conversion step is aborted.

```
PER0047     ILLEGAL 'FILIN'/'FILOUT' STATEMENT DURING VOLUME PROCESSING
```

Meaning
A previously issued VOLIN/VOLOUT statement has initiated a volume-oriented conversion step.
In interactive mode the FILIN/FILOUT statement is ignored,
in batch mode the conversion step is aborted.

```
PER0048     RETURN CODE X'(&00)' OF USER ROUTINE '(&02)' NOT PERMITTED, LINK='(&01)'
```

◆ Warranty: Y

Meaning
The user routine provides an illegal return code.
The conversion step is aborted.
(&00): return code provided
(&01): link name
(&02): name respectively entry point of user routine.

Response
Correct the user routine and repeat the conversion step.

```
PER0049     ONLY ONE 'VOLIN' STATEMENT IS ALLOWED
```

Meaning
Only one VOLIN statement is allowed in a conversion step.

Response
Combine the VOLIN statements to make one single VOLIN statement, or distribute them over several conversion steps.

PER0050 `ILLEGAL 'EDIT'/'POSIT' STATEMENT DURING FILE PROCESSING`

Meaning
A previously issued FILIN statement has initiated a file-oriented conversion step.
In interactive mode the EDIT/POSIT statement is ignored,
in batch mode the conversion step is aborted.

PER0051 `WARNING: XHCS NOT ACTIVE, STATEMENT CCSN '(&00)' IGNORED`

Meaning
Subsystem XHCS is not active but there is a statement CCSN.
The CCSN will be ignored.
(&00): statement CCSN.

PER0052 `WARNING: XHCS NOT ACTIVE, FILE CCSN '(&00)' IGNORED, LINK='(&01)'`

Meaning
Subsystem XHCS is not active but the file has got a CCSN.
The CCSN will be ignored.
(&00): CCSN of file
(&01): link name.

PER0053 `FILE TYPE OR OPEN MODE WRONG FOR FILE '(&00)', LINK='(&01)'`
◆ Warranty: Y

Meaning
The output file has to be an ISAM respectively NK-ISAM file and has to be opened in open mode INOUT because FILLER=OUTPUT was specified in the statement SET-RECORD-MAPPING.
The conversion step is aborted.
(&00): file name
(&01): link name.

Response
Correct the statement and repeat the conversion step.

PER0054 `WARNING: KEY '(&00)' MISSING IN FILE '(&02)', LINK='(&01)'`
◆ Warranty: Y

Meaning
There is no record with this key in the file. The conversion step is continued.
This message is sent no more than ten times for every link name.
(&00): missing key
(&01): link name
(&02): file name.

PER0055 'VOLIN' STATEMENT MISSING FOR VOLUME PROCESSING

Meaning
A VOLOUT statement was issued, but no VOLIN statement.

PER0056 ILLEGAL COMBINATION OF FILLER=OUTPUT IN SET-RECORD-MAPPING STATEMENT WITH OTHER OPERANDS OR STATEMENTS, LINK='(&00)'

Meaning
FILLER=OUTPUT in the SET-RECORD-MAPPING statement was specified
for the file with link name (&00). This specification must not
be combined with:
SET-PAGE-LAYOUT statement,
SET-GROUP-ATTRIBUTES statement or
operand OVERWRITE=NO in ASSIGN-OUTPUT-FILE statement
for the same file.
(&00): link name.

Response
In interactive mode correct the invalid statements,
in batch mode repeat the conversion step after correcting
the statements.

PER0057 UNRECOVERABLE DEVICE ERROR DURING POSITIONING
◆ Warranty: Y

Meaning
An unrecoverable device error has occurred during
positioning the input tape.
The conversion step is aborted.

Response
Check the device or contact the operator.

PER0058 OPERAND '(&00)' ONLY ALLOWED FOR SYSTEM ADMINISTRATOR

Meaning
(&00): VSNNEW = operand of the VOLOUT statement
PEOT= operand value of the FORWARD operand of the POSIT
statement.

Response
Contact the system administrator.

```
PER0059    '(&00)' REACHED AFTER (&01) (&02) ON INPUT TAPE
```

Meaning
Beginning-of-tape or end-of-tape was reached premature on the input tape.
(&00): BEGINNING-OF-TAPE (BOT)
END-OF-TAPE (EOT)
DOUBLE-TAPE-MARK (DTM)
(&01): number of blocks/tapemarks
(&02): BOCKS
TAPE MARKS.

```
PER0060    ERROR IN ASSIGNMENT OF 'SYSDTA' TO DISK DRIVE, LINK='(&00)'
```

Meaning
(&00): link name.

Response
Check the state of the disk drive.

```
PER0061    ERROR IN ASSIGNMENT OF 'SYSDTA' TO CARD READER, LINK='(&00)'
```

Meaning
(&00): link name.

Response
Check the state of the card reader.

```
PER0062    OUTPUT INTO REMAIN FILE NOT ALLOWED, LINK='(&00)'
```

Meaning
A link name used in the statement START-TAPE-PROCESSING is the link name of a remain file.
(&00): link name.

Response
Check the link names used.

PER0063 `(&00) ERROR FOR (&01) (&02), LINK='(&03)'. REPLY (I=IGNORE; S=SKIP; H=HALT)`

Meaning
A parity or length error has occurred in the input file with link name
(&03) at record/block number (&02).
(&00): PARITY
WLRERR
(&01): BLOCK
RECORD
(&02): block/record counter
(&03): link name.

Response
Possible responses:
- I: ignore error or
- S: skip invalid block/record or
- H: abort conversion step.

PER0064 `RECORD LENGTH FIELD DESTROYED, LINK='(&00)'`

Meaning
When forming a variable output record by means of the SET-RECORD-MAPPING (RECORD) statement the record length field has been overwritten.
(&00): link name.

Response
Use start position 5 for the output record or work with fixed-length records.

PER0065 `WARNING: GROUP BREAK USER LINE INCOMPLETE, LINK='(&00)'`

Meaning
A field of the group break line exceeds the line length produced by LINE-SIZE (LLENGTH), COLUMN-SIZE (LGROUP) and OUTPUT-FORMAT (MODE) of the SET-PAGE-LAYOUT (FORMAT) statement, or it is outside the input record.
The given field is not transferred. The message is only output for the first line that hits the above condition.
(&00): link name.

Response
- Short group break line or
- enlarge formated lines or
- use another part of the input record.

`PER0066    CODE TRANSLATION TABLE IS MISSING OR INVALID`

Meaning
In the CODE-TRANSLATION (ALTER) operand of the SET-RECORD-MAPPING (RECORD)
statement a translation table file has been specified that either
is not available or cannot be read.

Response
Check the translation table file.

`PER0067    ILLEGAL 'DEVICE' OPERAND IN 'EDIT'/'POSIT' STATEMENT`

Meaning
When a VOLOUT statement is used during tape editing it must contain the operand DEVICE=PRINTER or DISPLAY.

`PER0068    ILLEGAL OPERATION IN USER ROUTINE, LINK='(&00)'`

Meaning
If OPEN=UPDATE is specified in the FILIN statement or /ADD-FILE-LINK
(FILE)-command it is not possible to modify the record read
in a user module or to insert an additional record.
(&00): link name.

Response
Check the user routine and repeat the run.

`PER0069    ERROR DURING TAPE DUPLICATION, LINK='(&00)'`

◆ Warranty: Y

Meaning
Because of an error no complete copy of the input tape(s) could be produced.
(&00): link name.

Response
Retry duplication with a different output tape or with several
output tapes or wait until another device becomes free.

PER0070 `PERCON INTERRUPTED. REPLY (DISP = DISPLAY STATE; CONT = CONTINUE; START = NEXT CONV.; FIN = NEXT INPUT FILE; TERM = ABORT)`

Meaning
PERCON was interrupted by the break key and the /INTR respectively /SEND-MESSAGE command.
A reaction is requested.

Response
Possible responses:
- DISP:show current processing state or
- CONT:continue processing or
- START: terminate the conversion step or
- FIN:stop reading from actual input file or
- TERM:terminate PERCON run abnormally.

PER0071 `WARNING: ILLEGAL SUM FIELD FORMAT, FORMAT 'Z' ASSUMED`

Meaning
In the SUM operand of the GROUP statement a sum field has been specified whose input format is not equal to 'P'/'Z' or whose output format is not equal to 'Z'/'+Z'/'-Z'.

PER0072 `DUPLICATE OPERAND IN '(&00)' STATEMENT`

Meaning
PERCON reports that an operand in the (&00) statement has been specified more than once.
(&00): invalid statement.

Response
In interactive mode correct the invalid statement,
in batch mode repeat the conversion step after correcting the statement.

PER0073 `STATEMENTS WITH ISO-CCSN '(&00)' NOT SUPPORTED`

Meaning
Statements with ISO-CCSN are not supported.
(&00): statement CCSN.

Response
Correct the statement CCS.

PER0074 `FILE WITH ISO-CCSN '(&00)' NOT SUPPORTED, LINK='(&01)'`

Meaning
Files with ISO-CCSN are not supported.
(&00): CCSN of file
(&01): link name.

Response
Correct the file CCS.

PER0075 `STATEMENT CCSN '(&00)' UNKNOWN`

Meaning
Statement CCSN is unknown.
(&00): statement CCSN.

Response
Correct the statement CCS.

PER0076 `FILE CCSN '(&00)' UNKNOWN, LINK='(&01)'`

Meaning
File CCSN is unknown.
(&00): CCSN of file
(&01): link name.

Response
Correct the file CCS.

PER0077 `INSTALLATION ERROR OF STATEMENT CCS '(&00)'`

Meaning
Statement CCS was faultily installed.
(&00): statement CCSN.

Response
Contact the system administrator.

PER0078 `INSTALLATION ERROR OF FILE CCS '(&00)', LINK='(&01)'`

Meaning
File CCS was faultily installed.
(&00): CCSN of file
(&01): link name.

Response
Contact the system administrator.

```
PER0079     RECORD LENGTH SPECIFIKATION TOO SMALL, LINK='(&00)'
```

Meaning
For a variable input/output file a record length of less than 5
was specified or for an ISAM file the key does not lie completely
inside the record.
(&00): link name.

Response
Increase the record length by means of /ADD-FILE-LINK (FILE) command
or FILIN/FILOUT statement.

```
PER0080     DEVICE TYPE UNKNOWN
```

Meaning
The requested device type is not available in this BS2000-version.

Response
Consult the system administrator what device types have been
generated for this system.

```
PER0081     NO STANDARD LABELS ON VOLUME '(&00)' FOR LINK='(&01)'
```

Meaning
A continuation tape of an input MF/MV set does not contain standard
labels, in contrast to the initial input tape.
no standard labels.
(&00): VSN
(&01): link name.

Response
Check whether the continuation tape really belongs to the MF/MV set.

```
PER0082     WRONG FILE CONTINUATION ON VOLUME '(&00)', LINK='(&01)'
```

Meaning
The last file section from the previous tape is not continued in
the first file section of the continuation tape.
(&00): VSN
(&01): link name.

Response
Check whether the specified input tapes really form an MF/MV set
sequence.

PER0083 `ONLY OWNER OR SYSTEM ADMINISTRATOR MAY USE VOLUME '(&00)', LINK='(&01)'`

Meaning
The access indicator in the VOL1 label does not allow the user to process this input tape.
(&00): VSN
(&01): link name.

Response
Ask the tape owner or system administrator to process the tape.

PER0084 `NOT ENOUGH OUTPUT TAPES SPECIFIED, *SCRAT IS REQUESTED, LINK='(&00)'`

Meaning
The specified output tape set was too small for a complete copy.
The operator is requested to mount a scratch tape.
Duplication will be continued on this tape.
(&00): link name.

PER0085 `ISAM KEY FIELD DESTROYED, LINK='(&00)'`

Meaning
When forming an output record by means of the statement SET-RECORD-MAPPING with FILLER=OUTPUT or by means of the user interface for output the ISAM key field has been overwritten.
The conversion step is aborted.
(&00): link name.

Response
Correct the statement and repeat the conversion step.

PER0086 `WARNING: BYTE COUNTER OVERFLOW AT RECORD (&00), LINK='(&01)'`

Meaning
The internal byte counter has reached its maximum value of 2**32 -1 and cannot be incremented. However, processing of the conversion step continues.
(&00): record counter
(&01): link name.

Response
Check wether file processing is error-free after the specified record counter.

PER0087 `CONTRADICTORY OPERANDS '(&00)' AND '(&01)' IN '(&03)' STATEMENT, LINK='(&02)'`

Meaning
(&00): operand 1
(&01): operand 2
(&02): link name
(&03): statement.

Response
Remove or correct one of the specified operands.

PER0088 `ILLEGAL 'DEVICE' OPERAND IN FILOUT STATEMENT, LINK='(&00)'`

Meaning
If a VOLIN statement has been specified the operand DEVICE=PRINTER or =DISPLAY is not allowed in the FILOUT statement.

Response
Issue a FILOUT statement with 'FILENAME=filename' in order to convert the tape into a file or issue a VOLOUT statement with 'DEVICE=PRINTER or =DISPLAY' in order to convert the tape to the printer or the screen.

PER0089 `DUPLICATION PROHIBITED DURING TAPE EDITING`

Meaning
Only editing or duplication is permitted in the same conversion step, not both.

PER0090 `'FORMAT' STATEMENT NOT ALLOWED DURING TAPE DUPLICATION`

Meaning
During duplication (copying of tapes) a FORMAT statement is not allowed.

Response
Remove the FORMAT statement.

PER0091 `ILLEGAL 'DEVICE=PRINTER OR DISPLAY' OPERAND DURING DUPLICATION`

Meaning
During duplication the operand DEVICE=PRINTER or =DISPLAY in the VOLOUT statement is not allowed.

PER0092 `WARNING: REPAIRING OF FILE '(&00)', LINK='(&01)'`

- Warranty: Y

Meaning
An error was detected when the file (&00) with link name (&01) was opened. A repair follows by means of macro VERIF in the OPENC exit.
(&00): file name
(&01): link name.

Response
Check if the output is complete.

PER0093 `REPAIRING OF FILE '(&00)', LINK='(&01)'. REPLY (Y=YES; N=NO)`

- Warranty: Y

Meaning
An error was detected when the file (&00) with link name (&01) was opened. It is possible to repair the file by means of macro VERIF in the OPENC exit.
(&00): file name
(&01): link name.

Response
Possible responses:
- Y: repair the file
- N: no repairing of the file.

PER0094 `FILE '(&00)' HAS TO BE REPAIRED, LINK='(&01)'`

- Warranty: Y

Meaning
An error was detected when the file (&00) with link name (&01) was opened. A repairing by means of macro VERIF is not done in the OPENC exit.
(&00): file name
(&01): link name.

Response
Repair the file and repeat the conversion step.

PER0095 `OBJECT MODULE PCNSR9 WITH LINK ERROR X'(&00)'. RUN IS ABORTED`

- Warranty: Y

Meaning
A LINK error occurred when dynamically loading object module PCNSR9.
(&00): error code of the LINK macro.

Response
Check the module library OMLPERCON.

PER0096 `FILE '(&02)' WITHOUT ALTERNATE KEY '(&00)', LINK='(&01)'`

Meaning
The alternate key specified in the statement SET-RECORD-MAPPING
was not found. The conversion step is aborted.
(&00): alternate key
(&01): link name
(&02): file name.

Response
Correct the statement and repeat the conversion step.

PER0097 `WARNING: FILE '(&01)' HAS NO ALTERNATE INDEX, LINK='(&00)'. PROCESSING CONTINUES`

Meaning
(&00): link name
(&01): file name.

PER0098 `ISAM KEY '(&01)' ALREADY EXISTS, LINK='(&00)'`

- Warranty: Y

Meaning
A record could not be inserted into the output file because
a record containing the key already exists and it was said
OVERWRITE=NO in the ASSIGN-OUTPUT-FILE statement.
(&00): link name
(&01): ISAM key.

PER0099 `DMS ERROR CODE X'(&01)' DURING DETERMINING ALTERNATE KEYS, LINK='(&00)'. FURTHER INFORMATION: /HELP-MSG DMS(&01)`

- Warranty: Y

Meaning
A DMS error was detected during determining alternate keys.
The conversion step is terminated.
(&00): link name
(&01): DMS error code.

PER0100 `DMS ERROR CODE X'(&01)' DURING CREATING ALTERNATE KEYS, LINK='(&00)'. FURTHER INFORMATION: /HELP-MSG DMS(&01)`

- Warranty: Y

Meaning
A DMS error was detected during creating alternate keys.
The conversion step is terminated.
(&00): link name
(&01): DMS error code.

PER0101 `ALTERNATE INDEX '(&01)' ALREADY DEFINED WITH OTHER ATTRIBUTES, LINK='(&00)'`

- Warranty: Y

Meaning
The alternate index to be created already exists. However, the attributes
defined for it are not the same as those defined in the input file
specified
in the statement ASSIGN-OUTPUT-FILE in the parameter ALTERNATE-INDEX.
The conversion step is terminated.
(&00): link name
(&01): alternate index.

PER0102 `DUPLICATE VALUES FOR ALTERNATE INDEX '(&01)', LINK='(&00)'`

- Warranty: Y

Meaning
Duplicate values of the alternate index exists in several records.
However, the alternate index has to be created with DUPKEY=NO.
The conversion step is terminated.
(&00): link name
(&01): alternate index.

PER0103 `RECORD TOO SMALL FOR ALTERNATE INDEX '(&01)', LINK='(&00)'`

- Warranty: Y

Meaning
When creating an alternate index, a record was read that is too
short to accommodate the alternate index.
The conversion step is terminated.
(&00): link name
(&01): alternate index.

PER0104 `*** PERCON VERSION '(&00)' NOT AVAILABLE. RUN IS ABORTED ***`

- Warranty: Y

Meaning
A PERCON version required by a start command or subprogram call is not installed. The run is aborted.
(&00): PERCON version required.

Response
Choose another version or have the required version installed in system.

PER0105 `SYNTACTICAL ERROR IN VERSION '(&00)'. RUN IS ABORTED`

- Warranty: Y

Meaning
The syntax of the version given as parameter of a PERCON subprogram call is incorrect.
Following syntax of version is allowed:
n.n /n.na /n.nann
nn.n / nn.na / nn.nann
(n ... a decimal character / a ... an alphabetic character)
(&00): wrong version.

Response
Check the version for subprogram call und repeat the run after correcting the version.

PER0106 `SUBSYSTEM '(&00)' NOT AVAILABLE AT THE MOMENT. RUN IS ABORTED`

- Warranty: Y

Meaning
(&00): name of subsystem.

Response
Repeat the PERCON run when subsystem will be available.

PER0107 `ERROR CODE X'(&00)' DURING MACRO '(&01)'. RUN IS ABORTED`

- Warranty: Y

Meaning
(&00): macro error code
(&01): name of the macro.

Response
Check macro error code and react accordingly.
Repeat the run.

PER0109 `WARNING: DMS ERROR X'(&01)'. OUTPUT FILE '(&02)' WITHOUT CCSN '(&03)', LINK='(&00)'. FURTHER INFORMATION: /HELP-MSG DMS(&01)`

◆ Warranty: Y

Meaning
The CCSN could not be written into the catalog entry of the output file. PERCON is continued.
(&00): link name
(&01): DMS error code
(&02): name of output file
(&03): CCSN.

Response
Use /MODIFY-FILE-ATTRIBUTES command for putting the CCSN in the catalog entry.

PER0110 `BS2000-VERSION LOWER THAN VERSION '(&00)'. RUN IS ABORTED`

◆ Warranty: Y

Meaning
PERCON needs at least the BS2000 version (&00).

PER0111 `OBJECT MODULE PCNSR9 IN LIBRARY '(&01)' WITH BIND ERROR X'(&00)'. RUN IS ABORTED`

◆ Warranty: Y

Meaning
A BIND error occurred when dynamically loading object module PCNSR9.
(&00): error code of the BIND macro
(&01): module library used.

Response
Check the module library used.

PER0112 `STATEMENTS WITH UNICODE CCSN '(&00)' NOT SUPPORTED`

Meaning
Statements with a Unicode CCSN are not supported.
(&00): statement CCSN.

Response
Correct the statement CCS.

PER0113 `WARNING: STRUCTURE OPERAND 'OUTPUT-LENGTH' TOO SMALL FOR UNICODE AT RECORD (&00), LINK='(&01)'`

- Warranty: Y

Meaning
In the OUTPUT-FIELDS operand of the SET-RECORD-MAPPING statement a field is used, whose structure operand OUTPUT-LENGTH is insufficient for output and format conversion.
Fields will be cut off on the right side. The message is only output for the first record that hits the above condition.
(&00): record counter
(&01): link name

Response
In interactive mode correct the invalid operand, in batch mode repeat the conversion step after correction.

PER0114 `FILE WITH UNICODE-CCSN NOT SUPPORTED FOR '(&00)', LINK='(&01)'`

Meaning
Files with an Unicode-CCSN are not supported for SYSLST, SYSOUT and SYSDTA.
(&00): SYSLST, SYSOUT, SYSDTA
(&01): link name.

Response
Correct the file CCS.

PER0115 `FILE WITH UNICODE CCSN NOT SUPPORTED FOR '(&00)', LINK='(&01)'`

Meaning
Files with a Unicode CCSN are not supported for ISAM, BTAM and PAM:
(&00): ISAM, BTAM, PAM
(&01): link name.

Response
Correct the file CCS.

PER0116 `RECORD LENGTH ODD FOR UTF16, LINK='(&00)'.`

Meaning
Record length must be even for UTF16.
(&00): link name

Response
Correct record length.

`PER0117` `WARNING: AT LEAST ONE CHARCTER HAS NO EQUIVALENT AT RECORD (&00), LINK='(&01)'`

- Warranty: Y

Meaning
At least one character of the input record does not have an equivalent in the output record. Default character is taken. The message is only output for the first record that hits the above condition.
(&00): record counter
(&01): link name

`PER0118` `WARNING: OUTPUT CCS PARTIAL COMPATIBLE TO INPUT CCS, LINK='(&00)'`

- Warranty: Y

Meaning
The output CCS is only partial compatible to the input CCS.
Some characters of the input file might be possibly replaced in the output file by X'00'.
(&00): link name of the output file.

Response
Use another output CCS.

Related publications

Ordering manuals

The manuals are available as online manuals, see *http://manuals.fujitsu-siemens.com*, or in printed form which must be paid and ordered separately at *http://FSC-manualshop.com*.

[1] **BS2000/OSD-BC**
Introductory Guide to DMS
User Guide

[2] **XHCS** (BS2000/OSD)
8-Bit Code Processing in BS2000/OSD
User Guide

[3] **BS2000/OSD-BC**
Introductory Guide to Systems Support
User Guide

[4] **SORT** (BS2000/OSD)
User Guide

[5] **BINDER** (BS2000/OSD)
User Guide

[6] **SDF** (BS2000/OSD)
Introductory Guide to the SDF Dialog Interface
User Guide

[7] **JV** (BS2000/OSD)
Job Variables
User Guide

[8] **BS2000/OSD-BC**
Executive Macros
User Guide

[9] **BS2000/OSD-BC**
Commands (several volumes)
User Guides

[10] **ARCHIVE** (BS2000/OSD)
User Guide

[11] **BS2000/OSD-BC**
DMS Macros
User Guide

[12] **BLSSERV**
Dynamic Binder Loader / Starter in BS2000/OSD
User Guide

[13] **SDF-P** (BS2000/OSD)
Programming in the Command Language
User Guide

[14] **Unicode in BS2000/OSD**
Introduction

Index